Extracting Information from Worldometer Website using Webscraping

In [81]:

```
# Import necessary packages

import requests
from bs4 import BeautifulSoup as bs
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import plotly
import plotly.graph_objects as go
import plotly.offline as pyo
from plotly.offline import init_notebook_mode
import plotly.express as px
%matplotlib inline
```

In [41]:

```
# URL of the webpage used for scraping

url = 'https://www.worldometers.info/coronavirus/?fbclid=IwAR35ZFiRZJ8tyBCwazX2N-k7yJjZOLDQiZSA_MsJAfdK74s8f2a_Dg
```

In [42]:

```
# Getting the response of the page and creating a soup object

response = requests.get(url)
soup = bs(response.text,'html.parser')
```

In [43]:

```
# Information in the website is stored as a table, below method is used to extract table information
table = soup.find('table',{'id':'main_table_countries_today'})
```

In [44]:

```
# Extracting header data

headers = []

for i in table.find_all('th'):
    title = i.text.replace('\n','').replace('\xa0','')
    headers.append(title)

# Creating a data frame with headers

df = pd.DataFrame(columns = headers)


# Extracting Table data

for row in table.find_all('tr')[1:]:
    if row.find_all('tr', class_='total_row_world row_continent'):
        pass
    else:
        data = row.find_all('td')
        raw_data = [td.text.strip().replace('\n','').replace(',','').replace('+','') for td in data]
        length = len(df)
        df.loc[length] = raw_data
```

In [45]:

```
df.head()
```

Out[45]:

	#	Country,Other	TotalCases	NewCases	TotalDeaths	NewDeaths	TotalRecovered	NewRecovered	ActiveCases	Serious,Critical	...	TotalTe
0		North America	45358843	16658	964271	933	36514266	16040	7880306	27951	...	
1		Asia	66566049	22346	974961	317	61922187	25259	3668901	40652	...	
2		South America	36372158		1114558		34123544		1134056	24422	...	
3		Europe	53554906	1741	1153235		48512396	1005	3889275	9826	...	
4		Africa	7392458		185867		6499829		706762	4856	...	

5 rows × 22 columns

Exploratory Data Analysis

Data Cleaning

In [46]:

```
# Dropping unnecessary columns
```

```
df.drop(df.columns[15:],axis=1,inplace=True)
```

In [47]:

```
df.head(7)
```

Out[47]:

	#	Country,Other	TotalCases	NewCases	TotalDeaths	NewDeaths	TotalRecovered	NewRecovered	ActiveCases	Serious,Critical	TotCases/1I po
0		North America	45358843	16658	964271	933	36514266	16040	7880306	27951	
1		Asia	66566049	22346	974961	317	61922187	25259	3668901	40652	
2		South America	36372158		1114558		34123544		1134056	24422	
3		Europe	53554906	1741	1153235		48512396	1005	3889275	9826	
4		Africa	7392458		185867		6499829		706762	4856	
5		Oceania	136005	677	1825	3	96326	338	37854	149	
6			721		15		706		0	0	

In [48]:

```
# Rename few column names

df.rename(columns={'Country,Other':'Country','TotalCases':'Total Cases','NewCases':'New Cases','TotalDeaths':'Tot
```

In [49]:

```
# The first few rows belong to total world and total continents. Will create a new dataset only for continents

continent_df = df[0:7]
```

In [50]:

```
continent_df
```

Out[50]:

	#	Country	Total Cases	New Cases	Total Deaths	New Deaths	Total Recovered	New Recovered	Active Cases	Serious,Critical	TotCases/1M pop	Deaths/1M pop	Total Tests	Tests/1M pop
0		North America	45358843	16658	964271	933	36514266	16040	7880306	27951				
1		Asia	66566049	22346	974961	317	61922187	25259	3668901	40652				
2		South America	36372158		1114558		34123544		1134056	24422				
3		Europe	53554906	1741	1153235		48512396	1005	3889275	9826				
4		Africa	7392458		185867		6499829		706762	4856				
5		Oceania	136005	677	1825	3	96326	338	37854	149				
6			721		15		706		0	0				

In [51]:

```
# Drop first few rows from original dataset

df.drop(df.index[0:8],inplace=True)
```

In [52]:

```
# Set a index value

df.set_index('#',inplace=True)
```

In [53]:

```
df.head(2)
```

Out[53]:

	Country	Total Cases	New Cases	Total Deaths	New Deaths	Total Recovered	New Recovered	Active Cases	Serious,Critical	TotCases/1M pop	Deaths/1M pop	Total Tests	Tests/1M pop
#													
1	USA	37896582		640093		30289989		6966500	20862	113739	1921	559519820	1679289
2	India	32285101		432552		31478405		374144	8944	23139	310	496629524	355937

In [54]:

```
# All the numbers are stored as object data type, convert them into numeric

for labels in df:
    if labels!='Country':
        df[labels] = pd.to_numeric(df[labels],errors='coerce')
```

In [55]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 230 entries, 1 to
Data columns (total 14 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   Country           230 non-null    object
 1   Total Cases       230 non-null    int64  
 2   New Cases         12 non-null     float64
 3   Total Deaths      217 non-null    float64
 4   New Deaths        9 non-null      float64
 5   Total Recovered   229 non-null    float64
 6   New Recovered     10 non-null     float64
 7   Active Cases      229 non-null    float64
 8   Serious,Critical  158 non-null    float64
 9   TotCases/1M pop   221 non-null    float64
 10  Deaths/1M pop     208 non-null    float64
 11  Total Tests       210 non-null    float64
 12  Tests/1M pop      210 non-null    float64
 13  Population        220 non-null    float64
dtypes: float64(12), int64(1), object(1)
memory usage: 27.0+ KB
```

In [56]:
```python
df.drop(df.tail(8).index,inplace=True)
```

In [57]:
```python
# Store the dataset into csv file

df.to_csv(r'C:/Users/pc/OneDrive/Desktop/DaataScienceProjects/Covid-Datasets/corona.csv')
```

Exploring the Data

Finding information about cases, recoveries,deaths across the world

In [58]:
```python
# Finding Total Number of Deaths across the world

total_deaths = df['Total Deaths'].sum()
print(f'Total Deaths Across the world: {total_deaths}')

total_recovered = df['Total Recovered'].sum()
print(f'Total Recoveries Across the world: {total_recovered}')
```

```
Total Deaths Across the world: 4394732.0
Total Recoveries Across the world: 185948589.0
```

In [59]:
```python
# Percentage of total population infected with Covid

total_population = df['Population'].sum()
total_cases = df['Total Cases'].sum()
percentage_of_population_infected = (round((total_cases/total_population),4)*100)
```

In [60]:
```python
cases_dict = {
    'total_population':total_population,
    'total_cases':total_cases,
    'percentage_of_population_infected':percentage_of_population_infected
}

cases_df = pd.DataFrame(cases_dict,index=[0])
cases_df
```

Out[60]:

	total_population	total_cases	percentage_of_population_infected
0	7.844313e+09	209381140	2.67

In [61]:
```python
virus_dict = {
    'total_deaths': total_deaths,
    'total_recovered':total_recovered,
    'total_cases':total_cases
}
```

In [62]:
```python
virus_df = pd.DataFrame(virus_dict,index=[0])
virus_df
```

Out[62]:

	total_deaths	total_recovered	total_cases
0	4394732.0	185948589.0	209381140

Visualizing New Cases

In [64]:

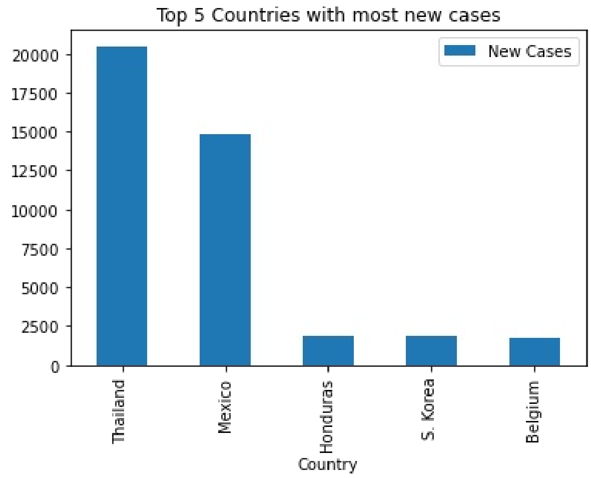
```
new_cases_df = df[df['New Cases'].notnull()]
```

In [65]:

```
case_df = new_cases_df[['Country','New Cases']].sort_values(by='New Cases',ascending=False)
```

In [66]:

```
# Top 5 countries most new cases

case_df[0:5].plot(kind='bar',x='Country',y='New Cases',title='Top 5 Countries with most new cases');
```



Visualizing Active Cases and Serious cases among them

In [67]:

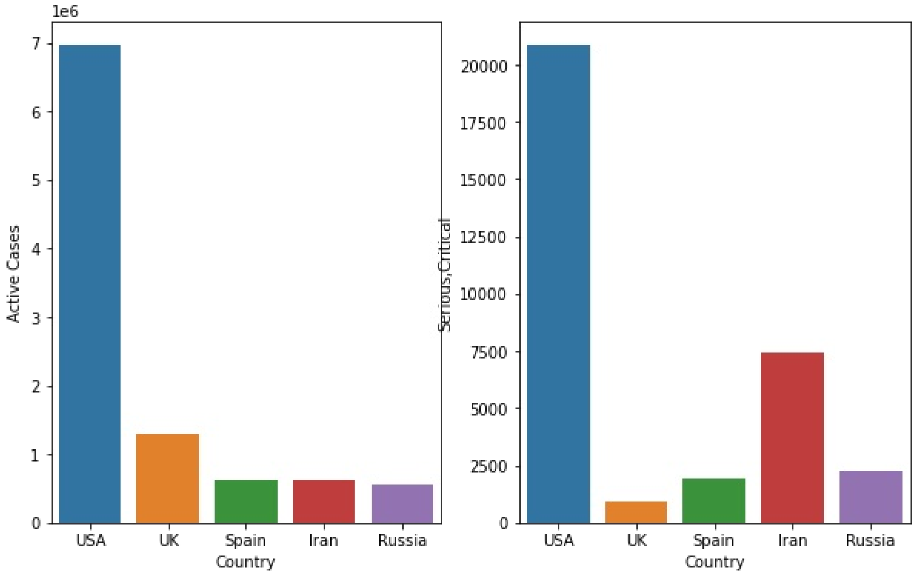
```
active_cases_df = df[['Country','Active Cases','Serious,Critical']]
```

In [68]:

```
top5_active_cases_df=active_cases_df.sort_values(by='Active Cases',ascending=False)[0:5]
```

In [69]:

```
plt.figure(figsize=(10,6))
plt.subplot(1,2,1)
sns.barplot(x='Country', y= 'Active Cases',data = top5_active_cases_df);
plt.subplot(1,2,2)
sns.barplot(x='Country', y= 'Serious,Critical',data = top5_active_cases_df);
```



Visualizing continent dataset

In [70]:

```
continent_df.set_index('#',inplace=True)
```

In [71]:

```
continent_df.rename(columns = {'Country':'Continent'},inplace=True)
```

```
C:\Users\pc\anaconda3\lib\site-packages\pandas\core\frame.py:4296: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#retur
ning-a-view-versus-a-copy
```

In [72]:

```
for labels in continent_df:
    if labels!='Continent':
        continent_df[labels] = pd.to_numeric(continent_df[labels])
```

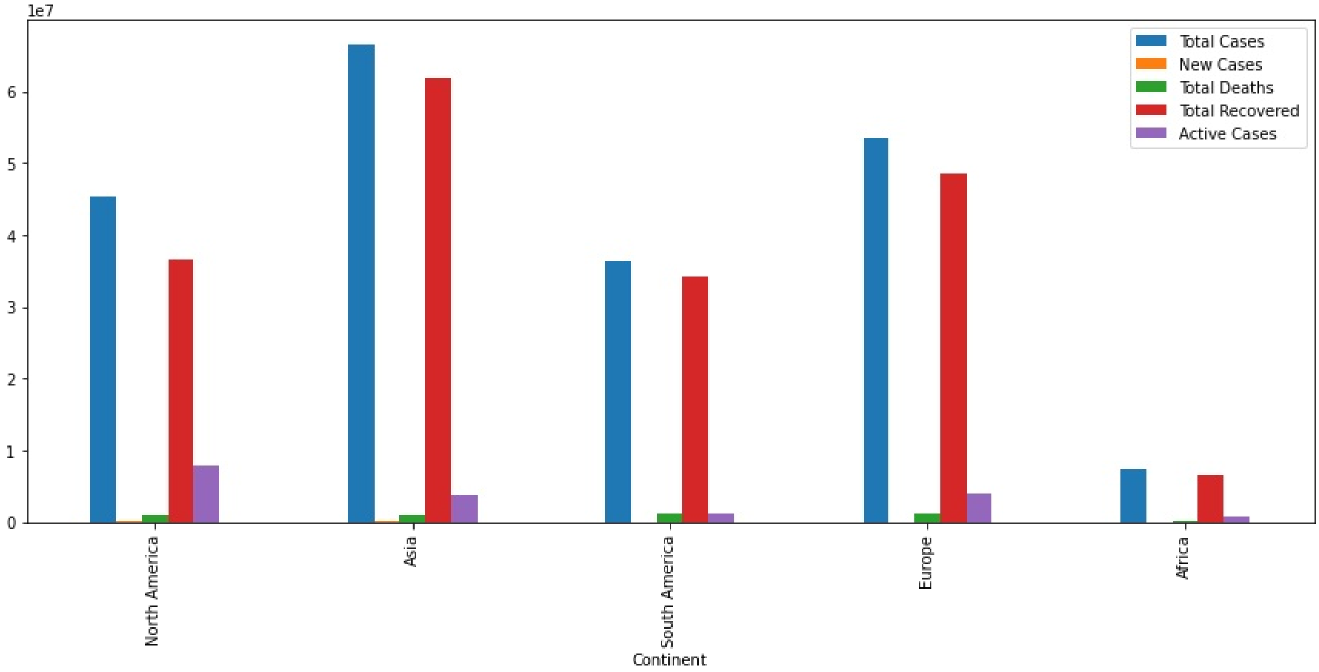
```
<ipython-input-72-3eca48db00fb>:3: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#retur
ning-a-view-versus-a-copy
```

In [73]:

```
new_continent_df = continent_df[['Continent','Total Cases','New Cases','Total Deaths','Total Recovered','Active C
```

In [74]:

```
new_continent_df[0:5].plot(kind='bar',x='Continent',figsize=(15,6));
```



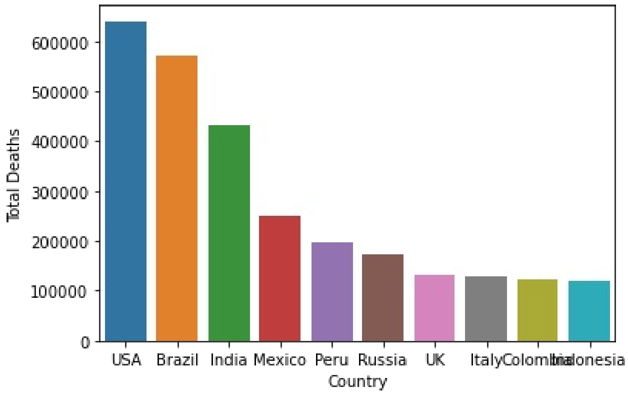
Top 10 countries with Most number of Deaths

In [75]:

```
# Find top 10 countries with most number of deaths

top10_df = df[['Country','Total Deaths']].sort_values(by='Total Deaths',ascending=False)[0:10]
```

In [76]:

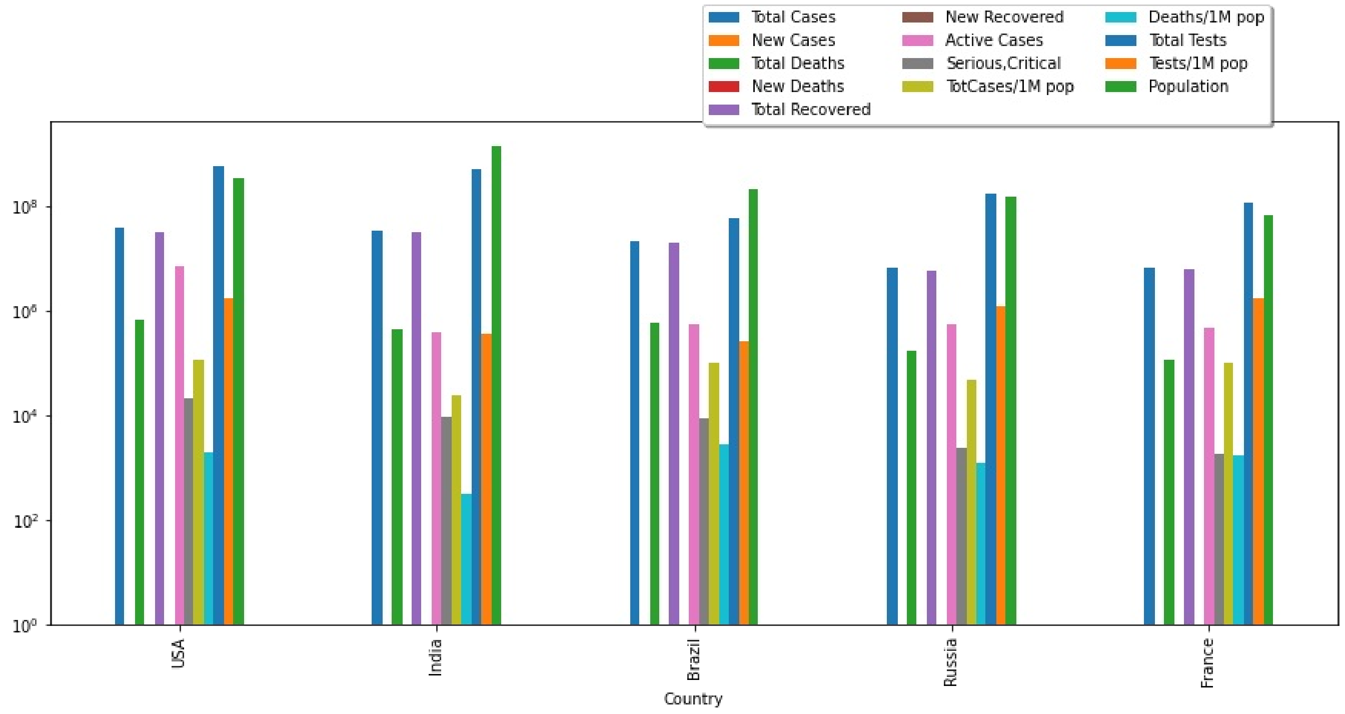
```
# Seaborn bar plot representing Country and Number of Deaths

sns.barplot(x='Country',y='Total Deaths',data=top10_df);
```



Top 5 Countries Most Affected

Top 5 Countries Most Affected

```
In [77]: df[0:5].plot(kind='bar',x='Country',figsize=(15,6),log=True);

         plt.legend(loc='upper left', bbox_to_anchor=(0.5, 1.25),
                   ncol=3, fancybox=True, shadow=True);
```



Inferences made from dataset and visualizations

1. Of the total population in the world, about 2.4% people are affected with covid.
2. Total cases were 189749829, Total Deaths recorded were 4083258 and Total Recovered people were 171445052
3. The top 5 countries which are most affected with the covid virus are USA,India,Brazil,Russia and France.
4. Comparing continents, Africa has the least recorded cases and deaths whereas Asia tops the list.
5. There has been a reduction in rise of new cases in many parts of the world. Although Mexico has recorded highest number of rise in new cases.
6. USA has highest number of active cases

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js