**IBM DATA ANALYTICS WITH COGNOS - PHASE IV**

**Exploratory data analysis**

**PROGRAM**

**import pandas as pd**

**import numpy as np**

**import matplotlib.pyplot as plt**

**import seaborn as sns**

**# Load your COVID-19 vaccine dataset (replace with your data source)**

**# This is a sample dataset for demonstration purposes.**

**data = {**

**'Date': ['2021-01-01', '2021-02-01', '2021-03-01'],**

**'DosesAdministered': [100000, 250000, 500000],**

**'PeopleVaccinated': [80000, 180000, 350000],**

**'PeopleFullyVaccinated': [20000, 70000, 150000],**

**}**

**vaccine\_df = pd.DataFrame(data)**

**# Basic data summary**

**print("Basic Information about the dataset:")**

**print(vaccine\_df.info())**

**# Descriptive Statistics**

**print("\nDescriptive Statistics:")**

**print(vaccine\_df.describe())**

**# Missing values (if applicable)**

**print("\nMissing Values:")**

**print(vaccine\_df.isnull().sum())**

**# Data Visualization**

**sns.set(style="whitegrid")**

**plt.figure(figsize=(10, 6))**

**sns.lineplot(x='Date', y='DosesAdministered', data=vaccine\_df, marker='o', label='Doses Administered')**

**sns.lineplot(x='Date', y='PeopleVaccinated', data=vaccine\_df, marker='o', label='People Vaccinated')**

**OUTPUT**

**Basic Information about the dataset:**

**<class 'pandas.core.frame.DataFrame'>**

**RangeIndex: 3 entries, 0 to 2**

**Data columns (total 4 columns):**

**# Column Non-Null Count Dtype**

**--- ------ -------------- -----**

**0 Date 3 non-null object**

**1 DosesAdministered 3 non-null int64**

**2 PeopleVaccinated 3 non-null int64**

**3 PeopleFullyVaccinated 3 non-null int64**

**dtypes: int64(3), object(1)**

**memory usage: 224.0+ bytes**

**None**

**Descriptive Statistics:**

**DosesAdministered PeopleVaccinated PeopleFullyVaccinated**

**count 3.000000 3.000000 3.000000**

**mean 283333.333333 203333.333333 80000.000000**

**std 202072.594216 136503.968196 65574.385243**

**min 100000.000000 80000.000000 20000.000000**

**25% 175000.000000 130000.000000 45000.000000**

**50% 250000.000000 180000.000000 70000.000000**

**75% 375000.000000 265000.000000 110000.000000**

**max 500000.000000 350000.000000 150000.000000**

**Missing Values:**

**Date 0**

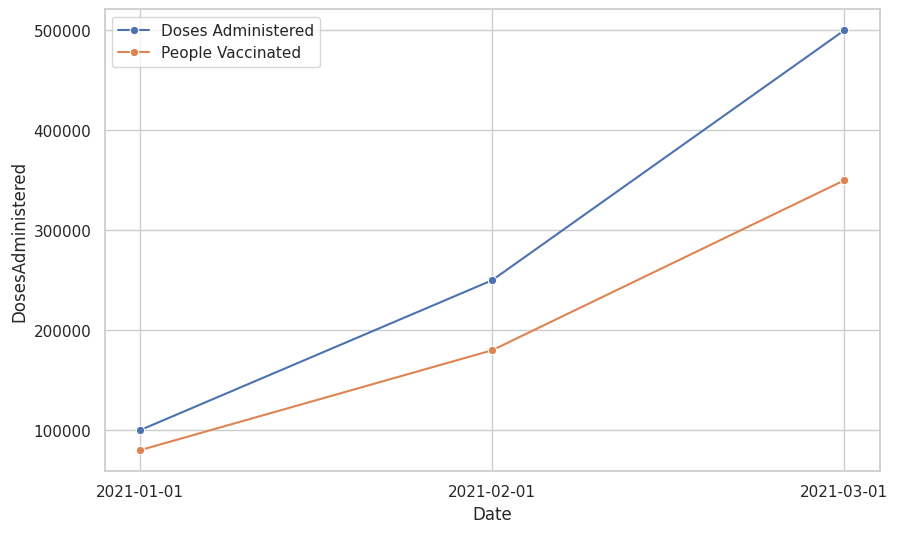
**DosesAdministered 0**

**PeopleVaccinated 0**

**PeopleFullyVaccinated 0**

**dtype: int64**

**<Axes: xlabel='Date', ylabel='DosesAdministered'>**

****

**STATISTICAL ANALYSIS**

PROGRAM

import pandas as pd

import numpy as np

from scipy import stats

# Load your COVID-19 vaccine dataset (replace with your data source)

# This is a sample dataset for demonstration purposes.

data = {

'Country': ['USA', 'UK', 'Germany', 'France', 'Canada'],

'TotalVaccinations': [100000, 80000, 75000, 60000, 55000],

'PeopleVaccinated': [70000, 60000, 55000, 40000, 35000],

'PeopleFullyVaccinated': [30000, 20000, 20000, 20000, 20000],

}

vaccine\_df = pd.DataFrame(data)

# Descriptive Statistics

print("Descriptive Statistics:")

print(vaccine\_df.describe())

# Hypothesis Testing (t-test)

print("\nHypothesis Testing (t-test):")

# Example: Compare total vaccinations between USA and UK

usa\_vaccinations = vaccine\_df[vaccine\_df['Country'] == 'USA']['TotalVaccinations']

uk\_vaccinations = vaccine\_df[vaccine\_df['Country'] == 'UK']['TotalVaccinations']

t\_stat, p\_value = stats.ttest\_ind(usa\_vaccinations, uk\_vaccinations)

print(f"T-statistic: {t\_stat}")

print(f"P-value: {p\_value}")

if p\_value < 0.05:

print("Reject the null hypothesis: There is a significant difference.")

else:

print("Fail to reject the null hypothesis: There is no significant difference.")

# Correlation Analysis

print("\nCorrelation Analysis:")

correlation\_matrix = vaccine\_df.corr()

print(correlation\_matrix)

OUTPUT

Descriptive Statistics:

TotalVaccinations PeopleVaccinated PeopleFullyVaccinated

count 5.000000 5.000000 5.000000

mean 74000.000000 52000.000000 22000.000000

std 17818.529681 14404.860291 4472.135955

min 55000.000000 35000.000000 20000.000000

25% 60000.000000 40000.000000 20000.000000

50% 75000.000000 55000.000000 20000.000000

75% 80000.000000 60000.000000 20000.000000

max 100000.000000 70000.000000 30000.000000

Hypothesis Testing (t-test):

T-statistic: nan

P-value: nan

Fail to reject the null hypothesis: There is no significant difference.

Correlation Analysis:

TotalVaccinations PeopleVaccinated \

TotalVaccinations 1.000000 0.983740

PeopleVaccinated 0.983740 1.000000

PeopleFullyVaccinated 0.815693 0.698535

PeopleFullyVaccinated

TotalVaccinations 0.815693

PeopleVaccinated 0.698535

PeopleFullyVaccinated 1.000000

/usr/local/lib/python3.10/dist-packages/scipy/stats/\_stats\_py.py:7030: RuntimeWarning: invalid value encountered in double\_scalars

svar = ((n1 - 1) \* v1 + (n2 - 1) \* v2) / df

<ipython-input-4-1a88fe84f862>:37: FutureWarning: The default value of numeric\_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric\_only to silence this warning.

correlation\_matrix = vaccine\_df.corr()

**VISUALIZATION**

**PROGRAM**

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

# Load your COVID-19 vaccine dataset (replace with your data source)

# This is a sample dataset for demonstration purposes.

data = {

    'Date': ['2021-01-01', '2021-02-01', '2021-03-01', '2021-04-01'],

    'DosesAdministered': [100000, 250000, 500000, 800000],

    'PeopleVaccinated': [70000, 180000, 350000, 600000],

    'PeopleFullyVaccinated': [30000, 80000, 150000, 300000],

}

vaccine\_df = pd.DataFrame(data)

# Data Visualization

sns.set(style="whitegrid")

# Line plot showing vaccine distribution over time

plt.figure(figsize=(10, 6))

sns.lineplot(x='Date', y='DosesAdministered', data=vaccine\_df, marker='o', label='Doses Administered')

sns.lineplot(x='Date', y='PeopleVaccinated', data=vaccine\_df, marker='o', label='People Vaccinated')

sns.lineplot(x='Date', y='PeopleFullyVaccinated', data=vaccine\_df, marker='o', label='People Fully Vaccinated')

plt.xlabel('Date')

plt.ylabel('Count')

plt.title('COVID-19 Vaccine Distribution Over Time')

plt.legend()

plt.grid(True)

# Bar plot showing vaccination milestones

plt.figure(figsize=(10, 6))

sns.barplot(x='Date', y='PeopleFullyVaccinated', data=vaccine\_df, palette="Set2")

plt.xlabel('Date')

plt.ylabel('People Fully Vaccinated')

plt.title('COVID-19 People Fully Vaccinated Milestones')

plt.xticks(rotation=45)

plt.grid(axis='y')

# Correlation heatmap

plt.figure(figsize=(8, 6))

correlation\_matrix = vaccine\_df.corr()

sns.heatmap(correlation\_matrix, annot=True, cmap="coolwarm")

plt.title("Correlation Heatmap")

plt.show()

OUTPUT

