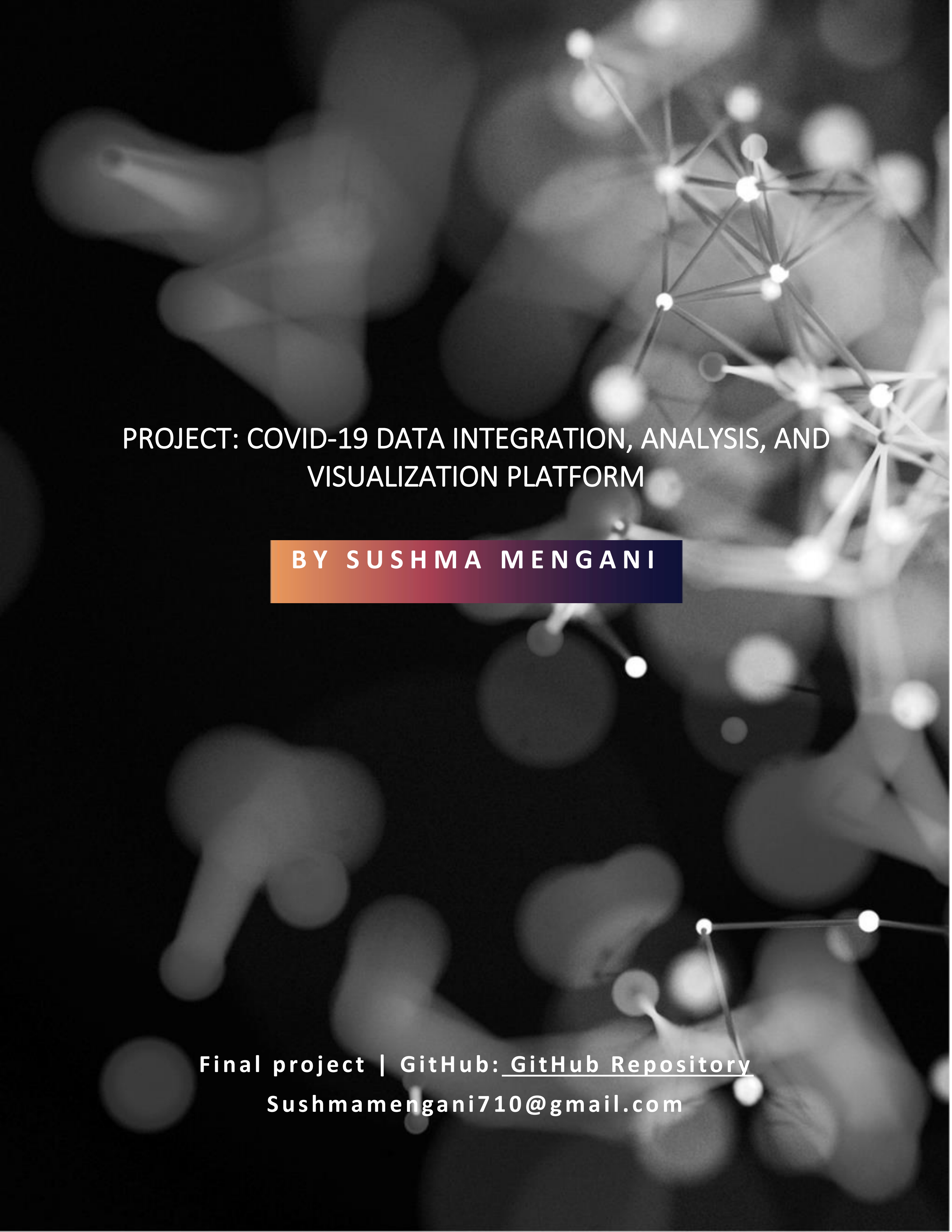


PROJECT: COVID-19 DATA INTEGRATION, ANALYSIS, AND VISUALIZATION PLATFORM

BY SUSHMA MENGANI

Final project | GitHub: GitHub Repository

Sushmamengani710@gmail.com

Introduction

The project aimed to create an integrated data analytics and visualization platform using Snowflake, AWS, Python, NoSQL database (MongoDB), and visualization libraries (Dash/Plotly/Matplotlib). My platform focused on analyzing the COVID-19 dataset, providing insights into the virus's spread, effects, and patterns across regions and populations. In addition to that, It focused specially on global economy and effects on individual countries.

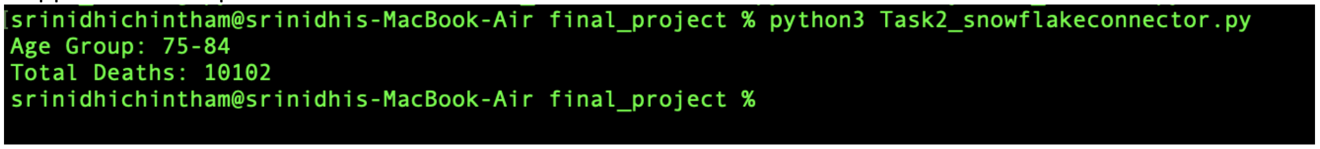
My Implementation

Snowflake Trial Account Setup

- Created a free Snowflake trial account on AWS in the Ohio region.
- obtained my COVID19_EPIDEMIOLOGICAL_DATA from the Snowflake Marketplace
- Configured Snowflake Resource monitoring using SQL from Snowflake Resource Monitor Documentation

Data Exploration and Enhancement

- Analyzed the COVID-19 dataset from Snowflake using SQL to understand its structure and patterns.
- Used Python and Snowflake to augment the dataset by combining it with demographic and economic data from external sources.
- To augment this dataset, I combined it with global economic data sourced from Kaggle Dataset
- To establish a connection to Snowflake using Python, I explored multiple methods until I identified one that would be successful but I got an error with that as well. Initially, I created a new Network Security Policy, which facilitated the connection. However, after a week, I encountered login issues due to a change in my IP address. In response, I updated the Network Security Policy accordingly, allowing me to regain access and after some research I got to know that It is not necessary to add the Network Security Policy. I have attached the relevant py file, I successfully connected to Snowflake with **import.snowflake.connector**
- To integrate my downloaded data with Snowflake, I created a new database and loaded my data into it. I have chosen this approach as I couldn't directly load my data into the shared database (covid_19_epidemiological_data). While there are alternative methods involving exporting and loading data into my private database, creating a new database and loading the data proved to be a more straightforward and clear process. This setup allows me to easily join any tables from the created database (Economic_data) to the COVID database.
- I verified my connection by executing a simple query, and I can confirm that the output is as expected. Below is a snippet of the output:



```
srinidhichintham@srinidhis-MacBook-Air final_project % python3 Task2_snowflakeconnector.py
Age Group: 75-84
Total Deaths: 10102
srinidhichintham@srinidhis-MacBook-Air final_project %
```

- I used MongoDB to construct a NoSQL schema for storing URLs, user information, and corresponding comments regarding both Economic data and the post-COVID vaccination effects.

Data Modeling in NoSQL (MongoDB)

- Designed a schema in MongoDB to store supplementary data, such as user comments and annotations. Included the schema design in the final report.

Detailed implementation:

In pursuit of enhanced data accessibility and interaction, I started on the development of an API using the Python frameworks Dash and Flask. This API serves as a robust interface, empowering users to query Snowflake for data based on their inputs, interact with a NoSQL database for relevant additional data or metadata, perform on-the-fly data processing, and implement caching mechanisms for frequently requested data.

I was facing many error when I tried using (covid_19_epidemiological_data) because it is shared database so I was not able to query the data then I have used my Database (Economic_data) with which I was able to connect successfully.

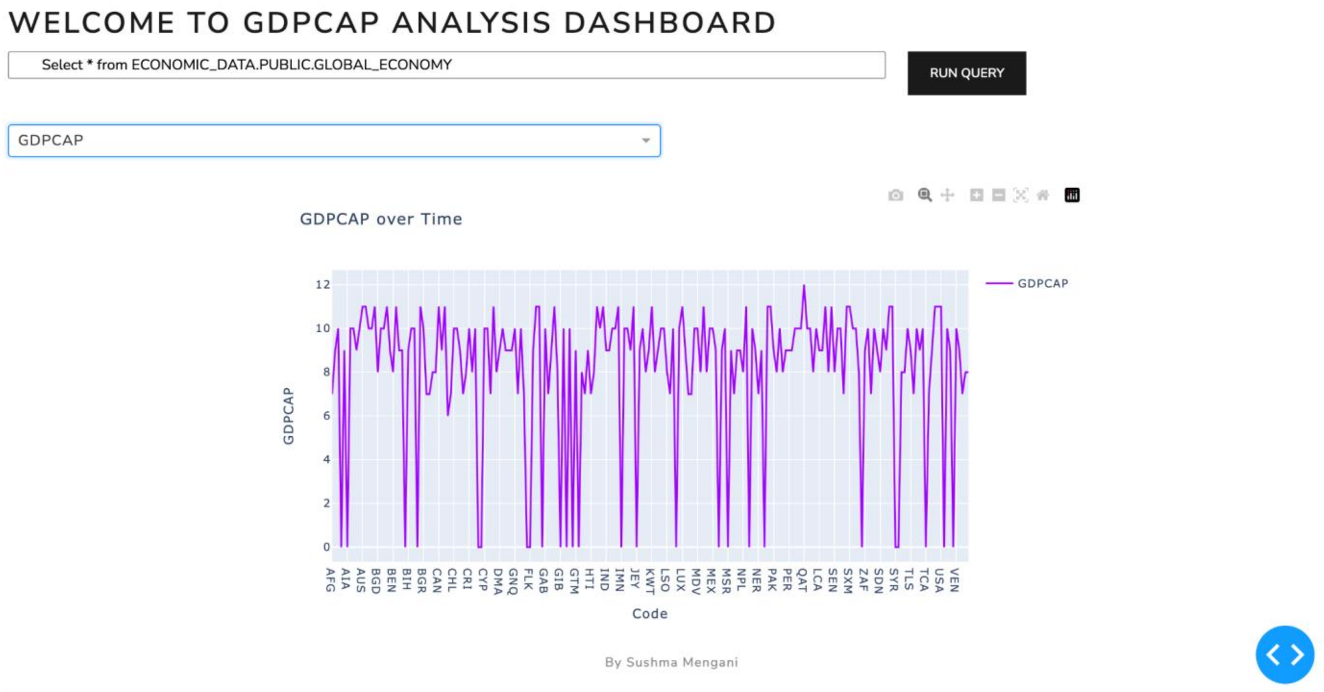
The code for this API is encapsulated within the provided Python script, where I employed Dash for creating interactive dashboards and Flask for the underlying web server. The functionalities of the API encompass the following key features:

This API integrates seamlessly with Snowflake, allowing users to execute SQL queries and retrieve data based on their inputs. This is achieved through the implementation of caching mechanisms to optimize the retrieval process. Leveraging MongoDB, the API interacts with relevant additional data or metadata.
and it also facilitates on-the-fly data processing, enabling users to dynamically analyze and visualize data. Utilizing Matplotlib and Seaborn, the API generates visual insights into the impact of COVID-19 on the global economy, mortality trends, and other relevant aspects.

To enhance performance and responsiveness, it incorporates a caching mechanism for frequently requested data. This optimization ensures that users experience minimal latency when accessing commonly queried information.
The provided code not only demonstrates the technical implementation of these features but also showcases the utilization of external libraries and frameworks to create an intuitive and dynamic data analysis dashboard. By running the Flask and Dash apps, users can interact with the API and explore visualizations tailored to their queries.

This is the Sample Query which I have used and I can see the output :

```
Select * from ECONOMIC_DATA.PUBLIC.GLOBAL_ECONOMY
```


WELCOME TO GDPCAP ANALYSIS DASHBOARD

Select * from ECONOMIC_DATA.PUBLIC.GLOBAL_ECONOMY

RUN QUERY
GDPCAP
GDPCAP over Time
GDPCAP
Code
AFG AIA AUS BGD BEN BIH BGR CAN CHL CRI CYP DMA GNQ FLK GAB GIB GTM HTI IND IMN JEY KWT LSO LUX MDV MEX MSR NPL NER PAK PER QAT LCA SEN SXM ZAF SDN SYR TLS TCA USA VEN
0 2 4 6 8 10 12
By Sushma Mengani

This is my how my final API dashboard looks, it is publicly accessible via URL http://3.72.21.155:8054 provides direct access to this API, allowing users to explore and interact with the developed features.
I have hosted my **api** on **EC2 instance, configured inbound security rules** and hence it is publicly accessible.

API Development with Python

- Developed a robust API using Flask.
- Enabled the API to query Snowflake for data based on user inputs.
- Integrated with the NoSQL database for additional data and metadata.
- Implemented on-the-fly data processing and caching for frequently requested data.

Detailed implementation:

Utilizing the visualization capabilities within Snowflake, I enhanced the visualizations using Python libraries such as Matplotlib and Seaborn. which enabled me to create visualizations that provide valuable insights into the progression of the COVID-19 pandemic.

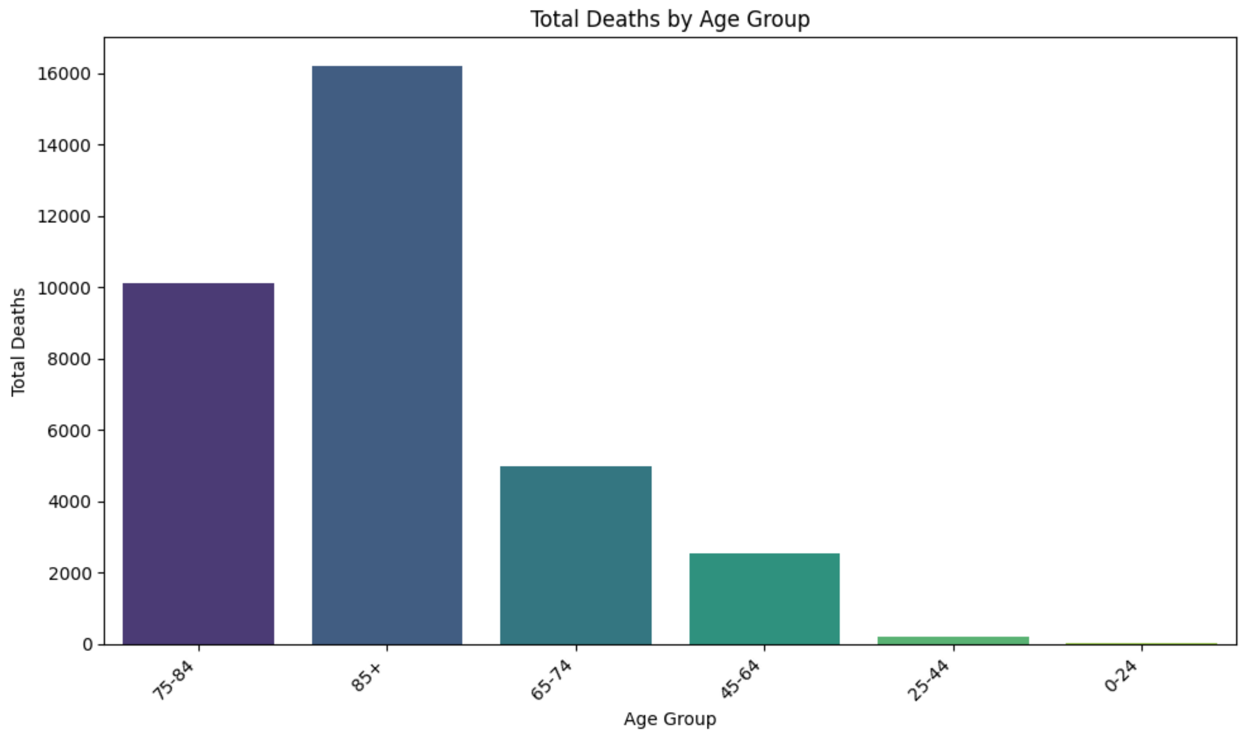
Interactive Visualization with Python

- Utilized Dash/Plotly to create interactive web dashboards for visualizing metrics like infection rates, mortality rates, and demographic breakdown.
- Provided flexibility for users to add annotations or comments, stored in the NoSQL DB.

Here are some of my visualizations:

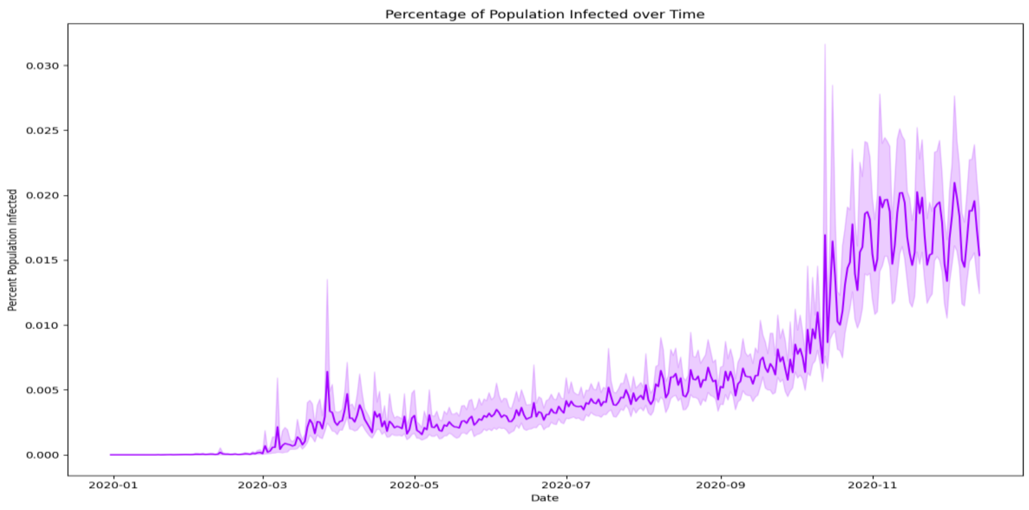
Mortality Across Age Groups :

To gain insights into mortality trends, I crafted an SQL query targeting the scs_be_detailed_mortality table. The query focused on aggregating total deaths based on distinct age groups. The resulting dataset was translated into a compelling visual representation using Seaborn and Matplotlib. The bar graph vividly illustrates the distribution of total deaths among various age categories. This initial visualization played a crucial role as a starting point, confirming the smooth processing of data and instilling confidence for further, more intricate visualizations in the subsequent stages of our analysis**.**



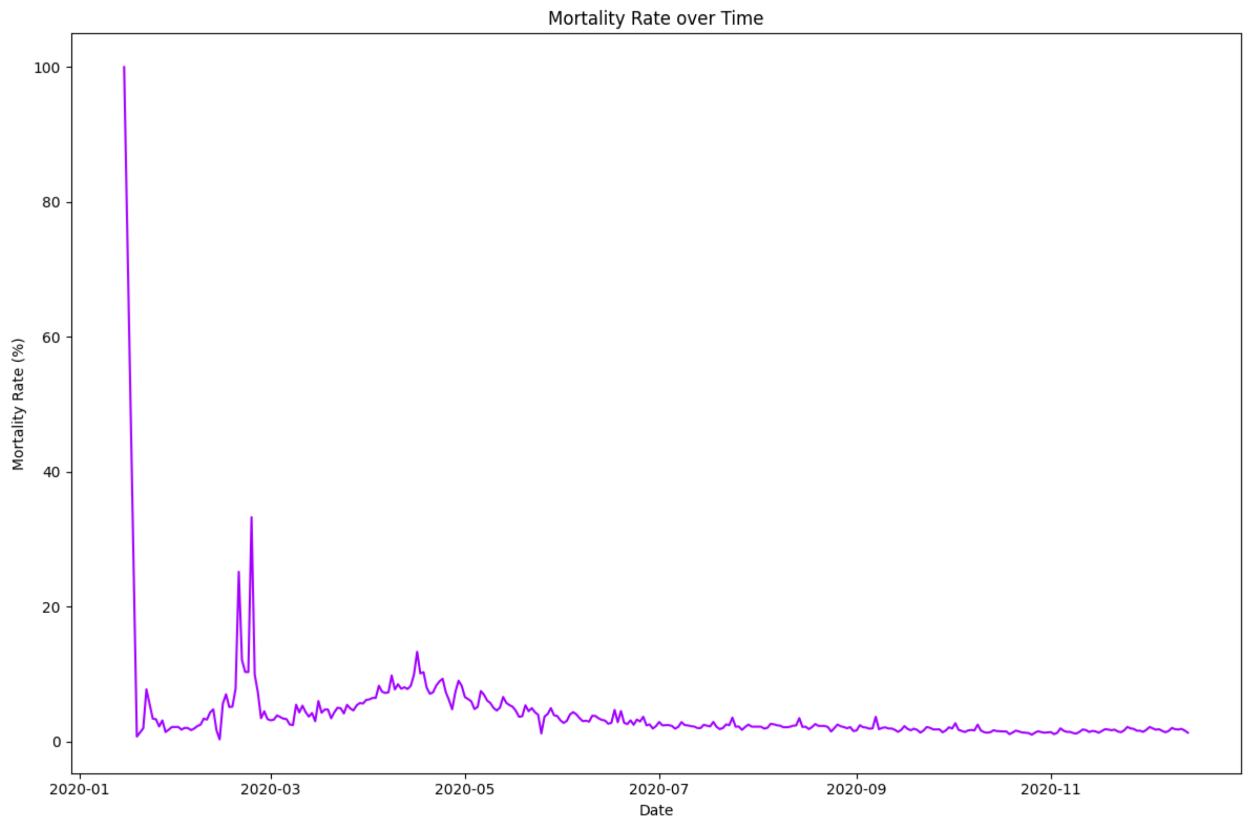
Tracking Infection Rates:

I used Matplotlib and Seaborn to visualize the dynamic changes in the percentage of the population infected with COVID-19 and the percentage of deaths relative to cases over different dates. These visualizations are useful in tracking the impact of COVID-19



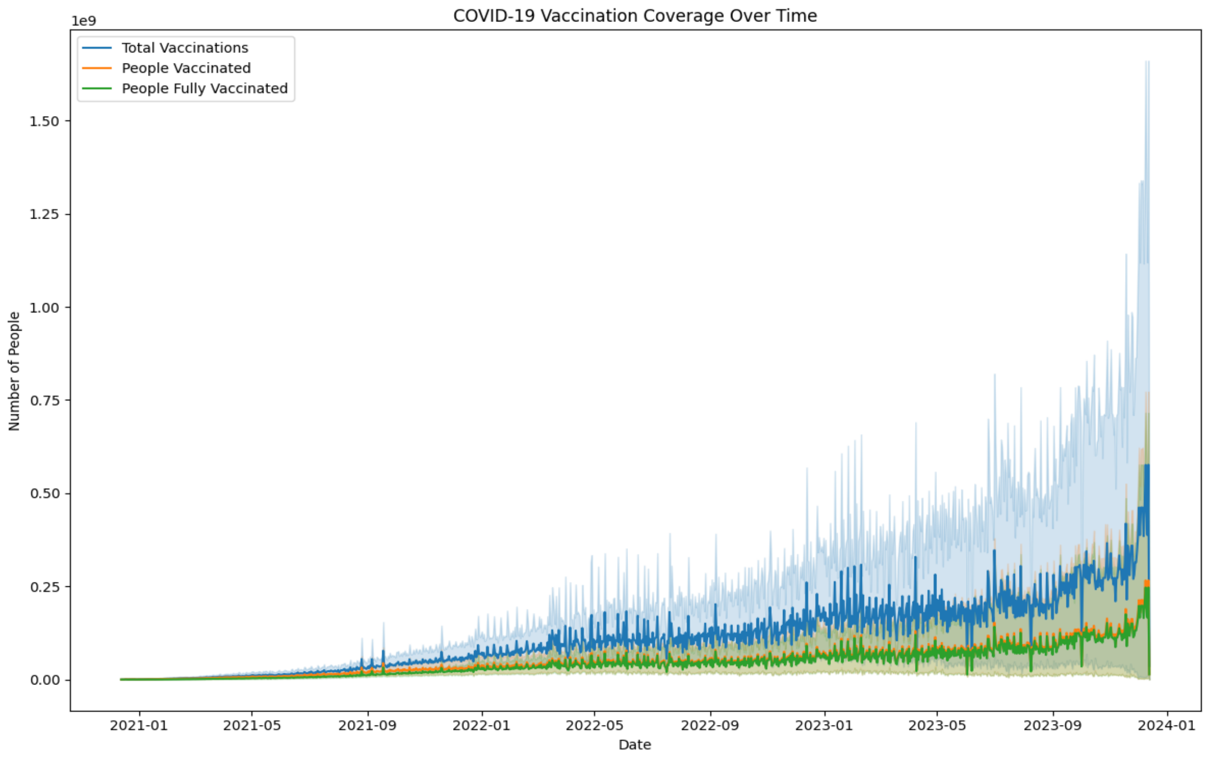
Mortality Rate Over Time:

Shows the percentage of deaths attributed to COVID-19 relative to the total number of reported cases across different dates.The visualization is particularly useful for tracking the impact of COVID-19 and evaluating the effectiveness of public health interventions and healthcare resource allocation strategies.



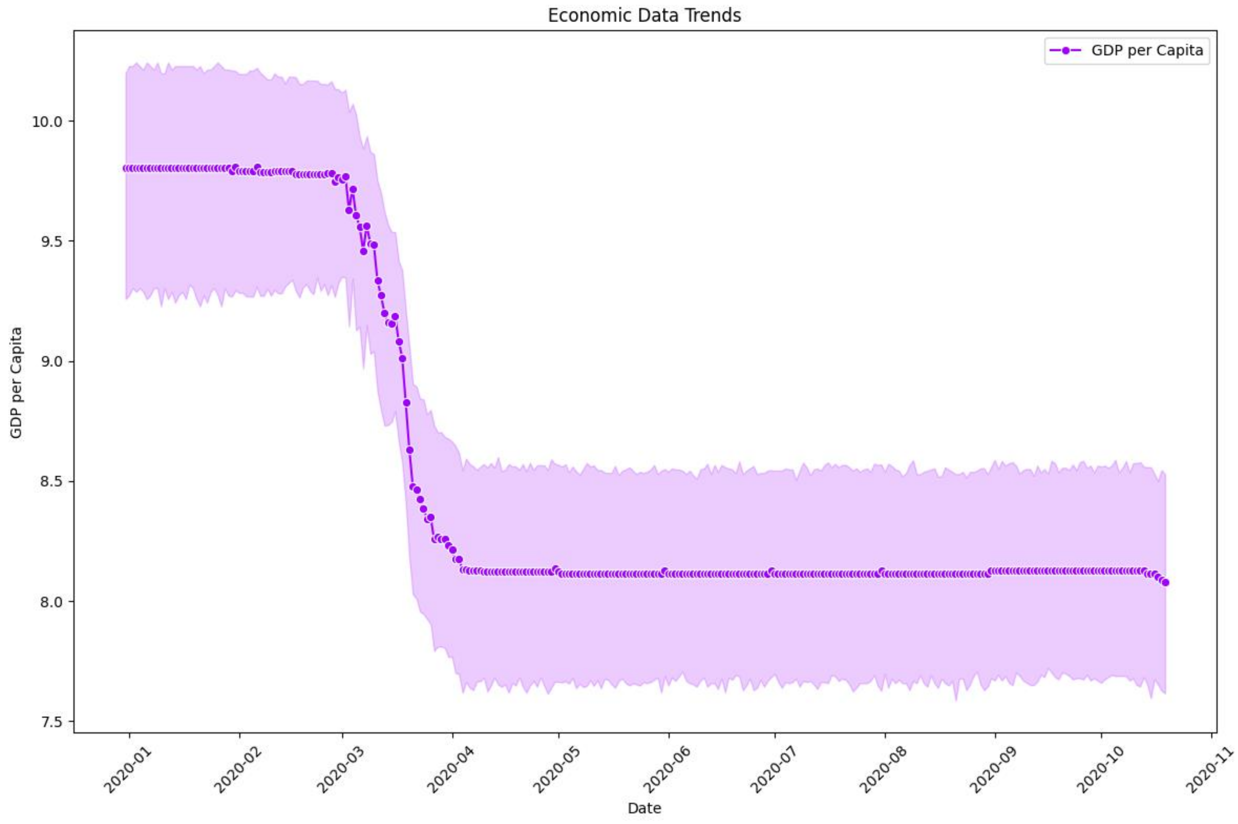
Monitoring Vaccination Progress:

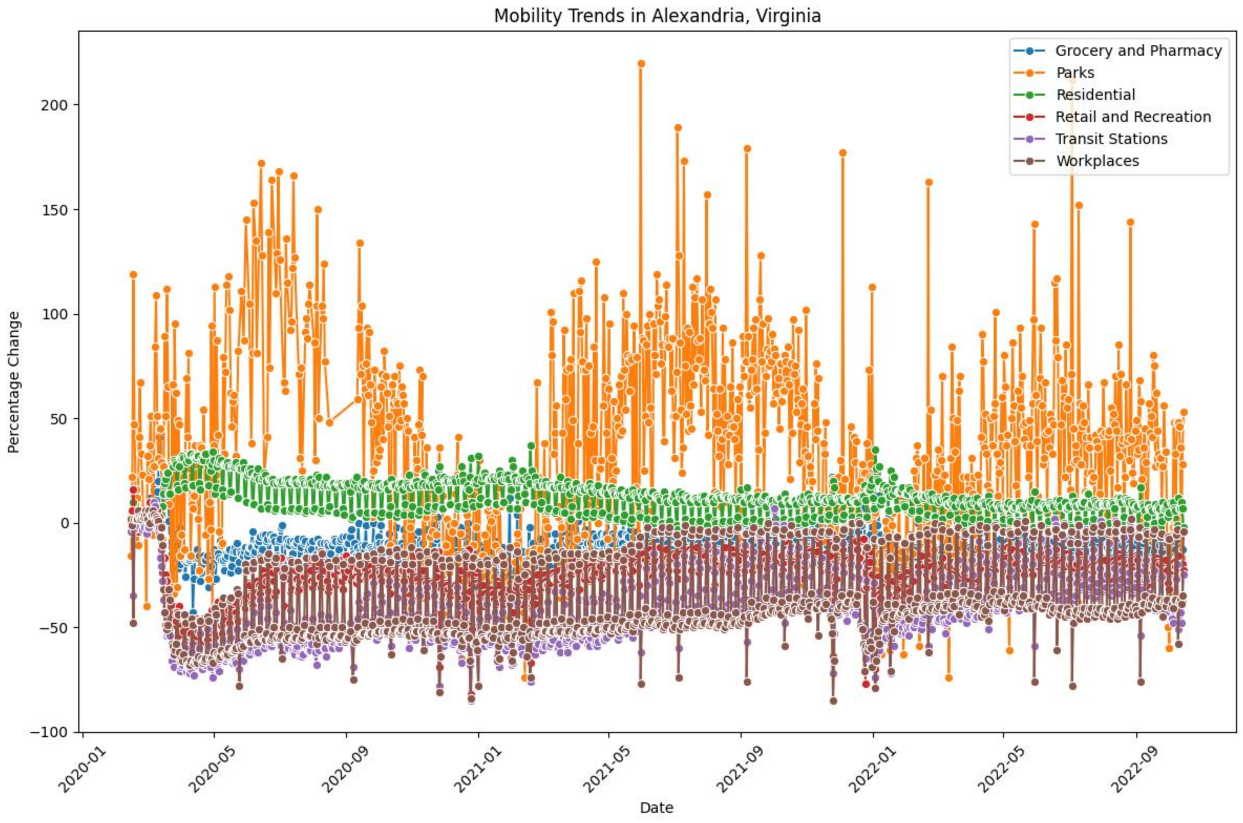
Leveraging the same Python libraries, I tracked the progress of COVID-19 vaccination efforts. The visualizations showcase the number of people partially and fully vaccinated over time, providing a comprehensive view of vaccination coverage across different countries. This is crucial for assessing vaccination efforts and understanding the evolving landscape of immunization.



GDP (Economy) Impact during COVID-19:

In addition to that, I explored the effects of the pandemic on the global economy. By utilizing Matplotlib and Seaborn, I created visualizations to illustrate how GDP was affected during the COVID-19 period. These visual insights contribute to a better understanding of the economic repercussions of the pandemic.





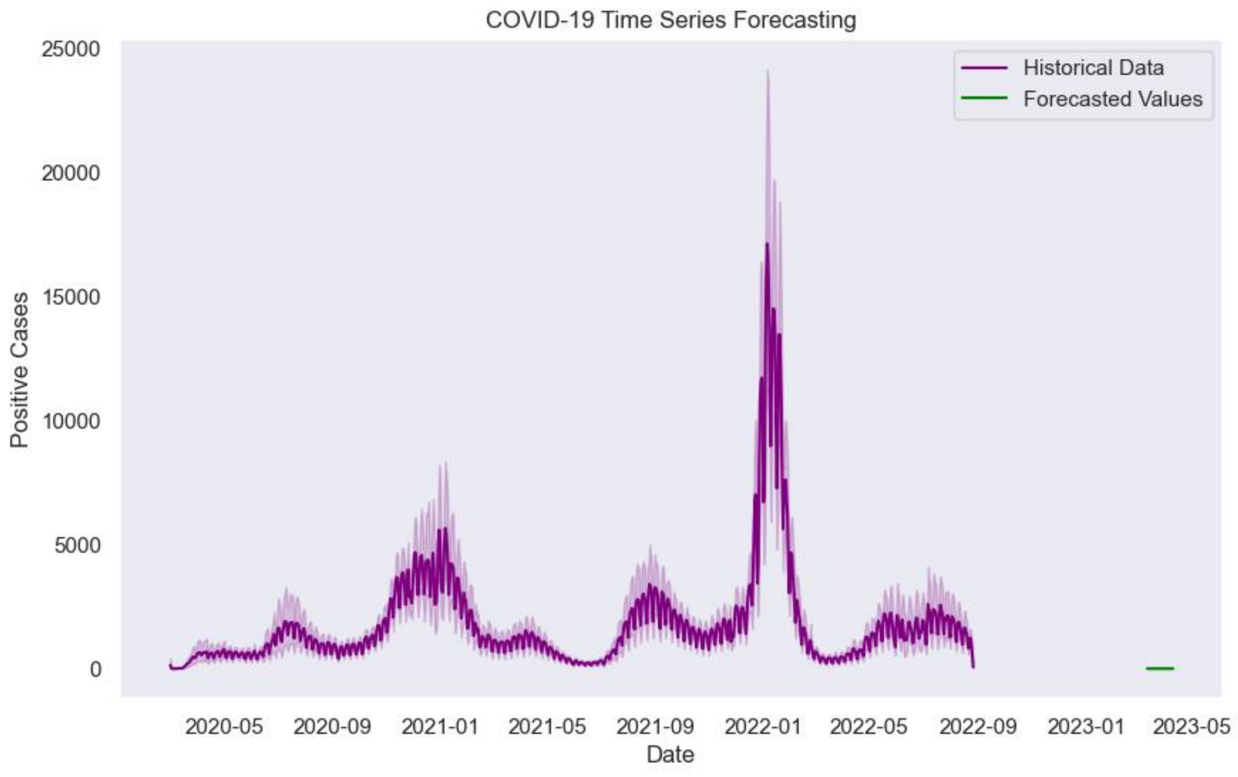
Analytical Features

- Implemented Time Series Forecasting using Python for predicting future infection rates.
- Bonus: Conducted clustering to segment regions based on similarities in COVID-19 spread patterns.

For time series forecasting, I Used the ARIMA (AutoRegressive Integrated Moving Average) model (statsmodels.tsa.arima.model) This predictive model allowed me to anticipate the potential increase in COVID-19 cases and calculate the mortality rate for specific countries.

COVID-19 Cases Forecasting:

Using ARIMA model, I conducted time series forecasting to predict the future trajectory of COVID-19 cases. This model takes account of historical data, identifying patterns and trends that aid in potential increases in cases. This forecasting capability is crucial for proactive planning and resource allocation.

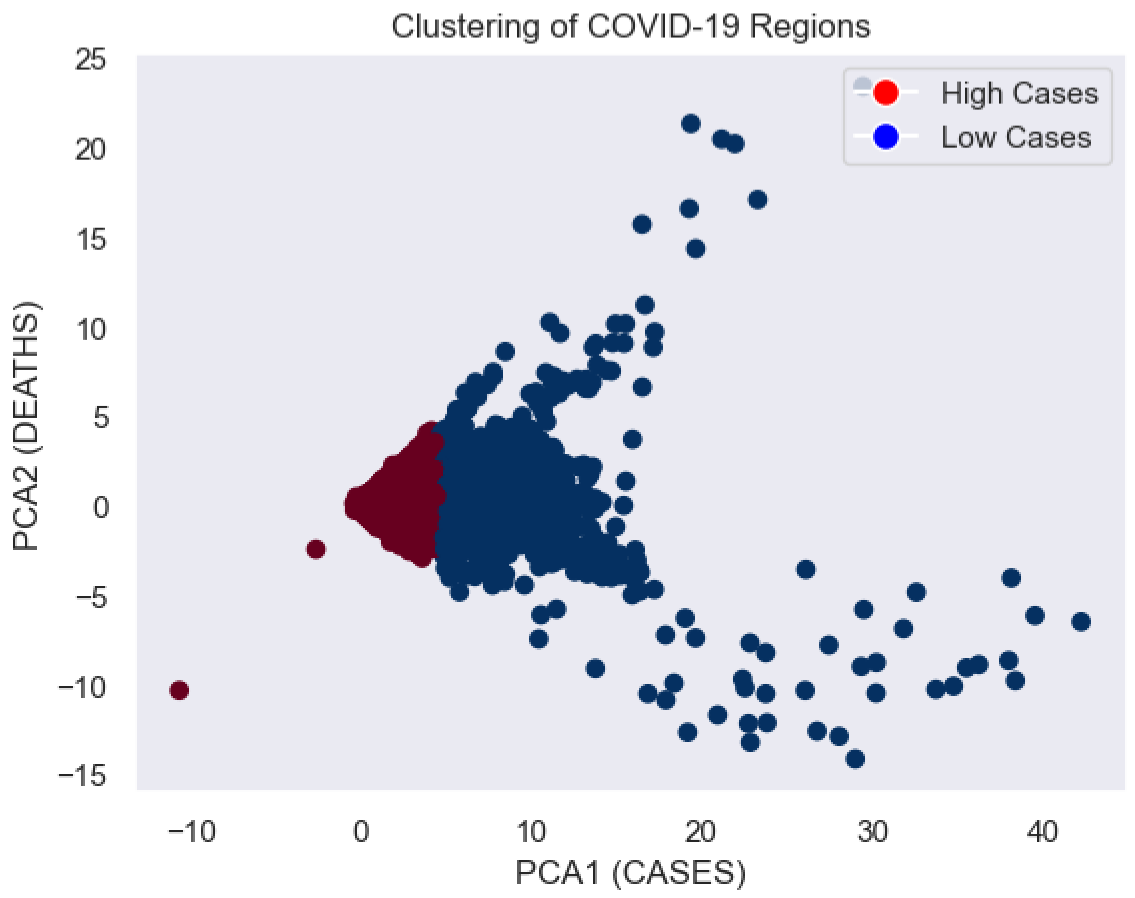


Mortality Rate Estimation:

Additionally, the ARIMA model was applied to estimate the mortality rate for particular countries. By analyzing historical data, the model provides insights into the expected mortality rates, assisting in understanding the severity of the pandemic's impact on different regions.

These ARIMA-based forecasts contribute to a more informed decision-making process, helping to guide public health strategies and interventions.

In order to gain deeper insights into the regional dynamics of COVID-19, I applied clustering for the table which has essential information such as the number of cases and deaths across different country regions. The resulting clustered data was visualized in a scatter plot, where each point represents a region, positioned in the two principal components space. The color-coded clusters, denoting regions with low and high cases, provide a clear representation of the distinct patterns in COVID-19 across different geographical areas. This visualization is useful in identifying regions with similar characteristics and resource allocations.



Performance Optimization

- Implemented caching in the API for frequently requested data.
- To optimize the performance of my SQL queries, I used Snowflake features to ensure efficient data retrieval and processing. This optimization is crucial for enhancing the overall speed and responsiveness of the system.

Pattern Recognition with MATCH_RECOGNIZE

- I used MATCH_RECOGNIZE feature in Snowflake to get the dynamic patterns of COVID-19 cases, deaths, and testing trends across distinct geographical regions. At the beginning it was hard to select what is the relevant data to use the MATCH_RECOGNIZE but I figured out these queries would be appropriate

Detailed Implementation:

First, identifying sequences where both the number of COVID-19 cases and deaths exhibit an increase, thereby illuminating periods of heightened activity in specific areas.

Second, the analysis extends to capture sequences reflecting the fluctuating dynamics of positive, negative, and inconclusive test results within defined regions. This provides useful insights into the varied testing trends witnessed across different geographical areas.

Furthermore, a focused query where both deaths and confirmed cases experience sudden upswings in country regions. Notably, this query excludes instances where the mortality rate is null, ensuring a refined examination of impactful events.

Project Structure Sharing

- Shared all project structures with Snowflake account: AUKSKCD.LP35718. Locator: IP42871
- All the files were committed to Github account: GitHub Repository