**COVID-19 VACCINE ANALYSIS**

**Phase 3: Development Part 1**

**Topic**: Start building the Covid-19 Vaccine analysis project by loading and pre-processing the dataset.



**Introduction:**

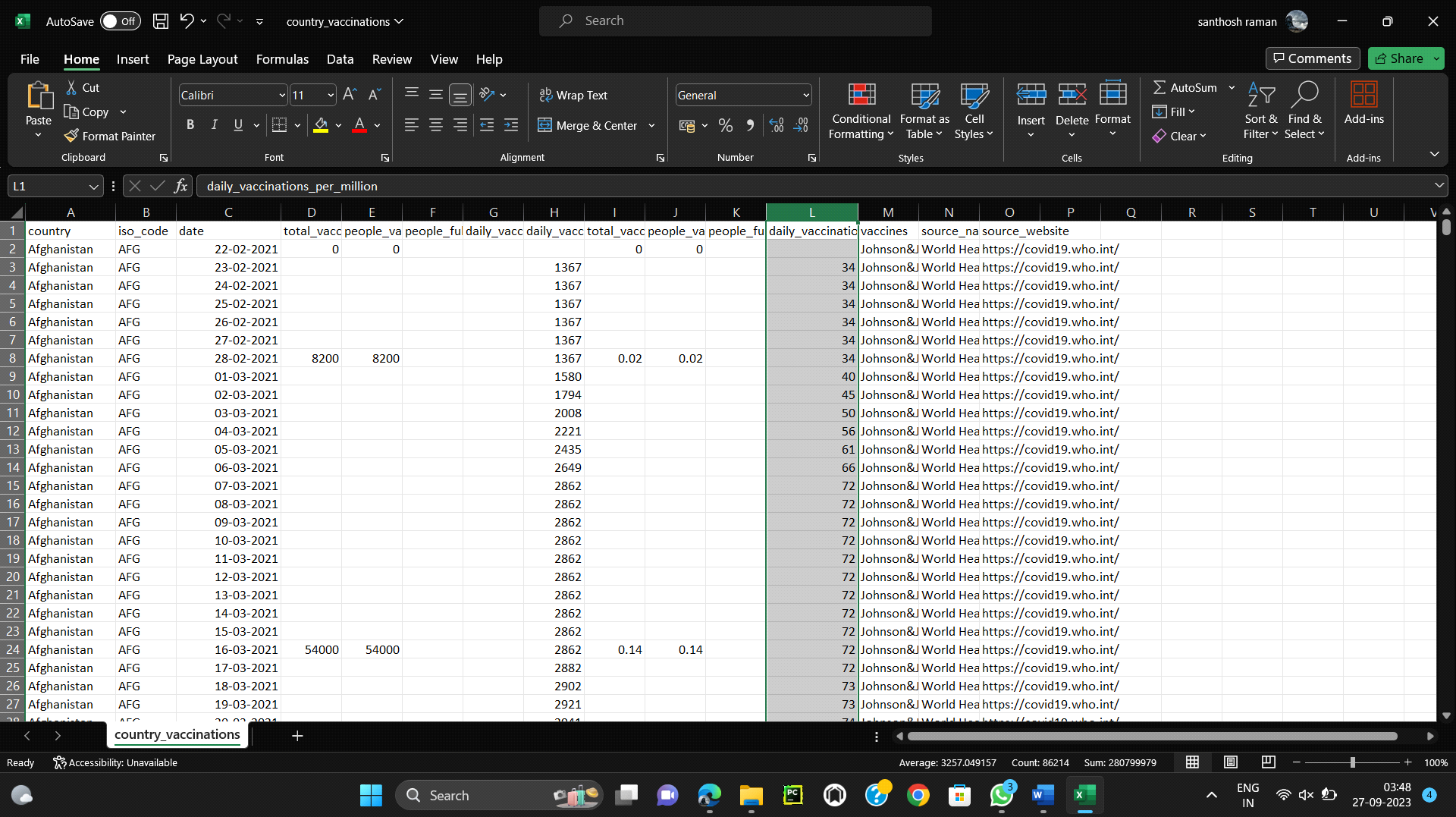
* The analysis of COVID-19 vaccines involves a multifaceted assessment of their development, clinical trials, regulatory approvals, safety and efficacy profiles, distribution, challenges, and broader public health impact.
* Many received Emergency Use Authorization (EUA) initially, allowing for their rapid deployment, later followed by full regulatory approval. Furthermore, the ongoing evolution of the virus necessitates continuous research, development, and adaptation.
* Developing a Covid-19 Vaccine analysis project is a data-driven process that involves harnessing the power of machine learning to analyze the data from the collected data it involves exact analysis about the present, the past and the future. This Journey begins with the fundamental steps of data loading and preprocessing the given dataset.
* This Introduction will guide you through the initial steps of the process. We’ll explore how to import essential libraries, load the Covid-19 dataset, and perform critical preprocessing steps. Data preprocessing is crucial as it helps clean, format, and prepare the data for further analysis. This Includes handling missing values, encoding categorical variables, and clustering that the data is appropriately scaled.

**Dataset:**

The dataset for the given Covid-19 Project can be downloaded from the link given below.

‘’ <https://www.kaggle.com/datasets/gpreda/covid-world-vaccination-progress> ‘’

The given dataset is picturized as below:



**Loading and Preprocessing of dataset:**

Loading and preprocessing the dataset is an important first step in building up any machine learning model. By loading and preprocessing the dataset, we can ensure the machine learning algorithm is able to learn from the data effectively and accurately.

**1.Loading the dataset:**

To load a dataset in Python, we make use of various libraries, such as Pandas, NumPy, and scikit-learn, depending on the dataset's format and our specific requirements. The term "dataset" is quite broad, so the method that we use may vary depending on whether we have a CSV file, Excel file, SQL database, or some other format. Since we are given a csv file, we make use of the read\_csv() method to read the given dataset file.

***PROGRAM:***

Since we are given two datasets, we are going to load both of these datasets separately.

1.Dataset named country\_vaccinations:

import numpy as np

import pandas as pd

country\_vaccinations=pd.read\_csv("country\_vaccinations.csv")

print(country\_vaccinations)

2. Dataset named country\_vaccinatioins\_by\_manufacturer:

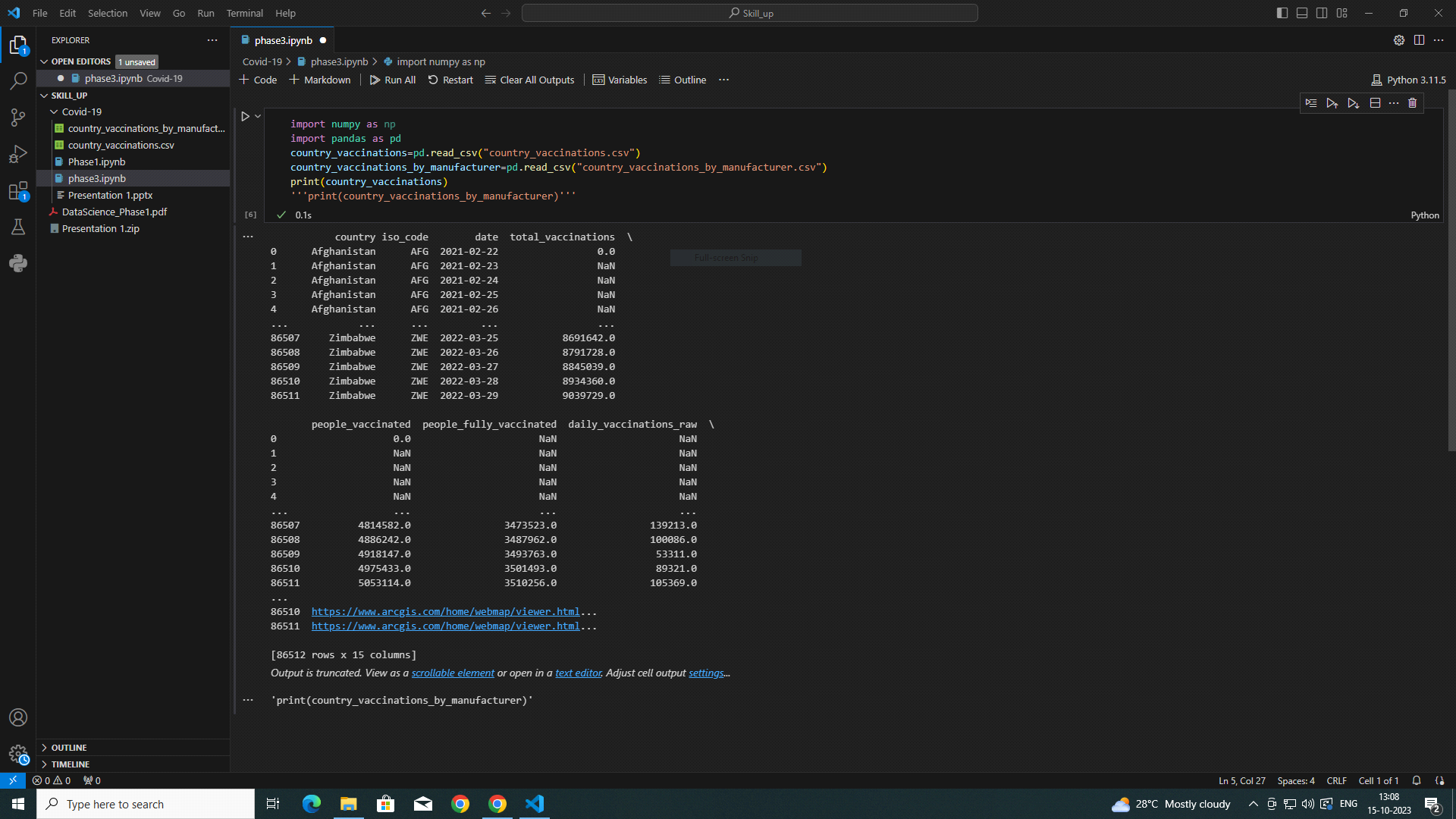
import numpy as np

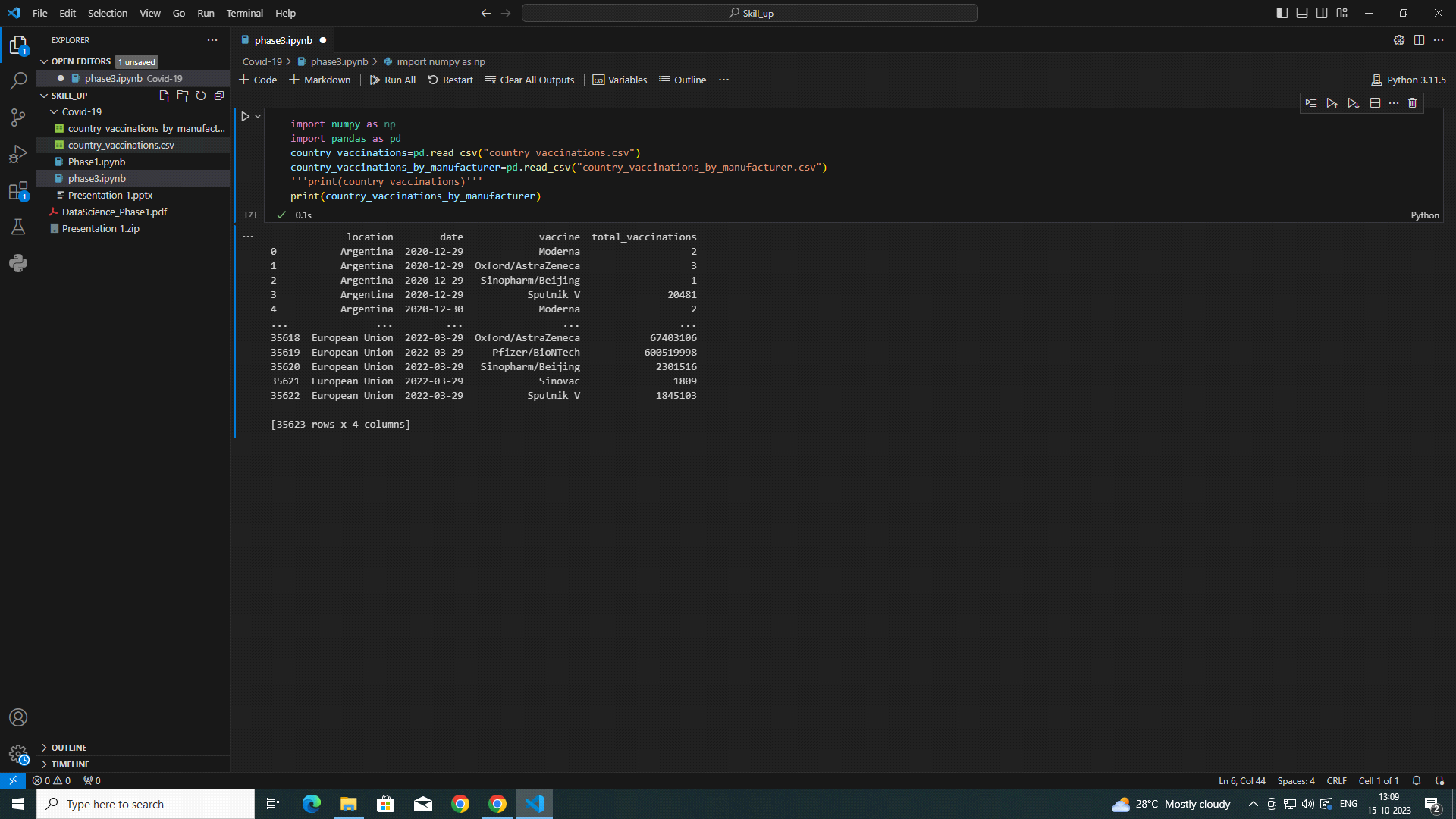
import pandas as pd

country\_vaccinations\_by\_manufacturer=pd.read\_csv("country\_vaccinations\_by\_manufacturer.csv")

print(country\_vaccinations\_by\_manufacturer)

***OUTPUT:***





**2.Preprocessing the dataset:**

* Data preprocessing is the process of cleaning, transforming, and integrating the data in order to make it ready for analysis.
* This may involve removing errors and inconsistencies, handling missing values, transforming the data into a consistent format, and scaling the data to a suitable range.

***2.1Data Cleansing:***

1) Data cleansing for the dataset country\_vaccinations are represented pictorially below with their source code. It Includes removing missing values in the dataset, dropping those missing values.

import pandas as pd

df = pd.read\_csv('country\_vaccinations.csv')

missing\_values = df.isnull().sum()

print("Missing values in the dataset:")

print(missing\_values)

df\_cleaned = df.dropna()

# df\_cleaned = df.fillna(0)

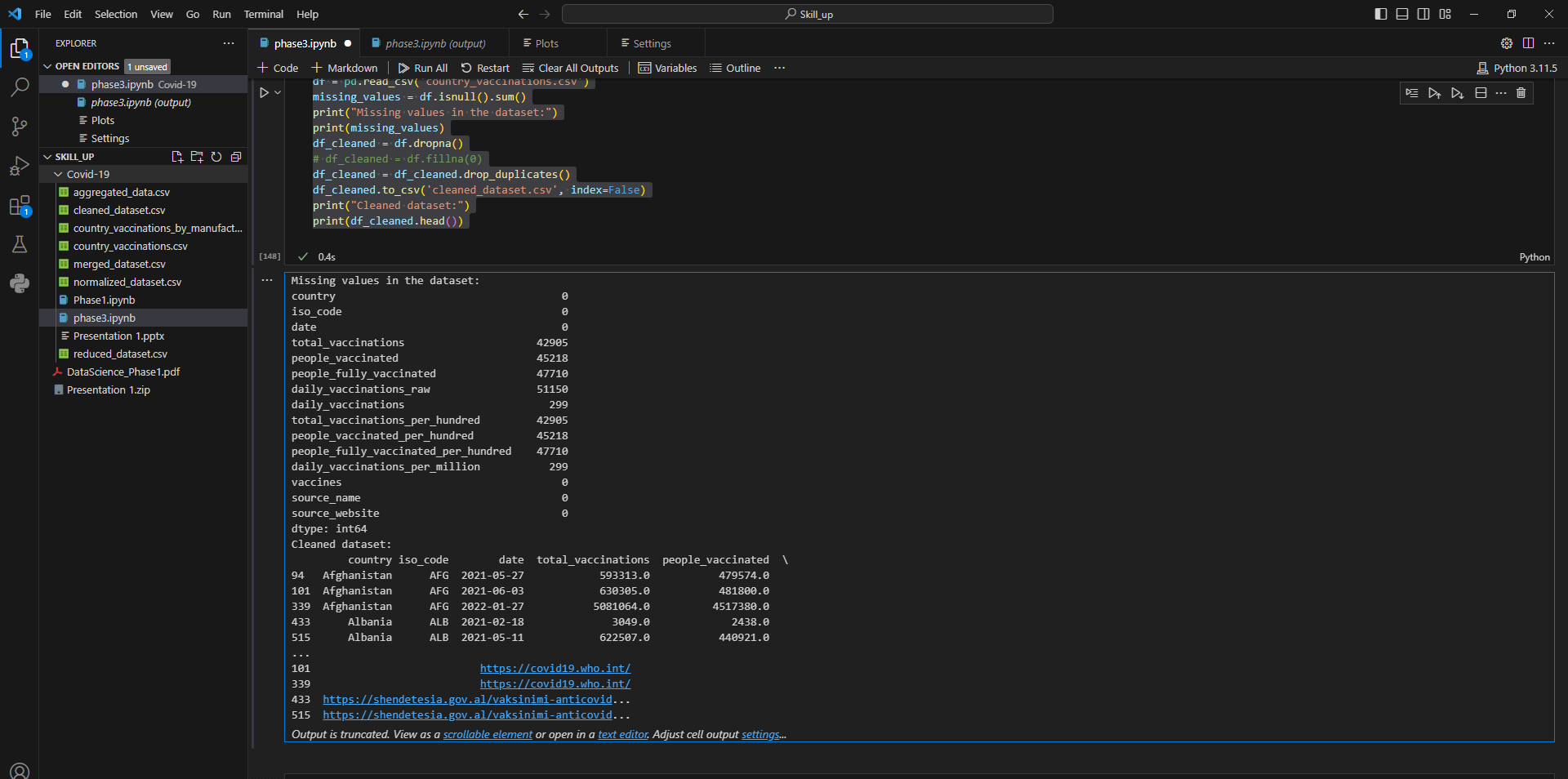
df\_cleaned = df\_cleaned.drop\_duplicates()

df\_cleaned.to\_csv('cleaned\_dataset.csv', index=False)

print("Cleaned dataset:")

print(df\_cleaned.head())

***Output:***



2)Data cleansing for the dataset country\_vaccinations\_by\_manufacture is performed using the following code and represented pictorially below with their output:

import pandas as pd

df = pd.read\_csv('country\_vaccinations\_by\_manufacturer.csv')

missing\_values = df.isnull().sum()

print("Missing values in the dataset:")

print(missing\_values)

df\_cleaned = df.dropna()

# df\_cleaned = df.fillna(0)

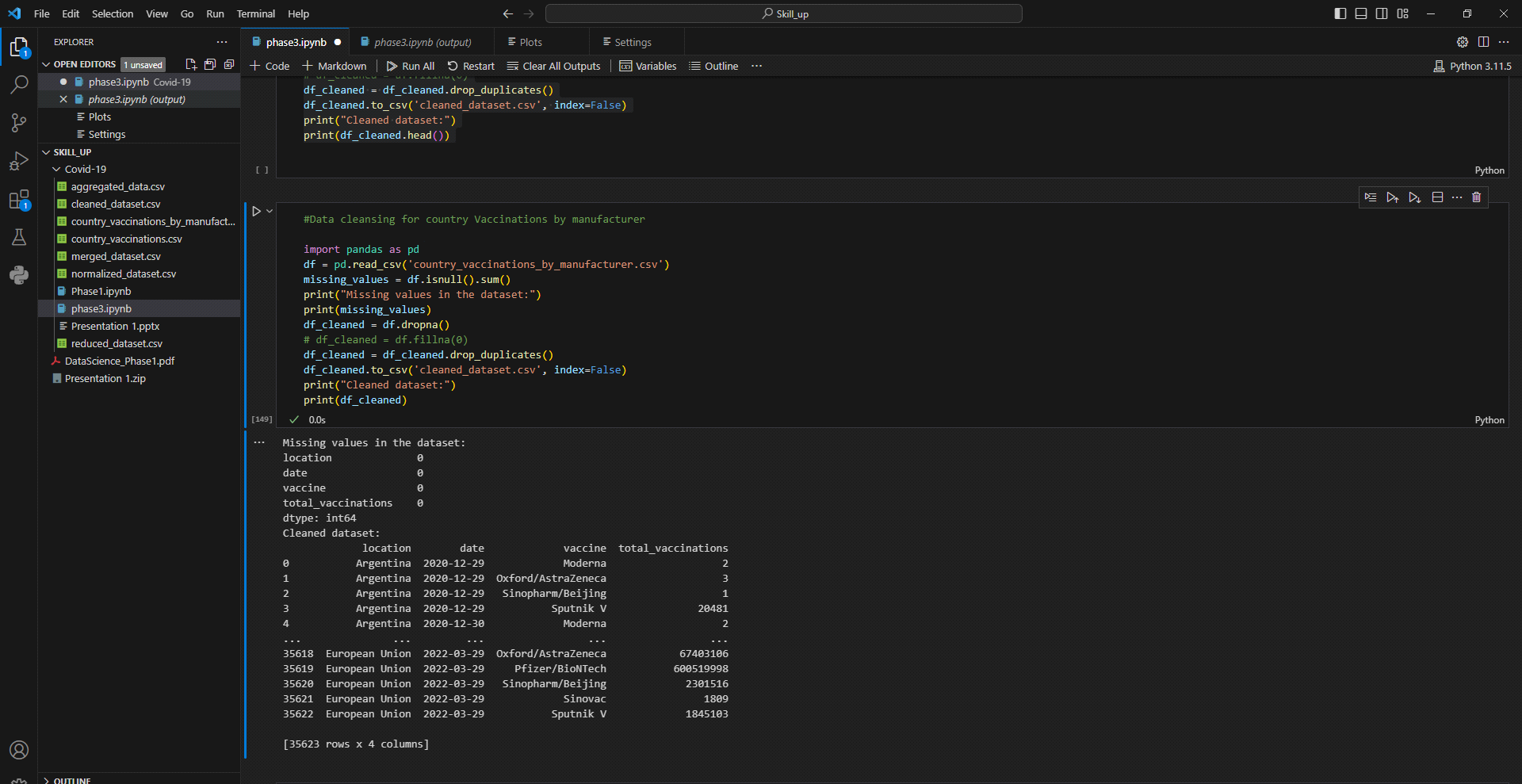
df\_cleaned = df\_cleaned.drop\_duplicates()

df\_cleaned.to\_csv('cleaned\_dataset.csv', index=False)

print("Cleaned dataset:")

print(df\_cleaned)

***OUTPUT:***



***2.2Data Integration:***

Data integration, as a vital component of data preprocessing, involves merging and harmonizing data from diverse sources into a unified data set. Imagine a sample dataset related to COVID-19 vaccinations, where information comes from multiple channels like healthcare providers, government agencies, and research institutions. Each source may have its own data format, terminology, and structure. Data integration in this context involves bringing together these heterogeneous datasets, aligning data fields, and ensuring consistency. It helps in harmonizing the data to form a single, coherent dataset. This integrated dataset can then be used to analyze vaccination trends, vaccine effectiveness, and public health outcomes.

So, we have created an integrated dataset by making use of the two different datasets that we are given. The integrated data sets are created separately and stored as csv files.

The pictorial representation of integrated dataset’s output is given below along with the source code:

***Program:***

import pandas as pd

df1 = pd.read\_csv('country\_vaccinations.csv')

df2 = pd.read\_csv('country\_vaccinations\_by\_manufacturer.csv')

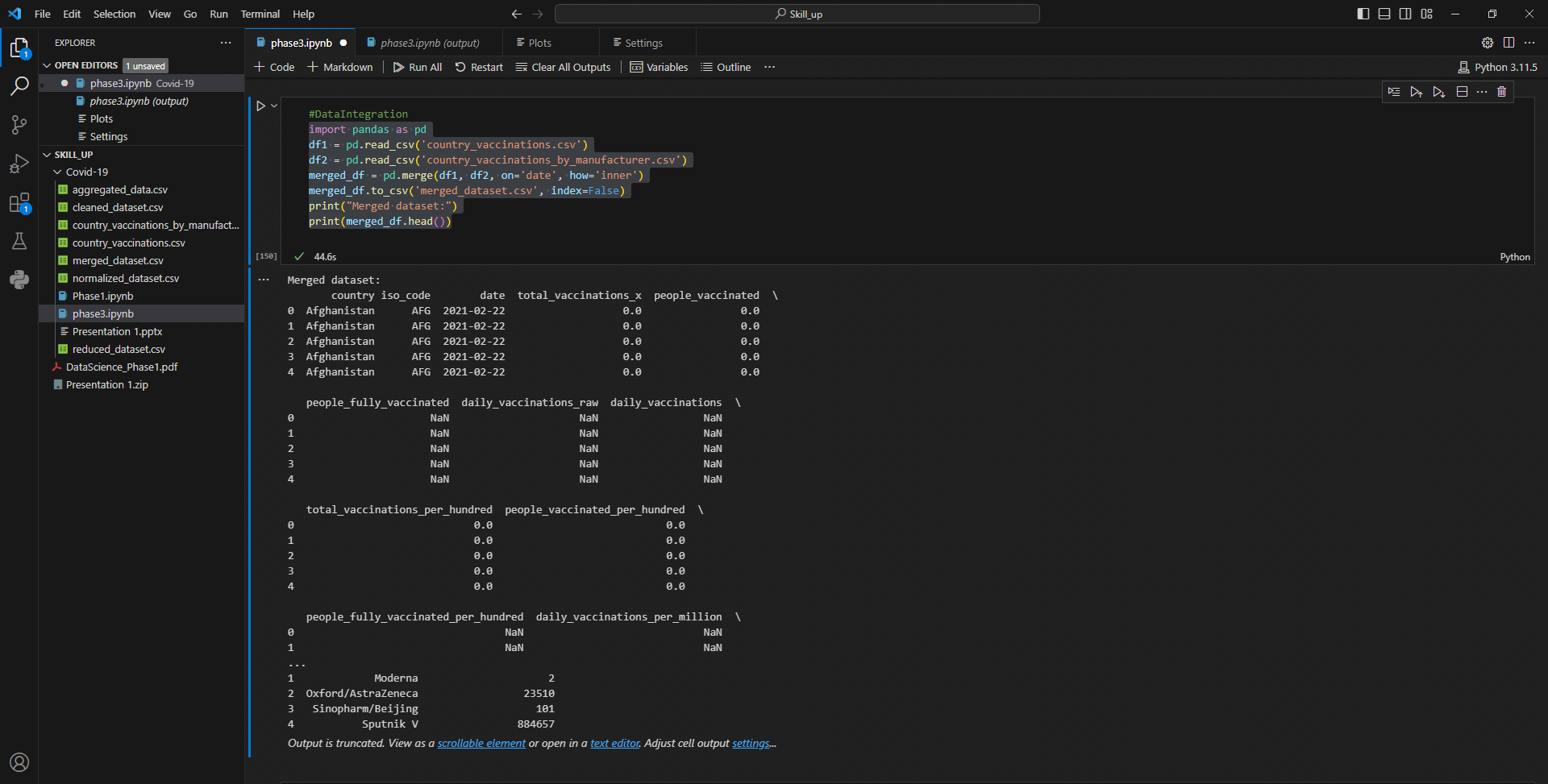
merged\_df = pd.merge(df1, df2, on='date', how='inner')

merged\_df.to\_csv('merged\_dataset.csv', index=False)

print("Merged dataset:")

print(merged\_df.head())

***Output****:*



***2.3 Data Transformation:***

Data transformation, a key step in data preprocessing, involves converting and reshaping data to make it suitable for analysis or modeling. This process may include changing data types, scaling, encoding categorical variables, and aggregating data. In a given dataset, data transformation helps ensure data consistency and prepares it for advanced analytics. For example, you can transform textual descriptions into numerical features, normalize numerical values, and aggregate data for summary statistics. Data transformation is essential for extracting meaningful insights and patterns from the data, making it a fundamental aspect of data preprocessing.

Data Transformation includes various subprocess which are listed as below:

* Changing Data Types
* Encoding Categorical Variables
* Normalizing data in dataset
* Aggregating data

***2.3.1Changing Data Types:***

Changing data types, as part of data transformation, involves converting the type of data in a dataset from one format to another. This process is essential when dealing with a dataset that contains data in formats that are not suitable for the analysis or tasks you want to perform. Changing data types ensures that the dataset's content is in the right format for analysis and modeling, preventing issues like data type incompatibility and enabling more effective data processing.

Here in the data set country\_vaccinations\_by\_manufacturer we have changed the data type of the column total vaccination into ‘object’ from 'int64’ by making use of the method astype() suffixing the column name before it.

***Program:***

import pandas as pd

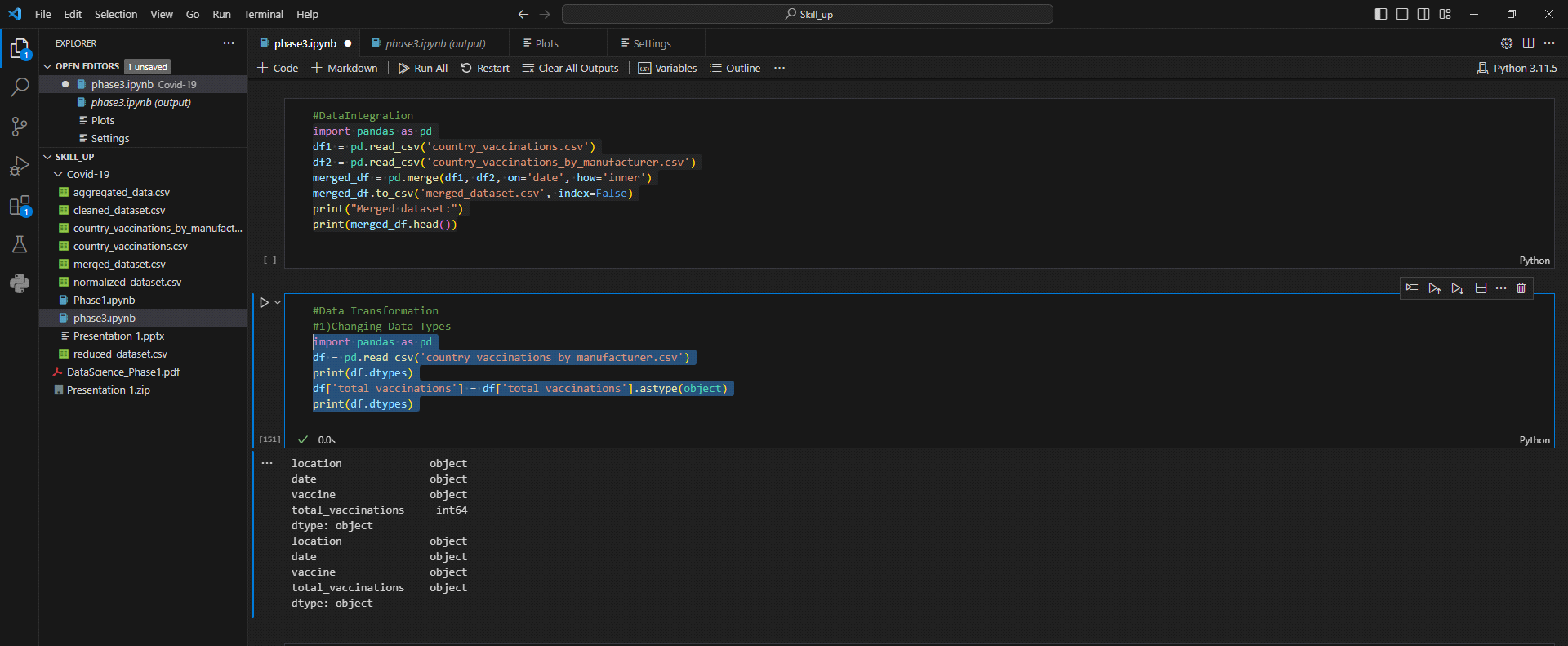
df = pd.read\_csv('country\_vaccinations\_by\_manufacturer.csv')

print(df.dtypes)

df['total\_vaccinations'] = df['total\_vaccinations'].astype(object)

print(df.dtypes)

***Output:***



***2.3.2 Encoding Categorical Variables:***

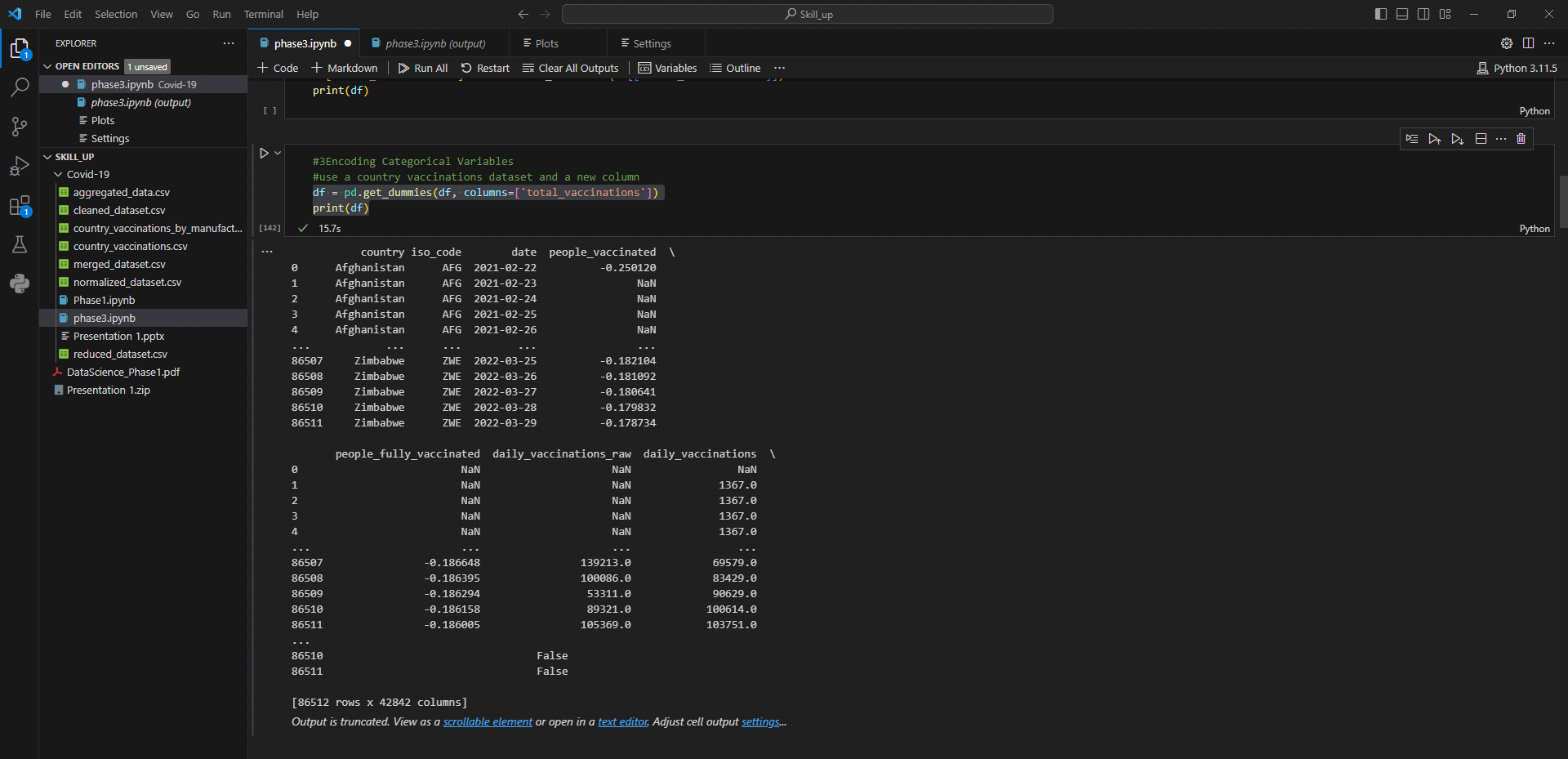
Encoding categorical variables is a crucial step in data transformation, particularly when working with machine learning models or data analysis. Categorical variables represent categories or labels rather than a numeric values. To make these variables compatible with most machine learning algorithms, you need to encode them into numerical form.

***Program:***

df = pd.get\_dummies(df, columns=['total\_vaccinations'])

print(df)

***Output:***



***2.3.3 Normalizing data in dataset:***

Normalizing data is an essential data transformation technique used to scale numerical features within a dataset to a common range. The goal is to make different features or variables comparable, eliminate the influence of differing units or scales, and ensure that no single feature dominates the analysis. Normalization typically scales the data to a range between 0 and 1, but it can also involve other scales depending on the specific needs of the analysis.

We have normalized the dataset country and vaccinations, and its source code and output are given as follows:

***Program:***

import pandas as pd

from sklearn.preprocessing import StandardScaler

df = pd.read\_csv('country\_vaccinations.csv')

columns\_to\_normalize=['total\_vaccinations','people\_vaccinated','people\_fully\_vaccinated']

scaler = StandardScaler()

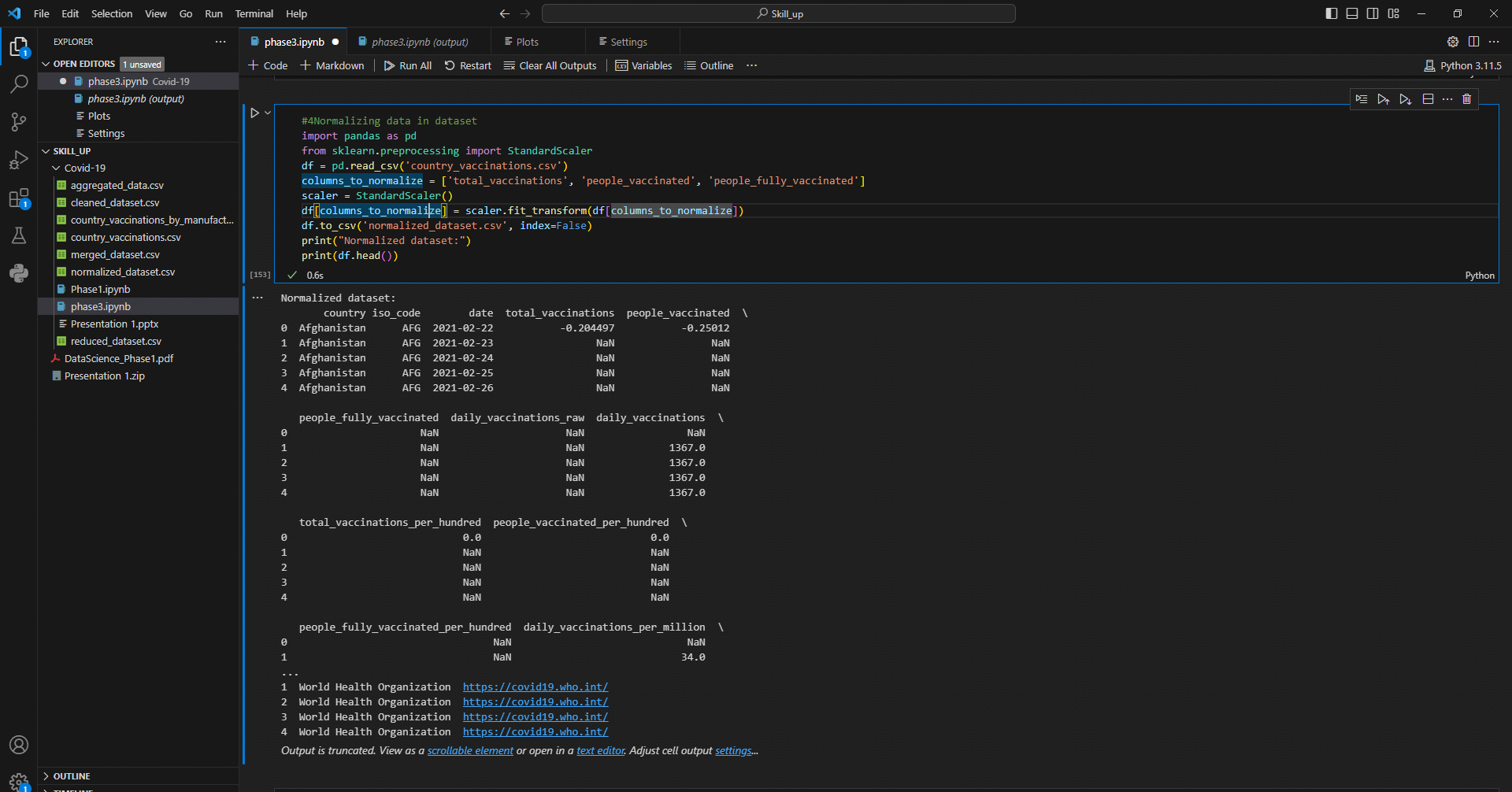
df[columns\_to\_normalize] = scaler.fit\_transform(df[columns\_to\_normalize])

df.to\_csv('normalized\_dataset.csv', index=False)

print("Normalized dataset:")

print(df.head())

***Output:***



***2.3.4 Aggregating data:***

Aggregating data in a dataset, as part of data transformation, involves the process of summarizing, grouping, or reducing data to obtain a more concise and understandable representation of the information. It is especially useful when dealing with large datasets or when you want to analyze data at a higher level of granularity. Aggregation often involves applying functions like sum, count, mean, median, or other statistical functions to groups of data points based on one or more grouping criteria.

***Program:***

import pandas as pd

df = pd.read\_csv('country\_vaccinations\_by\_manufacturer.csv')

grouped\_data = df.groupby('date')

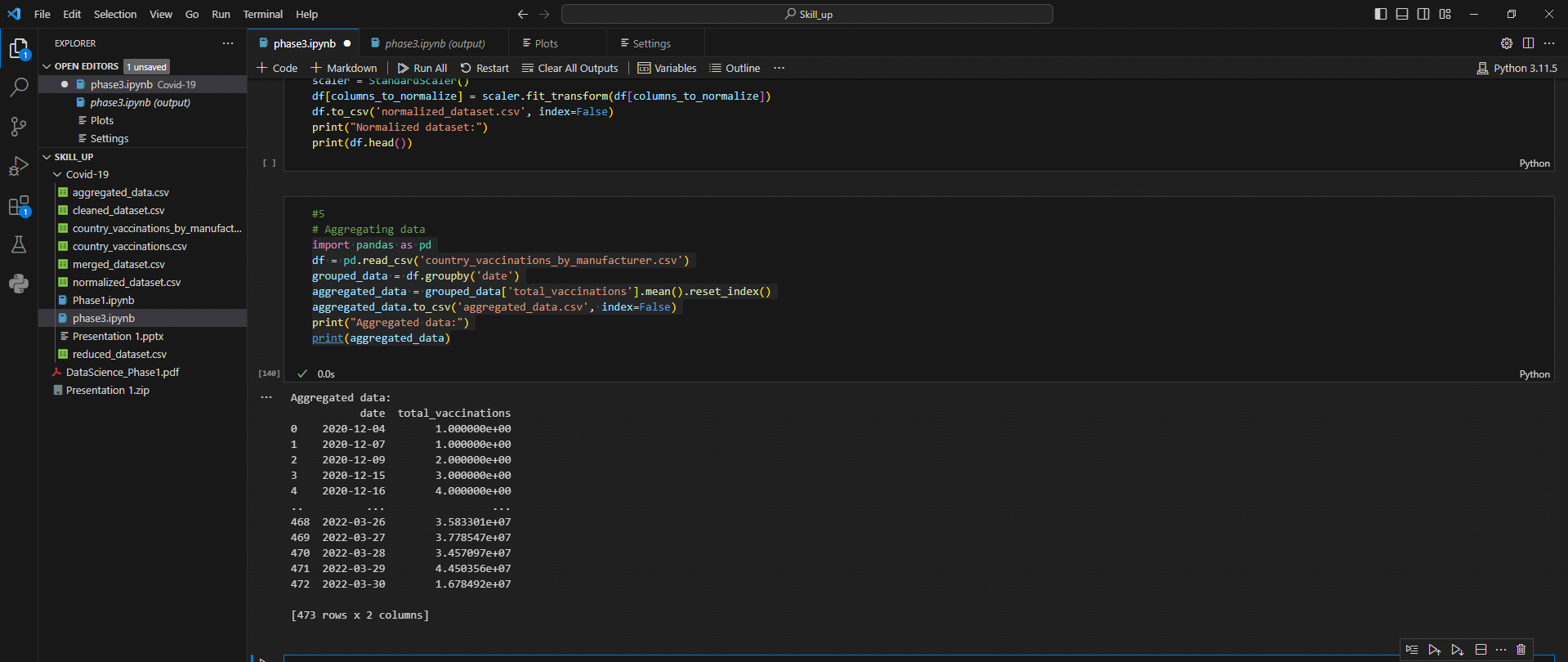
aggregated\_data = grouped\_data['total\_vaccinations'].mean().reset\_index()

aggregated\_data.to\_csv('aggregated\_data.csv', index=False)

print("Aggregated data:")

print(aggregated\_data)

***Output:***



***2.4 Data Reduction:***

Data reduction, in the context of data preprocessing, refers to the process of reducing the volume but producing the same or similar analytical results. It involves techniques and methods to simplify and condense a dataset while retaining the essential information. Data reduction is often necessary for various reasons, including improving the efficiency of data processing, reducing storage requirements, and mitigating the curse of dimensionality in machine learning. Data Reduction simply reduces the data by reducing their size.

***Program:***

import pandas as pd

from sklearn.decomposition import PCA

df = pd.read\_csv('country\_vaccinations.csv')

df=df.dropna()

selected\_features = df[['people\_vaccinated', 'people\_fully\_vaccinated']]

n\_components = 2

pca = PCA(n\_components=n\_components)

reduced\_data = pca.fit\_transform(selected\_features)

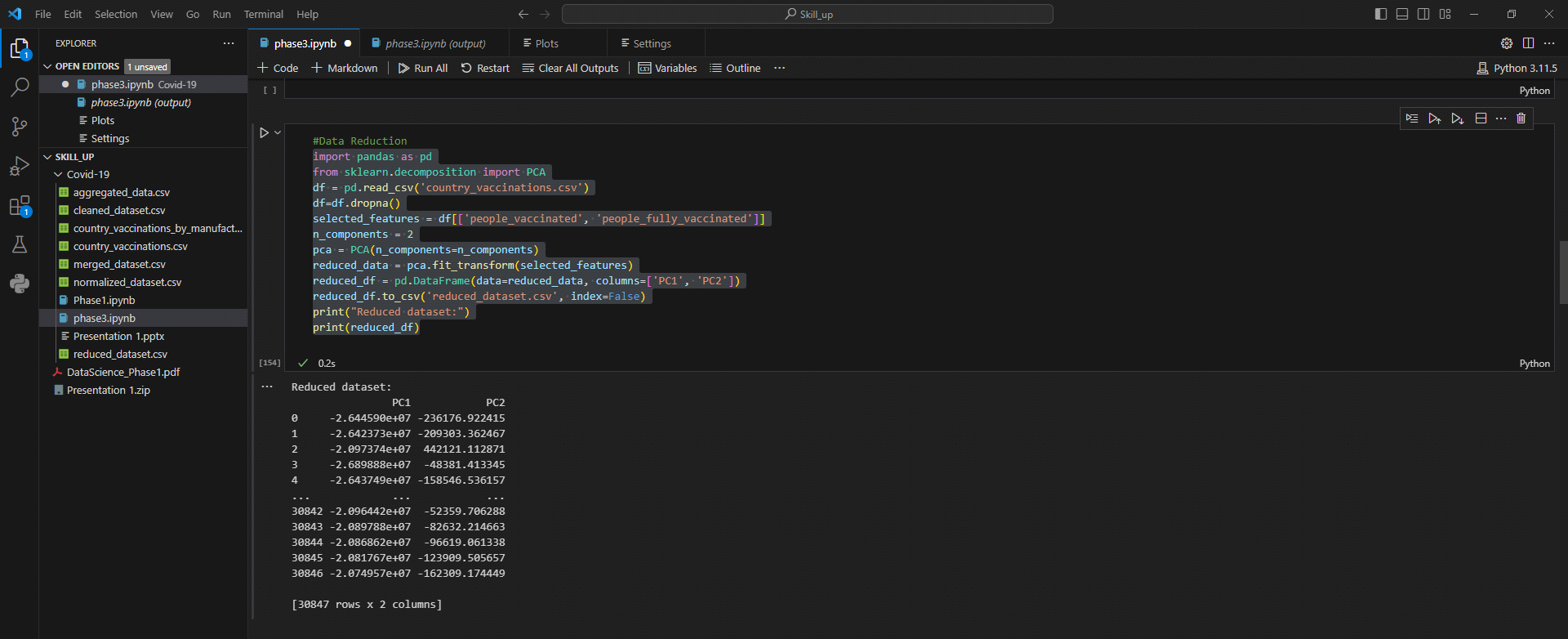
reduced\_df = pd.DataFrame(data=reduced\_data, columns=['PC1', 'PC2'])

reduced\_df.to\_csv('reduced\_dataset.csv', index=False)

print("Reduced dataset:")

print(reduced\_df)

***Output:***



***Visualizing The preprocessed datasets:***

Visualizing preprocessed datasets is an essential step in exploratory data analysis (EDA) and data-driven decision-making. Once you've cleaned and transformed your data, creating visualizations can help you understand the data, identify patterns, and communicate insights effectively.

* So as a very first part of visualization we have visualized the merged dataset of both country\_vaccinations and country\_vaccinations\_by\_manufacturer by means of histogram.

***Program:***

import numpy as np

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

df=pd.read\_csv('merged\_dataset.csv')

plt.hist(df['total\_vaccinations\_per\_hundred'], bins=20, color='skyblue')

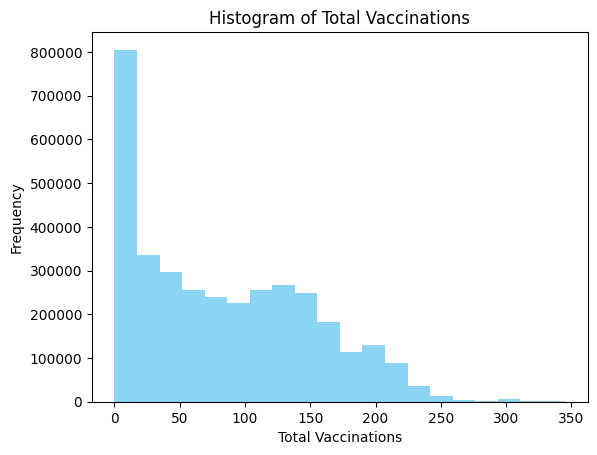
plt.title('Histogram of Total Vaccinations')

plt.xlabel('Total Vaccinations')

plt.ylabel('Frequency')

plt.show()

***Output:***



* Following this is the Joint plot representation of the reduced dataset that we have created during data reduction.

***Program:***

import numpy as np

import pandas as pd

import seaborn as sns

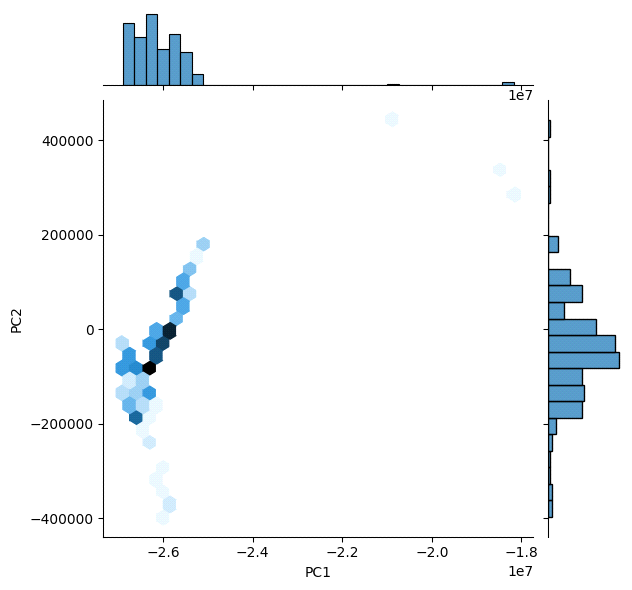
import matplotlib.pyplot as plt

df=pd.read\_csv('reduced\_dataset.csv')

df=df.head(200)

print(sns.jointplot(df,x='PC1',y='PC2',kind='hex'))

***Output:***



* Thirdly we have made use of the pair plot visualizing technique to visualize the reduced data sets.

***Program:***

import numpy as np

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

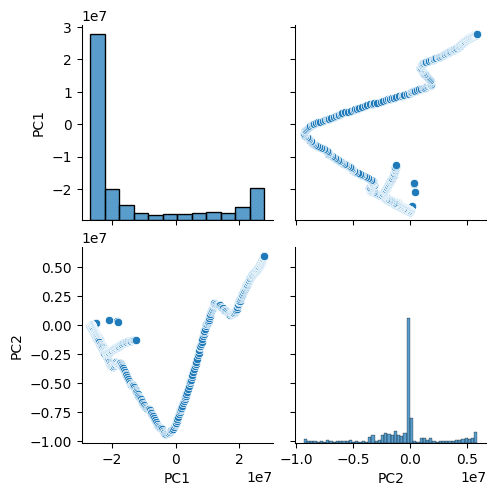
df=pd.read\_csv('reduced\_dataset.csv')

df=df.head(1000)

plt.figure(figsize=(12,8))

sns.pairplot(df)

***Output:***



* Following these pair plots we have made use of the merged dataset to visualize it in histogram as it contains a combination of different columns and rows.

***Program:***

import numpy as np

import pandas as pd

df=pd.read\_csv('merged\_dataset.csv')

print(df.hist(figsize=(100,80)))

***Output:***



* Being last we have calculated the correlations for the country\_vaccinations dataset and by making use of it we even have plotted a heatmap for it which is visualized below.

***Program:***

import numpy as np

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

df=pd.read\_csv('country\_vaccinations.csv')

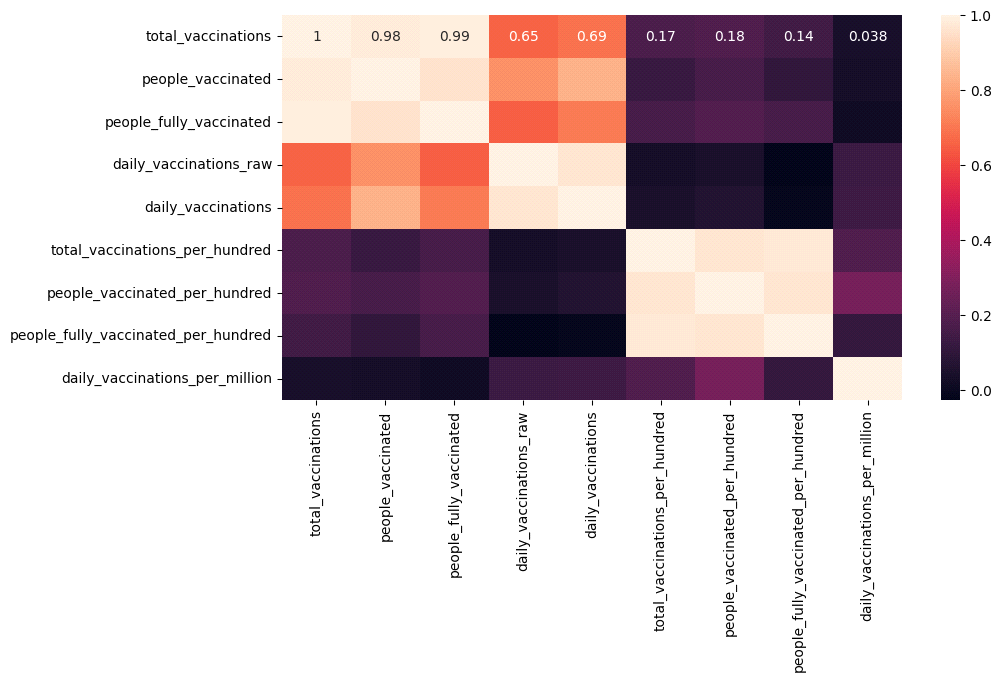
df.corr(numeric\_only=True)

print(df)

plt.figure(figsize=(10,5))

print(sns.heatmap(df.corr(numeric\_only=True),annot=True))

***Output:***



***Conclusion:***

* In the quest of Covid-19 vaccine Analysis, we have embarked on a crucial journey that begins with loading and preprocessing the dataset. We have traversed through essential steps, starting with importing the necessary libraries to facilitate data manipulation and analysis.
* Understanding the data's structure, characteristics, and any potential issues exploratory data analysis (EDA)is essential for Informed decision-making.
* Data Preprocessing emerged as a pivotal aspect of this process.it involves cleaning, transforming, and refining the dataset to ensure that it aligns with the requirement of machine learning algorithms.
* With these foundational steps, completed, our dataset is now primed for the subsequent stages of building and training a Covid-19 vaccine analysis model.