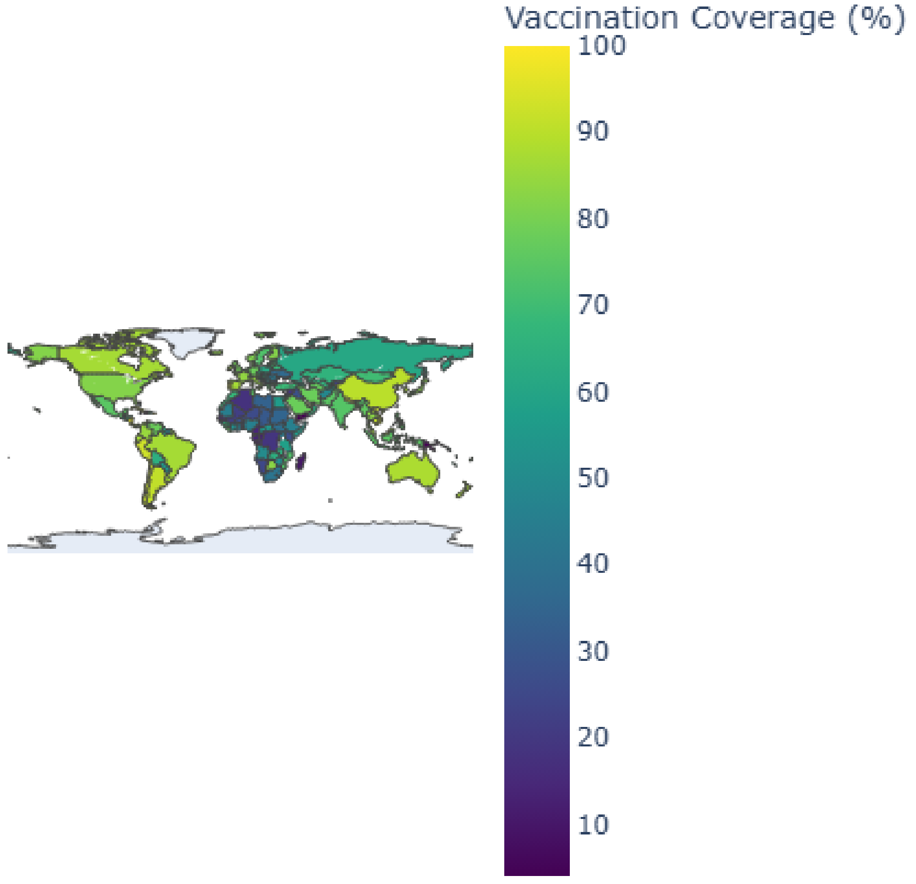
a Choropleth Map

May 14, 2025

[1]:
```python
import pandas as pd
import plotly.express as px

# Read the CSV file
df = pd.read_csv('vaccination-data.csv')

# Create the choropleth map
fig = px.choropleth(
    df,
    locations='ISO3',  # Column containing country codes
    color='PERSONS_VACCINATED_1PLUS_DOSE_PER100',  # Column for color scale
    hover_name='COUNTRY',  # Country name in hover tooltip
    color_continuous_scale='Viridis',  # Color scale
    title='COVID-19 Vaccination Coverage (% of Population with at least 1␣
 ↪dose)',
    labels={
        'PERSONS_VACCINATED_1PLUS_DOSE_PER100': 'Vaccination Coverage (%)'
    }
)

# Update layout
fig.update_layout(
    title_x=0.5,  # Center the title
    geo=dict(
        showframe=False,
        showcoastlines=True,
        projection_type='equirectangular'
    ),
    width=1000,
    height=600
)

# Show the map
fig.show()

# Optional: Save the map to HTML file
fig.write_html("vaccination_choropleth.html")
```

D-19 Vaccination Coverage (% of Population with at least 1



```
[ ]:
```

Data Cleaning

May 14, 2025

[1]:
```python
import pandas as pd

# Read the CSV file
df = pd.read_csv('vaccination-data.csv')

# List of countries of interest
countries_of_interest = ['Kenya', 'USA', 'India', 'South Africa', 'Nigeria']

# Filter for countries of interest
df_filtered = df[df['COUNTRY'].isin(countries_of_interest)]

# Convert DATE_UPDATED to datetime
df_filtered['DATE_UPDATED'] = pd.to_datetime(df_filtered['DATE_UPDATED'])

# Drop rows where DATE_UPDATED or TOTAL_VACCINATIONS is missing
df_cleaned = df_filtered.dropna(subset=['DATE_UPDATED', 'TOTAL_VACCINATIONS'])

# Fill missing numeric values with 0 for vaccination columns
numeric_columns = [
    'PERSONS_VACCINATED_1PLUS_DOSE',
    'TOTAL_VACCINATIONS_PER100',
    'PERSONS_VACCINATED_1PLUS_DOSE_PER100',
    'PERSONS_LAST_DOSE',
    'PERSONS_LAST_DOSE_PER100',
    'PERSONS_BOOSTER_ADD_DOSE',
    'PERSONS_BOOSTER_ADD_DOSE_PER100'
]

df_cleaned[numeric_columns] = df_cleaned[numeric_columns].fillna(0)

# Display cleaned data
print("\nCleaned Vaccination Data:")
print(df_cleaned[['COUNTRY', 'DATE_UPDATED', 'TOTAL_VACCINATIONS',␣
  ↪'PERSONS_VACCINATED_1PLUS_DOSE']].to_string())

# Save cleaned data to new CSV
df_cleaned.to_csv('cleaned_vaccination_data.csv', index=False)
```

```python
print("\nCleaned data saved to 'cleaned_vaccination_data.csv'")

# Display summary statistics
print("\nSummary Statistics:")
print(df_cleaned[['TOTAL_VACCINATIONS', 'PERSONS_VACCINATED_1PLUS_DOSE']].
 ↪describe())
```

```
Cleaned Vaccination Data:
          COUNTRY DATE_UPDATED  TOTAL_VACCINATIONS
PERSONS_VACCINATED_1PLUS_DOSE
81          Kenya   2023-12-31        2.375043e+07
1.449437e+07
110       Nigeria   2023-12-31        1.330480e+08
9.382944e+07
159  South Africa   2023-12-31        4.179881e+07
2.421095e+07
174         India   2023-11-23        2.208365e+09
1.027420e+09

Cleaned data saved to 'cleaned_vaccination_data.csv'

Summary Statistics:
       TOTAL_VACCINATIONS  PERSONS_VACCINATED_1PLUS_DOSE
count        4.000000e+00                   4.000000e+00
mean         6.017406e+08                   2.899887e+08
std          1.072151e+09                   4.928888e+08
min          2.375043e+07                   1.449437e+07
25%          3.728672e+07                   2.178181e+07
50%          8.742342e+07                   5.902019e+07
75%          6.518773e+08                   3.272270e+08
max          2.208365e+09                   1.027420e+09

/tmp/ipykernel_291/2976005593.py:13: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_filtered['DATE_UPDATED'] = pd.to_datetime(df_filtered['DATE_UPDATED'])
```

[ ]:

Data Loading & Exploration

May 14, 2025

[3]:
```python
# Import pandas library
import pandas as pd

# Load the CSV file
# FIXED: Update the file path to the correct location of your CSV file
# Either provide the absolute path correctly or use a relative path if the file␣
 ↪is in the same directory
df = pd.read_csv('vaccination-data.csv')  # If the file is in the same␣
 ↪directory as your notebook
# Alternatively: df = pd.read_csv('/correct/path/to/vaccination-data.csv')

# Display column names
print("Columns in the dataset:")
print(df.columns)
print("\n")

# Display first 5 rows
print("First 5 rows of the dataset:")
print(df.head())
print("\n")

# Check missing values
print("Missing values in each column:")
print(df.isnull().sum())
```

```
Columns in the dataset:
Index(['COUNTRY', 'ISO3', 'WHO_REGION', 'DATA_SOURCE', 'DATE_UPDATED',
       'TOTAL_VACCINATIONS', 'PERSONS_VACCINATED_1PLUS_DOSE',
       'TOTAL_VACCINATIONS_PER100', 'PERSONS_VACCINATED_1PLUS_DOSE_PER100',
       'PERSONS_LAST_DOSE', 'PERSONS_LAST_DOSE_PER100', 'VACCINES_USED',
       'FIRST_VACCINE_DATE', 'NUMBER_VACCINES_TYPES_USED',
       'PERSONS_BOOSTER_ADD_DOSE', 'PERSONS_BOOSTER_ADD_DOSE_PER100'],
      dtype='object')



First 5 rows of the dataset:
       COUNTRY ISO3 WHO_REGION DATA_SOURCE DATE_UPDATED  TOTAL_VACCINATIONS  \
0        Aruba  ABW       AMRO   REPORTING   2023-12-29            217124.0
```

```
1  Afghanistan  AFG       EMRO   REPORTING   2023-12-31          22964750.0
2       Angola  AGO       AFRO   REPORTING   2023-12-31          27819132.0
3     Anguilla  AIA       AMRO   REPORTING   2023-12-29             24864.0
4      Albania  ALB       EURO   REPORTING   2023-12-23           3088966.0

   PERSONS_VACCINATED_1PLUS_DOSE  TOTAL_VACCINATIONS_PER100  \
0                        90493.0                      203.0
1                     19151369.0                       59.0
2                     16550642.0                       85.0
3                        10858.0                      166.0
4                      1349255.0                      107.0

   PERSONS_VACCINATED_1PLUS_DOSE_PER100  PERSONS_LAST_DOSE  \
0                                  85.0            84363.0
1                                  49.0         18370386.0
2                                  50.0          9609080.0
3                                  72.0            10382.0
4                                  47.0          1279333.0

   PERSONS_LAST_DOSE_PER100  VACCINES_USED FIRST_VACCINE_DATE  \
0                      79.0            NaN         2021-02-17
1                      47.0            NaN         2021-02-22
2                      29.0            NaN         2021-03-10
3                      69.0            NaN         2021-02-05
4                      44.0            NaN         2021-01-13

   NUMBER_VACCINES_TYPES_USED  PERSONS_BOOSTER_ADD_DOSE  \
0                         NaN                   35659.0
1                         NaN                 2729940.0
2                         NaN                 3067091.0
3                         NaN                    3231.0
4                         NaN                  402371.0

   PERSONS_BOOSTER_ADD_DOSE_PER100
0                             33.0
1                              7.0
2                              9.0
3                             22.0
4                             14.0


Missing values in each column:
COUNTRY                                 0
ISO3                                    0
WHO_REGION                              6
DATA_SOURCE                             0
DATE_UPDATED                            7
TOTAL_VACCINATIONS                      6
```

```
PERSONS_VACCINATED_1PLUS_DOSE             6
TOTAL_VACCINATIONS_PER100                 8
PERSONS_VACCINATED_1PLUS_DOSE_PER100      8
PERSONS_LAST_DOSE                         6
PERSONS_LAST_DOSE_PER100                  8
VACCINES_USED                           215
FIRST_VACCINE_DATE                       19
NUMBER_VACCINES_TYPES_USED              215
PERSONS_BOOSTER_ADD_DOSE                 20
PERSONS_BOOSTER_ADD_DOSE_PER100          31
dtype: int64
```

[ ]:

```

```

Exploratory Data Analysis (EDA)

May 14, 2025

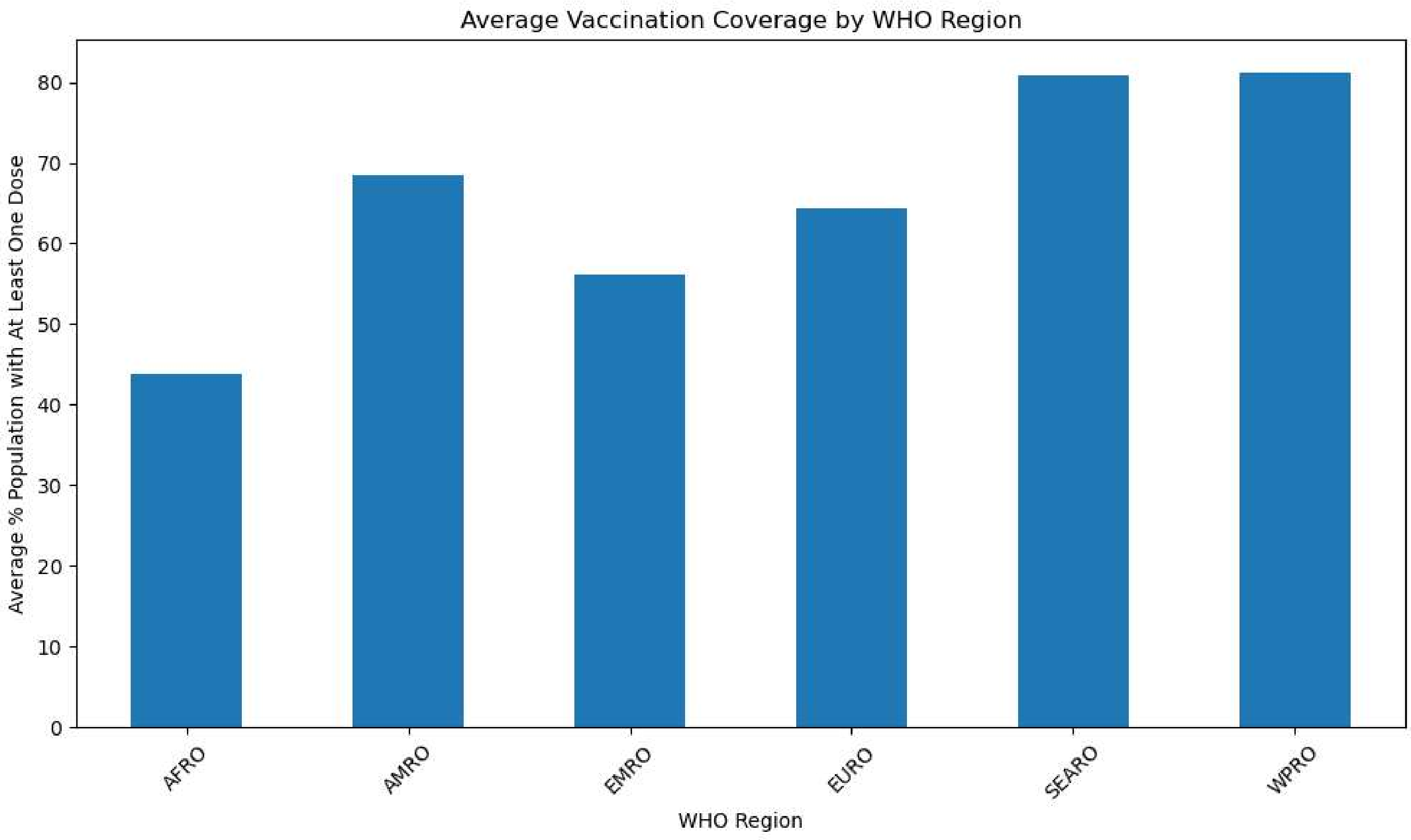
[1]:
```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np

# Read the data
df = pd.read_csv('vaccination-data.csv')

# 1. Basic Data Cleaning
# Convert percentage columns to numeric, removing '%' if present
percentage_cols = [col for col in df.columns if 'PER100' in col]
for col in percentage_cols:
    df[col] = pd.to_numeric(df[col], errors='coerce')

# 2. Top 10 Countries by Total Vaccinations
plt.figure(figsize=(12, 6))
top_10_vacc = df.nlargest(10, 'TOTAL_VACCINATIONS')
sns.barplot(data=top_10_vacc, x='TOTAL_VACCINATIONS', y='COUNTRY')
plt.title('Top 10 Countries by Total Vaccinations')
plt.xlabel('Total Vaccinations')
plt.ticklabel_format(style='plain', axis='x')
plt.tight_layout()
plt.savefig('top_10_vaccinations.png')
plt.close()

# 3. Vaccination Coverage Analysis
plt.figure(figsize=(12, 6))
top_10_coverage = df.nlargest(10, 'PERSONS_VACCINATED_1PLUS_DOSE_PER100')
sns.barplot(data=top_10_coverage,
          x='PERSONS_VACCINATED_1PLUS_DOSE_PER100',
          y='COUNTRY')
plt.title('Top 10 Countries by Vaccination Coverage (%)')
plt.xlabel('Percentage of Population with At Least One Dose')
plt.tight_layout()
plt.savefig('top_10_coverage.png')
plt.close()
```

```
# 4. Regional Analysis
plt.figure(figsize=(10, 6))
region_avg = df.groupby('WHO_REGION')['PERSONS_VACCINATED_1PLUS_DOSE_PER100'].
  ↪mean()
region_avg.plot(kind='bar')
plt.title('Average Vaccination Coverage by WHO Region')
plt.xlabel('WHO Region')
plt.ylabel('Average % Population with At Least One Dose')
plt.xticks(rotation=45)
plt.tight_layout()
plt.savefig('regional_coverage.png')
plt.close()

# 5. Correlation Analysis
numeric_cols = ['TOTAL_VACCINATIONS', 'PERSONS_VACCINATED_1PLUS_DOSE',
                'TOTAL_VACCINATIONS_PER100',␣
  ↪'PERSONS_VACCINATED_1PLUS_DOSE_PER100',
                'PERSONS_BOOSTER_ADD_DOSE_PER100']
correlation = df[numeric_cols].corr()

plt.figure(figsize=(10, 8))
sns.heatmap(correlation, annot=True, cmap='coolwarm', center=0)
plt.title('Correlation Between Vaccination Metrics')
plt.tight_layout()
plt.savefig('correlation_matrix.png')
plt.close()

# 6. Print Summary Statistics
print("\nSummary Statistics:")
print(df[numeric_cols].describe())

# 7. Calculate Global Vaccination Progress
total_global = {
    'Total Vaccinations': df['TOTAL_VACCINATIONS'].sum(),
    'People with At Least One Dose': df['PERSONS_VACCINATED_1PLUS_DOSE'].sum(),
    'People Fully Vaccinated': df['PERSONS_LAST_DOSE'].sum(),
    'People with Booster': df['PERSONS_BOOSTER_ADD_DOSE'].sum()
}

print("\nGlobal Vaccination Progress:")
for metric, value in total_global.items():
    print(f"{metric}: {value:,.0f}")
```

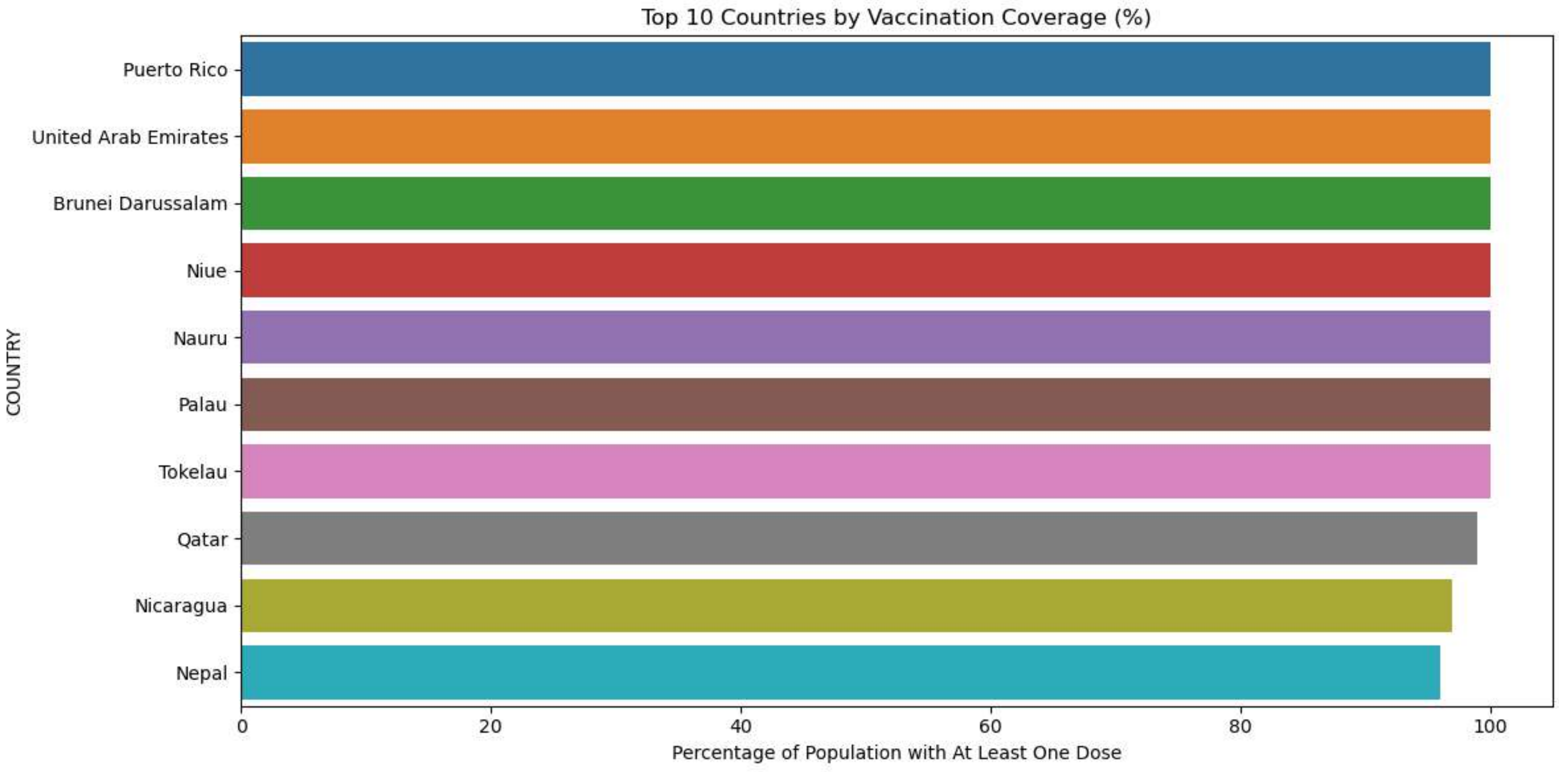
```
Summary Statistics:
       TOTAL_VACCINATIONS  PERSONS_VACCINATED_1PLUS_DOSE  \
count        2.090000e+02                   2.090000e+02
```

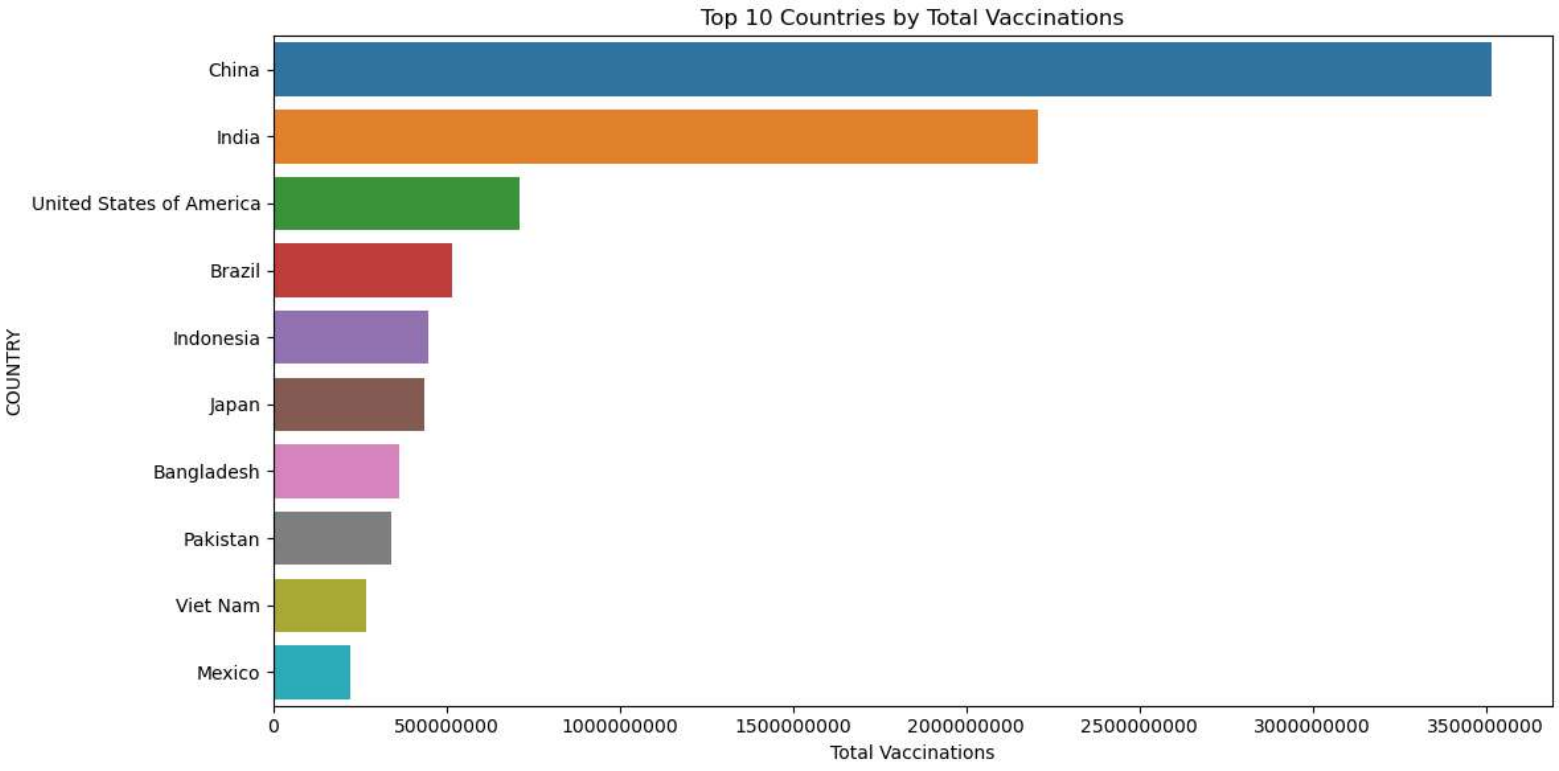
```
mean        6.535034e+07                   2.684238e+07
std         2.967499e+08                   1.190477e+08
min         4.619000e+03                   1.638000e+03
25%         9.034240e+05                   4.511490e+05
50%         7.372208e+06                   3.746041e+06
75%         2.732206e+07                   1.356837e+07
max         3.516881e+09                   1.318027e+09

       TOTAL_VACCINATIONS_PER100  PERSONS_VACCINATED_1PLUS_DOSE_PER100  \
count                 207.000000                            207.000000
mean                  153.932367                             62.768116
std                    85.721752                             24.061901
min                     4.000000                              4.000000
25%                    82.000000                             45.000000
50%                   154.000000                             67.000000
75%                   221.500000                             83.000000
max                   470.000000                            100.000000

       PERSONS_BOOSTER_ADD_DOSE_PER100
count                       184.000000
mean                         33.125000
std                          23.949186
min                           1.000000
25%                          10.000000
50%                          31.000000
75%                          56.000000
max                          83.000000

Global Vaccination Progress:
Total Vaccinations: 13,658,220,081
People with At Least One Dose: 5,610,057,847
People Fully Vaccinated: 5,167,774,734
People with Booster: 2,507,135,738
```

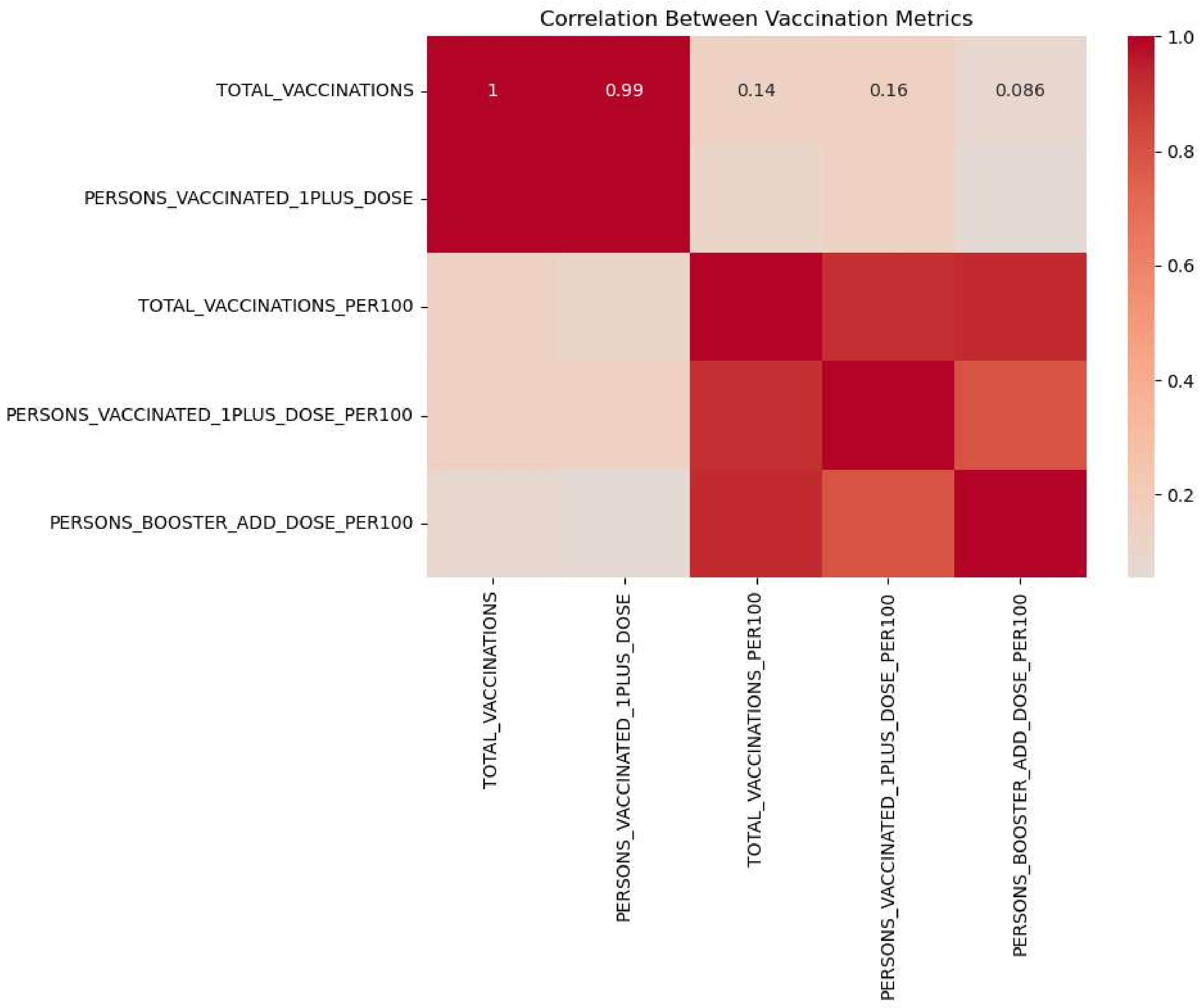
[ ]:



Average Vaccination Coverage by WHO Region
Average % Population with At Least One Dose
WHO Region
0
10
20
30
40
50
60
70
80
AFRO
AMRO
EMRO
EURO
SEARO
WPRO


Top 10 Countries by Vaccination Coverage (%)
COUNTRY
Puerto Rico
United Arab Emirates
Brunei Darussalam
Niue
Nauru
Palau
Tokelau
Qatar
Nicaragua
Nepal
0
20
40
60
80
100
Percentage of Population with At Least One Dose


Top 10 Countries by Total Vaccinations
China
India
United States of America
Brazil
Indonesia
Japan
Bangladesh
Pakistan
Viet Nam
Mexico
COUNTRY
0
500000000
1000000000
1500000000
2000000000
2500000000
3000000000
3500000000
Total Vaccinations

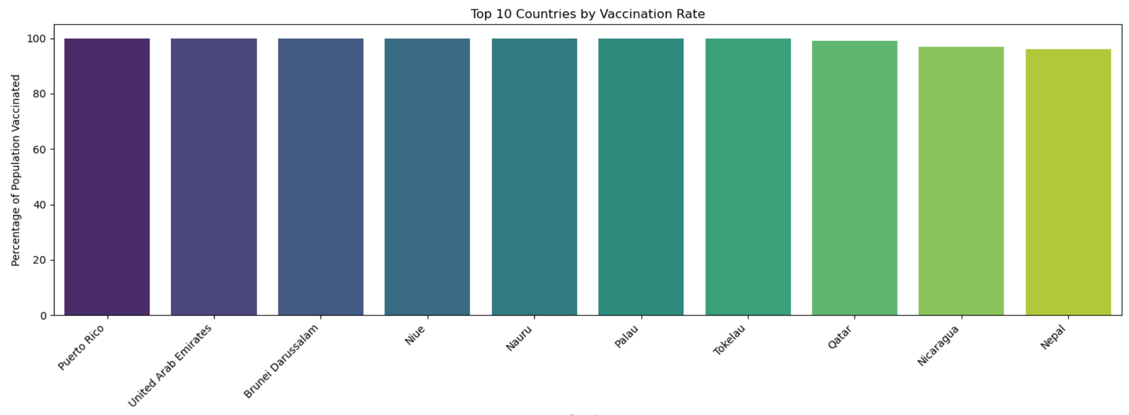

Correlation Between Vaccination Metrics
TOTAL_VACCINATIONS
PERSONS_VACCINATED_1PLUS_DOSE
TOTAL_VACCINATIONS_PER100
PERSONS_VACCINATED_1PLUS_DOSE_PER100
PERSONS_BOOSTER_ADD_DOSE_PER100
1
0.99
0.14
0.16
0.086
1.0
0.8
0.6
0.4
0.2

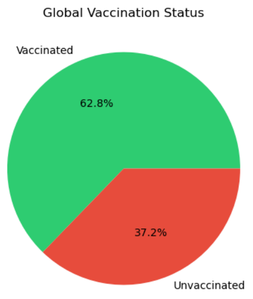
Visualizing Vaccination Progress

May 14, 2025

[1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Read the CSV file
df = pd.read_csv('vaccination-data.csv')

# Clean and prepare the data
df['PERSONS_VACCINATED_1PLUS_DOSE_PER100'] = pd.
 ↪to_numeric(df['PERSONS_VACCINATED_1PLUS_DOSE_PER100'], errors='coerce')

# Select top 10 countries by vaccination percentage
top_10_countries = df.nlargest(10, 'PERSONS_VACCINATED_1PLUS_DOSE_PER100')

# Create a figure with two subplots
plt.figure(figsize=(15, 10))

# Plot 1: Bar chart of vaccination percentages
plt.subplot(2, 1, 1)
sns.barplot(data=top_10_countries,
            x='COUNTRY',
            y='PERSONS_VACCINATED_1PLUS_DOSE_PER100',
            palette='viridis')
plt.xticks(rotation=45, ha='right')
plt.title('Top 10 Countries by Vaccination Rate')
plt.xlabel('Country')
plt.ylabel('Percentage of Population Vaccinated')

# Plot 2: Pie chart for global vaccination status
plt.subplot(2, 1, 2)
global_stats = df[['PERSONS_VACCINATED_1PLUS_DOSE_PER100']].mean()
remaining = 100 - global_stats['PERSONS_VACCINATED_1PLUS_DOSE_PER100']

plt.pie([global_stats['PERSONS_VACCINATED_1PLUS_DOSE_PER100'], remaining],
        labels=['Vaccinated', 'Unvaccinated'],
        autopct='%1.1f%%',
        colors=['#2ecc71', '#e74c3c'])
```

```
plt.title('Global Vaccination Status')

# Adjust layout and display
plt.tight_layout()
plt.show()

# Additional analysis: Print statistics
print("\nVaccination Statistics:")
print("-" * 50)
print(f"Global average vaccination rate:␣
↪{global_stats['PERSONS_VACCINATED_1PLUS_DOSE_PER100']:.2f}%")
print(f"Number of countries analyzed: {len(df)}")
print("\nTop 5 Most Vaccinated Countries:")
for idx, row in top_10_countries.head().iterrows():
    print(f"{row['COUNTRY']}: {row['PERSONS_VACCINATED_1PLUS_DOSE_PER100']:.
↪1f}%")
```





```
Vaccination Statistics:
--------------------------------------------------
Global average vaccination rate: 62.77%
Number of countries analyzed: 215

Top 5 Most Vaccinated Countries:
```

```
Puerto Rico: 100.0%
United Arab Emirates: 100.0%
Brunei Darussalam: 100.0%
Niue: 100.0%
Nauru: 100.0%
```

```
[ ]:
```