Data Analytics with Cognos

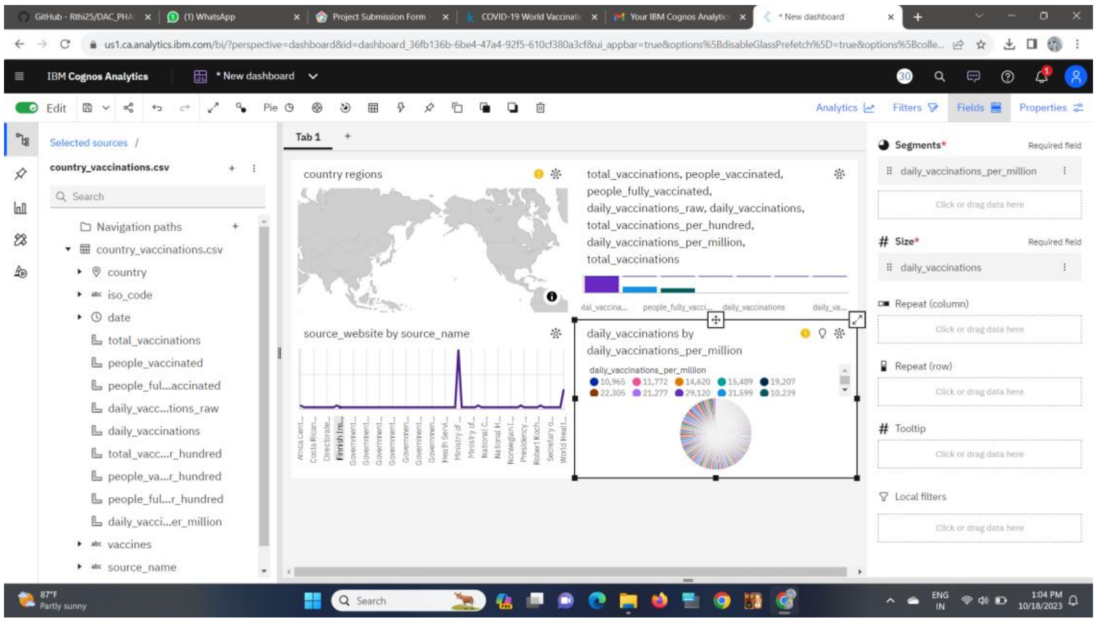
Group2 – Phase_3

Project Title: COVID-19 Vaccines Analysis

Group Members;

SOBIYA (TL)
MOHANKUMAR
NANDHAN
MEENAKSHISUNDARAM
VASANTH KUMAR

IBM Cognos Visualization:



COVID-19 VACCINES ANALYSIS

Phase 3: Development Part 1

Problem:

Begin conducting the Covid-19 vaccines analysis by collecting and preprocessing the data.
Collect and preprocess the Covid-19 vaccine data for analysis.

Program:

STEP 1: DATA COLLECTION

First, you need to obtain the COVID-19 vaccine data from a reliable source. This data can often be found on government health websites, research institutions, or platforms like Kaggle. Download the dataset in a format such as CSV or Excel.

STEP 2: DATA PREPROCESSING

Once you have the dataset, you'll need to preprocess it to make it suitable for analysis. Here's a generic outline of what your preprocessing steps might look like:

```python
 #This Python 3 environment comes with many helpful analytics libraries
installed
# It is defined by the kaggle/python Docker image:
https://github.com/kaggle/docker-python
# For example, here's several helpful packages to load

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

# Input data files are available in the read-only "../input/" directory
# For example, running this (by clicking run or pressing Shift+Enter) will
list all files under the input directory

import matplotlib.pyplot as plt

# plotly
# import plotly.plotly as py
from plotly.offline import init_notebook_mode, iplot, plot
import plotly.express as px
import plotly as py
init_notebook_mode(connected=True)
import plotly.graph_objs as go
import scipy

# seaborn library
import seaborn as sns

# word cloud library
from wordcloud import WordCloud

import os
for dirname, _, filenames in
os.walk('/country_vaccinations_by_manufacturer.csv'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

# You can write up to 20GB to the current directory (/kaggle/working/)
that gets preserved as output when you create a version using "Save & Run
All"
# You can also write temporary files to /kaggle/temp/, but they won't be
saved outside of the current session
```

```python
data = pd.read_csv("/country_vaccinations_by_manufacturer.csv")
```

```
data.head()
```

OUTPUT:

	location	date	vaccine	total_vaccinations
0	Argentina	2020-12-29	Moderna	2
1	Argentina	2020-12-29	Oxford/AstraZeneca	3
2	Argentina	2020-12-29	Sinopharm/Beijing	1
3	Argentina	2020-12-29	Sputnik V	20481
4	Argentina	2020-12-30	Moderna	2

```
report = (data)
report
```

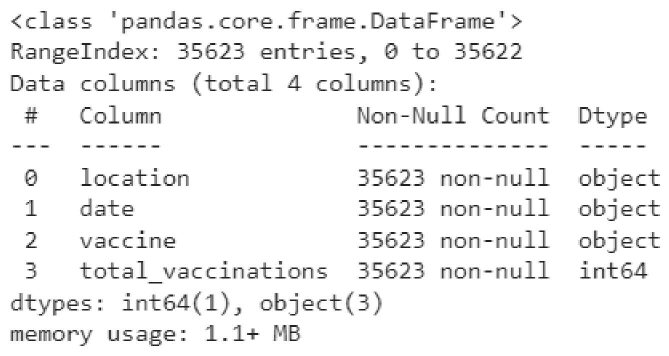
OUTPUT:

	location	date	vaccine	total_vaccinations
0	Argentina	2020-12-29	Moderna	2
1	Argentina	2020-12-29	Oxford/AstraZeneca	3
2	Argentina	2020-12-29	Sinopharm/Beijing	1
3	Argentina	2020-12-29	Sputnik V	20481
4	Argentina	2020-12-30	Moderna	2
...	...	...	...	...
35618	European Union	2022-03-29	Oxford/AstraZeneca	67403106
35619	European Union	2022-03-29	Pfizer/BioNTech	600519998
35620	European Union	2022-03-29	Sinopharm/Beijing	2301516
35621	European Union	2022-03-29	Sinovac	1809
35622	European Union	2022-03-29	Sputnik V	1845103

5623 rows × 4 columns

```
data.info()
```

OUTPUT:



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 35623 entries, 0 to 35622
Data columns (total 4 columns):
 #   Column              Non-Null Count  Dtype 
---  ------              --------------  ----- 
 0   location            35623 non-null  object
 1   date                35623 non-null  object
 2   vaccine             35623 non-null  object
 3   total_vaccinations  35623 non-null  int64 
dtypes: int64(1), object(3)
memory usage: 1.1+ MB
```

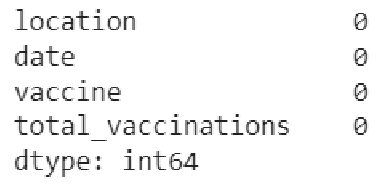
```
data.shape
```

OUTPUT:

```
(35623, 4)
```

```
data.isna().sum()
```

OUTPUT:

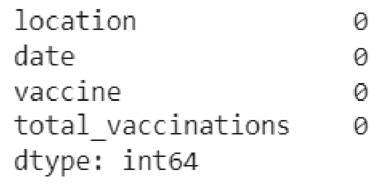


```
location              0
date                  0
vaccine               0
total_vaccinations    0
dtype: int64
```

```
data = data.drop(data[data.total_vaccinations.isna()].index)
```

```
data.isna().sum()
```

OUTPUT:



```
location              0
date                  0
vaccine               0
total_vaccinations    0
dtype: int64
```

```
check_data = data.drop(data[data.total_vaccinations.isna()].index)
```

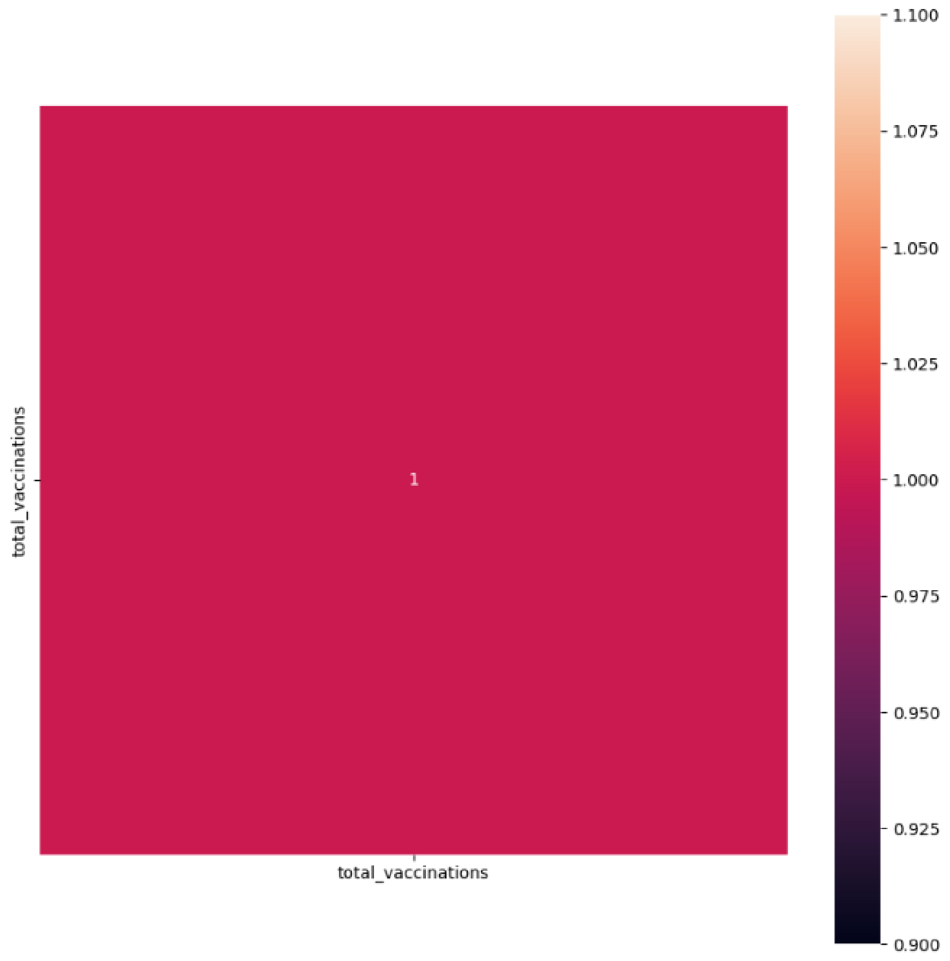
```
check_data.head()
```

OUTPUT:

	location	date	vaccine	total_vaccinations
0	Argentina	2020-12-29	Moderna	2
1	Argentina	2020-12-29	Oxford/AstraZeneca	3
2	Argentina	2020-12-29	Sinopharm/Beijing	1
3	Argentina	2020-12-29	Sputnik V	20481
4	Argentina	2020-12-30	Moderna	2

```
plt.subplots(figsize = (10,10))
sns.heatmap(data.corr(), annot = True, square = True)
plt.show()
```

OUTPUT:



```
diff = check_data.total_vaccinations.mean() -
check_data.total_vaccinations.mean()
```

```
data.total_vaccinations =
data.total_vaccinations.fillna(data.total_vaccinations - diff)
```

```
data.isna().sum()
```

OUTPUT:

```
location              0
date                  0
vaccine               0
total_vaccinations    0
dtype: int64
```

```
data["date"] = pd.to_datetime(data["date"])
data = data.sort_values("date", ascending = True )
data["date"] = data["date"].dt.strftime("%Y-%m-%d")
```

```
unique_dates = data["date"].unique()
```

```
data.head()
```

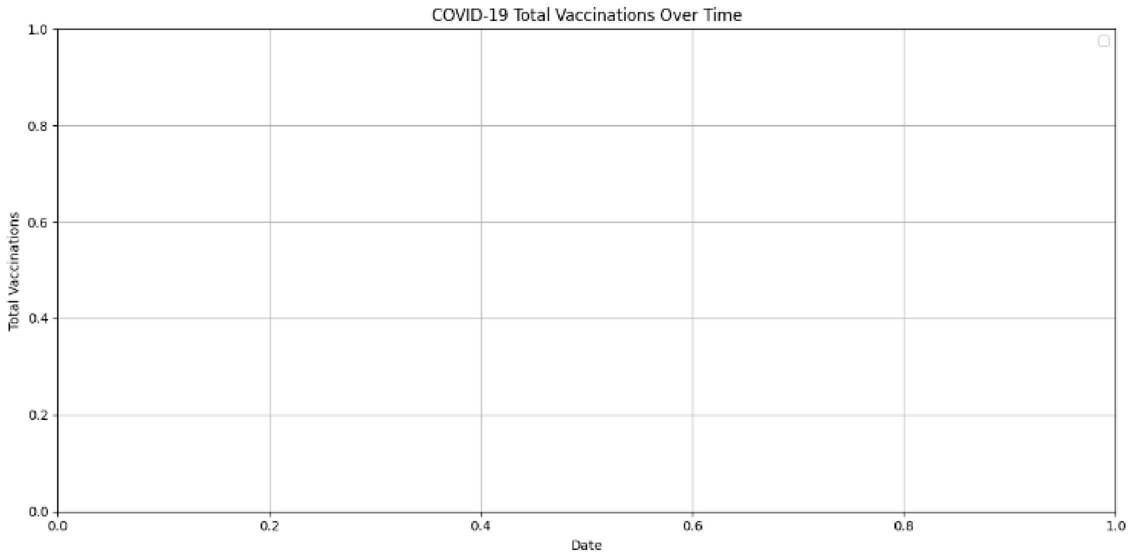
OUTPUT:

	location	date	vaccine	total_vaccinations
16644	Latvia	2020-12-04	Moderna	1
16645	Latvia	2020-12-07	Pfizer/BioNTech	1
16646	Latvia	2020-12-09	Pfizer/BioNTech	2
16647	Latvia	2020-12-15	Pfizer/BioNTech	3
16648	Latvia	2020-12-16	Pfizer/BioNTech	4

```
# Import necessary libraries
import pandas as pd
import matplotlib.pyplot as plt

# Load the dataset
data = pd.read_csv('country_vaccinations_by_manufacturer.csv')
```

```
# Data Preprocessing

# Handling missing values (if any)
data.fillna(0, inplace=True)  # Filling NaN values with 0, you can choose different strategies based on your use case

# Feature engineering (if needed)

# Data Analysis and Visualization
# Plotting total vaccinations over time
plt.figure(figsize=(12, 6))

plt.xlabel('Date')
plt.ylabel('Total Vaccinations')
plt.title('COVID-19 Total Vaccinations Over Time')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```

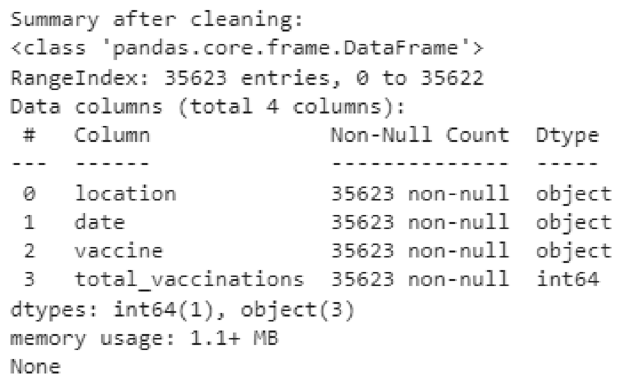
OUTPUT:



```
import pandas as pd

# Load the dataset
data = pd.read_csv('country_vaccinations_by_manufacturer.csv')

# Data Cleaning
# Handling Missing Values
data.dropna(subset=['date', 'vaccine'], inplace=True)
```

```
  # Drop rows where Date or Total Vaccinations is missing

# Handling Duplicates
data.drop_duplicates(inplace=True)  # Drop duplicate rows if any

# Handling Outliers (Example: Removing rows where total vaccinations are
negative)


# Resetting Index
data.reset_index(drop=True, inplace=True)

# Save cleaned data to a new CSV file
data.to_csv('cleaned_covid_vaccine_data.csv', index=False)

# Summary of cleaned data
print("Summary after cleaning:")
print(data.info())
```

OUTPUT:



```
Summary after cleaning:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 35623 entries, 0 to 35622
Data columns (total 4 columns):
 #   Column              Non-Null Count  Dtype 
---  ------              --------------  ----- 
 0   location            35623 non-null  object
 1   date                35623 non-null  object
 2   vaccine             35623 non-null  object
 3   total_vaccinations  35623 non-null  int64 
dtypes: int64(1), object(3)
memory usage: 1.1+ MB
None
```

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from plotly.offline import init_notebook_mode, iplot, plot
import plotly as py
init_notebook_mode(connected=True)
import plotly.graph_objs as go
import plotly.express as px
```

```
def getDf():
    df = pd.read_csv('/country_vaccinations_by_manufacturer.csv')
    return df
```

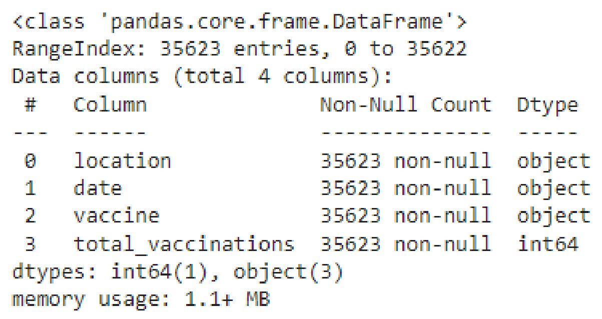
```
df = getDf()

df.columns
```

OUTPUT:

Index(['location', 'date', 'vaccine', 'total_vaccinations'], dtype='object')
CodeText

```
df.info()
```

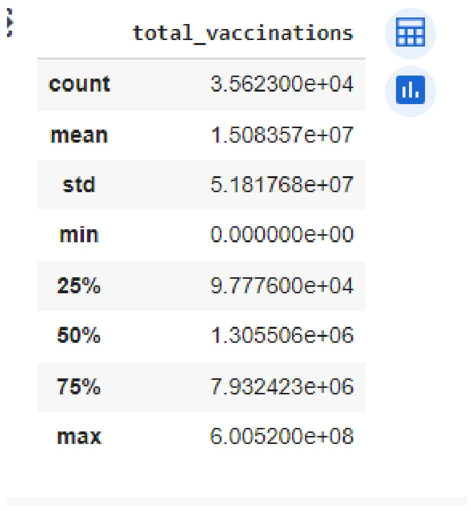
OUTPUT:



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 35623 entries, 0 to 35622
Data columns (total 4 columns):
 #   Column              Non-Null Count  Dtype 
---  ------              --------------  ----- 
 0   location            35623 non-null  object
 1   date                35623 non-null  object
 2   vaccine             35623 non-null  object
 3   total_vaccinations  35623 non-null  int64 
dtypes: int64(1), object(3)
memory usage: 1.1+ MB
```

```
df.describe()
```

OUTPUT:



	total_vaccinations
count	3.562300e+04
mean	1.508357e+07
std	5.181768e+07
min	0.000000e+00
25%	9.777600e+04
50%	1.305506e+06
75%	7.932423e+06
max	6.005200e+08