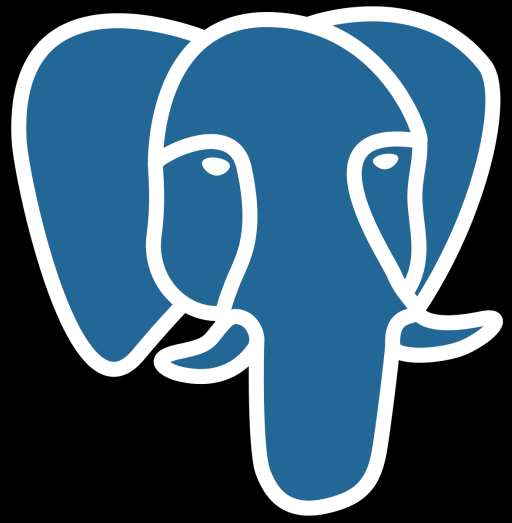
MIMICiii Workshop

Jeon Young Seok
Research assistant/Ph.D student
Saw Swee Hock School of Public Health, NUS
Github: Youngseok0001
Email: ephjys@nus.edu.sg

Objectives

- Introduction of MIMICiii Schema
- Working with the MIMICiii database using SQL
- Simple statistical analysis using Python3 in Colab environment



MIMICiii?

- A publicly available database released from MIT
- De-identified records from Beth Israel Deaconess Medical Center between 2001 ~ 2012.
- Contains detailed information of patients' history (Demographic, Diagnosis, Vital signs, Medication, Notevents, etc)
- Only Patients who stayed in(Intense Care Unit) ICU

Some Numbers

27 tables

ADMISSIONS, CAREGIVERS, CHARTEVENTS, …….

46,000 patients

46,520 distinct patients who have gone to ICU at least once

59,000 admissions

A patient may have been admitted to the hospital more than once

60,000 icustays

A single admission case could lead to multiple icustays

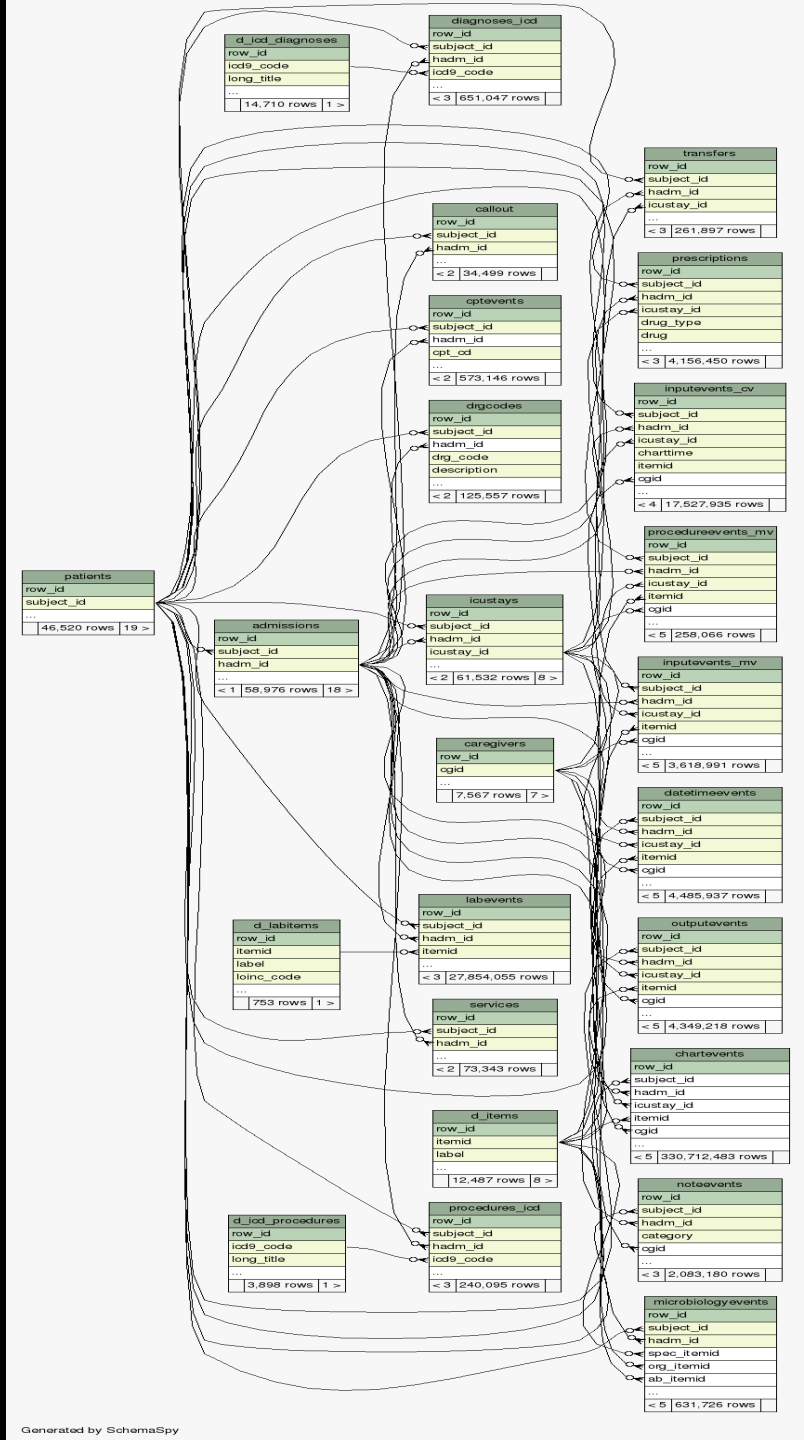
5,000 chartevents per admission case

5,000 chart observations(vital, medication, procedure, etc)

MIMICiii Schema

- Patients: a list of unique patients in MIMIC
- Admissons: a list of unique hospital admissions
- Icustays: a list of unique ICU stays.
- Diagnoses_icd: ICD diagnoses for patients
- Chartevents: Contains all chart data

https://mit-lcp.github.io/mimic-schema-spy/relationships.html



The 3 core tables

Field name	Type
subject_id	INTEGER
gender	STRING
dob	DATETIME
dod	DATETIME
expire_flag	INTEGER

Patients

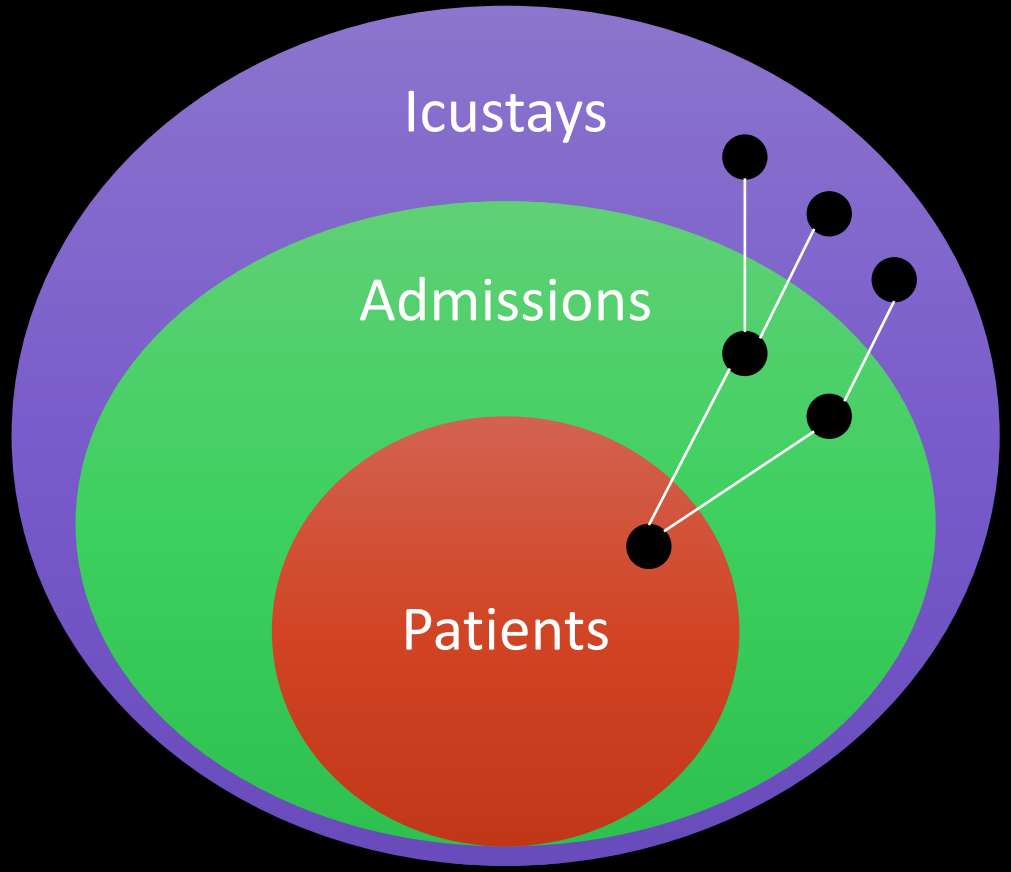
Field name	Type
subject_id	INTEGER
hadm_id	INTEGER
admittime	DATETIME
dischtime	DATETIME
deathtime	DATETIME

Admissions

Field name	Type
subject_id	INTEGER
hadm_id	INTEGER
icustay_id	INTEGER
intime	DATETIME
outtime	DATETIME

Icustays

Relationship between the three tables



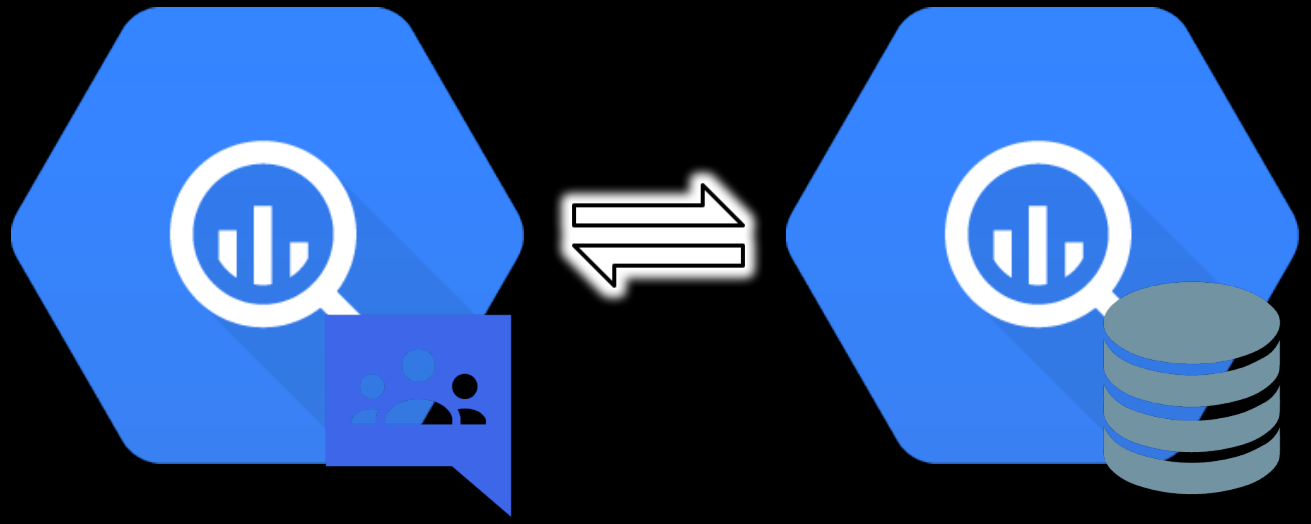
Diagnoses_icd and Chartevents

Column	Type
subject_id	int4
hadm_id	int4
seq_num	int4
icd9_code	varchar

Column	Type
subject_id	int4
hadm_id	int4
icustay_id	int4
itemid	int4
charttime	timestamp
value	varchar
valuenum	float8
valueuom	varchar

BigQuery Setup

Working in BigQuery

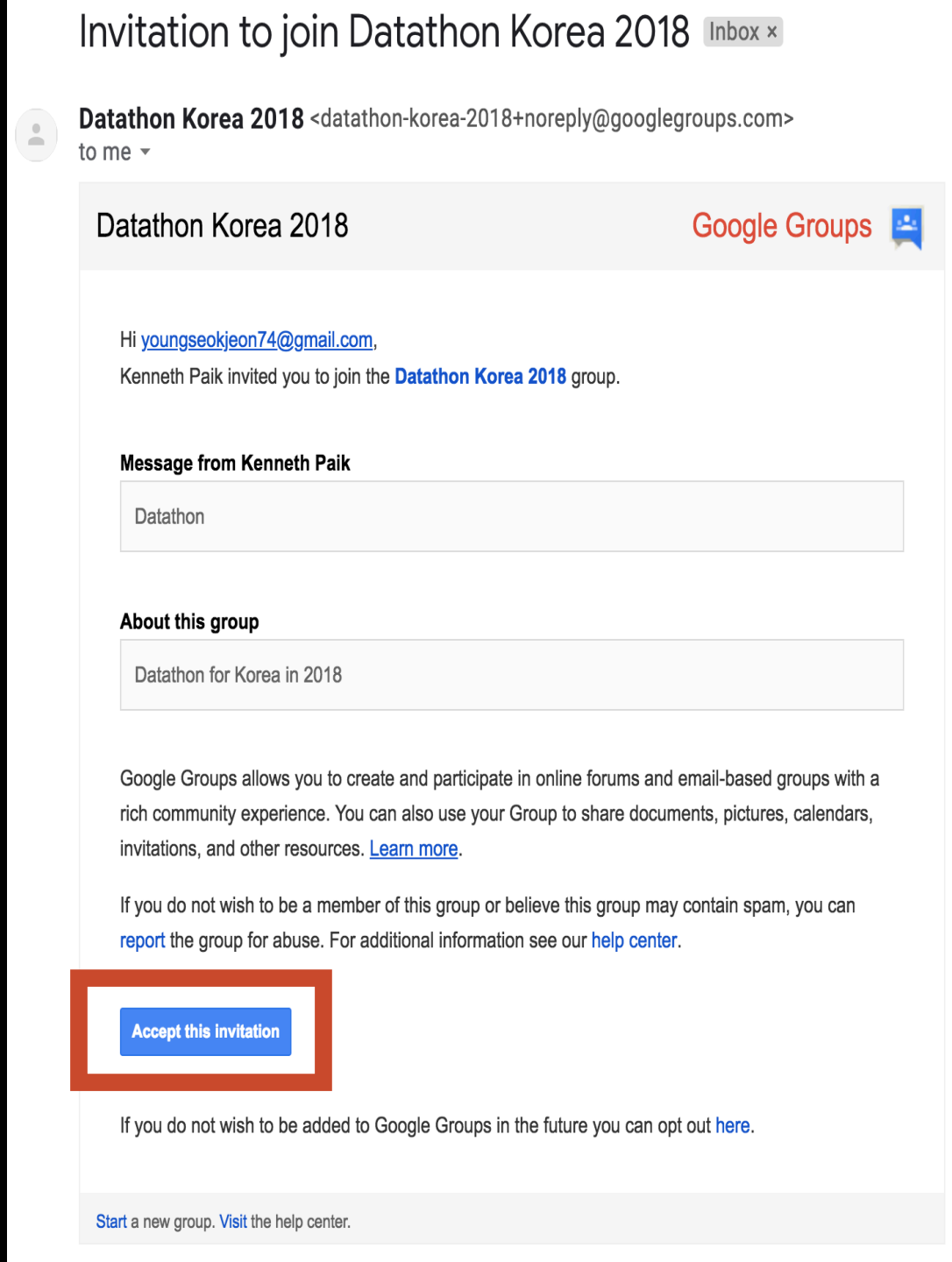


BigQuery Project
korea-datathon-2018

Google Cloud Storage
Physionet-data:MIMIC

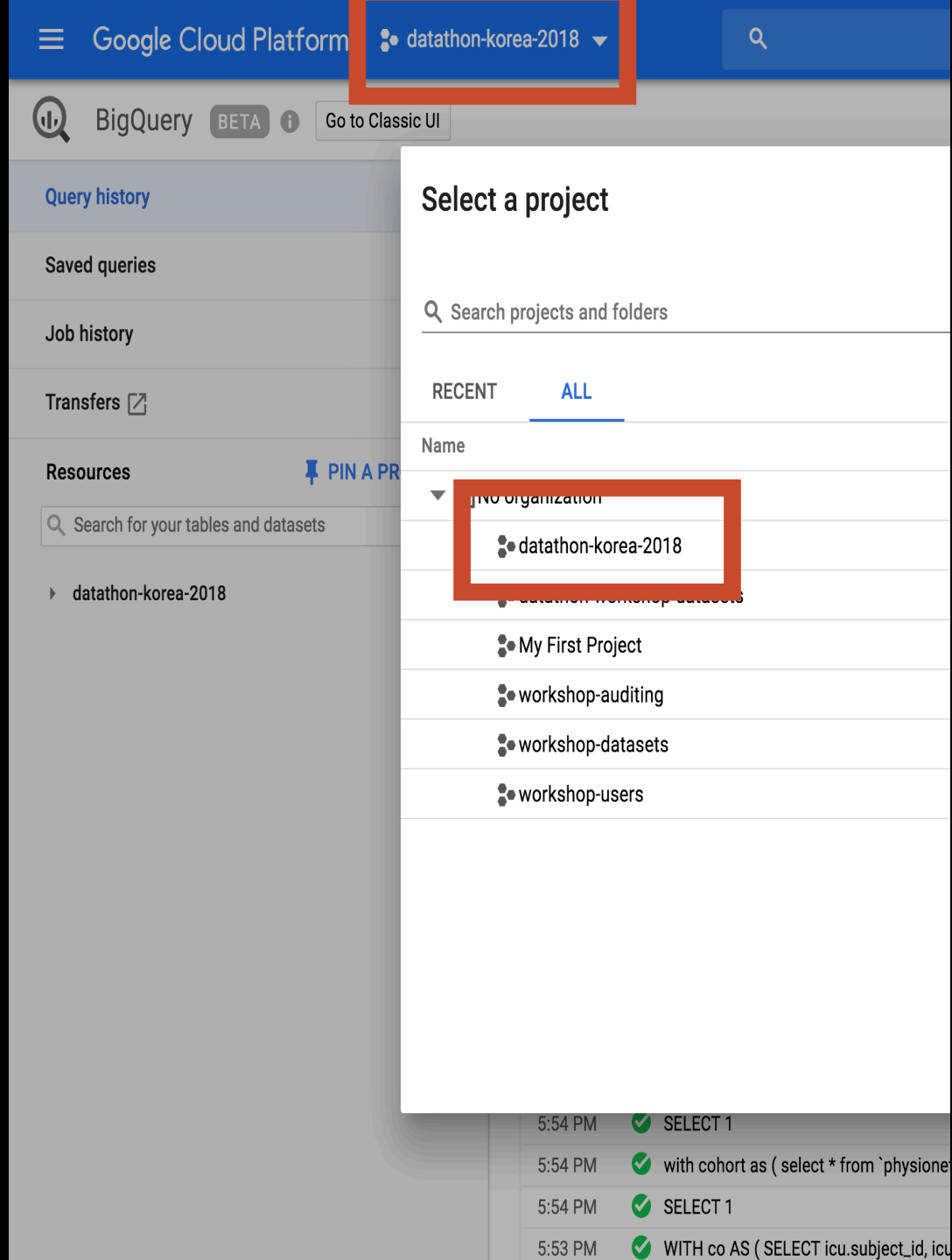
Working in BigQuery

- Go to Gmail and accept the invitation
- Go to “My group”
- You will now see “Datathon Korea 2018”



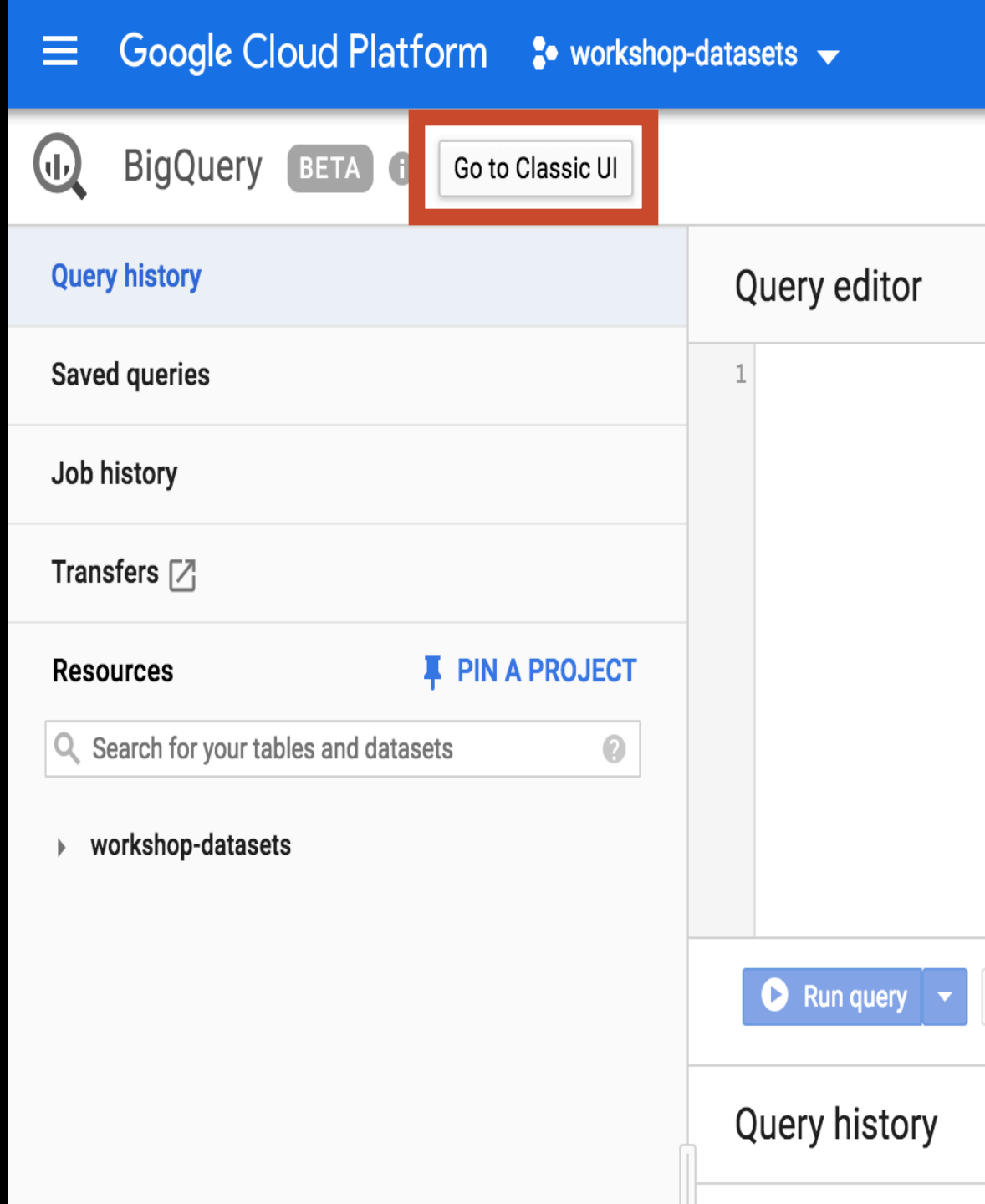
Working in BigQuery

- Go to the link below
  console.cloud.google.com/bigquery
- You are now a member of "Datathon-korea-2018"



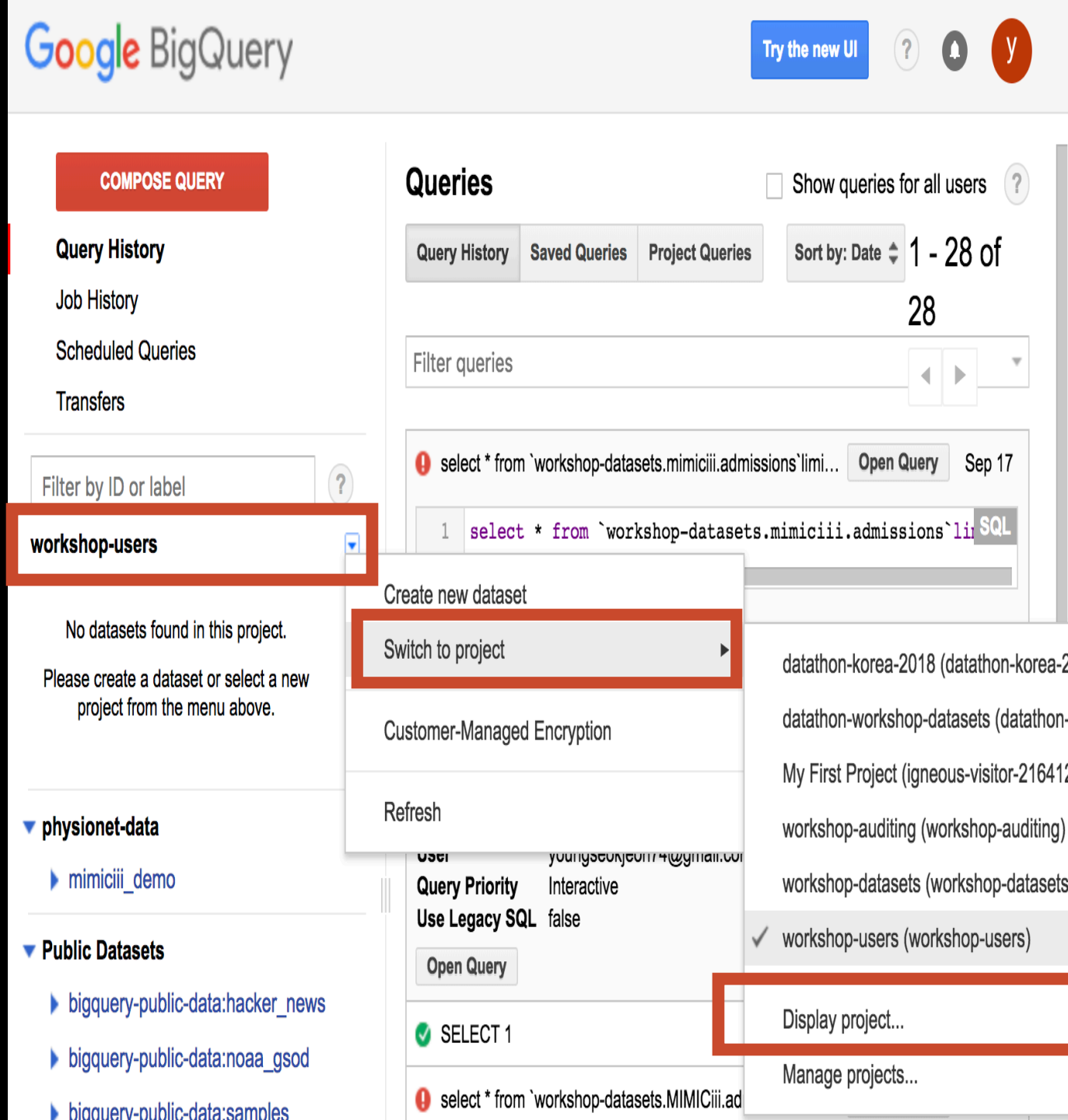
Working in BigQuery

- Switch to “classical UI”



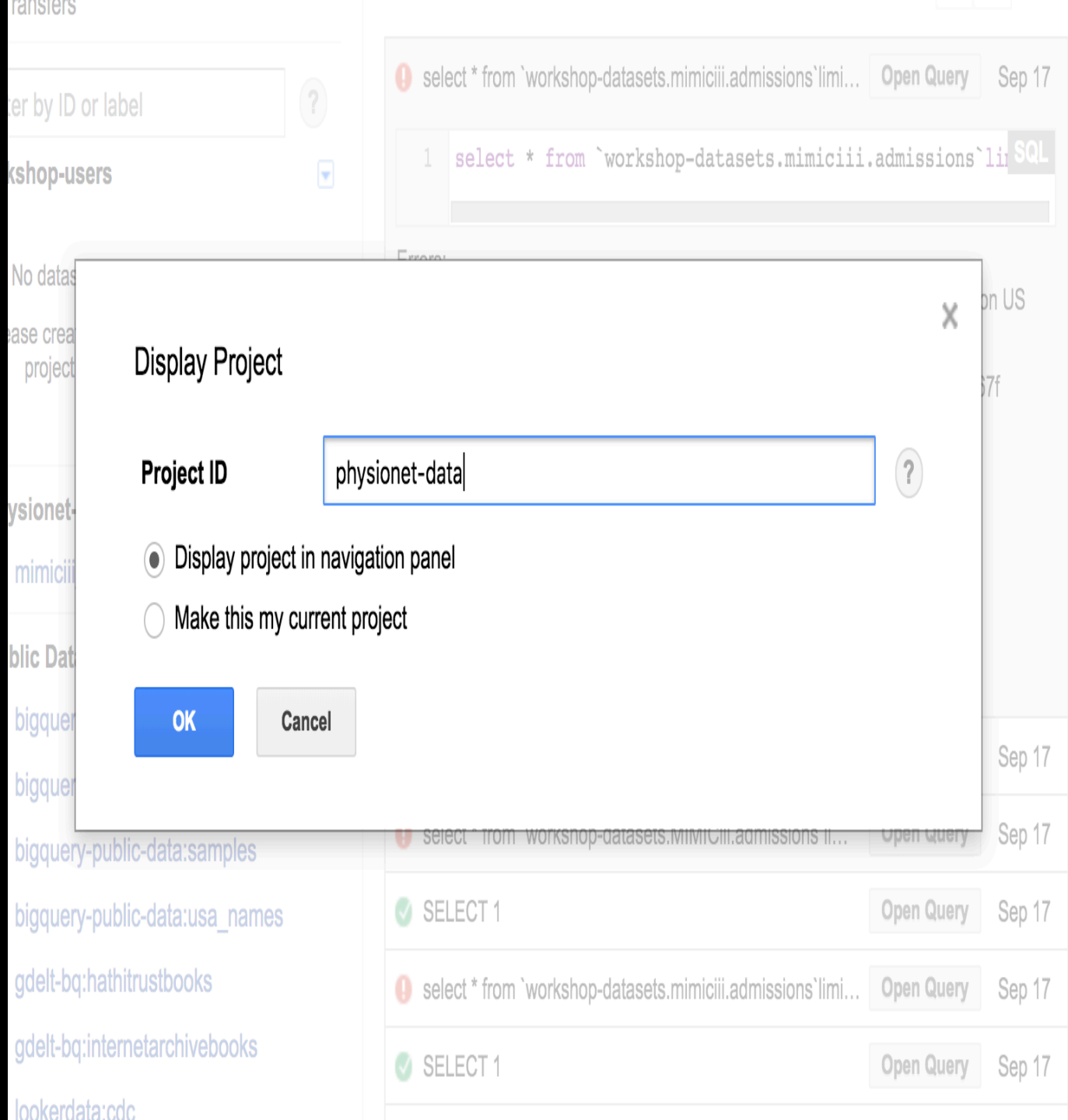
Working in BigQuery

- Select “Switch to project”
- Select “Display project”



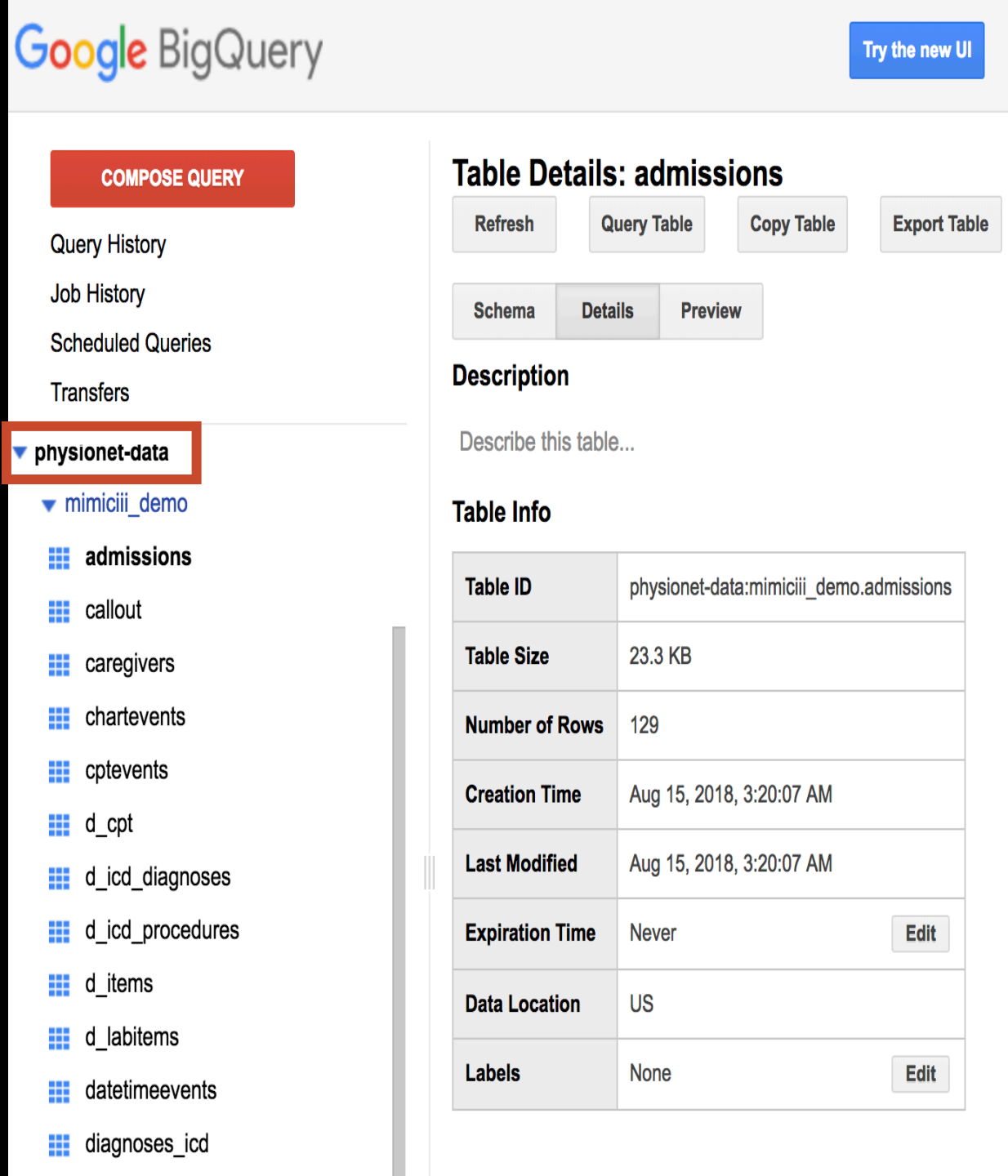
Working in BigQuery

- Type “Physionet-data” and click OK



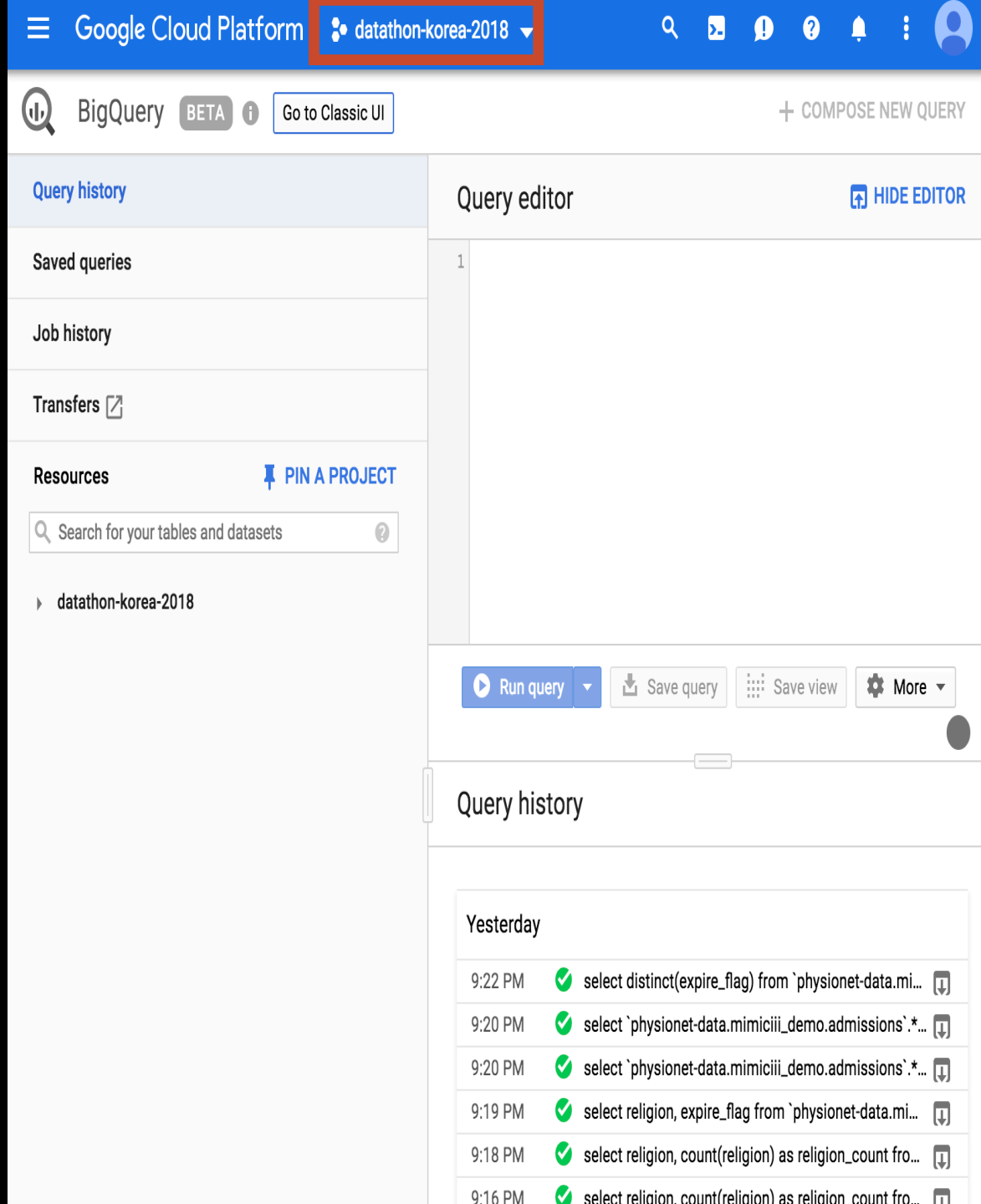
Working in BigQuery

- Physionet-data is now available to you
- You can now browse tables



Working in BigQuery

- Open a new window
- Go to the New UI
- Switch your project to “datathon-korea-2018”
- Now you can query from mimiciii database



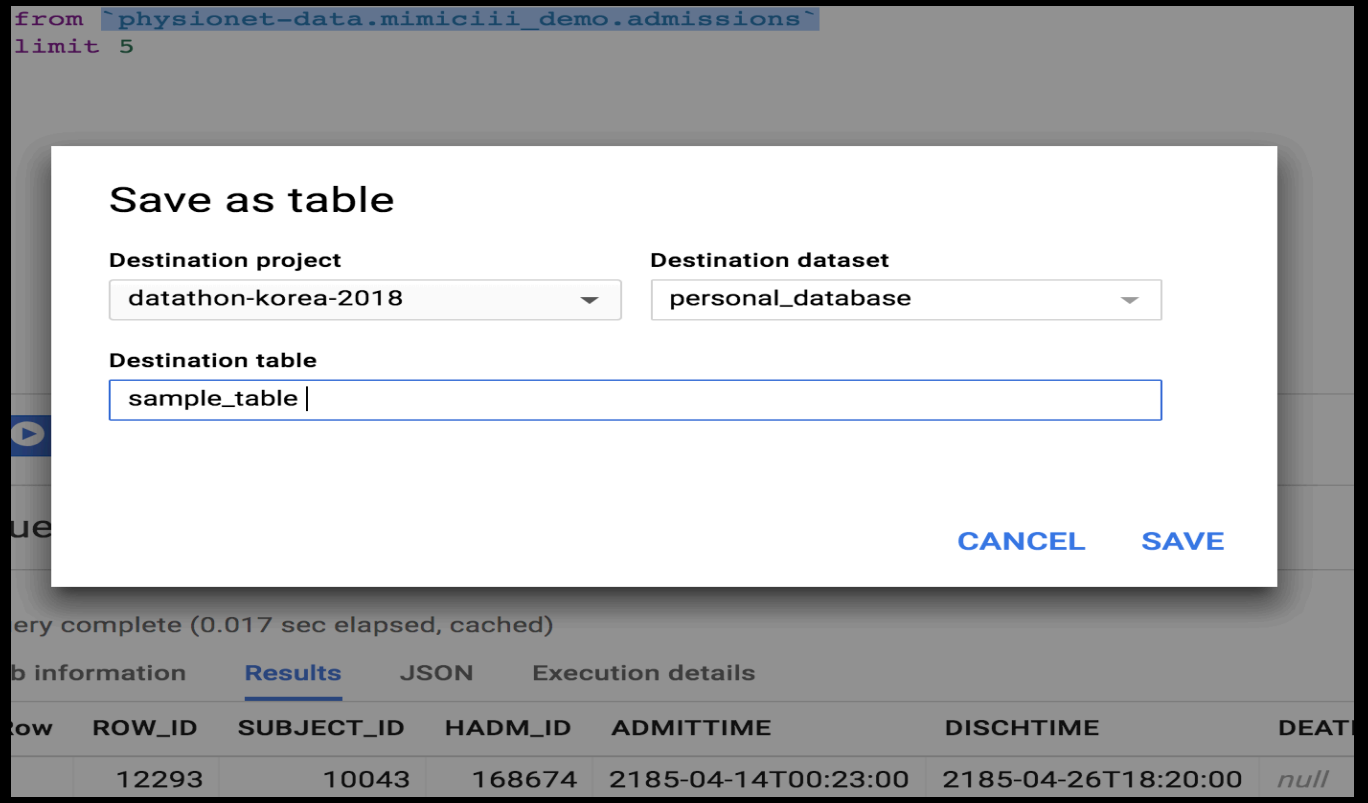
Working in BigQuery

Query editor

```
1 select *
2 from `physionet-data.mimiciii_demo.admissions`
3 limit 5
4
5
```

You must specify the source of database!

Working in BigQuery



It is also possible to save your query results in your project

Working in BigQuery

```
1 select subject_id, hadm_id, admittime, dob
2 from `physionet-data.mimiciii_demo.patients`
3 left join `datathon-korea-2018.personal_database.sample_table` using (subject_id)
4 limit 5
5
```

You can even merge your own table with mimiciii tables

Using Colab with BigQuery

- Colab is a Jupyter-like service within google cloud sever
- It is possible to query database in Bigquery
- Your are able to develop machine learning models with pre-installed machine learning packages such as Tensorflow, Sklearn and Pytorch.
- Currently only Python is supported

Using Colab with BigQuery

- Go to the link below

https://github.com/Youngseok0001/korea-datathon

Working in Colab

- Please click Python colab highlighted

README.md

Korea-datathon-2018

Welcome to the korea datathon workshop!

1. We have prepared tutorials to get you started on BigQuery, the tool to filter, join, aggregate and extract data from the raw datasets for analysis. In the tutorial, a couple of comprehensive examples are included to show how to view the datasets, run transformations and analyze them.

2. please start from the Python colab (a copy is available in the tutorials folder as well), which is a Jupyter notebook hosted in Google Drive, and can be shared with other people for collaboration. It has the most comprehensive examples, including how to train machine learning models on the MIMIC demo dataset with Tensorflow.

3. I am also uploading the answers for today's workshop.

4. Here are the list of useful links for participants who wants to have clearer understanding on MIMICiii data and SQL syntax.

- MIMICiii schema visualisation with SchemaSpy
- Sqls and models used for Mornin's Echo project
- Official MIMIC github repo for code sharing

Working in Colab

- Let us import necessary libraries first *(please remove `createtableone`library)
- Each cell can be executed by pressing Shitt+Enter

Setup

Before running any cell in this colab, please make sure there is a green check ma
button to connect to a random backend.

You can now run the following cell by clicking on the triangle button when you ho
Shift+Enter.

```
# Import libraries
import numpy as np
import tensorflow as tf
import os
import re
import pandas as pd
import matplotlib.pyplot as plt
from sklearn import linear_model


# Imports for printing out pretty pandas dataframes
from IPython.display import display, HTML

# Imports for accessing Datathon data using Google BigQuery
from google.colab import auth
from google.cloud import bigquery
```

Working in Colab

- Run the next three cosecutive cells
- You will asked to provide an authentification code
- Set your Google project environment
- Create a wrapper function for easy query

Before running any queries using BigQuery, you need to first authenticate yo
follow a link to log in using your Gmail account, and accept the data access
which you should paste back to the cell below and press enter.

```
auth.authenticate_user()
```

The data-hosting project physionet-data has read-only access, as a result, y
"**datathon-korea-2018**" where we have been testing our sql queries will be u

Note that during the actual datathon, all participants will be divided into tea

```
[ ]  project_id='datathon-korea-2018'
     os.environ["GOOGLE_CLOUD_PROJECT"]=project_id
```

Let's define a sql wrapper with our project_id configured

```
[ ]  # Read data from BigQuery into pandas dataframes.
     def run_query(query):
       return pd.io.gbq.read_gbq(query, project_id=project_
```

OK, that's it for setup, now let's get our hands on the MIMIC demo data!

Sample Queries

```
run_query("""
SELECT *
FROM `physionet-data.mimiciii_demo.patients`
WHERE subject_id = 10006
OR subject_id = 10040
OR subject_id = 10111
""")
```

	ROW_ID	SUBJECT_ID	GENDER	DOB	DOD	DOD_HOSP	DOD_SSN	EXPIRE_FLAG
0	9467	10006	F	2094-03-05	2165-08-12	2165-08-12	2165-08-12	1
1	9499	10040	F	2061-10-23	2150-09-05	2150-09-05	2150-09-05	1
2	9566	10111	F	2097-01-16	2180-02-01	2180-02-01	2180-02-01	1

- This query selects all columns from the patients table where subject_id is either 10006, 10040 or 10111.
- Since patients table is a table which contains a list of unique subject_id and their basic informations,it will only output 3 observations.

Sample Queries

```
run_query("""
with `icu-adm` as
(
select *
from `physionet-data.mimiciii_demo.icustays`
left join `physionet-data.mimiciii_demo.admissions` using (subject_id, hadm_id)
),

`icu-adm-pat` as
(
select *
from `icu-adm`
left join `physionet-data.mimiciii_demo.patients` using (subject_id)
)

select icustay_id, hadm_id, subject_id, dob, admittime, dischtime , intime, outtime,ethnicity, gender
from `icu-adm-pat`
limit 5


""")
```

	icustay_id	hadm_id	subject_id	dob	admittime	dischtime	intime	outtime	ethnicity	gender
0	204881	199207	10017	2075-09-21	2149-05-26 17:19:00	2149-06-03 18:42:00	2149-05-29 18:52:29	2149-05-31 22:19:17	WHITE	F
1	295043	170883	10124	2108-12-20	2192-04-16 20:57:00	2192-05-15 19:28:00	2192-04-24 02:29:49	2192-04-26 23:59:45	WHITE	F
2	293429	168803	40503	2097-11-14	2186-07-06 19:59:00	2186-07-07 19:00:00	2186-07-06 19:59:36	2186-07-07 20:48:07	WHITE	F
3	279529	182879	40601	2112-01-20	2184-08-04 05:44:00	2184-08-10 15:30:00	2184-08-04 05:45:07	2184-08-06 17:26:43	WHITE	F
4	249695	168233	10088	2029-07-09	2107-01-29 04:00:00	2107-02-10 12:00:00	2107-01-29 04:02:15	2107-01-30 18:58:45	WHITE	M

- This query merges the three core tables, “patietns”, “admissions” and “icustays”
- Note that intime always happens later than admittime

Sample Queries

```
##1
df = run_query("""
      SELECT icustay_id, max(valuenum) as HeartRate_Max
      FROM `physionet-data.mimiciii_demo.chartevents`
      WHERE itemid = 211
      GROUP BY icustay_id;

      """)
df.head()
```

	icustay_id	HeartRate_Max
0	248755	76.0
1	234989	88.0
2	228977	92.0
3	258147	96.0
4	243600	96.0

- You will often need to aggregate values over each icustay_id.
- The query above uses the “group_by” cluase to get the max HeartRate of each ICU case.

Sample Queries

```
age_raw as
(
  select hadm_id,
         datetime_diff(admittime ,dob, day)/365 as crude_age
  from `physionet-data.mimiciii_demo.patients`
  left join `physionet-data.mimiciii_demo.admissions` using (subject_id)
),

age as
(
  select  hadm_id,
          case when  crude_age  > 91.5 then 91.5 else
                     crude_age  end age
  from age_raw
)
```

- It is very critical to note that patients with their age above 91.5 are de-identified by shifting their date of birth. You will therefore often see patients to their age well above 200. Please set their age to 91.5

Sample Queries

```
select subject_id, hadm_id, age, icd9_code, short_title
from (
select cohort.subject_id ,cohort.hadm_id, age.age, primary_admit_diag.icd9_code, short_title,
       row_number() over (partition by cohort.subject_id order by age.age asc) row_num
from cohort
left join age using (hadm_id)
left join primary_admit_diag using (hadm_id)
left join `physionet-data.mimiciii_demo.d_icd_diagnoses` using (icd9_code)
    )
where row_num = 1
order by subject_id desc
```

- If your interest is only getting the patients' primary diagnoses from their first hospital admission, use row_number or rank function to partition subject_id and order them with either their age of admittime.

Sample Queries

```
select subject_id, hadm_id, age, icd9_code, short_title
from (
select cohort.subject_id ,cohort.hadm_id, age.age, primary_admit_diag.icd9_code, short_title,
       row_number() over (partition by cohort.subject_id order by age.age asc) row_num
from cohort
left join age using (hadm_id)
left join primary_admit_diag using (hadm_id)
left join `physionet-data.mimiciii_demo.d_icd_diagnoses` using (icd9_code)
    )
where row_num = 1
order by subject_id desc
```

- If your interest is only getting the patients' primary diagnoses from their first hospital admission, use row_number or rank function to partition subject_id and order them with either their age of admittime.

General Workflow

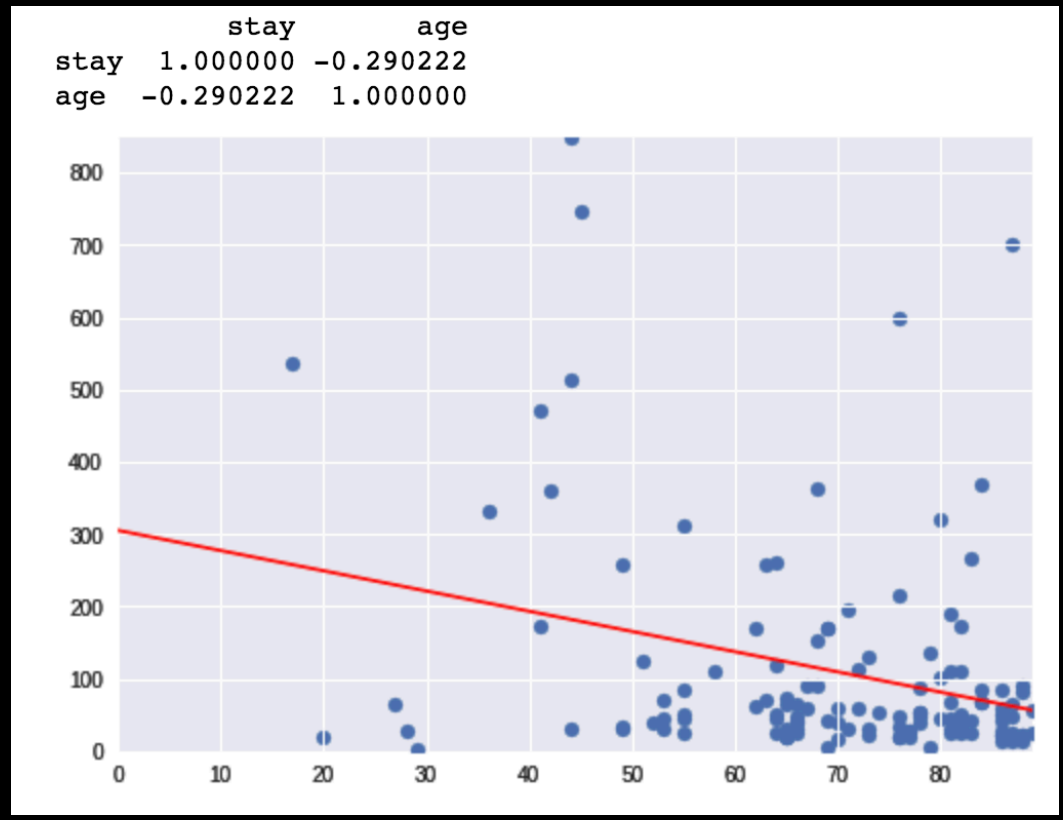
- The General workflow with MIMICiii data is as follows
  1. Cohort selection
  2. Diagnosis
  3. Labtest Results
  4. VitalSign
  5. Merge all the tables above

- The reason for having features from various tables is to adjust for possible confounders.
- Please refer to my python script for more details.

Modeling: linear regression

- One possible relation that we will look for is if there is any +ve or –ve correlation between patients' age and their length of stay in ICU
- It can actually be observed that there is a slight relationship
- However they are very week, evident from their correlation matrix.

	stay	age
stay	1.000000	-0.290222
age	-0.290222	1.000000



Modeling:Logistic regressions

- As shown from the previous plot, It may not actually be suitable to use linear regression as our model because most patients are old in general and thus not much data point is present in younger cohort.
- It is therefore not suitable to conclude that they actually follow a liear relationship(It could be non-linear)
- Using logistic regression might be a safer choice in this case.
- The example given from the script set cases with the length of stay < 1 day as short and long for >1 day
- Logistic regression was then used to calculate the OR of staying longer for every unit(year) incerase in age.