**PROJECT TITTLE: COVID 19 VACCINE ANALYSIS**

**Packages to load:**

import os

for dirname, \_, filenames in os.walk('/kaggle/input'):

for filename in filenames:

print(os.path.join(dirname, filename))

**Items in the dataset:**

* Countries
* Dates
* Vaccines
* Total Vaccinations

**Desired data to find:**

* Most commonly used vaccines in countries
* Average daily vaccination count in countries -Number of countries where vaccines are used
* Choropleth map of the most used vaccine

**Loading the dataset:**

**Input:**

import pandas as pd

import plotly.express as px

import plotly.graph\_objects as go

from folium.features import Choropleth

import folium

from folium.features import Tooltip

import seaborn as sns

**Input:**

df = pd.read\_csv("/kaggle/input/covid-world-vaccination-progress/country\_vaccinations\_by\_manufacturer.csv")

df.head(10)

**Output:**



**Input:**

df["location"].nunique()

**Output:**

43

**Input:**

df.isnull().sum()

**Output:**

location 0

date 0

vaccine 0

total\_vaccinations 0

dtype: int64

**Input:**

df.dtypes

**Output:**

location object

date object

vaccine object

total\_vaccinations int64

dtype: object

It would be better to convert the Date column to the datetime type.

**Input:**

df['date'] = pd.to\_datetime(df['date'])

In our dataset, the Total Vaccinations represent the cumulative sum of vaccinations up to that date. To express the usage of different vaccines by countries, we need to clean the dataset and transform it.

**Input:**

data=pd.DataFrame(columns=['Country', 'Vaccine', 'Total\_vaccine'])

for country in df["location"].unique():

for vaccine in df["vaccine"].unique():

filtered\_data = df[(df['location'] == country) & (df['vaccine'] == vaccine)]

total\_count = filtered\_data['total\_vaccinations'].max()

data = pd.concat([data, pd.DataFrame({'Country': [country], 'Vaccine': [vaccine], 'Total\_vaccine': [total\_count]})], ignore\_index=True)

data.head(10)

**Output:**



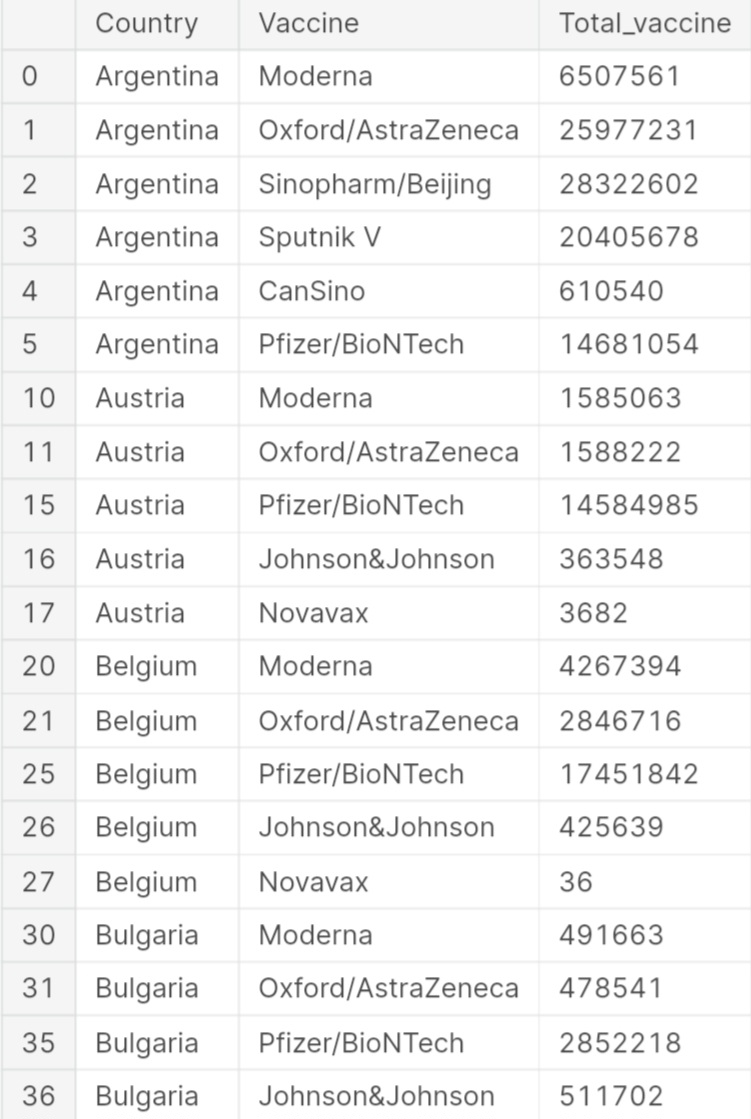
Since our new dataset includes rows for all countries and vaccine brands, we need to handle missing data.

**Input:**

data.dropna(axis=0,inplace=True)

data.head(20)

**Output:**



**Input:**

data\_2=pd.DataFrame(columns=['Country', 'Vaccine'])

data["Total\_vaccine"] = pd.to\_numeric(data["Total\_vaccine"], errors="coerce")

for country in data["Country"].unique():

new\_data = data[data["Country"] == country]

max\_vaccine = new\_data.loc[new\_data["Total\_vaccine"].idxmax(), "Vaccine"]

data\_2 = pd.concat([data\_2, pd.DataFrame({'Country': [country], 'Vaccine': [max\_vaccine]})], ignore\_index=True)

data\_2.head()

**Output:**

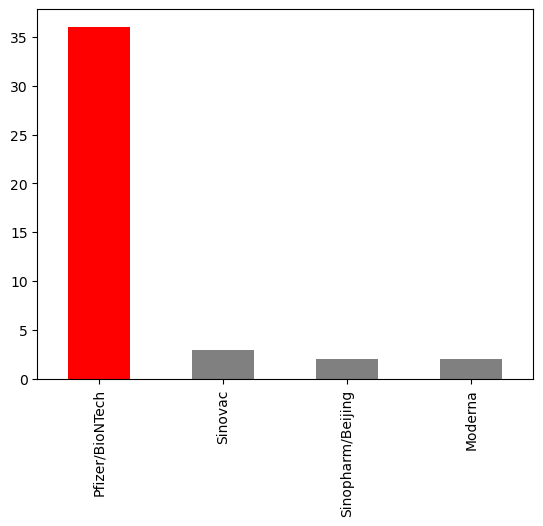


**Input:**

data\_2["Vaccine"].value\_counts().plot(kind="bar", color=["Red","Gray","Gray","Gray"])

**Output:**

3<Axes: >



Since the BioNTech vaccine is more widely used, I prefer to focus on analyzing it.Since the dataset does not provide the daily vaccination count, we can calculate the average vaccination count by dividing the total vaccinations by the number of days between the first and last date.

**Input:**

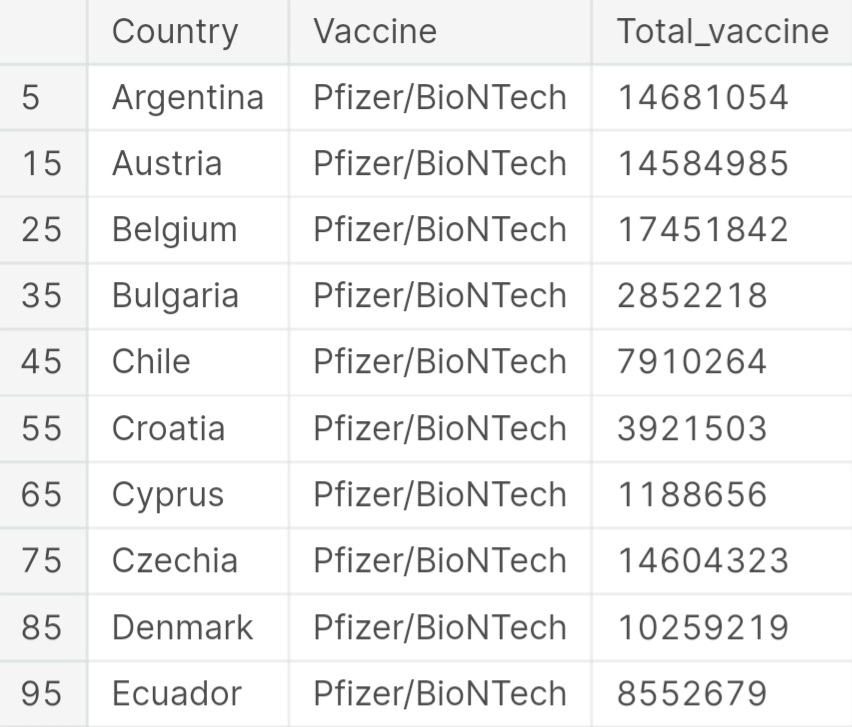
number\_of\_days = (df["date"].max() -df["date"].min() ).days

dtfrm=data[data["Vaccine"]=="Pfizer/BioNTech"]

dtfrm = dtfrm.drop(dtfrm[dtfrm['Country'] == 'European Union'].index)

dtfrm.head(10)

**Output:**



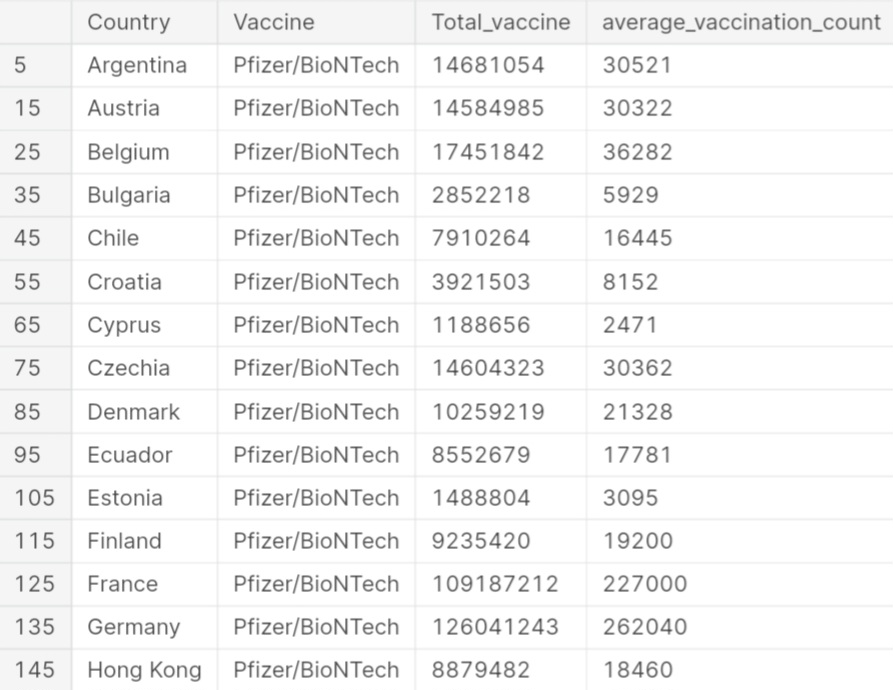
**Input:**

dtfrm["average\_vaccination\_count"] = dtfrm["Total\_vaccine"] / number\_of\_days

dtfrm["average\_vaccination\_count"] =dtfrm["average\_vaccination\_count"].astype(int)

dtfrm.head(15)

**Output:**



**Input:**

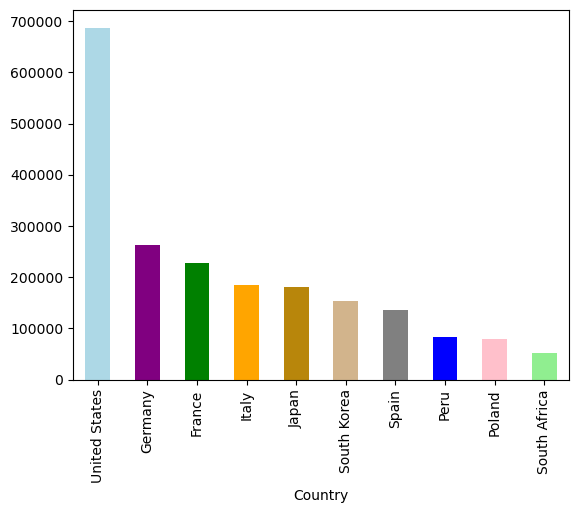
dtfrm.set\_index("Country",inplace=True)

color=["Lightblue","Purple","Green","Orange","darkgoldenrod","tan","Gray","Blue","Pink","Lightgreen"]

dtfrm["average\_vaccination\_count"].sort\_values(ascending=False).head(10).plot(kind="bar",color=color)

**Output:**

<Axes: xlabel='Country'>

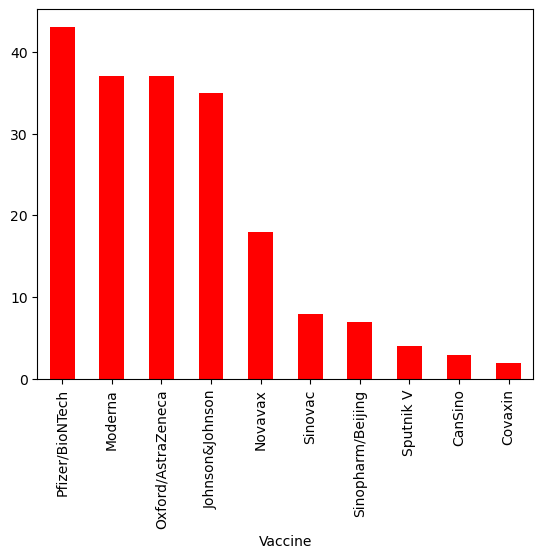


**Input:**

number\_of\_vaccines = data.groupby('Vaccine')['Country'].nunique()

number\_of\_vaccines.sort\_values(ascending=False).plot(kind="bar",color="r")

**Output:**



Visualizing country-level data on a map is a logical choice. Therefore, we will create a choropleth map showing the usage of the BioNTech vaccine by countries.

**Input:**

fig = px.choropleth(data\_frame=dtfrm,

locations=dtfrm.index,

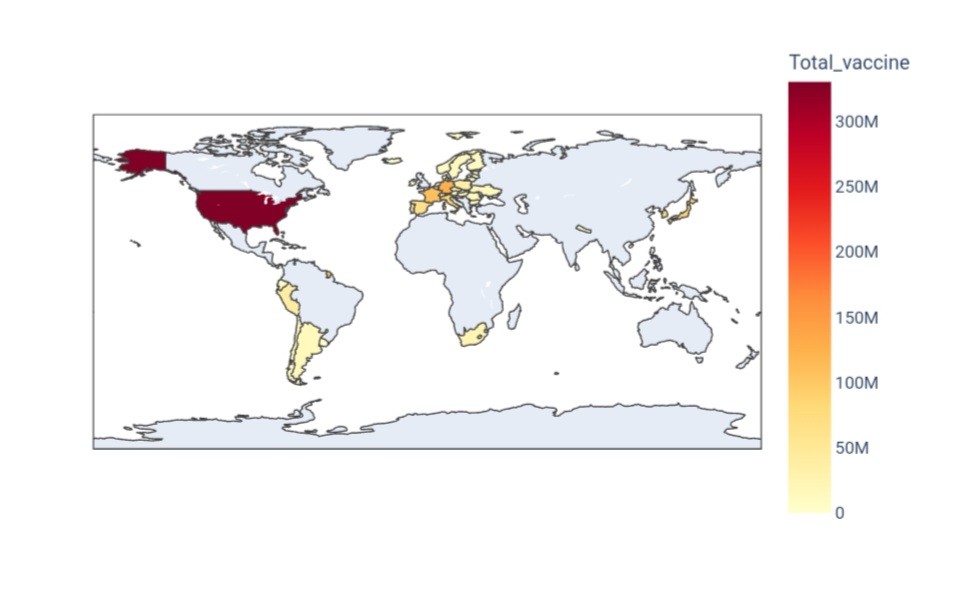
locationmode='country names',

color='Total\_vaccine',

color\_continuous\_scale='YlOrRd',

title='Ülkelerde Yapılan Biontech Aşıları')

fig.update\_layout(title\_x=0.5)



We can also create the same visualization using the Folium library.

**Input:**

m = folium.Map(location=[0, 0], zoom\_start=2)

Choropleth(

geo\_data='https://raw.githubusercontent.com/johan/world.geo.json/master/countries.geo.json',

name='choropleth',

data=dtfrm,

columns=[dtfrm.index, 'Total\_vaccine'],

key\_on='feature.properties.name',

fill\_color='YlOrRd',

fill\_opacity=0.7,

line\_opacity=0.2,

legend\_name='Aşı Sayısı',

).add\_to(m)

**Output:**

<folium.features.Choropleth at 0x7d414f2b7430>

**Input:**

m

**Output:**

