```
#Import Libraries

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
```

```
#Load Dataset

df = pd.read_csv("country_wise_latest.csv")
```

```
# Quick Data Overview

print("Shape of dataset:", df.shape)
print("\nColumns:", df.columns)
print("\nMissing values:\n", df.isnull().sum())
df.head()
```

```
Shape of dataset: (187, 15)

Columns: Index(['Country/Region', 'Confirmed', 'Deaths', 'Recovered', 'Active',
       'New cases', 'New deaths', 'New recovered', 'Deaths / 100 Cases',
       'Recovered / 100 Cases', 'Deaths / 100 Recovered',
       'Confirmed last week', '1 week change', '1 week % increase',
       'WHO Region'],
      dtype='object')

Missing values:
 Country/Region            0
Confirmed                  0
Deaths                     0
Recovered                  0
Active                     0
New cases                  0
New deaths                 0
New recovered              0
Deaths / 100 Cases         0
Recovered / 100 Cases      0
Deaths / 100 Recovered     0
Confirmed last week        0
1 week change              0
1 week % increase          0
WHO Region                 0
dtype: int64
```

| | Country/Region | Confirmed | Deaths | Recovered | Active | New cases | New deaths | New recovered | Deaths / 100 Cases | Recovered / 100 Cases | Deaths / 100 Recovered | Confirmed last week | 1 week change |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | Afghanistan | 36263 | 1269 | 25198 | 9796 | 106 | 10 | 18 | 3.50 | 69.49 | 5.04 | 35526 | 737 |
| **1** | Albania | 4880 | 144 | 2745 | 1991 | 117 | 6 | 63 | 2.95 | 56.25 | 5.25 | 4171 | 709 |
| **2** | Algeria | 27973 | 1163 | 18837 | 7973 | 616 | 8 | 749 | 4.16 | 67.34 | 6.17 | 23691 | 4282 |
| **3** | Andorra | 907 | 52 | 803 | 52 | 10 | 0 | 0 | 5.73 | 88.53 | 6.48 | 884 | 23 |
| **4** | Angola | 950 | 41 | 242 | 667 | 18 | 1 | 0 | 4.32 | 25.47 | 16.94 | 749 | 201 |

```
# Sort & Basic Summary
top10 = df.sort_values("Confirmed", ascending=False).head(10)
print("\nTop 10 Countries by Confirmed Cases:\n",
      top10[["Country/Region","Confirmed","Deaths","Recovered","Active"]])
```

```
Top 10 Countries by Confirmed Cases:
      Country/Region  Confirmed  Deaths  Recovered   Active
173              US    4290259  148011    1325804  2816444
23           Brazil    2442375   87618    1846641   508116
79            India    1480073   33408     951166   495499
138          Russia     816680   13334     602249   201097
154    South Africa     452529    7067     274925   170537
111          Mexico     395489   44022     303810    47657
132            Peru     389717   18418     272547    98752
35            Chile     347923    9187     319954    1878
177  United Kingdom     301708   45844       1437   25442
```

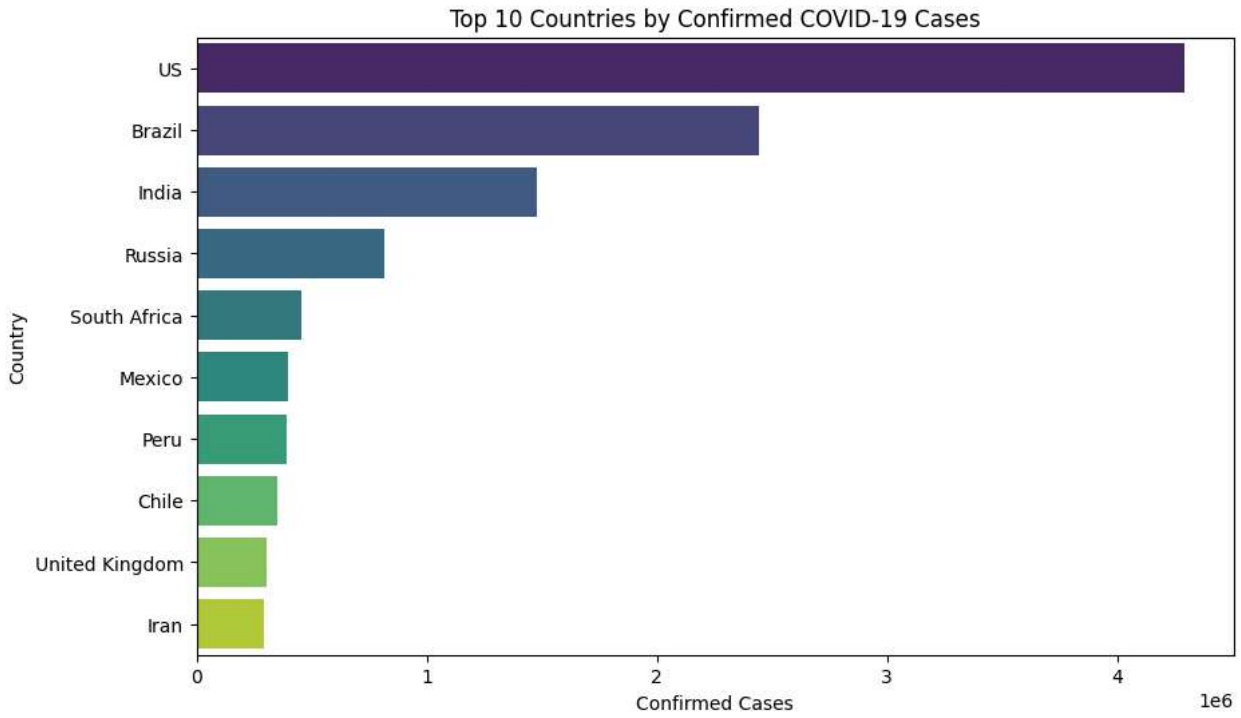
```
81            Iran     293606   15912     255144    22550
```

```
# Bar Plot - Top 10 Countries

plt.figure(figsize=(10,6))
sns.barplot(x="Confirmed", y="Country/Region", data=top10,
            palette="viridis")
plt.title("Top 10 Countries by Confirmed COVID-19 Cases")
plt.xlabel("Confirmed Cases")
plt.ylabel("Country")
plt.show()
```

```
/tmp/ipython-input-319047850.py:4: FutureWarning:


Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `l
```

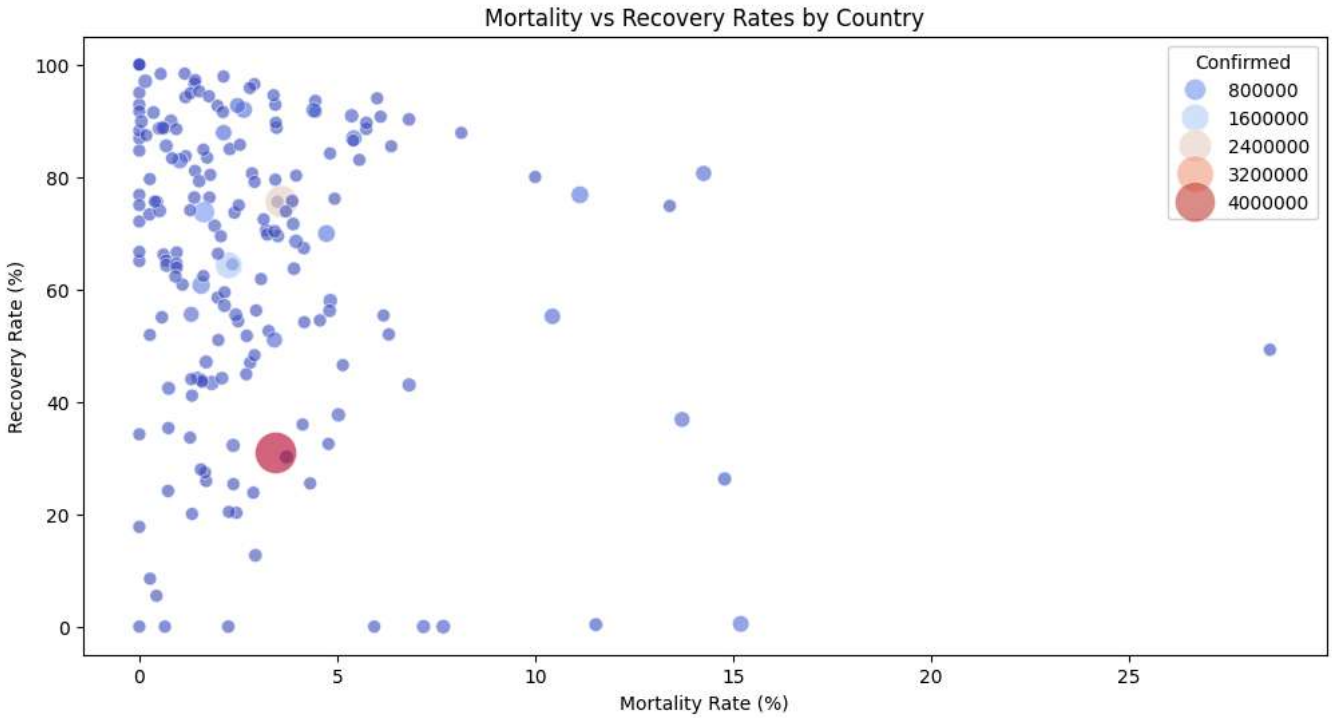



```
# Calculate Mortality & Recovery Rates

df["MortalityRate"] = (df["Deaths"] / df["Confirmed"]) * 100
df["RecoveryRate"] = (df["Recovered"] / df["Confirmed"]) * 100
```

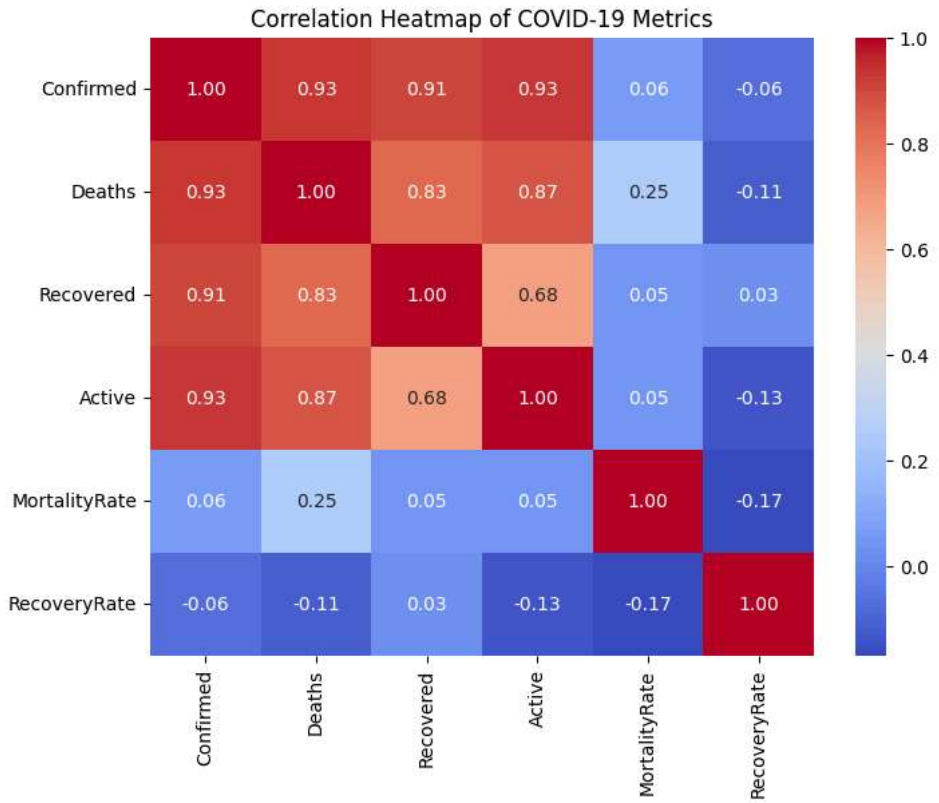
```
# Scatter Plot: Mortality vs Recovery

plt.figure(figsize=(12,6))
sns.scatterplot(x="MortalityRate", y="RecoveryRate",
                size="Confirmed", hue="Confirmed",
                data=df, alpha=0.6, palette="coolwarm", sizes=(50,500))
plt.title("Mortality vs Recovery Rates by Country")
plt.xlabel("Mortality Rate (%)")
plt.ylabel("Recovery Rate (%)")
plt.show()
```




```
# Heatmap - Correlation

plt.figure(figsize=(8,6))
sns.heatmap(df[["Confirmed","Deaths","Recovered","Active",
                "MortalityRate","RecoveryRate"]].corr(),annot=True,
                 cmap="coolwarm", fmt=".2f")
plt.title("Correlation Heatmap of COVID-19 Metrics")
plt.show()
```

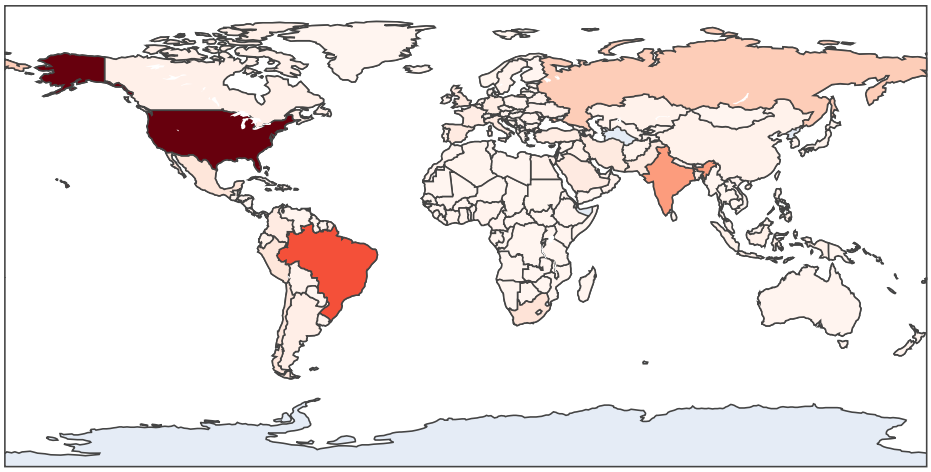



```
 # Interactive Choropleth Map (Confirmed Cases)

fig = px.choropleth(df, locations="Country/Region",
```

```
                    locationmode="country names",
                    color="Confirmed",
                    hover_name="Country/Region",
                    color_continuous_scale="Reds",
                    title="Global COVID-19 Confirmed Cases")
fig.show()
```

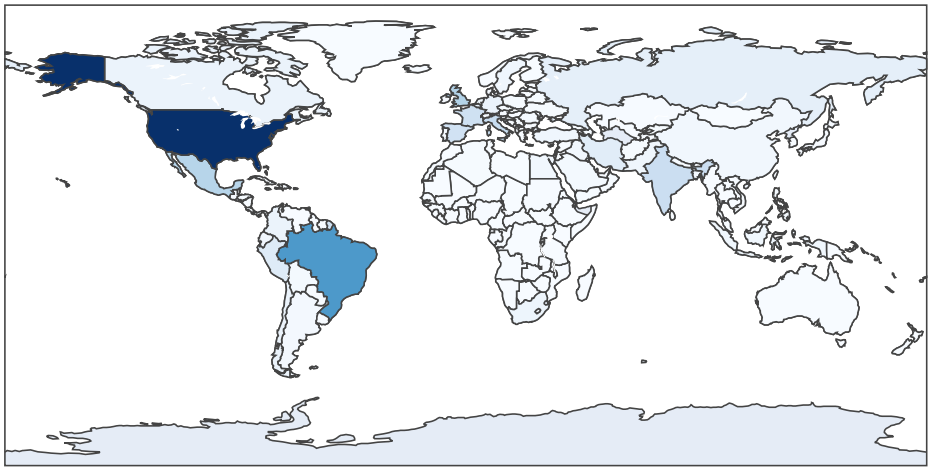
Global COVID-19 Confirmed Cases



```
# Interactive Choropleth Map (Deaths)

fig = px.choropleth(df, locations="Country/Region",
                    locationmode="country names",
                    color="Deaths",
                    hover_name="Country/Region",
                    color_continuous_scale="Blues",
                    title="Global COVID-19 Deaths")
fig.show()
```

Global COVID-19 Deaths

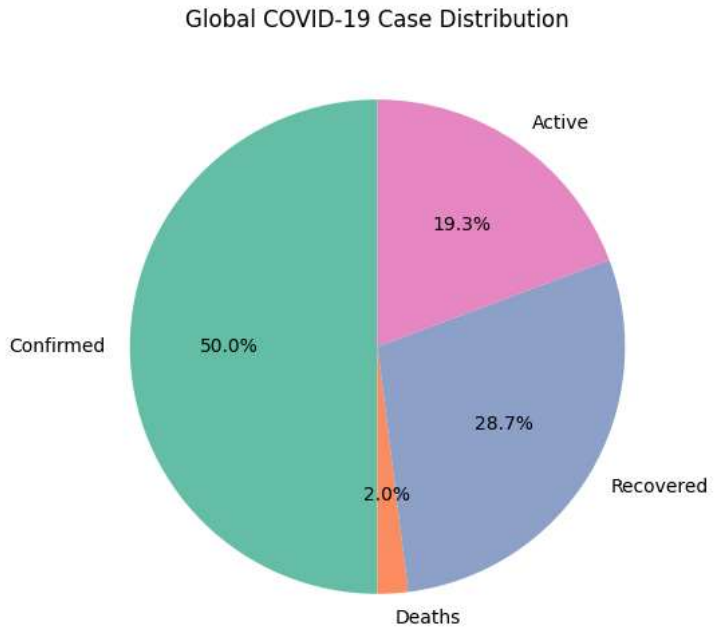


```
# Key Insights (printed summary)
print("🔹 Highest confirmed cases:", df.loc[df['Confirmed'].idxmax(),
          "Country/Region"])
print("🔹 Highest deaths:", df.loc[df['Deaths'].idxmax(),
          "Country/Region"])
print("🔹 Highest recovery rate:", df.loc[df['RecoveryRate'].idxmax(),
          "Country/Region"])
print("🔹 Highest mortality rate:", df.loc[df['MortalityRate'].idxmax(),
          "Country/Region"])
```

```
🔹 Highest confirmed cases: US
🔹 Highest deaths: US
🔹 Highest recovery rate: Dominica
🔹 Highest mortality rate: Yemen
```

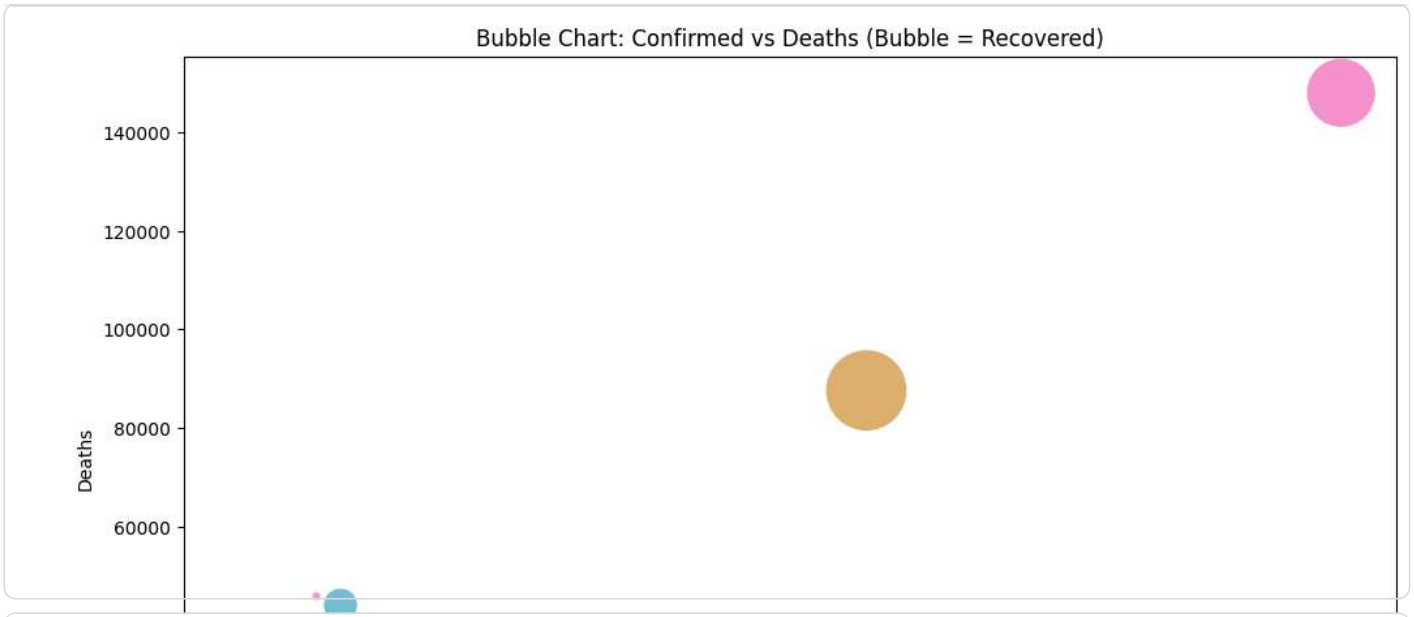
```
# Show global case distribution (Active vs Deaths vs Recovered)

global_cases = df[["Confirmed","Deaths","Recovered","Active"]].sum()
plt.figure(figsize=(6,6))
plt.pie(global_cases, labels=global_cases.index, autopct="%1.1f%%",
        startangle=90, colors=sns.color_palette("Set2"))
plt.title("Global COVID-19 Case Distribution")
plt.show()
```




```
# Show countries with bubble size = confirmed cases

plt.figure(figsize=(12,8))
sns.scatterplot(x="Confirmed", y="Deaths", size="Recovered",
                hue="Country/Region", data=df, alpha=0.7, legend=False,
                sizes=(20,2000))
plt.title("Bubble Chart: Confirmed vs Deaths (Bubble = Recovered)")
plt.show()
```




```
# Hierarchical view of cases by country

fig = px.treemap(df, path=["Country/Region"], values="Confirmed",
                 color="Deaths", hover_data=["Recovered"],
                 color_continuous_scale="Reds",
                 title="Treemap of COVID-19 Cases by Country")
fig.show()
```

Treemap of COVID-19 Cases by Country