University of Roehampton London

CREATING DATA VISUALIZATION WITH CORONAVIRUS COVID-19 DATA USING PYTHON AND TABLEAU

Submitted by:

Name: Arko Saha

Student ID: SAH21527466

Course code: CMP020L013S

Course Title: Data Visualization

Programme Title: MSc Data Science

Programme Code: 48PN0096

Department: School of Arts and Digital Industries

Academic Year: 2022

Date of Submission: 17/04/2023

Author Note

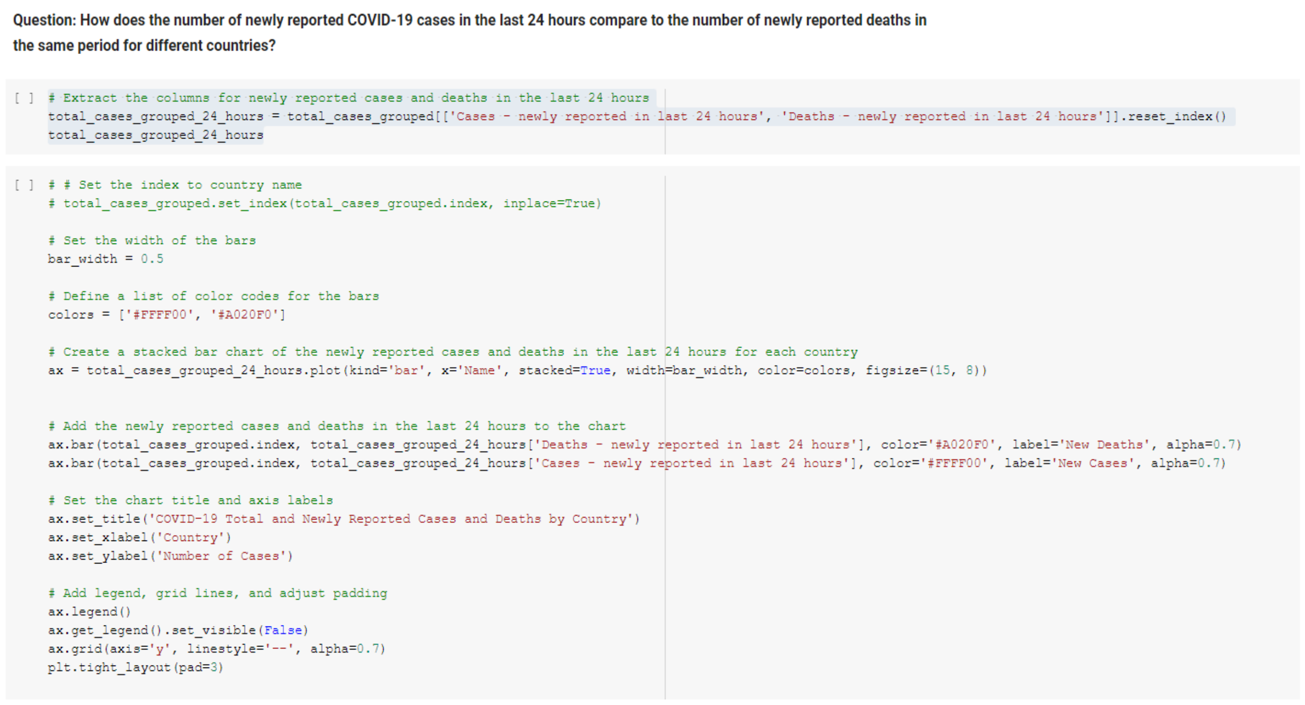
[These data visualizations were conducted under the scope of the final coursework for Data Visualization module in the spring term 2022-23]

University of Roehampton London

Task 1 – Python Visualization

This coursework served as an outstanding opportunity to enhance as well as refine the abilities in designing and building visualizations for specific data collection. We were given two sets of data visualization tasks - one with Python programming language and the other with Tableau. By undertaking these tasks, procuring the ability to identify a proper form of visualization technique case-by-case was one of the key takeaways, which can be vital in the future endeavors.

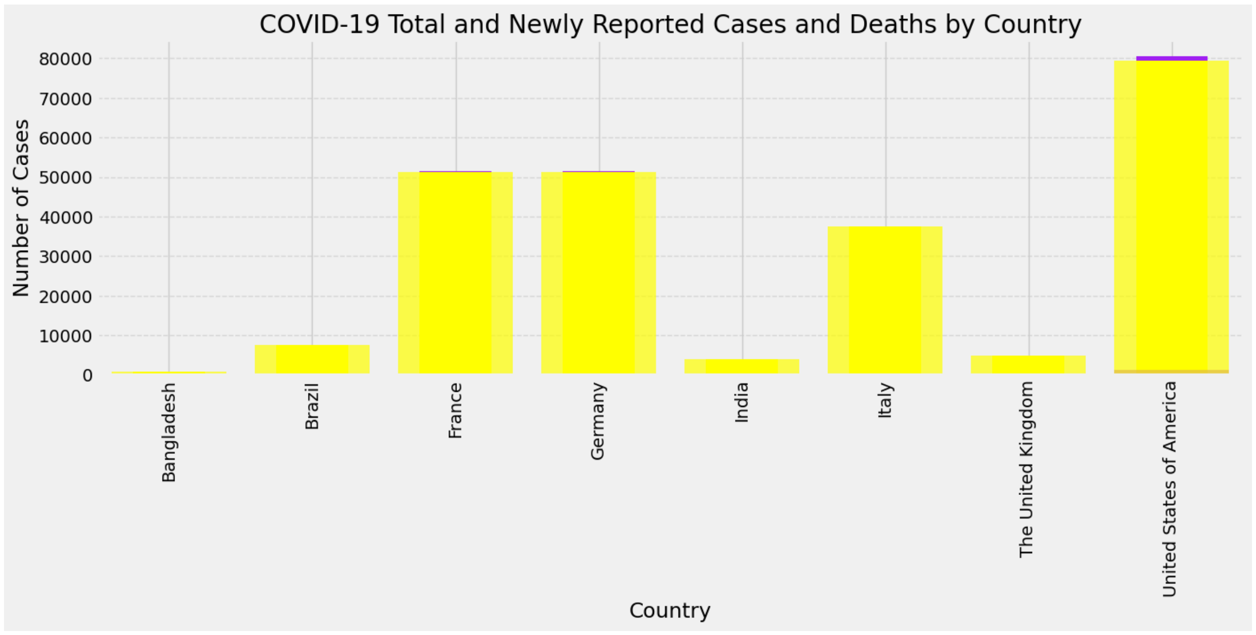
Question: How does the number of newly reported COVID-19 cases in the last 24 hours compare to the number of newly reported deaths in the same period for different countries?



Question: How does the number of newly reported COVID-19 cases in the last 24 hours compare to the number of newly reported deaths in the same period for different countries?

```
[ ] # Extract the columns for newly reported cases and deaths in the last 24 hours
    total_cases_grouped_24_hours = total_cases_grouped[['Cases - newly reported in last 24 hours', 'Deaths - newly reported in last 24 hours']].reset_index()
    total_cases_grouped_24_hours
```

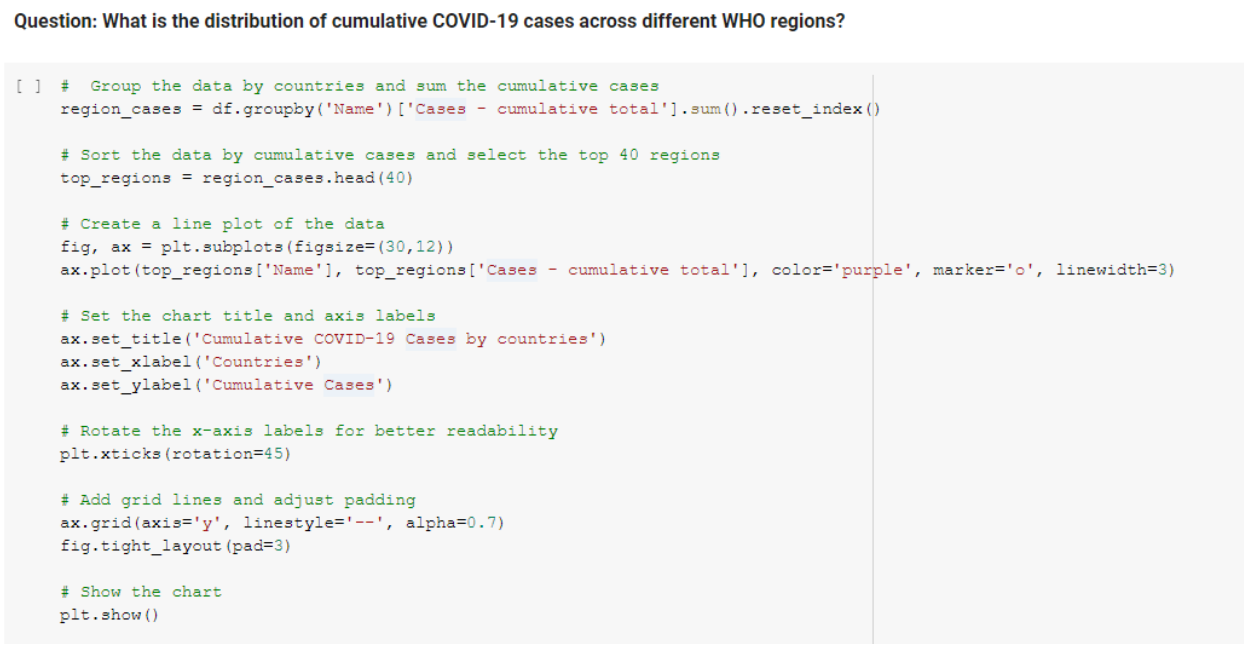
```
[ ] # # Set the index to country name
    # total_cases_grouped.set_index(total_cases_grouped.index, inplace=True)

    # Set the width of the bars
    bar_width = 0.5

    # Define a list of color codes for the bars
    colors = ['#FFFF00', '#A020F0']

    # Create a stacked bar chart of the newly reported cases and deaths in the last 24 hours for each country
    ax = total_cases_grouped_24_hours.plot(kind='bar', x='Name', stacked=True, width=bar_width, color=colors, figsize=(15, 8))


    # Add the newly reported cases and deaths in the last 24 hours to the chart
    ax.bar(total_cases_grouped.index, total_cases_grouped_24_hours['Deaths - newly reported in last 24 hours'], color='#A020F0', label='New Deaths', alpha=0.7)
    ax.bar(total_cases_grouped.index, total_cases_grouped_24_hours['Cases - newly reported in last 24 hours'], color='#FFFF00', label='New Cases', alpha=0.7)

    # Set the chart title and axis labels
    ax.set_title('COVID-19 Total and Newly Reported Cases and Deaths by Country')
    ax.set_xlabel('Country')
    ax.set_ylabel('Number of Cases')

    # Add legend, grid lines, and adjust padding
    ax.legend()
    ax.get_legend().set_visible(False)
    ax.grid(axis='y', linestyle='--', alpha=0.7)
    plt.tight_layout(pad=3)
```



This code creates a stacked bar chart showing the total and newly reported cases and deaths of COVID-19 by country. The data is first grouped by country and the columns for newly reported cases and deaths in the last 24 hours are extracted from the grouped data. This code provides a visual representation of the COVID-19 situation, allowing for easy comparison of total and newly reported cases and deaths by country. It can be useful for policymakers, researchers, and the public to understand and monitor the spread of COVID-19 across different regions.

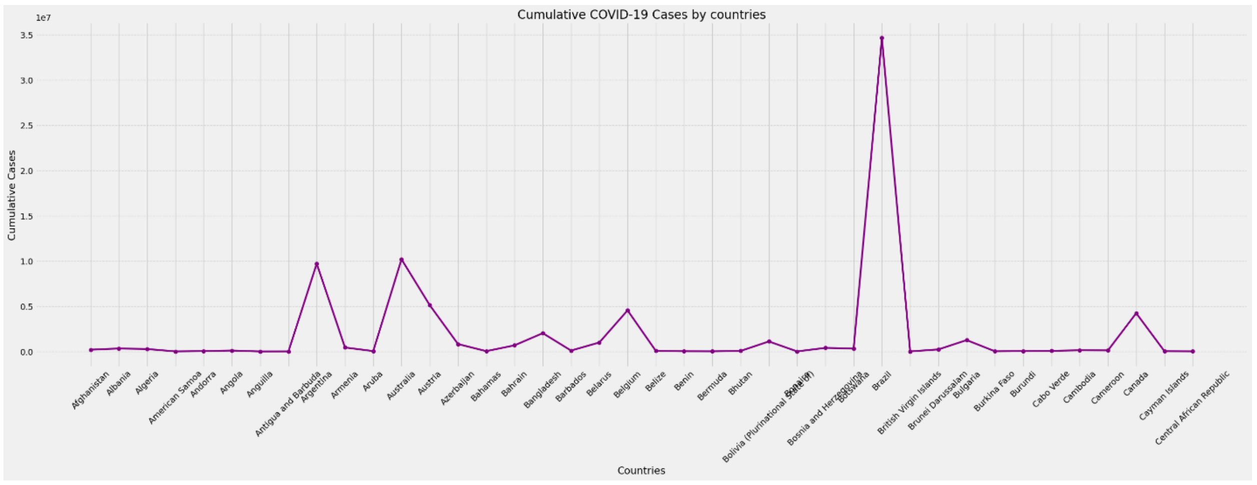
The stacked bar chart was chosen as a visual representation of the COVID-19 situation because it allows for easy comparison of the total and newly reported cases and deaths by country. By stacking the newly reported cases and deaths on top of the total cases and deaths, it is possible to see the overall trend and how the situation has changed in the last 24 hours.

Question: What is the distribution of cumulative COVID-19 cases across different WHO regions?



Question: What is the distribution of cumulative COVID-19 cases across different WHO regions?

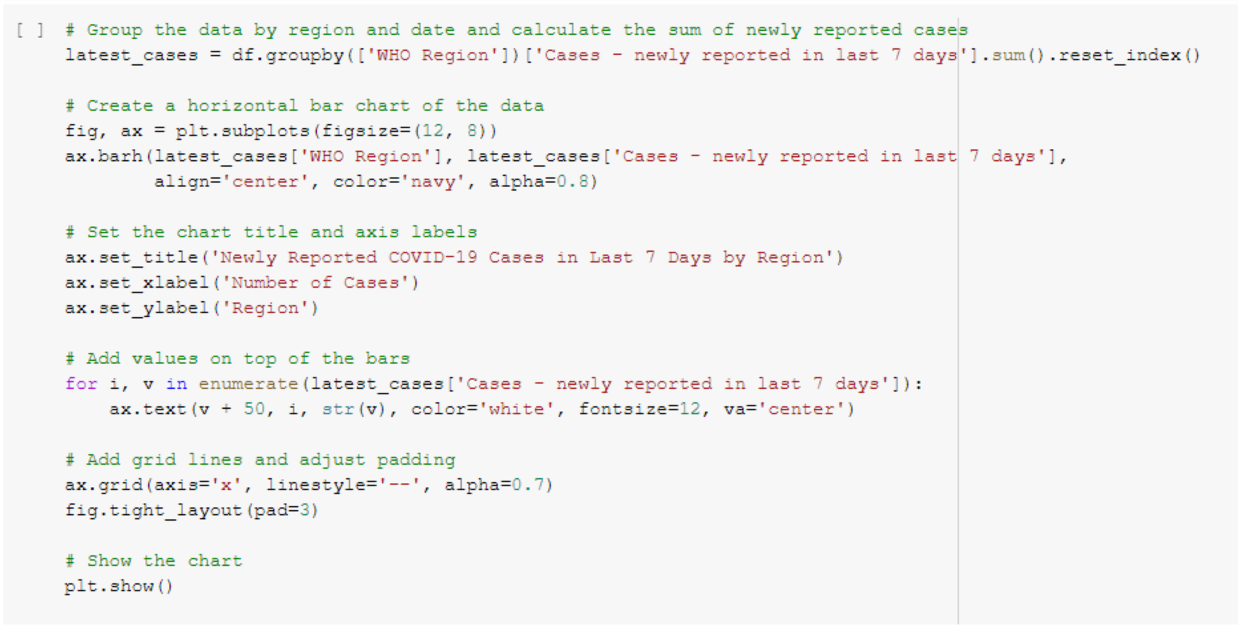
```
[ ] #  Group the data by countries and sum the cumulative cases
    region_cases = df.groupby('Name')['Cases - cumulative total'].sum().reset_index()

    # Sort the data by cumulative cases and select the top 40 regions
    top_regions = region_cases.head(40)

    # Create a line plot of the data
    fig, ax = plt.subplots(figsize=(30,12))
    ax.plot(top_regions['Name'], top_regions['Cases - cumulative total'], color='purple', marker='o', linewidth=3)

    # Set the chart title and axis labels
    ax.set_title('Cumulative COVID-19 Cases by countries')
    ax.set_xlabel('Countries')
    ax.set_ylabel('Cumulative Cases')

    # Rotate the x-axis labels for better readability
    plt.xticks(rotation=45)

    # Add grid lines and adjust padding
    ax.grid(axis='y', linestyle='--', alpha=0.7)
    fig.tight_layout(pad=3)

    # Show the chart
    plt.show()
```



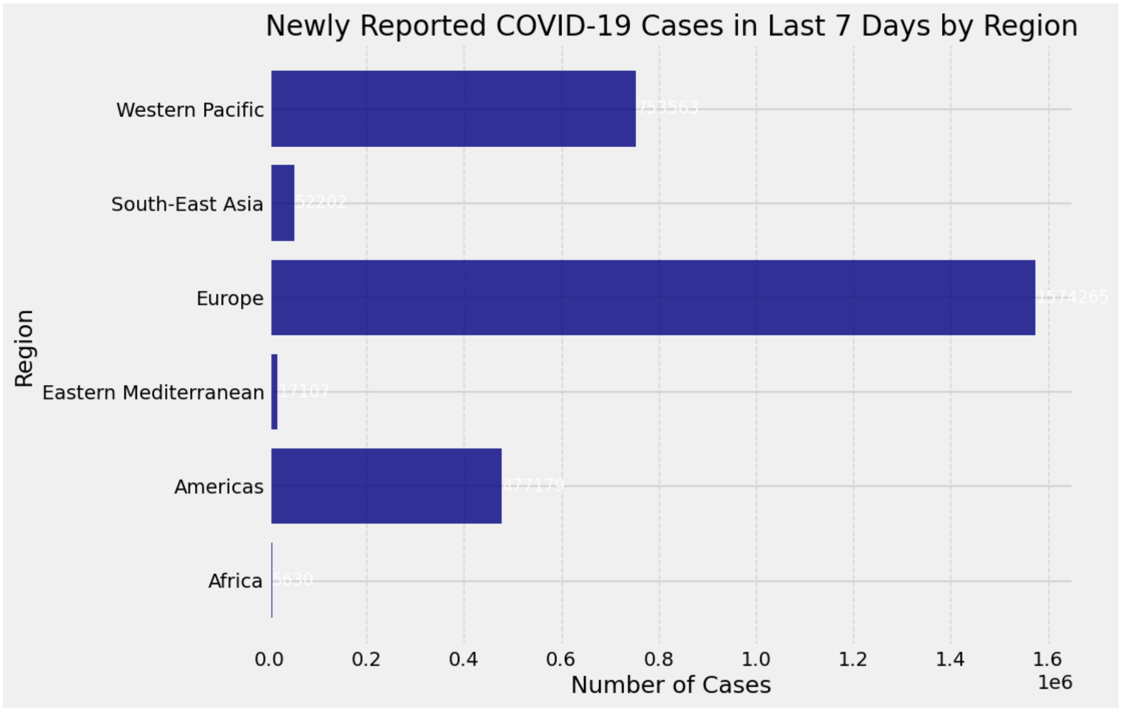
This code creates a line plot showing the cumulative number of COVID-19 cases for the top 40 countries. The purpose of this visualization is to show the countries that have been most affected by COVID-19 and the overall trend of cases across different countries.

The line plot is an effective visual representation for this data because it allows the viewer to quickly identify trends and patterns in the data, which is critical for understanding the severity of the COVID-19 pandemic in different regions of the world.

Question: How does the newly reported COVID-19 cases look like across different countries?



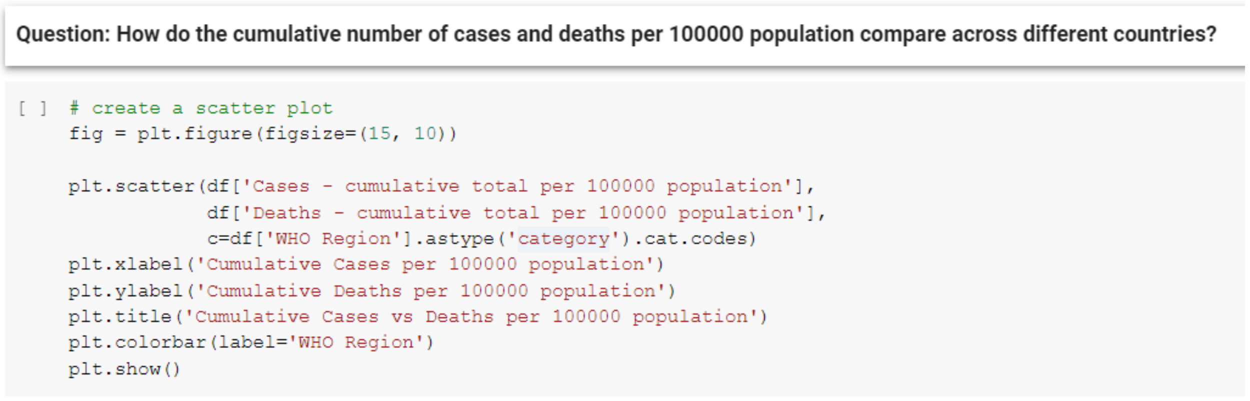
```
[ ] # Group the data by region and date and calculate the sum of newly reported cases
    latest_cases = df.groupby(['WHO Region'])['Cases - newly reported in last 7 days'].sum().reset_index()

    # Create a horizontal bar chart of the data
    fig, ax = plt.subplots(figsize=(12, 8))
    ax.barh(latest_cases['WHO Region'], latest_cases['Cases - newly reported in last 7 days'],
            align='center', color='navy', alpha=0.8)

    # Set the chart title and axis labels
    ax.set_title('Newly Reported COVID-19 Cases in Last 7 Days by Region')
    ax.set_xlabel('Number of Cases')
    ax.set_ylabel('Region')

    # Add values on top of the bars
    for i, v in enumerate(latest_cases['Cases - newly reported in last 7 days']):
        ax.text(v + 50, i, str(v), color='white', fontsize=12, va='center')

    # Add grid lines and adjust padding
    ax.grid(axis='x', linestyle='--', alpha=0.7)
    fig.tight_layout(pad=3)

    # Show the chart
    plt.show()
```



This code creates a horizontal bar chart to visualize the number of newly reported COVID-19 cases in the last 7 days by region. It allows for easy comparison between the different regions. Each bar represents a region, and the length of the bar represents the number of newly reported cases in the last 7 days. The bars are colored navy blue and have an alpha of 0.8 to improve visibility. The resulting plot helps in visualizing the current COVID-19 situation in different regions, making it easier to identify regions with higher numbers of cases.

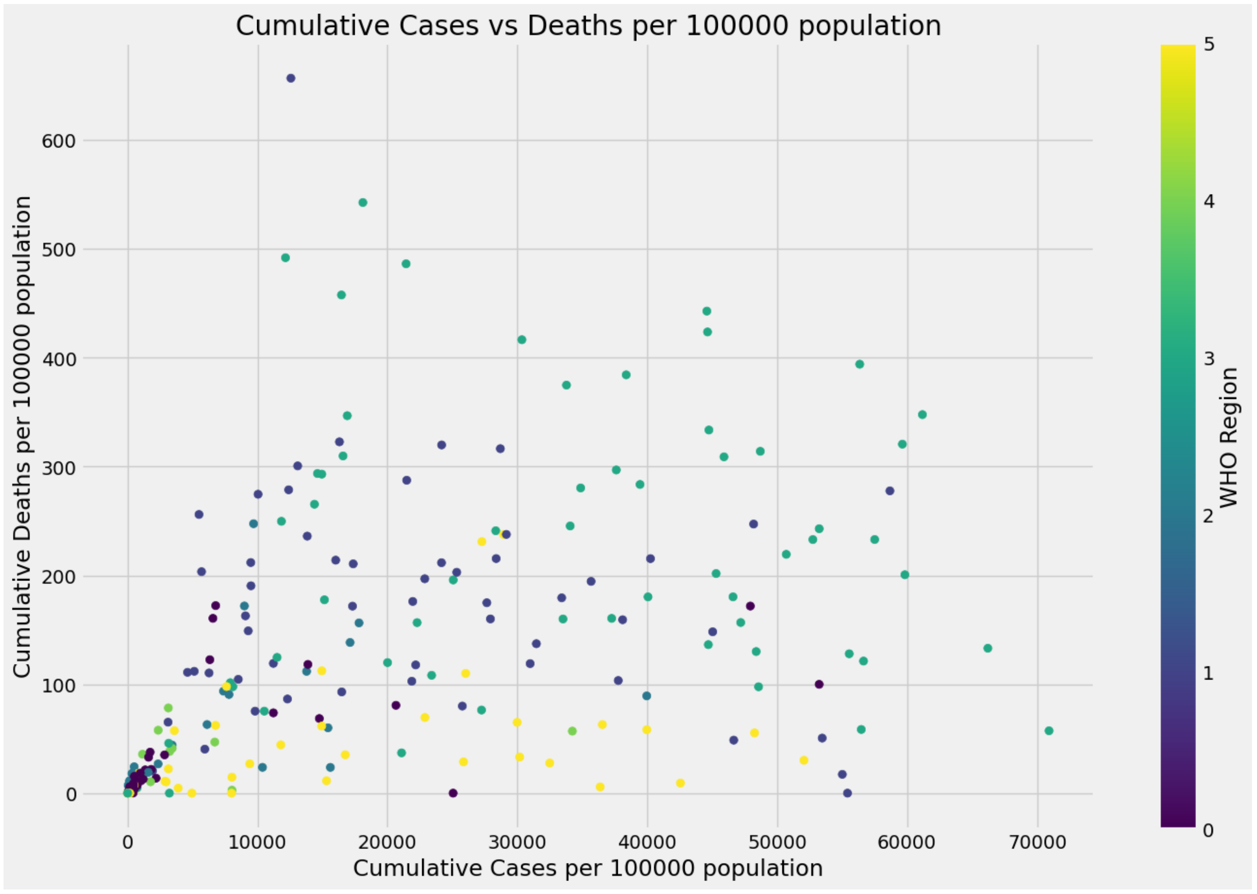
A horizontal bar chart was chosen to represent the newly reported COVID-19 cases in the last 7 days by region as it effectively displays the comparison between the different regions in terms of the number of cases. The horizontal orientation allows for longer labels to be displayed without overlapping, making it easier to read and interpret the data. Additionally, the bars are aligned to the center, making it easier to compare the values between the different regions. The use of color and the addition of values on top of the bars make it easier for the reader to distinguish and understand the data. Overall, a horizontal bar chart is an effective and intuitive way to present this type of data.

Question: How do the cumulative number of cases and deaths per 100000 population compare across different countries?



Question: How do the cumulative number of cases and deaths per 100000 population compare across different countries?

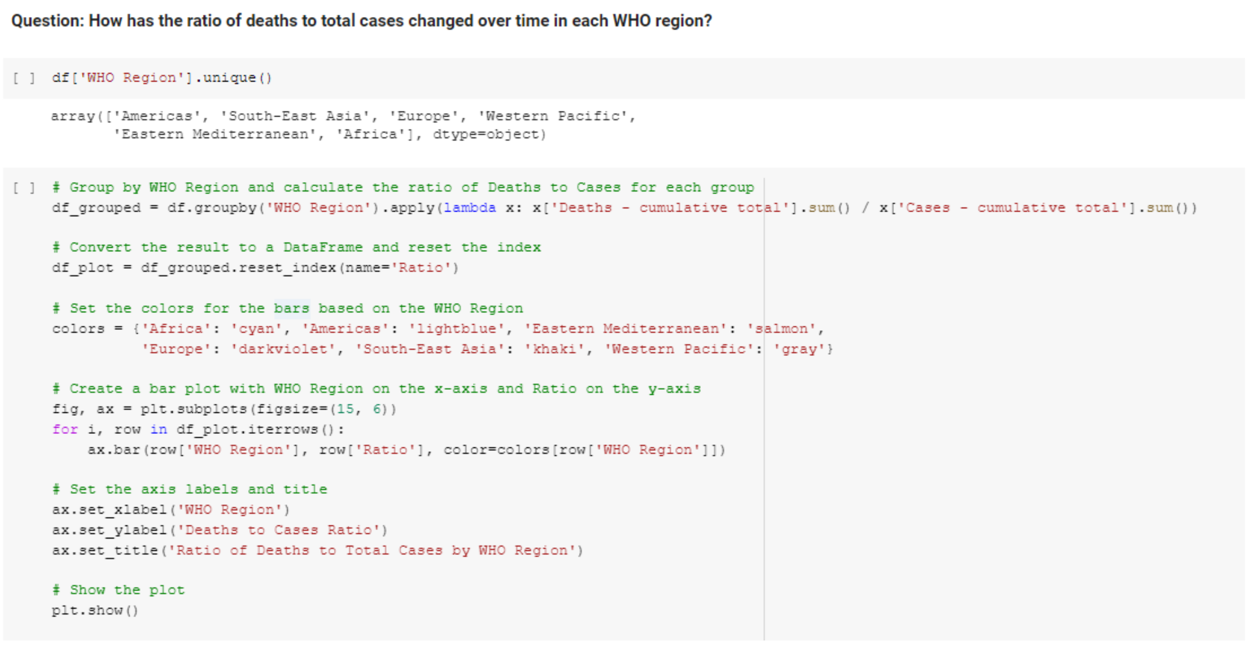
```
[ ] # create a scatter plot
    fig = plt.figure(figsize=(15, 10))

    plt.scatter(df['Cases - cumulative total per 100000 population'],
                df['Deaths - cumulative total per 100000 population'],
                c=df['WHO Region'].astype('category').cat.codes)
    plt.xlabel('Cumulative Cases per 100000 population')
    plt.ylabel('Cumulative Deaths per 100000 population')
    plt.title('Cumulative Cases vs Deaths per 100000 population')
    plt.colorbar(label='WHO Region')
    plt.show()
```



This code creates a scatter plot that displays the relationship between cumulative COVID-19 cases and deaths per 100,000 population across different regions of the world. The x-axis represents the cumulative number of cases per 100,000 population, and the y-axis represents the cumulative number of deaths per 100,000 population. Each point on the plot represents a country, and the color of the point corresponds to the region of the world to which the country belongs.

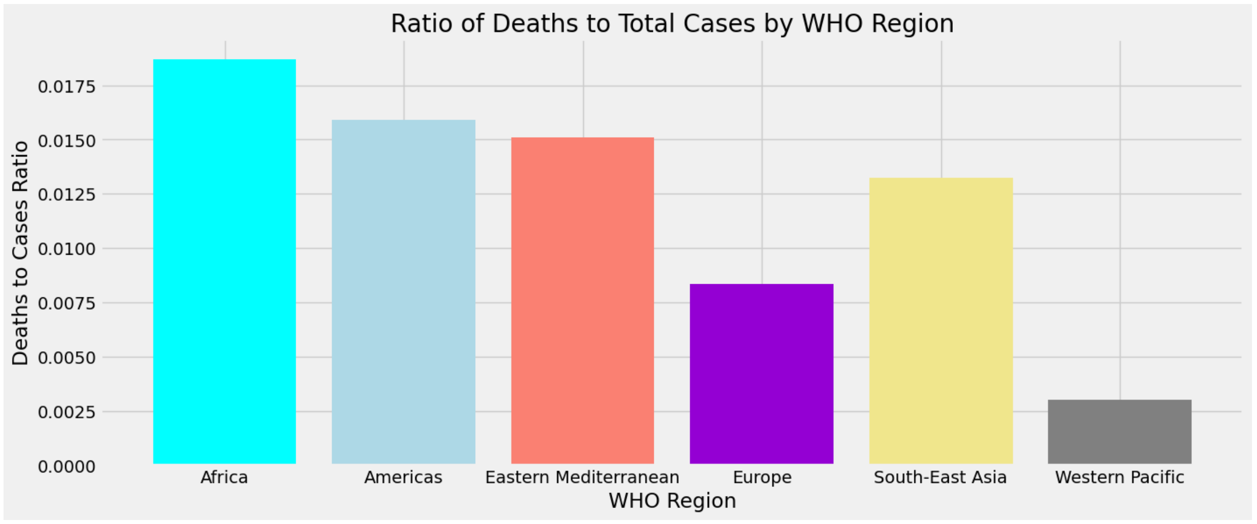
This visual representation was chosen because it allows us to visualize the relationship between two continuous variables, namely the cumulative cases per 100000 population and the cumulative deaths per 100000 population, and to see if there is a correlation between them. The use of color coding to represent the WHO regions adds an additional layer of information to the plot and makes it easier to interpret. The scatter plot allows us to see if there is a relationship between the two variables, and if so, how strong it is and whether there are any outliers or patterns in the data.

Question: How has the ratio of deaths to total cases changed over time in each WHO region?



Question: How has the ratio of deaths to total cases changed over time in each WHO region?

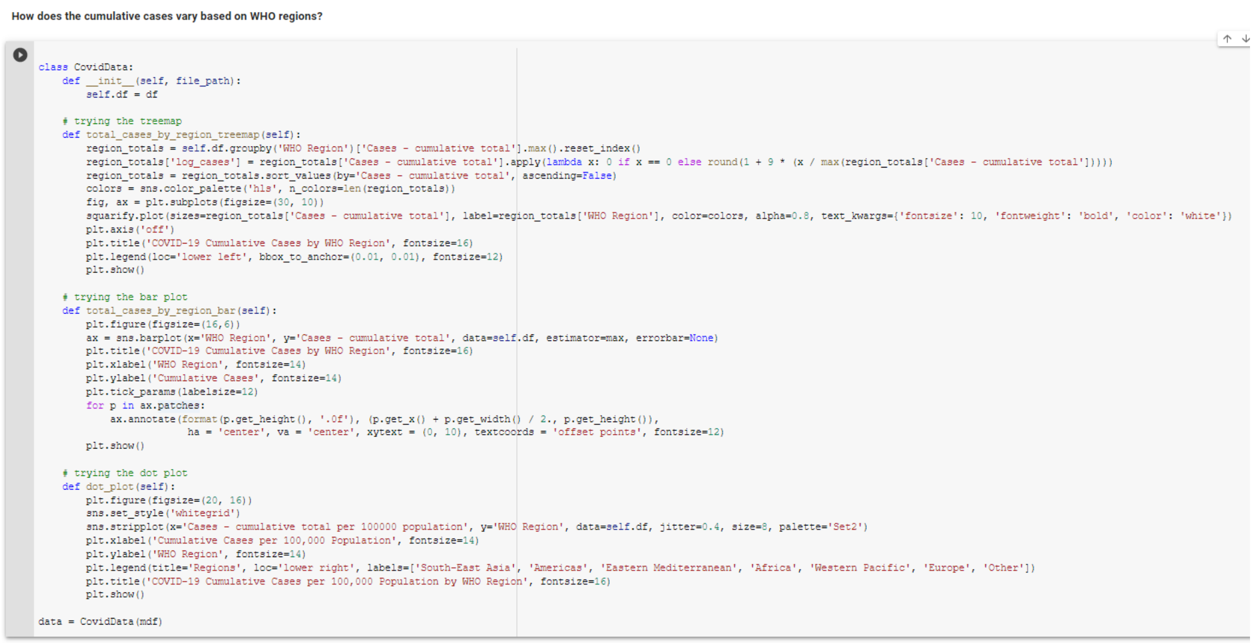
```
[ ]  df['WHO Region'].unique()

     array(['Americas', 'South-East Asia', 'Europe', 'Western Pacific',
            'Eastern Mediterranean', 'Africa'], dtype=object)

[ ]  # Group by WHO Region and calculate the ratio of Deaths to Cases for each group
     df_grouped = df.groupby('WHO Region').apply(lambda x: x['Deaths - cumulative total'].sum() / x['Cases - cumulative total'].sum())

     # Convert the result to a DataFrame and reset the index
     df_plot = df_grouped.reset_index(name='Ratio')

     # Set the colors for the bars based on the WHO Region
     colors = {'Africa': 'cyan', 'Americas': 'lightblue', 'Eastern Mediterranean': 'salmon',
               'Europe': 'darkviolet', 'South-East Asia': 'khaki', 'Western Pacific': 'gray'}

     # Create a bar plot with WHO Region on the x-axis and Ratio on the y-axis
     fig, ax = plt.subplots(figsize=(15, 6))
     for i, row in df_plot.iterrows():
         ax.bar(row['WHO Region'], row['Ratio'], color=colors[row['WHO Region']])

     # Set the axis labels and title
     ax.set_xlabel('WHO Region')
     ax.set_ylabel('Deaths to Cases Ratio')
     ax.set_title('Ratio of Deaths to Total Cases by WHO Region')

     # Show the plot
     plt.show()
```



The purpose of this code is to visualize the ratio of deaths to cases for each WHO Region, allowing for easy comparison across regions. By using a bar plot, the bars are more easily readable, and the color scheme provides an additional visual cue to distinguish between the different regions.

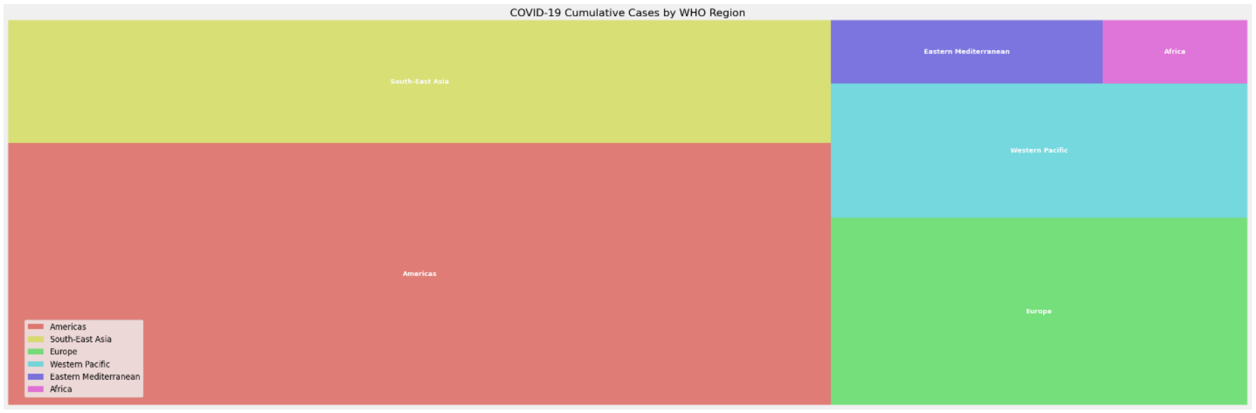
Some additional visualization problems

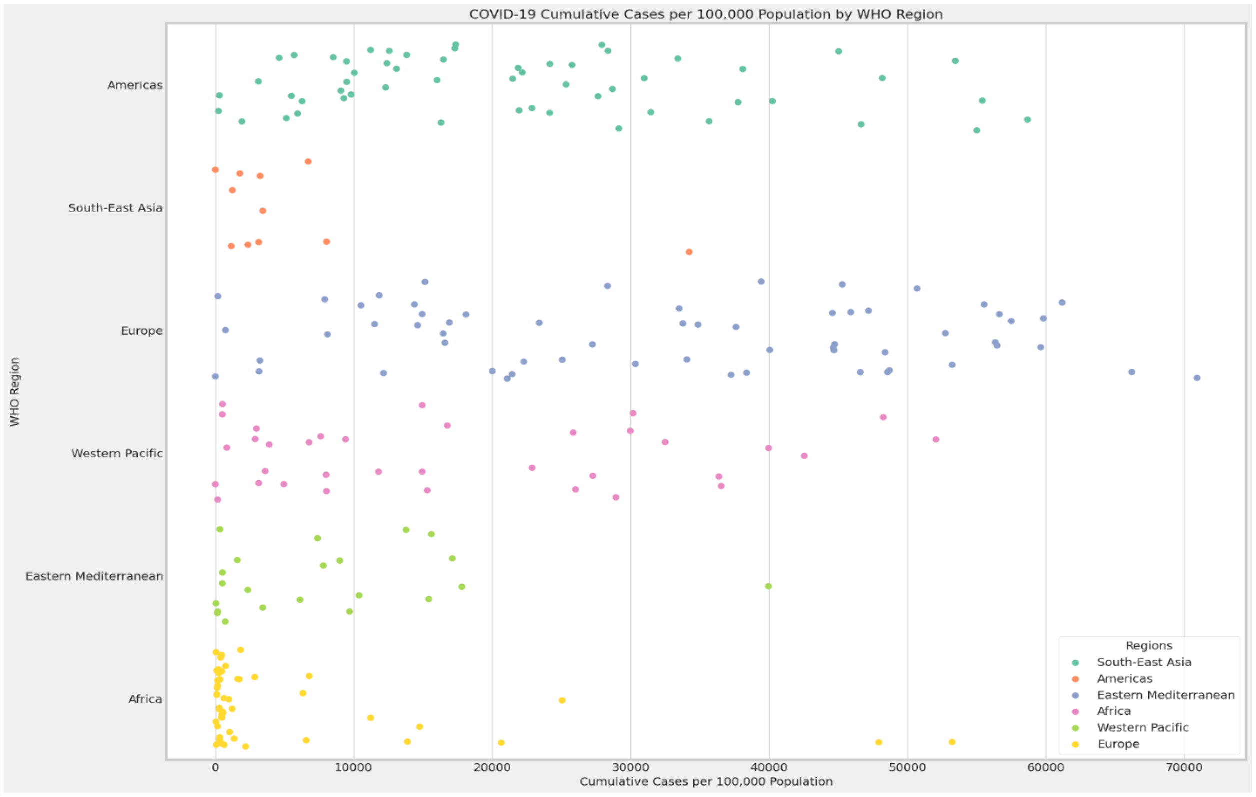
Question: How does the cumulative cases vary based on WHO regions?

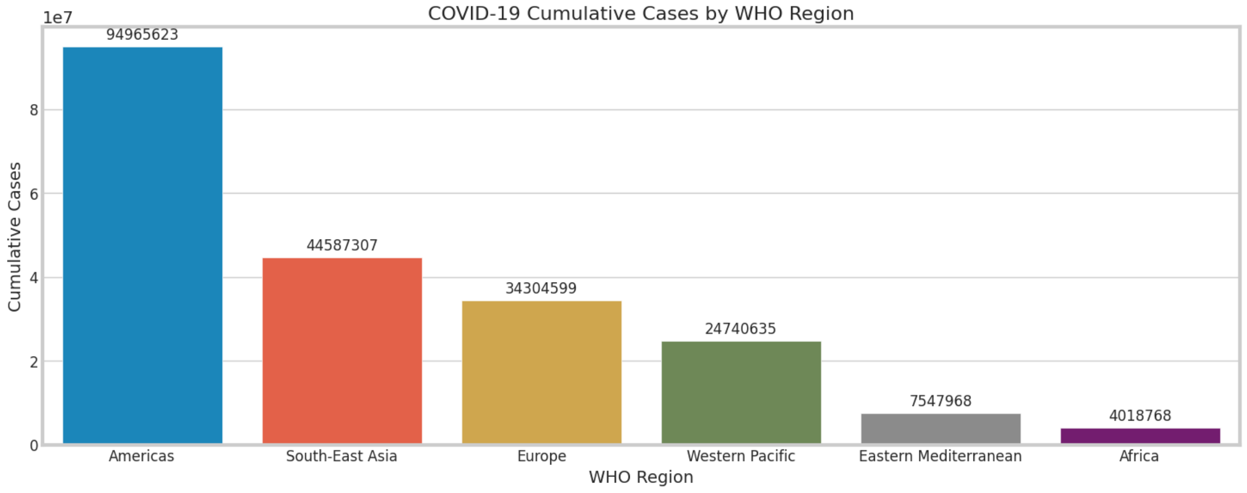


How does the cumulative cases vary based on WHO regions?

```python
class CovidData:
    def __init__(self, file_path):
        self.df = df

    # trying the treemap
    def total_cases_by_region_treemap(self):
        region_totals = self.df.groupby('WHO Region')['Cases - cumulative total'].max().reset_index()
        region_totals['log_cases'] = region_totals['Cases - cumulative total'].apply(lambda x: 0 if x == 0 else round(1 + 9 * (x / max(region_totals['Cases - cumulative total']))))
        region_totals = region_totals.sort_values(by='Cases - cumulative total', ascending=False)
        colors = sns.color_palette('hls', n_colors=len(region_totals))
        fig, ax = plt.subplots(figsize=(30, 10))
        squarify.plot(sizes=region_totals['Cases - cumulative total'], label=region_totals['WHO Region'], color=colors, alpha=0.8, text_kwargs={'fontsize': 10, 'fontweight': 'bold', 'color': 'white'})
        plt.axis('off')
        plt.title('COVID-19 Cumulative Cases by WHO Region', fontsize=16)
        plt.legend(loc='lower left', bbox_to_anchor=(0.01, 0.01), fontsize=12)
        plt.show()

    # trying the bar plot
    def total_cases_by_region_bar(self):
        plt.figure(figsize=(16,6))
        ax = sns.barplot(x='WHO Region', y='Cases - cumulative total', data=self.df, estimator=max, errorbar=None)
        plt.title('COVID-19 Cumulative Cases by WHO Region', fontsize=16)
        plt.xlabel('WHO Region', fontsize=14)
        plt.ylabel('Cumulative Cases', fontsize=14)
        plt.tick_params(labelsize=12)
        for p in ax.patches:
            ax.annotate(format(p.get_height(), '.0f'), (p.get_x() + p.get_width() / 2., p.get_height()),
                        ha = 'center', va = 'center', xytext = (0, 10), textcoords = 'offset points', fontsize=12)
        plt.show()

    # trying the dot plot
    def dot_plot(self):
        plt.figure(figsize=(20, 16))
        sns.set_style('whitegrid')
        sns.stripplot(x='Cases - cumulative total per 100000 population', y='WHO Region', data=self.df, jitter=0.4, size=8, palette='Set2')
        plt.xlabel('Cumulative Cases per 100,000 Population', fontsize=14)
        plt.ylabel('WHO Region', fontsize=14)
        plt.legend(title='Regions', loc='lower right', labels=['South-East Asia', 'Americas', 'Eastern Mediterranean', 'Africa', 'Western Pacific', 'Europe', 'Other'])
        plt.title('COVID-19 Cumulative Cases per 100,000 Population by WHO Region', fontsize=16)
        plt.show()

data = CovidData(mdf)
```


COVID-19 Cumulative Cases by WHO Region
South-East Asia
Eastern Mediterranean
Africa
Western Pacific
Americas
Europe
Americas
South-East Asia
Europe
Western Pacific
Eastern Mediterranean
Africa


COVID-19 Cumulative Cases per 100,000 Population by WHO Region
WHO Region
Americas
South-East Asia
Europe
Western Pacific
Eastern Mediterranean
Africa
0
10000
20000
30000
40000
50000
60000
70000
Cumulative Cases per 100,000 Population
Regions
South-East Asia
Americas
Eastern Mediterranean
Africa
Western Pacific
Europe


COVID-19 Cumulative Cases by WHO Region
1e7
94965623
44587307
34304599
24740635
7547968
4018768
Cumulative Cases
0
2
4
6
8
Americas
South-East Asia
Europe
Western Pacific
Eastern Mediterranean
Africa
WHO Region

The code defines a class ***CovidData*** with three methods that create different visualizations of COVID-19 data grouped by WHO regions:

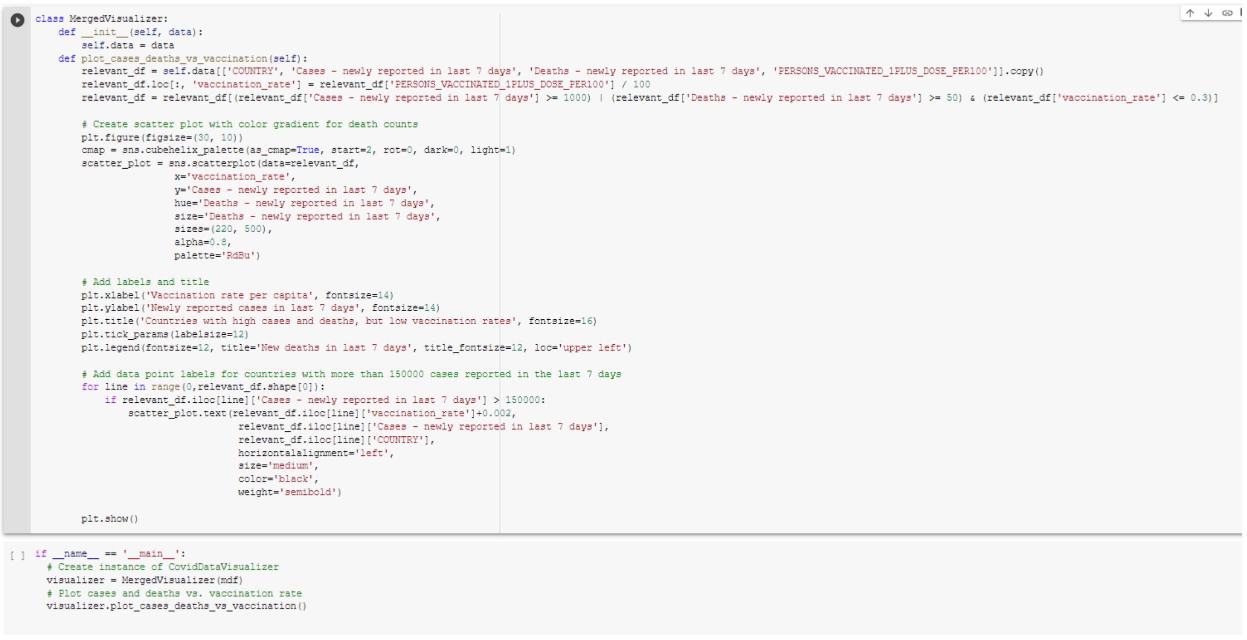
- Tree map
- Dot plot
- Bar plot

The bar plot was used to compare the total number of cases in different WHO regions. It is an effective way to compare values across different categories and can help identify any significant differences in the data.

The tree map was used to represent the total number of cases by WHO region, and the size of each block represented the number of cases. The tree map is an effective way to display hierarchical data and compare the relative sizes of different categories.

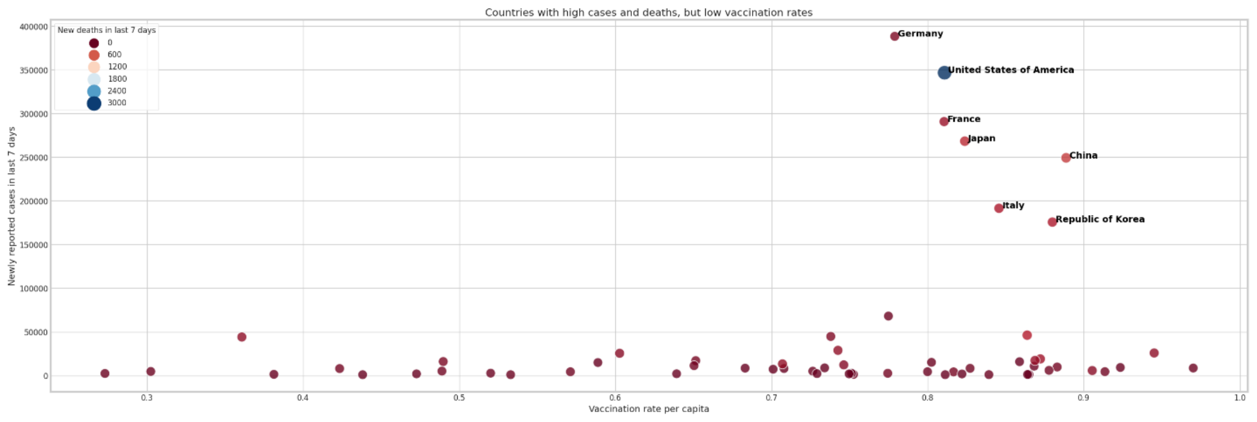
The dot plot was used to show the distribution of cases per 100,000 population in each WHO region. It is a useful way to show the distribution of data within each category and can help identify any outliers or clusters in the data.

Question: How does the cumulative cases vary based on WHO regions?



```
class MergedVisualizer:
    def __init__(self, data):
        self.data = data
    def plot_cases_deaths_vs_vaccination(self):
        relevant_df = self.data[['COUNTRY', 'Cases - newly reported in last 7 days', 'Deaths - newly reported in last 7 days', 'PERSONS_VACCINATED_1PLUS_DOSE_PER100']].copy()
        relevant_df.loc[:, 'vaccination_rate'] = relevant_df['PERSONS_VACCINATED_1PLUS_DOSE_PER100'] / 100
        relevant_df = relevant_df[(relevant_df['Cases - newly reported in last 7 days'] >= 1000) | (relevant_df['Deaths - newly reported in last 7 days'] >= 50) & (relevant_df['vaccination_rate'] <= 0.3)]

        # Create scatter plot with color gradient for death counts
        plt.figure(figsize=(30, 10))
        cmap = sns.cubehelix_palette(as_cmap=True, start=2, rot=0, dark=0, light=1)
        scatter_plot = sns.scatterplot(data=relevant_df,
                        x='vaccination_rate',
                        y='Cases - newly reported in last 7 days',
                        hue='Deaths - newly reported in last 7 days',
                        size='Deaths - newly reported in last 7 days',
                        sizes=(220, 500),
                        alpha=0.8,
                        palette='RdBu')

        # Add labels and title
        plt.xlabel('Vaccination rate per capita', fontsize=14)
        plt.ylabel('Newly reported cases in last 7 days', fontsize=14)
        plt.title('Countries with high cases and deaths, but low vaccination rates', fontsize=16)
        plt.tick_params(labelsize=12)
        plt.legend(fontsize=12, title='New deaths in last 7 days', title_fontsize=12, loc='upper left')

        # Add data point labels for countries with more than 150000 cases reported in the last 7 days
        for line in range(0,relevant_df.shape[0]):
            if relevant_df.iloc[line]['Cases - newly reported in last 7 days'] > 150000:
                scatter_plot.text(relevant_df.iloc[line]['vaccination_rate']+0.002,
                                  relevant_df.iloc[line]['Cases - newly reported in last 7 days'],
                                  relevant_df.iloc[line]['COUNTRY'],
                                  horizontalalignment='left',
                                  size='medium',
                                  color='black',
                                  weight='semibold')

        plt.show()
```

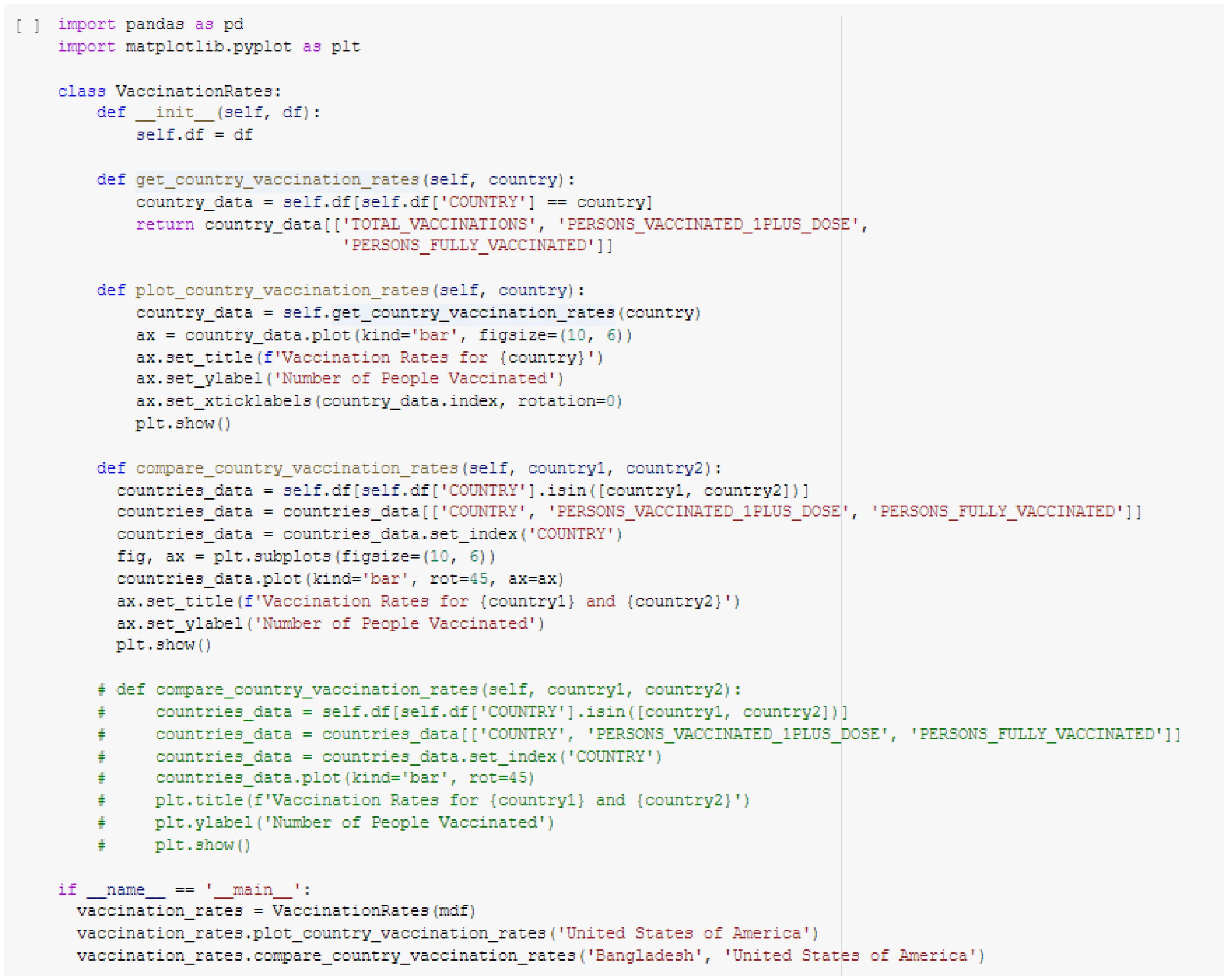
```
[ ] if __name__ == '__main__':
      # Create instance of CovidDataVisualizer
      visualizer = MergedVisualizer(mdf)
      # Plot cases and deaths vs. vaccination rate
      visualizer.plot_cases_deaths_vs_vaccination()
```



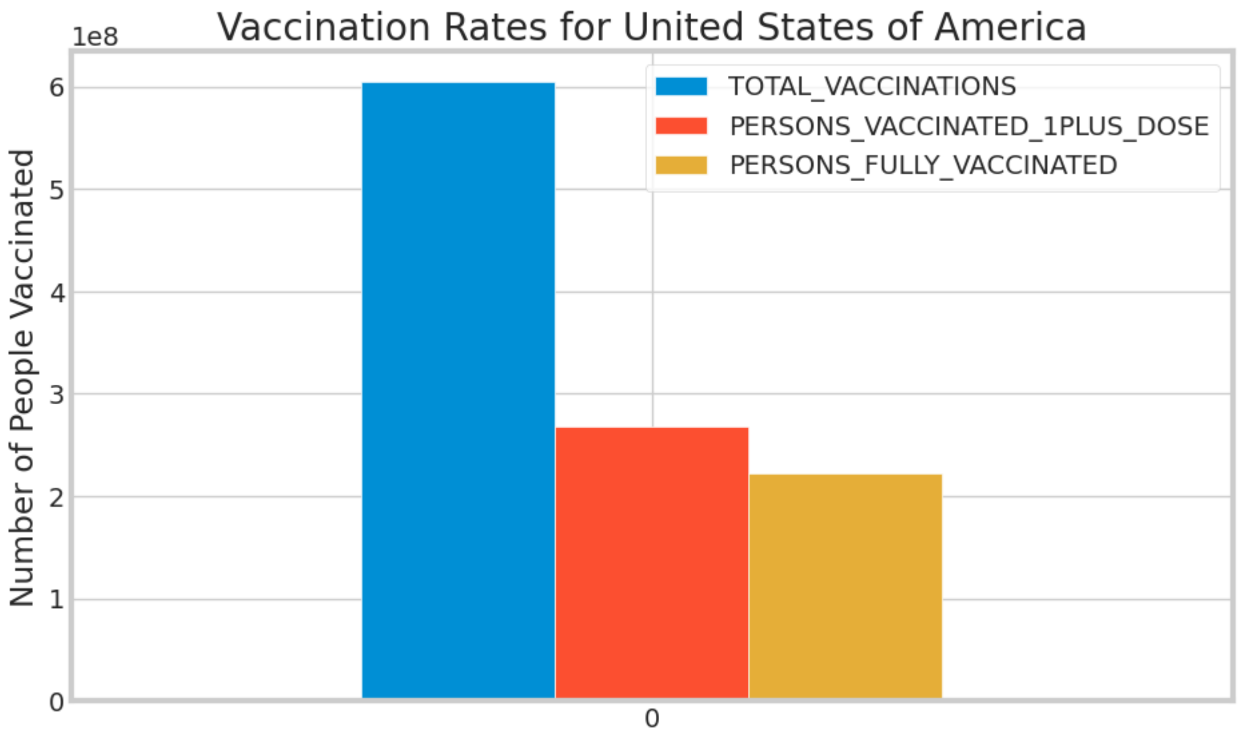
This code defines a class called ***MergedVisualizer*** that has a method ***plot_cases_deaths_vs_vaccination*** to create a scatter plot of COVID-19 cases and deaths reported in the last 7 days vs vaccination rate per capita for countries that have high cases and deaths but low vaccination rates. The scatter plot uses a color gradient to represent the number of newly reported deaths, with blue representing lower numbers and red representing higher numbers. The size of the data points also represents the number of newly reported deaths, with larger sizes indicating higher numbers. For better visualization, the selected countries have either at least 1000 newly reported cases in the last 7 days, or at least 50 newly reported deaths in the last 7 days, and have a vaccination rate less than or equal to 0.3.

This visualization was chosen because it provides an intuitive and easy-to-understand way to compare different countries based on their vaccination rates, case counts, and death counts. The color and size coding of the data points makes it easy to identify countries that have particularly high or low values for these metrics. Additionally, the labels for countries with more than 150,000 newly reported cases in the last 7 days provide additional information about the most severely affected countries.

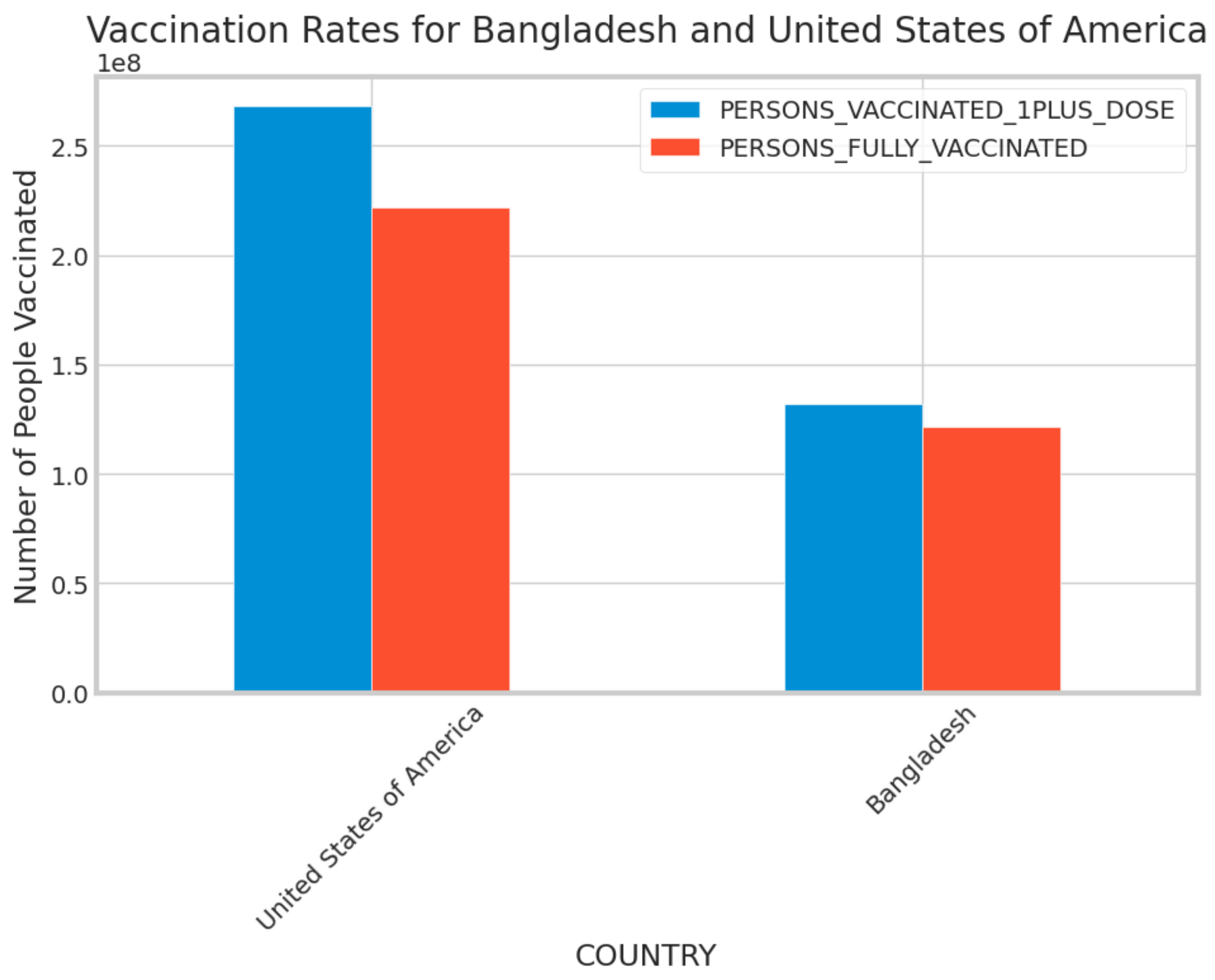
Question: How do vaccination rates (total vaccinations and fully vaccinated persons) vary across different countries?



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

    class VaccinationRates:
        def __init__(self, df):
            self.df = df

        def get_country_vaccination_rates(self, country):
            country_data = self.df[self.df['COUNTRY'] == country]
            return country_data[['TOTAL_VACCINATIONS', 'PERSONS_VACCINATED_1PLUS_DOSE',
                                 'PERSONS_FULLY_VACCINATED']]

        def plot_country_vaccination_rates(self, country):
            country_data = self.get_country_vaccination_rates(country)
            ax = country_data.plot(kind='bar', figsize=(10, 6))
            ax.set_title(f'Vaccination Rates for {country}')
            ax.set_ylabel('Number of People Vaccinated')
            ax.set_xticklabels(country_data.index, rotation=0)
            plt.show()

        def compare_country_vaccination_rates(self, country1, country2):
          countries_data = self.df[self.df['COUNTRY'].isin([country1, country2])]
          countries_data = countries_data[['COUNTRY', 'PERSONS_VACCINATED_1PLUS_DOSE', 'PERSONS_FULLY_VACCINATED']]
          countries_data = countries_data.set_index('COUNTRY')
          fig, ax = plt.subplots(figsize=(10, 6))
          countries_data.plot(kind='bar', rot=45, ax=ax)
          ax.set_title(f'Vaccination Rates for {country1} and {country2}')
          ax.set_ylabel('Number of People Vaccinated')
          plt.show()

        # def compare_country_vaccination_rates(self, country1, country2):
        #     countries_data = self.df[self.df['COUNTRY'].isin([country1, country2])]
        #     countries_data = countries_data[['COUNTRY', 'PERSONS_VACCINATED_1PLUS_DOSE', 'PERSONS_FULLY_VACCINATED']]
        #     countries_data = countries_data.set_index('COUNTRY')
        #     countries_data.plot(kind='bar', rot=45)
        #     plt.title(f'Vaccination Rates for {country1} and {country2}')
        #     plt.ylabel('Number of People Vaccinated')
        #     plt.show()

    if __name__ == '__main__':
      vaccination_rates = VaccinationRates(mdf)
      vaccination_rates.plot_country_vaccination_rates('United States of America')
      vaccination_rates.compare_country_vaccination_rates('Bangladesh', 'United States of America')
```


Vaccination Rates for United States of America
1e8
Number of People Vaccinated
0
1
2
3
4
5
6
0
TOTAL_VACCINATIONS
PERSONS_VACCINATED_1PLUS_DOSE
PERSONS_FULLY_VACCINATED

.



This code analyzes and visualizes COVID-19 vaccination rates for different countries. The main objective is to help people understand how many people in each country have received at least one dose of the COVID-19 vaccine, as well as how many people have been fully vaccinated. The class called ***"VaccinationRates"*** has three methods:

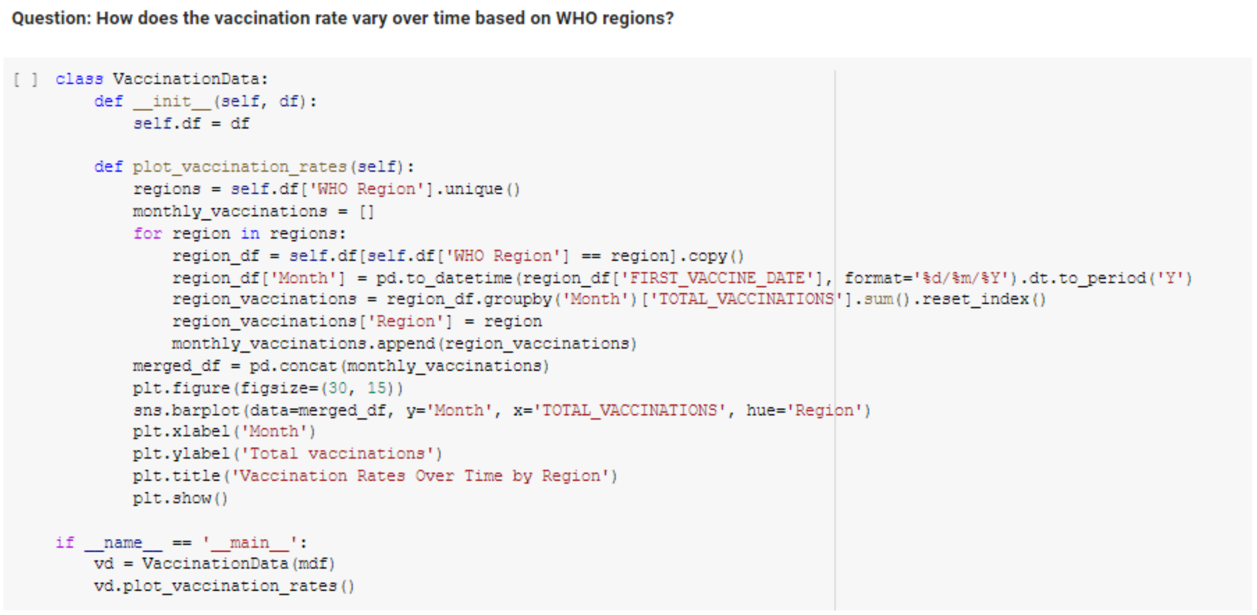
- ***"get_country_vaccination_rates"*** takes a country name as input and returns the number of people vaccinated in that country.
- ***"plot_country_vaccination_rates"*** takes a country name as input and plots a bar chart showing the number of people vaccinated in that country.

- ***"compare_country_vaccination_rates"*** takes two country names as input and plots a bar chart comparing the number of people vaccinated in those two countries.

In the ***plot_country_vaccination_rates*** function, a bar chart is used to represent the vaccination rates for a single country. Bar charts are a common choice for visualizing categorical data such as the number of people vaccinated. The bars are arranged vertically or horizontally, with each bar representing a category, in this case, different types of vaccinations. The length of each bar represents the quantity or value associated with the category. Therefore, the height of each bar shows the number of people vaccinated.

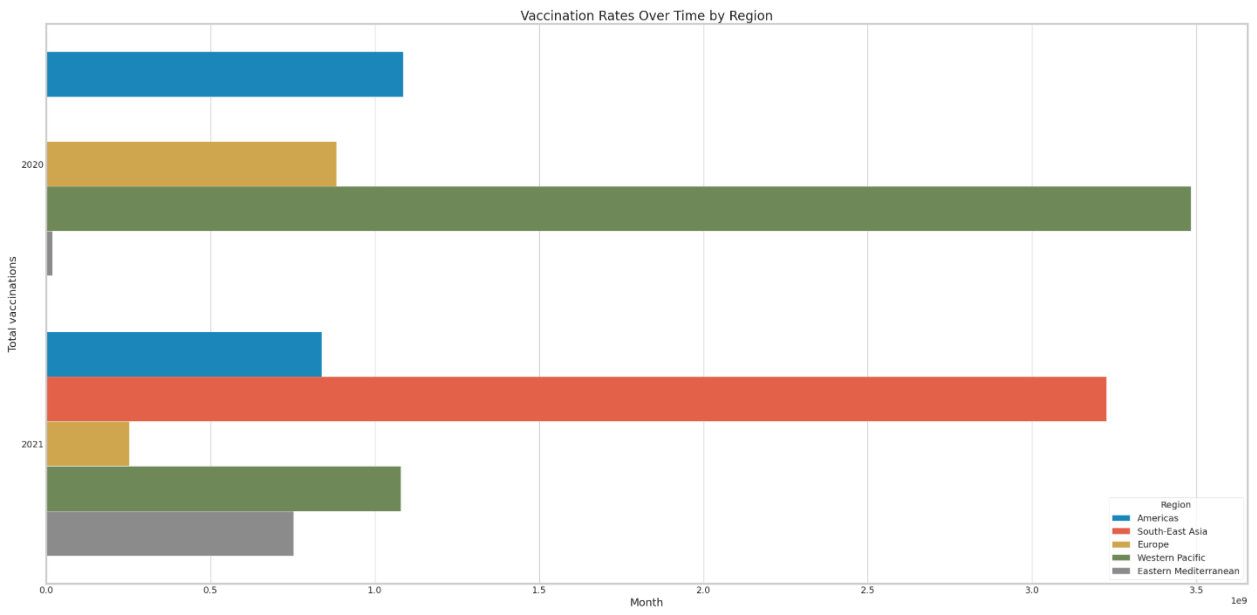
In the ***compare_country_vaccination_rates*** function, a grouped bar chart is used to compare the vaccination rates between two countries. Grouped bar charts show bars for different groups, side-by-side. Each group consists of multiple bars representing the different categories, in this case, different types of vaccinations. By using a grouped bar chart, it is easy to visually compare the vaccination rates of different types between two countries. The length of each bar within a group represents the quantity or value associated with the category, and the height of each group shows the total number of people vaccinated in each country.

Question: How does the vaccination rate vary over time based on WHO regions?



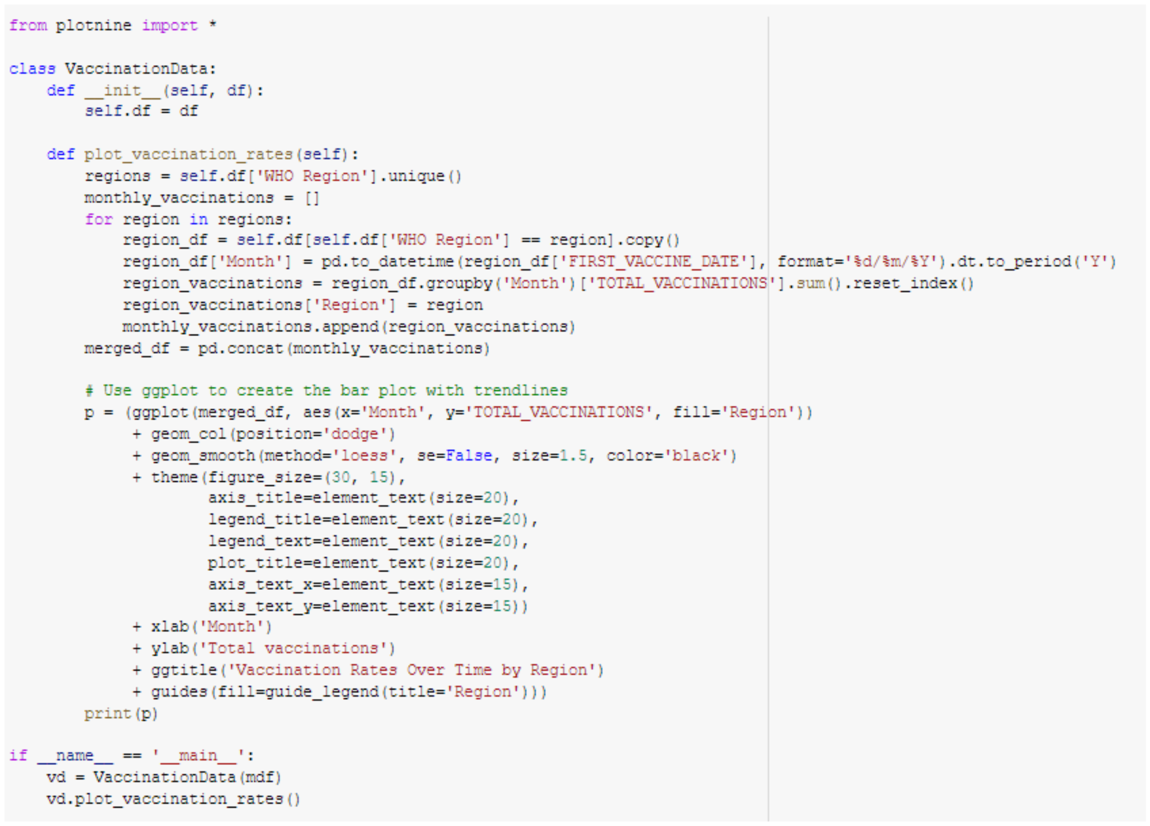
Question: How does the vaccination rate vary over time based on WHO regions?

```
[ ] class VaccinationData:
        def __init__(self, df):
            self.df = df

        def plot_vaccination_rates(self):
            regions = self.df['WHO Region'].unique()
            monthly_vaccinations = []
            for region in regions:
                region_df = self.df[self.df['WHO Region'] == region].copy()
                region_df['Month'] = pd.to_datetime(region_df['FIRST_VACCINE_DATE'], format='%d/%m/%Y').dt.to_period('Y')
                region_vaccinations = region_df.groupby('Month')['TOTAL_VACCINATIONS'].sum().reset_index()
                region_vaccinations['Region'] = region
                monthly_vaccinations.append(region_vaccinations)
            merged_df = pd.concat(monthly_vaccinations)
            plt.figure(figsize=(30, 15))
            sns.barplot(data=merged_df, y='Month', x='TOTAL_VACCINATIONS', hue='Region')
            plt.xlabel('Month')
            plt.ylabel('Total vaccinations')
            plt.title('Vaccination Rates Over Time by Region')
            plt.show()

    if __name__ == '__main__':
        vd = VaccinationData(mdf)
        vd.plot_vaccination_rates()
```

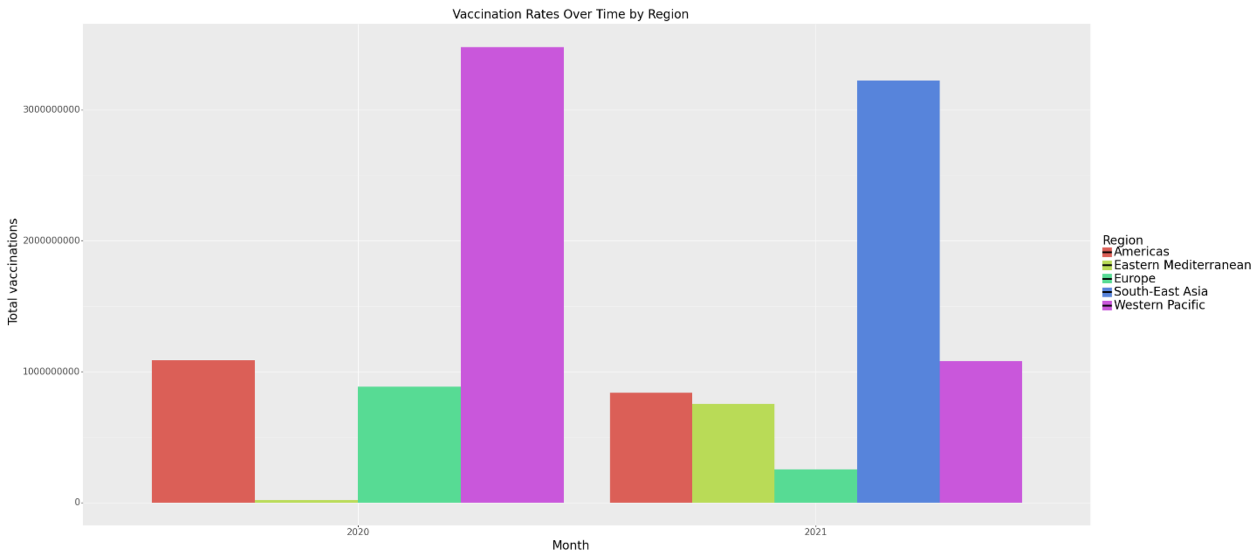


This code analyzes vaccination data and visualizes it to see how many people are getting vaccinated over time in different regions of the world. It does this by first grouping the data by region and month and then adding up the total number of vaccinations in each month for each region. It then merges all the data together and creates a bar chart that shows the total vaccinations for each region over time. This helps to see how vaccination rates are changing over time in different parts of the world. The code is using the seaborn library to create the bar chart, which allows for customization of the chart's appearance.

However, the following is the code snippet how I transformed the similar code using "*ggplot*" library on python:



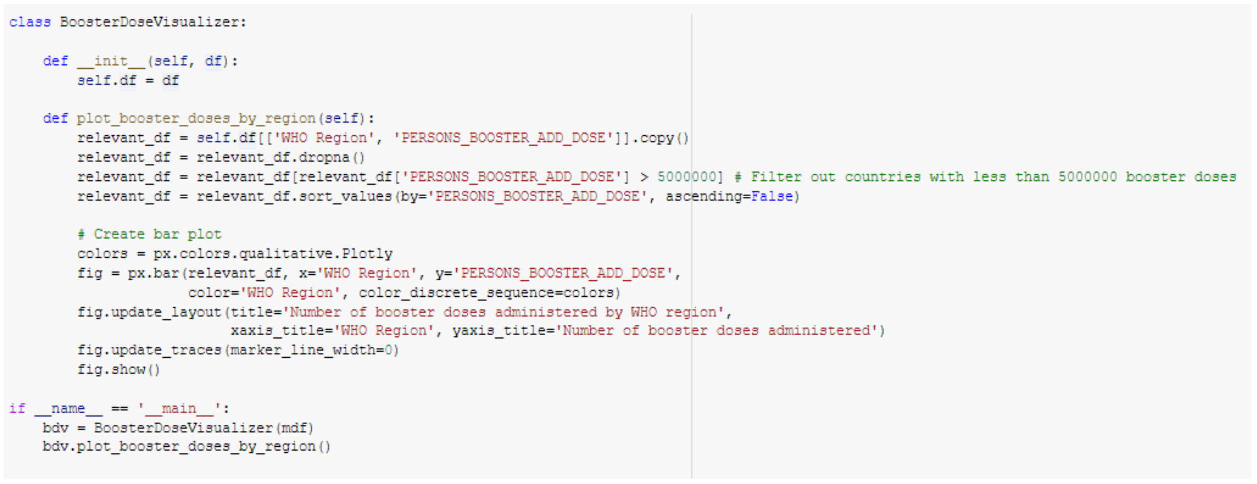
```python
from plotnine import *

class VaccinationData:
    def __init__(self, df):
        self.df = df

    def plot_vaccination_rates(self):
        regions = self.df['WHO Region'].unique()
        monthly_vaccinations = []
        for region in regions:
            region_df = self.df[self.df['WHO Region'] == region].copy()
            region_df['Month'] = pd.to_datetime(region_df['FIRST_VACCINE_DATE'], format='%d/%m/%Y').dt.to_period('Y')
            region_vaccinations = region_df.groupby('Month')['TOTAL_VACCINATIONS'].sum().reset_index()
            region_vaccinations['Region'] = region
            monthly_vaccinations.append(region_vaccinations)
        merged_df = pd.concat(monthly_vaccinations)

        # Use ggplot to create the bar plot with trendlines
        p = (ggplot(merged_df, aes(x='Month', y='TOTAL_VACCINATIONS', fill='Region'))
             + geom_col(position='dodge')
             + geom_smooth(method='loess', se=False, size=1.5, color='black')
             + theme(figure_size=(30, 15),
                     axis_title=element_text(size=20),
                     legend_title=element_text(size=20),
                     legend_text=element_text(size=20),
                     plot_title=element_text(size=20),
                     axis_text_x=element_text(size=15),
                     axis_text_y=element_text(size=15))
             + xlab('Month')
             + ylab('Total vaccinations')
             + ggtitle('Vaccination Rates Over Time by Region')
             + guides(fill=guide_legend(title='Region')))
        print(p)

if __name__ == '__main__':
    vd = VaccinationData(mdf)
    vd.plot_vaccination_rates()
```



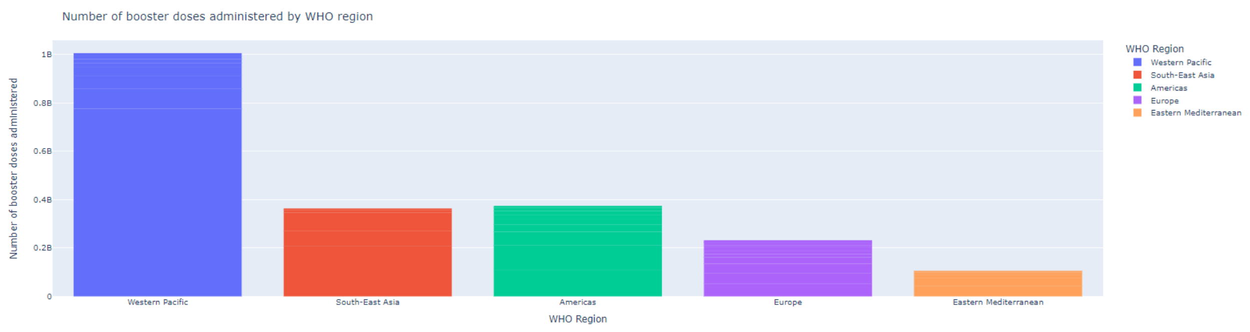
This code is a Python program that uses the *plotnine* library to create a bar plot with trend lines of vaccination rates over time by region. The program takes a dataset of vaccination data as input and groups the data by WHO regions. It then calculates the monthly total vaccinations for each region and creates a new dataset with this information. The *plotnine* library is then used to create a bar plot where the x-axis represents the month, the y-axis represents the total vaccinations, and the fill represents the region. The bars are positioned side-by-side, and trendlines are added to each bar to show the overall trend for each region. The resulting plot is displayed in the output.

The chosen visual representation, which is a bar plot with trendlines, is effective in showing the overall trend of vaccination rates over time for each region, as well as the comparison between regions. The bars represent the total number of vaccinations per month, while the trendlines show the general trend or pattern of increase or decrease over time. Using different colors to fill the bars and a legend to identify the regions, it's easy to compare the vaccination rates between different regions. Additionally, the *ggplot* library used in this code allows for customization of the plot aesthetics and layout, resulting in a clear and visually appealing representation of the data.

Question: How does the number of booster doses administered vary by region?

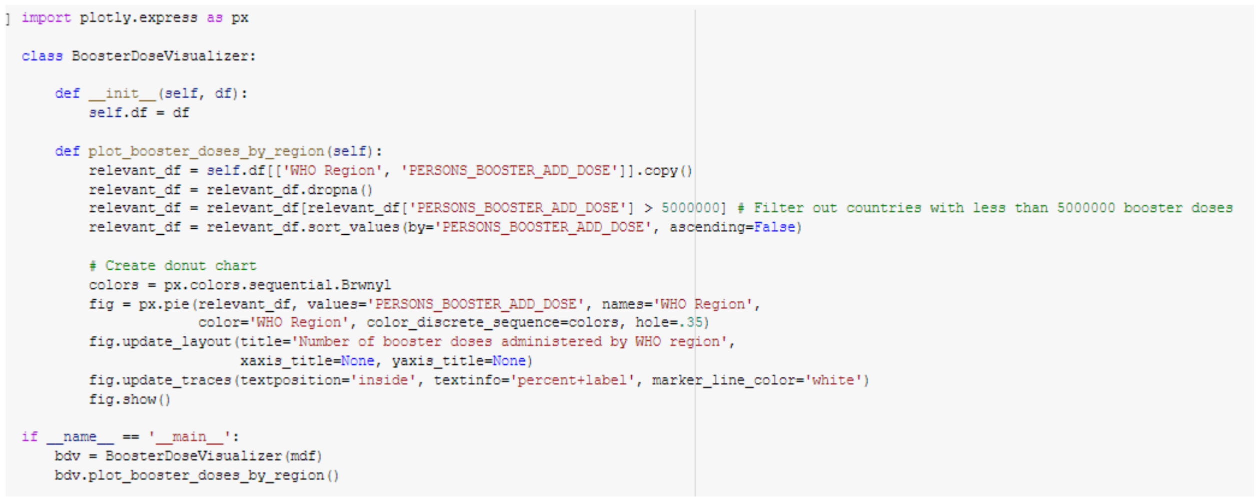


```python
class BoosterDoseVisualizer:

    def __init__(self, df):
        self.df = df

    def plot_booster_doses_by_region(self):
        relevant_df = self.df[['WHO Region', 'PERSONS_BOOSTER_ADD_DOSE']].copy()
        relevant_df = relevant_df.dropna()
        relevant_df = relevant_df[relevant_df['PERSONS_BOOSTER_ADD_DOSE'] > 5000000] # Filter out countries with less than 5000000 booster doses
        relevant_df = relevant_df.sort_values(by='PERSONS_BOOSTER_ADD_DOSE', ascending=False)

        # Create bar plot
        colors = px.colors.qualitative.Plotly
        fig = px.bar(relevant_df, x='WHO Region', y='PERSONS_BOOSTER_ADD_DOSE',
                     color='WHO Region', color_discrete_sequence=colors)
        fig.update_layout(title='Number of booster doses administered by WHO region',
                          xaxis_title='WHO Region', yaxis_title='Number of booster doses administered')
        fig.update_traces(marker_line_width=0)
        fig.show()

if __name__ == '__main__':
    bdv = BoosterDoseVisualizer(mdf)
    bdv.plot_booster_doses_by_region()
```

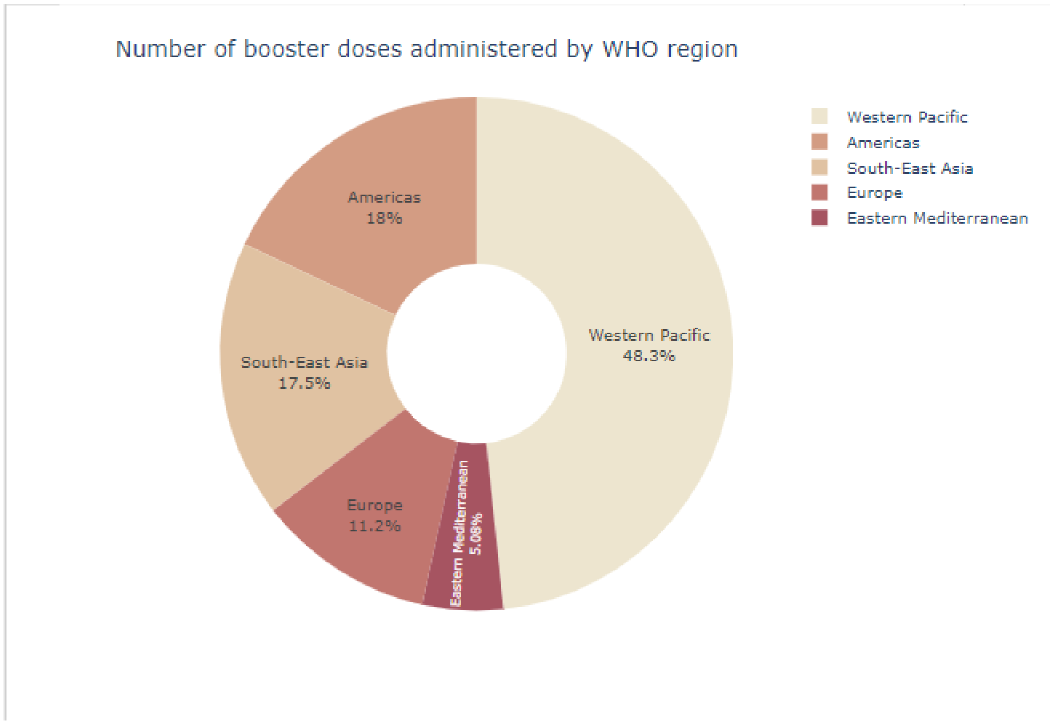


This code defines a class called "***BoosterDoseVisualizer***" that helps visualize the number of booster doses administered in different regions of the world. The class takes a dataframe as input, which contains information about the number of booster doses administered by each country in the world. It then processes the data and filters out countries that have administered less than 5 million booster doses. Finally, the class creates a bar plot that shows the number of booster doses administered in each WHO region of the world. Each bar in the plot is colored according to the region it represents. The plot is displayed using the Plotly library.

Now, let's see another form of visual representation that I tried:



```
] import plotly.express as px

class BoosterDoseVisualizer:

    def __init__(self, df):
        self.df = df

    def plot_booster_doses_by_region(self):
        relevant_df = self.df[['WHO Region', 'PERSONS_BOOSTER_ADD_DOSE']].copy()
        relevant_df = relevant_df.dropna()
        relevant_df = relevant_df[relevant_df['PERSONS_BOOSTER_ADD_DOSE'] > 5000000] # Filter out countries with less than 5000000 booster doses
        relevant_df = relevant_df.sort_values(by='PERSONS_BOOSTER_ADD_DOSE', ascending=False)

        # Create donut chart
        colors = px.colors.sequential.Brwnyl
        fig = px.pie(relevant_df, values='PERSONS_BOOSTER_ADD_DOSE', names='WHO Region',
                     color='WHO Region', color_discrete_sequence=colors, hole=.35)
        fig.update_layout(title='Number of booster doses administered by WHO region',
                          xaxis_title=None, yaxis_title=None)
        fig.update_traces(textposition='inside', textinfo='percent+label', marker_line_color='white')
        fig.show()

if __name__ == '__main__':
    bdv = BoosterDoseVisualizer(mdf)
    bdv.plot_booster_doses_by_region()
```



This program creates a donut chart using Plotly to display the number of booster doses administered in each WHO region. The data is first filtered to include only countries with more than 5 million booster doses administered. The donut chart shows each region as a slice of the donut with the size of the slice representing the proportion of booster doses administered in that region. The chart also includes labels showing the percentage and number of booster doses for each region.

A donut chart was chosen for this data visualization because it effectively displays the proportion of booster doses administered by each WHO region in a visually appealing way. The central hole in the chart can help draw the viewer's attention to the data in the center and provide a clear distinction between the regions. Additionally, the color-coded regions and the labels inside the chart make it easier for the viewer to quickly identify which region each section corresponds to, and the proportion of booster doses administered in that region.

Task 2 – Tableau Visualization

In this part of the coursework, several datasets from various reliable sources were put together and analyzed using Tableau. The objectives are discussed below with its rationale as well as their visual encoding.

Objective: Daily cases across the world

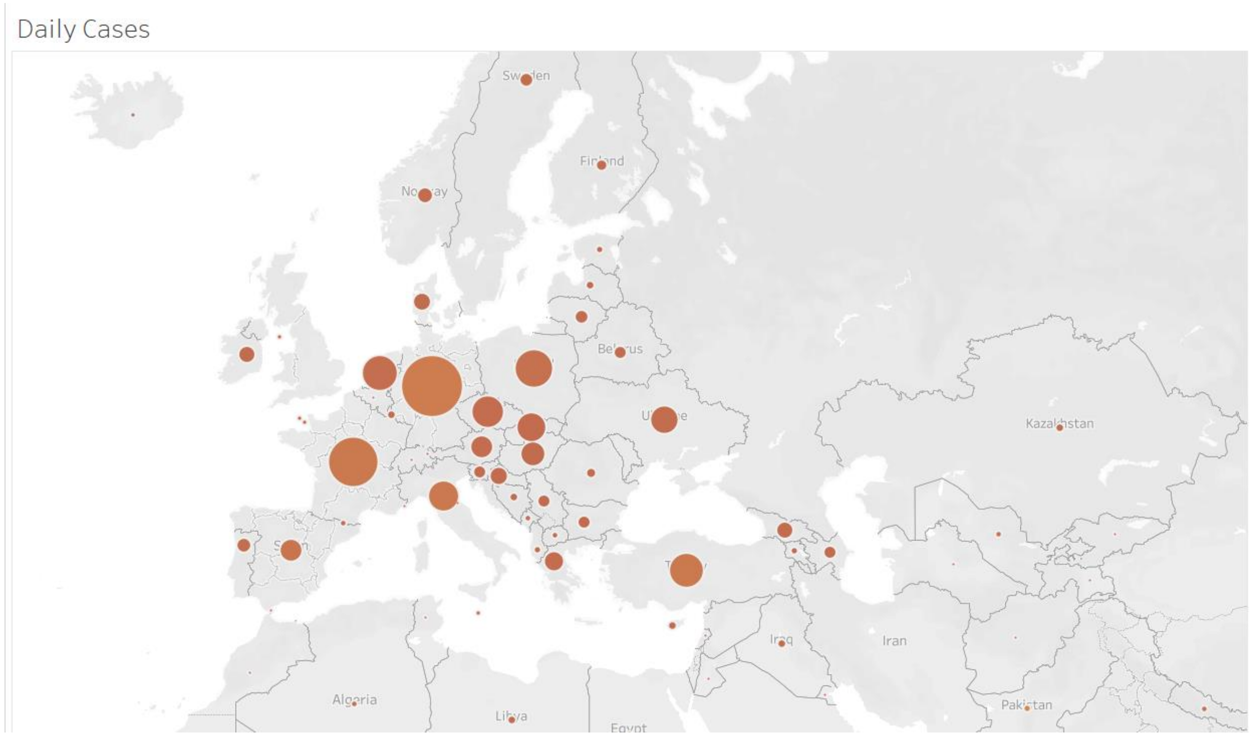


Figure 1: Daily cases across the world

In this visualization, I chose the COVID-19 dataset from [1] that was uploaded on Moodle as a reference. I decided to visualize this in a world choropleth map as this may provide a better visualization on the daily cases across the countries all over the world. To perform this action, I chose the SUM measure on Tableau and sorted the countries according to the cases newly recorded in the last 24 hours. In addition, I color-coded this visualization with respect to the total vaccinations of each country in 'red-green divergence' color scale, which gives us the advantage of understanding the map within seconds as the red areas are supposed to have lower vaccination rates whereas the green areas have higher vaccination rates.

Objective: Daily deaths across the continents

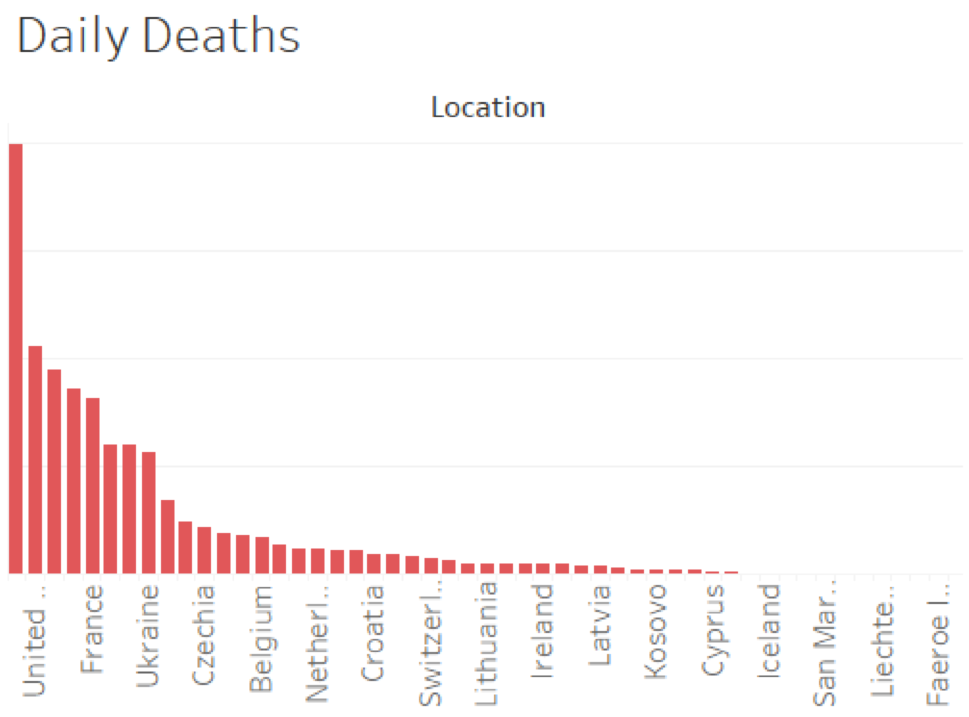


Figure 2: Daily deaths across the continents

In this visualization, a bar chart approach was chosen to visualize the daily deaths across the countries in different continents which was extracted from the COVID-19 dataset in [2]. This visualization has been color-coded with respect to the continents, which has also been implemented as a filter that has been included in the final dashboard to navigate through the continents and visualize the data with ease. This one also leveraged the SUM function on the "Daily deaths" variable to implement the visualization.

Objective: Total cases versus deaths ratio across the countries

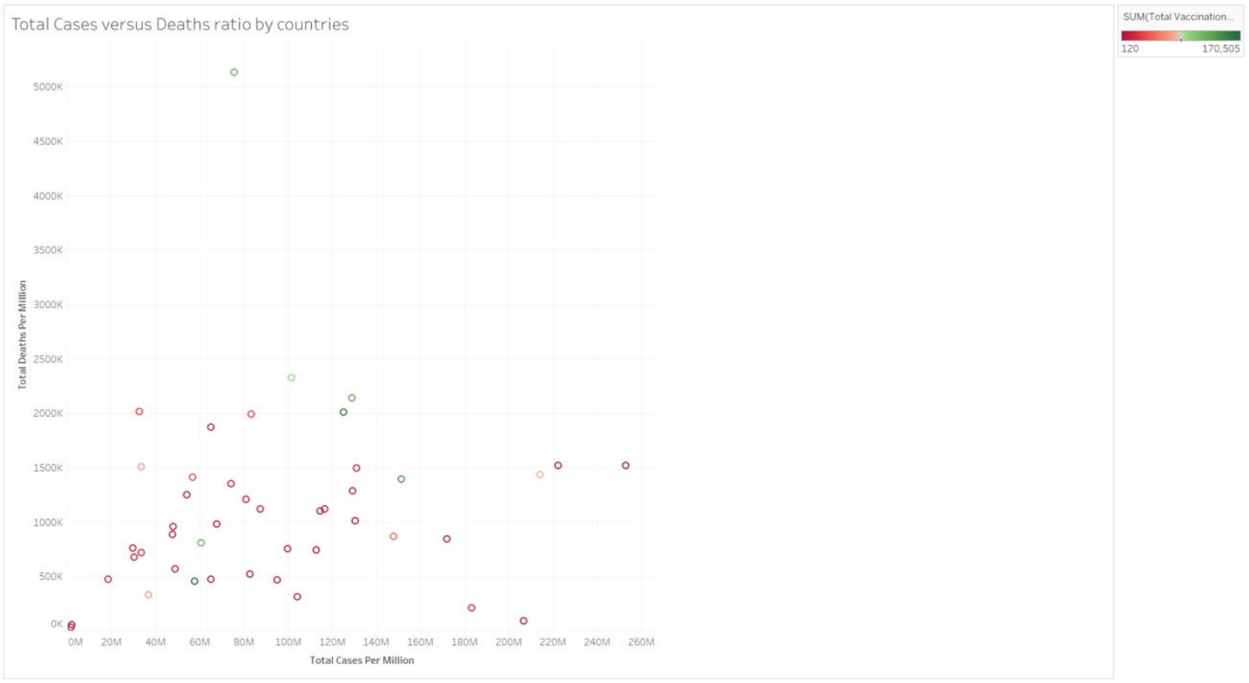


Figure 3: Total cases versus deaths ratio across the countries

In this visualization, a scattered chart was prepared to visualize the total cases versus deaths by countries where the data was derived from [2] and color-coded based on the total vaccinations across the countries in "red-green divergence" color scale to clearly specify the distinction between the danger zones where the vaccination rates are low and the safer areas where the vaccination rates are comparatively higher.

Objective: Visualizing different measures by a dropdown filter

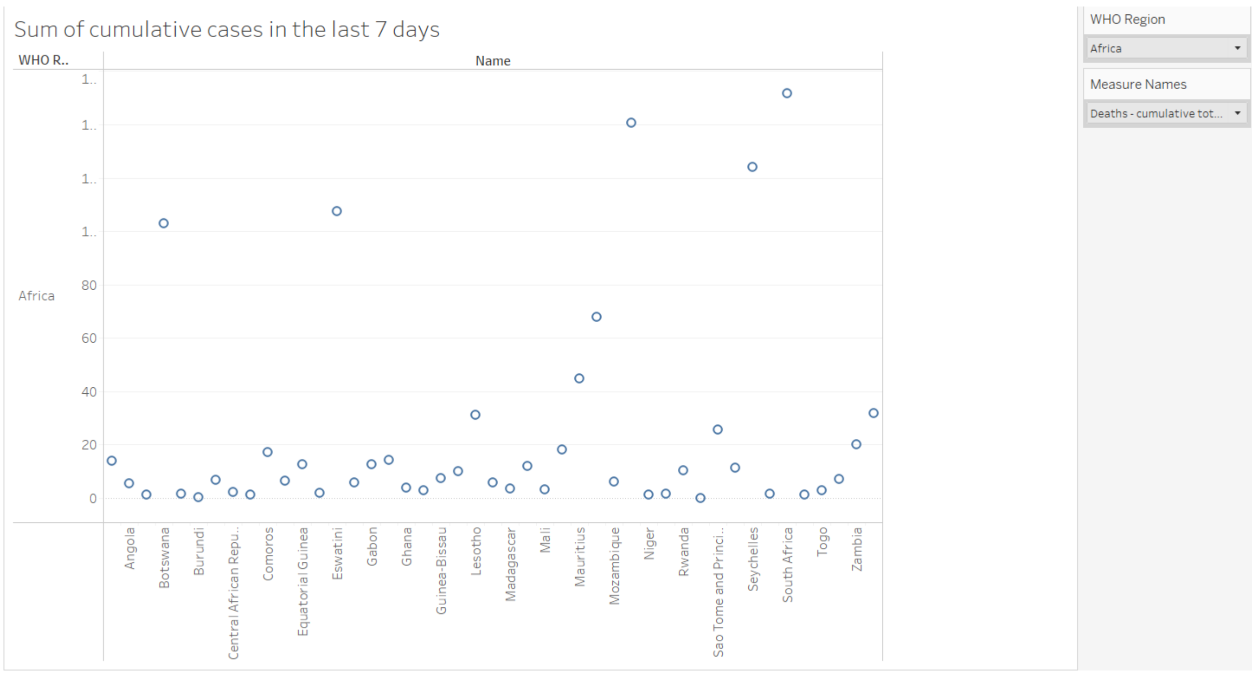
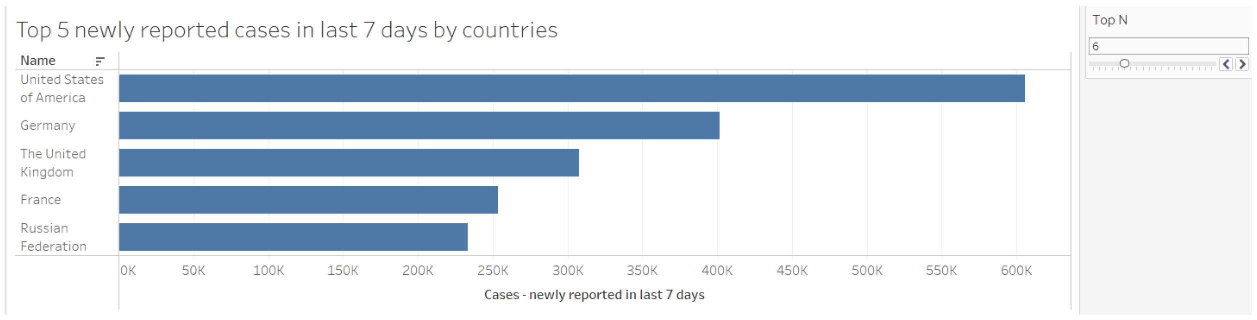


Figure 4: Total deaths in the past 7 days by countries

For this part of the visualization, another scattered plot has been implemented on the cumulative cases in the past 7 days based on the data found in [1] and [2]. In this case, two filters were set to navigate through the data that allows a user to choose a particular continent as well as a particular measure at a time, and the data has been color-coded according to the measure names.

Objective: Visualizing top 5 countries with highest newly reported cases



This visualization showcased an instance where a custom measure was generated by using "calculated field" on Tableau to identify the top countries that have the highest numbers of newly reported cases in the last 7 days. The calculated field allows a user to choose a user-defined number of countries in the plot by using a simple slider. To create this visualization, an additional parameter was also created.

Objective: Average Cumulative cases by countries and WHO regions

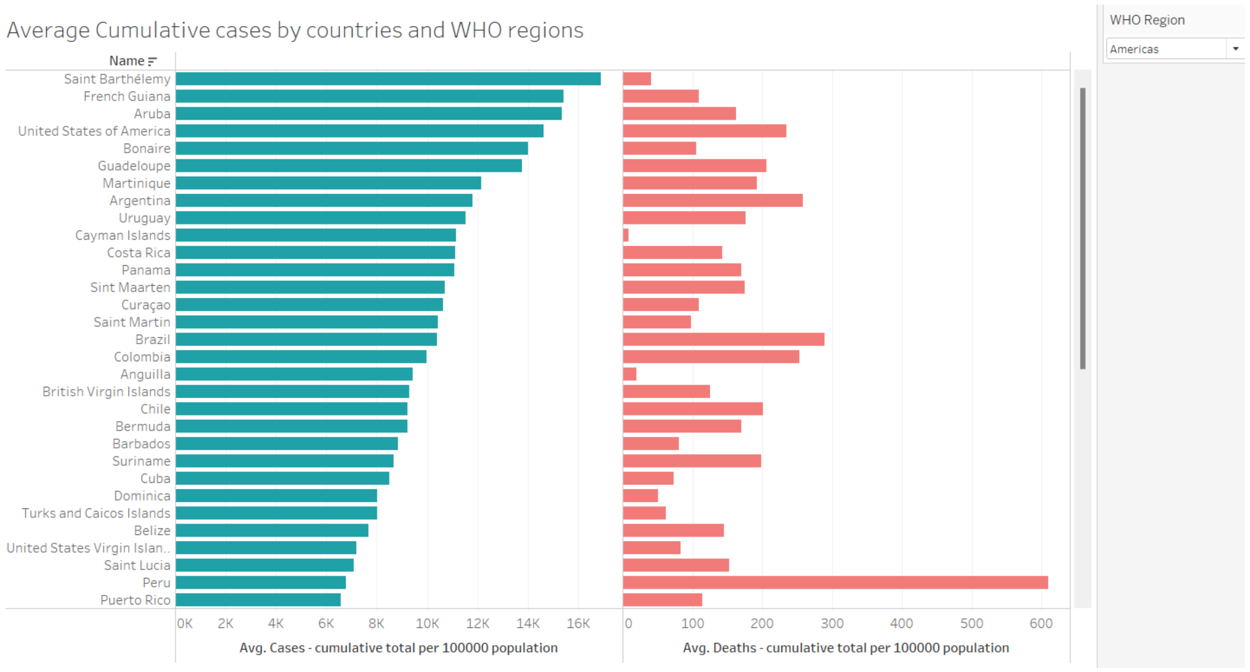
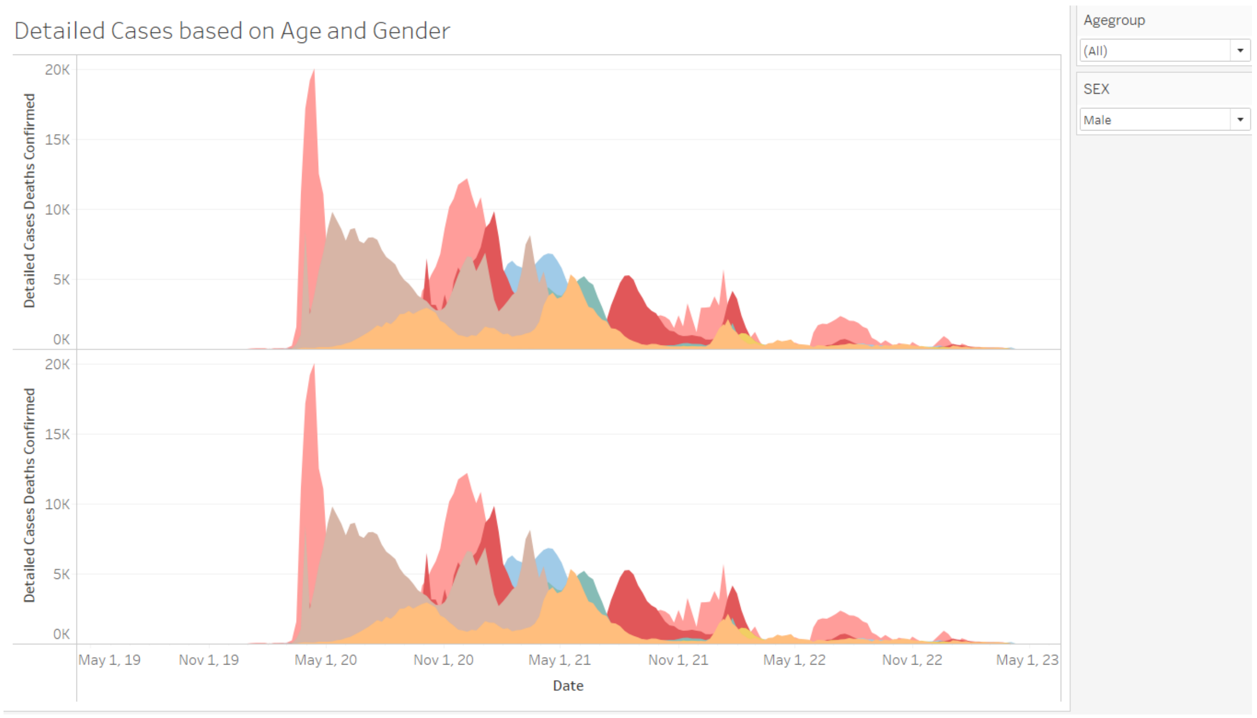


Figure 5: Average cumulative cases by countries and WHO regions

This visualization was intended to present with a side-by-side bar chart to visualize the average number of cumulative cases versus the average number of cumulative deaths across the countries. This visualization contains another filter “WHO region” based on which the numbers will vary for each of the countries per 100000 population. This visualization provides a good presentation on the case-versus-death contrast ratio in each country.

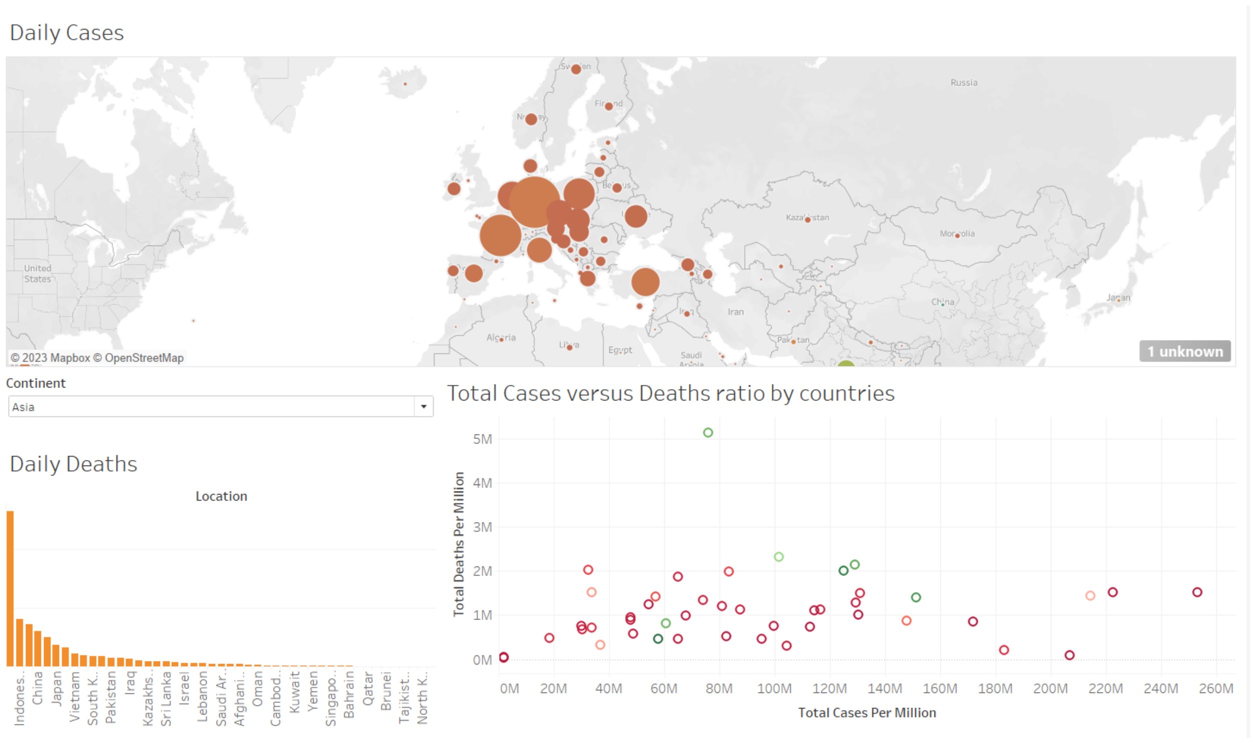
Objective: Detailed Cases based on Age and Gender



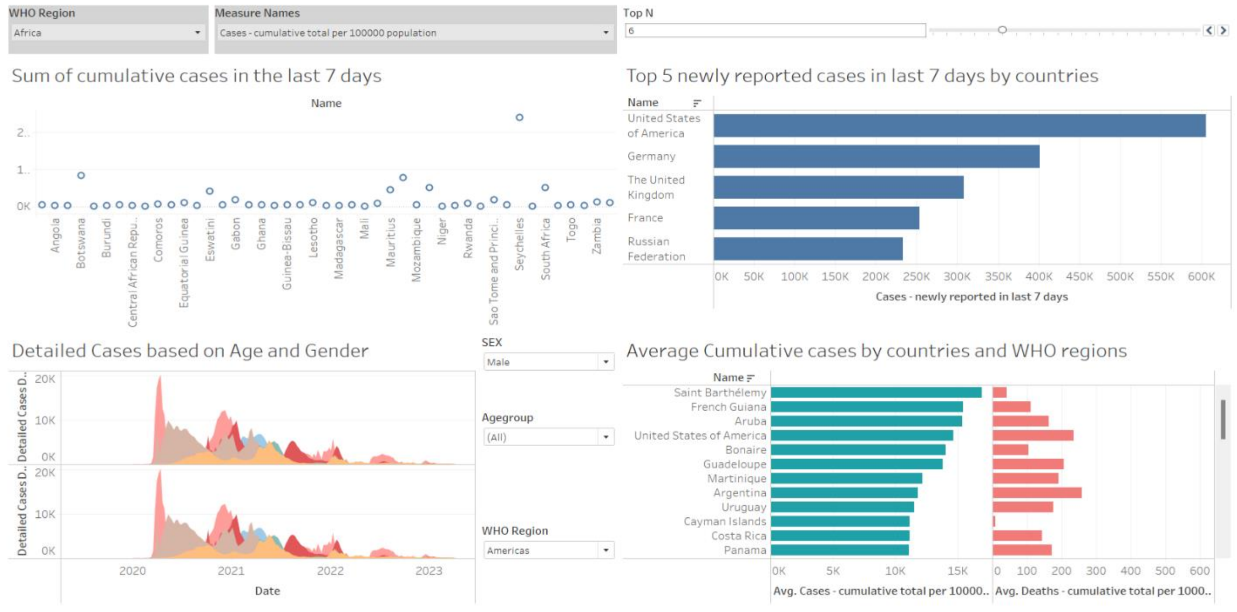
In this coursework, detailed confirmed cases across the countries all over the world have been visualized based on two preset filters 'Age group' and 'Sex', which were derived from [3] that we collected from an online work. In this visual representation, the SUM of the detailed confirmed cases as well as detailed confirmed deaths were considered and plotted along the X-axis with color-coding by the country names. This kind of visualization imparts a beautiful yet meaningful knowledge over the existing data across the countries, gender, and age group.

DASHBOARD

The complete interactive Tableau dashboard can be visualized and used from Tableau Public. Clicking the images below should lead to the project on the website.



Continent: Asia


Sum of cumulative cases in the last 7 days
Name
Top 5 newly reported cases in last 7 days by countries
Cases - newly reported in last 7 days
Detailed Cases based on Age and Gender
Date
Average Cumulative cases by countries and WHO regions
Avg. Cases - cumulative total per 10000..
Avg. Deaths - cumulative total per 1000..

WHO Region: Africa

Measure Names: Cases - cumulative total per 100000 population

Top N: 6

SEX: Male

Agegroup: (All)

WHO Region: Americas

University of Roehampton London

References

[1] WHO COVID-19 Dashboard. Geneva: World Health Organization, 2020. Available online: https://covid19.who.int/

[2] Edouard Mathieu, Hannah Ritchie, Lucas Rodés-Guirao, Cameron Appel, Charlie Giattino, Joe Hasell, Bobbie Macdonald, Saloni Dattani, Diana Beltekian, Esteban Ortiz-Ospina and Max Roser (2020) - "Coronavirus Pandemic (COVID-19)". Published online at OurWorldInData.org. Retrieved from: 'https://ourworldindata.org/coronavirus' [Online Resource]

[3] World Health Organization – "WHO COVID-19 Detailed Surveillance Data". Published online at app.powerbi.com. Can be retrieved from: "https://app.powerbi.com/view?r=eyJrIjoiYWRiZWVkNWUtNmM0Ni00MDAwLTljYWMtN2EwNTM3YjQzYmRmIiwidCI6ImY2MTBjMGI3LWJkMjQtNGIzOS04MTBiLTNkYzI4MGFmYjU5MCIsImMiOjh9" [Online Resource]