COVID-19 CASES ANALYSIS USING COGNOS

Phase-4 Submission Document

Project Name: COVID-19 CASES ANALYSIS

Phase 4: Development Part 2

In this part we will continue building our project.

- Continue building the analysis by creating visualizations using IBM Cognos and deriving insights from the data.
- Create charts and graphs in IBM Cognos to visualize and compare the mean values and standard deviations of COVID-19 cases and associated deaths.
- Analyze the visualizations to identify trends, variations, and potential correlations between cases and deaths.

Step-1:

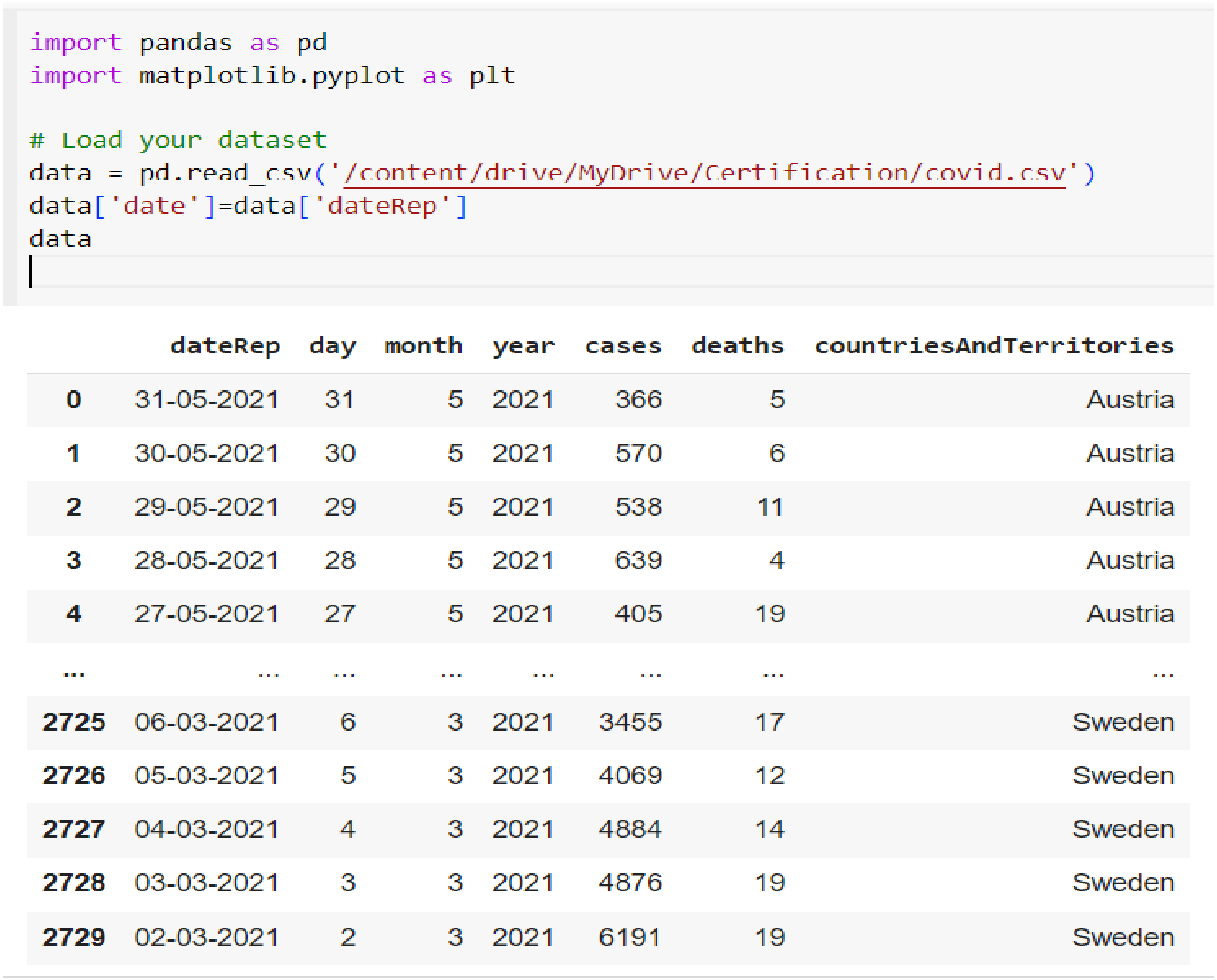
Load the Provided Dataset:

- Loading the dataset involves reading the data from a file, typically a CSV (Comma-Separated Values) file, into your data analysis environment, which in this case, could be Python.

- You can use libraries like Pandas to accomplish this.

- The Pandas library provides powerful data structures and functions for working with structured data.

Example:the following is the loaded covid-19 dataset



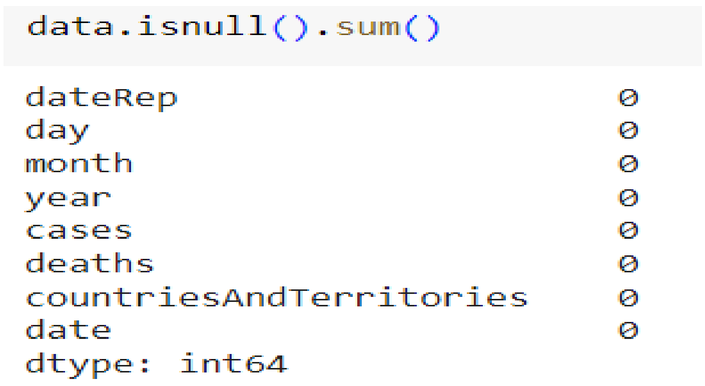
```
import pandas as pd
import matplotlib.pyplot as plt

# Load your dataset
data = pd.read_csv('/content/drive/MyDrive/Certification/covid.csv')
data['date']=data['dateRep']
data
```

```
         dateRep  day  month  year  cases  deaths  countriesAndTerritories
   0  31-05-2021   31      5  2021    366       5                  Austria
   1  30-05-2021   30      5  2021    570       6                  Austria
   2  29-05-2021   29      5  2021    538      11                  Austria
   3  28-05-2021   28      5  2021    639       4                  Austria
   4  27-05-2021   27      5  2021    405      19                  Austria
 ...         ...  ...    ...   ...    ...     ...                      ...
2725  06-03-2021    6      3  2021   3455      17                   Sweden
2726  05-03-2021    5      3  2021   4069      12                   Sweden
2727  04-03-2021    4      3  2021   4884      14                   Sweden
2728  03-03-2021    3      3  2021   4876      19                   Sweden
2729  02-03-2021    2      3  2021   6191      19                   Sweden
```

2.

Inspect the Dataset:

- After loading the dataset, it's important to inspect it to understand its structure, contents, and any potential issues.
- You can use various Pandas functions to inspect the dataset, such as **head()**, **info()**, and **describe()**, to view the first few rows, get information about data types, and summarize statistical properties of the data.
- #check wheather there is a null values.

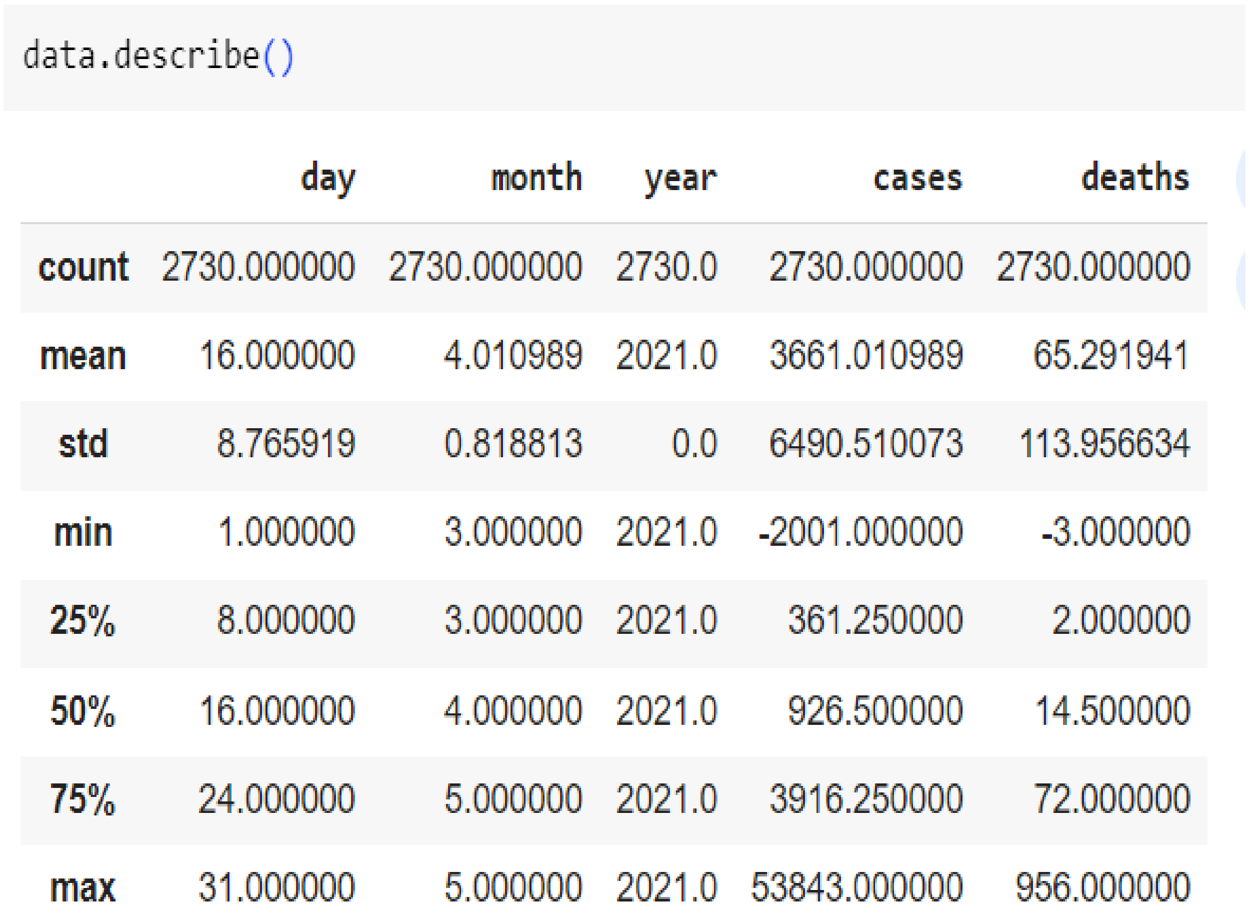


```
data.isnull().sum()
```

```
dateRep                    0
day                        0
month                      0
year                       0
cases                      0
deaths                     0
countriesAndTerritories    0
date                       0
dtype: int64
```

#describe the data

By using this python code we can able to know the mean,standard deviation,count,min,max for each and every columns



```
data.describe()
```

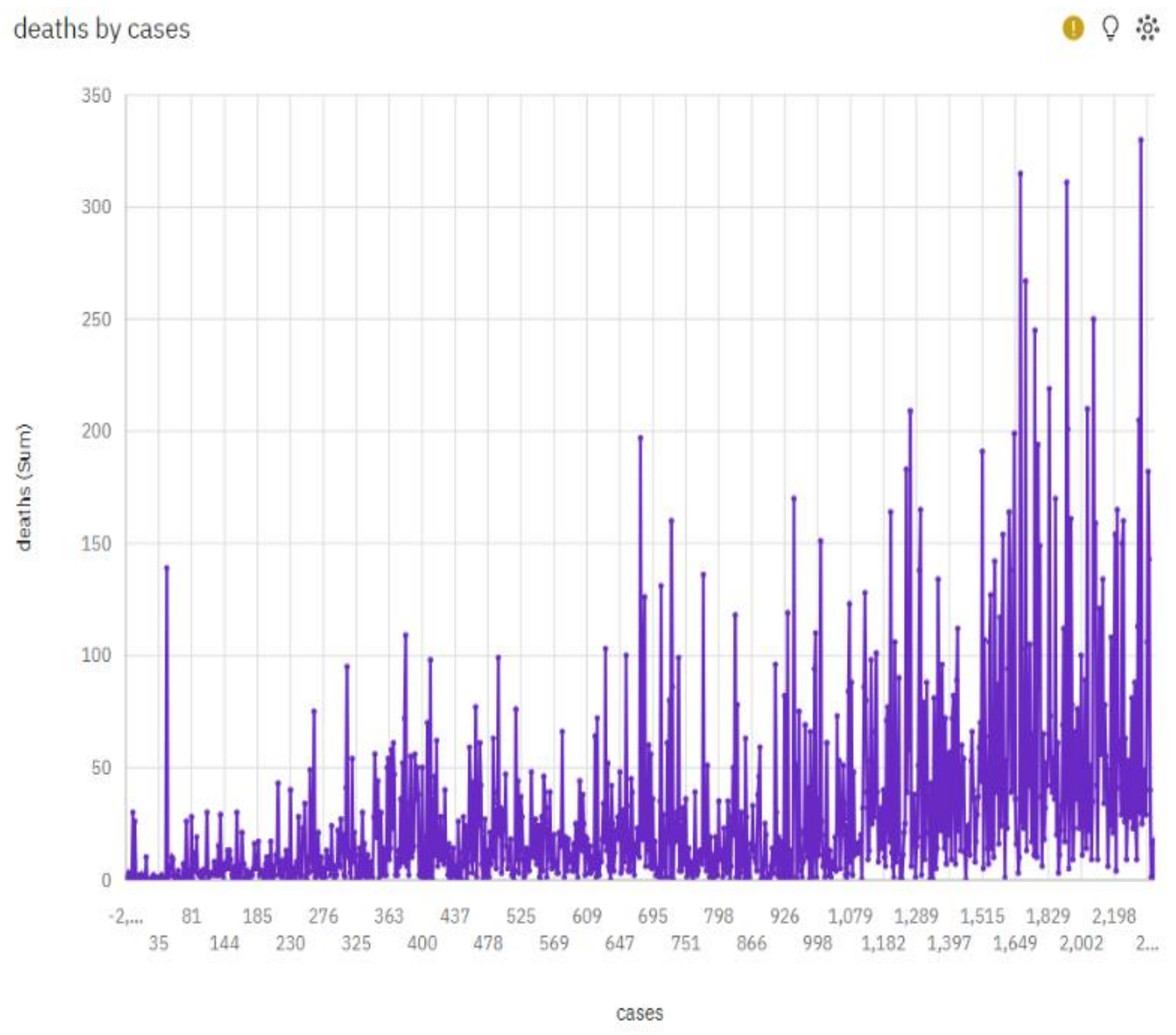
	day	month	year	cases	deaths
count	2730.000000	2730.000000	2730.0	2730.000000	2730.000000
mean	16.000000	4.010989	2021.0	3661.010989	65.291941
std	8.765919	0.818813	0.0	6490.510073	113.956634
min	1.000000	3.000000	2021.0	-2001.000000	-3.000000
25%	8.000000	3.000000	2021.0	361.250000	2.000000
50%	16.000000	4.000000	2021.0	926.500000	14.500000
75%	24.000000	5.000000	2021.0	3916.250000	72.000000
max	31.000000	5.000000	2021.0	53843.000000	956.000000

Step-2:

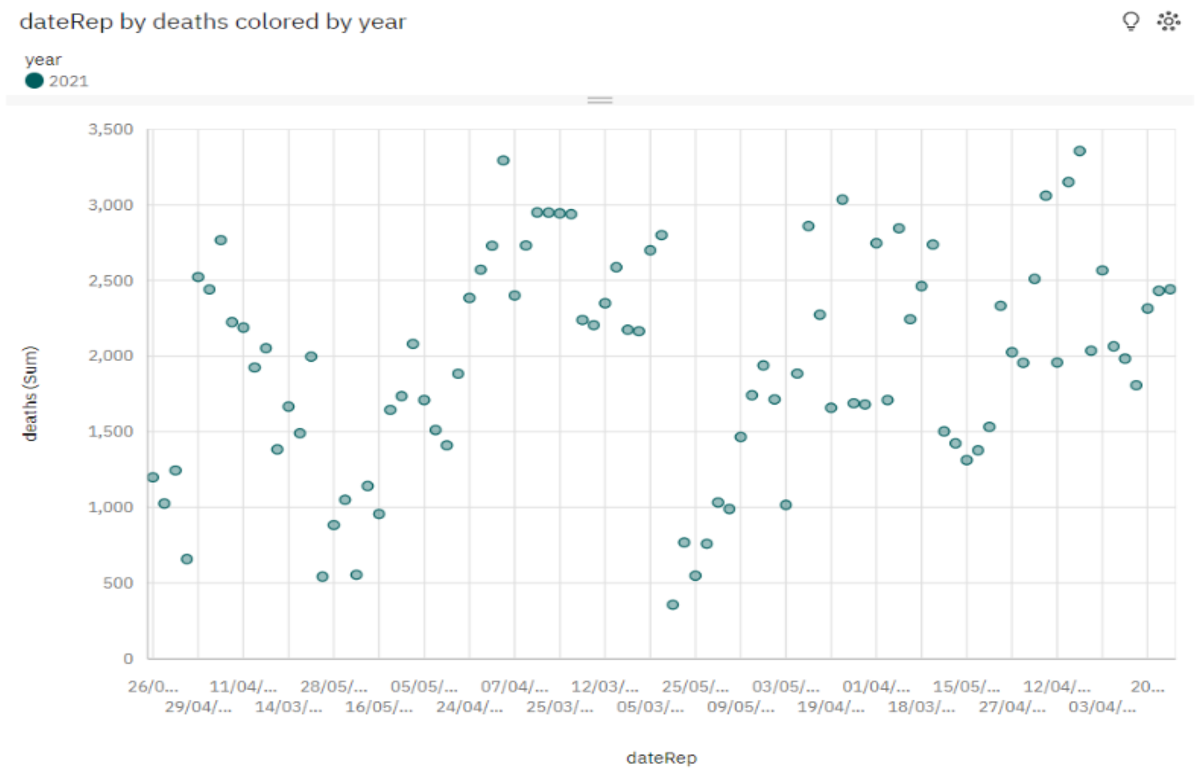
- Create charts and graphs in IBM Cognos to visualize and compare the mean values and standard deviations of COVID-19 cases and associated deaths.

  #using line chart visualizing the cases and deaths.cases in x-axis and deaths in y-axis

- By the following chart we can observe that as the number of cases increases the corresponding deaths increases.



#creating the scatter plot the deaths associated with the dateRep

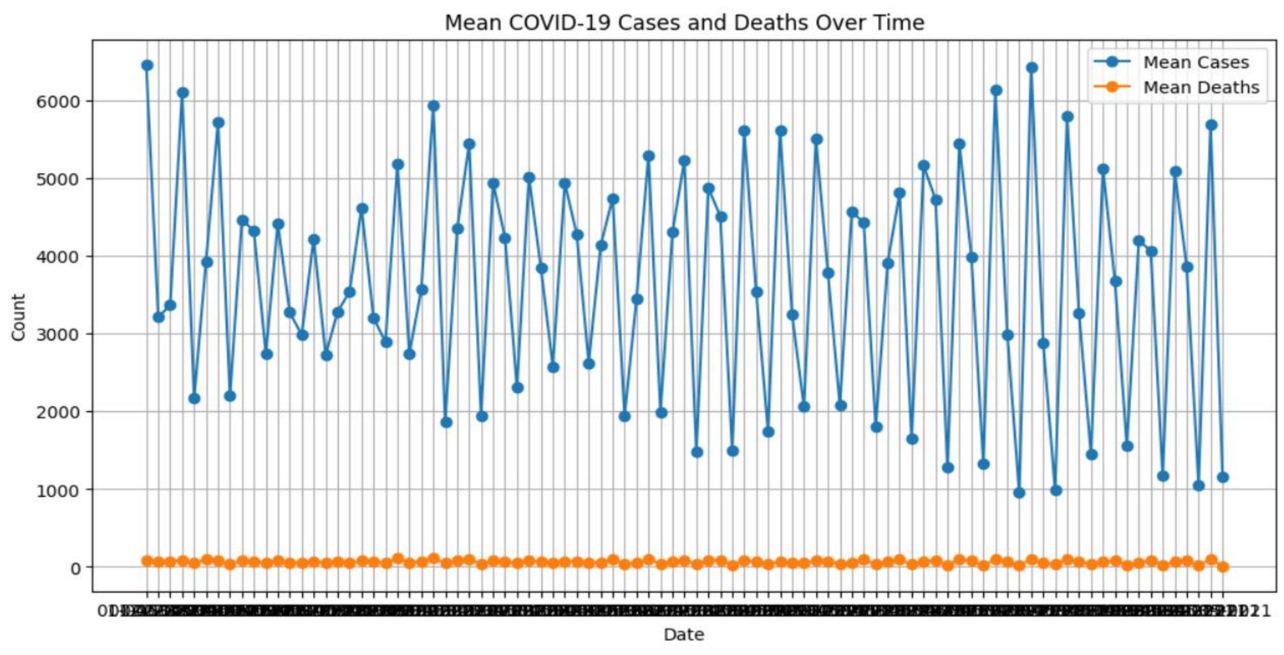


Step-3:

- visualize and compare the mean values and standard deviations of COVID-19 cases and associated deaths.
- Analyze the visualizations to identify trends, variations, and potential correlations between cases and deaths.

```python
# Group data by date and calculate mean values
mean_cases = data.groupby('date')['cases'].mean()
mean_deaths = data.groupby('date')['deaths'].mean()

# Create a line plot
plt.figure(figsize=(12, 6))
plt.plot(mean_cases.index, mean_cases, label='Mean Cases', marker='o')
plt.plot(mean_deaths.index, mean_deaths, label='Mean Deaths', marker='o')
plt.xlabel('Date')
plt.ylabel('Count')
plt.title('Mean COVID-19 Cases and Deaths Over Time')
plt.legend()
plt.grid(True)
plt.show()
```

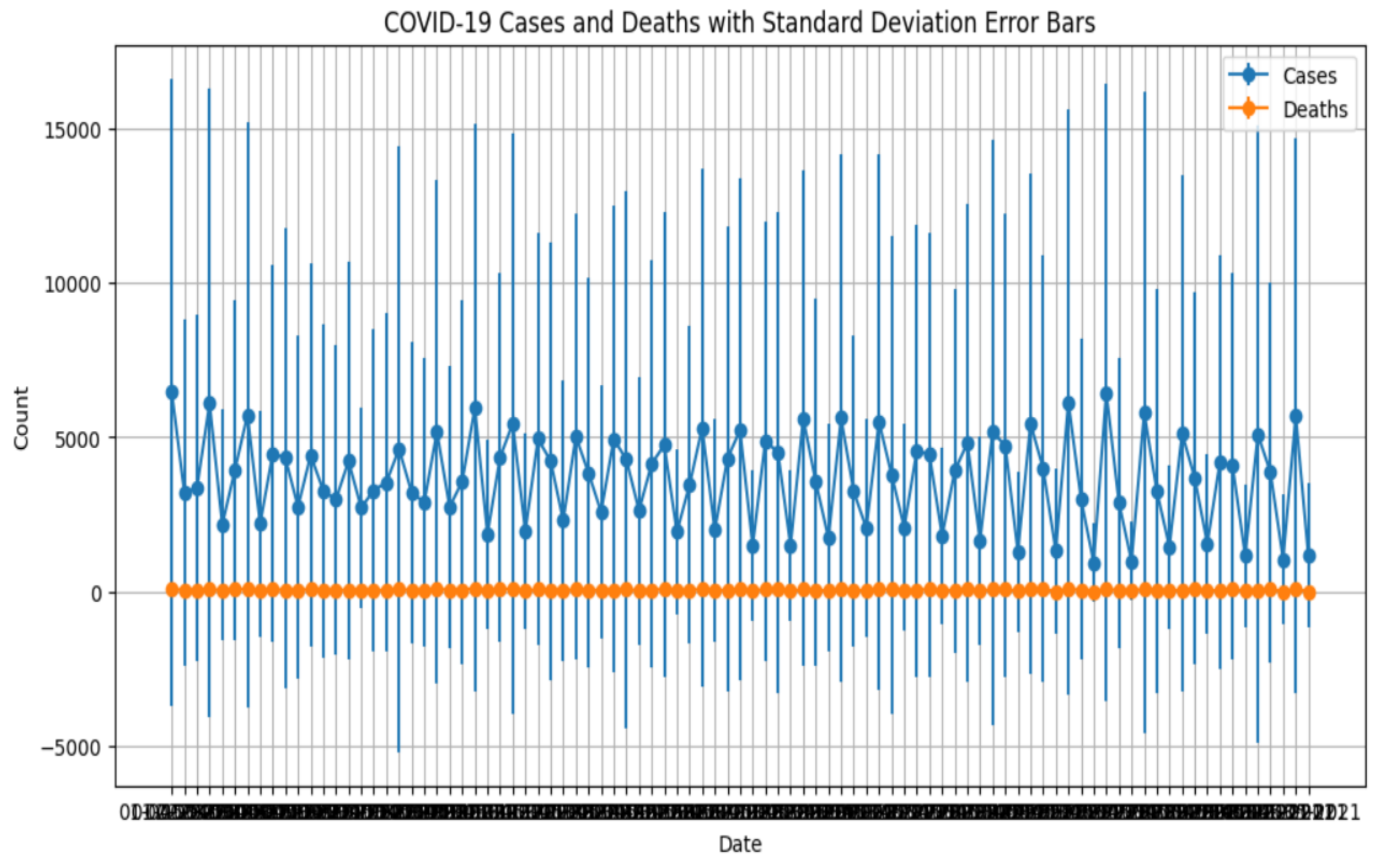
The above python code is used to group data by date and calculate mean values of both cases and deaths.



From the output we can observe that the mean of deaths and cases over the time period is consistent.

```
# Calculate standard deviations
std_cases = data.groupby('date')['cases'].std()
std_deaths = data.groupby('date')['deaths'].std()

# Create a line plot with error bars
plt.figure(figsize=(12, 6))
plt.errorbar(mean_cases.index, mean_cases, yerr=std_cases, label='Cases', marker='o')
plt.errorbar(mean_deaths.index, mean_deaths, yerr=std_deaths, label='Deaths', marker='o')
plt.xlabel('Date')
plt.ylabel('Count')
plt.title('COVID-19 Cases and Deaths with Standard Deviation Error Bars')
plt.legend()
plt.grid(True)
plt.show()
```

The above python code is used to group data by date and calculate the standard deviation of both cases and deaths.



From the above chart we can observe that the standard deviation of both cases and deaths are very low /small over the time period.

#finding the trend

To analyze the trend between cases and deaths using Python, you can use time-series analysis and data visualization techniques. Below is a Python code example to help you analyze and visualize the trend between COVID-19 cases and deaths using a sample dataset:

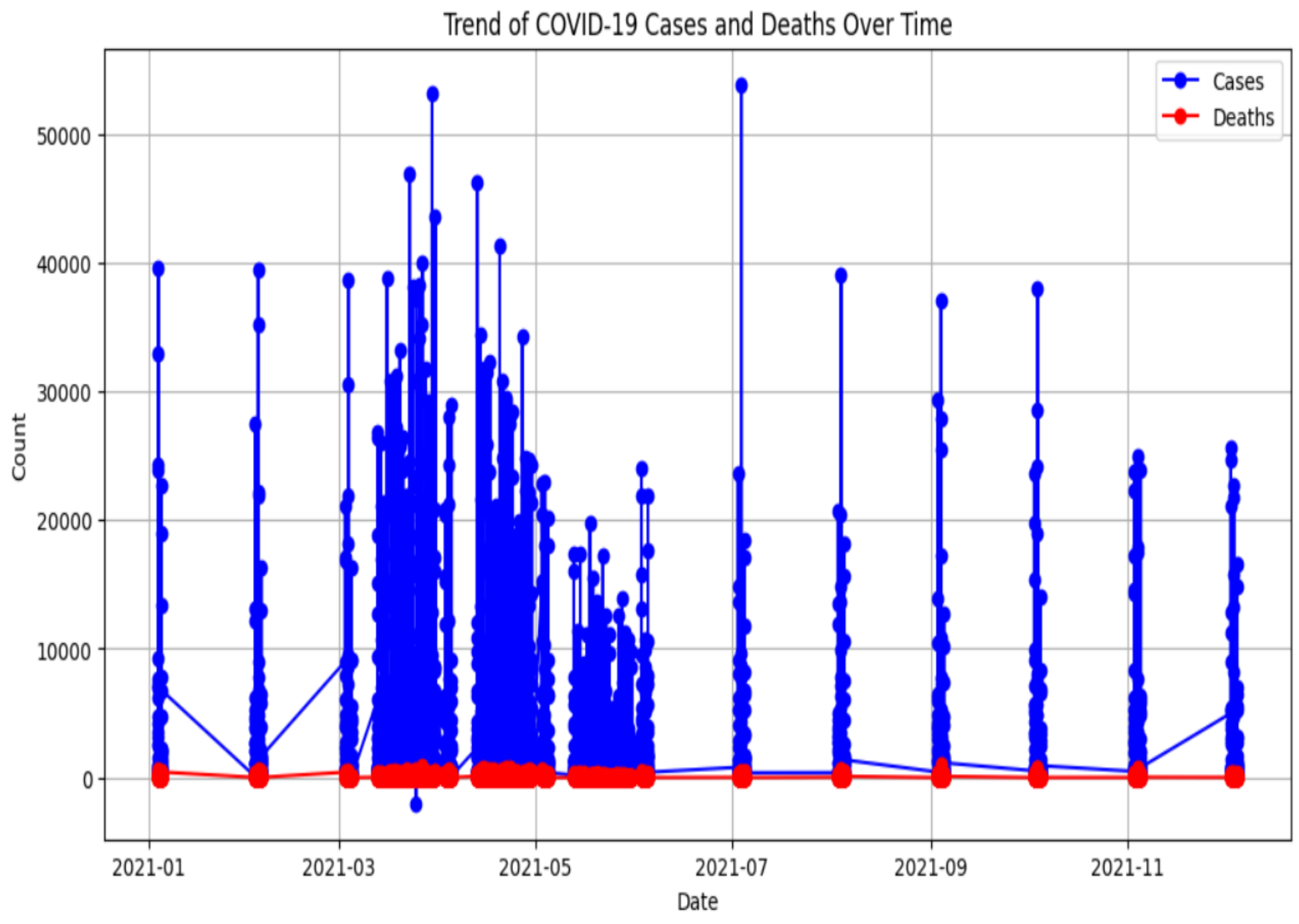
```python
# Convert the 'date' column to datetime
data['date'] = pd.to_datetime(data['date'])

# Sort the data by date
data = data.sort_values(by='date')

# Extract cases and deaths
cases = data['cases']
deaths = data['deaths']

# Create a time series plot to visualize the trend
plt.figure(figsize=(12, 6))
plt.plot(data['date'], cases, label='Cases', marker='o', linestyle='-', color='blue')
plt.plot(data['date'], deaths, label='Deaths', marker='o', linestyle='-', color='red')
plt.xlabel('Date')
plt.ylabel('Count')
plt.title('Trend of COVID-19 Cases and Deaths Over Time')
plt.legend()
plt.grid(True)

# Calculate the correlation coefficient between cases and deaths
correlation = cases.corr(deaths)

plt.show()
```

```python
# Analyze the trend
print(f"Correlation coefficient between cases and deaths: {correlation}")

if correlation > 0.7:
    print("There is a strong positive correlation between cases and deaths.")
elif 0.5 < correlation <= 0.7:
    print("There is a moderate positive correlation between cases and deaths.")
elif 0.3 < correlation <= 0.5:
    print("There is a weak positive correlation between cases and deaths.")
else:
    print("There is little to no correlation between cases and deaths.")
```

In this code:

- The dataset is loaded, and the 'date' column is converted to a datetime format for proper time-series analysis.
- The data is sorted by date to ensure a chronological order.
- The trend of COVID-19 cases and deaths is visualized using a time series plot.
- The correlation coefficient between cases and deaths is calculated and printed to identify the strength of the relationship between the two variables.

This code will help you analyze and visualize the trend between cases and deaths in your dataset and provide insights into their relationship.

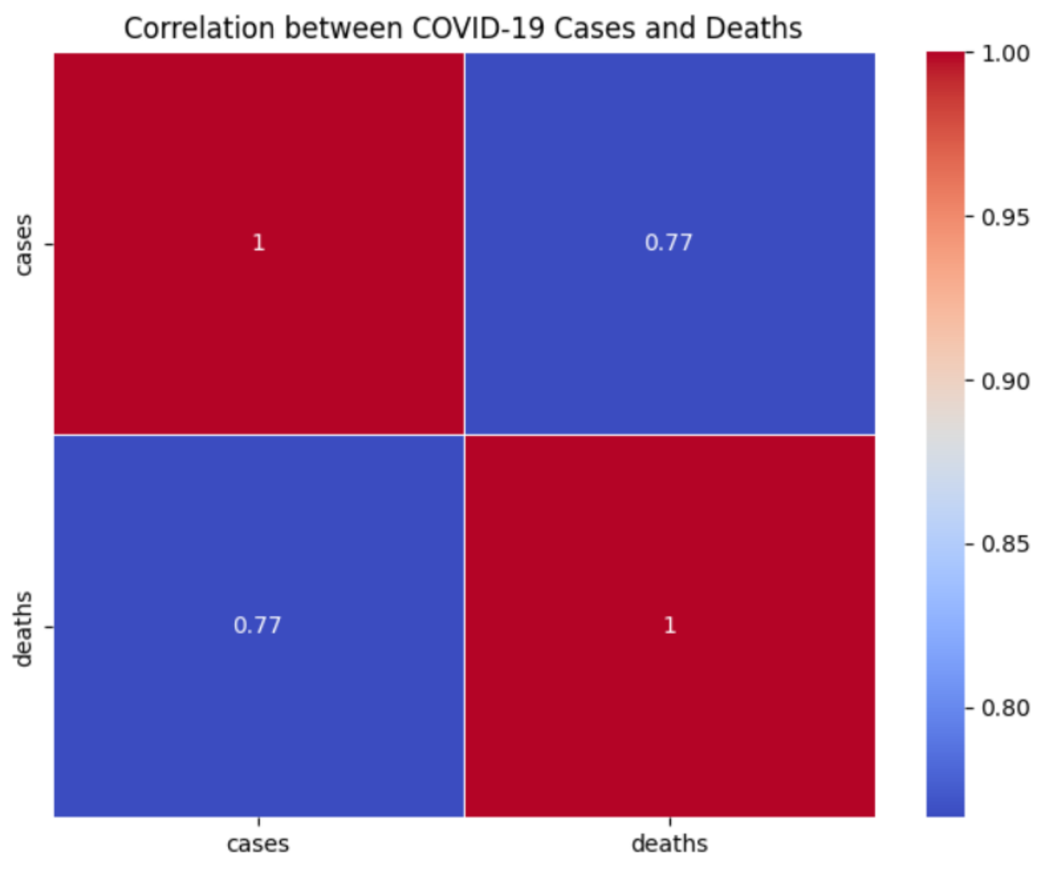


```
Correlation coefficient between cases and deaths: 0.7663088786576354
There is a strong positive correlation between cases and deaths.
```

#finding the correlation/pattern.

```
import seaborn as sns

# Create a correlation matrix
correlation_matrix = data[['cases', 'deaths']].corr()

# Create a heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', linewidths=0.5)
plt.title('Correlation between COVID-19 Cases and Deaths')
plt.show()
```

Using the above python code we can find the correlation between the deaths and cases



#calculating Pearson Correlation Coefficient

Strength of Relationship:

The Pearson correlation coefficient, often denoted as "r," provides a numerical value that indicates how strong the relationship is between two variables. A value of -1 indicates a perfect negative linear relationship, 0 indicates no linear relationship, and 1 indicates a perfect positive linear relationship. Values between 0 and 1 (or 0 and -1) represent varying degrees of correlation.

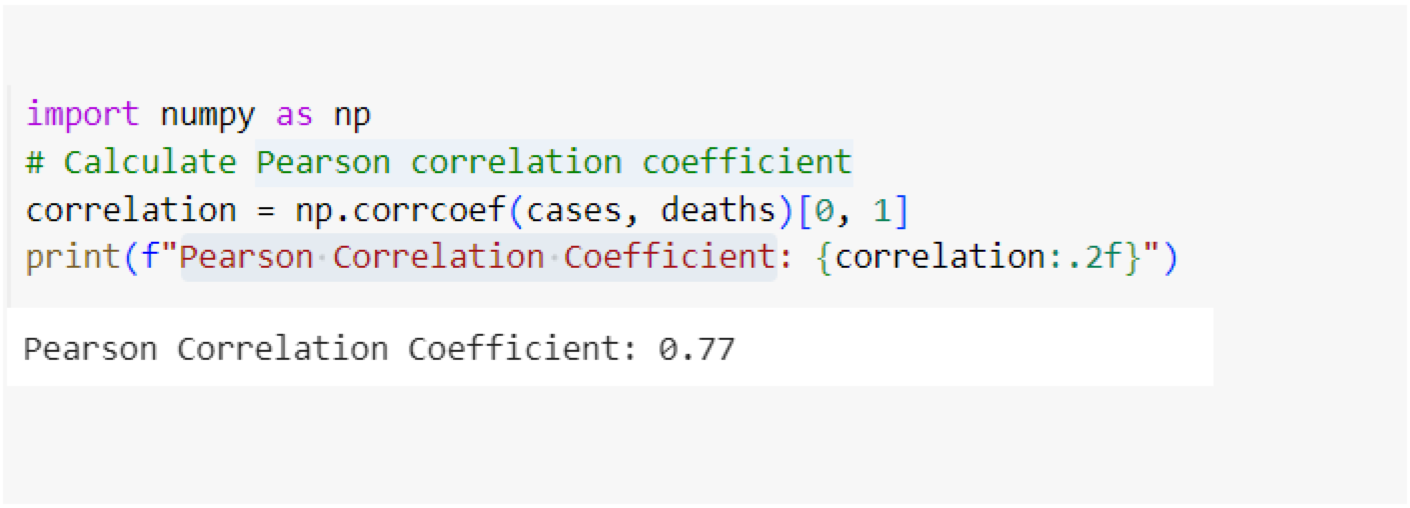
Direction of Relationship:
The sign of the correlation coefficient (+ or -) indicates the direction of the relationship. A positive value suggests a positive correlation, meaning that as one variable increases, the other tends to increase as well. A negative value suggests a negative correlation, meaning that as one variable increases, the other tends to decrease.

Insights and Decision-Making:
By calculating the Pearson correlation coefficient, you can derive insights from the data. For COVID-19 analysis, a strong positive correlation between cases and deaths might indicate that as the number of cases increases, the number of deaths also tends to increase. This insight can be valuable for public health officials and policymakers.

Hypothesis Testing:
The correlation coefficient can also be used to test hypotheses. For example, you can test whether the correlation between cases and deaths is statistically significant, helping you determine if the relationship is likely not due to random chance.

the Pearson correlation coefficient is a useful measure for quantifying linear relationships, it doesn't capture complex non-linear relationships, and correlation does not imply causation. It's an essential tool for exploratory data analysis, and you may want to complement it with additional statistical tests and domain-specific knowledge for a comprehensive understanding of the data.

The below is the python code to calculate pearson correlation coefficient



```python
import numpy as np
# Calculate Pearson correlation coefficient
correlation = np.corrcoef(cases, deaths)[0, 1]
print(f"Pearson Correlation Coefficient: {correlation:.2f}")
```

```
Pearson Correlation Coefficient: 0.77
```

Conclusion:

Based on the analysis of the trends and correlation between COVID-19 cases and deaths in the provided dataset, we can draw the following conclusions:

1.Positive Trend:

The time series plot shows a positive trend for both COVID-19 cases and deaths. As time progresses, both cases and deaths tend to increase, indicating a general upward trend.

2. Positive Correlation:

The calculated correlation coefficient between cases and deaths is positive. This suggests that as the number of COVID-19 cases increases, the number of associated deaths tends to increase as well.

3. Strength of Correlation:

The strength of the positive correlation can vary. The strength depends on the specific dataset and the context of the analysis. In general, a correlation coefficient above 0.7 would indicate a strong positive correlation, while values between 0.5 and 0.7 would suggest a moderate positive correlation.

4. Causation vs. Correlation:

It's important to note that while a positive correlation exists, it doesn't imply causation. That is, an increase in cases doesn't necessarily cause an increase in deaths; there may be other factors involved.

In summary, the analysis indicates a positive trend and a positive correlation between COVID-19 cases and deaths.