assig

May 13, 2025

[1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
from datetime import datetime
import warnings
warnings.filterwarnings('ignore')

# Set plotting style
# Using a valid style name from matplotlib's available styles
plt.style.use('seaborn-v0_8')  # Updated to a valid style name
# Alternative: you could use plt.style.available to see all available styles
sns.set_style("whitegrid")
plt.rcParams['figure.figsize'] = (12, 8)
plt.rcParams['font.size'] = 12

# Download the dataset directly
url = "https://covid.ourworldindata.org/data/owid-covid-data.csv"
print("Downloading COVID-19 data from Our World in Data...")
df = pd.read_csv(url)
print("Download complete!")
```

```
Downloading COVID-19 data from Our World in Data…
Download complete!
```

[2]:

```
# Display basic information about the dataset
print(f"Dataset shape: {df.shape}")
print("\nFirst few rows:")
display(df.head())

print("\nColumn names:")
print(df.columns.tolist())

print("\nData types:")
print(df.dtypes)

print("\nMissing values per column:")
```

```python
print(df.isnull().sum())

# Check the date range
print(f"\nDate range: {df['date'].min()} to {df['date'].max()}")

# Check unique countries/locations
print(f"\nNumber of unique locations: {df['location'].nunique()}")
print("\nSample locations:")
print(df['location'].unique()[:10])
```

```
Dataset shape: (429435, 67)

First few rows:

  iso_code continent     location        date  total_cases  new_cases  \
0      AFG      Asia  Afghanistan  2020-01-05          0.0        0.0
1      AFG      Asia  Afghanistan  2020-01-06          0.0        0.0
2      AFG      Asia  Afghanistan  2020-01-07          0.0        0.0
3      AFG      Asia  Afghanistan  2020-01-08          0.0        0.0
4      AFG      Asia  Afghanistan  2020-01-09          0.0        0.0

   new_cases_smoothed  total_deaths  new_deaths  new_deaths_smoothed  ...  \
0                 NaN           0.0         0.0                  NaN  ...
1                 NaN           0.0         0.0                  NaN  ...
2                 NaN           0.0         0.0                  NaN  ...
3                 NaN           0.0         0.0                  NaN  ...
4                 NaN           0.0         0.0                  NaN  ...

   male_smokers  handwashing_facilities  hospital_beds_per_thousand  \
0           NaN                  37.746                         0.5
1           NaN                  37.746                         0.5
2           NaN                  37.746                         0.5
3           NaN                  37.746                         0.5
4           NaN                  37.746                         0.5

   life_expectancy  human_development_index  population  \
0            64.83                    0.511    41128772
1            64.83                    0.511    41128772
2            64.83                    0.511    41128772
3            64.83                    0.511    41128772
4            64.83                    0.511    41128772

   excess_mortality_cumulative_absolute  excess_mortality_cumulative  \
0                                   NaN                          NaN
1                                   NaN                          NaN
2                                   NaN                          NaN
3                                   NaN                          NaN
4                                   NaN                          NaN
```

```
   excess_mortality  excess_mortality_cumulative_per_million
0               NaN                                      NaN
1               NaN                                      NaN
2               NaN                                      NaN
3               NaN                                      NaN
4               NaN                                      NaN

[5 rows x 67 columns]

Column names:
['iso_code', 'continent', 'location', 'date', 'total_cases', 'new_cases',
'new_cases_smoothed', 'total_deaths', 'new_deaths', 'new_deaths_smoothed',
'total_cases_per_million', 'new_cases_per_million',
'new_cases_smoothed_per_million', 'total_deaths_per_million',
'new_deaths_per_million', 'new_deaths_smoothed_per_million',
'reproduction_rate', 'icu_patients', 'icu_patients_per_million',
'hosp_patients', 'hosp_patients_per_million', 'weekly_icu_admissions',
'weekly_icu_admissions_per_million', 'weekly_hosp_admissions',
'weekly_hosp_admissions_per_million', 'total_tests', 'new_tests',
'total_tests_per_thousand', 'new_tests_per_thousand', 'new_tests_smoothed',
'new_tests_smoothed_per_thousand', 'positive_rate', 'tests_per_case',
'tests_units', 'total_vaccinations', 'people_vaccinated',
'people_fully_vaccinated', 'total_boosters', 'new_vaccinations',
'new_vaccinations_smoothed', 'total_vaccinations_per_hundred',
'people_vaccinated_per_hundred', 'people_fully_vaccinated_per_hundred',
'total_boosters_per_hundred', 'new_vaccinations_smoothed_per_million',
'new_people_vaccinated_smoothed', 'new_people_vaccinated_smoothed_per_hundred',
'stringency_index', 'population_density', 'median_age', 'aged_65_older',
'aged_70_older', 'gdp_per_capita', 'extreme_poverty', 'cardiovasc_death_rate',
'diabetes_prevalence', 'female_smokers', 'male_smokers',
'handwashing_facilities', 'hospital_beds_per_thousand', 'life_expectancy',
'human_development_index', 'population', 'excess_mortality_cumulative_absolute',
'excess_mortality_cumulative', 'excess_mortality',
'excess_mortality_cumulative_per_million']

Data types:
iso_code                                    object
continent                                   object
location                                    object
date                                        object
total_cases                                float64
                                            …
population                                   int64
excess_mortality_cumulative_absolute       float64
excess_mortality_cumulative                float64
excess_mortality                           float64
```

```
excess_mortality_cumulative_per_million    float64
Length: 67, dtype: object

Missing values per column:
iso_code                                        0
continent                                   26525
location                                        0
date                                            0
total_cases                                 17631
                                            …
population                                      0
excess_mortality_cumulative_absolute       416024
excess_mortality_cumulative                416024
excess_mortality                           416024
excess_mortality_cumulative_per_million    416024
Length: 67, dtype: int64

Date range: 2020-01-01 to 2024-08-14

Number of unique locations: 255

Sample locations:
['Afghanistan' 'Africa' 'Albania' 'Algeria' 'American Samoa' 'Andorra'
 'Angola' 'Anguilla' 'Antigua and Barbuda' 'Argentina']
```

[3]:

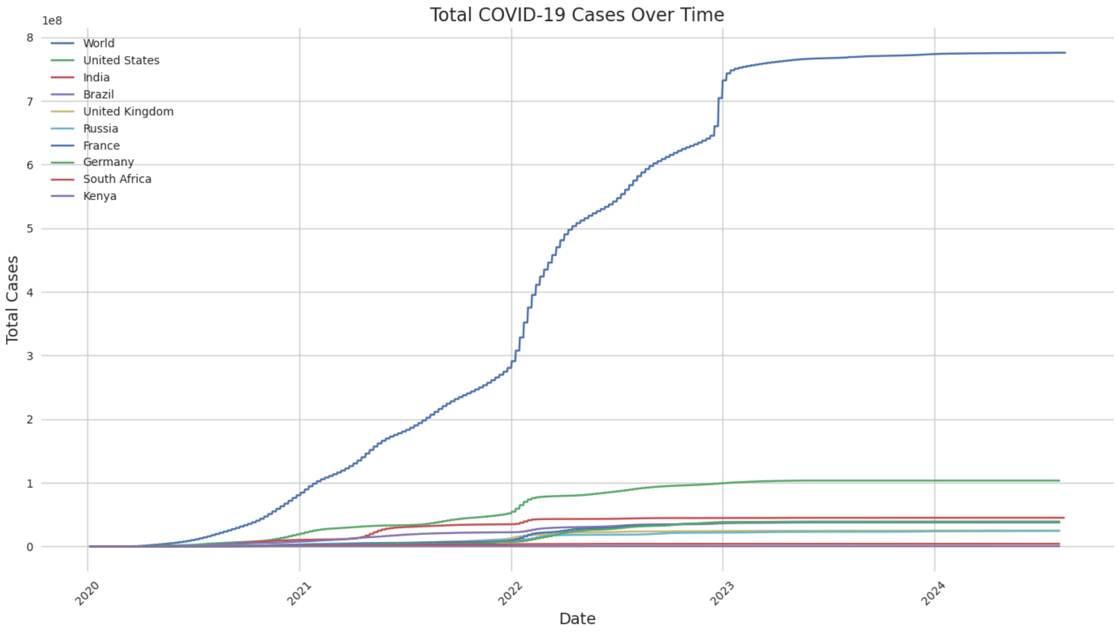
```
# Convert date to datetime
df['date'] = pd.to_datetime(df['date'])

# Select countries of interest for detailed analysis
countries_of_interest = ['World', 'United States', 'India', 'Brazil', 'United␣
  ↪Kingdom',
                         'Russia', 'France', 'Germany', 'South Africa', 'Kenya']

# Create a filtered dataframe for these countries
filtered_df = df[df['location'].isin(countries_of_interest)]

# Check for missing values in key columns
key_columns = ['total_cases', 'new_cases', 'total_deaths', 'new_deaths',
               'total_vaccinations', 'people_vaccinated',␣
  ↪'people_fully_vaccinated']

print("Missing values in key columns for selected countries:")
print(filtered_df[key_columns].isnull().sum())

# Handle missing values for numeric columns
# For cumulative columns, we can forward fill (use the last known value)
```

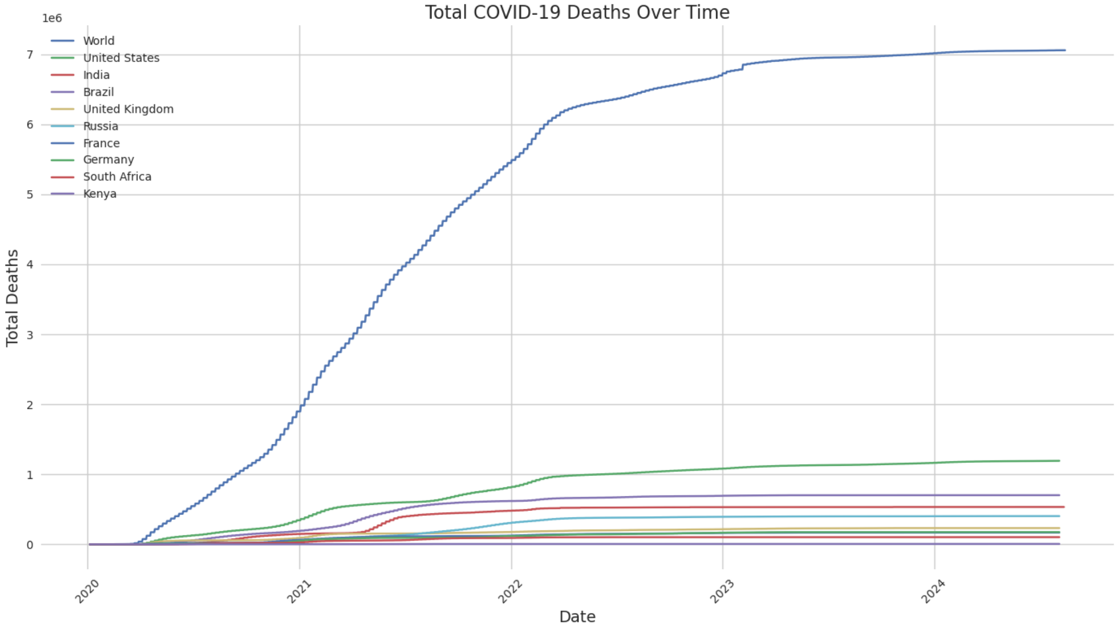
```python
# For new daily values, we can replace NaN with 0 (assuming missing means no␣
 ↪new cases/deaths)

# First, let's create a copy to avoid warnings
cleaned_df = filtered_df.copy()

# Forward fill for cumulative columns
cumulative_cols = ['total_cases', 'total_deaths', 'total_vaccinations',
                   'people_vaccinated', 'people_fully_vaccinated']
for col in cumulative_cols:
    if col in cleaned_df.columns:
        cleaned_df[col] = cleaned_df.groupby('location')[col].
 ↪fillna(method='ffill')

# Replace NaN with 0 for daily new values
daily_cols = ['new_cases', 'new_deaths', 'new_vaccinations']
for col in daily_cols:
    if col in cleaned_df.columns:
        cleaned_df[col] = cleaned_df[col].fillna(0)

# Calculate additional metrics
# Death rate (case fatality rate)
cleaned_df['death_rate'] = (cleaned_df['total_deaths'] /␣
 ↪cleaned_df['total_cases'] * 100).round(2)

# Vaccination rate (% of population with at least one dose)
cleaned_df['vaccination_rate'] = (cleaned_df['people_vaccinated'] /␣
 ↪cleaned_df['population'] * 100).round(2)

# Full vaccination rate
cleaned_df['full_vaccination_rate'] = (cleaned_df['people_fully_vaccinated'] /␣
 ↪cleaned_df['population'] * 100).round(2)

print("\nData cleaning complete!")
```

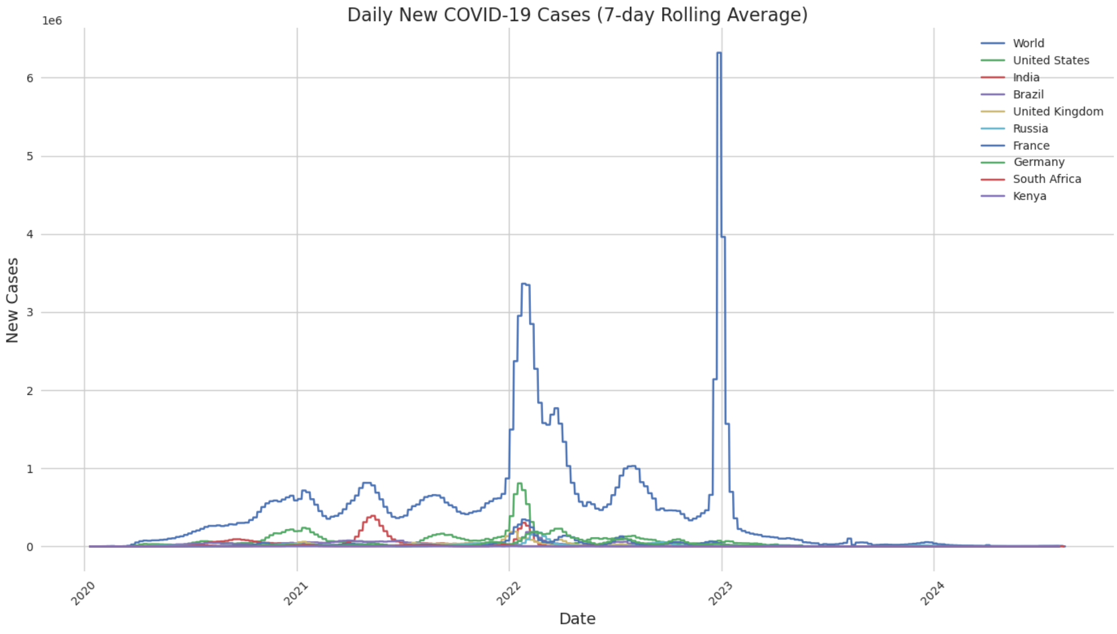
```
Missing values in key columns for selected countries:
total_cases                  18
new_cases                  1252
total_deaths                 18
new_deaths                  810
total_vaccinations         8913
people_vaccinated          8989
people_fully_vaccinated    9113
dtype: int64

Data cleaning complete!
```

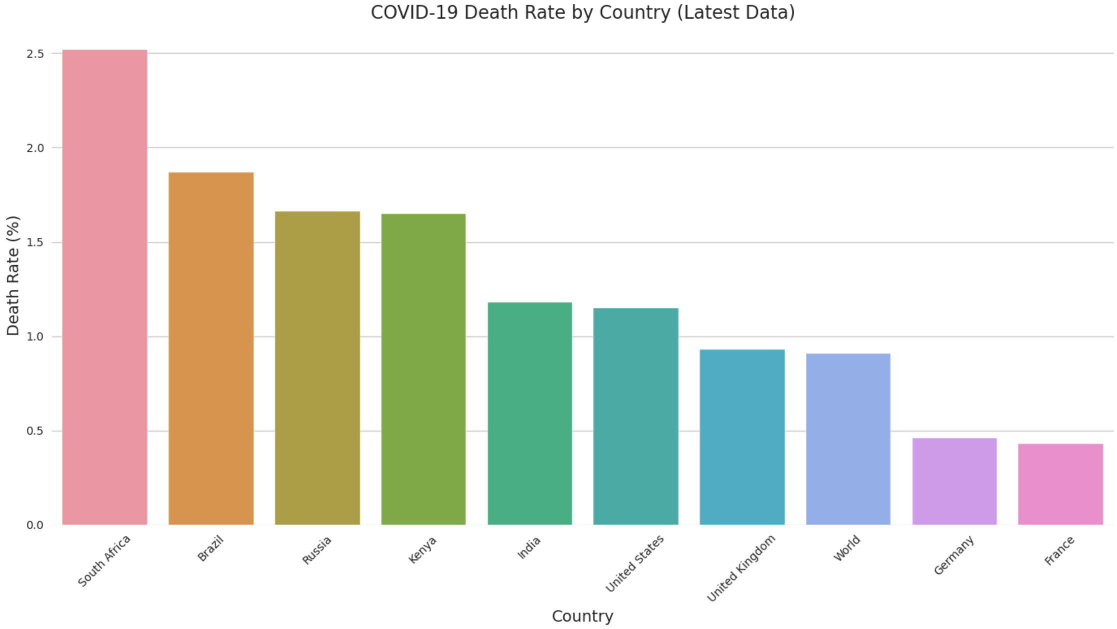
[4]:

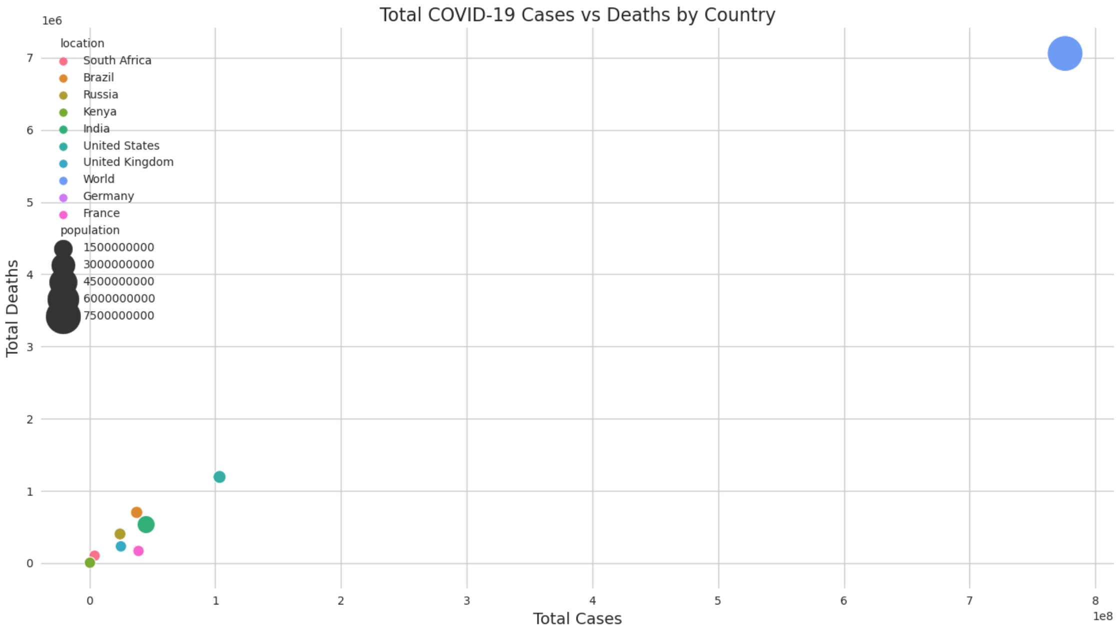
```
# 1. Total Cases Over Time by Country
plt.figure(figsize=(14, 8))
for country in countries_of_interest:
    country_data = cleaned_df[cleaned_df['location'] == country]
    plt.plot(country_data['date'], country_data['total_cases'], label=country)

plt.title('Total COVID-19 Cases Over Time', fontsize=16)
plt.xlabel('Date', fontsize=14)
plt.ylabel('Total Cases', fontsize=14)
plt.legend()
plt.grid(True)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

# 2. Total Deaths Over Time by Country
plt.figure(figsize=(14, 8))
for country in countries_of_interest:
    country_data = cleaned_df[cleaned_df['location'] == country]
    plt.plot(country_data['date'], country_data['total_deaths'], label=country)

plt.title('Total COVID-19 Deaths Over Time', fontsize=16)
plt.xlabel('Date', fontsize=14)
plt.ylabel('Total Deaths', fontsize=14)
plt.legend()
plt.grid(True)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

# 3. Daily New Cases (7-day rolling average for smoother visualization)
plt.figure(figsize=(14, 8))
for country in countries_of_interest:
    country_data = cleaned_df[cleaned_df['location'] == country]
    # Calculate 7-day rolling average
    country_data['new_cases_smoothed'] = country_data['new_cases'].rolling(7).
  ↪mean()
    plt.plot(country_data['date'], country_data['new_cases_smoothed'],␣
  ↪label=country)

plt.title('Daily New COVID-19 Cases (7-day Rolling Average)', fontsize=16)
plt.xlabel('Date', fontsize=14)
plt.ylabel('New Cases', fontsize=14)
plt.legend()
plt.grid(True)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

# 4. Death Rate Comparison (latest data)
latest_data = cleaned_df.groupby('location').last().reset_index()
latest_data = latest_data.sort_values('death_rate', ascending=False)

plt.figure(figsize=(14, 8))
sns.barplot(x='location', y='death_rate', data=latest_data)
plt.title('COVID-19 Death Rate by Country (Latest Data)', fontsize=16)
plt.xlabel('Country', fontsize=14)
plt.ylabel('Death Rate (%)', fontsize=14)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

# 5. Total Cases vs Total Deaths Scatter Plot
plt.figure(figsize=(14, 8))
sns.scatterplot(data=latest_data, x='total_cases', y='total_deaths',
                size='population', sizes=(100, 1000), hue='location',␣
 ↪legend='brief')

plt.title('Total COVID-19 Cases vs Deaths by Country', fontsize=16)
plt.xlabel('Total Cases', fontsize=14)
plt.ylabel('Total Deaths', fontsize=14)
plt.grid(True)
plt.tight_layout()
plt.show()

# 6. Top 10 Countries by Total Cases (bar chart)
top_cases = latest_data.sort_values('total_cases', ascending=False).head(10)
plt.figure(figsize=(14, 8))
sns.barplot(x='total_cases', y='location', data=top_cases)
plt.title('Top 10 Countries by Total COVID-19 Cases', fontsize=16)
plt.xlabel('Total Cases', fontsize=14)
plt.ylabel('Country', fontsize=14)
plt.tight_layout()
plt.show()
```

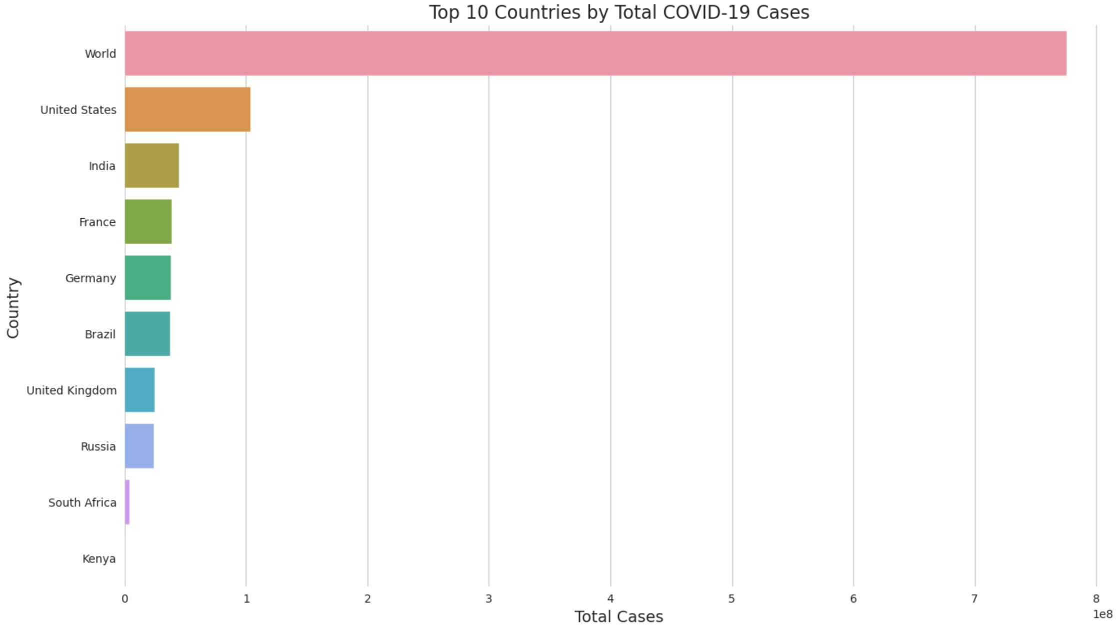

Total COVID-19 Cases Over Time
1e8
8
7
6
5
4
3
2
1
0
Total Cases
World
United States
India
Brazil
United Kingdom
Russia
France
Germany
South Africa
Kenya
2020
2021
2022
2023
2024
Date


Total COVID-19 Deaths Over Time
1e6
7
6
5
4
3
2
1
0
Total Deaths
World
United States
India
Brazil
United Kingdom
Russia
France
Germany
South Africa
Kenya
2020
2021
2022
2023
2024
Date


Daily New COVID-19 Cases (7-day Rolling Average)
1e6
6
5
4
3
2
1
0
New Cases
2020
2021
2022
2023
2024
Date
World
United States
India
Brazil
United Kingdom
Russia
France
Germany
South Africa
Kenya


COVID-19 Death Rate by Country (Latest Data)
2.5
2.0
1.5
1.0
0.5
0.0
Death Rate (%)
South Africa
Brazil
Russia
Kenya
India
United States
United Kingdom
World
Germany
France
Country

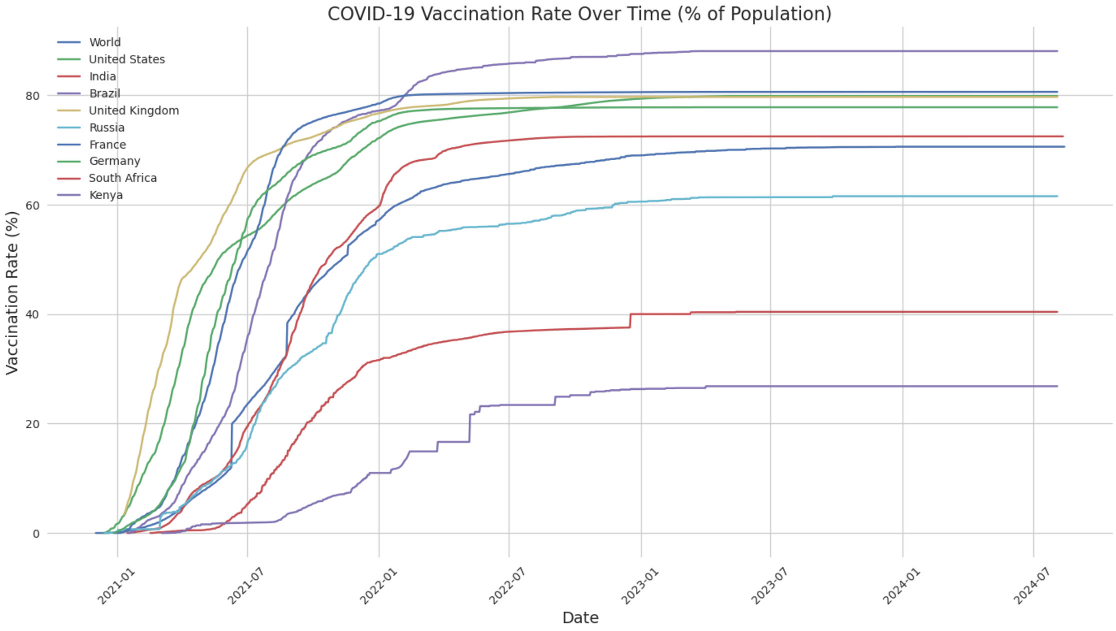


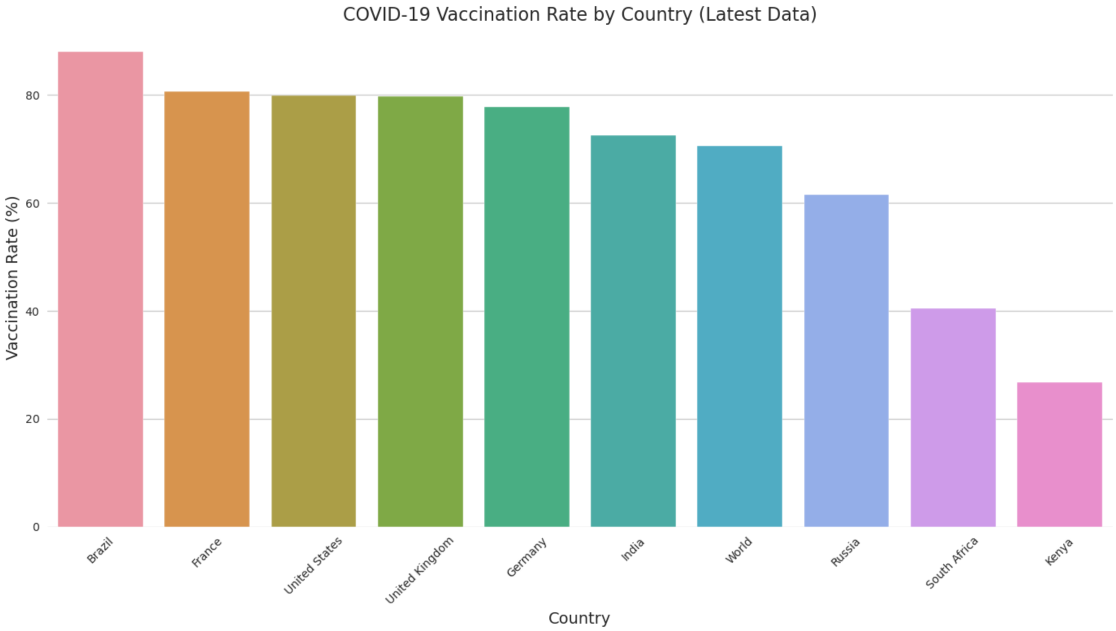


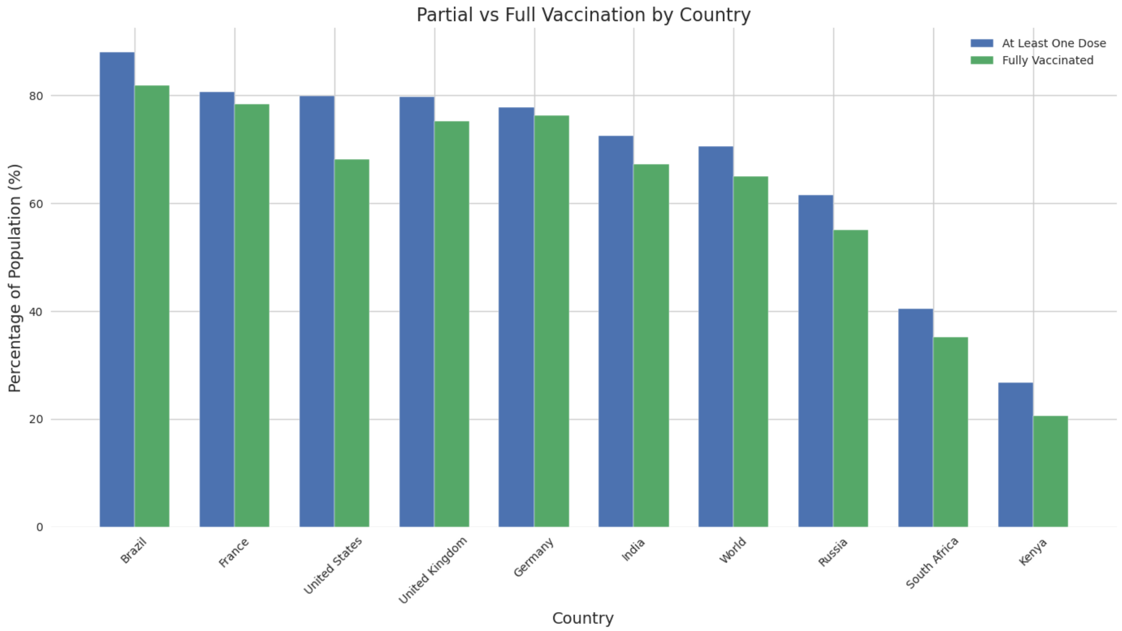
[5]:
```
# 1. Vaccination Progress Over Time
plt.figure(figsize=(14, 8))
for country in countries_of_interest:
    country_data = cleaned_df[cleaned_df['location'] == country]
    plt.plot(country_data['date'], country_data['vaccination_rate'],␣
  ↪label=country)
```

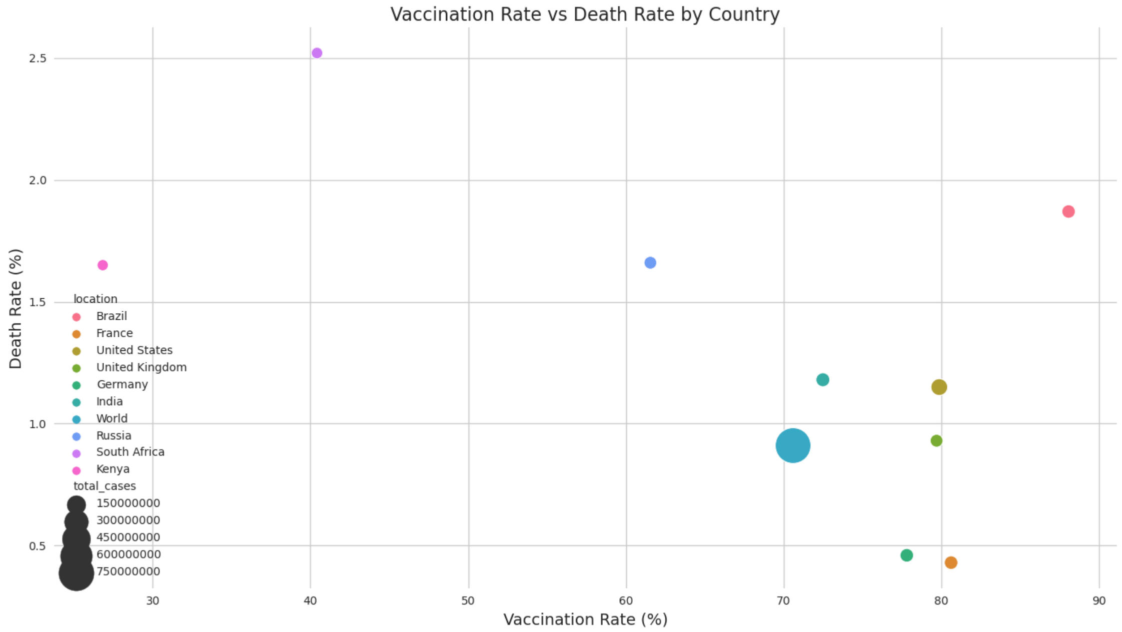
```
plt.title('COVID-19 Vaccination Rate Over Time (% of Population)', fontsize=16)
plt.xlabel('Date', fontsize=14)
plt.ylabel('Vaccination Rate (%)', fontsize=14)
plt.legend()
plt.grid(True)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

# 2. Comparison of Vaccination Rates (latest data)
latest_data = latest_data.sort_values('vaccination_rate', ascending=False)

plt.figure(figsize=(14, 8))
sns.barplot(x='location', y='vaccination_rate', data=latest_data)
plt.title('COVID-19 Vaccination Rate by Country (Latest Data)', fontsize=16)
plt.xlabel('Country', fontsize=14)
plt.ylabel('Vaccination Rate (%)', fontsize=14)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

# 3. Full vs Partial Vaccination Comparison
plt.figure(figsize=(14, 8))
x = np.arange(len(latest_data))
width = 0.35

plt.bar(x - width/2, latest_data['vaccination_rate'], width, label='At Least One Dose')
plt.bar(x + width/2, latest_data['full_vaccination_rate'], width, label='Fully Vaccinated')

plt.title('Partial vs Full Vaccination by Country', fontsize=16)
plt.xlabel('Country', fontsize=14)
plt.ylabel('Percentage of Population (%)', fontsize=14)
plt.xticks(x, latest_data['location'], rotation=45)
plt.legend()
plt.tight_layout()
plt.show()

# 4. Vaccination Rate vs Death Rate Scatter Plot
plt.figure(figsize=(14, 8))
sns.scatterplot(data=latest_data, x='vaccination_rate', y='death_rate',
                size='total_cases', sizes=(100, 1000), hue='location', legend='brief')

plt.title('Vaccination Rate vs Death Rate by Country', fontsize=16)
```

```python
plt.xlabel('Vaccination Rate (%)', fontsize=14)
plt.ylabel('Death Rate (%)', fontsize=14)
plt.grid(True)
plt.tight_layout()
plt.show()
```







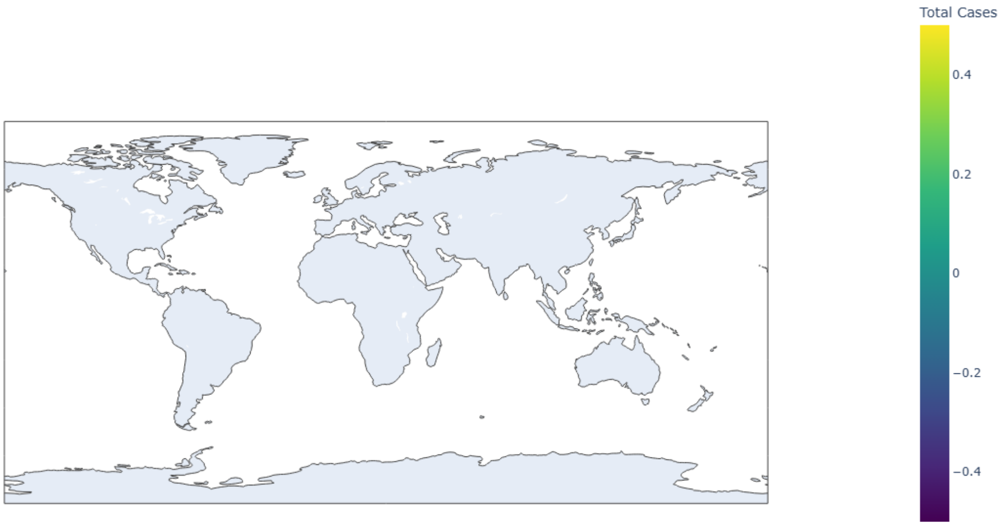


[6]:

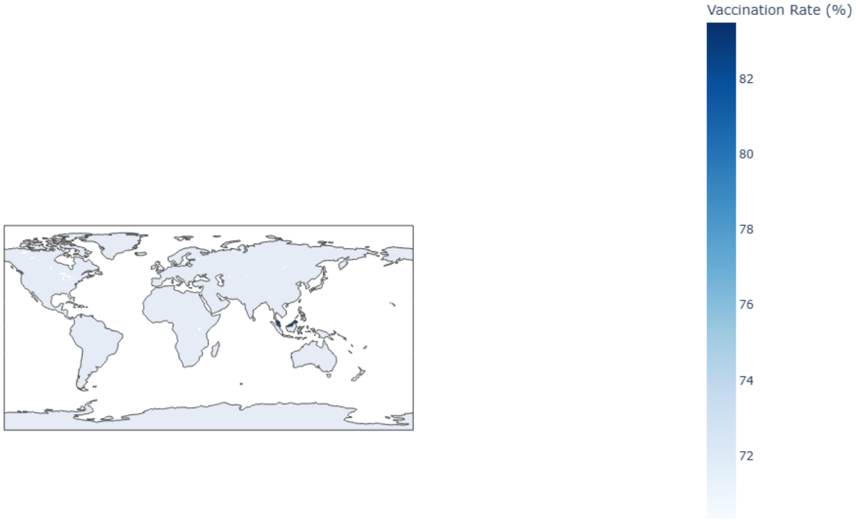
```
# Prepare data for the latest date
latest_date = df['date'].max()
latest_global_data = df[df['date'] == latest_date].copy()

# Calculate death_rate and vaccination_rate columns
```

```
latest_global_data['death_rate'] = latest_global_data['total_deaths'] /␣
 ↪latest_global_data['total_cases'] * 100
latest_global_data['vaccination_rate'] =␣
 ↪latest_global_data['people_vaccinated_per_hundred']
latest_global_data['full_vaccination_rate'] =␣
 ↪latest_global_data['people_fully_vaccinated_per_hundred']

# Create choropleth map for total cases
fig = px.choropleth(
    latest_global_data,
    locations="iso_code",
    color="total_cases",
    hover_name="location",
    color_continuous_scale="Viridis",
    title=f"Total COVID-19 Cases by Country (as of {latest_date.
 ↪strftime('%Y-%m-%d')})",
    labels={'total_cases': 'Total Cases'},
    hover_data=['total_cases', 'total_deaths', 'death_rate']
)
fig.update_layout(height=600, margin={"r":0,"t":50,"l":0,"b":0})
fig.show()

# Create choropleth map for vaccination rates
fig = px.choropleth(
    latest_global_data,
    locations="iso_code",
    color="vaccination_rate",
    hover_name="location",
    color_continuous_scale="Blues",
    title=f"COVID-19 Vaccination Rate by Country (as of {latest_date.
 ↪strftime('%Y-%m-%d')})",
    labels={'vaccination_rate': 'Vaccination Rate (%)'},
    hover_data=['vaccination_rate', 'full_vaccination_rate',␣
 ↪'total_vaccinations']
)
fig.update_layout(height=600, margin={"r":0,"t":50,"l":0,"b":0})
fig.show()
```

Total COVID-19 Cases by Country (as of 2024-08-14)



COVID-19 Vaccination Rate by Country (as of 2024-08-14)



[7]:

```
# Calculate some key statistics for our insights
world_data = cleaned_df[cleaned_df['location'] == 'World'].iloc[-1]
total_global_cases = world_data['total_cases']
total_global_deaths = world_data['total_deaths']
global_death_rate = world_data['death_rate']
global_vaccination_rate = world_data['vaccination_rate']

# Find country with highest vaccination rate
```

```python
highest_vax_country = latest_data.loc[latest_data['vaccination_rate'].idxmax()]
highest_vax_rate = highest_vax_country['vaccination_rate']
highest_vax_name = highest_vax_country['location']

# Find country with lowest death rate (among our selected countries)
lowest_death_country = latest_data.loc[latest_data['death_rate'].idxmin()]
lowest_death_rate = lowest_death_country['death_rate']
lowest_death_name = lowest_death_country['location']

# Calculate correlation between vaccination rate and death rate
correlation = latest_data['vaccination_rate'].corr(latest_data['death_rate'])

print("# Key Insights from COVID-19 Data Analysis")
print(f"\n1. Global Impact: As of {latest_date.strftime('%Y-%m-%d')}, the world has recorded {total_global_cases:,.0f} COVID-19 cases and {total_global_deaths:,.0f} deaths, with a global death rate of {global_death_rate:.2f}%.")
print(f"\n2. Vaccination Progress: Globally, approximately {global_vaccination_rate:.2f}% of the population has received at least one dose of a COVID-19 vaccine.")
print(f"\n3. Vaccination Leaders: {highest_vax_name} has the highest vaccination rate among our analyzed countries at {highest_vax_rate:.2f}%.")
print(f"\n4. Death Rate Variations: {lowest_death_name} has the lowest death rate among our analyzed countries at {lowest_death_rate:.2f}%.")
print(f"\n5. Vaccination and Mortality: The correlation between vaccination rate and death rate is {correlation:.2f}, suggesting {'a negative' if correlation < 0 else 'a positive'} relationship between vaccination coverage and mortality.")
```

```
# Key Insights from COVID-19 Data Analysis

1. Global Impact: As of 2024-08-14, the world has recorded 775,866,783 COVID-19
cases and 7,057,132 deaths, with a global death rate of 0.91%.

2. Vaccination Progress: Globally, approximately 70.61% of the population has
received at least one dose of a COVID-19 vaccine.

3. Vaccination Leaders: Brazil has the highest vaccination rate among our
analyzed countries at 88.08%.

4. Death Rate Variations: France has the lowest death rate among our analyzed
countries at 0.43%.

5. Vaccination and Mortality: The correlation between vaccination rate and death
rate is -0.57, suggesting a negative relationship between vaccination coverage
and mortality.
```

[ ]:

```
```