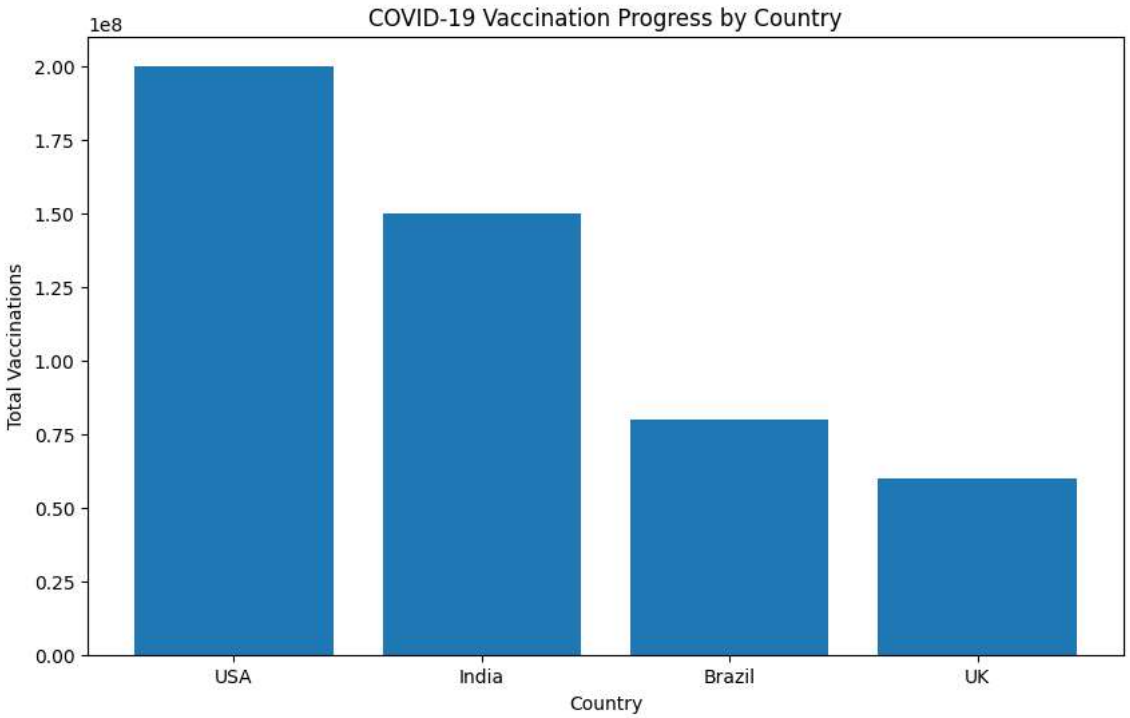
```
from google.colab import drive
drive.mount('/content/drive/')
```

```
Drive already mounted at /content/drive/; to attempt to forcibly remount, call drive.mount("/content/drive/", force_remount=True).
```

visulaization

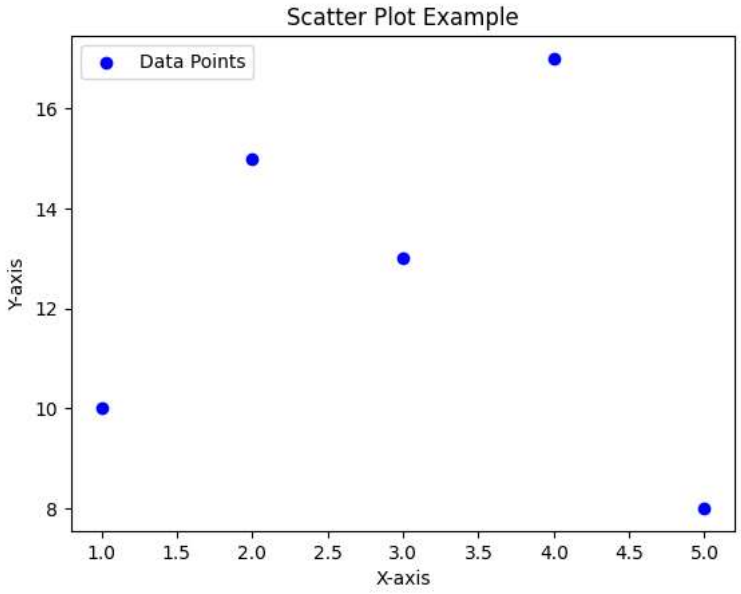
```
import matplotlib.pyplot as plt
import pandas as pd

# Sample data
data = {
    'Country': ['USA', 'India', 'Brazil', 'UK'],
    'Total Vaccinations': [200000000, 150000000, 80000000, 60000000],
}

df = pd.DataFrame(data)

plt.figure(figsize=(10, 6))
plt.bar(df['Country'], df['Total Vaccinations'])
plt.title('COVID-19 Vaccination Progress by Country')
plt.xlabel('Country')
plt.ylabel('Total Vaccinations')
plt.show()
```



visualization for scatter plots

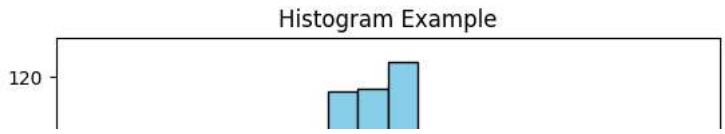
```
import matplotlib.pyplot as plt

# Sample data
x = [1, 2, 3, 4, 5]
y = [10, 15, 13, 17, 8]

# Create a scatter plot
plt.scatter(x, y, label='Data Points', color='blue', marker='o')

# Add labels and a title
plt.xlabel('X-axis')
```

```
plt.ylabel('Y-axis')
plt.title('Scatter Plot Example')

# Add a legend
plt.legend()

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



visualization for histogram

```
import matplotlib.pyplot as plt
import numpy as np

# Sample data (replace this with your own data)
data = np.random.randn(1000)  # Generating random data for demonstration

# Create a histogram
plt.hist(data, bins=20, color='skyblue', edgecolor='black')

# Add labels and a title
plt.xlabel('Value')
plt.ylabel('Frequency')
plt.title('Histogram Example')

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



statistical analysis

Double-click (or enter) to edit

step 1:To find mean ,median,standard deviation?

```
import numpy as np

data = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9, 10])

# Mean
mean_value = np.mean(data)

# Median
median_value = np.median(data)

# Standard Deviation
std_deviation = np.std(data)

print(f"Mean: {mean_value}, Median: {median_value}, Standard Deviation: {std_deviation}")
```

```
Mean: 5.5, Median: 5.5, Standard Deviation: 2.8722813232690143
```

Hypothesis Testing with SciPy:

```
from scipy.stats import ttest_ind

# Sample data for two groups
group1 = [23, 25, 28, 32, 35]
group2 = [18, 22, 26, 30, 33]

# Perform t-test
t_statistic, p_value = ttest_ind(group1, group2)

print(f"T-statistic: {t_statistic}, P-value: {p_value}")
```

```
T-statistic: 0.8049434044064967, P-value: 0.44411480871450937
```

Linear Regression with StatsModels:

```
import statsmodels.api as sm
import numpy as np

# Sample data
x = np.array([1, 2, 3, 4, 5])
y = np.array([2, 4, 5, 4, 5])

# Add a constant term to the independent variable
x = sm.add_constant(x)

# Fit the model
model = sm.OLS(y, x)
results = model.fit()

# Print regression results
print(results.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.600
Model:                            OLS   Adj. R-squared:                  0.467
Method:                 Least Squares   F-statistic:                     4.500
Date:                Thu, 19 Oct 2023   Prob (F-statistic):              0.124
Time:                        07:03:24   Log-Likelihood:                -5.2598
No. Observations:                   5   AIC:                             14.52
Df Residuals:                       3   BIC:                             13.74
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          2.2000      0.938      2.345      0.101      -0.785       5.185
x1             0.6000      0.283      2.121      0.124      -0.300       1.500
==============================================================================
Omnibus:                        nan   Durbin-Watson:                   2.017
Prob(Omnibus):                  nan   Jarque-Bera (JB):                0.570
Skew:                         0.289   Prob(JB):                        0.752
Kurtosis:                     1.450   Cond. No.                         8.37
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
/usr/local/lib/python3.10/dist-packages/statsmodels/stats/stattools.py:74: ValueWarning: omni_normtest is not valid with less than 8 ob
  warn("omni_normtest is not valid with less than 8 observations; %i "
```

performing exploratory data analysis

```
from google.colab import drive
drive.mount('/content/drive')


import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Load your dataset
data = pd.read_csv('/content/drive/MyDrive/covid-vaccine-willingness-and-people-vaccinated-by-country.csv')

# Display the first few rows of the dataset
print(data.head())

# Summary statistics
print(data.describe())

# Data distribution visualization
# Example: Histogram of a numeric column
plt.figure(figsize=(8, 6))
plt.title("Histogram of a Numeric Column")

plt.xlabel('Value')
plt.ylabel('Frequency')
plt.show()

# Example: Count plot for a categorical column
plt.figure(figsize=(8, 6))
plt.title("Count Plot of a Categorical Column")

plt.xticks(rotation=45)
plt.show()

# Correlation heatmap (for numeric columns)
correlation_matrix = data.corr()
plt.figure(figsize=(10, 8))
plt.title("Correlation Heatmap")
sns.heatmap(correlation_matrix, annot=True, cmap="coolwarm", linewidths=.5)
plt.show()
```

```
      Entity Code         Day  people_vaccinated_per_hundred  \
0  Australia  AUS  2021-02-28                           0.13
1     Canada  CAN  2021-01-31                           2.24
2     Canada  CAN  2021-02-28                           3.61
3     Canada  CAN  2021-03-31                          13.26
4     Canada  CAN  2021-04-30                          32.67

   willingness_covid_vaccinate_this_week_pct_pop  \
0                                          52.91
1                                          54.26
2                                          53.56
3                                          51.96
4                                          36.12

   uncertain_covid_vaccinate_this_week_pct_pop  \
0                                        19.03
1                                        15.56
2                                        15.65
3                                        12.21
4                                        10.09

   unwillingness_covid_vaccinate_this_week_pct_pop
0                                            27.93
1                                            27.94
2                                            27.18
3                                            22.57
4                                            21.12
       people_vaccinated_per_hundred  \
count                       49.000000
mean                        11.295306
std                         12.917389
min                          0.000000
25%                          2.280000
50%                          5.490000
75%                         13.690000
max                         50.620000

       willingness_covid_vaccinate_this_week_pct_pop  \
count                                      49.000000
mean                                       47.548980
std                                        10.784062
min                                        19.570000
25%                                        40.160000
50%                                        51.270000
75%                                        54.230000
max                                        67.400000

       uncertain_covid_vaccinate_this_week_pct_pop  \
count                                     49.000000
mean                                      15.423061
std                                        5.929520
min                                        3.990000
25%                                       11.960000
50%                                       15.100000
75%                                       17.710000
max                                       33.990000
```