

**ADTA 5240**  
**Harvesting, Storing, and Retrieving Data**  
**Final Project**

**Healthcare Monitoring System**

**Team**

|  |  |  |
| --- | --- | --- |
| **Name** | **ID** | **Email** |
| Naga Sai Sivani, Tutika | 11703058 | NagaSaiSivaniTutika@my.unt.edu |
| Niharika, Madhadi | 11659474 | Niharikamadhadi@my.unt.edu |
| Sujit Rayudu | 11708299 | SujitRayudu@my.unt.edu |
| Binod Nepal | 11271123 | Binodnepal@my.unt.edu |

**Table of Contents**

[Project Management 2](#_Toc412644203)

[Data Collection & Ingestion 3](#_Toc480151308)

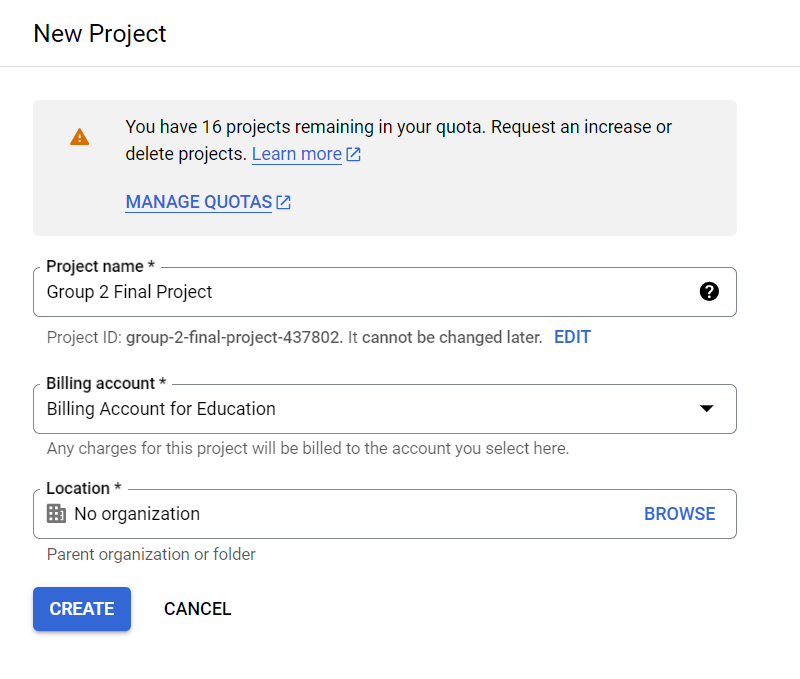
[Data Architecture 15](#_Toc71254706)

[Data Processing 21](#_Toc1225924890)

[Data Analysis & Visualizations 27](#_Toc1066998451)

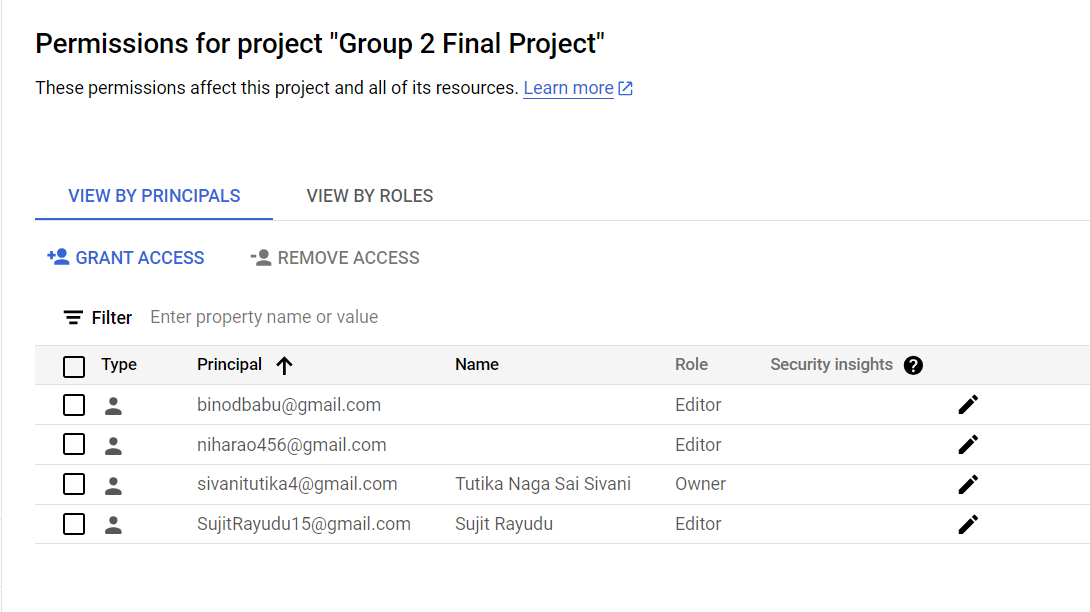
# **Project Management**

1. Creating GCP project, assigning roles and establishing infrastructure.



*fig1. project creation*

We created a new project, to build setup infrastructure, storage and analytics using GCP.



*fig2. assigning roles*

Next, we assigned roles to each member using GCP IAM Admin. We made sure to implement granularity, so that each member can access only what they absolutely need.

# **Data Collection & Ingestion**

1. Dataset Selection

We went through many datasets, everyone contributed in the research and finally we settled with two sources of data for static and streaming datasets.

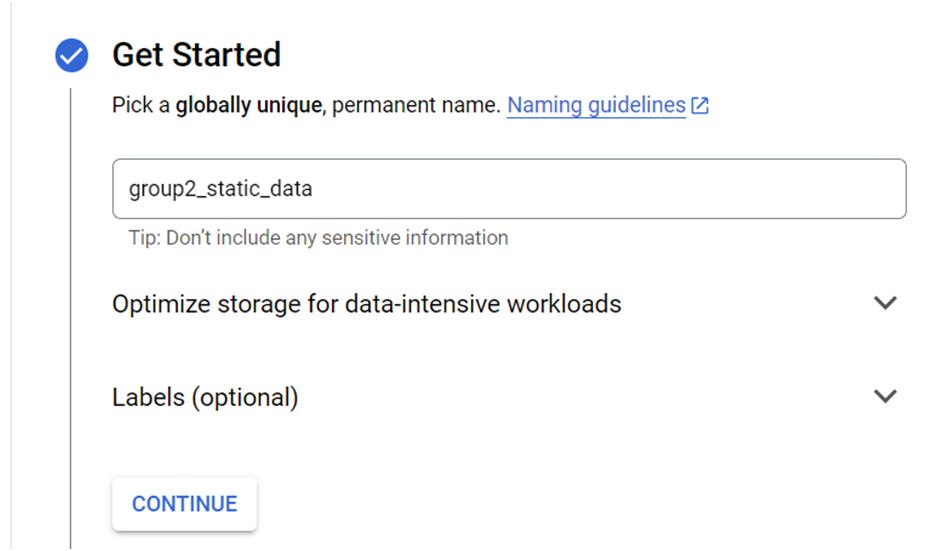
* For static data we selected two data files periodic\_vitals and aperiodic\_vitals from eICU crd (eICU Collaborative Research Database demo).

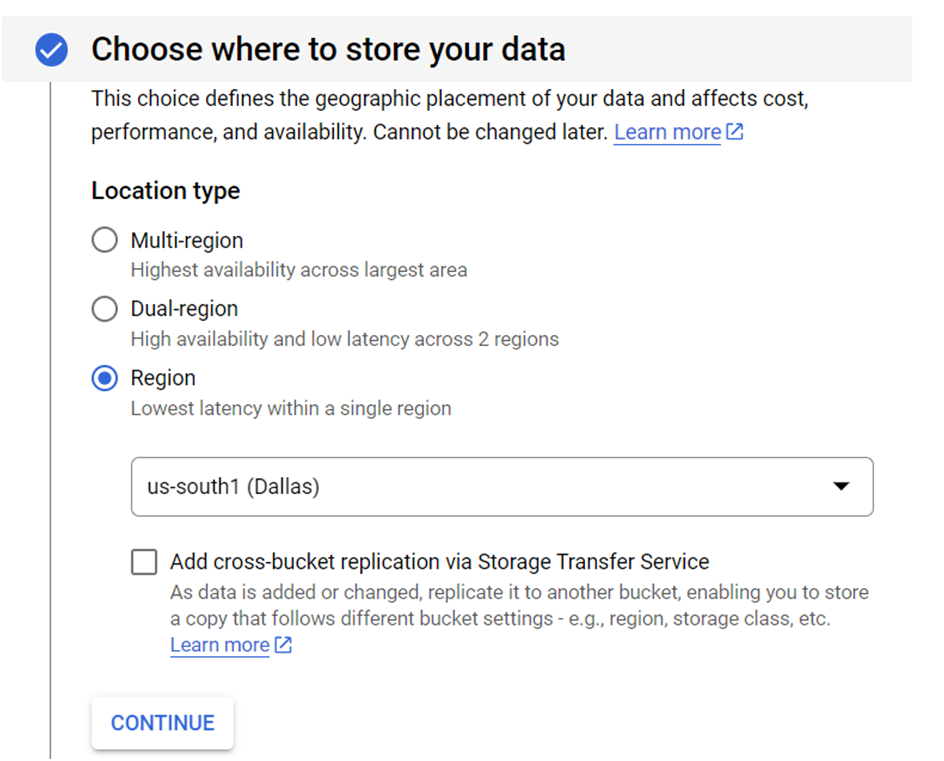
Source: <https://physionet.org/content/eicu-crd-demo/2.0.1/>

* For dynamic data we selected VitalDB cases openDataset webAPI to collect streaming data.

Source: <https://vitaldb.net/dataset/?query=api>

1. Setting up Data Infrastructure



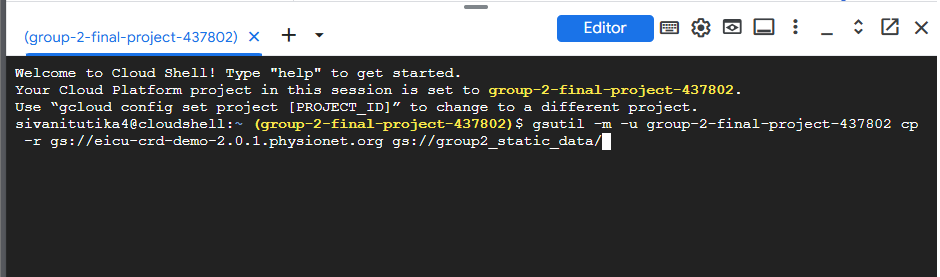
 

*fig. 3,4,5 Creating Google Cloud Storage Bucket*

We created a Google Cloud Storage Buckets, to store our static and streaming data. We created various GCS buckets for various stages of our data.

1. Static Data Ingestion

We first tried to data ingestion by downloading the dataset on our device and then upload to the GCS bucket, but as the data was huge we were not able to download. So we then tried to ingest data directly into GCS using GCP Cloud shell.

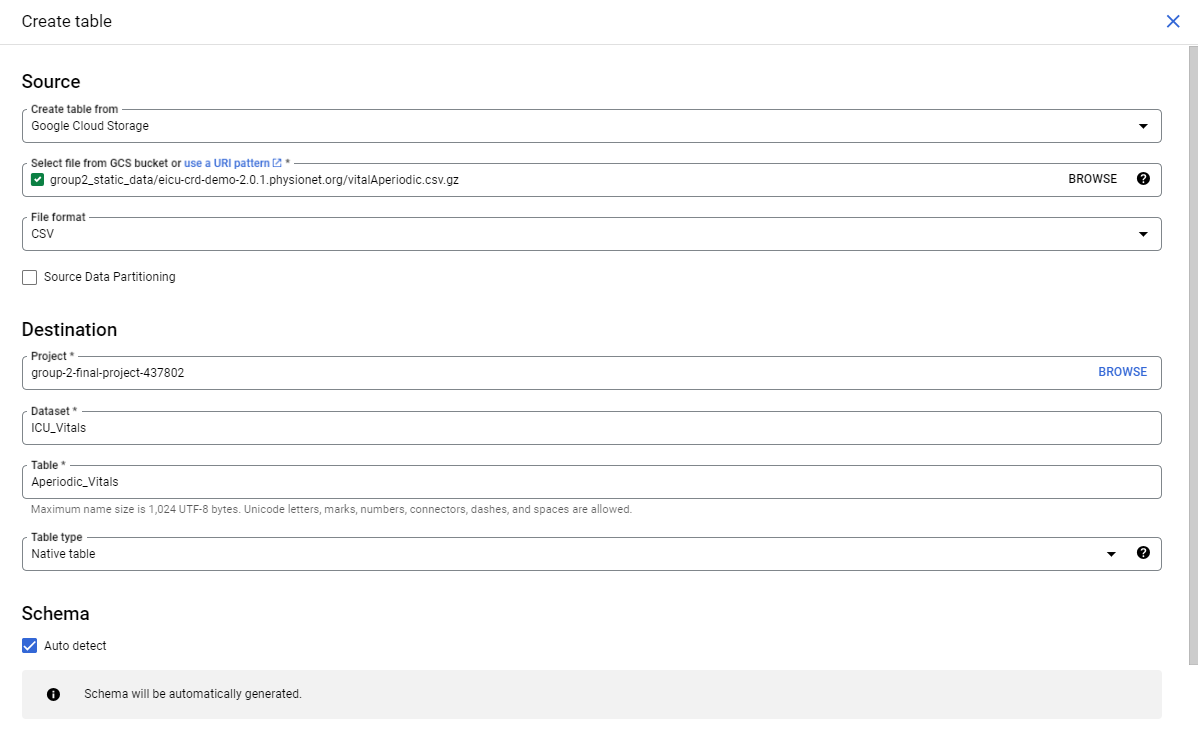
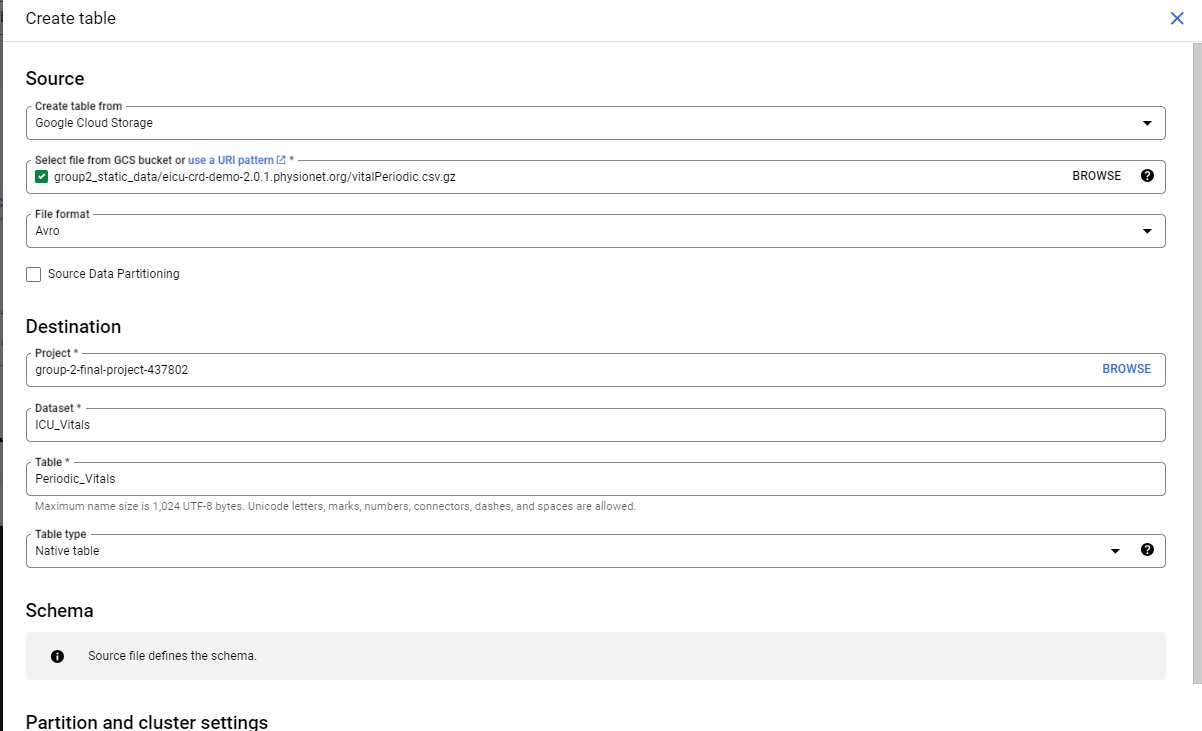
*fig. 6 Static data ingestion using gsutil*

*fig. 7 eicu-crd data in GCS bucket*

We uploaded static data using cloud shell and gsutil directly into our group2\_static\_data bucket to skip downloading data to our local device.

**Command used:**

gsutil -m -u group-2-final-project-437802 cp -r gs://eicu-crd-demo-2.0.1.physionet.org gs://group2\_static\_data/

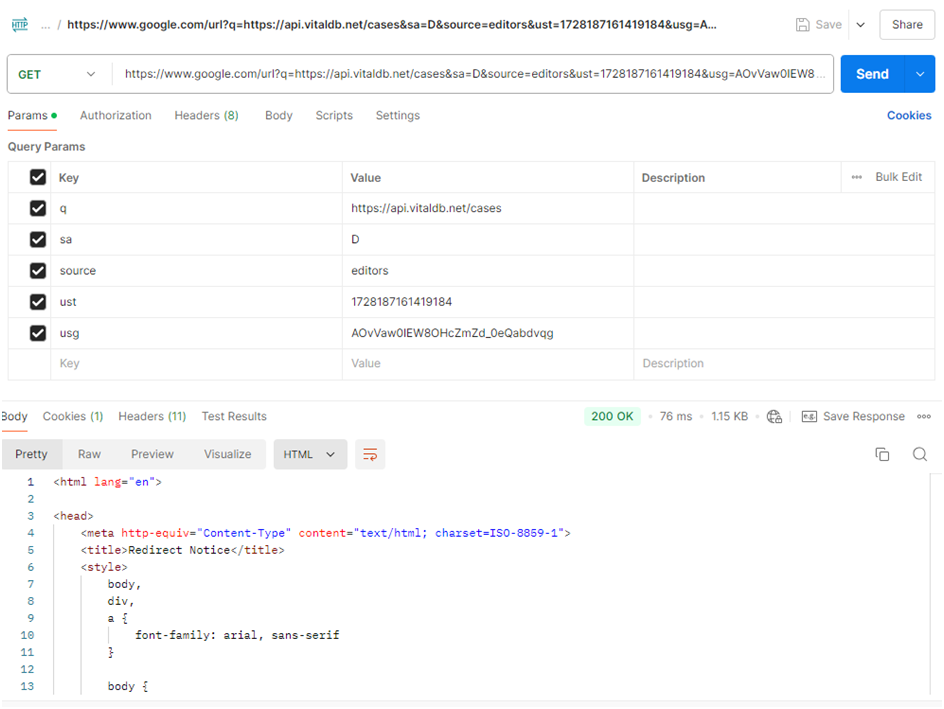


*fig. 8 Created datasets & tables in BigQuery*

Next, to use the data in BigQuery we had to import our csv files from storage buckets to BigQuery tables. So, we created a dataset ICU\_vitals to store our data into two table periodic\_vitals and apriodic\_vitals.

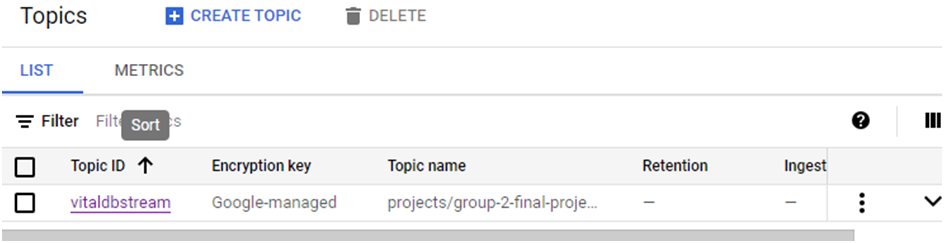
1. Streaming Data Ingestion

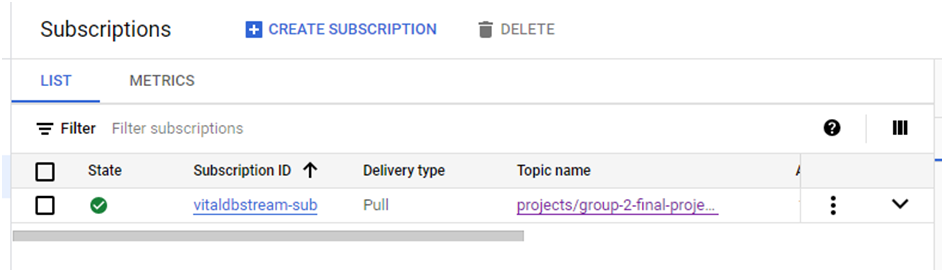
For streaming data, as mentioned above we selected vitalDB open dataset WebAPI where we studied about vitals of ICU cases.



*fig 9. Testing API using Postman*

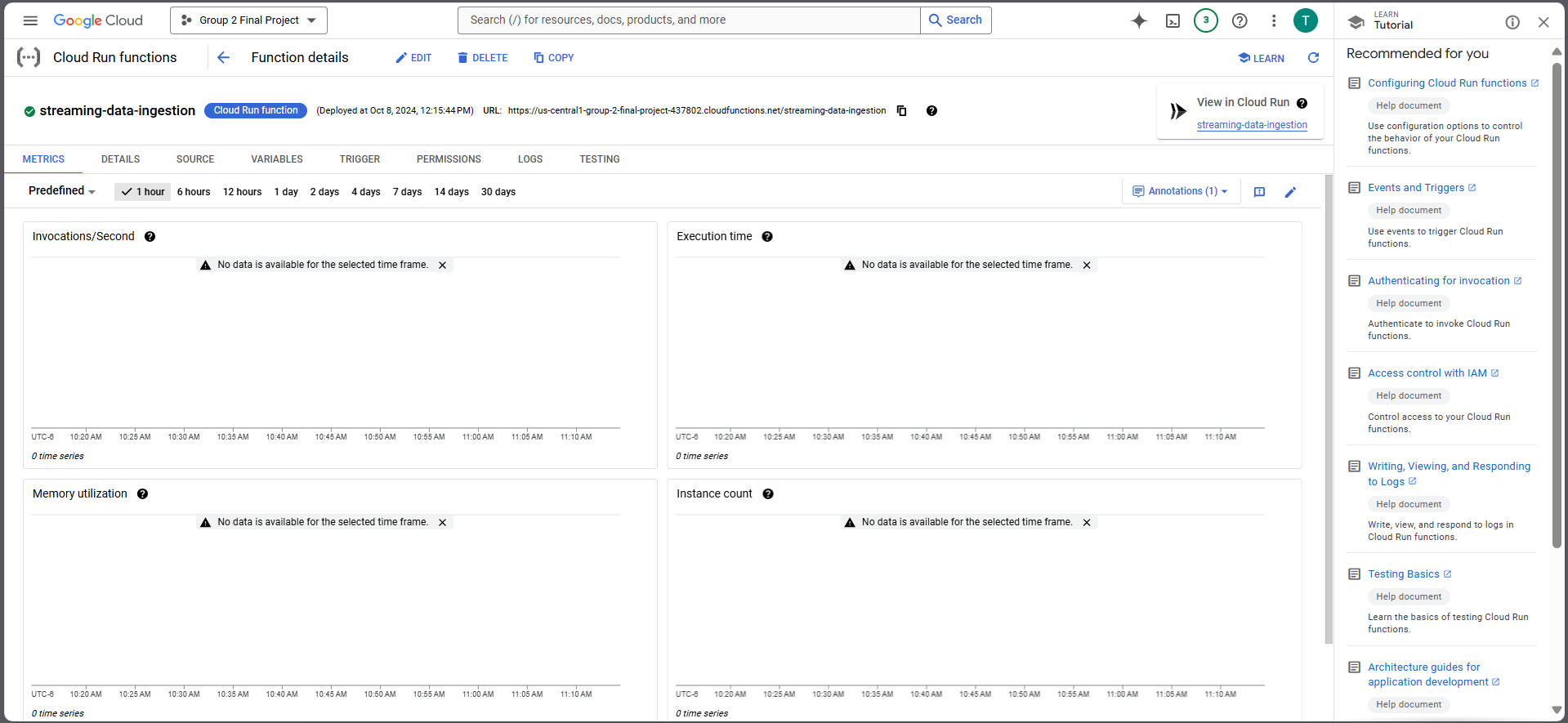
We first tested our API using postman to make sure the API is working and the data is favourable to our project.



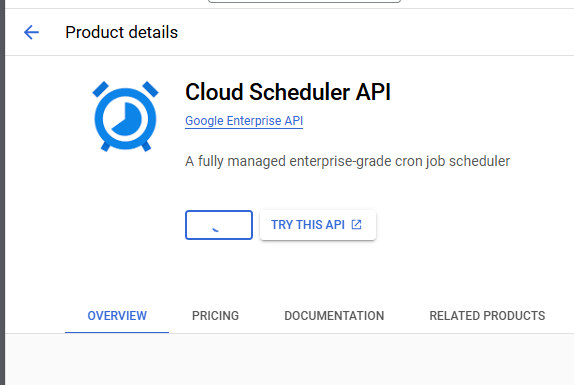


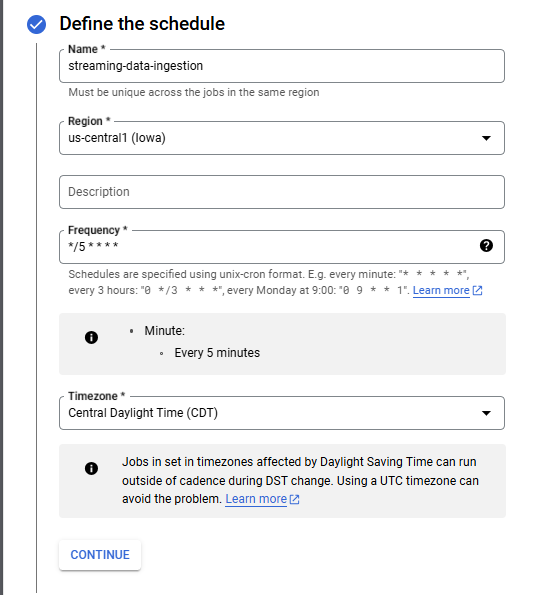
*fig. 10,11 GCP Pub/Sub service creation*

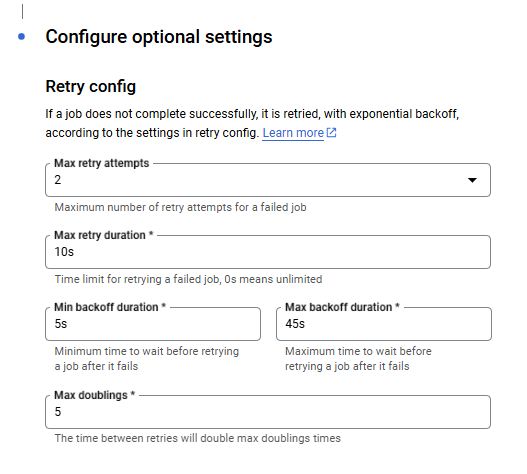
We created a Pub/Sub topic to pull data from the API with a trigger. The trigger was configured using to pull data on a daily basis.

*fig. 12 Cloud Run function Configuration*

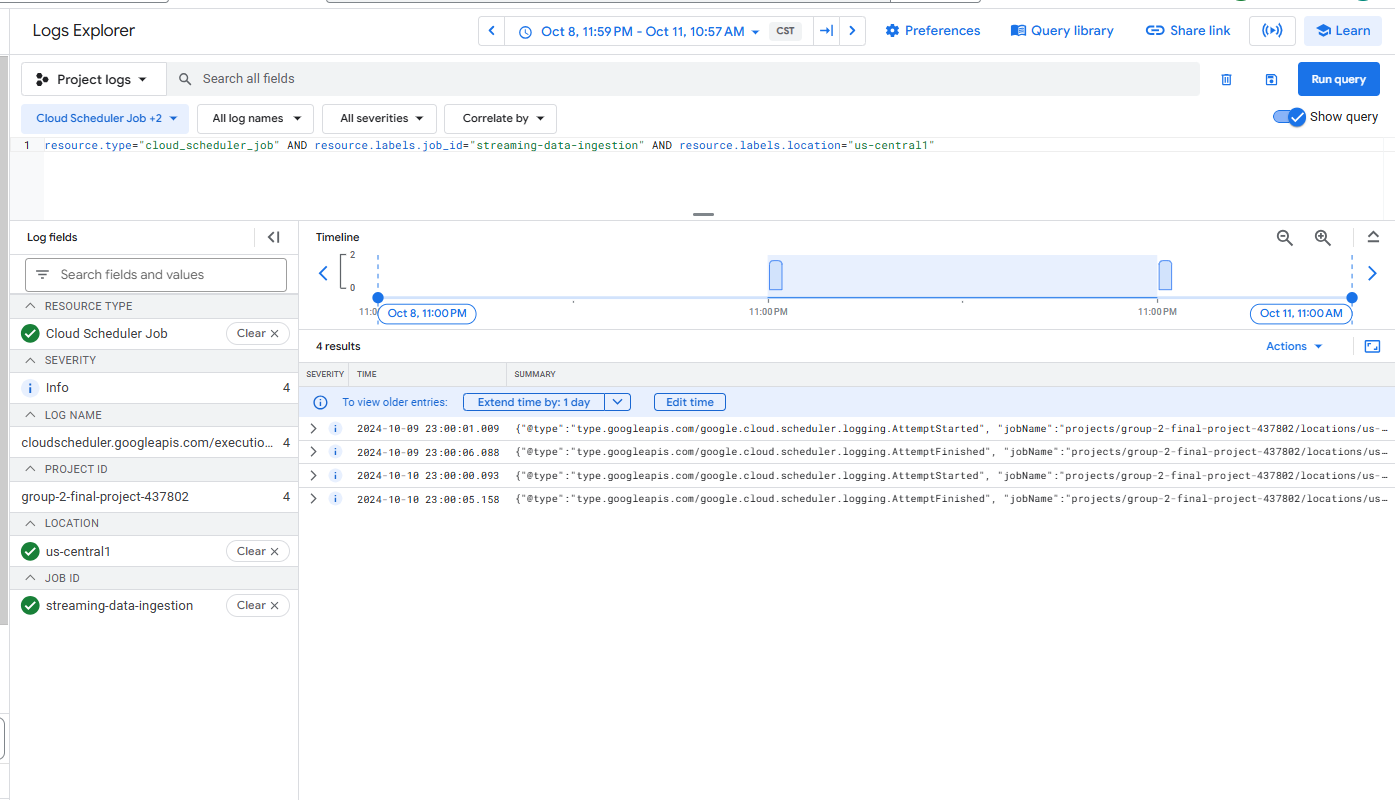
The cloud run function configured with python code helps in pull data at our command from the API and storing it in GCS bucket.



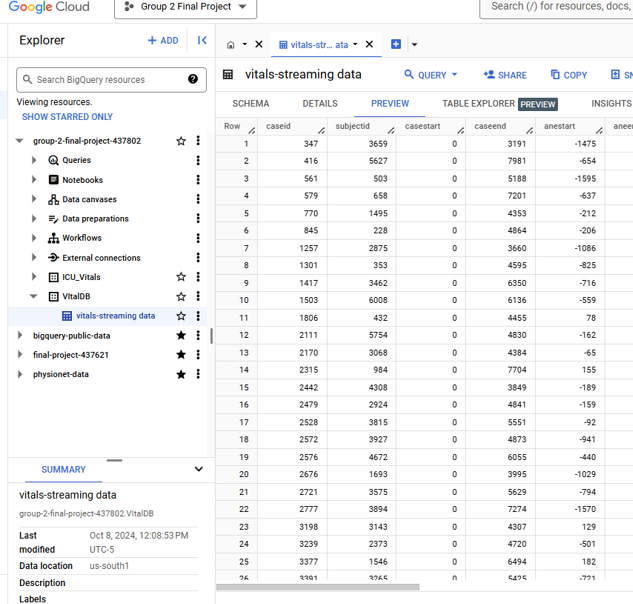




*fig. 13, 14, 15 Data schedular configurations*

To automate ingestion of streaming data we used Cloud Scheduler API that ingests data into Google Cloud Storage every night at 12:00AM. The following are the successful ingestion of data into our storage bucket for 3 continuous days. fig 16. Cloud Schedular logs

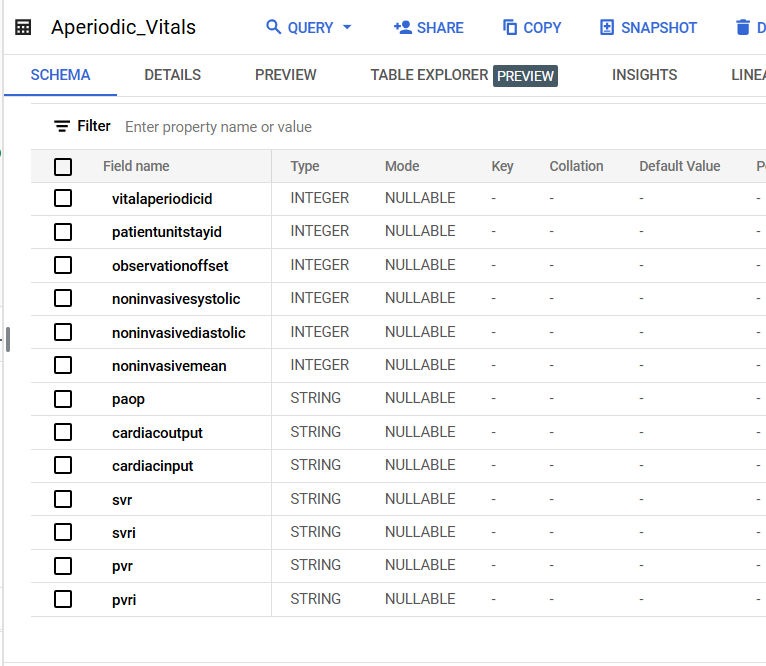
The above dashboard shows the successful streaming of data, that can be used to track real-time health monitoring systems data in Hospitals and nursing homes.

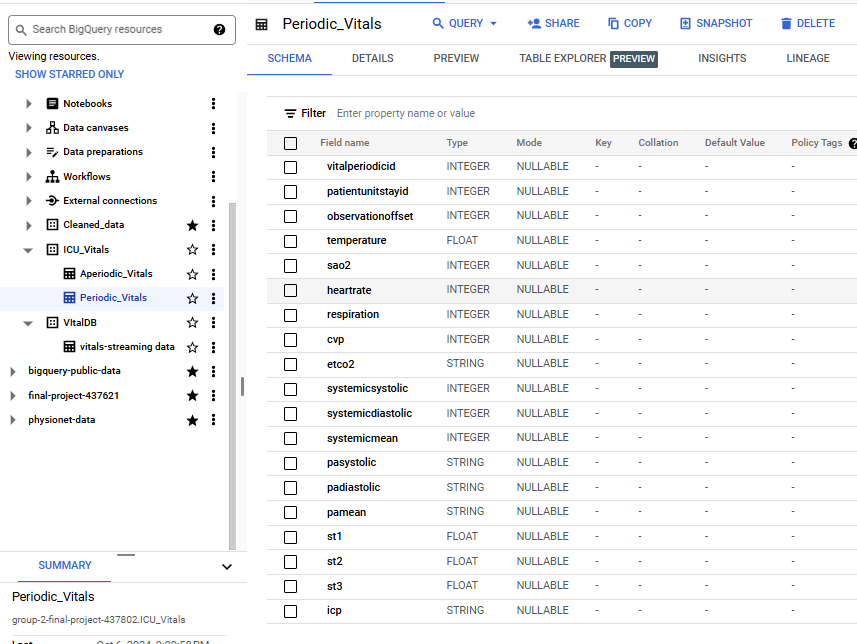
*fig. 17 Streaming data in BigQuery*

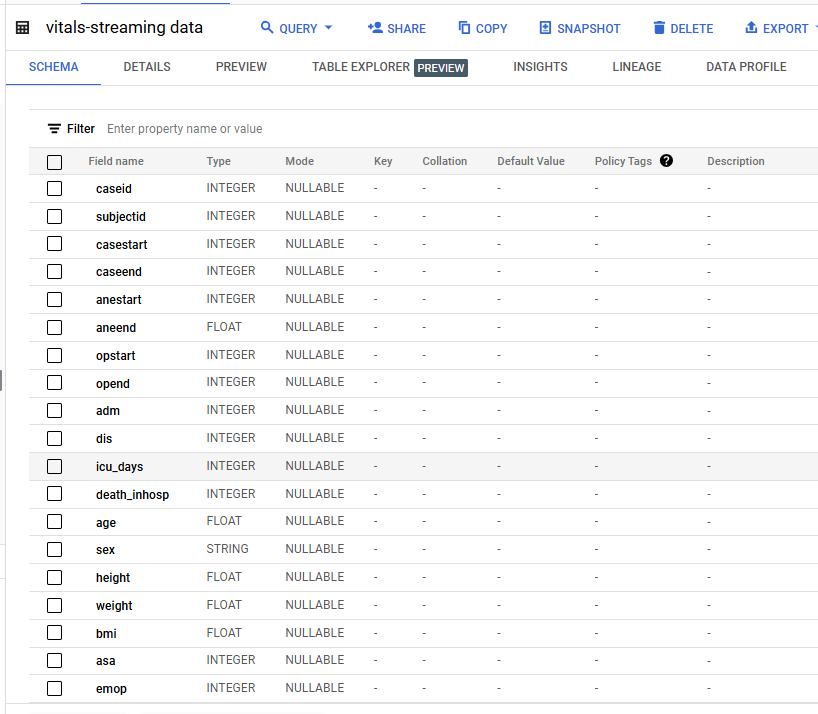
Similar to static data, we created dataset and table for storing streaming data in BigQuery for processing.

**Schema**

The following are schemas of our raw data before cleaning and pre-processing.

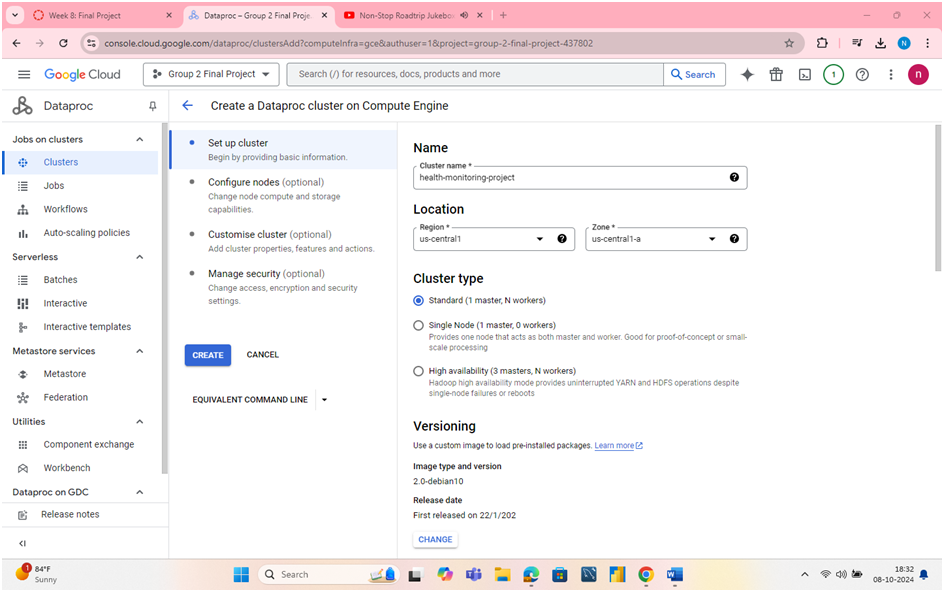
*fig. 18 Aperiodic table schema*

*fig. 19 Periodic table schema*

*fig. 20 VitalDB streaming data schema*

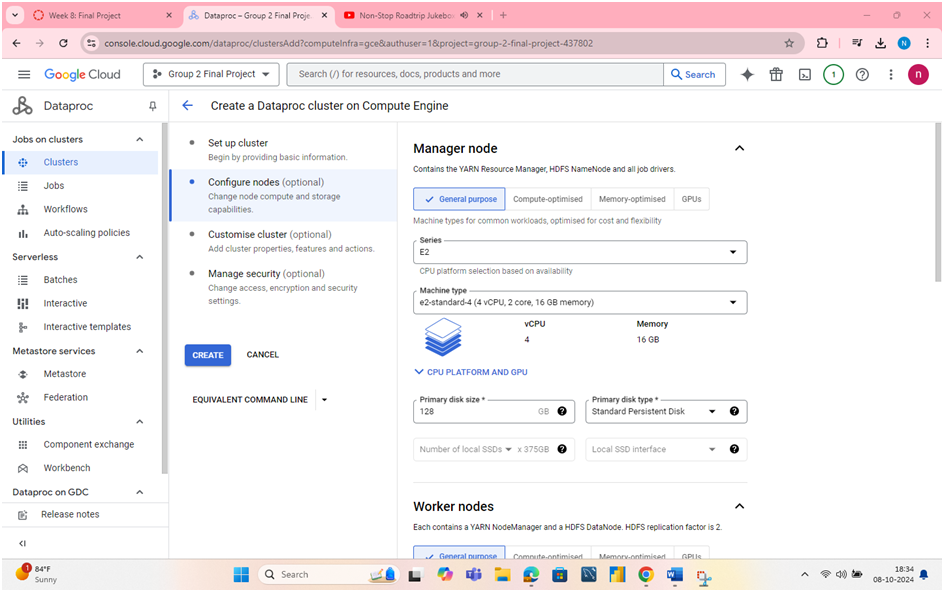
Above are the schemas of the data we are working with.

# **Data Architecture**

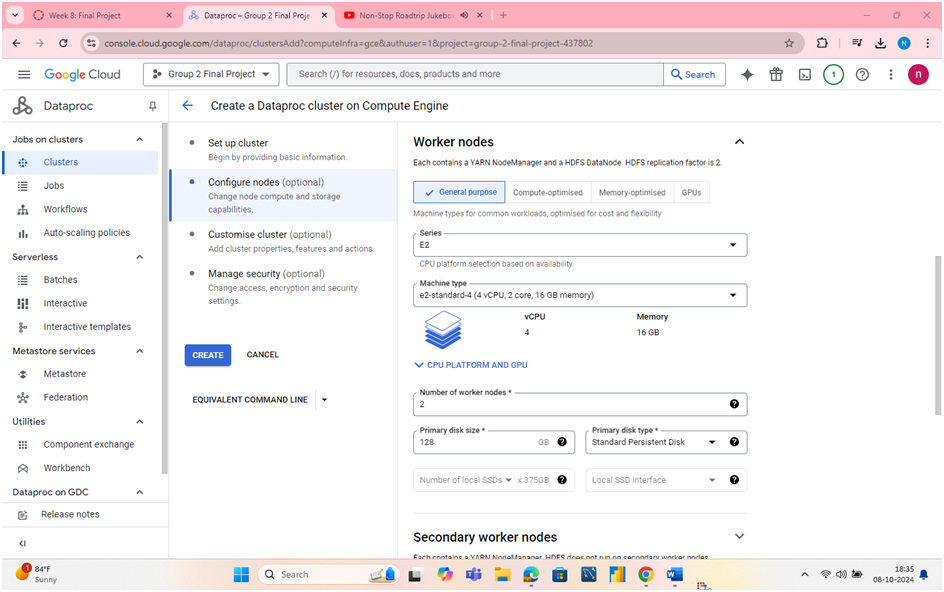


*fig. 21 Dataproc cluster creation*

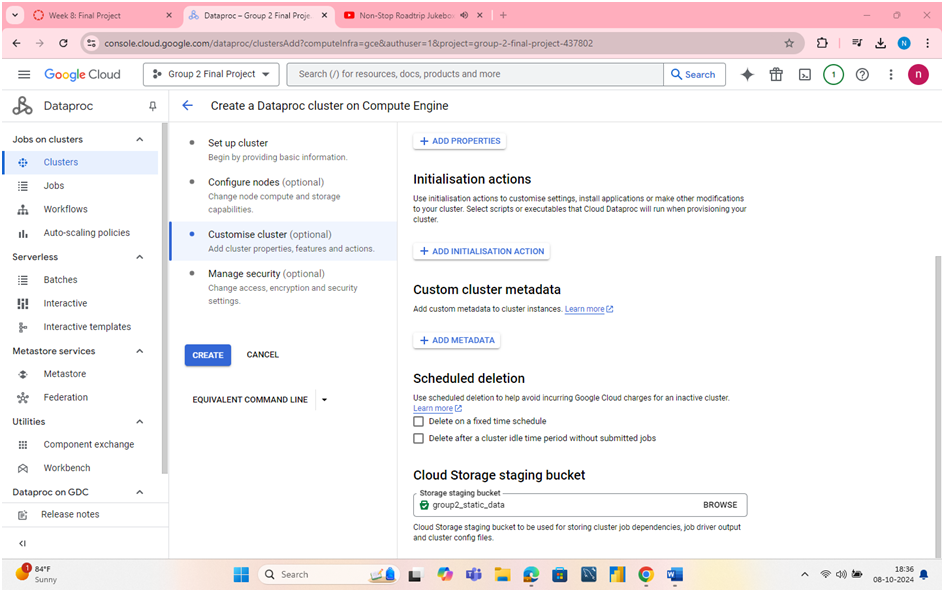
Created a Dataproc cluster, its manager and worker nodes on the compute engine to use HIVE to store and process CSV data as tables with HIVE.



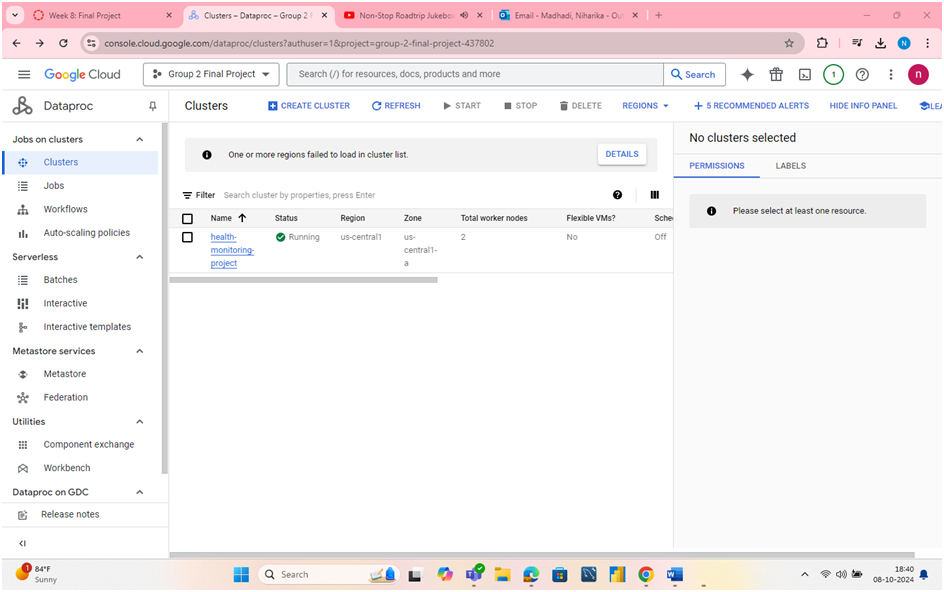
*fig. 22 Dataproc Manager Node configuration*



*fig. 23 Dataproc Worker node configuration*



*fig. 24 Dataproc cluster & Cloud Storage Bucket Integration*



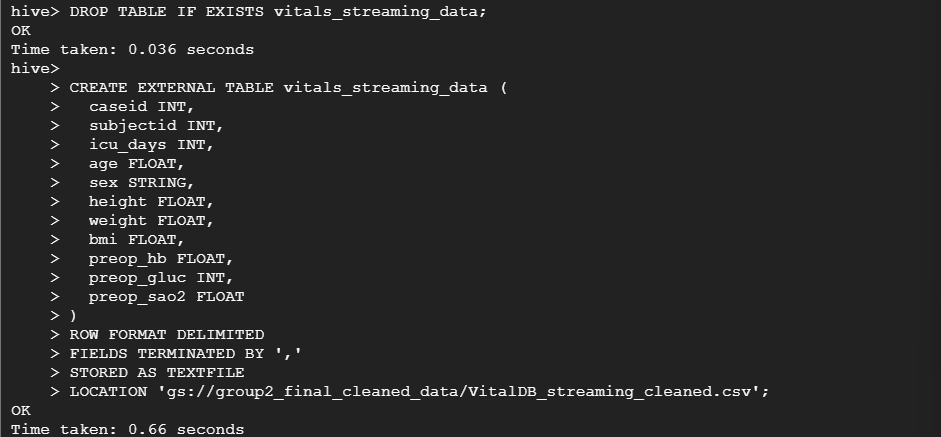
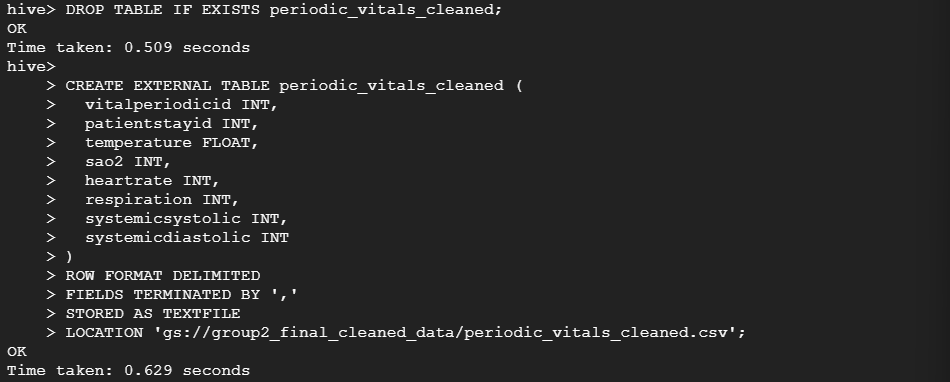
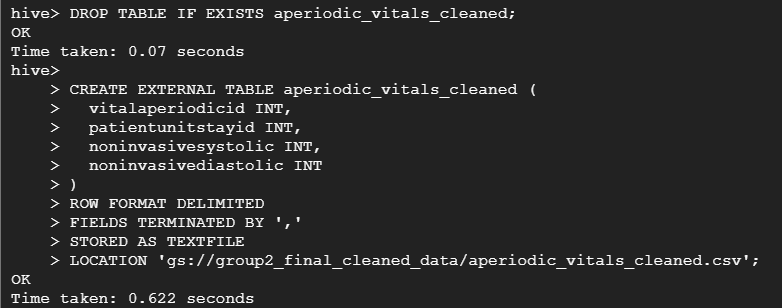
*fig. 25 Running Dataproc cluster*



*fig. 26 All VM Instances are live*

The above screenshots show creation & configuration of dataproc clusters. We were able to create and run them successfully, next we used them using SSH shell to run HIVE Queries to form tables from CSV files.

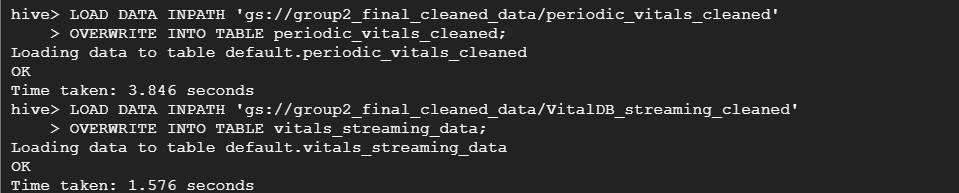
Hive

fig. 27, 28, 29 Creating HIVE tables from CSV files

We did some subject study and eliminated some features and retained only some critical features of patients to analyze. We included only those features in these tables.

Inserting data from csv files

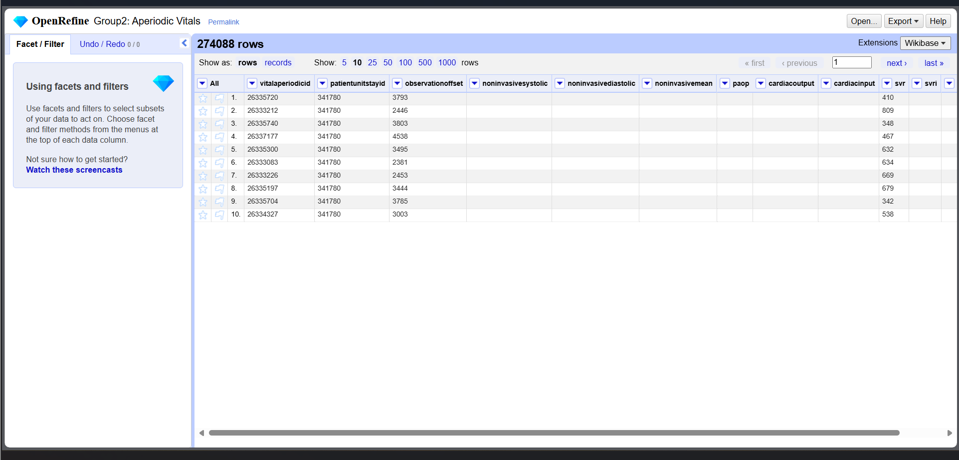


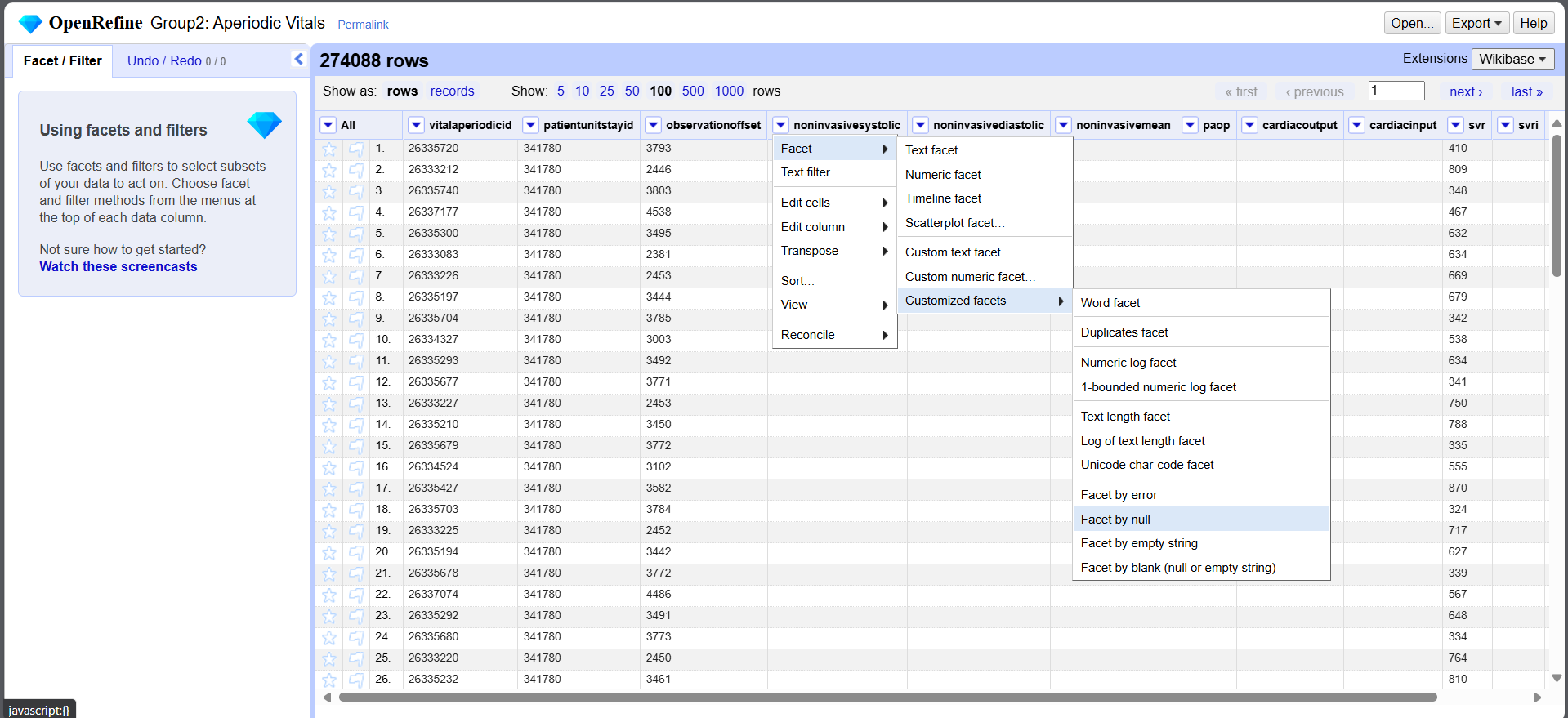
*fig. 30, 31 Inserting data into HIVE tables*

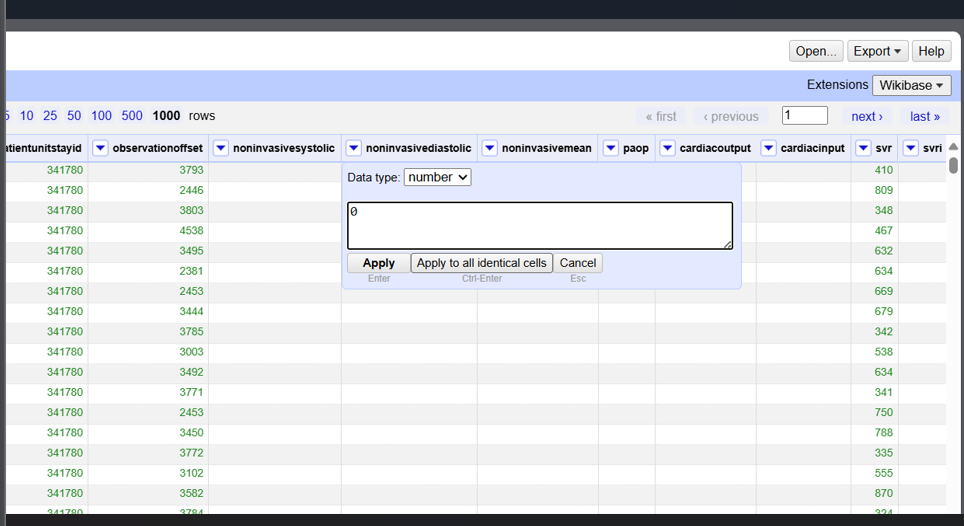
Next, we inserted our CSV data stored in Google Cloud Storage buckets into the HIVE tables.

# **Data Processing**

1. Data processing using OpenRefine



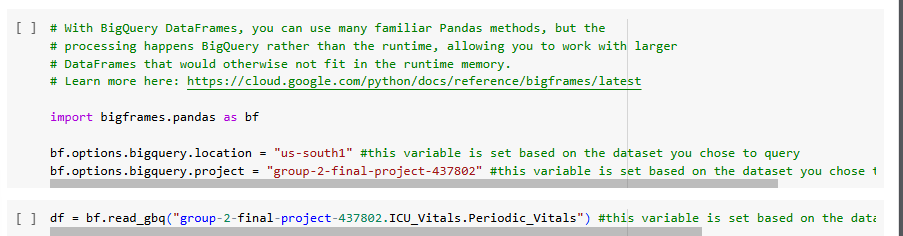
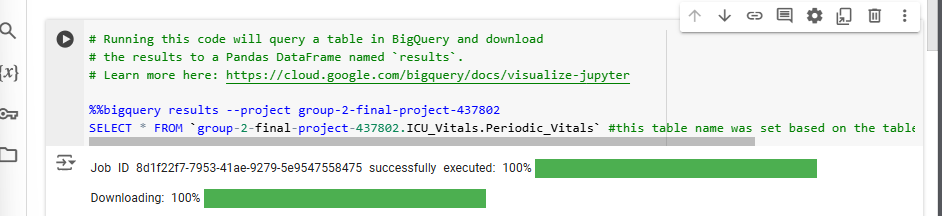


*fig. 32, 33, 34 Data processing*

For aperiodic\_vitals we used OpenRefine to replace Null Values, change each feature to it’s relevant datatype.

We were not able to use OpenRefine for other datasets as it was limited by 1GB memory.

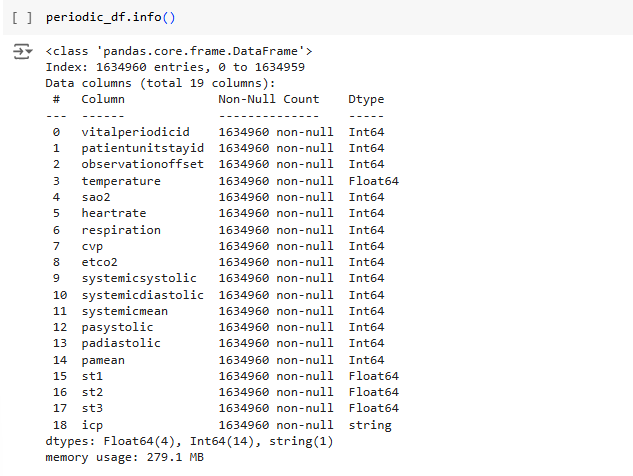
1. Data processing using BigQuery library and Google Colab.

fig. 35, 36 Loading data from BigQuery to Google Colab

We used bigquery commands to import data from our project to Google Colab, then converted them into pandas dataframe as it’s easy to work with.

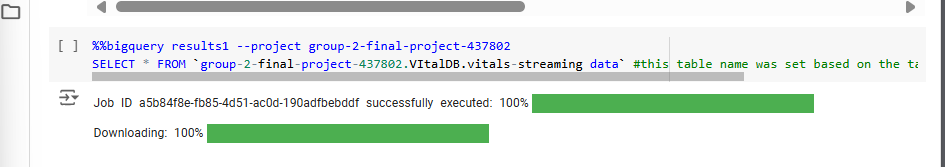
fig 37. Before processing

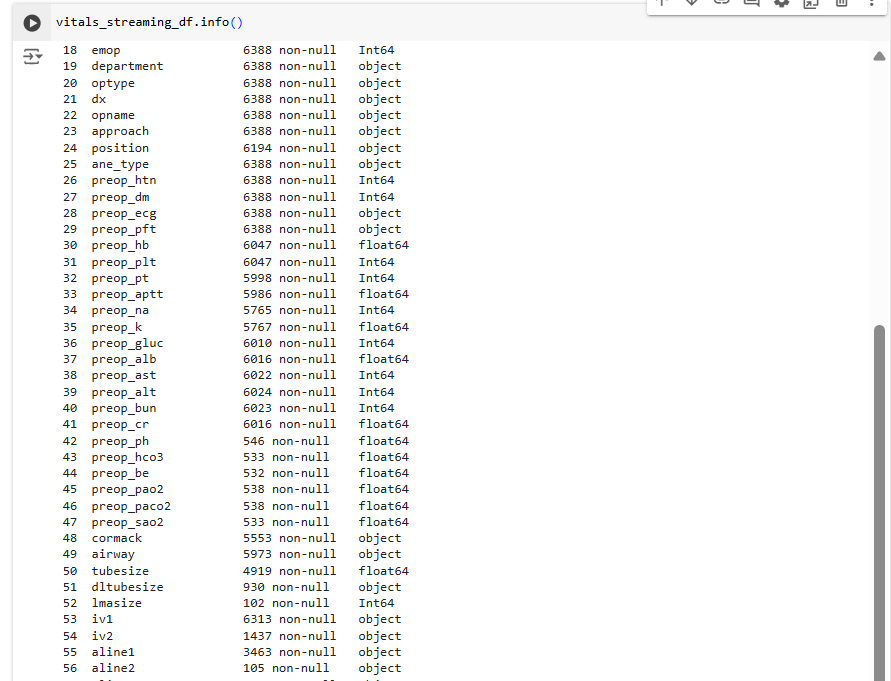
We can see that there are many null values, that we cannot process, so our main task was to handle these NULL values and also many parameters like systemicsystolic are numeric but are stored as string. So we need to change these features datatypes.

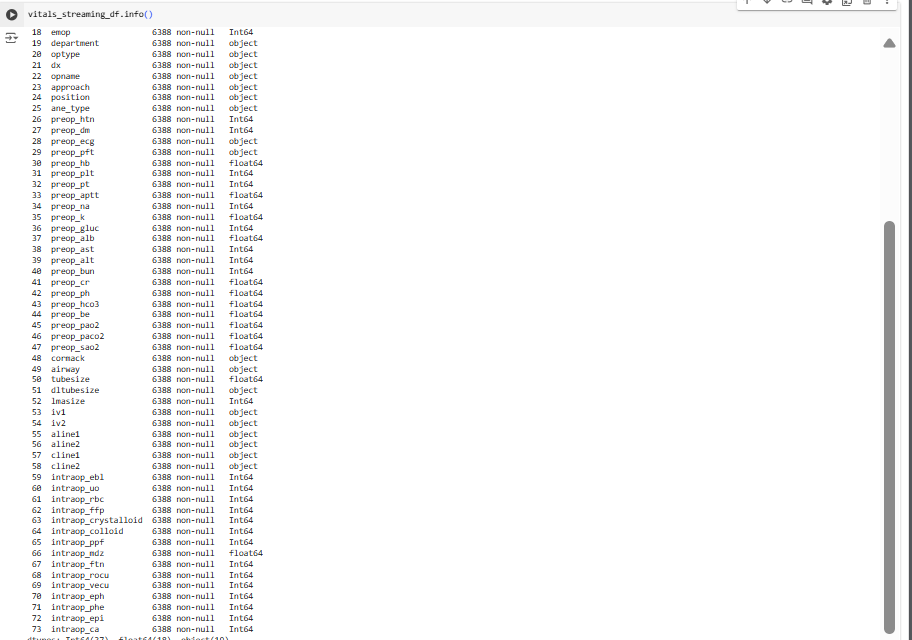
*fig. 38 After processing*

We can see there are no NULL values, and data types have been changed to relevant data types.

Processing VitalDb

fig. 39 Loading VitalDB Streaming data

fig. 40 VitalDB data before processing

fig. 41 VitalDB data after processing

Converted all features to relevant data types and filled null values.

fig. 42 Writing files back to BigQuery

We have used Google Colab only for processing, as pandas is easier to work with huge data, and we were able to overcome our data limitation from OpenRefine. After processing, the cleaned and processed data was written back to Google Cloud Storage Bucket.

# **Data Analysis & Visualizations**

Analysis using BigQuery

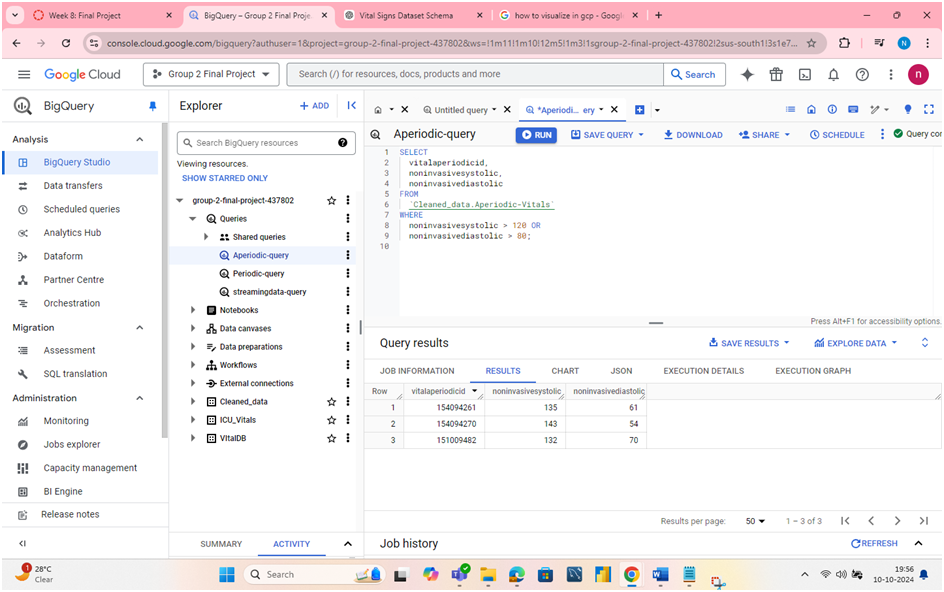


fig. 43 Query to find patients with hypertension

This Query helps us to identify patients in ICU with high blood pressure, helping doctors and nurses to act on situation.

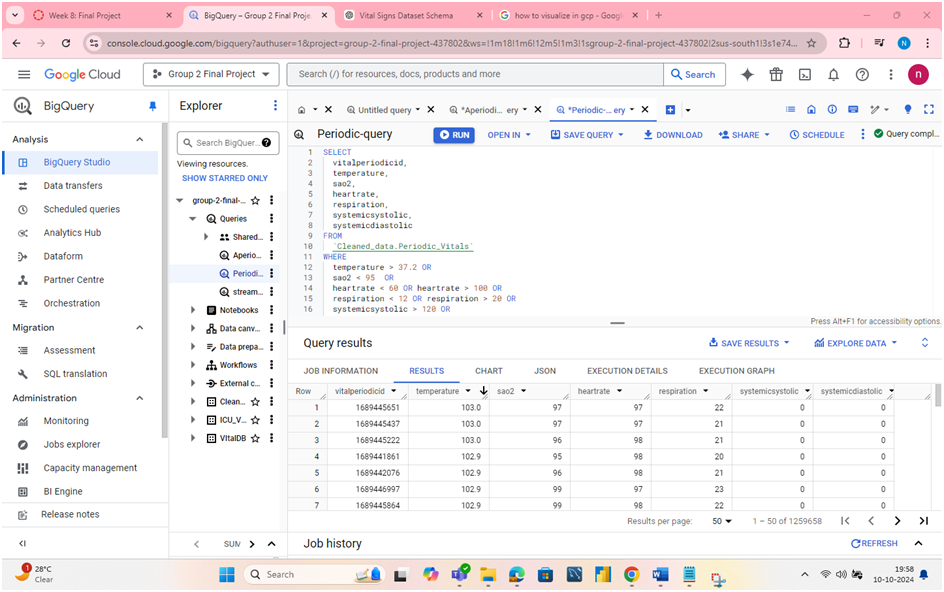


fig. 44 Query to find patients with abnormal conditions

This query helps us to identify patients with abnormal monitor readings on different parameters.

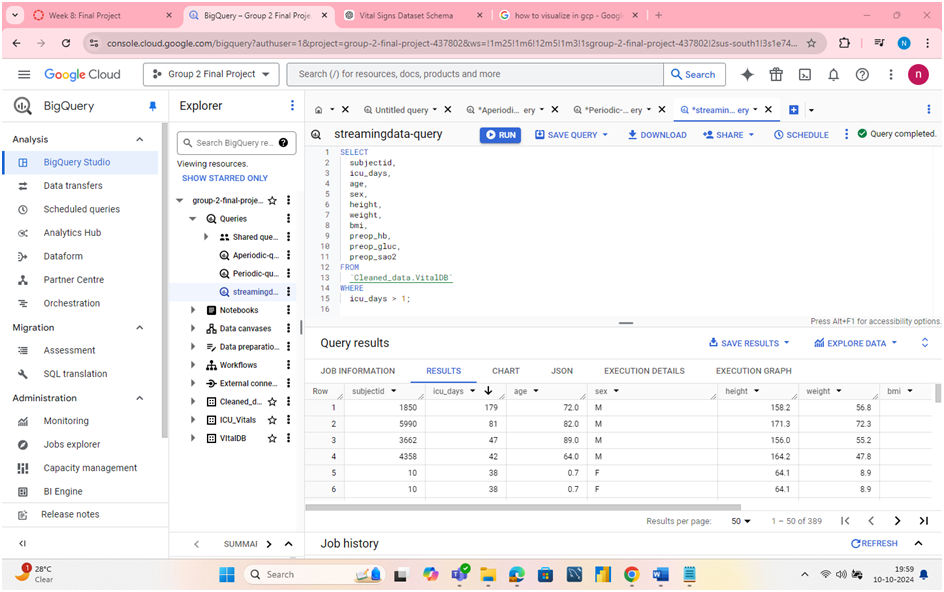


fig. 45 Query to see vitals of patients who are there in ICU for more than one day.

Helps to track patients who are in ICU for more than one day, indicating serious conditions needing attention.

Data Visualizations

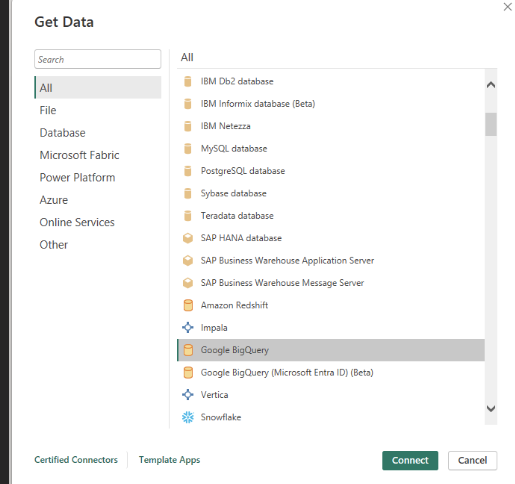
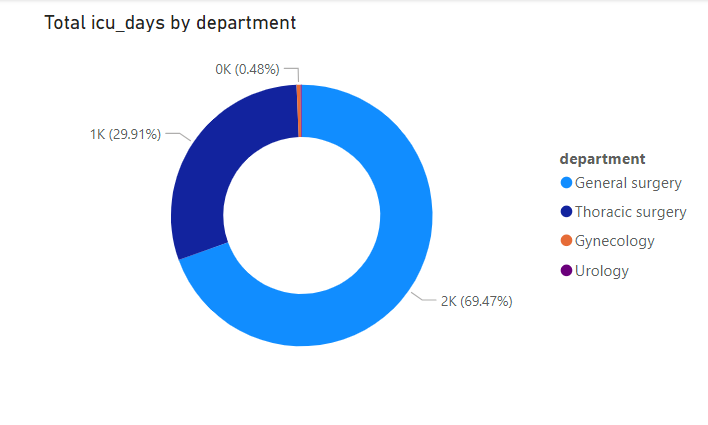
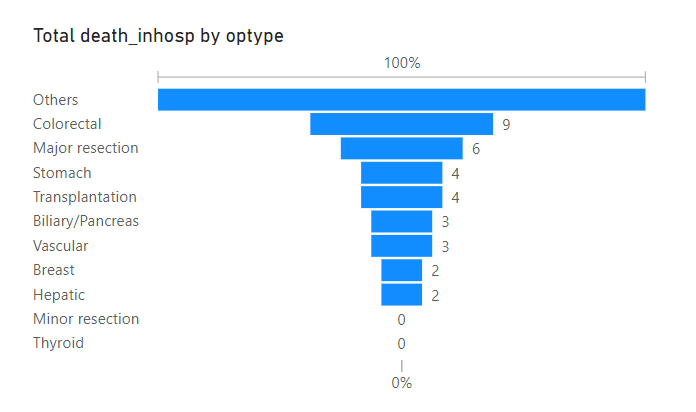


fig. 46 Using PowerBI with BigQuery

For Data visualization we’ve used PowerBI Desktop. DataSource is Google BigQuery.

fig. 47 ICU occupation by different departments

Helps to visualize which department is using resources in ICU. We can see General Surgery department has patients who stay more in ICU days.

Fig.48 Percentage of deaths by cause

The analysis states that highest percentage of deaths are by colorectal diseases.

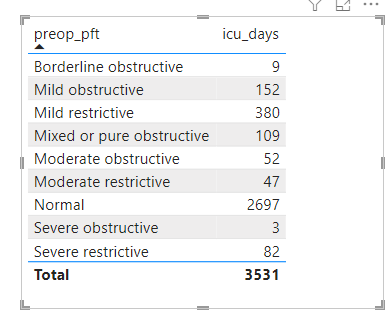
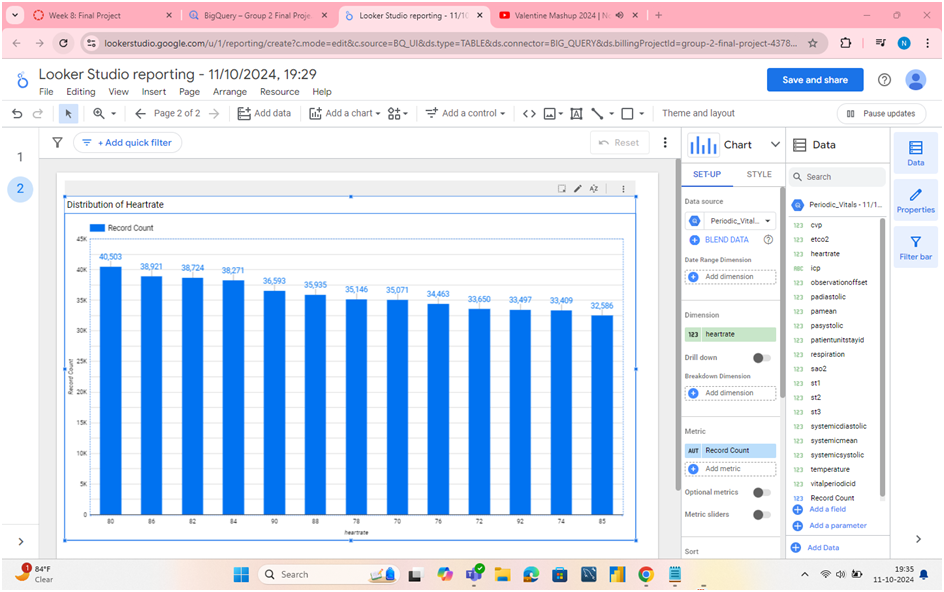


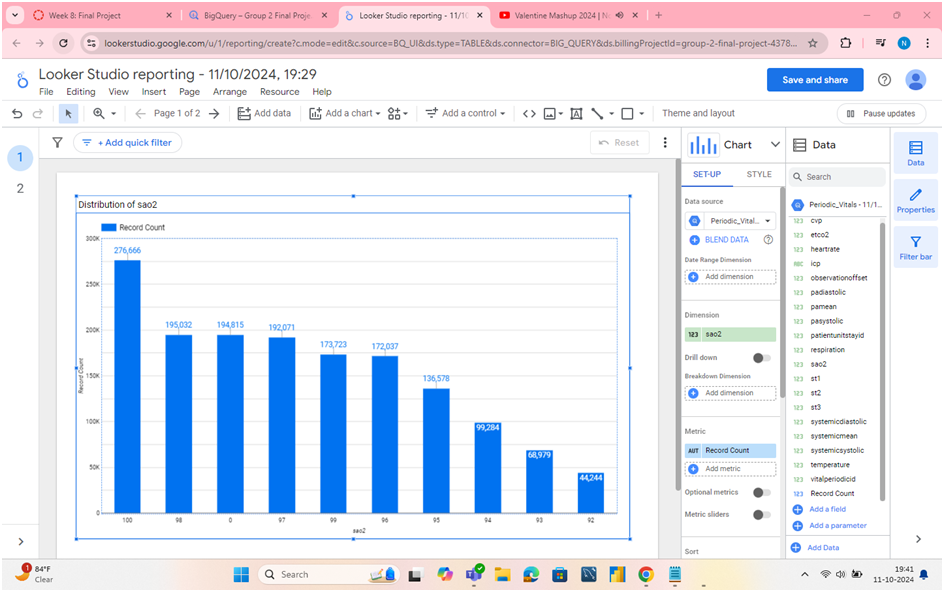
Fig. 49 ICU days based on Pre-operation pulmonary conditions

Helps in predicting how many days patient may occupy ICU based on their pre-operation pulmonary condition.



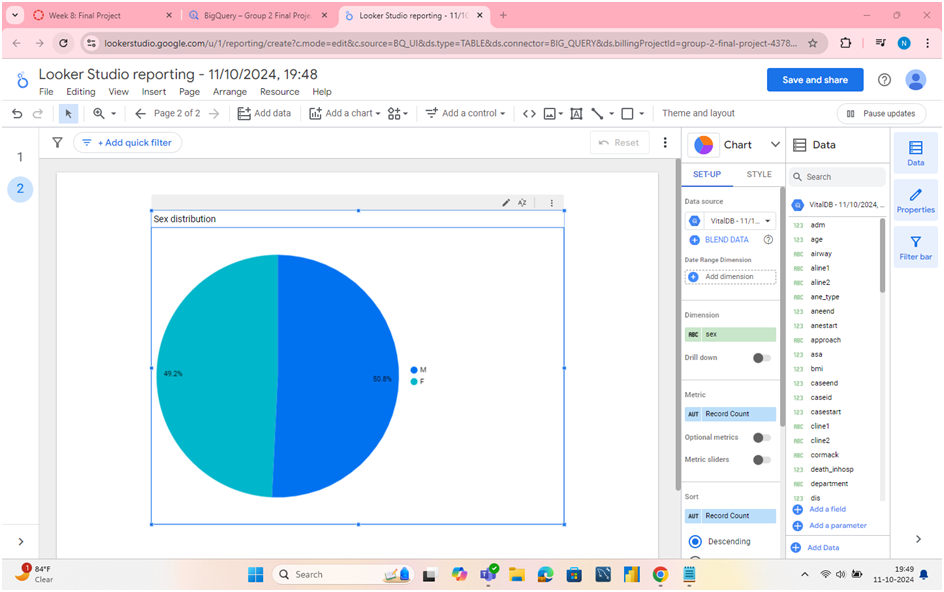
*fig. 50 Distribution of Heartrate*

The distribution of heartrate is given in bar chart and is sorted in descending order. We can see that highest number of people have heartrate of 80 and least is for heartrate of 85.



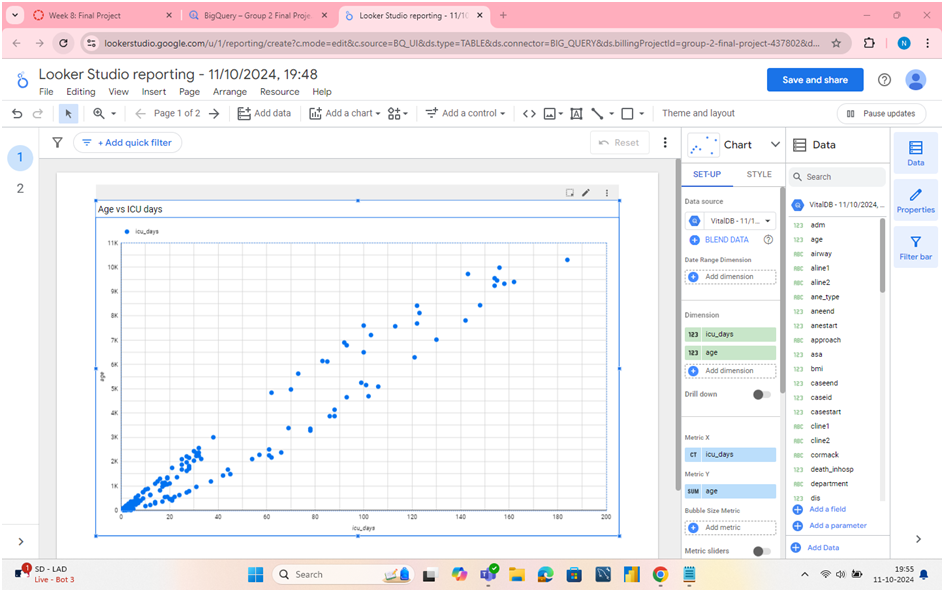
*fig. 51 Distribution of saO2*

The distribution of sao2 is given in bar chart and is sorted in descending order.We can see the that most people have sao2 as 100 which reflects a healthy oxygen level in blood.



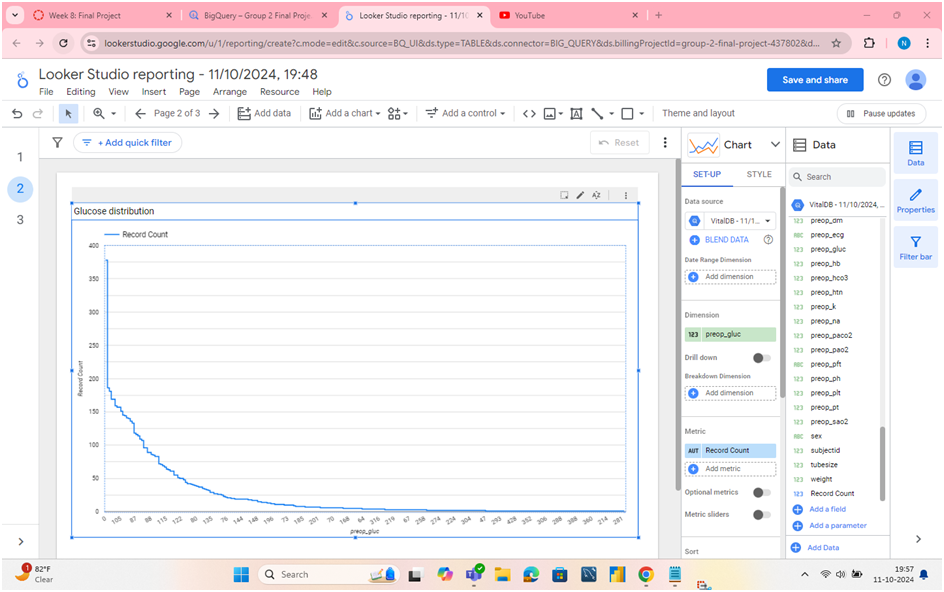
*Fig. 52 Data distribution by Sex*

Sex distribution is given in pie chart. Both male and female have equal distribution. Hence, our dataset is balanced in terms of sex.



*fig. 53 Scatter plot for Age Vs ICU days*

Then we have plotted scatter chart for number of days in ICU and age. The ICU days increases as the age increase. It shows a linear graph.

fig. 54 Glucose distribution before operation

In the line chart of glucose levels before operation, looks like decreasing curve and all points given as steps. These levels are important as based on the glucose levels operations are performed. Highest is for glucose level of 105 and least is for level of 281.