Program

```
vaccine_df.isnull().sum()  # Check for missing values
```

Output

```
country                                0
iso_code                               0
date                                   0
total_vaccinations                 42905
people_vaccinated                  45218
people_fully_vaccinated            47710
daily_vaccinations_raw             51150
daily_vaccinations                   299
total_vaccinations_per_hundred     42905
people_vaccinated_per_hundred      45218
people_fully_vaccinated_per_hundred 47710
daily_vaccinations_per_million       299
vaccines                               0
source_name                            0
source_website                         0
dtype: int64
```

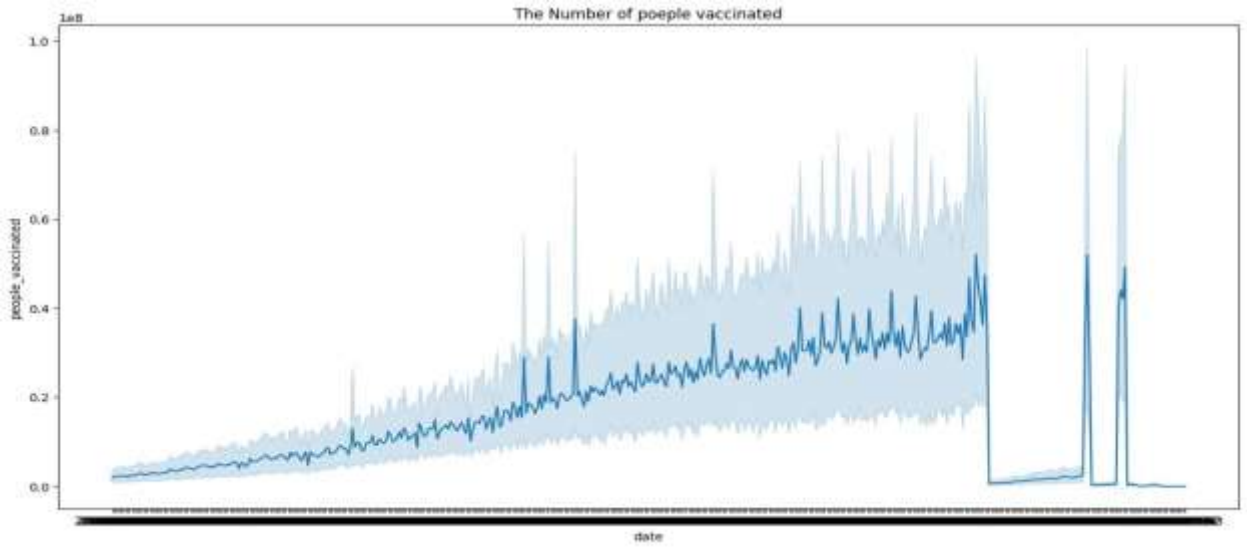
```
# Explore statistics

vaccine_df.describe()
```

Output

	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_vaccinations_raw	daily_va
count	4.360700e+04	4.129400e+04	3.880200e+04	3.536200e+04	8.6213(
mean	4.592964e+07	1.770508e+07	1.413830e+07	2.705996e+05	1.3130!
std	2.246004e+08	7.078731e+07	5.713920e+07	1.212427e+06	7.6823!
min	0.000000e+00	0.000000e+00	1.000000e+00	0.000000e+00	0.0000(
25%	5.264100e+05	3.494642e+05	2.439622e+05	4.668000e+03	9.0000(
50%	3.590096e+06	2.187310e+06	1.722140e+06	2.530900e+04	7.3430(
75%	1.701230e+07	9.152520e+06	7.559870e+06	1.234925e+05	4.4098(
max	3.263129e+09	1.275541e+09	1.240777e+09	2.474100e+07	2.2424!

```
import numpy as
np  import
pandas as pd
import
matplotlib.pyplot
as plt import
seaborn as sns
plt.figure(figsize
=(16,8))
sns.lineplot(x=vaccine_df.date,
y=vaccine_df.people_vaccinated) plt.title('The
Number of poeple vaccinated') plt.show()
```

Output



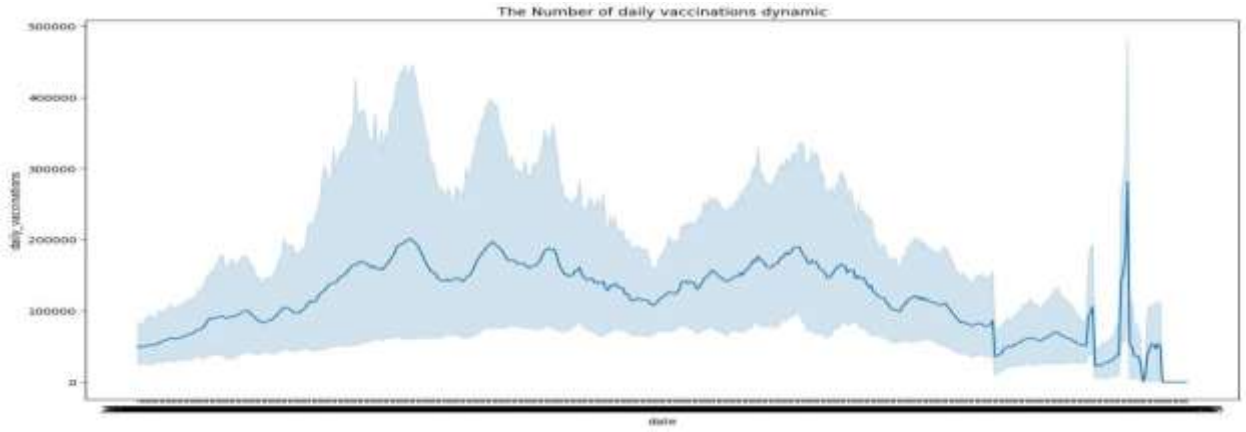

```
plt.figure(figsize=(16,8))
```

sns.lineplot(x=vaccine_df.date,

y=vaccine_df.daily_vaccinations) plt.title('The

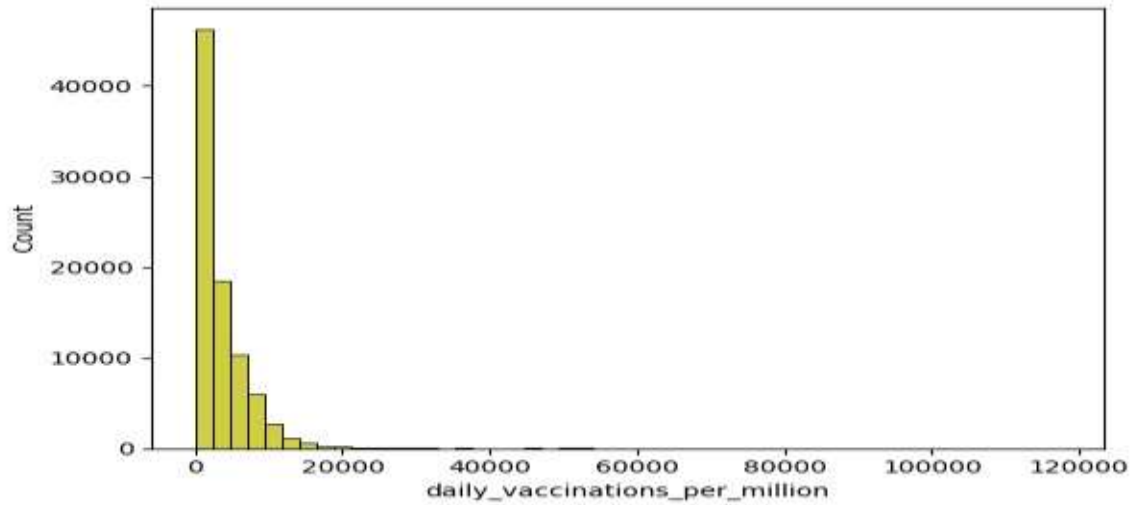
Number of daily vaccinations dynamic') plt.show()

Output



sns.histplot(vaccine_df, x='daily_vaccinations_per_million', bins=50, color='y')

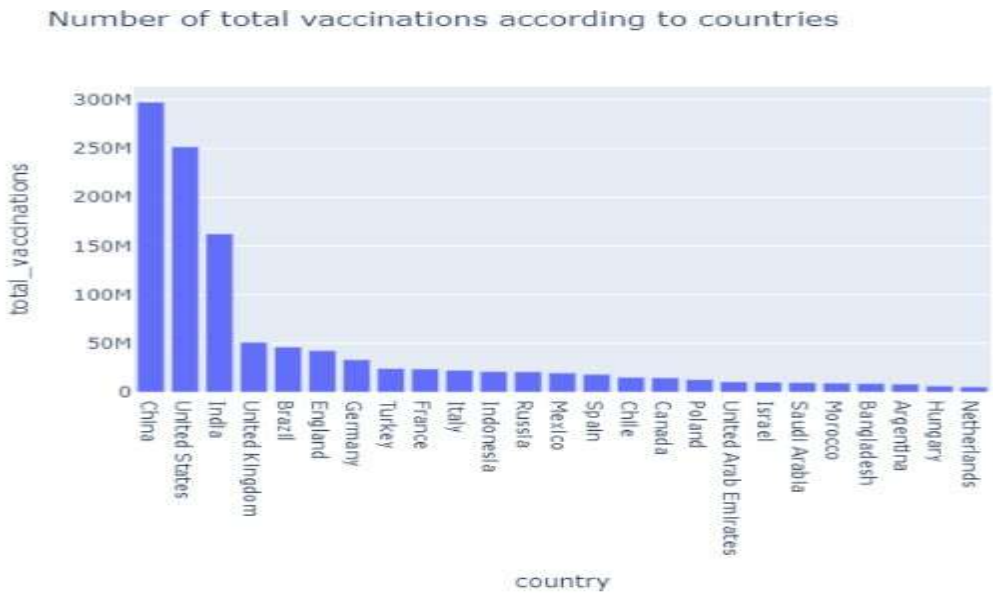
Output



data = vaccine_df[['country','total_vaccinations']].nlargest(25,'total_vaccinations')

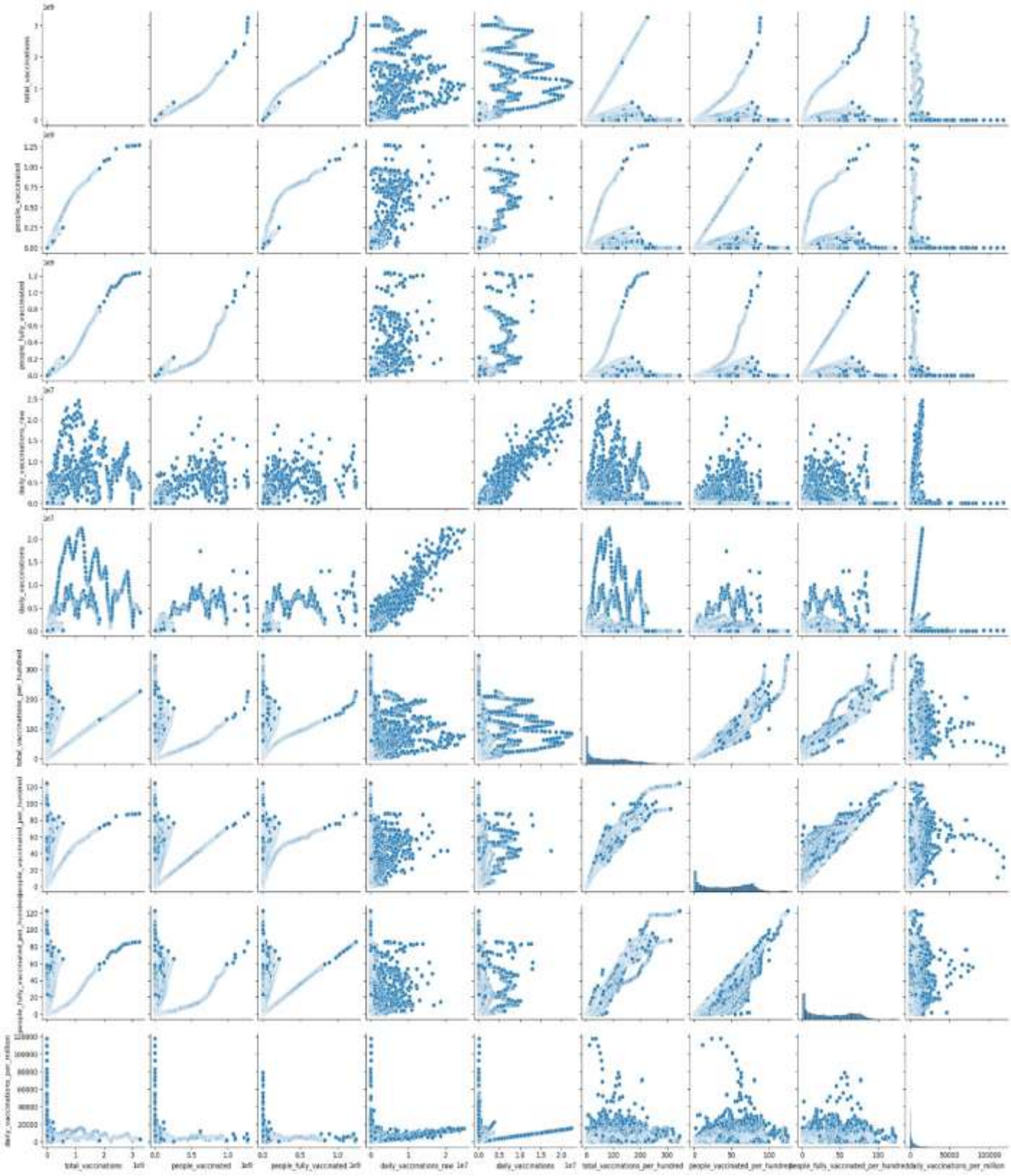
fig = px.bar(data, x = 'country',y = 'total_vaccinations',title="Number of total vaccinations according to countries",)fig.show()

Output



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

sns.pairplot(vaccine_df) **Output**

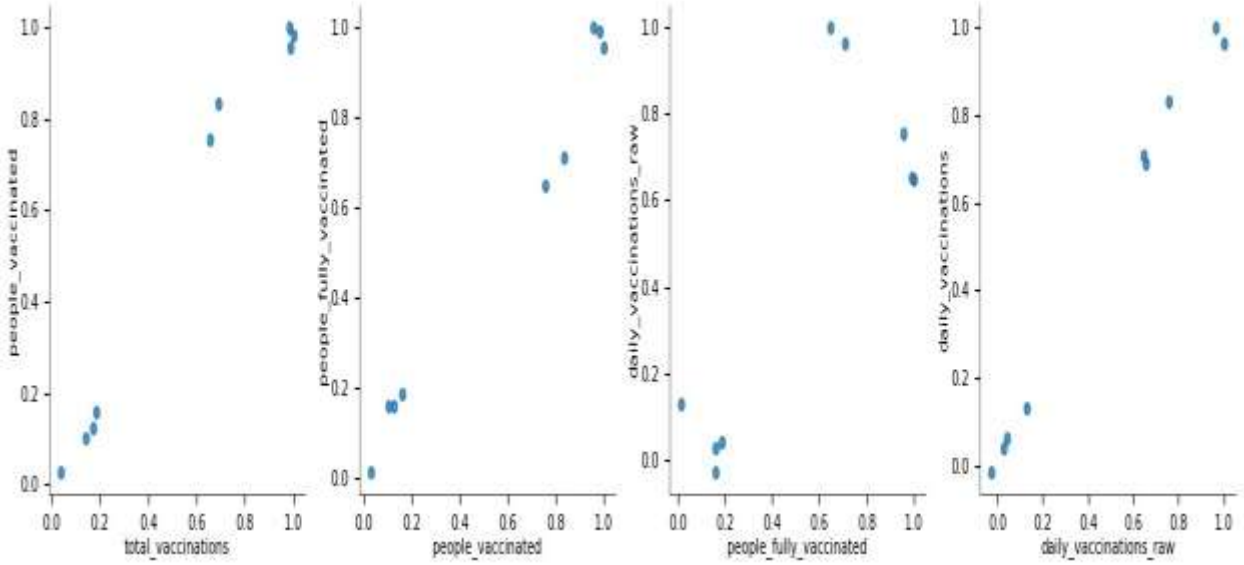



```
vaccine_df.corr(numeric_only=True)
```

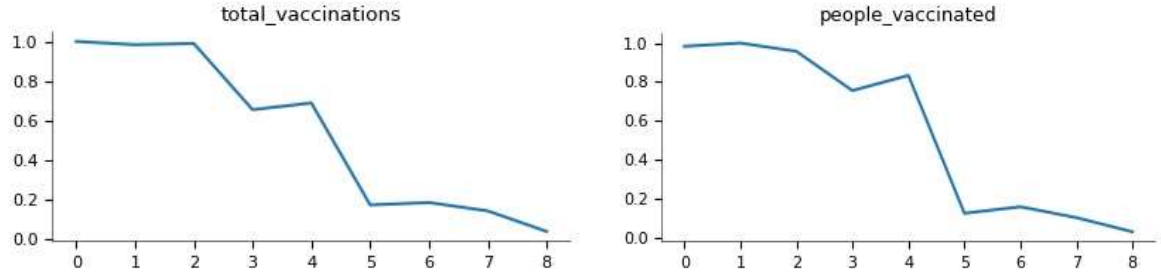
Output

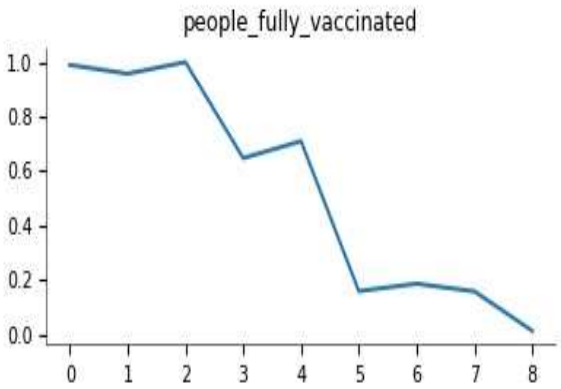
index	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_vaccinations_raw	daily_vaccinations	total_vaccinations_per_hundred
total_vaccinations	1.0	0.983438280596498	0.989681338130297	0.654711788190802	0.688501888912114	0.172297100046722
people_vaccinated	0.983438280596498	1.0	0.957599480057860	0.755540225078966	0.833433297482937	0.123938035999737
people_fully_vaccinated	0.989681338130297	0.957599480057860	1.0	0.647573972663791	0.709768165406591	0.159026253335580
daily_vaccinations_raw	0.654711788190802	0.755540225078966	0.647573972663791	1.0	0.965516572583910	0.029328840509383
daily_vaccinations	0.688501888912114	0.833433297482937	0.709768165406591	0.965516572583910	1.0	0.042227286198858
total_vaccinations_per_hundred	0.172297100046722	0.123938035999737	0.159026253335580	0.029328840509383	0.042227286198858	1.0
people_vaccinated_per_hundred	0.184649054590190	0.157775317674897	0.186368734714912	0.042445074564014	0.062566862456959	0.965329313791278
people_fully_vaccinated_per_hundred	0.142252148700157	0.101717390787256	0.158283358797382	-0.027884824018809	-0.014054926124652	0.975454630947068
daily_vaccinations_per_million	0.038298146800641	0.028720142515809	0.013220268364296	0.131078103995991	0.133822261913642	0.184608884596478

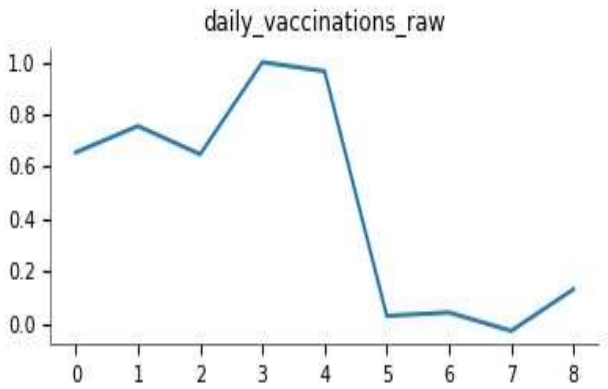
2-d Distributions



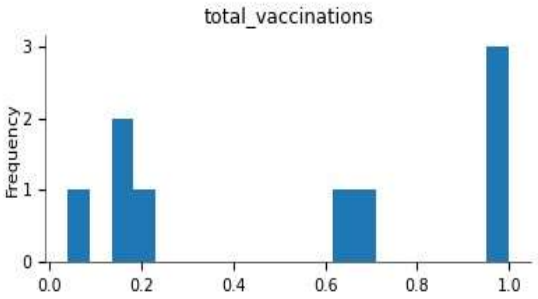
Values

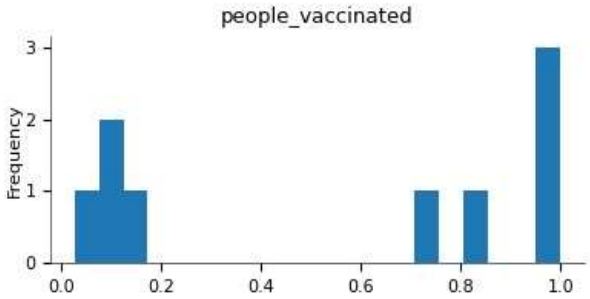


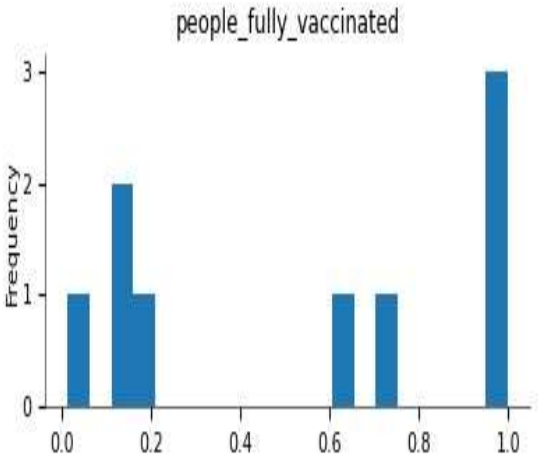


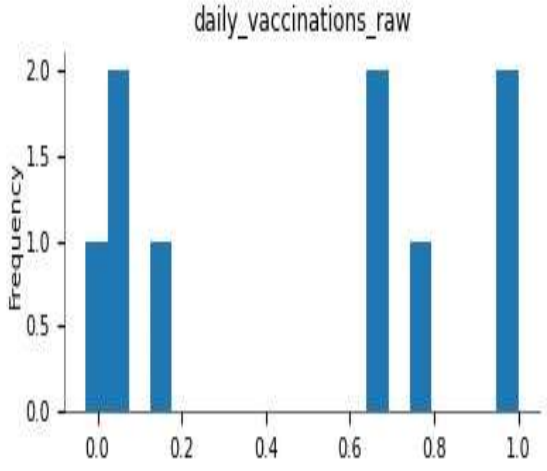


Distributions





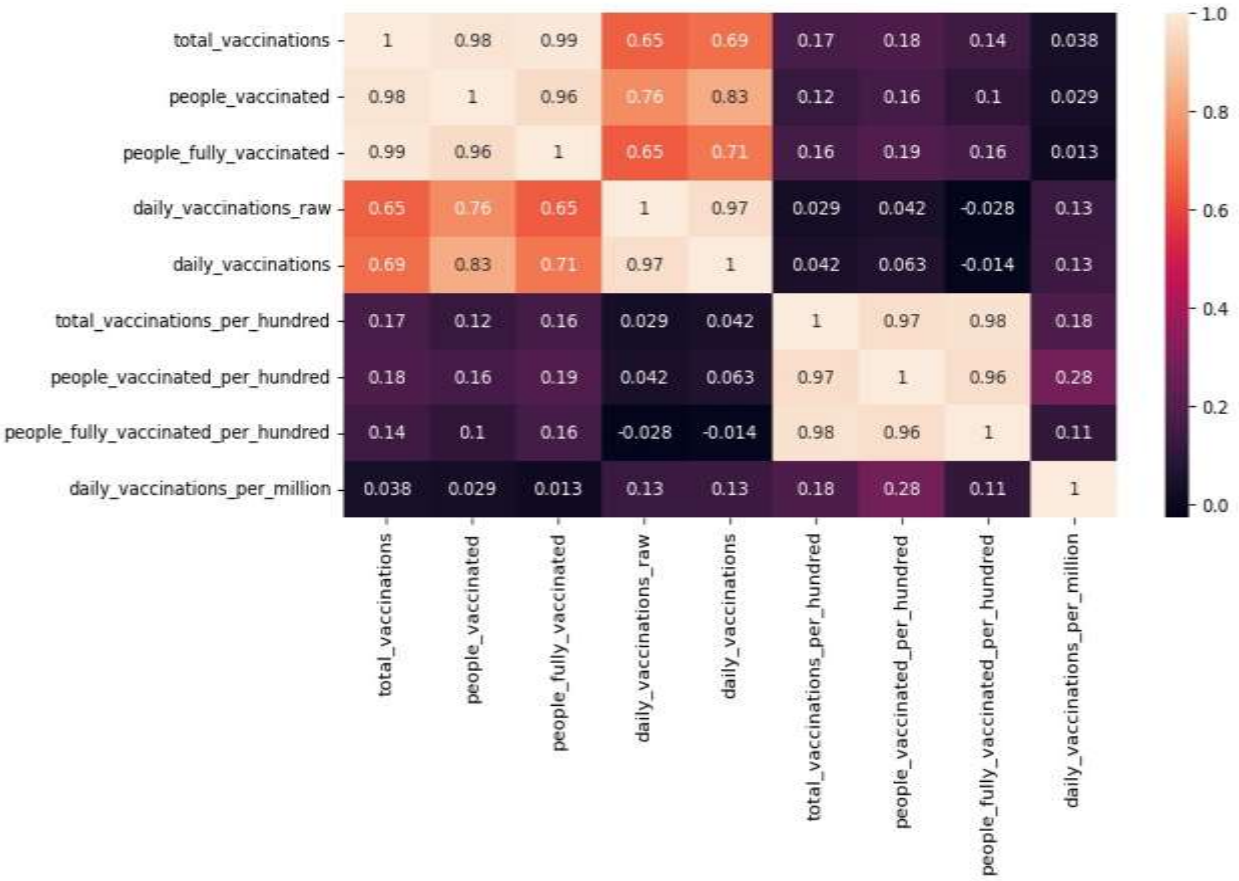




plt.figure(figsize=(10,5))

sns.heatmap(vaccine_df.corr(numeric_only = True), annot=True)

Output



1. **Importing Required Libraries**

   import numpy as np

   import

pandas as pd
import
seaborn as
sns import
matplotlib.p
yplot as plt
import
plotly.expres
s as px
import
plotly.graph
_objs as go
from
plotly.offline
import
init_noteboo
k_mode,
iplot, plot

2. **Loading Data Sets Read data from CSV files**

```
df_manufacturer =
pd.read_csv('/content/drive/MyDrive/Naan
Mudalvan/country_vaccinations_by_manufacturer.csv')
df_vaccinations =
pd.read_csv('/content/drive/MyDrive/Naan
Mudalvan/country_vaccinations.csv')
```

Columns Present in Given Data Sets

df_manufacturer.colum

ns *Output:*

['location', 'date', 'vaccine', 'total_vaccinations']

df_vaccinations.co
lumns *Output:*

['country', 'iso_code', 'date', 'total_vaccinations', 'people_vaccinated',
'people_fully_vaccinated',

'daily_vaccinations_raw', 'daily_vaccinations', 'total_vaccinations_per_hundred',

'people_vaccinated_per_hundred', 'people_fully_vaccinated_per_hundred',

'daily_vaccinations_per_million', 'vaccines', 'source_name', 'source_website']

Shape of
DataFrames

df_manufacturer.
shape *Output:*

(35623, 4)

df_vaccinations.
shape *Output:*

(86512, 15)

Information about

Given Data Sets

df_manufacturer.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_vaccinations.
info() *Output:*

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 86512 entries, 0 to 86511
Data columns (total 15 columns):
 #   Column                               Non-Null Count  Dtype
---  ------                               --------------  -----
 0   country                              86512 non-null  object
 1   iso_code                             86512 non-null  object
 2   date                                 86512 non-null  object
 3   total_vaccinations                   43607 non-null  float64
 4   people_vaccinated                    41294 non-null  float64
 5   people_fully_vaccinated              38802 non-null  float64
 6   daily_vaccinations_raw               35362 non-null  float64
 7   daily_vaccinations                   86213 non-null  float64
 8   total_vaccinations_per_hundred       43607 non-null  float64
 9   people_vaccinated_per_hundred        41294 non-null  float64
 10  people_fully_vaccinated_per_hundred  38802 non-null  float64
 11  daily_vaccinations_per_million       86213 non-null  float64
 12  vaccines                             86512 non-null  object
 13  source_name                          86512 non-null  object
 14  source_website                       86512 non-null  object
dtypes: float64(9), object(6)
memory usage: 9.9+ MB
```

Vaccines Manufactured on a Particular Date
df_manufacturer = df_manufacturer[df_manufacturer.date == '2022-02-04'] df_manufacturer.head() *Output:*

	location	date	vaccine	total_vaccinations
2305	Argentina	2022-02-04	CanSino	468481
2306	Argentina	2022-02-04	Moderna	5318406
2307	Argentina	2022-02-04	Oxford/AstraZeneca	25606912
2308	Argentina	2022-02-04	Pfizer/BioNTech	11225368
2309	Argentina	2022-02-04	Sinopharm/Beijing	27396208

Country-Wise Vaccination Status on a Particular Date

df_vaccinations = df_vaccinations[df_vaccinations.date == '2022-02-04'] df_vaccinations.head()

Output:

Checking for Missing Values

df_manufacturer.isna(). sum() *Output:*

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

df_vaccinations.isna(). sum() *Output:*

```
country                                0
iso_code                               0
date                                   0
total_vaccinations                   112
people_vaccinated                    119
people_fully_vaccinated              115
daily_vaccinations_raw               132
daily_vaccinations                     0
total_vaccinations_per_hundred       112
people_vaccinated_per_hundred        119
people_fully_vaccinated_per_hundred  115
daily_vaccinations_per_million         0
vaccines                               0
source_name                            0
source_website                         0
dtype: int64
```

Dropping Missing Values

df_vaccinations.isna().sum() *Output:*

```
country                                0
iso_code                               0
date                                   0
total_vaccinations                   112
people_vaccinated                    119
people_fully_vaccinated              115
daily_vaccinations_raw               132
daily_vaccinations                     0
total_vaccinations_per_hundred       112
people_vaccinated_per_hundred        119
people_fully_vaccinated_per_hundred  115
daily_vaccinations_per_million         0
vaccines                               0
source_name                            0
source_website                         0
dtype: int64
```

df_vaccinations = df_vaccinations.drop(df_vaccinations[df_vaccinations.total_vaccinations.isna()].index) df_vaccinations = df_vaccinations.drop(df_vaccinations[df_vaccinations.people_vaccinated.isna()].index) df_vaccinations = df_vaccinations.drop(df_vaccinations[df_vaccinations.daily_vaccinations_raw.isna()].index)

Checking for Null Values

df_vaccinations.isnull().
sum() *Output:*

```
country                              0
iso_code                             0
date                                 0
total_vaccinations                   0
people_vaccinated                    0
people_fully_vaccinated              1
daily_vaccinations_raw               0
daily_vaccinations                   0
total_vaccinations_per_hundred       0
people_vaccinated_per_hundred        0
people_fully_vaccinated_per_hundred  1
daily_vaccinations_per_million       0
vaccines                             0
source_name                          0
source_website                       0
dtype: int64
```

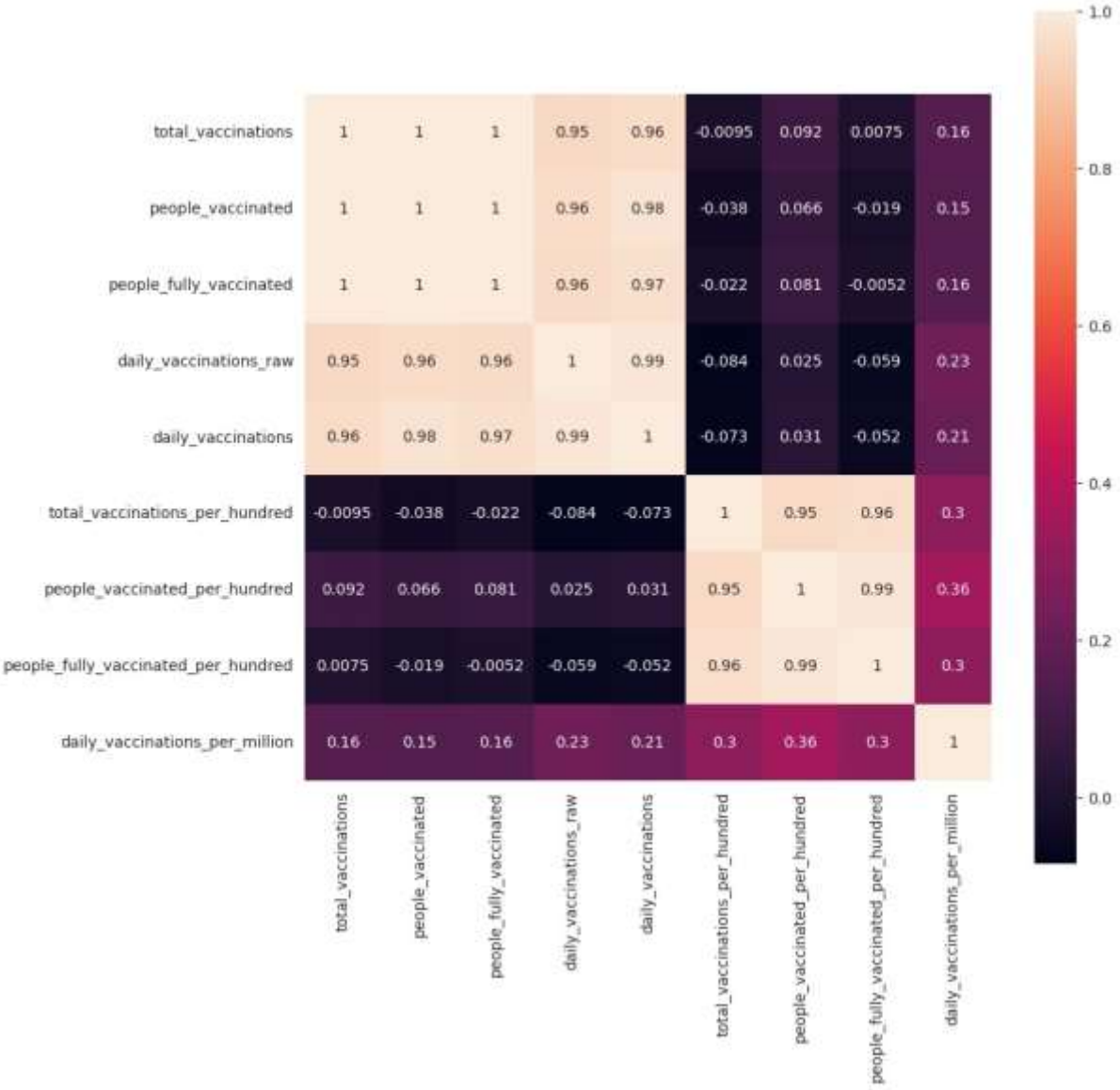
Filling Mean Values df_vaccinations = df_vaccinations.fillna(df_vaccinations.mean())
df_vaccinations.isnull().sum() *Output:*

```
country                              0
iso_code                             0
date                                 0
total_vaccinations                   0
people_vaccinated                    0
people_fully_vaccinated              0
daily_vaccinations_raw               0
daily_vaccinations                   0
total_vaccinations_per_hundred       0
people_vaccinated_per_hundred        0
people_fully_vaccinated_per_hundred  0
daily_vaccinations_per_million       0
vaccines                             0
source_name                          0
source_website                       0
dtype: int64
```

Checking for Duplicated Records
duplicate_rows = df_vaccinations[df_vaccinations.duplicated()]
print(len(duplicate_rows)) print(duplicate_rows)
Output:

```
0
Empty DataFrame
Columns: [country, iso_code, date, total_vaccinations,
Index: []
```

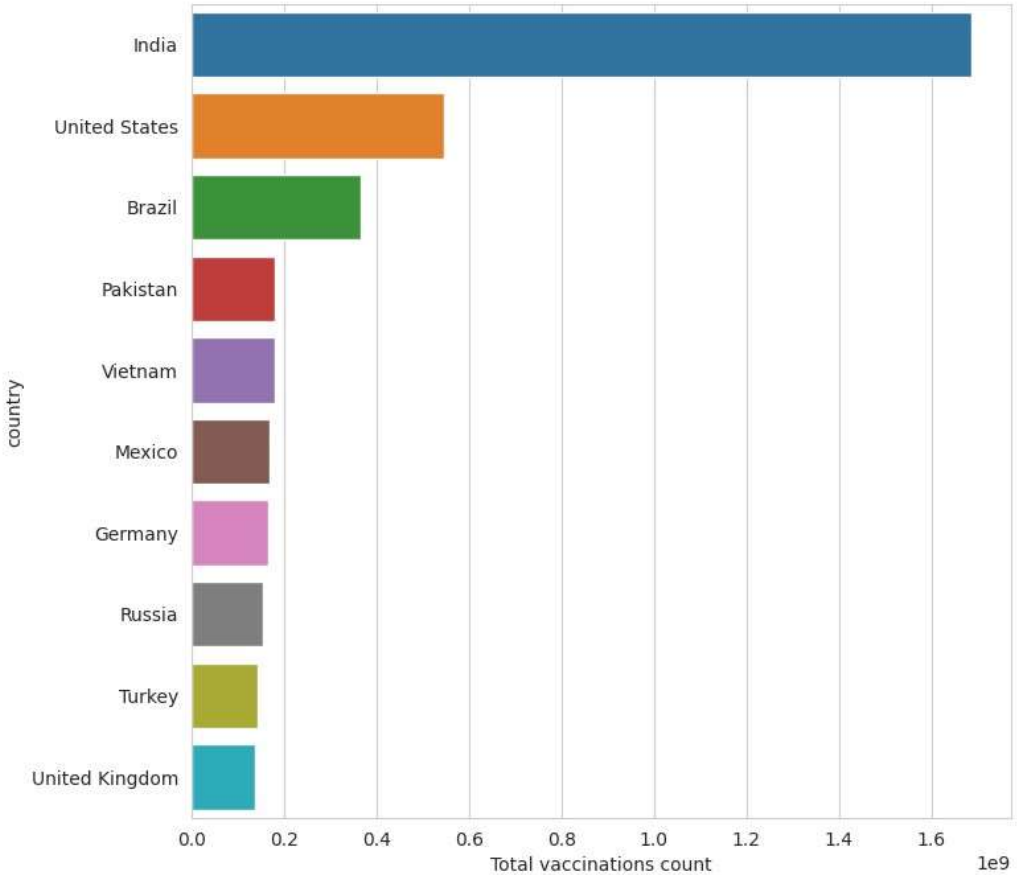
Heatmap Visualization to Check Correlation Between Attributes plt.subplots(figsize=(10, 10)) sns.heatmap(df_vaccinations.corr(), annot=True, square=True) plt.show()



Top Countries in Vaccination Utilization
df_vaccinations["Total_vaccinations_count"] = df_vaccinations.groupby("country").total_vaccinations.tail(1)

```
country
India             1.687048e+09
United States     5.469684e+08
Brazil            3.677782e+08
Pakistan          1.823960e+08
Vietnam           1.816654e+08
Mexico            1.685357e+08
Germany           1.666940e+08
Russia            1.553786e+08
Turkey            1.427355e+08
United Kingdom    1.384598e+08
Name: Total_vaccinations_count, dtype: float64
```

```
x = df_vaccinations.groupby("country")["Total_vaccinations_count"].mean().sort_values(ascending
=False).head(10)
sns.set_style("whitegrid")
plt.figure(figsize=(8, 8)) ax
= sns.barplot(x=x.values,
y=x.index)
ax.set_xlabel("Total
vaccinations count")
plt.show()
```



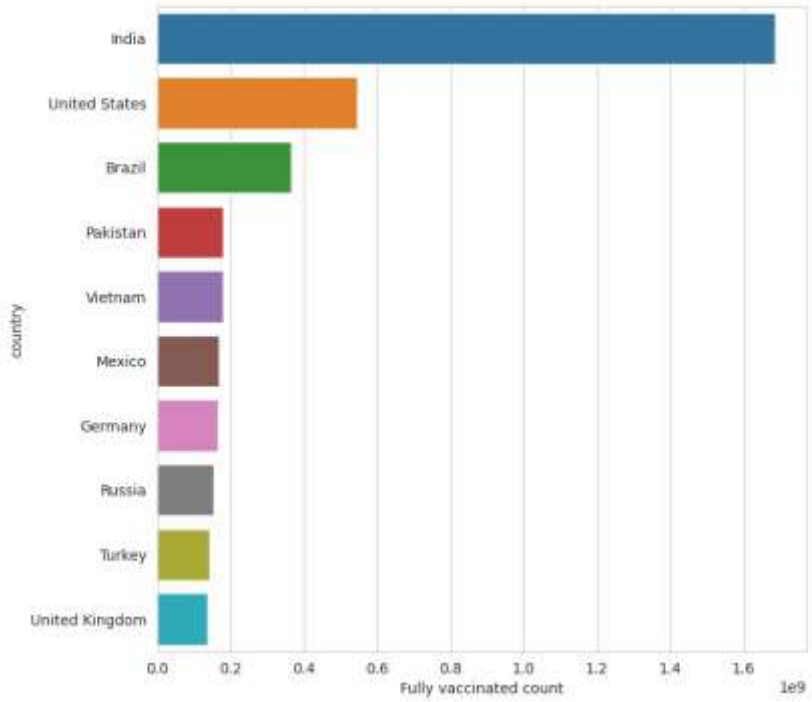
Fully Vaccinated Count

df_vaccinations["Full_vaccinations_count"] = df_vaccinations.groupby("country").people_fully_vaccinated.tail(1)

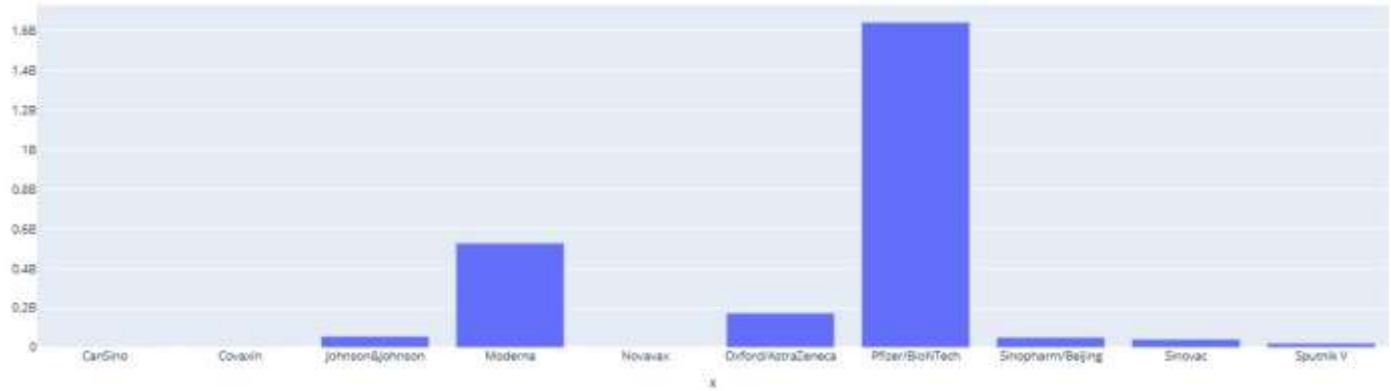
```
country
India            724768356.0
United States    213893460.0
Brazil           150682483.0
Pakistan          84731497.0
Mexico            77478070.0
Vietnam           74187748.0
Russia            70232028.0
Germany           61873548.0
Iran              54405243.0
Turkey            52489431.0
Name: Full_vaccinations_count, dtype: float64
```

x =

df_vaccinations.groupby("country")["Full_vaccinations_count"].mean().sort_values(ascending= False).head(10) sns.set_style("whitegrid") plt.figure(figsize=(8, 8)) ax = sns.barplot(x=x.values, y=x.index) ax.set_xlabel("Fully vaccinated count") plt.show()

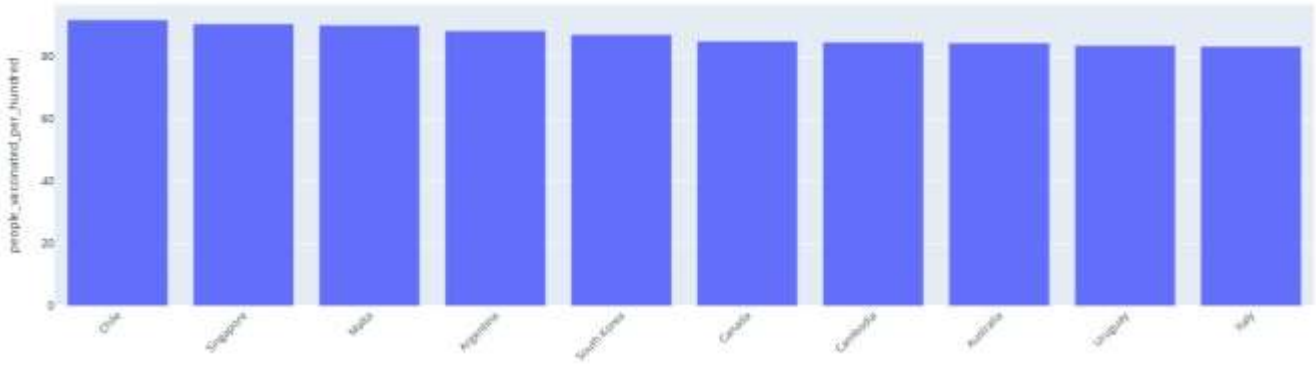


Most Commonly Used Vaccines in the World total = df_manufacturer.groupby('vaccine').sum() px.bar(x=total.index, y=total['total_vaccinations'], title='Most Used Vaccine in the World')



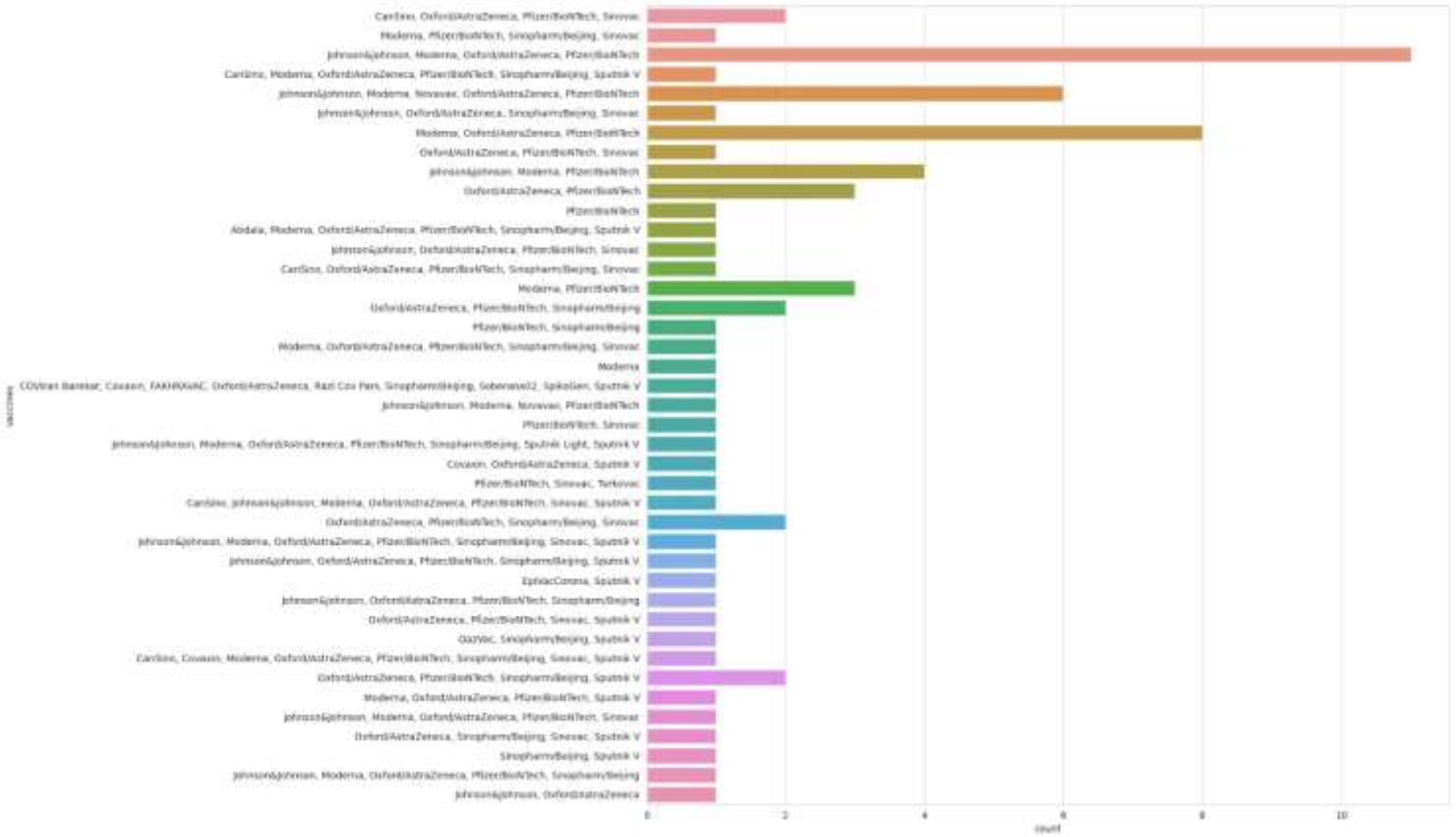
People Vaccinated per Hundred for the Date 2022-02-04

df_vaccinations = df_vaccinations[df_vaccinations['date'] == '2022-02-04'] df_vaccinations = df_vaccinations.sort_values(by='people_vaccinated_per_hundred', ascending=False) fig = px.bar(df_vaccinations.head(10), x='country', y='people_vaccinated_per_hundred', title='People Vaccinated per Hundred for the Date 2022-02-04') fig.update_layout(xaxis_tickangle=-45) fig.show()



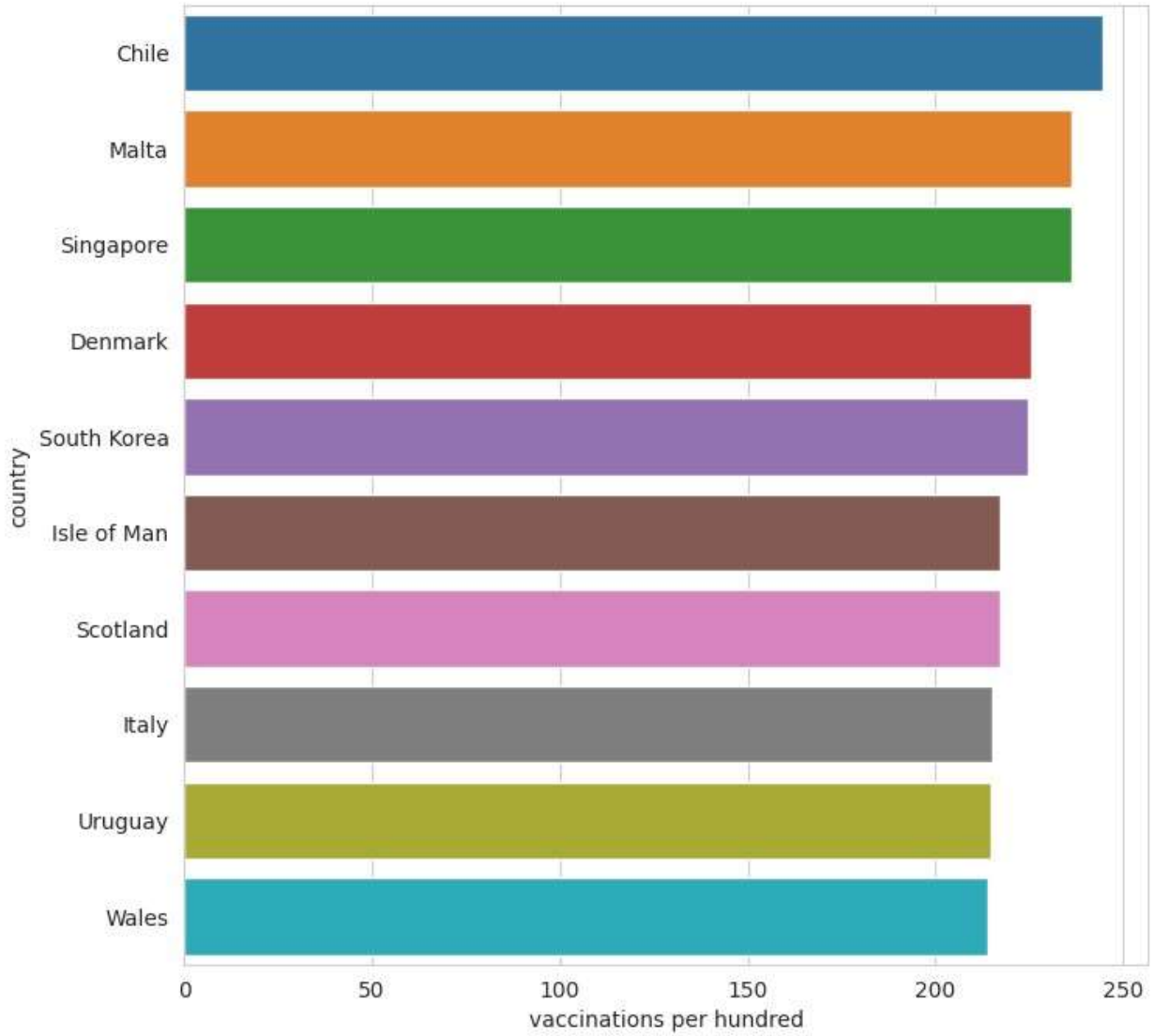
Type of Vaccine Utilized vs Count

plt.figure(figsize=(15, 15)) sns.countplot(y="vaccines", data=df_vaccinations) plt.show()



Vaccination per Hundred Top Countries

df_vaccinations["Total_vaccinations_per_hundred"] = df_vaccinations.groupby("country").total_vaccinations_per_hundred.tail(1) x = df_vaccinations.groupby("country")["Total_vaccinations_per_hundred"].mean().sort_values(asc ending=False).head(10) plt.figure(figsize=(8, 8)) ax = sns.barplot(x=x.values, y=x.index) ax.set_xlabel("Vaccinations per hundred") plt.show()



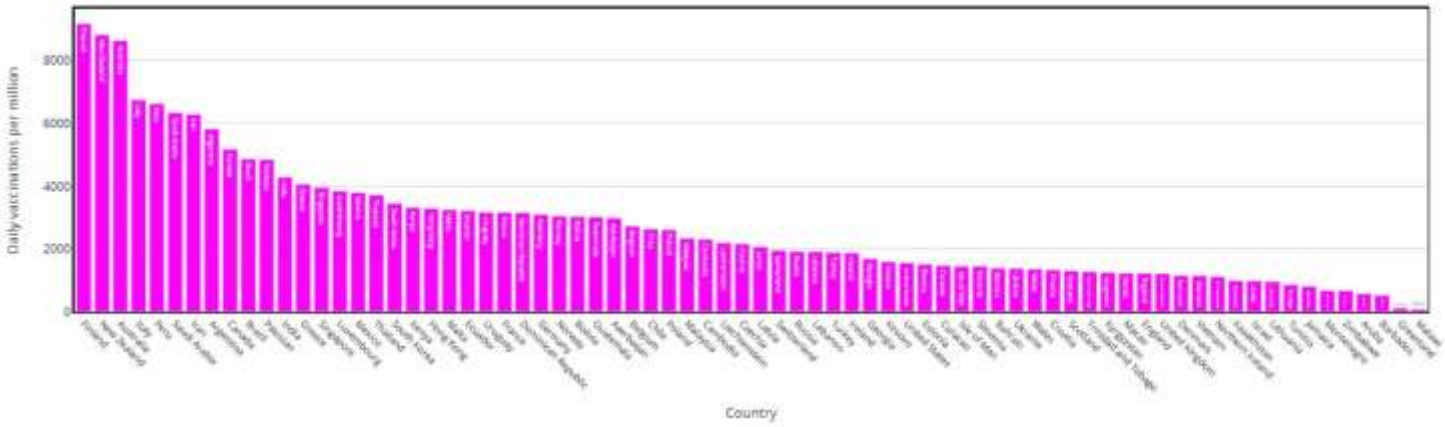
Country-Wise Daily Vaccination per Million

```
def trace_bar(data, feature, title, xlab,
ylab, color):     data =
data.sort_values(feature,
ascending=False)     trace = go.Bar(
x=data['country'],
y=data[feature],
marker=dict(color=color),
text=data['country']

    )
```

```
   data = [trace]     layout =
dict(         title=title,
xaxis=dict(             title=xlab,
showticklabels=True,
tickangle=45,
zeroline=True,
zerolinewidth=1,
zerolinecolor='grey',
showline=True,
linewidth=2,
linecolor='black',
mirror=True,
tickfont=dict(size=10,
color='black'),

       ),


yaxis=dict(
title=ylab,
gridcolor='lig
htgrey',
zeroline=True
,
zerolinewidth
=1,
zerolinecolor=
'grey',
showline=Tru
e,
linewidth=2,
linecolor='bla
ck',
mirror=True

       ),
       plot_bgcolor='rgba(0, 0, 0, 0)',
paper_bgcolor='rgba(0, 0, 0, 0)',
hovermode='closest'

   )
```

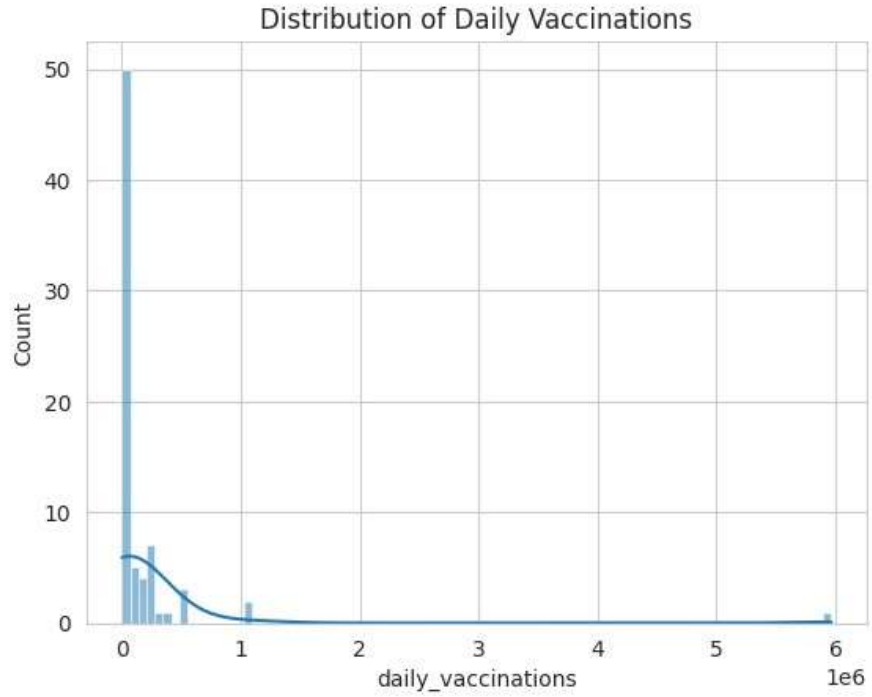
fig = dict(data=data,
layout=layout) iplot(fig)

trace_bar(df_vaccinations, 'daily_vaccinations_per_million', 'Daily Vaccinations per Million per Country', 'Country', 'Daily Vaccinations per Million', 'blue')



Distribution of Daily Vaccine

sns.histplot(df_vaccinations['daily_vaccination s'], kde=True) plt.title("Distribution of Daily Vaccinations") plt.show()



Statistical Analysis of Given Data

Sets Descriptive Statistics:

- *Mean:* The average of a set of values.
- *Median:* The middle value of a sorted dataset.
- *Standard Deviation:* A measure of the amount of variation or dispersion in a set of values.

Total Vaccinations Statistical Analysis

df_manufacturer['total_vaccinations'].des
cribe() *Output:*

	total_vaccinations
count	1.600000e+02
mean	1.574315e+07
std	5.730594e+07
min	0.000000e+00
25%	2.378710e+05
50%	1.569373e+06
75%	9.042346e+06
max	5.821192e+08

df_vaccinations.describe() *Output:*

	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_vaccinations_raw	daily_vaccinations	total_vaccinations_per_hundred	people_vaccinated_per_hundred	people_fully_vaccinat
count	7.400000e+01	7.400000e+01	7.400000e+01	7.400000e+01	7.400000e+01	74.000000	74.000000	
mean	7.341445e+07	3.528194e+07	3.046394e+07	2.092764e+05	1.883525e+05	156.099054	66.902568	
std	2.094558e+08	1.149865e+08	8.904082e+07	6.877380e+05	7.106291e+05	55.557085	16.881493	
min	6.936300e+04	2.667600e+04	2.667400e+04	0.000000e+00	7.000000e+00	9.610000	7.670000	
25%	3.041524e+06	1.438836e+06	1.296678e+06	3.582500e+03	3.229500e+03	119.387500	57.685000	
50%	1.300018e+07	6.382784e+06	5.586156e+06	3.372050e+04	2.401850e+04	166.300000	72.355000	
75%	6.246783e+07	2.616142e+07	2.915660e+07	1.294548e+05	1.485206e+05	198.355000	80.177500	
max	1.657048e+09	9.467174e+08	7.247684e+08	5.530745e+06	5.964928e+06	244.500000	91.900000	

People Fully Vaccinated Statistical Analysis

df_manufacturer['people_fully_vaccinated'].describe() *Output:*

```
count    7.400000e+01
mean     3.046394e+07
std      8.904082e+07
min      2.607400e+04
25%      1.296678e+06
50%      5.586156e+06
75%      2.915660e+07
max      7.247684e+08
Name: people_fully_vaccinated, dtype: float64
```

Daily Vaccinations Statistical Analysis

df_manufacturer['daily_vaccinations'].des
cribe() *Output:*

```
count    7.400000e+01
mean     1.883525e+05
std      7.106291e+05
min      7.000000e+00
25%      3.229500e+03
50%      2.401850e+04
75%      1.485200e+05
max      5.964928e+06
Name: daily_vaccinations, dtype: float64
```

Total Vaccinations in Country

Statistical Analysis

df_vaccinations['total_vaccinations'].des

cribe() *Output:*

```
count    7.400000e+01
mean     7.341445e+07
std      2.094558e+08
min      6.936300e+04
25%      3.041524e+06
50%      1.300016e+07
75%      6.246783e+07
max      1.687048e+09
Name: total_vaccinations, dtype: float64
```

People Fully Vaccinated in Country Statistical Analysis

df_vaccinations['people_fully_vaccinated'].describe() *Output:*

```
count    7.400000e+01
mean     3.046394e+07
std      8.904082e+07
min      2.607400e+04
25%      1.296678e+06
50%      5.586156e+06
75%      2.915660e+07
max      7.247684e+08
Name: people_fully_vaccinated, dtype: float64
```

Most Used Vaccine in the World

df_manufacturer['vaccine'].value_co
unts() *Output:*

```
Pfizer/BioNTech       39
Moderna               35
Johnson&Johnson       33
Oxford/AstraZeneca    30
Novavax                8
Sinovac                6
Sinopharm/Beijing      4
CanSino                2
Sputnik V              2
Covaxin                1
Name: vaccine, dtype: int64
```

Daily Vaccinations per Million Top Countries

df_vaccinations.groupby("country")["daily_vaccinations_per_million"].mean().sort
_values(asce nding=False).head(20)

Output:

```
country
Finland          9154.0
New Zealand      8800.0
Australia        8621.0
Italy            6733.0
Peru             6609.0
Saudi Arabia     6330.0
Iran             6280.0
Argentina        5814.0
Canada           5165.0
Brazil           4864.0
Pakistan         4841.0
India            4281.0
Greece           4055.0
Singapore        3951.0
Luxembourg       3845.0
Mexico           3792.0
Thailand         3718.0
South Korea      3447.0
Kenya            3315.0
Hong Kong        3293.0
Name: daily_vaccinations_per_million, dtype: float64
```

Preferred Vaccine in India x = df_vaccinations[df_vaccinations["country"] == "India"] z = x.vaccines.value_counts() c = list(z.index) print(c) *Output:*

```
['Covaxin, Oxford/AstraZeneca, Sputnik V']
```