PHASE 3

COVID 19 VACCINE ANALYSIS

PHASE 3 OVERVIEW:

In this phase, We are aiming to prepare our data set for further analysis phases of the project.

DATA PREPARATION

- **Data Pre-processing**
- **Data visualization**

For Data Pre-Processing, we have documented the most crucial area of work for our project type that is data preparation which will help in ready extraction and quick access to the precise data set.

For Data Visualization, We have planned to make all the required visual models for various accepts of Covid-19 vaccination progress in the form of various graphical and pictorial representation such as bar chart, line graph, scatter plot and pie chart.

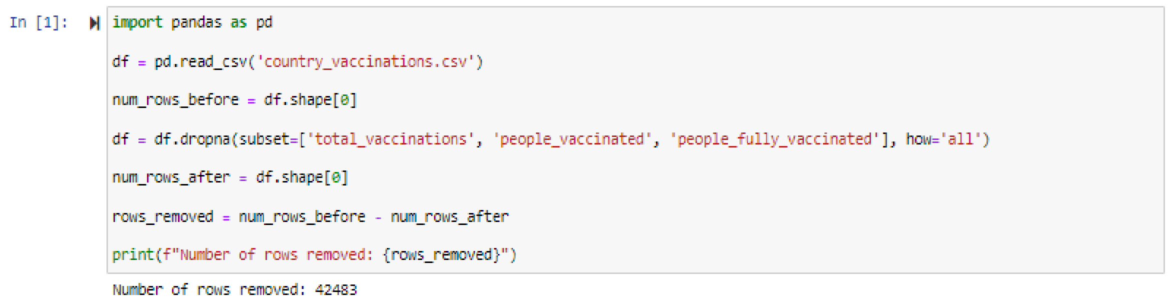
DATA PRE-PROCESSING:

Data preprocessing is an important step in the data science process. It refers to the cleaning, transforming, and integrating of data in order to make it ready for analysis.

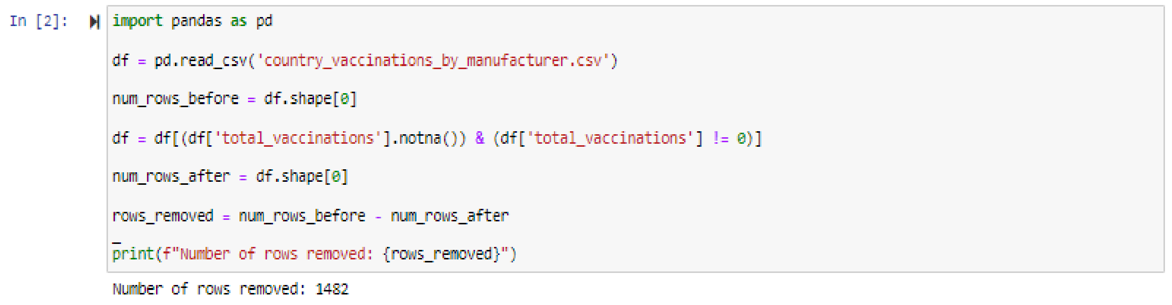
The goal of data preprocessing is to improve the quality of the data and to make it more suitable for the specific data science analysis task.

Data Cleaning:
- Handle missing values.
- Validate data for anomalies, outliers, and errors.
- Correct any inconsistent or erroneous entries.

  In our project, we utilized this data cleaning process involved removing rows where critical vaccination information was entirely absent - such as 'total_vaccinations', 'people_vaccinated', and 'people_fully_vaccinated'. The number of removed rows was then calculated, ensuring that our analysis was based on complete and reliable data.



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

         df = pd.read_csv('country_vaccinations.csv')

         num_rows_before = df.shape[0]

         df = df.dropna(subset=['total_vaccinations', 'people_vaccinated', 'people_fully_vaccinated'], how='all')

         num_rows_after = df.shape[0]

         rows_removed = num_rows_before - num_rows_after

         print(f"Number of rows removed: {rows_removed}")

         Number of rows removed: 42483
```

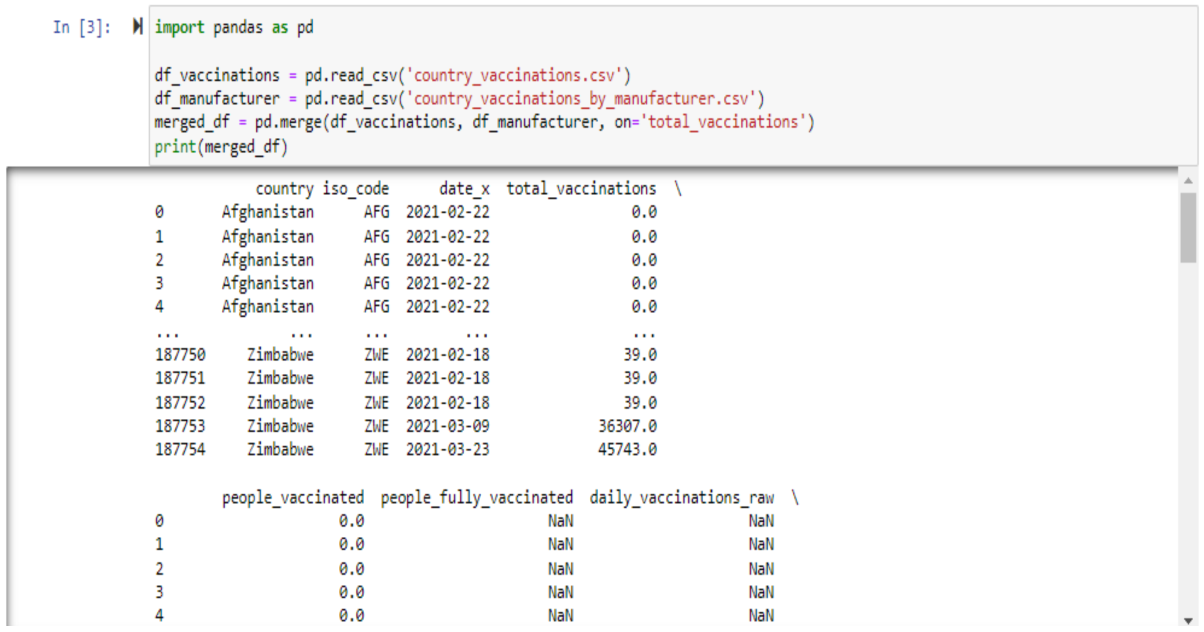


```
In [2]:  import pandas as pd

         df = pd.read_csv('country_vaccinations_by_manufacturer.csv')

         num_rows_before = df.shape[0]

         df = df[(df['total_vaccinations'].notna()) & (df['total_vaccinations'] != 0)]

         num_rows_after = df.shape[0]

         rows_removed = num_rows_before - num_rows_after

         print(f"Number of rows removed: {rows_removed}")

         Number of rows removed: 1482
```

<u>Data Integration</u>:

- Merge or concatenate multiple datasets if applicable, based on common identifiers.

By using the integration technique ,we merged two datasets—vaccination records and vaccine manufacturer information—based on the common field 'total_vaccinations'. This integration allows us to analyze vaccination data alongside details about the manufacturers involved.



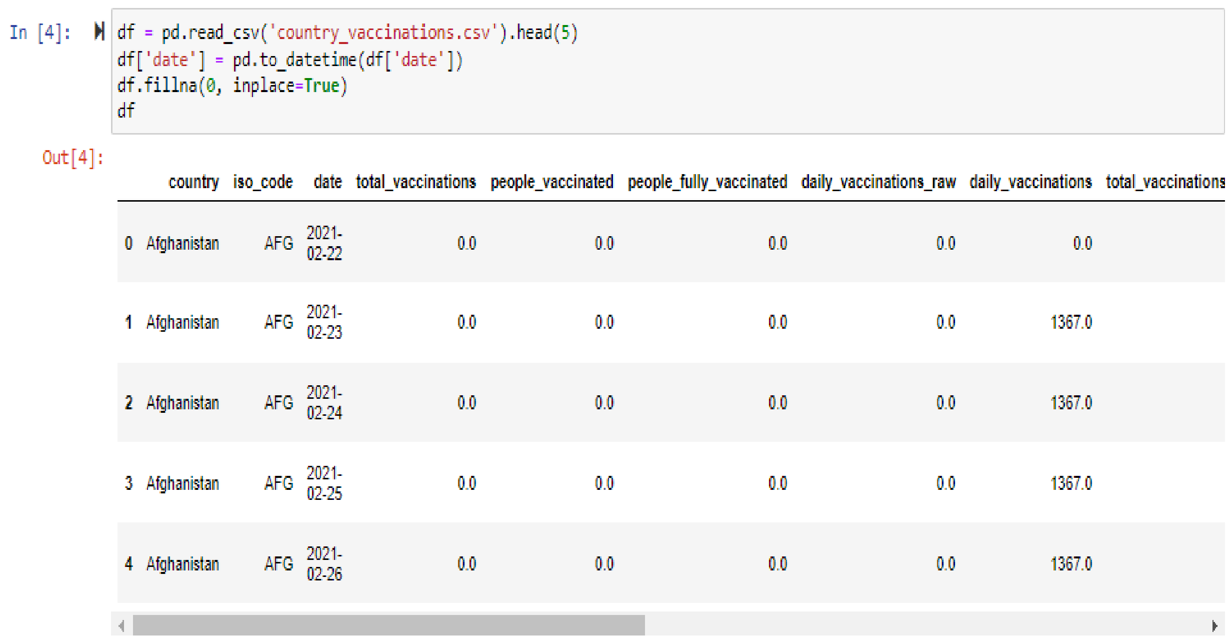
```
In [3]:  import pandas as pd

         df_vaccinations = pd.read_csv('country_vaccinations.csv')
         df_manufacturer = pd.read_csv('country_vaccinations_by_manufacturer.csv')
         merged_df = pd.merge(df_vaccinations, df_manufacturer, on='total_vaccinations')
         print(merged_df)
```

```
             country iso_code      date_x  total_vaccinations  \
0        Afghanistan      AFG  2021-02-22                 0.0
1        Afghanistan      AFG  2021-02-22                 0.0
2        Afghanistan      AFG  2021-02-22                 0.0
3        Afghanistan      AFG  2021-02-22                 0.0
4        Afghanistan      AFG  2021-02-22                 0.0
...              ...      ...         ...                 ...
187750      Zimbabwe      ZWE  2021-02-18                39.0
187751      Zimbabwe      ZWE  2021-02-18                39.0
187752      Zimbabwe      ZWE  2021-02-18                39.0
187753      Zimbabwe      ZWE  2021-03-09             36307.0
187754      Zimbabwe      ZWE  2021-03-23             45743.0

        people_vaccinated  people_fully_vaccinated  daily_vaccinations_raw  \
0                     0.0                      NaN                     NaN
1                     0.0                      NaN                     NaN
2                     0.0                      NaN                     NaN
3                     0.0                      NaN                     NaN
4                     0.0                      NaN                     NaN
```

Data Transformation:

- Normalize or standardize numerical features.
- Encode categorical variables.
- Handle date and time data.

We selected the first five rows of the dataset using the **.head(5)** method to get a quick overview of the data. We replaced any missing values in the DataFrame with zeros using **df.fillna(0, inplace=True)**. This ensures that missing data doesn't affect our analysis.



```
In [4]:  df = pd.read_csv('country_vaccinations.csv').head(5)
         df['date'] = pd.to_datetime(df['date'])
         df.fillna(0, inplace=True)
         df
```

Out[4]:

	country	iso_code	date	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_vaccinations_raw	daily_vaccinations	total_vaccinations
0	Afghanistan	AFG	2021-02-22	0.0	0.0	0.0	0.0	0.0	
1	Afghanistan	AFG	2021-02-23	0.0	0.0	0.0	0.0	1367.0	
2	Afghanistan	AFG	2021-02-24	0.0	0.0	0.0	0.0	1367.0	
3	Afghanistan	AFG	2021-02-25	0.0	0.0	0.0	0.0	1367.0	
4	Afghanistan	AFG	2021-02-26	0.0	0.0	0.0	0.0	1367.0	

DATA VISUALISATION:

Data visualization is transforming data or information into graphics to make it easier for the human brain to comprehend and get insights.

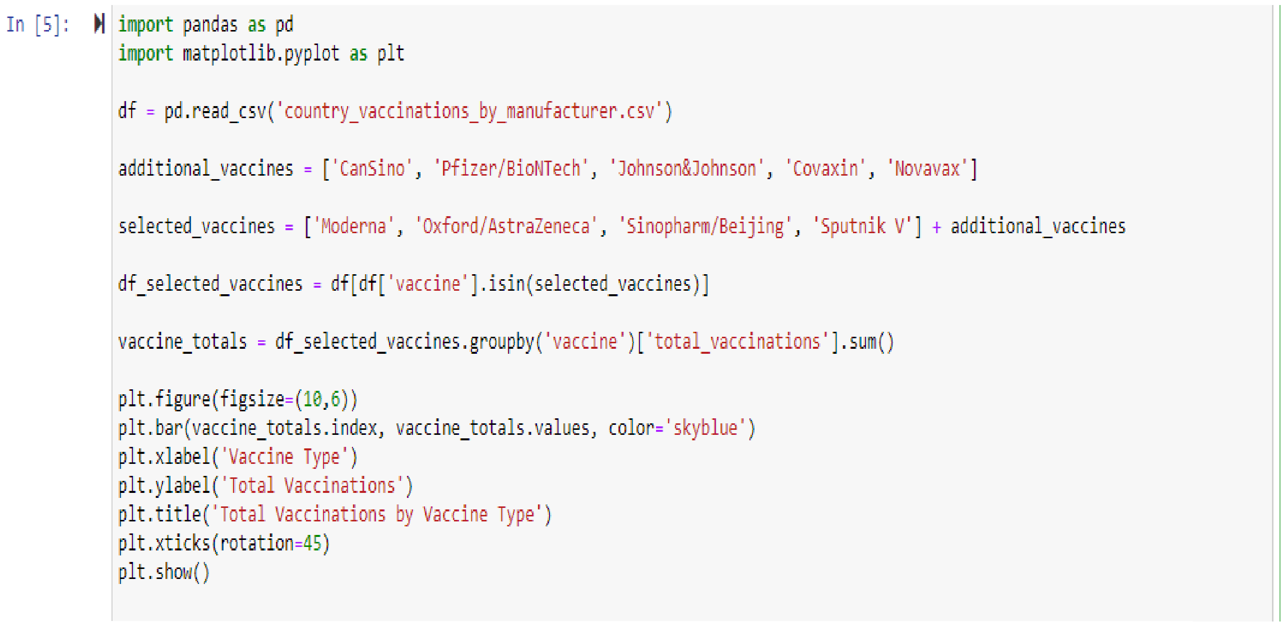
The purpose of data visualization projects is to identify patterns, trends, and anomalies or deviations in large datasets/big data (the main data for visualization projects); that otherwise would have been impossible. This is the final step of the data science process and data presentation architecture (DPA) .

The raw dataset for "Covid -19 Vaccine Analysis" provided has been extracted from kaggle, Loaded into Anaconda Jupyter notebook pre-processed and prepared using python(Pandas) for visualisation.

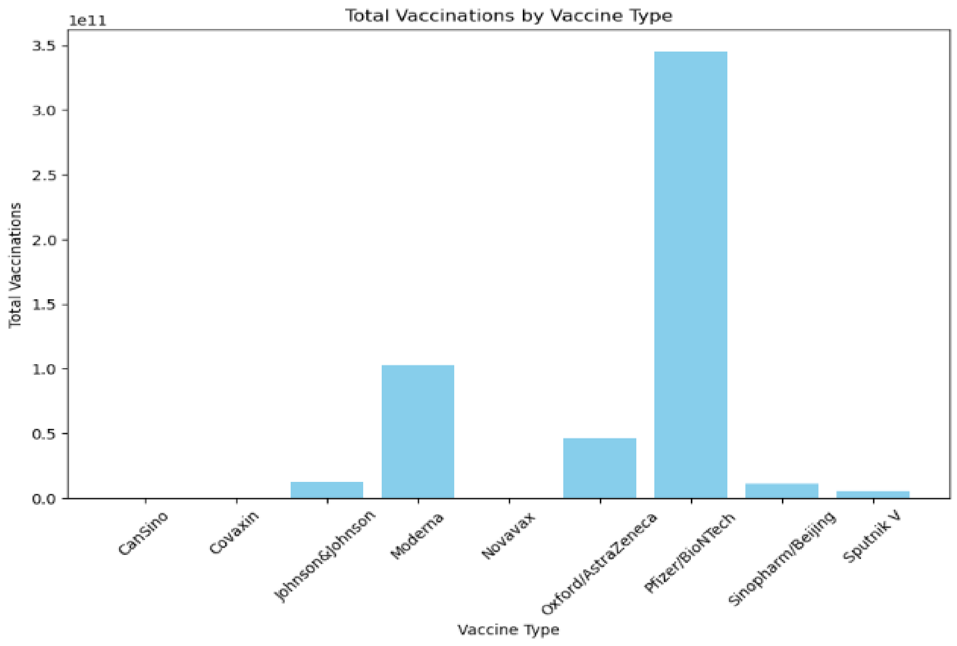
Bar graph:

A bar graph (or bar chart) is a graphical representation of data in which rectangular bars are used to compare the values of different categories**.**

We loaded a dataset containing manufacturer-specific vaccination information. After specifying key vaccines of interest, we filtered and summed the total vaccinations for each. This bar graph visually represents the total vaccinations for selected vaccine types, aiding in a clear comparison of their distribution.



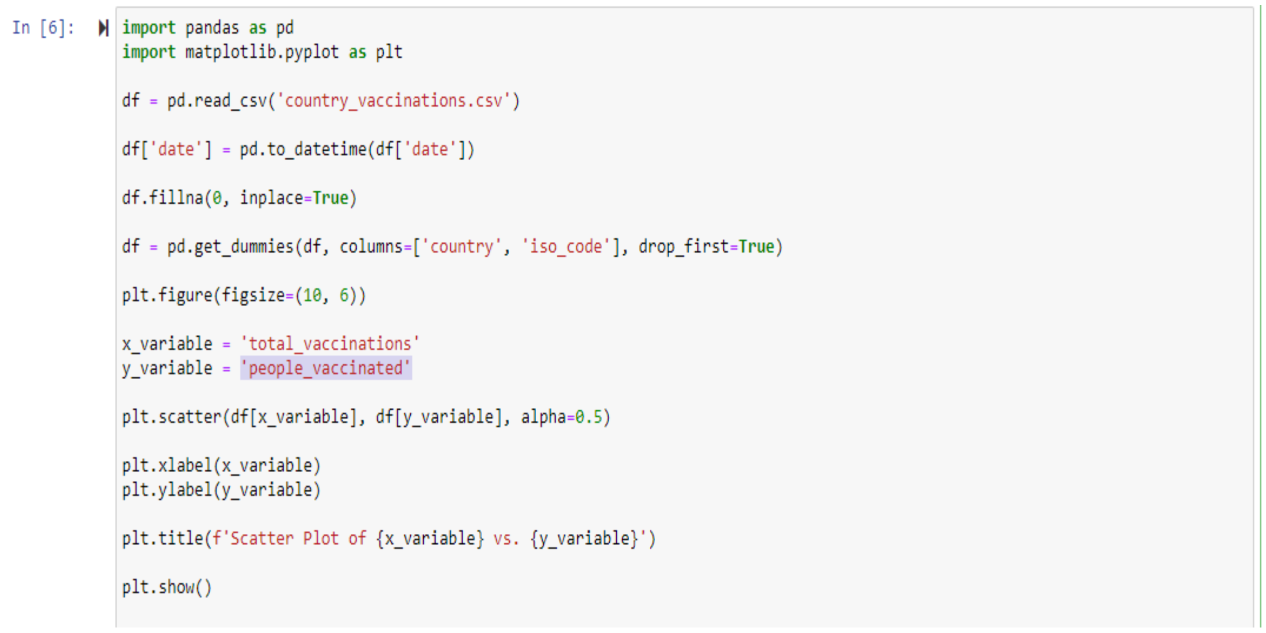
```
In [5]:  import pandas as pd
         import matplotlib.pyplot as plt

         df = pd.read_csv('country_vaccinations_by_manufacturer.csv')

         additional_vaccines = ['CanSino', 'Pfizer/BioNTech', 'Johnson&Johnson', 'Covaxin', 'Novavax']

         selected_vaccines = ['Moderna', 'Oxford/AstraZeneca', 'Sinopharm/Beijing', 'Sputnik V'] + additional_vaccines

         df_selected_vaccines = df[df['vaccine'].isin(selected_vaccines)]

         vaccine_totals = df_selected_vaccines.groupby('vaccine')['total_vaccinations'].sum()

         plt.figure(figsize=(10,6))
         plt.bar(vaccine_totals.index, vaccine_totals.values, color='skyblue')
         plt.xlabel('Vaccine Type')
         plt.ylabel('Total Vaccinations')
         plt.title('Total Vaccinations by Vaccine Type')
         plt.xticks(rotation=45)
         plt.show()
```



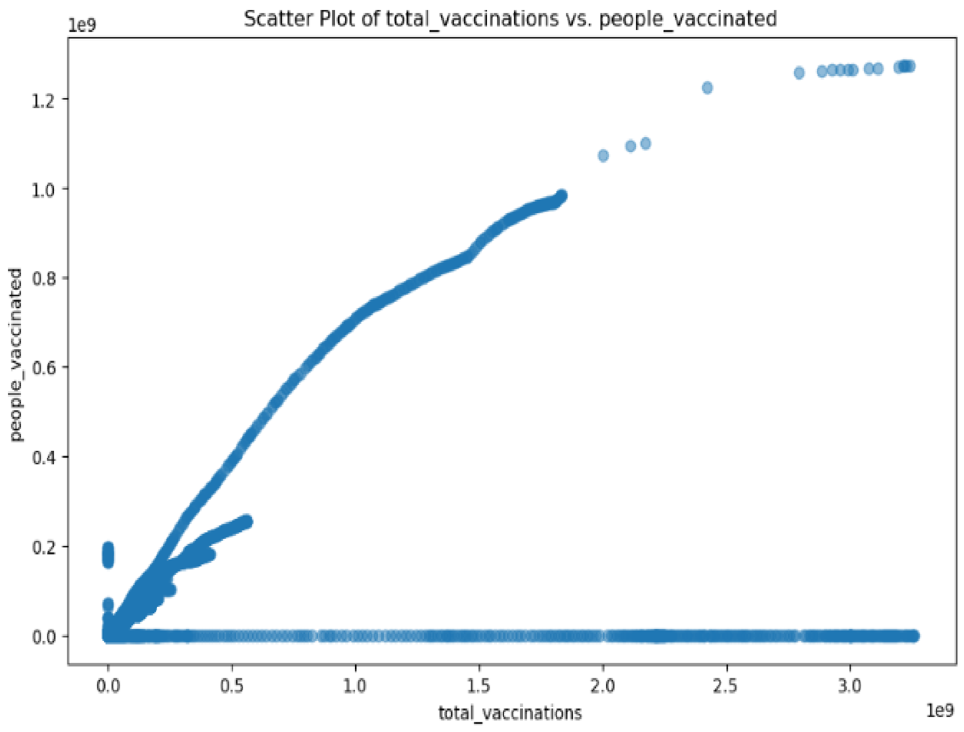
Scatter-Plot:

The purpose of the scatter plot is to display what happens to one variable when another variable is changed. The scatter plot is used to test a theory that the two variables are related.

The vaccination progress has been taken as the base theory involved in analysis and the two variable involved in the below scatter plot is 'total_vaccinations' and 'people_vaccinated' respectively.



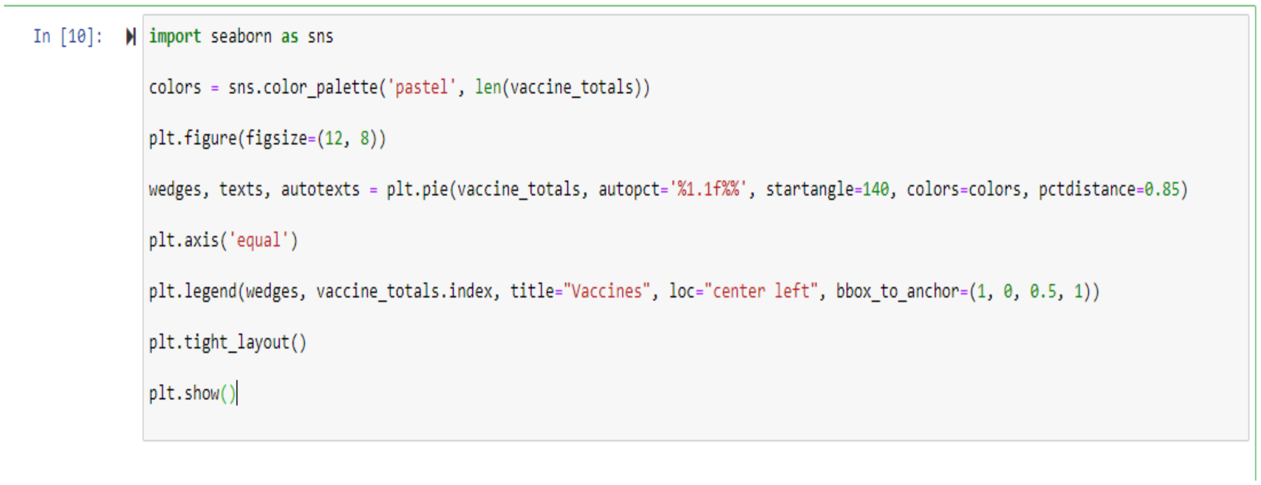
```
In [6]:  import pandas as pd
         import matplotlib.pyplot as plt

         df = pd.read_csv('country_vaccinations.csv')

         df['date'] = pd.to_datetime(df['date'])

         df.fillna(0, inplace=True)

         df = pd.get_dummies(df, columns=['country', 'iso_code'], drop_first=True)

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

         x_variable = 'total_vaccinations'
         y_variable = 'people_vaccinated'

         plt.scatter(df[x_variable], df[y_variable], alpha=0.5)

         plt.xlabel(x_variable)
         plt.ylabel(y_variable)

         plt.title(f'Scatter Plot of {x_variable} vs. {y_variable}')

         plt.show()
```

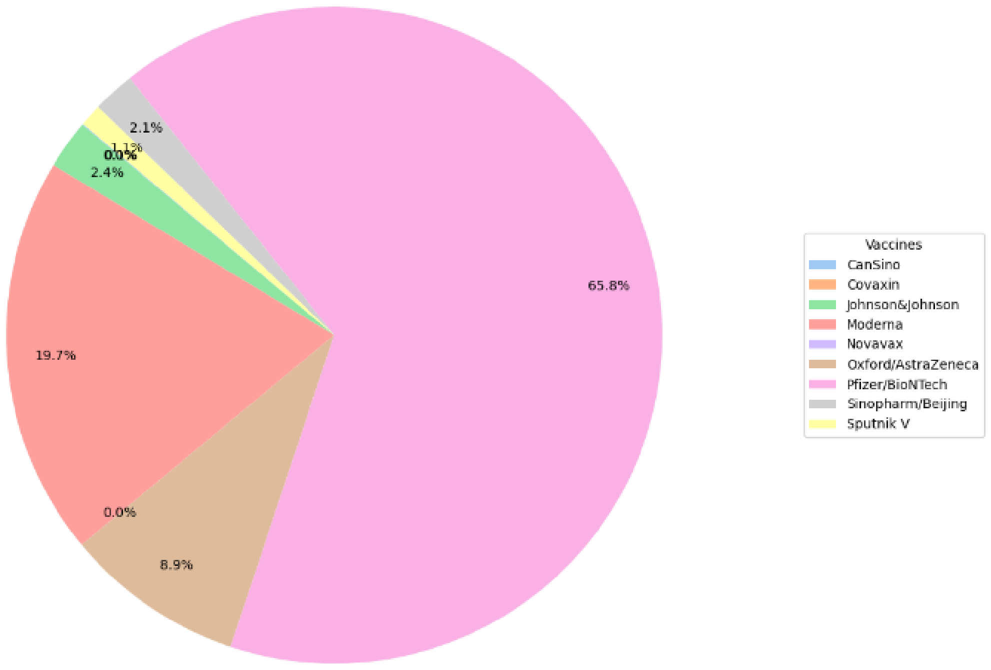


Pie-chart:

Determine the ratio or percentage that each component takes up out of the whole. The total sum of percentages should sum to 100%. Divide the circle into proportional sectors

We made a pie chart below to show how many people received each type of vaccine. Each slice represents a different vaccine, and the numbers inside the slices tell us the percentage. We used different colors to make each slice stand out. The key on the side tells us which color goes with each vaccine.


```
In [10]:  import seaborn as sns

          colors = sns.color_palette('pastel', len(vaccine_totals))

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

          wedges, texts, autotexts = plt.pie(vaccine_totals, autopct='%1.1f%%', startangle=140, colors=colors, pctdistance=0.85)

          plt.axis('equal')

          plt.legend(wedges, vaccine_totals.index, title="Vaccines", loc="center left", bbox_to_anchor=(1, 0, 0.5, 1))

          plt.tight_layout()

          plt.show()
```

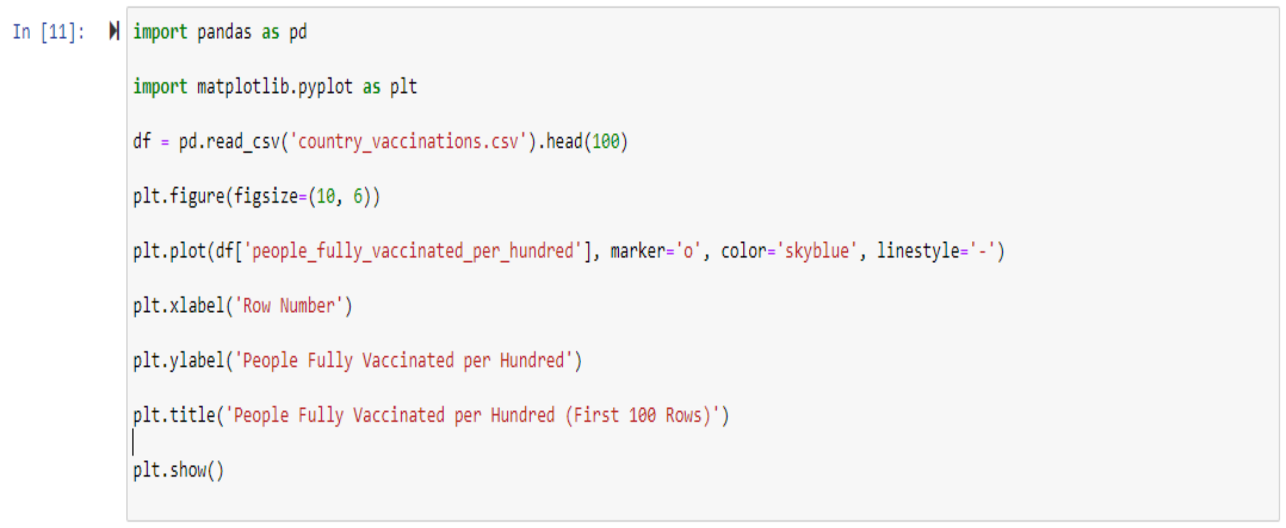


Line Plot:

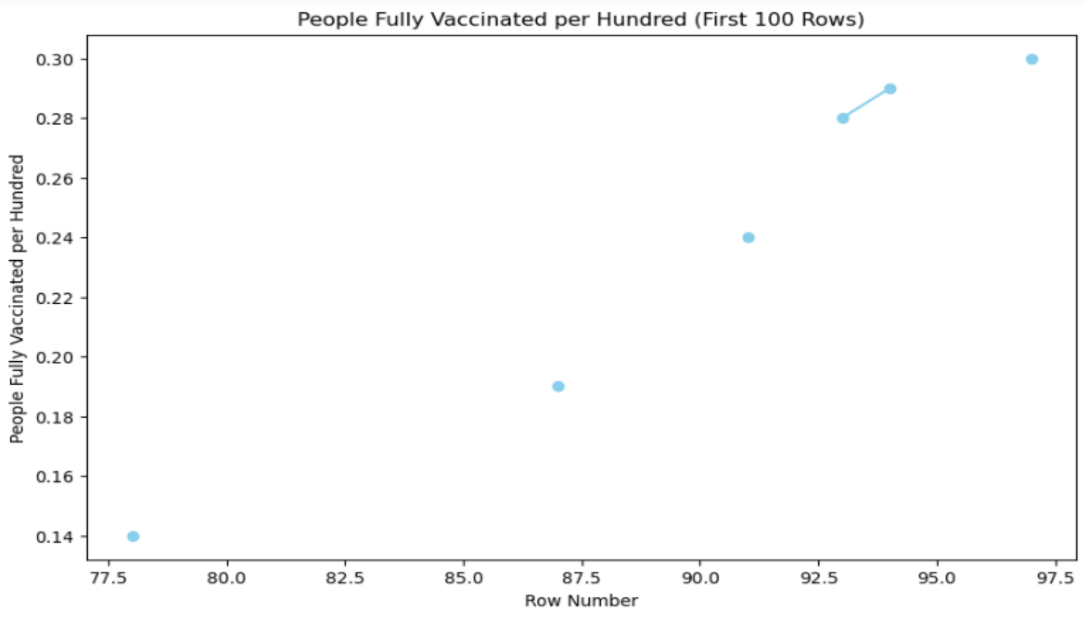
Line plot or a line chart—is a graph that uses lines to connect individual data points. A line graph displays quantitative values over a specified time interval.

The **Covid 19 Vaccine Analysis** has various content fields containing dynamic and vast range of data entries and it will be highly challenging for the data scientist to compare every single entry. This is where data visualization using line charts comes handy. Creating multiple line plots for individual fields will aid in quickly identifying and arriving at reliable insights-based decisions.

We made a line plot to show the people fully vaccinated per hundred.



```
In [11]:  import pandas as pd

          import matplotlib.pyplot as plt

          df = pd.read_csv('country_vaccinations.csv').head(100)

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

          plt.plot(df['people_fully_vaccinated_per_hundred'], marker='o', color='skyblue', linestyle='-')

          plt.xlabel('Row Number')

          plt.ylabel('People Fully Vaccinated per Hundred')

          plt.title('People Fully Vaccinated per Hundred (First 100 Rows)')

          plt.show()
```



CONCLUSION:

As mentioned above, we have removed missing values and cleaned(data cleaning) clumsy and vast data by removal of redundant dataset entries while also working on data integration to avoid complete removal of repetitive data and retain the entry by combining them.

Also we have displayed all visual elements that will provide the project's analysis phase an immense grip and clarity .

- to gain a quick and comparative knowledge of the world wide progress
- leading to clear and comprehensive analysis
- summing up to a more efficient and data driven decisions in various aspects.