```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline


# pd.set_option('display.max_rows', 500)
pd.set_option('display.max_columns', 30)
pd.set_option('display.float_format', '{:,.2f}'.format)



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

#data is from kaggle : https://www.kaggle.com/gpreda/covid-world-vaccination-progress


#Display first 5 rows

df_vaccination.head()
```

	country	iso_code	date	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_vaccinations_raw	daily_vaccination
0	Afghanistan	AFG	2021-02-22	0.00	0.00	NaN	NaN	Nal
1	Afghanistan	AFG	2021-02-23	NaN	NaN	NaN	NaN	1,367.0
2	Afghanistan	AFG	2021-02-24	NaN	NaN	NaN	NaN	1,367.0
3	Afghanistan	AFG	2021-02-25	NaN	NaN	NaN	NaN	1,367.0
4	Afghanistan	AFG	2021-02-26	NaN	NaN	NaN	NaN	1,367.0

```
df_vaccination.info()
```

```
<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
```

```
#Find the number or rows and columns

df_vaccination.shape

#There are 76095 rows and 15 columns
```

```
(86512, 15)
```

```
df_vaccination.isnull().sum()

#There are no empty rows for country, iso_code or date columns.
```

```
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
```

```
# General Overview of the calculations in data

df_vaccination.describe()
```

	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_vaccinations_raw	daily_vaccinations	total_vaccinations_pe
count	43,607.00	41,294.00	38,802.00	35,362.00	86,213.00	
mean	45,929,644.64	17,705,077.79	14,138,299.85	270,599.58	131,305.49	
std	224,600,360.18	70,787,311.50	57,139,201.72	1,212,426.60	768,238.77	
min	0.00	0.00	1.00	0.00	0.00	
25%	526,410.00	349,464.25	243,962.25	4,668.00	900.00	
50%	3,590,096.00	2,187,310.50	1,722,140.50	25,309.00	7,343.00	
75%	17,012,303.50	9,152,519.75	7,559,869.50	123,492.50	44,098.00	
max	3,263,129,000.00	1,275,541,000.00	1,240,777,000.00	24,741,000.00	22,424,286.00	

```
#drop the source_name,source_website and vaccine columns

df_vaccine_country = df_vaccination.drop(['source_name','source_website','vaccines'],axis=1)
df_vaccine_country.head()
```

	country	iso_code	date	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_vaccinations_raw	daily_vaccination
0	Afghanistan	AFG	2021-02-22	0.00	0.00	NaN	NaN	Nal
1	Afghanistan	AFG	2021-02-23	NaN	NaN	NaN	NaN	1,367.0
2	Afghanistan	AFG	2021-02-24	NaN	NaN	NaN	NaN	1,367.0
3	Afghanistan	AFG	2021-02-25	NaN	NaN	NaN	NaN	1,367.0
4	Afghanistan	AFG	2021-02-26	NaN	NaN	NaN	NaN	1,367.0

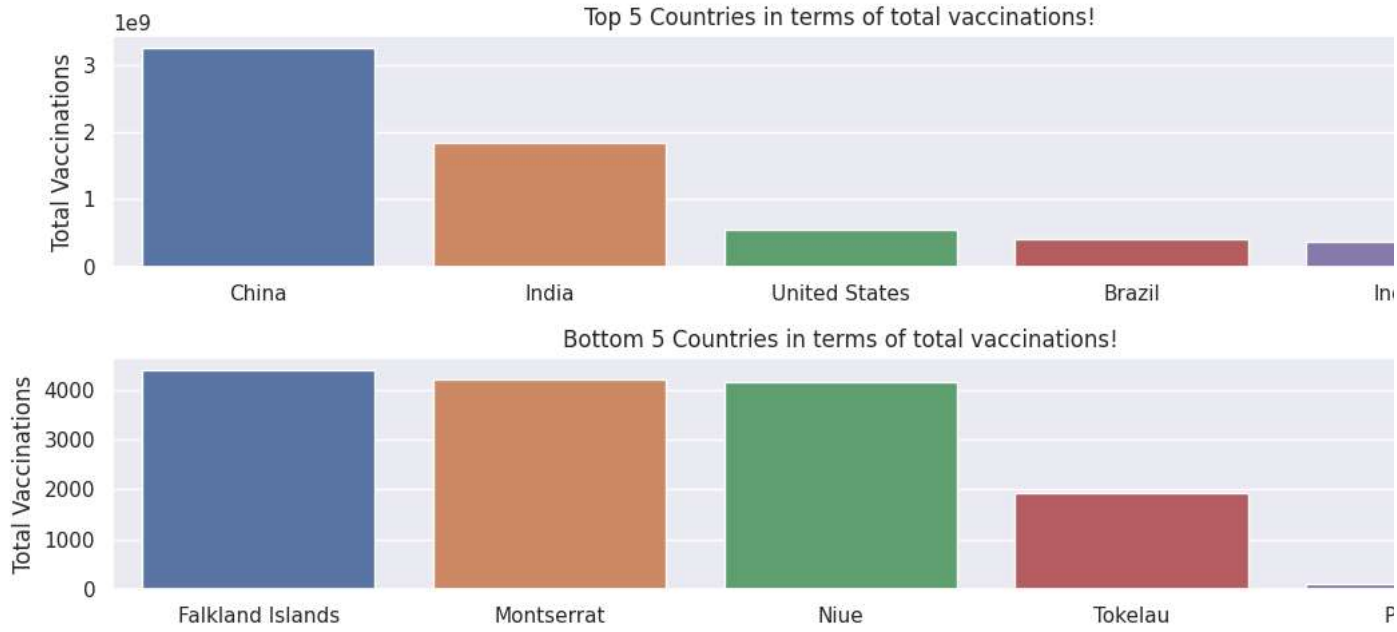
```
# convert Date column to date type and fill na values with 0 for calculation

df_vaccine_country["date"] = pd.to_datetime(df_vaccine_country["date"], format = '%Y-%m-%d')

df_vaccine_country = df_vaccine_country.replace([np.inf, -np.inf], np.nan)
df_vaccine_country = df_vaccine_country.fillna(0)
df_vaccine_country.isnull().sum()
```

```
country                            0
iso_code                           0
date                               0
total_vaccinations                 0
people_vaccinated                  0
```

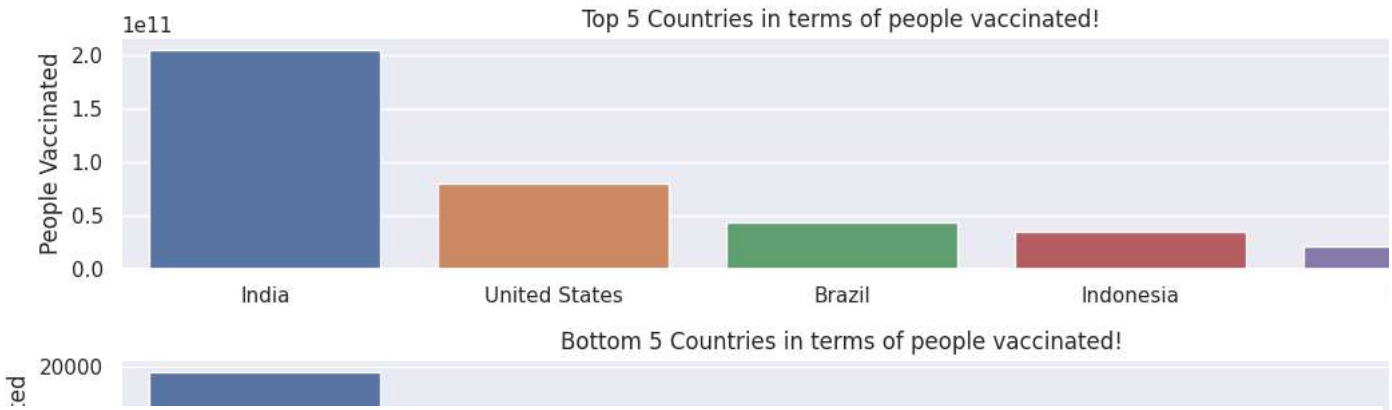
```
people_fully_vaccinated               0
daily_vaccinations_raw                0
daily_vaccinations                    0
total_vaccinations_per_hundred        0
people_vaccinated_per_hundred         0
people_fully_vaccinated_per_hundred   0
daily_vaccinations_per_million        0
dtype: int64
```

```
#Function to find total, avergae, maximum  and minimum of different vaccinations status by country
def vaccination_country(col_name,func_name):

    '''
    Function that requires vaccination column name, and sum/mean/max/min function name as string arguments.
    '''
    if func_name == 'sum':

        return  (df_vaccine_country[['country',col_name]].groupby(by='country')
                               .sum()
                               .sort_values(by=col_name,ascending= False)
                               .reset_index()
                           )

    elif func_name == 'mean':

        return  (df_vaccine_country[['country',col_name]].groupby(by='country')
                               .mean()
                               .sort_values(by=col_name,ascending= False)
                               .reset_index()
                              )
    elif func_name == 'max':

        return  (df_vaccine_country[['country',col_name]].groupby(by='country')
                               .max()
                               .sort_values(by=col_name,ascending= False)
                               .reset_index()
                              )
    elif func_name == 'min':

        return  (df_vaccine_country[['country',col_name]].groupby(by='country')
                               .min()
                               .sort_values(by=col_name,ascending= False)
                               .reset_index()
                              )


# Calculating different vaccinations for visualizations
max_total_vaccinations = vaccination_country('total_vaccinations','max')

sum_people_vaccinated = vaccination_country('people_vaccinated','sum')
sum_people_fully_vaccinated = vaccination_country('people_fully_vaccinated','sum')

avg_total_vaccinations = vaccination_country('total_vaccinations_per_hundred','mean')
avg_people_vaccinated = vaccination_country('people_vaccinated_per_hundred','mean')
avg_people_fully_vaccinated = vaccination_country('people_fully_vaccinated_per_hundred','mean')
avg_daily_vaccinations = vaccination_country('daily_vaccinations_per_million','mean')


#Function for Country with maximum and minimum daily vaccinations
def daily_vaccination_country(col_name,func_name):

    '''
    A function that requires daily_vaccination column and max/min function name as string arguments.
    '''

    daily_vaccination = (df_vaccine_country
                               .pivot_table(index='country',columns='date',values=col_name)
                                  )
    if func_name == 'max':

        daily_vaccination['Highest Daily Vaccination'] = daily_vaccination.max(axis=1)
        daily_vaccination['Date - Highest Daily Vaccination'] = daily_vaccination.idxmax(axis=1)
        daily_vaccination.sort_values(by='Highest Daily Vaccination',ascending=False,inplace=True)
        daily_vaccination.rename_axis('',axis=1,inplace=True)

        return daily_vaccination[['Highest Daily Vaccination','Date - Highest Daily Vaccination']].reset_index()

    elif func_name == 'min':
```

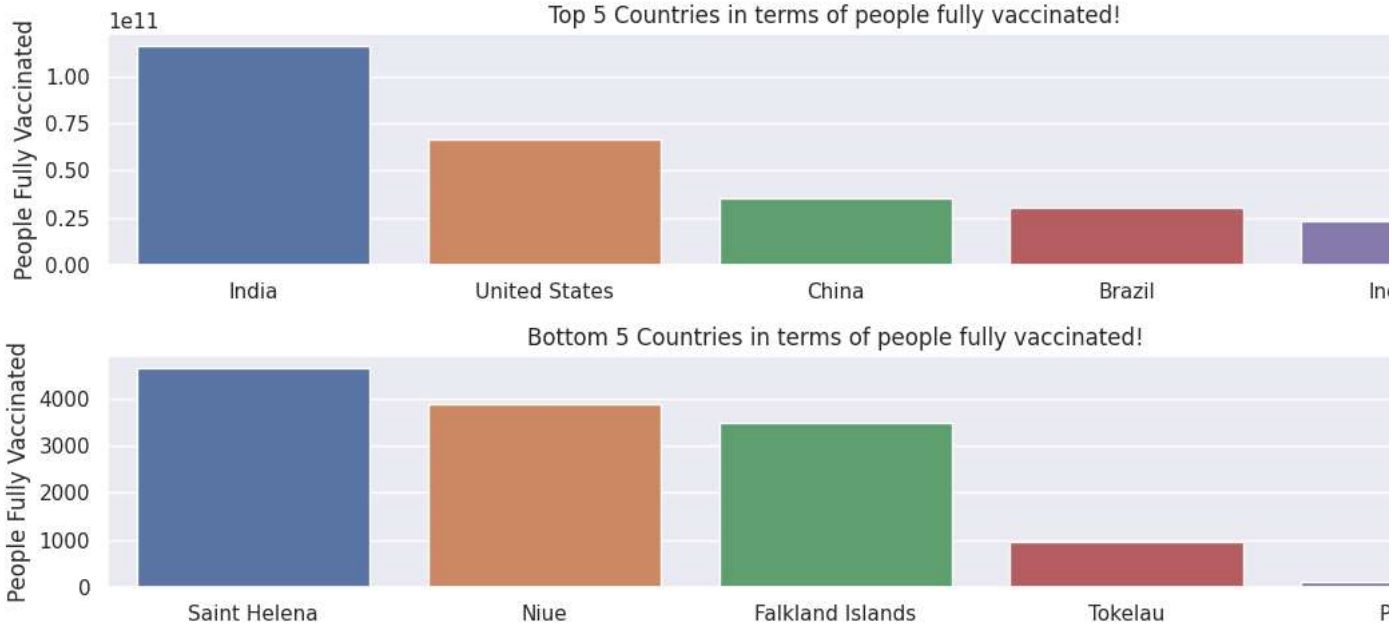
```
        daily_vaccination.replace(0.00,np.nan,inplace=True)
        daily_vaccination['Lowest Daily Vaccination'] = daily_vaccination.min(axis=1)
        daily_vaccination['Date - Lowest Daily Vaccination'] = daily_vaccination.idxmin(axis=1)
        daily_vaccination.sort_values(by='Lowest Daily Vaccination',ascending=False,inplace=True)
        daily_vaccination.rename_axis('',axis=1,inplace=True)

        return daily_vaccination[['Lowest Daily Vaccination','Date - Lowest Daily Vaccination']].reset_index()


#Calculating highest and lowest daily vaccination and the respective dates.
highest_daily_vaccination = daily_vaccination_country('daily_vaccinations','max')
lowest_daily_vaccination = daily_vaccination_country('daily_vaccinations','min')


#Set sns theme and default figsize for all the sns visualizations.
sns.set_theme(style='whitegrid')
sns.set(rc={'figure.figsize' : (12,5)})

fig, axes  = plt.subplots(2,1)

sns.barplot(x='country',y='total_vaccinations',data=max_total_vaccinations.head(),ax=axes[0])
axes[0].set(xlabel = '', ylabel = 'Total Vaccinations', title ='Top 5 Countries in terms of total vaccinations!')

sns.barplot(x='country',y='total_vaccinations',data=max_total_vaccinations.tail(),ax=axes[1])
axes[1].set(xlabel = '', ylabel = 'Total Vaccinations', title ='Bottom 5 Countries in terms of total vaccinations!')


fig.tight_layout()
plt.show()
```



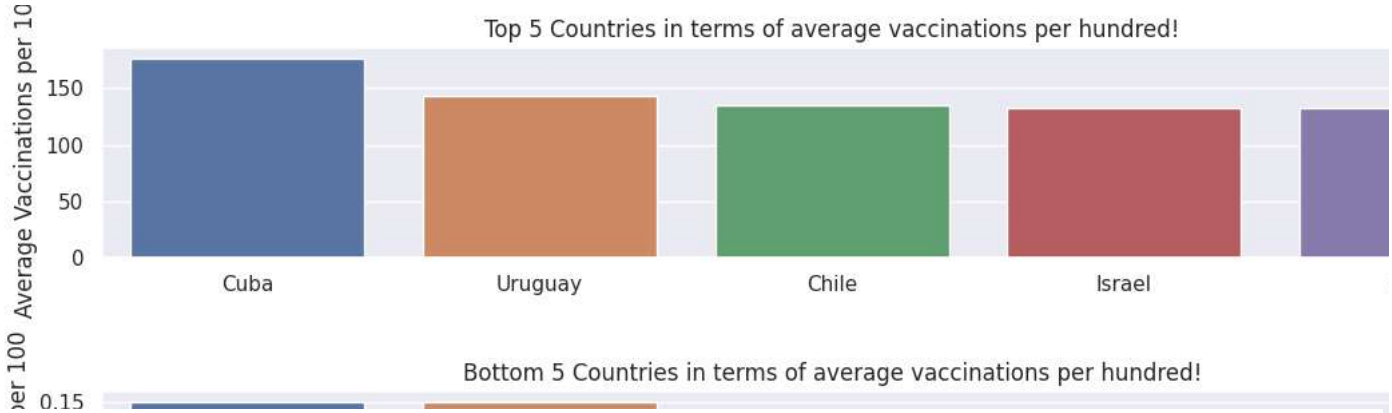

```
fig, axes = plt.subplots(2,1)

sns.barplot(x='country',y='people_vaccinated',data=sum_people_vaccinated.head(),ax=axes[0])
axes[0].set(xlabel = '', ylabel = 'People Vaccinated', title ='Top 5 Countries in terms of people vaccinated!')

sns.barplot(x='country', y='people_vaccinated',data=sum_people_vaccinated.tail(),ax=axes[1])
axes[1].set(xlabel = '', ylabel = 'People Vaccinated', title ='Bottom 5 Countries in terms of people vaccinated!')


fig.tight_layout()
plt.show()
```



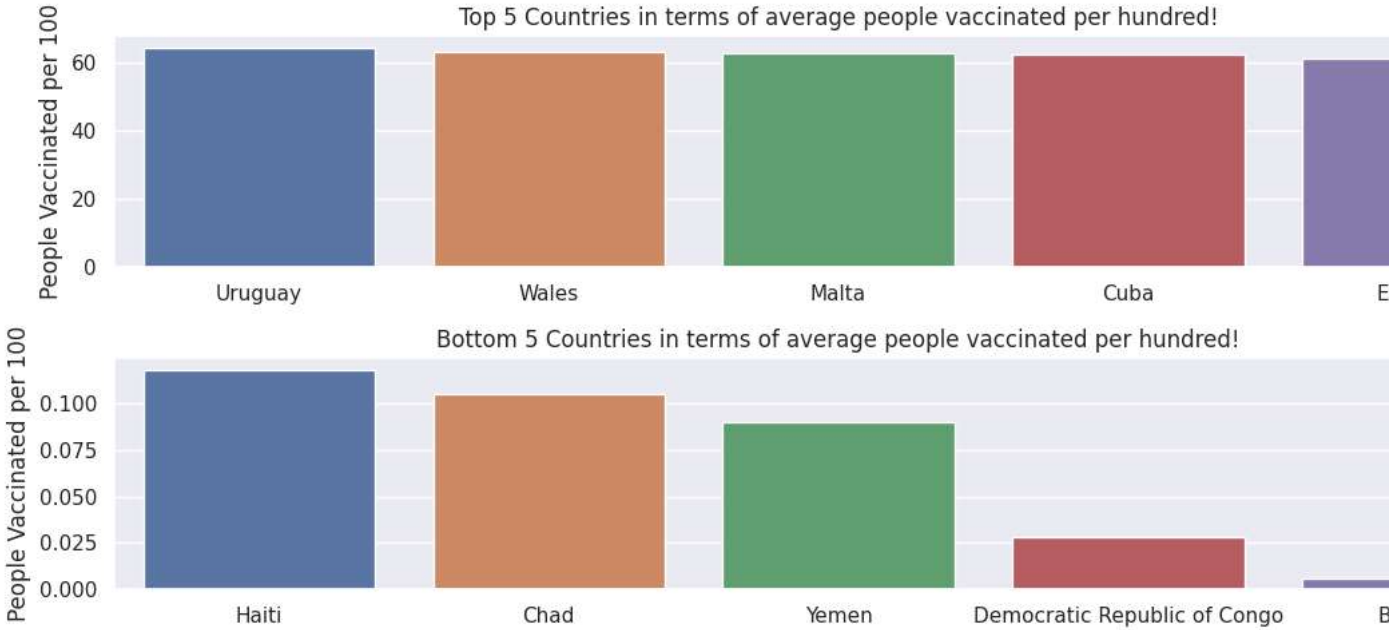

```
fig, axes = plt.subplots(2,1)

sns.barplot(x='country',y='people_fully_vaccinated',data=sum_people_fully_vaccinated.head(),ax=axes[0])
axes[0].set(xlabel = '', ylabel = 'People Fully Vaccinated', title ='Top 5 Countries in terms of people fully vaccinated!')

sns.barplot(x='country',y='people_fully_vaccinated',data=sum_people_fully_vaccinated.tail(),ax=axes[1])
axes[1].set(xlabel = '', ylabel = 'People Fully Vaccinated', title ='Bottom 5 Countries in terms of people fully vaccinated!')


# plt.ticklabel_format(style='plain', axis='y') #Uncomment if y label needs to display accurate values

fig.tight_layout()
plt.show()
```



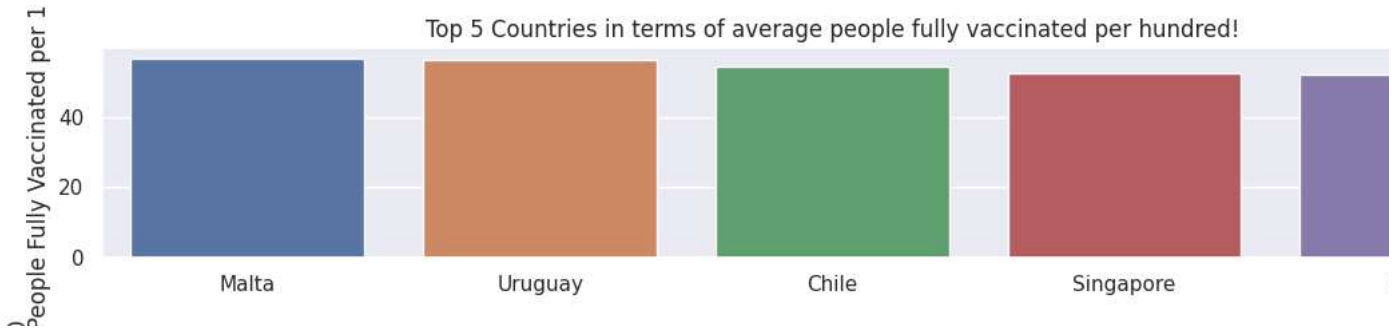

```
fig, axes = plt.subplots(2,1)

sns.barplot(x='country', y='total_vaccinations_per_hundred',data=avg_total_vaccinations.head(),ax=axes[0])
axes[0].set(xlabel='', ylabel='Average Vaccinations per 100', title='Top 5 Countries in terms of average vaccinations per hundred!')

sns.barplot(x='country', y='total_vaccinations_per_hundred',data=avg_total_vaccinations.tail(),ax=axes[1])
axes[1].set(xlabel='', ylabel='Average Vaccinations per 100', title='Bottom 5 Countries in terms of average vaccinations per hundred!')

fig.tight_layout(h_pad=3)
plt.show()
```



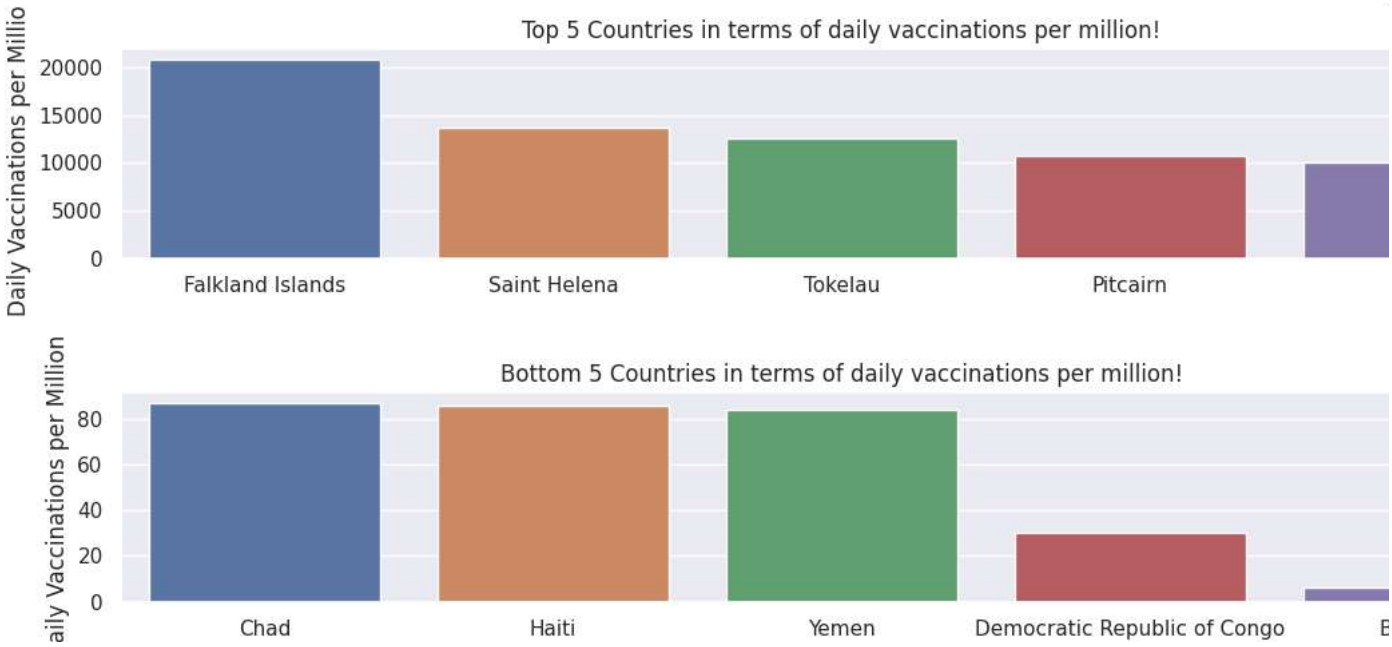

```
fig, axes = plt.subplots(2,1)

sns.barplot(x='country', y='people_vaccinated_per_hundred',data=avg_people_vaccinated.head(),ax=axes[0])
axes[0].set(xlabel='', ylabel='People Vaccinated per 100', title='Top 5 Countries in terms of average people vaccinated per hundred!')

sns.barplot(x='country', y='people_vaccinated_per_hundred',data=avg_people_vaccinated.tail(),ax=axes[1])
axes[1].set(xlabel='', ylabel='People Vaccinated per 100', title='Bottom 5 Countries in terms of average people vaccinated per hundred!')

fig.tight_layout()
plt.show()
```




```
fig, axes = plt.subplots(2,1)

sns.barplot(x='country', y='people_fully_vaccinated_per_hundred',data=avg_people_fully_vaccinated.head(),ax=axes[0])
axes[0].set(xlabel='', ylabel='People Fully Vaccinated per 100', title='Top 5 Countries in terms of average people fully vaccinated per hundr

sns.barplot(x='country', y='people_fully_vaccinated_per_hundred',data=avg_people_fully_vaccinated.tail(),ax=axes[1])
axes[1].set(xlabel='', ylabel='People Fully Vaccinated per 100', title='Bottom 5 Countries in terms of average people fully vaccinated per hu

fig.tight_layout(h_pad=3)
plt.show()
```




```
fig, axes = plt.subplots(2,1)

sns.barplot(x='country', y='daily_vaccinations_per_million',data=avg_daily_vaccinations.head(),ax=axes[0])
axes[0].set(xlabel='', ylabel='Daily Vaccinations per Million', title='Top 5 Countries in terms of daily vaccinations per million!')

sns.barplot(x='country', y='daily_vaccinations_per_million',data=avg_daily_vaccinations.tail(),ax=axes[1])
axes[1].set(xlabel='', ylabel='Daily Vaccinations per Million', title='Bottom 5 Countries in terms of daily vaccinations per million!')

fig.tight_layout(h_pad=3)
plt.show()
```




```
from plotly.offline import init_notebook_mode
import plotly.express as px
init_notebook_mode(connected=True)
```

```
#Top 5 country with highest total vaccinations
list(max_total_vaccinations['country'].head())
```

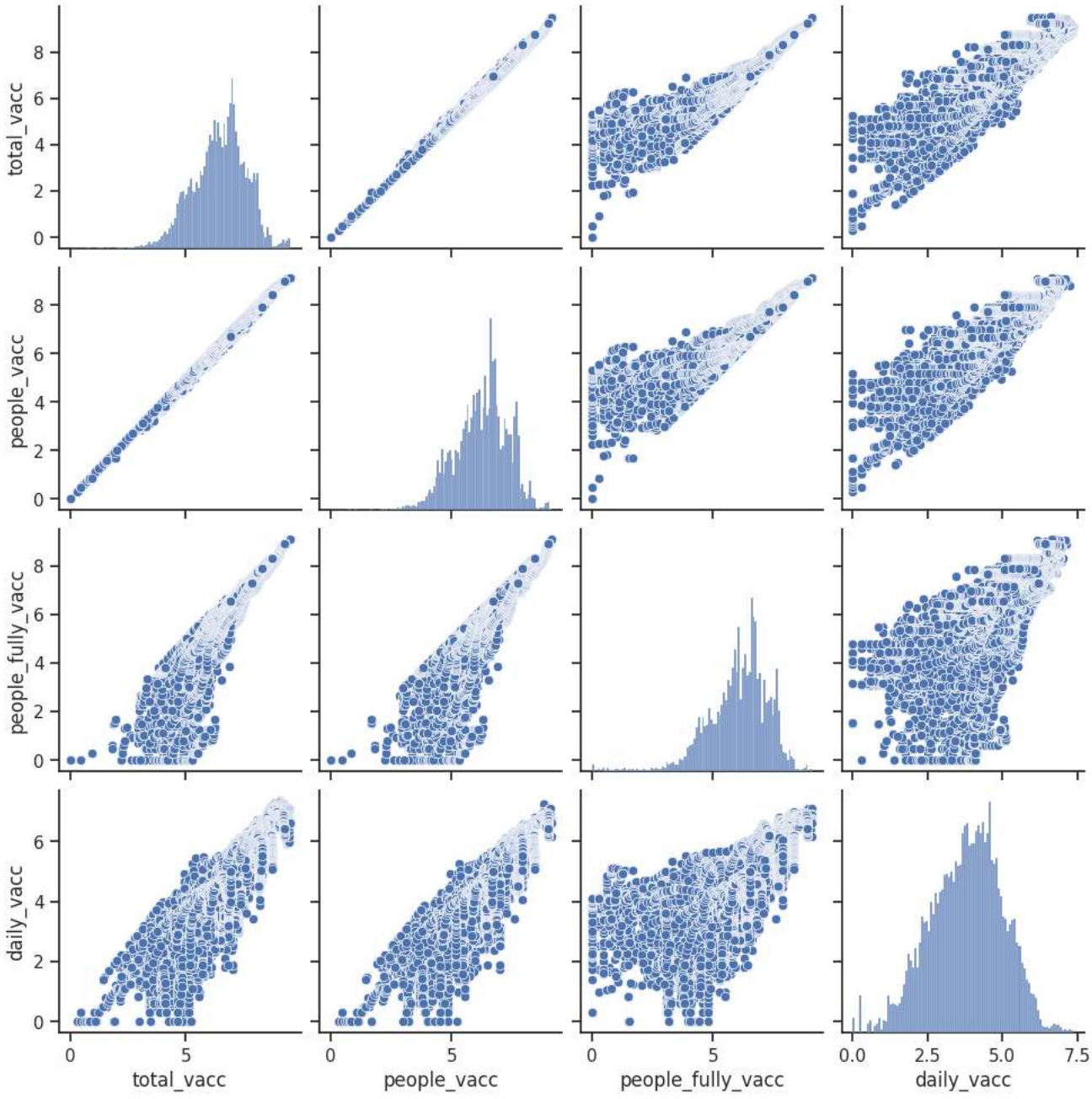
```
['China', 'India', 'United States', 'Brazil', 'Indonesia']
```

```
# Filter the top 5 countries and find their 30 day rolling average of total_vaccinations
top5_country_total = ['China', 'India', 'United States', 'Brazil', 'Indonesia']
top5_country_total_day = df_vaccine_country[df_vaccine_country['country'].isin(top5_country_total)].copy()
top5_country_total_day['30 - Day Rolling'] = top5_country_total_day['total_vaccinations'].rolling(window=30).mean()
```

```
#Plotting scatterplot matrix using Seaborn
#create dataframe with important features.
df_vaccination['total_vacc'] = np.log10(df_vaccination['total_vaccinations'])
df_vaccination['people_vacc'] = np.log10(df_vaccination['people_vaccinated'])
df_vaccination['people_fully_vacc'] = np.log10(df_vaccination['people_fully_vaccinated'])
df_vaccination['daily_vacc'] = np.log10(df_vaccination['daily_vaccinations'])

#drop the original nontransformed columns
df_vaccination = df_vaccination.drop(columns = ['total_vaccinations','people_vaccinated','people_fully_vaccinated', 'daily_vaccinations'])
```

```
covid_features = df_vaccination[['date', 'total_vacc', 'people_vacc' , 'people_fully_vacc' , 'daily_vacc']]
sns.set_theme(style="ticks")
sns.pairplot(covid_features)
```

```
/usr/local/lib/python3.10/dist-packages/pandas/core/arraylike.py:402: RuntimeWarning:

divide by zero encountered in log10

/usr/local/lib/python3.10/dist-packages/pandas/core/arraylike.py:402: RuntimeWarning:

divide by zero encountered in log10

/usr/local/lib/python3.10/dist-packages/pandas/core/arraylike.py:402: RuntimeWarning:

divide by zero encountered in log10

<seaborn.axisgrid.PairGrid at 0x794c64bc71c0>
```




```
vaccine_records=pd.read_csv('country_vaccinations.csv')
vaccine_records.head()
```

	country	iso_code	date	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_vaccinations_raw	daily_vaccination
0	Afghanistan	AFG	2021-02-22	0.00	0.00	NaN	NaN	Nal
1	Afghanistan	AFG	2021-02-23	NaN	NaN	NaN	NaN	1,367.0
2	Afghanistan	AFG	2021-02-24	NaN	NaN	NaN	NaN	1,367.0
3	Afghanistan	AFG	2021-02-25	NaN	NaN	NaN	NaN	1,367.0
			2021-					)

```
vaccine_records.shape
```

```
(86512, 15)
```

```
vaccine_records.describe()
```

	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_vaccinations_raw	daily_vaccinations	total_vaccinations_pe
count	43,607.00	41,294.00	38,802.00	35,362.00	86,213.00	
mean	45,929,644.64	17,705,077.79	14,138,299.85	270,599.58	131,305.49	
std	224,600,360.18	70,787,311.50	57,139,201.72	1,212,426.60	768,238.77	
min	0.00	0.00	1.00	0.00	0.00	
25%	526,410.00	349,464.25	243,962.25	4,668.00	900.00	
50%	3,590,096.00	2,187,310.50	1,722,140.50	25,309.00	7,343.00	
75%	17,012,303.50	9,152,519.75	7,559,869.50	123,492.50	44,098.00	
max	3,263,129,000.00	1,275,541,000.00	1,240,777,000.00	24,741,000.00	22,424,286.00	

```
vaccine_records['vaccines'].value_counts()
```

```
Johnson&Johnson, Moderna, Oxford/AstraZeneca, Pfizer/BioNTech                    7608
Moderna, Oxford/AstraZeneca, Pfizer/BioNTech                                     6263
Oxford/AstraZeneca                                                               6022
Oxford/AstraZeneca, Pfizer/BioNTech                                              4629
Johnson&Johnson, Moderna, Novavax, Oxford/AstraZeneca, Pfizer/BioNTech           3564
                                                                                 ...
Johnson&Johnson, Oxford/AstraZeneca, Sinovac                                      312
Moderna, Oxford/AstraZeneca, Pfizer/BioNTech, Sinovac, Sputnik V                  311
Johnson&Johnson, Moderna                                                          251
Johnson&Johnson, Pfizer/BioNTech, Sinopharm/Beijing                               228
EpiVacCorona, Oxford/AstraZeneca, QazVac, Sinopharm/Beijing, Sputnik V, ZF2001    190
Name: vaccines, Length: 84, dtype: int64
```

```
vaccine_records['country'].value_counts()
```

```
Norway                           482
Latvia                           480
Denmark                          476
United States                    471
Russia                           470
                                ...
Bonaire Sint Eustatius and Saba  146
Tokelau                          114
Saint Helena                      92
Pitcairn                          85
Falkland Islands                  67
Name: country, Length: 223, dtype: int64
```

```
#FINDING THE NULL ENTRIES#
vaccine_records.isnull().sum()
```

```
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
```

```
### Dropping missing values in data_frame ###

clean_data=vaccine_records.dropna()
clean_data.isnull().sum()
```

```
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
```

```
sub_records=clean_data[["country","date","total_vaccinations"]]
sub_records.head(10)
```

	country	date	total_vaccinations
94	Afghanistan	2021-05-27	593,313.00
101	Afghanistan	2021-06-03	630,305.00
339	Afghanistan	2022-01-27	5,081,064.00
433	Albania	2021-02-18	3,049.00
515	Albania	2021-05-11	622,507.00
516	Albania	2021-05-12	632,676.00
517	Albania	2021-05-13	638,338.00
518	Albania	2021-05-14	653,330.00
522	Albania	2021-05-18	688,947.00
523	Albania	2021-05-19	697,811.00

```
sub_entries=clean_data[["vaccines","date","total_vaccinations"]]
sub_entries.head(10)
```

	vaccines	date	total_vaccinations
94	Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi...	2021-05-27	593,313.00

```
sub_records.isnull().sum()
```

```
country              0
date                 0
total_vaccinations   0
dtype: int64
```

```
sub_entries.isnull().sum()
```

```
vaccines             0
date                 0
total_vaccinations   0
dtype: int64
```

523	Oxford/AstraZeneca, Pfizer/BioNTech, Sinovac, ...	2021-05-19	697,811.00

```
sub_records.shape
```

```
(30847, 3)
```

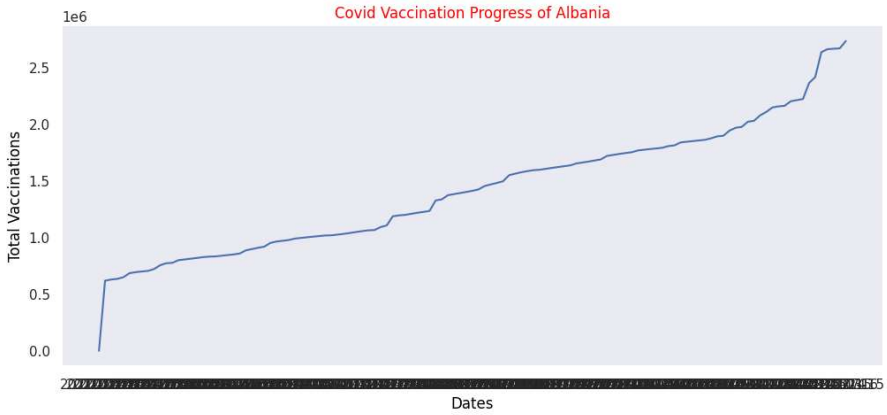
```
sub_entries.shape
```

```
(30847, 3)
```

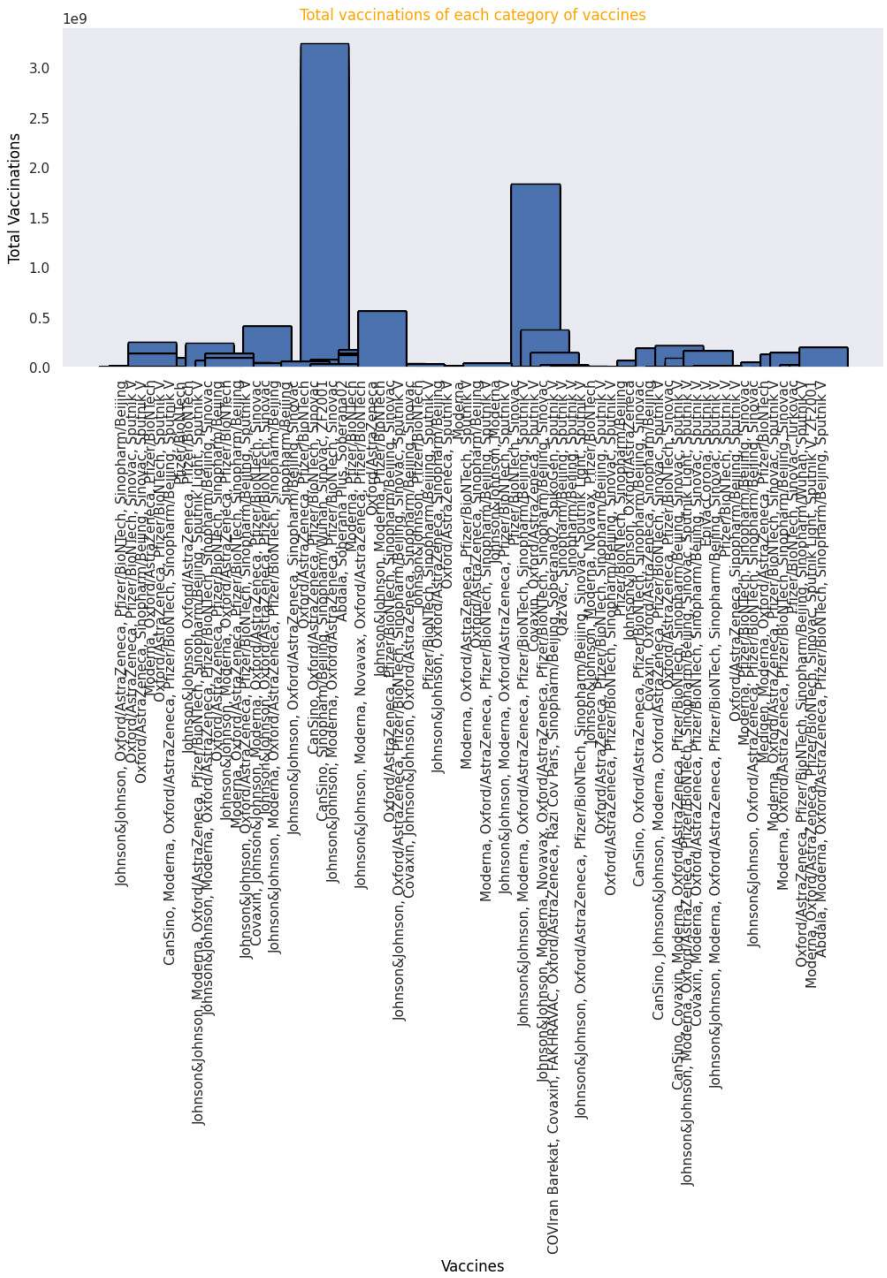
```
Albania_records=sub_records.loc[sub_records['country']=='Albania']
print(Albania_records)
```

```
     country        date  total_vaccinations
433  Albania  2021-02-18            3,049.00
515  Albania  2021-05-11          622,507.00
516  Albania  2021-05-12          632,676.00
517  Albania  2021-05-13          638,338.00
518  Albania  2021-05-14          653,330.00
..       ...         ...                 ...
787  Albania  2022-02-07        2,639,523.00
794  Albania  2022-02-14        2,665,804.00
795  Albania  2022-02-15        2,669,695.00
796  Albania  2022-02-16        2,673,183.00
823  Albania  2022-03-15        2,737,859.00

[123 rows x 3 columns]
```

```
plt.xlabel("Dates",color='black')
plt.ylabel("Total Vaccinations",color='black')
x=Albania_records["date"]
y=Albania_records["total_vaccinations"]
plt.title("Covid Vaccination Progress of Albania",color="red")
plt.plot(x,y)
plt.grid()
plt.show()
```




```
x=sub_entries["vaccines"]
y=sub_entries["total_vaccinations"]
plt.title("Total vaccinations of each category of vaccines",color="orange")
plt.xlabel("Vaccines",color="black")
plt.ylabel("Total Vaccinations",color="black")
plt.bar(x,y,width=5,edgecolor='black')
plt.xticks(rotation=90)
plt.grid()
plt.show()
```


Total vaccinations of each category of vaccines
1e9
Total Vaccinations
0.0
0.5
1.0
1.5
2.0
2.5
3.0
Vaccines