Cohort Creation within the OMOP Common Data Model

James Lazarus

31 March, 2021

Contents

1 Introduction

Cohort creation in general is the process of retrieving particular persons of interest from a data base in order to carry out a study. In this case, this tutorial is concerning cohort creation within the OMOP Common Data Model (CDM), specifically for carrying out medical research. The goal of this tutorial is to outline the cohort creation process as well as to show how analysis is carried out. Note that this document only concerns cohort creation in relation to Connected Yorkshire.

This tutorial will explain the OMOP CDM, an overall guide to Athena, retrieving the data necessary from the OMOP CDM (with an R library), Cohort examples such as an Asthma cohort as well as analysis, and finally an archive of SQL scripts as well as links to any resources of interest.

2 General Information on the OMOP Common Data Model

The OMOP CDM is a standardised medical data base that stores observational medical data. Where observational medical data provides a view of the patient while receiving health care. The CDM makes use of a standardised vocabulary containing all the necessary and appropriate corresponding standard healthcare concepts. An overview of the OMOP CDM can be seen in figure 1 below (note that not all relationships between tables are shown):

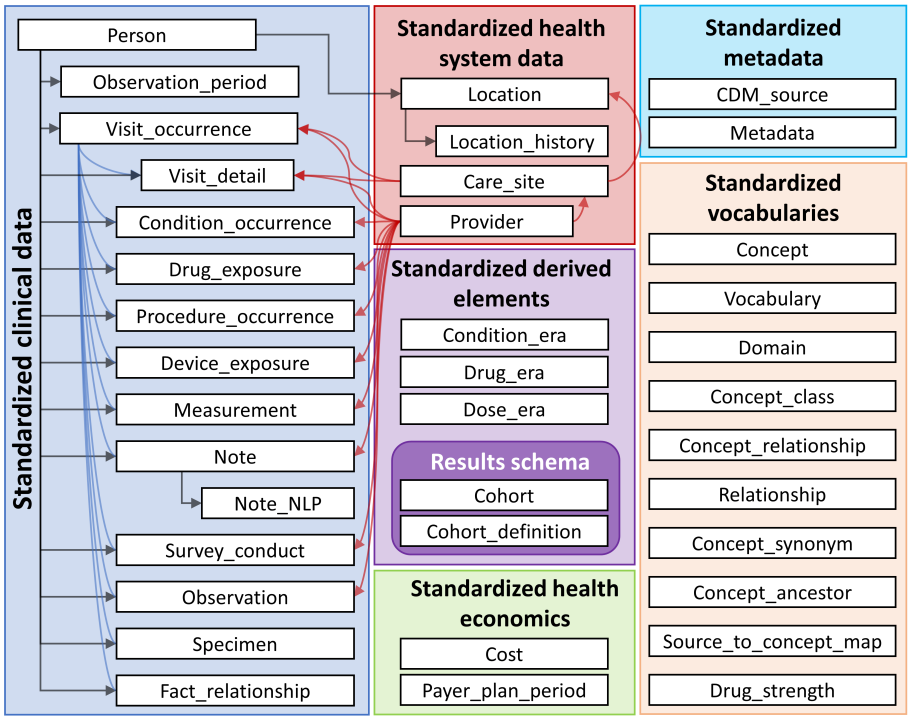


Figure 1: Overview of all tables in the CDM version 6.0 (Ohdsi)

The CDM is a "person-centric" model, meaning that all clinical event tables are linked to the "person table". This is outlined in Figure 1. With the start date, this allows a descriptive time line of all health care events by person (Ohdsi).

2.1 Model Conventions

It is important to note that the CDM is platform-independent, where data types are defined generically using ANSI SQL data types (VARCHAR, INTEGER, FLOAT, DATE, DATETIME, CLOB). The CDM does not use any date or datetime format. Thus, standard queries against CDM may vary within R (Ohdsi). Note that when querying in R, certain date and time columns do not import correctly because of this. Therefore it is recommended to query it as a string, and then convert it to a date. The functions outlined in this document all import date columns as strings.

Events of different nature are recorded within the CDM, and are thus organised into different domains. These events are thus stored in domain specific tables. Therefore, each standard concept has a correlating domain. Ensuring an unambiguous location for any code or concept. For example, signs, symptoms and diagnosis concepts are all parts of the Condition Domain, and are recorded in the CONDITION_CONCEPT_ID of the CONDITION_OCCURRENCE table. All together there are 30 domains (Ohdsi).

Many of the tables in the CDM have equivalent information in multiple places. For example, Source Value, Source Concept and Standard Concepts. This can get confusing, below lays out the differences between these values:

- **Source Vales:** The original representation of an event record in the source data. The codes can be from widely used coding systems (e.g. ICD9CM) or controlled vocabularies from the original data (e.g. F and M for female and male) (Ohdsi).
- **Concepts:** CDM specific entities that normalise clinical facts, and have unique ids across all domains(Ohdsi).
- **Source Concepts:** Represent the code used in the source, and are stored in [Event]_SOURCE_CONCEPT_ID field in data tables (Ohdsi).
- **Standard Concepts:** Define the meaning of a clinical entity uniquely across all databases. Have an equivalent non-standard concept in the standardised vocabularies. Stored in [Event]_CONCEPT_ID field of data tables (Ohdsi).

2.1.1 CDM standardised tables with Example

The CDM contains a lot of information, stored in many different tables. To show where information is stored, we will run through an example of a patient receiving treatment. In this case the patient is called Lauren.

- **Person Table:** This table contains the standard information that we know about Lauren. Information such as her age; sex; birthday, race and nationality. More on this later.
- **Observation Period Table:** Contains records of the amount of time clinical events happened to Lauren. These clinical events include; demographics; conditions diagnosed; procedures given; and drugs taken. The first record in the system is considered the start date of the observation, while the last is considered the end date of observation.
- **Visit Occurance:** Contains the encounters Lauren has had with the health care system, denoting the circumstances under which Lauren has received health care. The most common visit recordings are inpatient, outpatient, emergency department and non-medical institution visits.
- **Condition Occurance:** Records the diagnoses, signs, or symptoms of Lauren. For example, if the symptom shown is dysmenorrhea (painful menstrual cramps), this will be recorded in this table.
- **Drug Exposure:** Captures the drugs that Laura takes, such as prescription, over-the-counter medicines, and vaccines.
- **Procedure Occurance:** Captures activities or processes ordered or carried out on Lauren with a diagnostic or therapeutic purpose. For example, if an ultrasound was carried out on Lauren, this would be recorded here.

2.2 Stardised Vocabulary

As discussed in Model conventions, the standardised vocabulary is an integral part of the CDM, allowing for standardisation of methods, definitions and results by defining the content of the data.

All clinical events within the OMOP CDM are expressed as concepts. They are the fundamental building blocks of the data records, making almost all tables fully normalized with few exceptions. Concepts are stored in the concept table, as shown in the table 1:

Column	Explanation	Example
CONCEPT_ID	Primary Key	313217
CONCEPT_NAME	Description	Atrial Fibrillation
DOMAIN_ID	Domain	Condition
VOCABULARY_ID	Vocabulary	SNOMED
CONCEPT_CLASS_ID	Class in Vocabulary	Clinical Finding
STANDARD_CONCEPT	Standard, Source of Classification	S
CONCEPT_CODE	Code in Vocabulary	49436004
VALID_START_DATE	Valid during time interval	01-Jan-1970
VALID_END_DATE	Valid during time interval	31-Dec-2099
INVALID_REASON	Valid during time interval	

Table 1: Concept Table OMOP CDM

As stated in the model conventions section, each concept is assigned a domain, in the DOMAIN_ID field, which is an alphanumeric ID. Examples of such domain identifiers are:

- Condition
- Drug
- Procedure
- Visit
- Device
- Specimen
- etc.

Knowing the domain of a concept is important when trying to identify the correct concept id.

Concepts within the data base can either be recorded as stardard, non-standard or a classification concept.

Standard concepts are indicated using an "S" in the standard concept field, and are used to record data in the CDM fields ending in "_CONCEP_ID". Non-Standard Concepts are not used to represent clinical events, but are often found in source data. Thus, they are also called "Source Concepts", and are mapped to standard concepts. A good example of Standard vs Non-Standard concepts is defining the "Atrial fibrillation" in the condition domain. MESH code D001281, CIEL code 148203, SNOMED code 49436004, ICD9CM code 427.31 and Read code G573000 all define "Atrial fibrillation"; however only the SNOMED concept is Standard and therefore is the only code that represents the condition in the data.

Classification concepts are not standard either. However, they participate in the hierarchy, and can be used to perform hierarchical queries. A representation of this hierarchy can be seen in the figure 2:

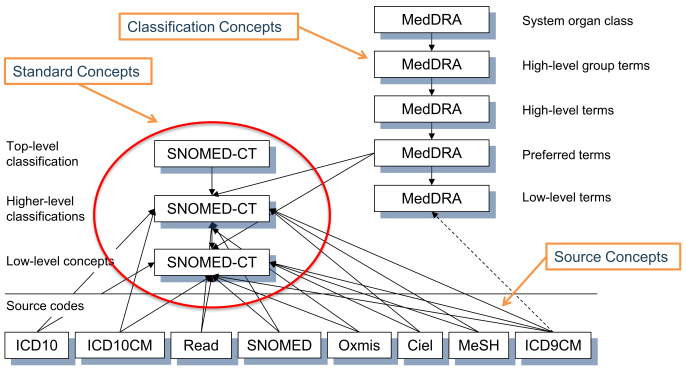


Figure 2: Hierarchical relationships in the condition domain (Ohdsi)

This is especially useful when querying for drugs within the OMOP CDM, as a clinician will often want to find drugs by category, and sometimes conditions. Thus, for the Condition domain and the Drug domain there are both standard concepts, and classification concepts that connect via a hierarchy. This can be seen in table 2.

Domain	**Standard Concept**	**Classifcation Concept**
Condition	SNOMED, ICDO3	MedDRA
Drug	RxNorm, RxNorm Extension, CVX	ATC

Table 2: Concept Table OMOP CDM

A good example in practice would be with the Salbutamol. The figure 3 shows an example for a classification concept:

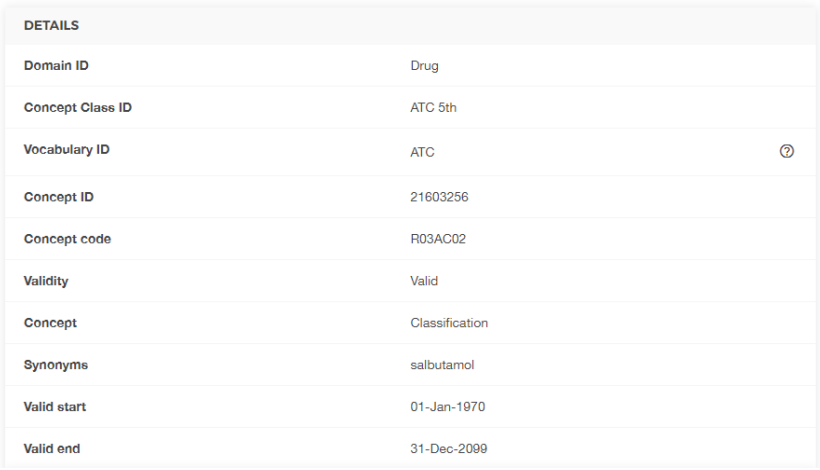
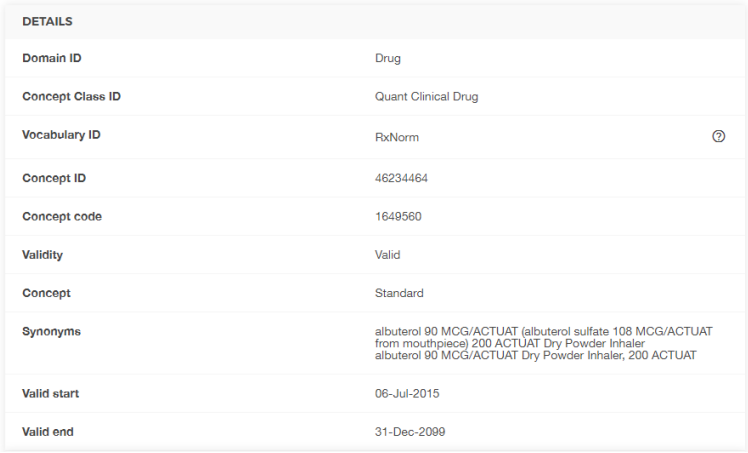


Figure 3: Aalbutamol; inhalant

While Figure 4 shows a Standard Concept, where Salbutamol is an ancestor to Albuterol:



DETAILS	
Domain ID	Drug
Concept Class ID	Quant Clinical Drug
Vocabulary ID	RxNorm
Concept ID	46234464
Concept code	1649560
Validity	Valid
Concept	Standard
Synonyms	albuterol 90 MCG/ACTUAT (albuterol sulfate 108 MCG/ACTUAT from mouthpiece) 200 ACTUAT Dry Powder Inhaler albuterol 90 MCG/ACTUAT Dry Powder Inhaler, 200 ACTUAT
Valid start	06-Jul-2015
Valid end	31-Dec-2099

Figure 4: 200 ACTUAT albuterol 0.09 MG/ACTUAT Dry Powder Inhaler

This relationship can be found in the concept_ancestor table in the CDM. The corresponding row can be seen in table 3:

Column	Value
ancestor_concept_id	21603256
descendant_concept_id	46234464
min_levels_of_separation	2
max_levels_of_separation	3

Table 3: Concept Ancestor Table example

Table 3 shows the hierarchical properties of the OMOP CDM, going so far as to define the levels of separation. Within a domain, standard and classification concepts are organized in a hierarchical structure and are stored within the concept_ancestor table. This allows for querying and retrieving a concepts hierarchical descendants. An in depth example of the Atrial fibrillation condition hierarchy can be seen in figure 5 below:

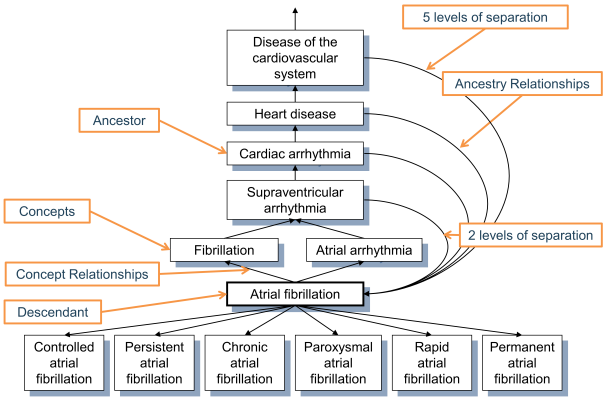


Figure 5: Hierarchy of the condition “Atrial fibrillation”

3 Constructing Cohorts

To demonstrate how to construct cohorts, a step by step guide will be followed through along with cohort examples, as well as further information such as how to use the Google Cloud Platform, so as to see these steps in practice. The two steps that outline the cohort creation process are:

- Defining and converting cohort into OMOP CDM concepts.
- Querying the CDM to retrieve cohort.

Note that all the following code has either been done in R or SQL, and has been executed in RStudio.

3.1 The Google Cloud platform

The OMOP CDM is held on the Google Cloud platform (GCP) and can be queried via BigQuery on the GCP. In order to access the GCP please use the link outlined below and log in.

https://console.cloud.google.com/bigquery

Figure 6 outlines the GCP:

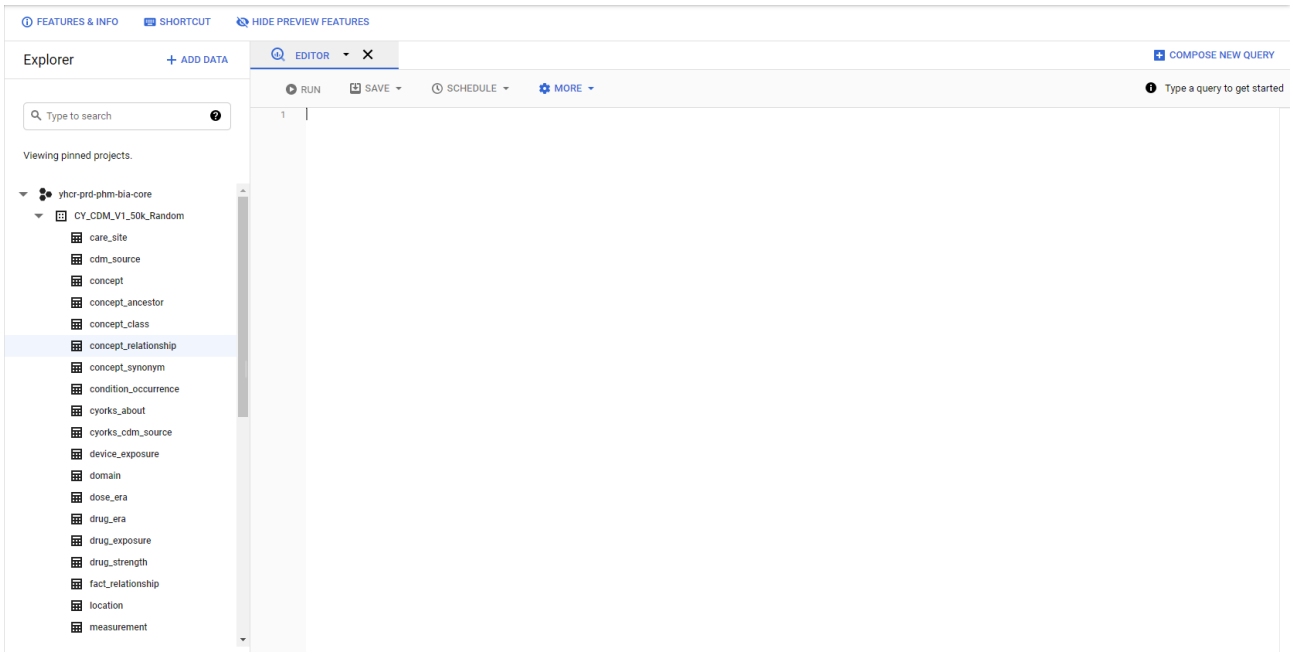


Figure 6: Big Query on the GCP

Via the explorer we can see the project, in which the OMOP CDM is located (CY_CDM_V1_50k_Random), under which is a list of all the OMOP tables. Next to this is the editor where SQL queries can be written and run on the CDM, and a table can be returned within the GCP.

This is particularly useful for when you would want to write your own SQL queries and test them out before running them in R (to run a query simply click the run button), as queries run faster on the GCP and it allows for quick debugging of queries. Figure 7 gives a query example:

Figure 7: Query Results (data has been cut out to preserve privacy)

Once the query has been run results are shown in the "Query Results" section, as seen in figure 7. The results of the query can be found in the results tab. While job information and execution details provide further information about the query such as the duration. The JSON tab shows the data in JSON format.

3.2 Defining and converting cohorts into OMOP CDM concepts.

When creating a cohort, to start with a definition of the cohort will be outlined by a clinician. A simple example of this can be seen below:

Patients suffering from Asthma and being perscribed a steroid and preventative inhaler

From this summary we can convert this cohort into specific parts and define the terms accordingly that we can query within the OMOP CDM. Note that the clinician may outline these concepts for you, but not always. Here the specific drugs were not identified, in which case it is important to ask a clinician for the specific drug names. Here is it beclometasone and salbutamol for the steroid and preventative inhalers respectively.

To begin with there are 3 concepts that are defined within this cohort definition:

- Asthma - Condition domain
- Prescription of beclometasone - Drug domain
- Prescription of salbutamol - Drug domain

Now that we have identified the 3 concepts needed to define this cohort we can begin with finding their condition concept ids, as this is the most accurate way to filter the CDM. To do this we need to use the Athena website which records the standardised vocabularies, link below:

https://athena.ohdsi.org/

Here, we can search for the appropriate terms and filter the domain and concept to exactly what is needed to find the correct id. Figure 8 shows searching for the Asthma condition in Athena:

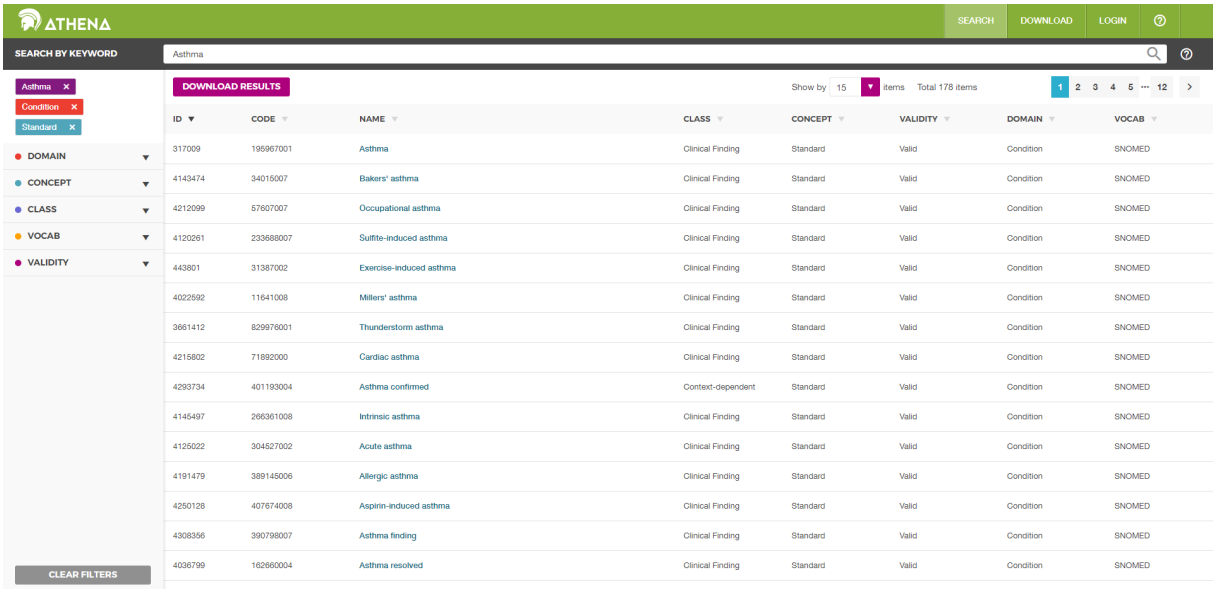


Figure 8: Asthma Condition Search

Note than filtering the domain to Conditions and the concept to standard means it is less likely for the Asthma concept found to be erroneous and not retrieve the data we need. Selecting the first option in the Athena search allows us to view the concept in more detail. Here we can find the Hierarchy, The related concepts, and details about the concept in Figure 9:

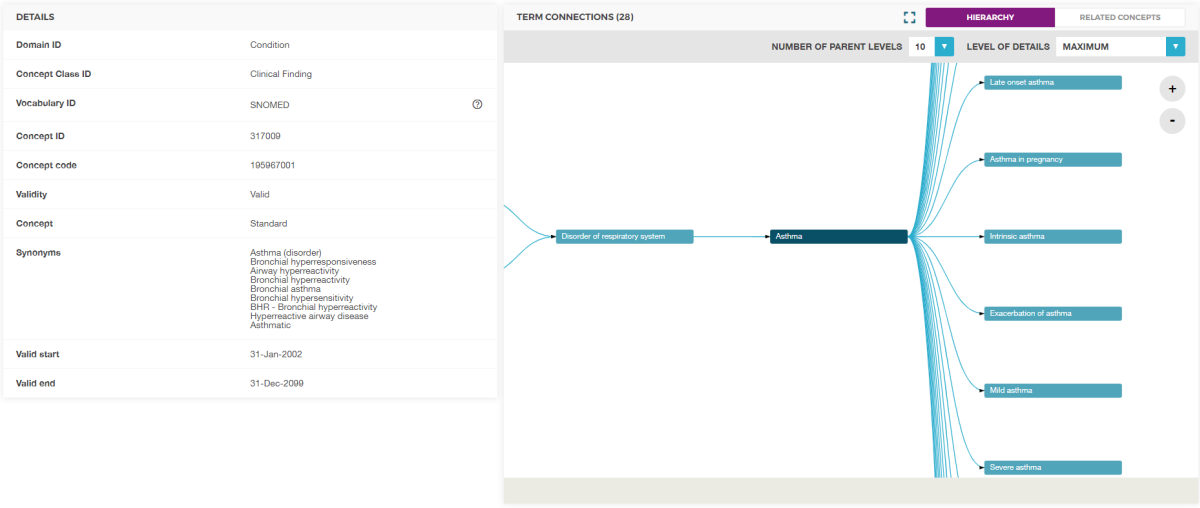


Figure 9: Asthma Concept

Looking at the hierarchy, it can be seen that this is the correct definition for Asthma as it subsumes no other definitions of Asthma and a lot of different types of Asthma subsume it. Next the same needs to be done with the prescriptions beclometasone and salbutamol, this can be seen in Figure 10 and 11.

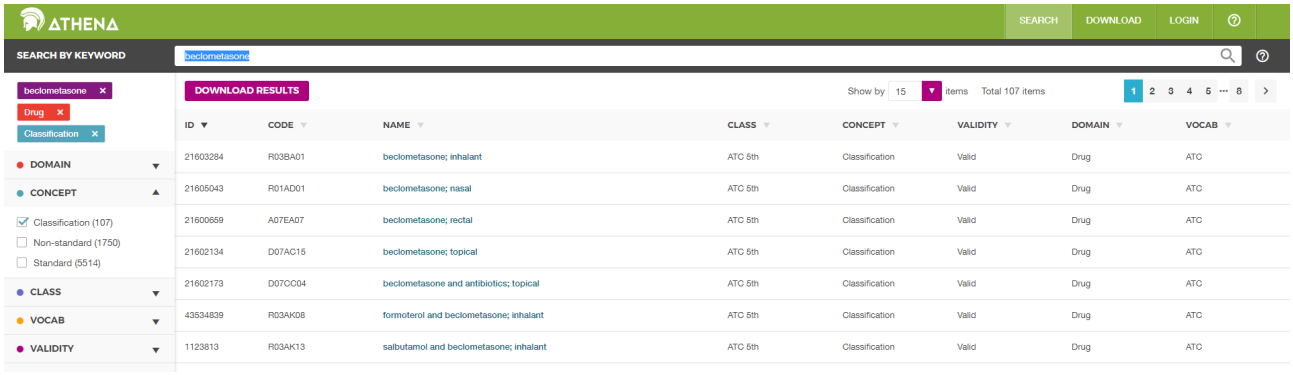


Figure 10: Beclometasone Drug Search

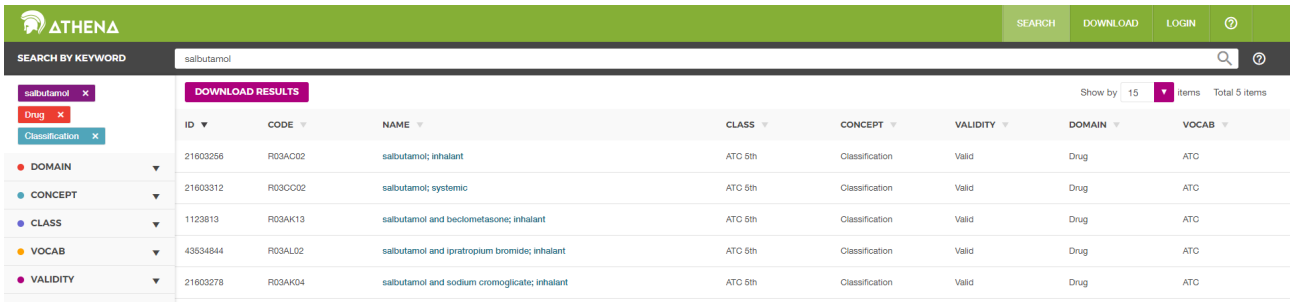


Figure 11: Salbutamol Drug Search

Here we set the domain to Drug and concept to classification when searching for the correct drug concept. A classification concept is used because there are a lot of drugs that are used to treat a variety of conditions, thus a cohort will often be defined using a classification rather than a standard concept, as a standard concept will be too specific.

The first row of both searches for Beclometasone and Salbutamol respectively are the correct searches, as we are specifically looking for an inhalant. Thus the Concept ids have been found for this Asthma cohort, as shown below:

- Asthma - 317009
- Beclometasone; inhalant - 21603284
- Salbutamol; inhalant - 21603256

These ids can now be used to find the cohort specifically. The next step in the Cohort creation process is to use these IDs to retrieve the cohorts.

3.3 Retrieving Cohorts from the OMOP CDM

Now that the appropriate ids have been found, we can load the cohort into R for analysis. There are 2 ways of doing this, the first way is using the CohortCreationLibrary functions, the other is to use standard SQL Queries. Both methods will be outlined in this section.

3.3.1 CohortCreationLibrary

To begin with we need to connect to bigquery through R, this will bring up a popout and you will be asked to log in. This done using the following code:

```r
library(bigrquery)
bq_projects()
project_id <- "yhcr-prd-phm-bia-core" # replace this with your project ID
bq_project_datasets(project_id) # displays data-bases available
```

Now that a connection has been established to the GCP, we can begin with installing the CohortCreationLibrary:

```r
library(devtools)
install_github("jlazaruss/CohortCreationLibrary")
library(CohortCreation)
```

This library contains 3 functions to help with querying the OMOP CDM. 2 of which are used for retrieving a specific table. These 2 are LoadCohortSubjects, and LoadPrescription. Both retrieve a variety of information useful to analyse cohorts. Documentation for the functions can be found here:

https://github.com/jlazaruss/CohortCreationLibrary

The LoadCohortSubjects function returns information based on the condition filtered and returns them in between a start date and an end date. It joins the condition_occurrence, person, concept, and visit_occurrence tables to retrieve the information about the person and the condition diagnosed. Table 4 shows each column returned along with its description:

Column	Description
condition_occurance_id	Primary key for condition occurrence
person_id	Person id of patient suffering from the condition
condition_concept_id	Value of condition
concept_name	Name of the condition
condition_start_date	Start date of the condition
condition_end_date	End date of the condition
condition_type_concept_id	The provenance of the Condition record
condition_status_concept_id	The point during the visit the diagnosis was given
visit_occurence_id	The visit id during which the condition occurred
condition_source_value	Provenance of the Condition record
Birthday	Birthday of patient
death_datetime	Day of death of patient
gender_source_value	Gender of patient
race_source_value	Race of gender
visit_start_datetime	Start date of visit to medical facility
visit_end_datetime	End date of visit to medical facility
visit_concept_id	Id of type of visit, e.g. an Inpatient visit
visit_type_concept_id	The provenance of the visit record
care_site_id	Id of where the visit took place
visit_source_concept_id	Source concept id of visit

Table 4: LoadCohortSubjects Table

Similarly the LoadPrescription function returns information based on the prescription filtered or a list of person ids and returns all the people who were given that prescription filtered by, or all the people in the person ids list. There is also an option depending whether you want to filter by the 4th or 5th level of the ATC drug concept, as a broader concept classification may be wanted (e.g. Glucocorticoids vs Salmeterol). It joins the drug_occurrence, concept, and concept_ancestor tables to retrieve the information about prescriptions. Table 5 shows each column returned along with its description:

Column	Description
drug_exposure_id	Primary key for drug exposure
person_id	Person id of patient taking drug
drug_concept_id	Value of drug
concept_name	Name of the drug
drug_exposure_start_date	Start date of the drug exposure
drug_exposure_end_date	End date of the condition
drug_type_concept_id	To delineate between prescription
sig	The instruction for the drug as written by the provider
visit_occurence_id	The visit id during which the condition occurred
does_unit_source_value	The dose unit of the drug given
ancestor_CONCEPT_ID	The classifying drug concept id
ancestor_concept_name	The classifying drug concept name
vocabulary_id	Vocabulary primary key
concept_class_id	Class of concept
standard_concept	Flag to determine where a Concept is a Standard
concept_code	The identifier of the Concept in the source vocabulary
valid_end_date	The date when the Concept became invalid

Table 5: LoadPrescription Table

Returning to the Asthma cohort we can thus use these two functions to build this cohort, using the concepts outlined in the previous sections. First we find all the patients suffering from Asthma, and then we can find all patients prescribed preventative or steroid inhalers. This is shown in the code below:

```
project_name_asthma <- "yhcr-prd-phm-bia-core"
data_base_name_asthma <- "CY_CDM_V1_50k_Random"

#Asthma - 317009
ASTHMA_PATIENTS <- LoadCohortSubjects(project_name_asthma, data_base_name_asthma,
                                      condition_concept = 317009, start_date = NULL,
                                      end_date = NULL)

#beclometasone; inhalant - 21603284 - Steroid
#salbutamol; inhalant - 21603256 - preventer
STEROID_PERSCRIPTIONS <- LoadPrescription(project_name_asthma, data_base_name_asthma,
                                          prescription = 21603284,  ATC_4th = FALSE)
PREVENTER_PERSCRIPTIONS <- LoadPrescription(project_name_asthma, data_base_name_asthma,
                                            prescription = 21603256,  ATC_4th = FALSE)
```

Then we can use the patient_id column to filter the prescriptions to return only the Asthma patients:

```
PREVENTER_PERSCRIPTIONS <- PREVENTER_PERSCRIPTIONS[PREVENTER_PERSCRIPTIONS$person_id
                                                   %in% ASTHMA_PATIENTS$person_id ,]
STEROID_PERSCRIPTIONS <- STEROID_PERSCRIPTIONS[STEROID_PERSCRIPTIONS$person_id
                                               %in% ASTHMA_PATIENTS$person_id ,]
```

Thus all the patients who suffer with Asthma, as well as the ones who have either a steroid or preventer inhaler have been found.

To show how these functions work with different cohorts, lets define some different cohorts along with their concepts to show how these functions are used in different situations:

- Patients suffering from type ii diabetes
- Patients with pre-diabetes

- Childhood Autism
- Whole Cohort
- Patients with COPD, with an exacerbation in the last 12 months.

Patients suffering from type ii diabetes, pre-diabetes and childhood autism are simple cohorts to make. However, instead of filtering the prescriptions with a concept id. We have to filter using the list of person_ids. As a prescription isn't given. The code for these two cohorts is as follows:

```
#Diabetes type ii Cohort
project_name_diabetes <- "yhcr-prd-phm-bia-core"
data_base_name_diabetes <- "CY_CDM_V1_50k_Random"

#Diabetes type ii - 201826
DIABETES_PATIENTS <- LoadCohortSubjects(project_name_diabetes,
                                        data_base_name_diabetes,
                                        condition_concept = 201826,
                                        start_date = NULL, end_date = NULL)


patients_diabetes <- unique(DIABETES_PATIENTS$person_id)
DIABETES_PERSCRIPTIONS <- LoadPrescription(project_name_diabetes,
                                           data_base_name_diabetes,
                                           person = patients_diabetes)
```

```
#Pre-diabetes Cohort
project_name_prediabetes <- "yhcr-prd-phm-bia-core"
data_base_name_prediabetes <- "CY_CDM_V1_50k_Random"

#Pre-Diabetes - 44808385
DIABETES_PATIENTS <- LoadCohortSubjects(project_name_prediabetes,
                                        data_base_name_prediabetes,
                                        condition_concept = 44808385,
                                        start_date = NULL, end_date = NULL)


patients_prediabetes <- unique(DIABETES_PATIENTS$person_id)
DIABETES_PERSCRIPTIONS <- LoadPrescription(project_name_prediabetes,
                                           data_base_name_prediabetes,
                                           person = patients_prediabetes)
```

```
#Childhood Autism Cohort
project_name_autism <- "yhcr-prd-phm-bia-core"
data_base_name_autism <- "CY_CDM_V1_50k_Random"

#Autistic disorder of childhood onset - 434902
AUTISM_PATIENTS <- LoadCohortSubjects(project_name_autism,
                                      data_base_name_autism,
                                      condition_concept = 434902,
                                      start_date = NULL, end_date = NULL)

patients_autism <- unique(AUTISM_PATIENTS$person_id)
AUTISM_PERSCRIPTIONS <- LoadPrescription(project_name_autism,
                                         data_base_name_autism,
                                         person = patients_autism)
```

Note where possible this should be avoided, as the query can take a long time to run, it is much more efficient to query the data with a prescription rather than with the patient ids.

It may be necessary sometimes to load the whole cohort in to see what data is available, or just to do a diagnostic report of the entirety of the data. To do this all that is needed is to leave all the options blank and all the data available will be loaded in. The code for this is as follows:

```
#Whole Cohort
project_name <- "yhcr-prd-phm-bia-core"
data_base_name <- "CY_CDM_V1_50k_Random"

PATIENTS <- LoadCohortSubjects(project_name,
                                    data_base_name,
                                    start_date = NULL, end_date = NULL)

PRESCRIPTIONS <- LoadPrescription(project_name,
                                    data_base_name)
```

The queries performed on the data base using these functions can vary in complexity. We can try query for a more complex cohort, for example:

Patients with COPD, with an exacerbation in the last 12 months, presently taking dual or triple therapy.

To start with we need to define the concept ID for COPD. Athena shows that there is a concept for COPD including an exacerbation, as shown in figure 12:

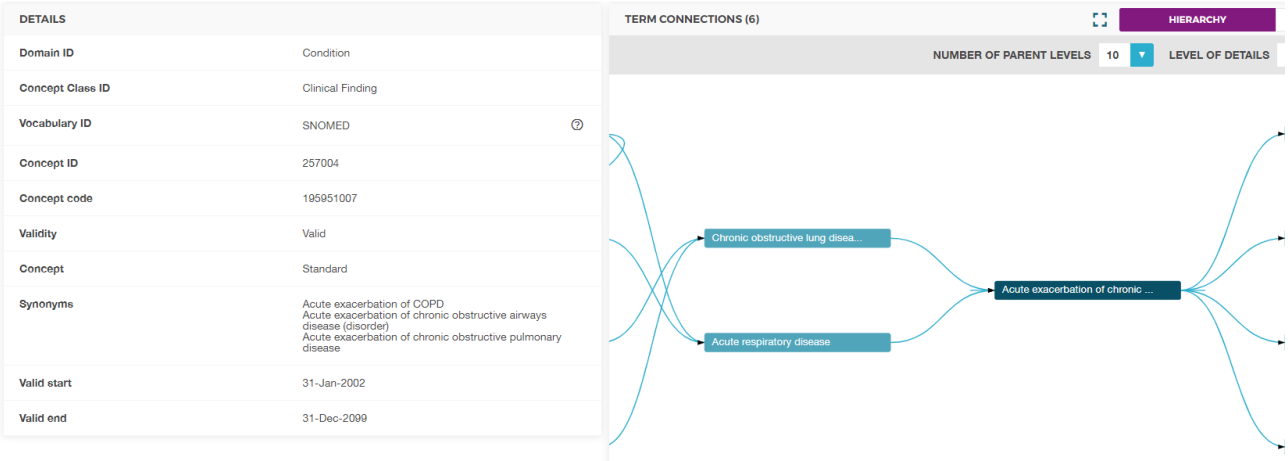


Figure 12: COPD, with an exacerbation

Next, dual prescription is a combination of prescriptions, more specifically 2 or more of ICS, LABA, LAMA (as defined by a clinician). The concept ids are defined as follows:

Prescription	**Concept ID**
ICS - Glucocorticoids	21603283
LABA - Selective beta-2-adrenoreceptor agonists	21603255
LAMA - Anticholinergics	21603292

Table 6: Concept IDs for dual or triple therapy

With the concepts defined we can start off by querying all the patients in the data base suffering from COPD,

with an exacerbation in the last 12 months, as well as all the prescriptions. Note for the last 12 months we can use the start date variable in the function to set these bounds. We can also define the prescriptions, note that the prescriptions are ATC 4th level rather than 5th level and thus ATC_4th flag variable needs to be set to true. The code is as follows:

```
#COPD Cohort
project_name_COPD <- "yhcr-prd-phm-bia-core"
data_base_name_COPD <- "CY_CDM_V1_50k_Random"
start_date <- as.Date('2020-01-01')

#Patients
PATIENTS_COPD <- LoadCohortSubjects(project_name_COPD,
                                    data_base_name_COPD,
                                    condition_concept = 257004,
                                    start_date = start_date, end_date = NULL)
patients_copd <- unique(PATIENTS_COPD$person_id)

#Prescriptions
PRESCRIPTIONS_Glucocorticoids <- LoadPrescription(project_name_COPD,
                                                  data_base_name_COPD,
                                                  prescription = 21603283,
                                                  ATC_4th = TRUE)
PRESCRIPTIONS_BETA2 <- LoadPrescription(project_name_COPD,
                                        data_base_name_COPD,
                                        prescription = 21603255,
                                        ATC_4th =TRUE)
PRESCRIPTIONS_Anticholinergics  <- LoadPrescription(project_name_COPD,
                                                    data_base_name_COPD,
                                                    prescription = 21603292,
                                                    ATC_4th = TRUE)
```

Next to filter the prescription data frames using the patients_copd list. We do this rather than filtering by the patients_copd in the function to have more accuracy choosing the prescriptions as well as speeding up the query. Code for this is as follows:

```
PRESCRIPTIONS_Glucocorticoids <-
  PRESCRIPTIONS_Glucocorticoids[PRESCRIPTIONS_Glucocorticoids$person_id
                                %in% patients_copd, ]
PRESCRIPTIONS_BETA2 <-
  PRESCRIPTIONS_BETA2[PRESCRIPTIONS_BETA2$person_id
                      %in% patients_copd, ]

PRESCRIPTIONS_Anticholinergics <-
  PRESCRIPTIONS_Anticholinergics[PRESCRIPTIONS_Anticholinergics$person_id
                                 %in% patients_copd, ]
```

We can then find a list of person ids that appear more than once in all three prescription data frames, indicating dual or triple therapy. These values are pushed into the variable x:

```
# Getting list of person ids from each prescription data frame
A <- unique(PRESCRIPTIONS_Glucocorticoids$person_id)
B <- unique(PRESCRIPTIONS_BETA2$person_id)
C <- unique(PRESCRIPTIONS_Anticholinergics$person_id)

#Joining all three lists
x <- c(A,B,C)
```

```
# Finding person ids that appear more than once
x <- unique(x[duplicated(x)])
print(length(x))
```

Note that the data being used is a subset, hence the low number of patients.

We can then finally filter all the data frames using variable x to get only the patients of interest in all the data frames:

```
#Patients
PATIENTS_COPD <- PATIENTS_COPD[PATIENTS_COPD$person_id %in% x, ]

#Prescriptions
PRESCRIPTIONS_Glucocorticoids <-
  PRESCRIPTIONS_Glucocorticoids[PRESCRIPTIONS_Glucocorticoids$person_id
                                %in% x, ]
PRESCRIPTIONS_BETA2 <-
  PRESCRIPTIONS_BETA2[PRESCRIPTIONS_BETA2$person_id
                      %in% x, ]

PRESCRIPTIONS_Anticholinergics <-
  PRESCRIPTIONS_Anticholinergics[PRESCRIPTIONS_Anticholinergics$person_id
                                 %in% x, ]
```

3.3.2 Standard SQL Queries

The other way to retrieve cohorts within the CDM is to run SQL queries on it specifically. Note the CohortCreationLibrary will still be used here as it contains the QueryOmop function which can be used to query the OMOP CDM tables.

In order to understand how to query the OMOP CDM, queries for each table of interest will be constructed, along with any necessary joins to extract as much information as possible. These tables include: Person Table, Observation Period Table, Visit Occurrence, Condition Occurrence, Drug Exposure, and Procedure Occurrence. Note to understand these tables in more detail a look up table can be found in the resources of interest section.

3.3.2.1 Person Table Basic demographic information is held in the person table. This table serves as the central identity management for all persons in the database. It contains records that uniquely identify each person or patient, and some demographic information (Ohdsi).

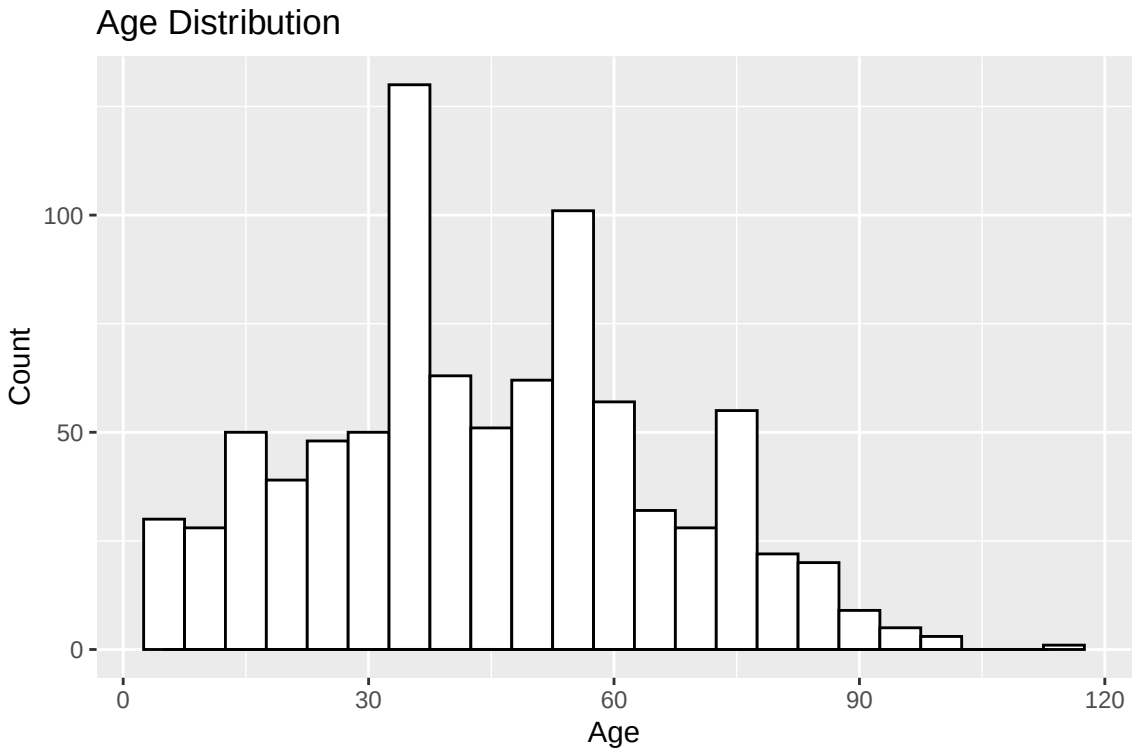
```
project_name <- "yhcr-prd-phm-bia-core"
sql <- "SELECT *, cast(CONCAT(year_of_birth,'-',month_of_birth,'-',day_of_birth) as string)
as Birthday FROM `yhcr-prd-phm-bia-core.CY_CDM_V1_50k_Random.person` LIMIT 1000"
person <- QueryOmop(project_name, sql)
```

Note that the birthday column has been loaded in as a string as discussed in chapter 2. We can use this data to join to other tables to get demographic information of patients suffering from certain illness.

For example we can visualize the age distribution using this table:

```
#Filtering for people who are only alive
person <- person[is.na(person$death_datetime),]
p <-  person #making duplicate variable

#Converting string to birthday
p$Birthday <- as.Date(p$Birthday)
p <- unique(p)
```

```r
p$age <- floor(age_calc(p$Birthday, units = "years"))

#Age Distribution
ggplot(p, aes(x=age)) +
  geom_histogram(aes(y=..count..), colour="black", fill="white", binwidth=5)+
  geom_density(alpha=.2, fill="#FF6666") +
  labs(title="Age Distribution",x="Age", y = "Count")
```



Other notable information includes gender and the race of the subject. For example we can retrieve the number of people race in the data frame:

```r
race_count <- as.data.frame(table(person$race_concept_id)) %>%
  arrange(desc(Freq)) %>% rename(concept_id = Var1)
```

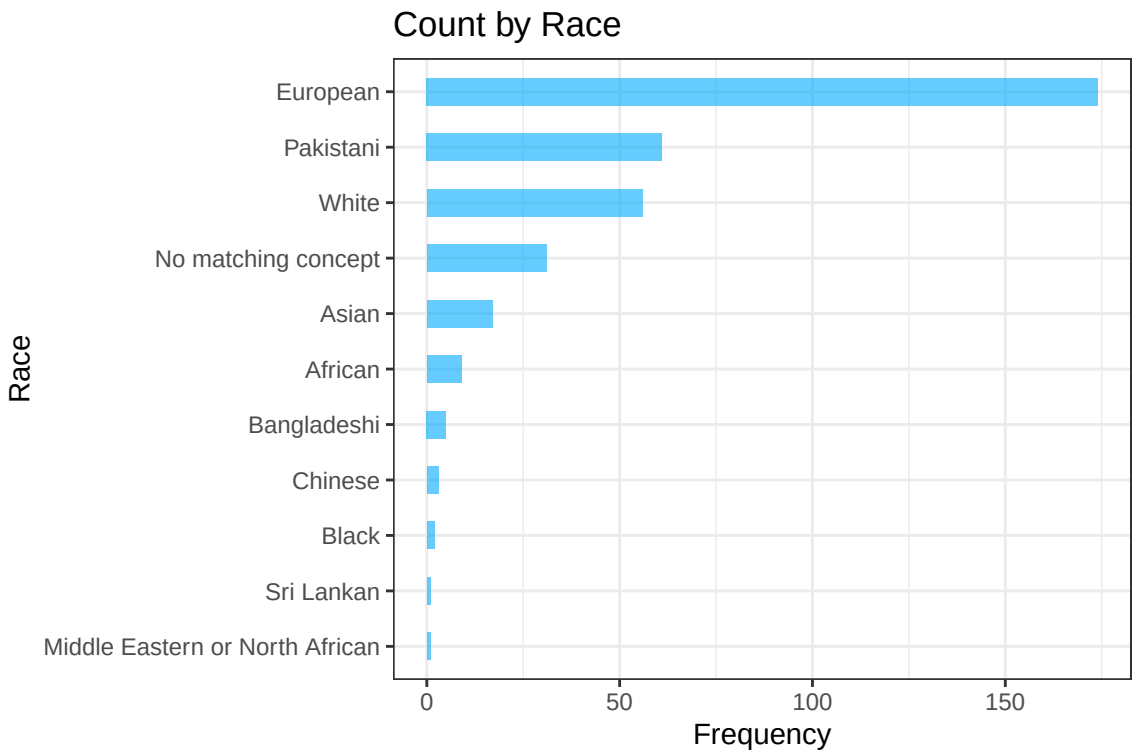
In order to get the meaning of each of these codes, we can either look them up in Athena, or query the Concept table and join it with race_count data frame. Code to query the concept table is as follows:

```r
project_name <- "yhcr-prd-phm-bia-core"
sql <- "SELECT concept_id, concept_name FROM `yhcr-prd-phm-bia-core.CY_CDM_V1_50k_Random.concept`"
concept <- QueryOmop(project_name, sql)
```

Then all we have to do is join the race_count data frame with the concept data frame and then we can get the individual values for each race:

```r
concept1 <-  concept[concept$concept_id %in% unique(person$race_concept_id) ,]
race_count <- merge(race_count, concept1) %>%
  arrange(desc(Freq))
race_count[-1,]  %>%
  ggplot(aes(x = reorder(concept_name, Freq), y = Freq)) +
  geom_bar(stat="identity", fill="#00abff", alpha=.6, width=.5) +
  coord_flip() +
```

```
  xlab("Race") + ylab("Frequency") + ggtitle("Count by Race") +
  theme_bw()
```



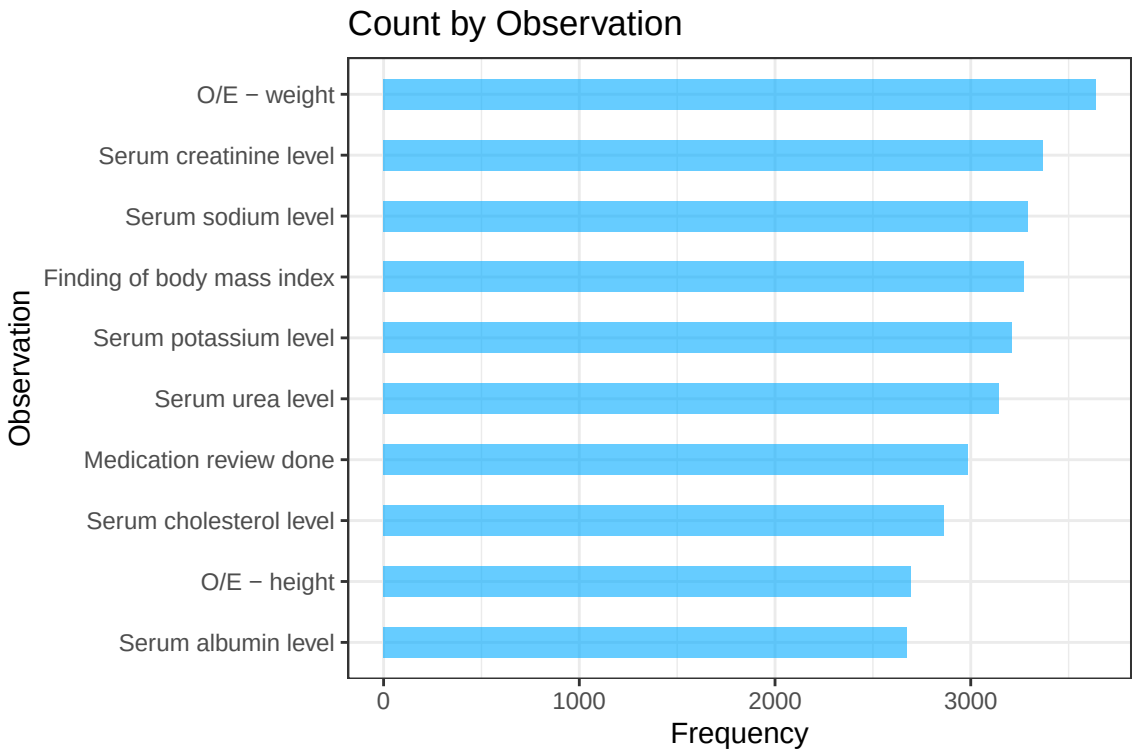
3.3.2.2 Observation Period Table This table contains records which define spans of time during which two conditions are expected to hold: (i) Clinical Events that happened to the Person are recorded in the Event tables, and (ii) absence of records indicate such events did not occur during this span of time (Ohdsi).

Lets use the person_ids in the person table to query the observation table to see what observations exist for this randomly selected cohort. Note the paste function can be used to join strings together in R:

```
project_name <- "yhcr-prd-phm-bia-core"
sql <- paste("SELECT * FROM `yhcr-prd-phm-bia-core.CY_CDM_V1_50k_Random.observation`
             where person_id in (", toString(person$person_id), ")")
observations <- QueryOmop(project_name, sql)
```

We can then join the observation table to the concept table to get the meaning of each observation concept id, and plot the 10 most common observations:

```
concept1 <-  concept[concept$concept_id %in% unique(observations$observation_concept_id) ,]
observation_count <- as.data.frame(table(observations$observation_concept_id)) %>%
  arrange(desc(Freq)) %>% rename(concept_id = Var1)
observation_count <- merge(observation_count, concept1) %>%
  arrange(desc(Freq))
head(observation_count, 10)  %>%
  ggplot(aes(x = reorder(concept_name, Freq), y = Freq)) +
  geom_bar(stat="identity", fill="#00abff", alpha=.6, width=.5) +
  coord_flip() +
  xlab("Observation") + ylab("Frequency") + ggtitle("Count by Observation") +
  theme_bw()
```



As it can be seen in the graph above, most observation entries for this cohort are simple recordings, such as weight, height and more complicated observations such as Urea levels. We can also retrieve the average value for an observation of a cohort from the observations recorded. For example the average weight taken for the cohort. Code as follows:

```
#retrieve latest weight recording for each r
weight_observations <- observations[observations$observation_concept_id == 4060333, ] %>%
  select(person_id, observation_concept_id, observation_date, value_as_number,
         unit_source_value) %>%
  drop_na()  %>%
  group_by(person_id) %>%
  slice(which.max(as.Date(observation_date, '%m/%d/%Y')))

#Calculate average weight for each person recorded
#(515 patients had their weight taken at some point)
mean(weight_observations$value_as_number)
```

```
## [1] 74.08308
```

3.3.2.3 Visit Occurrence This table contains Events where Persons engage with the healthcare system for a duration of time. They are often also called "Encounters". Visits are defined by a configuration of circumstances under which they occur, such as (i) whether the patient comes to a healthcare institution, the other way around, or the interaction is remote, (ii) whether and what kind of trained medical staff is delivering the service during the Visit, and (iii) whether the Visit is transient or for a longer period involving a stay in bed (Ohdsi).

In summary, this table returns information about the patients visit to a place where health care is received. For example, a patient can go to a GP to receive health care, or in an emergency they will go to A&E. Lets use the person_ids in the person table to query the Visit Occurrence table:

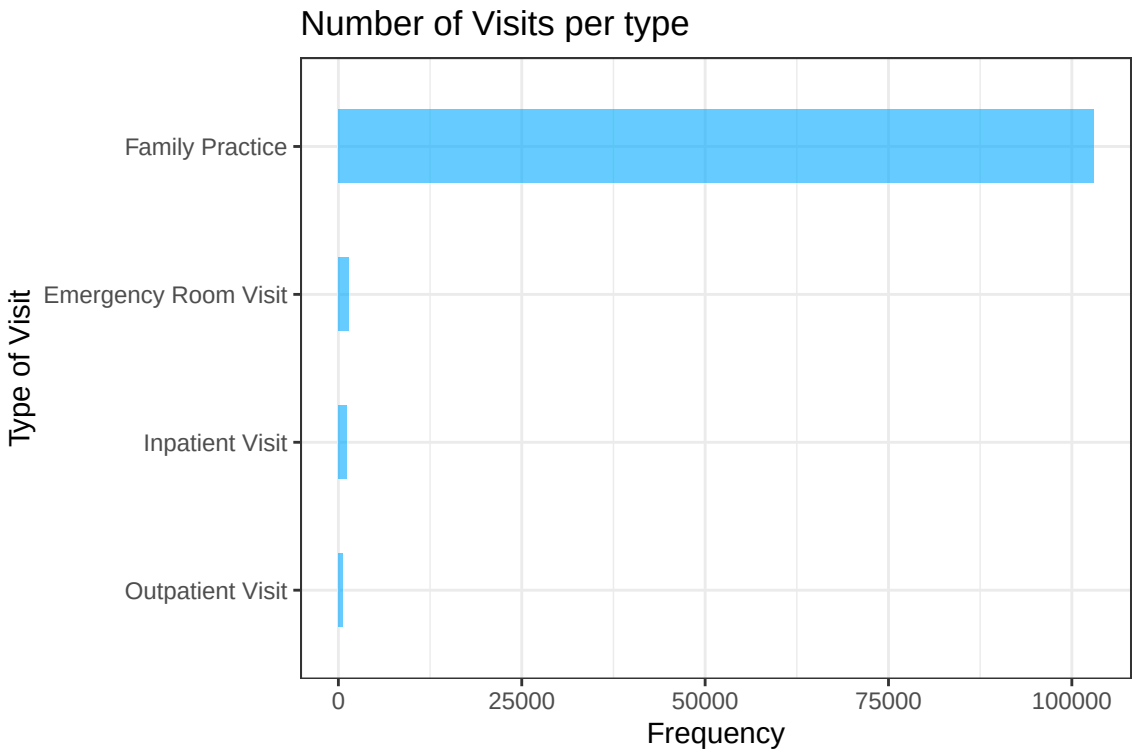
```
project_name <- "yhcr-prd-phm-bia-core"
sql <- paste("SELECT * FROM `yhcr-prd-phm-bia-core.CY_CDM_V1_50k_Random.visit_occurrence`
             where person_id in (", toString(person$person_id), ")")
visits <- QueryOmop(project_name, sql)
```

A particular column of interest here is the visit_concept_id, which records the kind of visit, for example; inpatient or outpatient. The reasons for these visits are defined below:

- Inpatient Visit: Person visiting hospital, at a Care Site, in bed, for duration of more than one day, with physicians and other Providers permanently available to deliver service around the clock (Ohdsi).
- Emergency Room and Inpatient Visit: Person visiting ER followed by a subsequent Inpatient Visit, where Emergency department is part of hospital, and transition from the ER to other hospital departments is undefined (Ohdsi)
- Non-hospital institution Visit: Person visiting dedicated institution for reasons of poor health, at a Care Site, long-term or permanently, with no physician but possibly other Providers permanently available to deliver service around the clock, for example a family practice visit. (Ohdsi)
- Outpatient Visit: Person visiting dedicated ambulatory healthcare institution, at a Care Site, within one day, without bed, with physicians or medical Providers delivering service during Visit (Ohdsi)
- Home Visit: Provider visiting Person, without a Care Site, within one day, delivering service (Ohdsi)
- Telehealth Visit: Patient engages with Provider through communication media (Ohdsi)
- Pharmacy Visit: Person visiting pharmacy for dispensing of Drug, at a Care Site, within one day (Ohdsi)
- Laboratory Visit: Patient visiting dedicated institution, at a Care Site, within one day, for the purpose of a Measurement (Ohdsi).
- Ambulance Visit: Person using transportation service for the purpose of initiating one of the other Visits, without a Care Site, within one day, potentially with Providers accompanying the Visit and delivering service (Ohdsi)
- Case Management Visit: Person interacting with healthcare system, without a Care Site, within a day, with no Providers involved, for administrative purposes (Ohdsi)

We can thus find information about a cohort regarding the visit, for example, under what circumstances did the visit take place. I.e. was it a trip to the GP, or an emergency room visit. Code for this as follows:

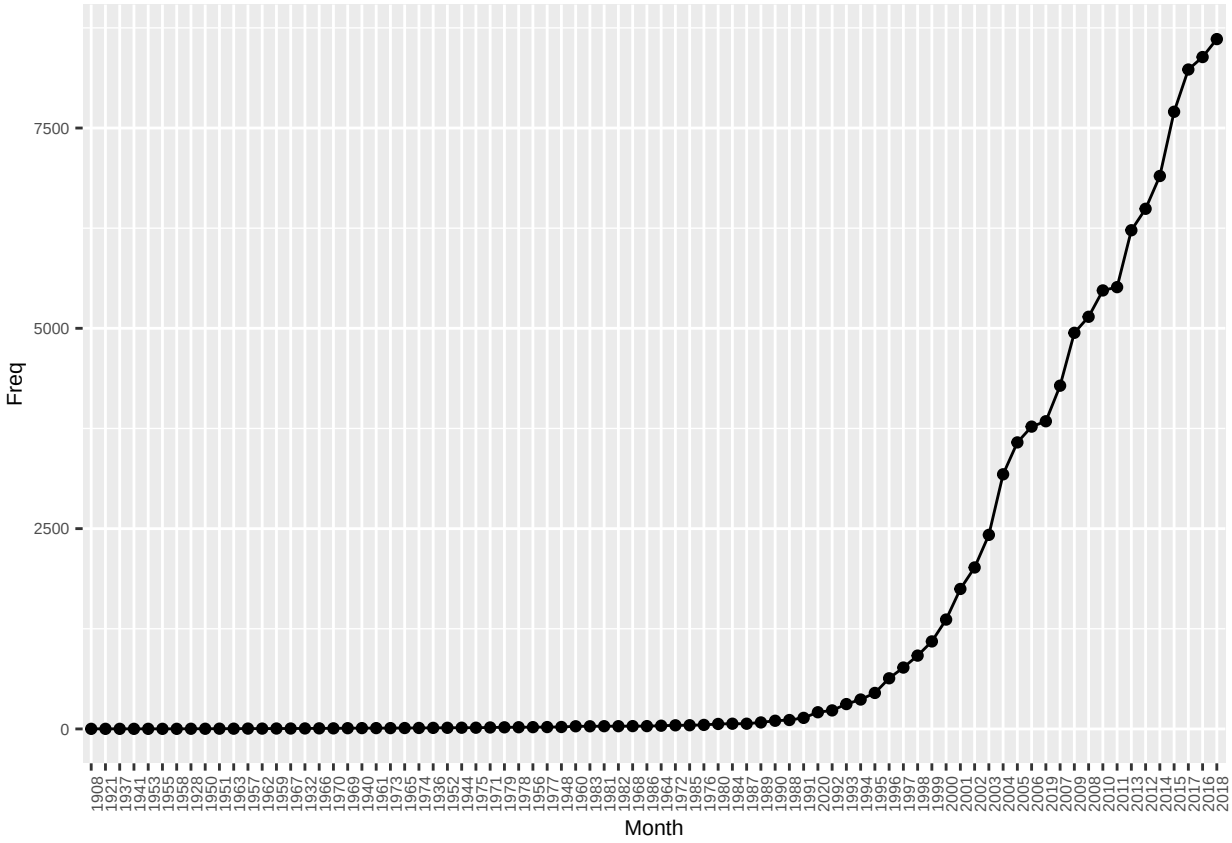
```
concept1 <-  concept[concept$concept_id %in% unique(visits$visit_concept_id) ,]
visits_count <- as.data.frame(table(visits$visit_concept_id)) %>%
  arrange(desc(Freq)) %>% rename(concept_id = Var1)
visits_count <- merge(visits_count, concept1) %>%
  arrange(desc(Freq))

visits_count  %>%
  ggplot(aes(x = reorder(concept_name, Freq), y = Freq)) +
  geom_bar(stat="identity", fill="#00abff", alpha=.6, width=.5) +
  coord_flip() +
  xlab("Type of Visit") + ylab("Frequency") + ggtitle("Number of Visits per type") +
  theme_bw()
```



We can thus find useful information about how a condition was diagnosed. For example; the majority of patients get diagnosed Asthma during a GP visit, or do they get diagnosed with Asthma after they've had an Asthma attack and have had to go to A and E.

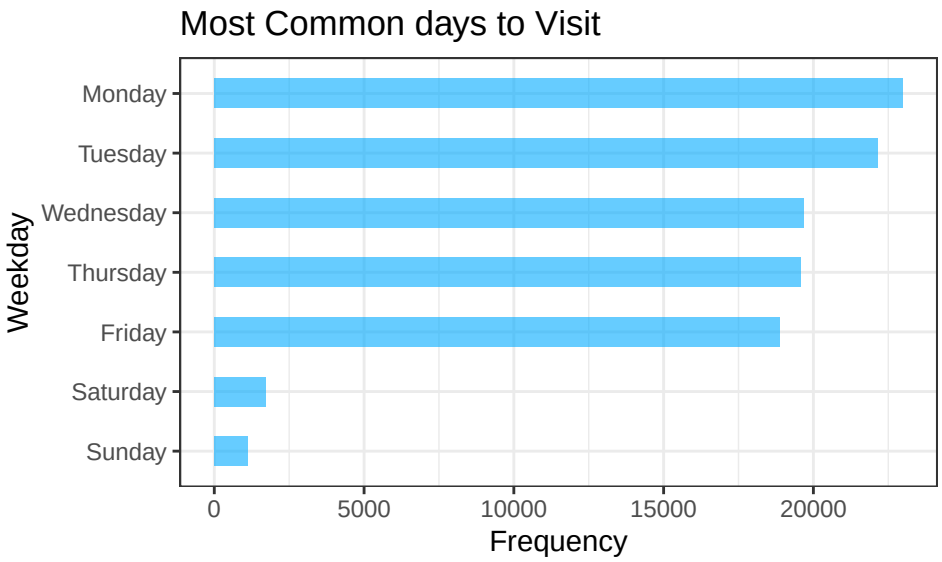
Not only this but because we have dates regarding the visitation, we can thus make time series graphs about the number of visits to a health care center, for example, the number of individual visits per year for a specific cohort:

```
visits$year <- format(as.Date(visits$visit_start_date, format="%d/%m/%Y"),"%Y")
visits_count <- as.data.frame(table(visits$year)) %>%
  arrange(desc(Freq)) %>% rename(Year = Var1)

visits_count$Freq <- as.numeric(as.character(visits_count$Freq))

visits_count  %>%
  ggplot(aes(x = reorder(Year, Freq), y = Freq)) +
  geom_point() +
  labs(x="Month", colour="Year") +
  geom_line(aes(group=1)) + theme(text = element_text(size=7.5),
                                  axis.text.x = element_text(angle = 90))
```



We can build on this further and plot the days of the week where most visitations occur:

```
visits$day <- weekdays(visits$visit_start_date)
visits_count <- as.data.frame(table(visits$day)) %>%
  arrange(desc(Freq)) %>% rename(Weekday = Var1)

visits_count  %>%
  ggplot(aes(x = reorder(Weekday, Freq), y = Freq)) +
  geom_bar(stat="identity", fill="#00abff", alpha=.6, width=.5) +
  coord_flip() +
  xlab("Weekday") + ylab("Frequency") + ggtitle("Most Common days to Visit") +
  theme_bw()
```


Most Common days to Visit
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
Sunday
Weekday
0
5000
10000
15000
20000
Frequency

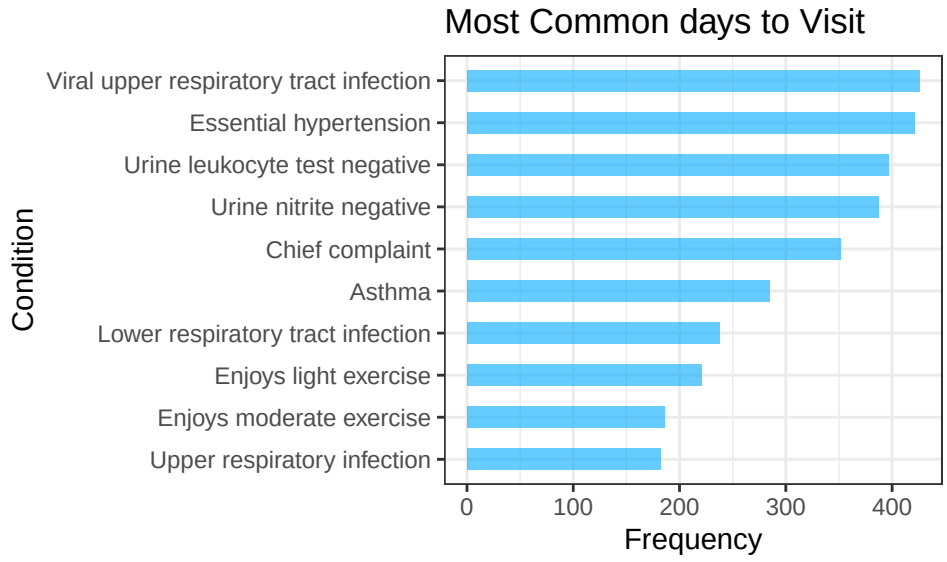
3.3.2.4 Condition Occurrence This table contains records of Events of a Person suggesting the presence of a disease or medical condition stated as a diagnosis, a sign, or a symptom, which is either observed by a Provider or reported by the patient (Ohdsi).

This table will be quite often of the most importance, as it contains information regarding the diagnosis of the patient. Particular columns of interest include. Lets use the person_ids in the person table, previously defined to query the Condition Occurrence table, as well as join with the concept table:

```
project_name <- "yhcr-prd-phm-bia-core"
sql <- paste("SELECT * FROM `yhcr-prd-phm-bia-core.CY_CDM_V1_50k_Random.condition_occurrence`
             where person_id in (", toString(person$person_id), ")")
conditions <- QueryOmop(project_name, sql)

concept1 <-  concept[concept$concept_id %in% unique(conditions$condition_concept_id) ,]
conditions <- merge(conditions, concept1, by.x = "condition_concept_id",
                    by.y = "concept_id")
```

To start with, lets find the 10 most common diseases in this cohort:

```
condition_count <- as.data.frame(table(conditions$concept_name)) %>%
  arrange(desc(Freq)) %>% rename(Condition = Var1)

#first rows removed due to being tests not diagnoses
head(tail(condition_count, -7), 10)  %>%
  ggplot(aes(x = reorder(Condition, Freq), y = Freq)) +
  geom_bar(stat="identity", fill="#00abff", alpha=.6, width=.5) +
  coord_flip() +
  xlab("Condition") + ylab("Frequency") + ggtitle("Most Common days to Visit") +
  theme_bw()
```



It can be seen that Viral upper respiratory tract infection is the most common condition in this cohort, we can then find out the demographic of people suffering from this condition by merging the person table and the conditions table:

```
#gets list of person_ids
conditions_viral <- conditions[conditions$concept_name ==
                                 "Viral upper respiratory tract infection", ]
```

```
#Finding demographic
conditions_viral <- merge(conditions_viral,
                          person[person$person_id %in% conditions_viral$person_id, ],
                          by = "person_id") %>% select(one_of(c("person_id",
                                                                "condition_start_date",
                                                                "condition_end_date",
                                                                "Birthday"))) %>%
                    group_by(person_id) %>%
                    slice(which.min(as.Date(condition_start_date, '%m/%d/%Y')))
```

We can then find then find the average age of diagnosis of the person suffering from this condition, to get an estimate of when this condition becomes prevalent:

```
conditions_viral$age_diag <- floor(age_calc(dob = as.Date(conditions_viral$Birthday),
                                            enddate = as.Date(conditions_viral$condition_start_date),
                                units = "days"))

#For accuracy convert days into years instead of converting straight into years
mean(as.numeric(conditions_viral$age_diag )) * 0.002738
```

```
## [1] 23.04218
```

Thus, the average age of the earliest diagnosis is 23.

3.3.2.5 Drug Exposure This table captures records about the exposure to a Drug ingested or otherwise introduced into the body. A Drug is a biochemical substance formulated in such a way that when administered to a Person it will exert a certain biochemical effect on the metabolism. Drugs include prescription and over-the-counter medicines, vaccines, and large-molecule biologic therapies. Radiological devices ingested or applied locally do not count as Drugs (Ohdsi).

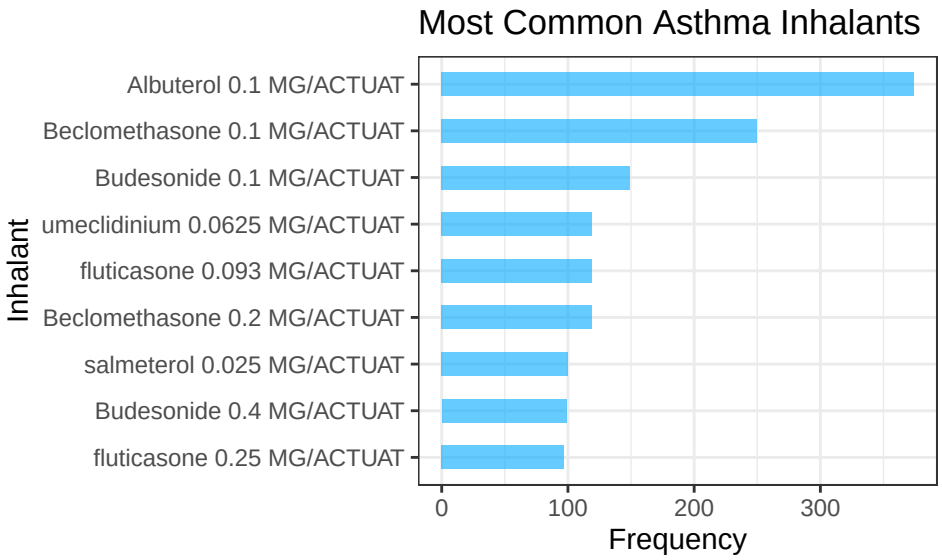
This table will be quite often of the most importance as well, as cohorts are normally set in bounds by both the condition and the prescription given. Lets use the person_ids in the person table, previously defined to query the Condition Occurrence table, as well as join with the concept table:

```
project_name <- "yhcr-prd-phm-bia-core"
sql <- paste("SELECT * FROM `yhcr-prd-phm-bia-core.CY_CDM_V1_50k_Random.drug_exposure`
             where person_id in (", toString(person$person_id), ")")
drugs <- QueryOmop(project_name, sql)

concept1 <-  concept[concept$concept_id %in% unique(drugs$drug_concept_id) ,]
drugs <- merge(drugs, concept1, by.x = "drug_concept_id",
                     by.y = "concept_id")
```

Using this table, we can then find what drugs the persons suffering from Asthma are taking, more specifically what inhalants, as this was also a very common diagnosis:

```
cond <- conditions[conditions$concept_name ==
             "Asthma", ]
drugs_viral <- drugs[drugs$person_id %in% cond$person_id, ]
drugs_count <- as.data.frame(table(drugs_viral$concept_name)) %>%
  arrange(desc(Freq)) %>% rename(concept_name = Var1)

#Retrieving specifically inhalants
drugs_count <- drugs_count[drugs_count$concept_name %ilike% "inhalan", ]
drugs_count$concept_name <- word(drugs_count$concept_name, 1,3, sep=" ")
head(drugs_count, 9) %>%
  ggplot(aes(x = reorder(concept_name, Freq), y = Freq)) +
```

```
  geom_bar(stat="identity", fill="#00abff", alpha=.6, width=.5) +
  coord_flip() +
  xlab("Inhalant") + ylab("Frequency") + ggtitle("Most Common Asthma Inhalants") +
  theme_bw()
```



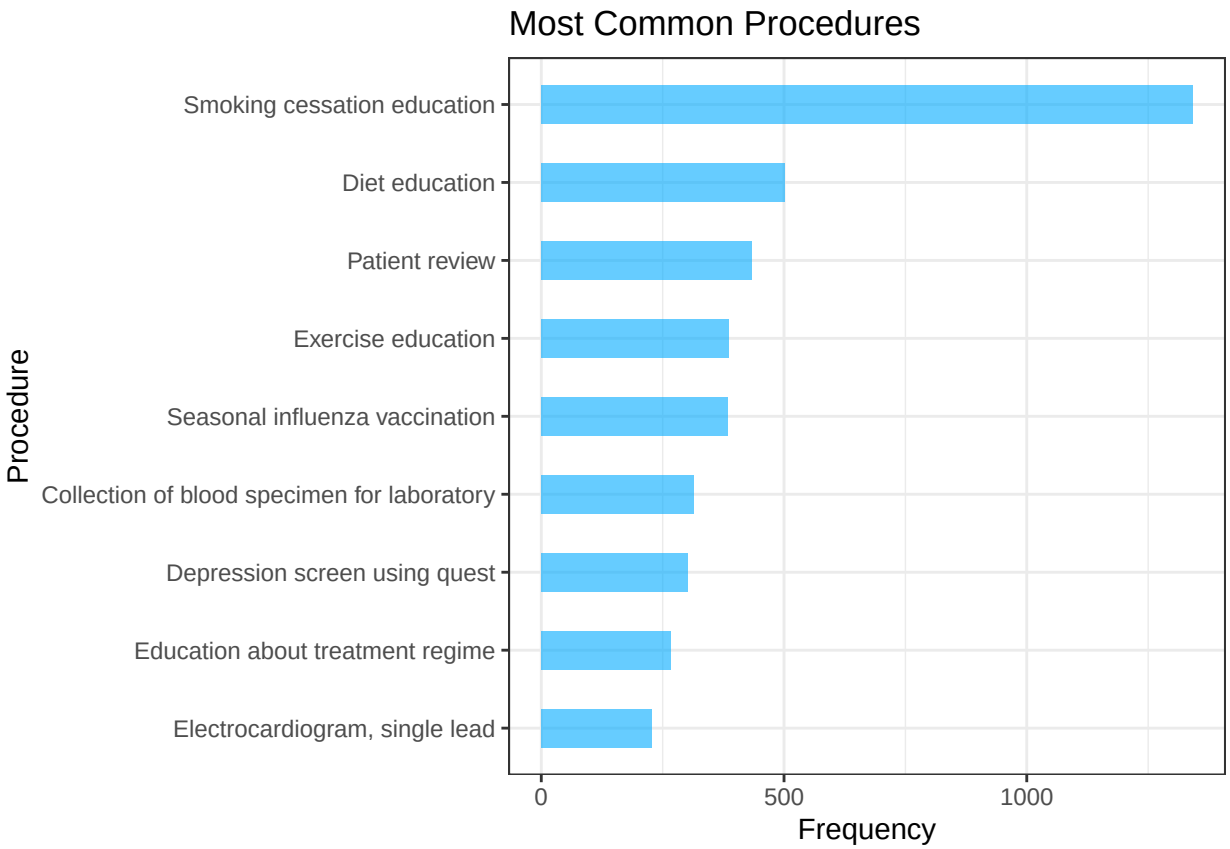
As we can see above, the two most common inhalers prescribed for this cohort are Albuterol 0.1MG and Beclomethasone 0.1MG.

3.3.2.6 Procedure Occurrence This table contains records of activities or processes ordered by, or carried out by, a healthcare provider on the patient with a diagnostic or therapeutic purpose (Ohdsi). An example of this would be surgery on a patient.

```
project_name <- "yhcr-prd-phm-bia-core"
sql <- paste("SELECT * FROM `yhcr-prd-phm-bia-core.CY_CDM_V1_50k_Random.procedure_occurrence`
             where person_id in (", toString(person$person_id), ")")
Procedures <- QueryOmop(project_name, sql)

concept1 <-  concept[concept$concept_id %in% unique(Procedures$procedure_concept_id) ,]
Procedures <- merge(Procedures, concept1, by.x = "procedure_concept_id",
                    by.y = "concept_id")
```

From this table we can see what procedures were carried on a patient, and on what visit. Lets see what procedures were most commonly given to this cohort:

```
procedure_count <- as.data.frame(table(Procedures$concept_name)) %>%
  arrange(desc(Freq)) %>% rename(concept_name = Var1)

head(procedure_count, 9) %>%
  ggplot(aes(x = reorder(concept_name, Freq), y = Freq)) +
  geom_bar(stat="identity", fill="#00abff", alpha=.6, width=.5) +
  coord_flip() +
  xlab("Procedure") + ylab("Frequency") + ggtitle("Most Common Procedures") +
  theme_bw()
```


Most Common Procedures
Smoking cessation education
Diet education
Patient review
Exercise education
Seasonal influenza vaccination
Collection of blood specimen for laboratory
Depression screen using quest
Education about treatment regime
Electrocardiogram, single lead
Procedure
0
500
1000
Frequency

4 Resources of interest

4.0.1 Detailed description of OMOP CDM:

https://ohdsi.github.io/CommonDataModel/cdm60.html

4.0.2 Book on the OHDSI collaborative:

https://ohdsi.github.io/TheBookOfOhdsi/

4.0.3 Queries from an earlier OMOP Model:

http://cdmqueries.omop.org/home

4.0.4 Workshops on the OMOP CDM:

https://www.ohdsi.org/tutorial-workshops/

5 Bibliography

Ohdsi. The Book of OHDSI Observational Health Data Sciences and Informatics. San Bernardino, Ca Ohdsi, 2019.

OHDSI. “Cdm60.Utf8.” Ohdsi.github.io, 19 Jan. 2021, ohdsi.github.io/CommonDataModel/cdm60.html. Accessed 30 Mar. 2021.