Data Prepatation

In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.graph_objects as go
import plotly.express as px
import os
import warnings
warnings.filterwarnings('ignore')
```

datasource: https://www.kaggle.com/datasets/imdevskp/corona-virus-report

In [2]:

```
files = os.listdir(r"C:\Users\rkuppi\OneDrive - DXC Production\Desktop\DataSets\04 Proje
```

In [3]:

```
files
```

Out[3]:

```
['country_wise_latest.csv',
 'covid_19_clean_complete.csv',
 'day_wise.csv',
 'full_grouped.csv',
 'usa_county_wise.csv',
 'worldometer_data.csv']
```

In [4]:

```
path = r"C:\Users\rkuppi\OneDrive - DXC Production\Desktop\DataSets\04 Project 3-- Covid
```

In [5]:

```
read_data = lambda path, filename: pd.read_csv(path + "/" + filename)
```

In [6]:

```
country_wise_data = read_data(path, "country_wise_latest.csv")
```

In [7]:

```
country_wise_data.head()
```

Out[7]:

	Country/Region	Confirmed	Deaths	Recovered	Active	New cases	New deaths	New recovered	Deaths / 100 Cases	Recovered / 100 Cases	Death 1 Recove
0	Afghanistan	36263	1269	25198	9796	106	10	18	3.50	69.49	5
1	Albania	4880	144	2745	1991	117	6	63	2.95	56.25	5
2	Algeria	27973	1163	18837	7973	616	8	749	4.16	67.34	6
3	Andorra	907	52	803	52	10	0	0	5.73	88.53	6
4	Angola	950	41	242	667	18	1	0	4.32	25.47	16

In [8]:

```
Covid_19_clean_complete = read_data(path, "covid_19_clean_complete.csv")
```

In [9]:

```
Covid_19_clean_complete.head()
```

Out[9]:

	Province/State	Country/Region	Lat	Long	Date	Confirmed	Deaths	Recovered	Active	WHO Re
0	NaN	Afghanistan	33.93911	67.709953	2020-01-22	0	0	0	0	Eas Mediterrar
1	NaN	Albania	41.15330	20.168300	2020-01-22	0	0	0	0	Eu

2	NaN	Algeria	28.03390	1.659600	2020-01-22	0	0	0	0	A
3	NaN	Andorra	42.50630	1.521800	2020-01-22	0	0	0	0	Eu
4	NaN	Angola	-11.20270	17.873900	2020-01-22	0	0	0	0	A

In [10]:

```
day_wise = read_data(path, "day_wise.csv")
```

In [11]:

```
day_wise.head()
```

Out[11]:

	Date	Confirmed	Deaths	Recovered	Active	New cases	New deaths	New recovered	Deaths / 100 Cases	Recovered / 100 Cases	Deaths / 100 Recovered	Nc count
0	2020-01-22	555	17	28	510	0	0	0	3.06	5.05	60.71	
1	2020-01-23	654	18	30	606	99	1	2	2.75	4.59	60.00	
2	2020-01-24	941	26	36	879	287	8	6	2.76	3.83	72.22	
3	2020-01-25	1434	42	39	1353	493	16	3	2.93	2.72	107.69	
4	2020-01-26	2118	56	52	2010	684	14	13	2.64	2.46	107.69	

In [12]:

```
fully_grouped_data = read_data(path, 'full_grouped.csv')
```

In [13]:

```
fully_grouped_data.head()
```

Out[13]:

	Date	Country/Region	Confirmed	Deaths	Recovered	Active	New cases	New deaths	New recovered	WHO Region
0	2020-01-22	Afghanistan	0	0	0	0	0	0	0	Eastern Mediterranean
1	2020-01-22	Albania	0	0	0	0	0	0	0	Europe
2	2020-01-22	Algeria	0	0	0	0	0	0	0	Africa
3	2020-01-22	Andorra	0	0	0	0	0	0	0	Europe
4	2020-01-22	Angola	0	0	0	0	0	0	0	Africa

In [14]:

```
worldmeter_data = read_data(path, 'worldometer_data.csv')
```

In [15]:

```
worldmeter_data.head()
```

Out[15]:

	Country/Region	Continent	Population	TotalCases	NewCases	TotalDeaths	NewDeaths	TotalRecovered	Ne
0	USA	North	3.311981e+08	5032179	NaN	162804.0	NaN	2576668.0	

		America						
1	Brazil	South America	2.127107e+08	2917562	NaN	98644.0	NaN	2047660.0
2	India	Asia	1.381345e+09	2025409	NaN	41638.0	NaN	1377384.0
3	Russia	Europe	1.459409e+08	871894	NaN	14606.0	NaN	676357.0
4	South Africa	Africa	5.938157e+07	538184	NaN	9604.0	NaN	387316.0

Out of all these data sets we use worldmeter_data as this consists of all the data that is required and new parameters can be derived

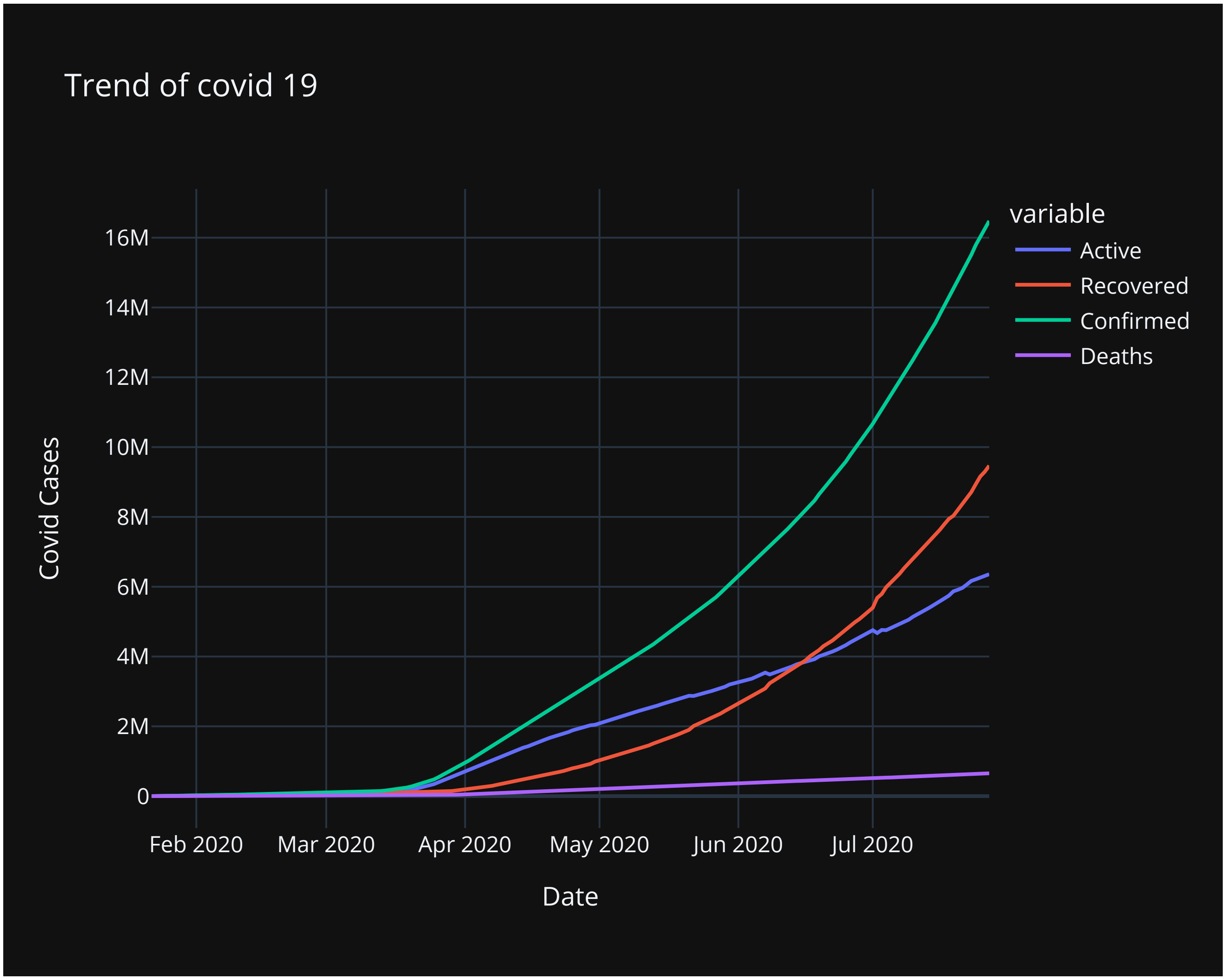
Trend of the data cases

In [16]:

```
import plotly.graph_objects as go
from plotly.offline import init_notebook_mode, iplot
init_notebook_mode(connected=True)
```

In [17]:

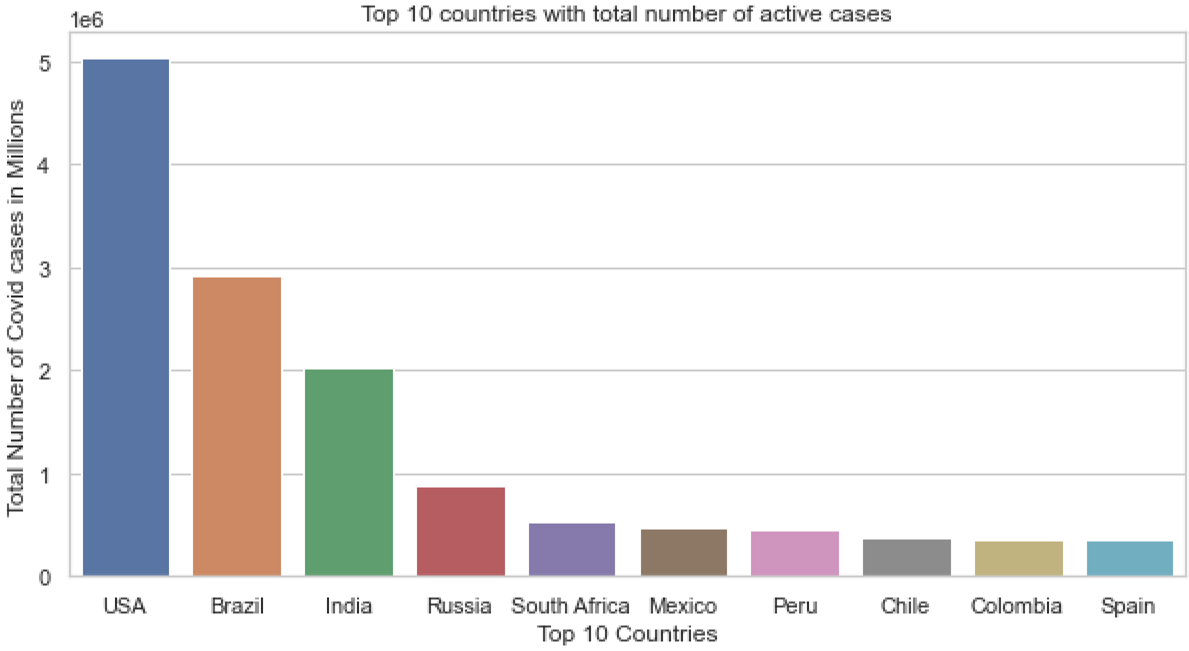
```
px.line(day_wise,x="Date",y=["Active", "Recovered","Confirmed","Deaths"],title="Trend of
        labels= {"value": "Covid Cases"}).show()
```



we will try to answer the top 10 countries with maximum Total cases

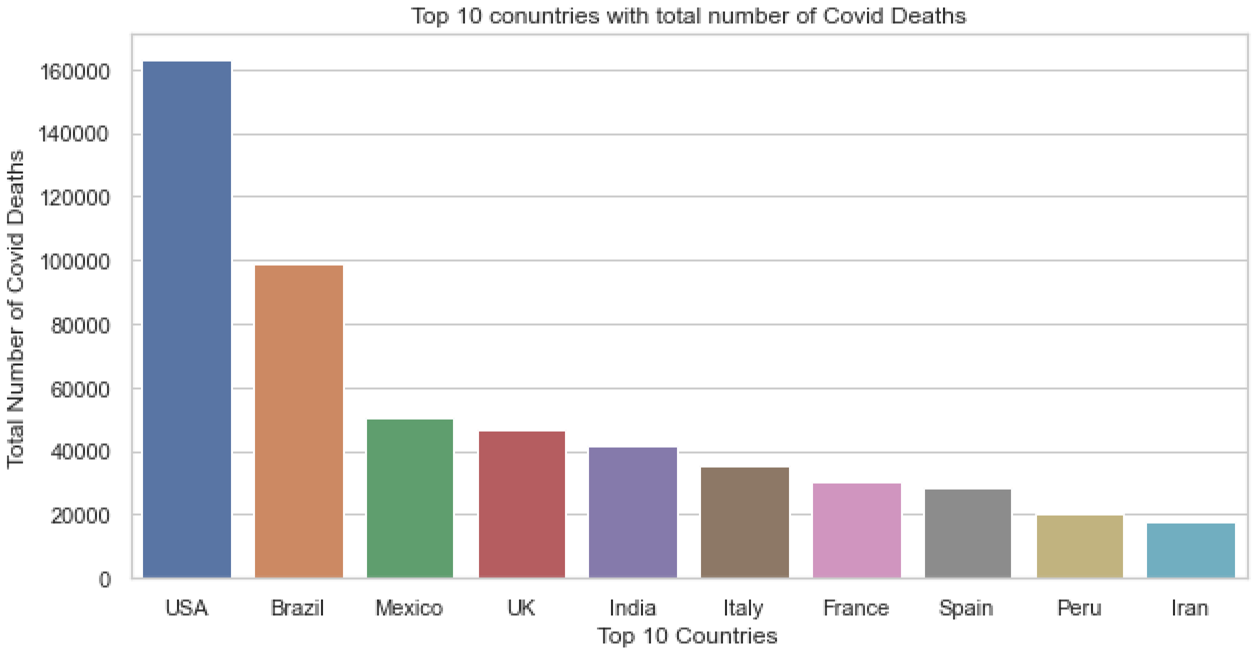
Total Covid cases

```
In [18]: #### We can get this data from worldmeter_data, which has Country/region. total populati
         sns.set(style="whitegrid")
         plt.figure(figsize=(10,5))
         sns.barplot(data = worldmeter_data.sort_values(by = "TotalCases", ascending = False).hea
             .set(xlabel='Top 10 Countries', ylabel='Total Number of Covid cases in Millions', ti
```



Total Deaths

```
In [19]: sns.set(style="whitegrid")
         plt.figure(figsize=(10,5))
         sns.barplot(data = worldmeter_data.sort_values(by= "TotalDeaths", ascending = False).hea
             .set(xlabel='Top 10 Countries', ylabel='Total Number of Covid Deaths', title = "Top
```

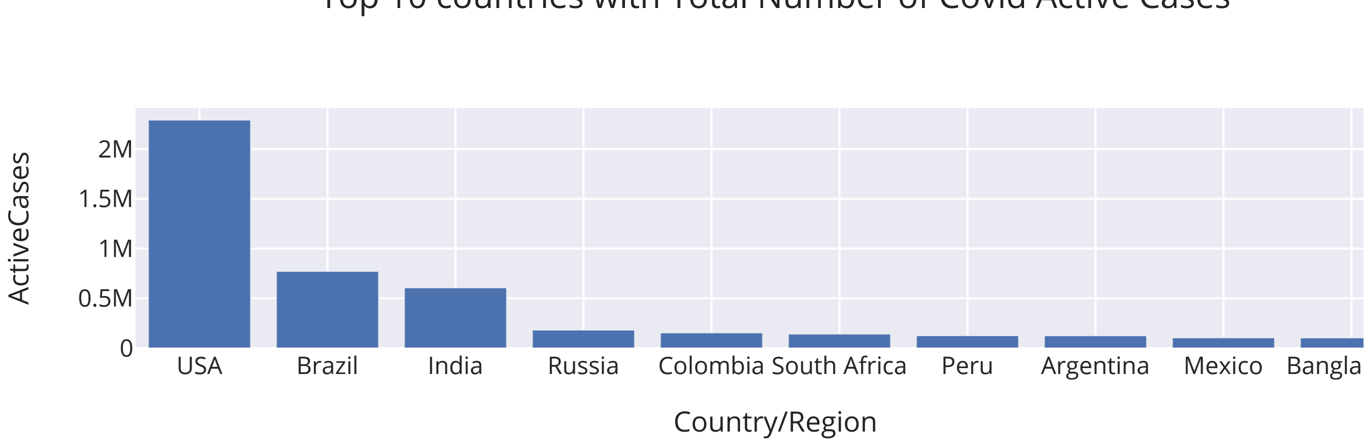


Total Active Cases

```
In [20]: Total_active_cases = px.bar(worldmeter_data.sort_values(by= "ActiveCases", ascending = F
                                     width=800, height=300)
         Total_active_cases.show()
```

Top 10 countries with Total Number of Covid Active Cases

Top 10 countries with Total Number of Covid Active Cases



In [21]:

```
# can we say the USA, Brazil, India, Russia are more effected contries by lokking at Tot
# No we are not lokking the percapita deta hence we cannot make make any colclusions abo
# Plotting the grapth to understand the senario, considering Tot Cases/1M pop, Deaths/1M
worldmeter_data.columns
```

Out[21]:

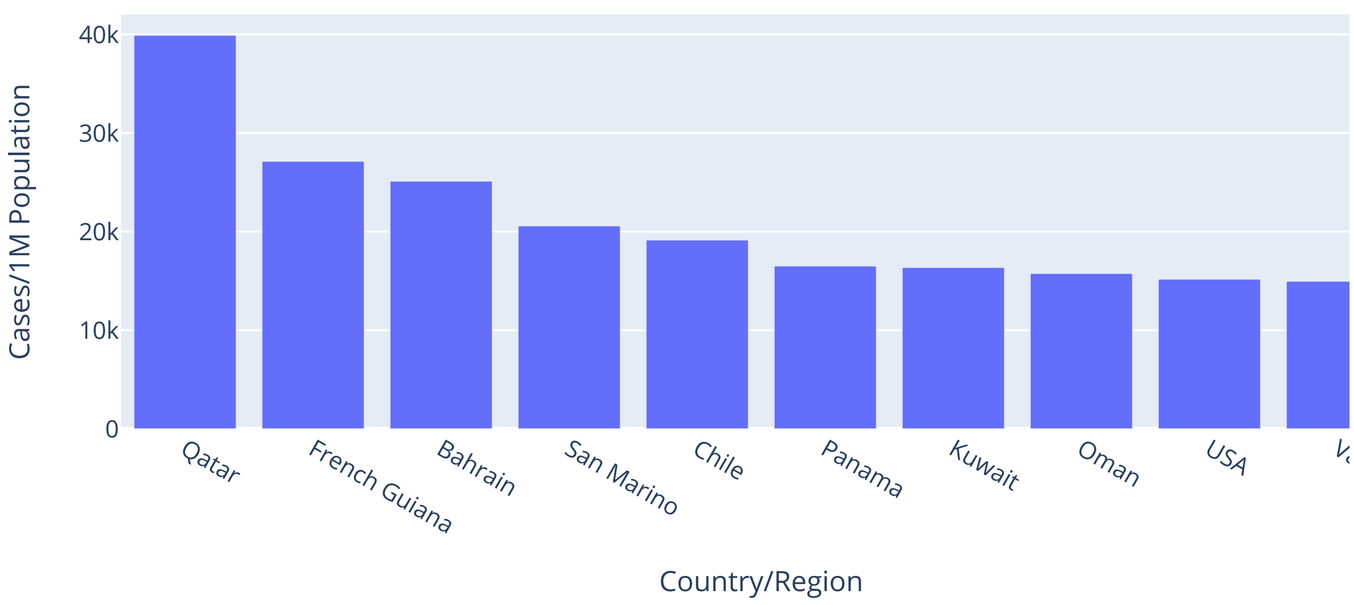
```
Index(['Country/Region', 'Continent', 'Population', 'TotalCases', 'NewCases',
       'TotalDeaths', 'NewDeaths', 'TotalRecovered', 'NewRecovered',
       'ActiveCases', 'Serious,Critical', 'Tot Cases/1M pop', 'Deaths/1M pop',
       'TotalTests', 'Tests/1M pop', 'WHO Region'],
      dtype='object')
```

Total cases/1M population

In [22]:

```
px.bar(worldmeter_data.sort_values(by= "Tot Cases/1M pop", ascending = False).head(10),
       labels={"Tot Cases/1M pop": "Cases/1M Population"}, height=400, width=800).show()
```

Top 10 countries with Total Number of Cases/1M population

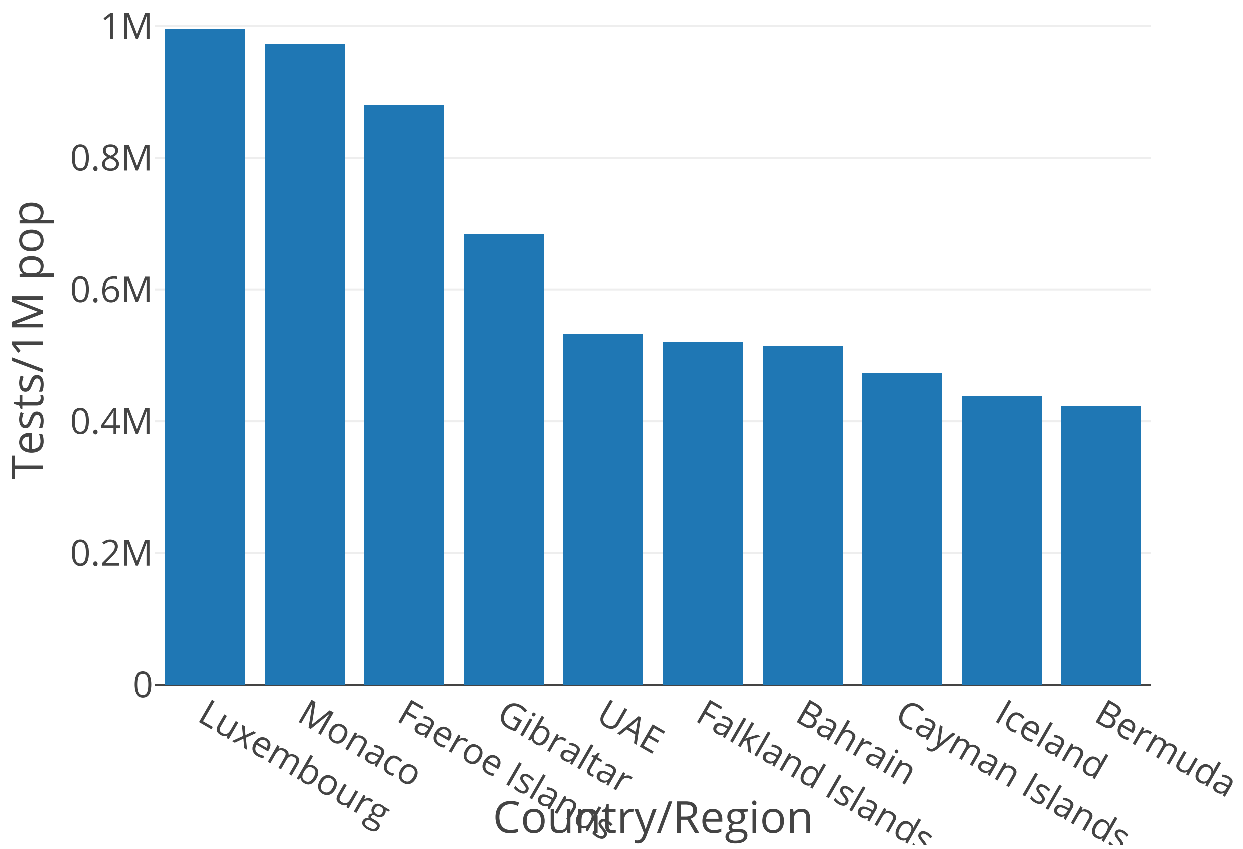


Even though the total number of cases in USA is very high, Qatar has more number of total covid cases when normalize the cases one total population

Deaths/1M pop

In [23]:
```
px.bar(worldmeter_data.sort_values(by= "Tests/1M pop", ascending = False).head(10), x =
```

Top 10 countries with maximum deaths/1Million



Even tough we have more number of covid cases in countries like USA, Brazil, India. Recovery was good, covid spread/Milliion is less and less number of deaths/Million are recorded

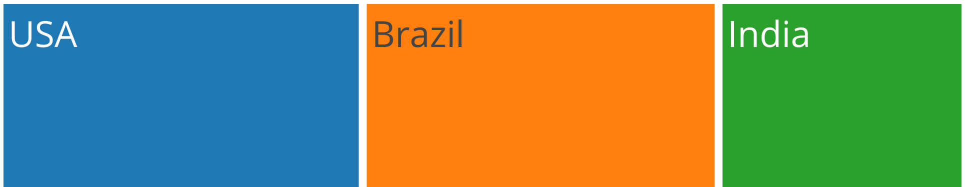
In [24]:
```
worldmeter_data.columns
```

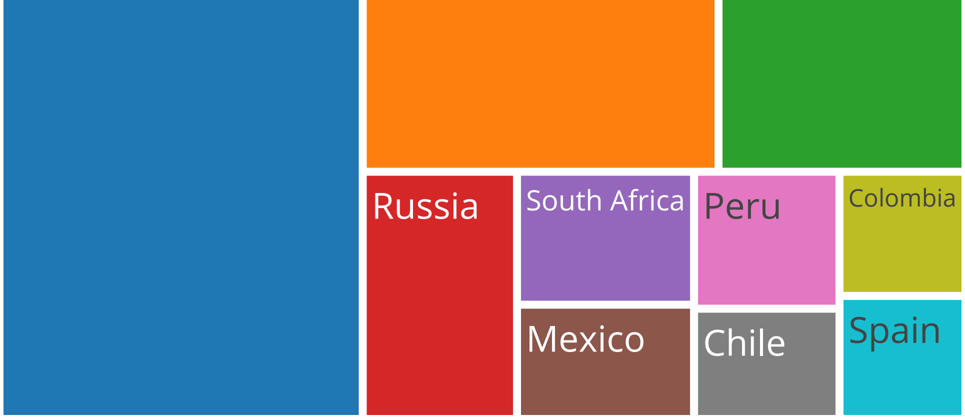
Out[24]:
```
Index(['Country/Region', 'Continent', 'Population', 'TotalCases', 'NewCases',
       'TotalDeaths', 'NewDeaths', 'TotalRecovered', 'NewRecovered',
       'ActiveCases', 'Serious,Critical', 'Tot Cases/1M pop', 'Deaths/1M pop',
       'TotalTests', 'Tests/1M pop', 'WHO Region'],
      dtype='object')
```

We can also represent the data with treemap

In [25]:
```
px.treemap(worldmeter_data.head(10),values="TotalCases",path=['Country/Region'],template
```

Top 10 countries with maximum number of Covid cases



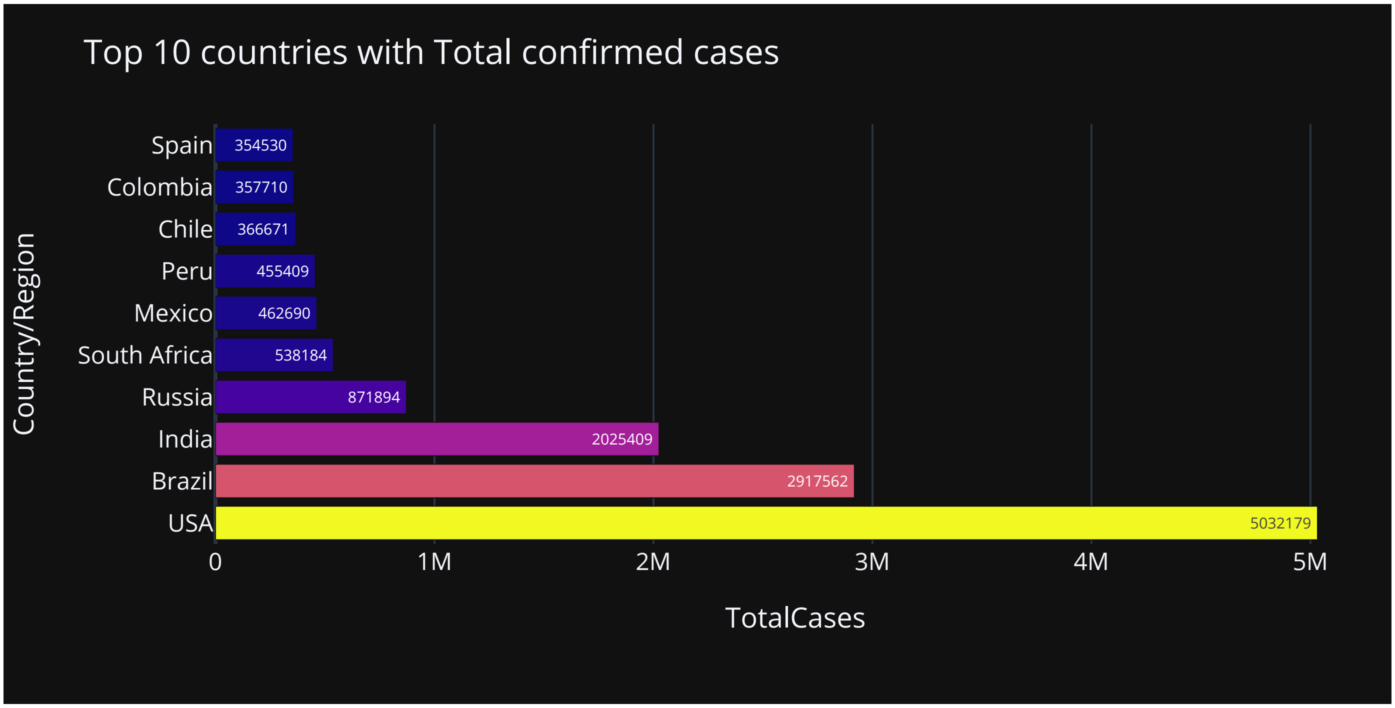


We can visualize the percentage of cases across all the countries with the above tree map

Top 10 countries with Total Confirmed Cases, Total Recovered Cases, Total Deaths,Total Active Cases

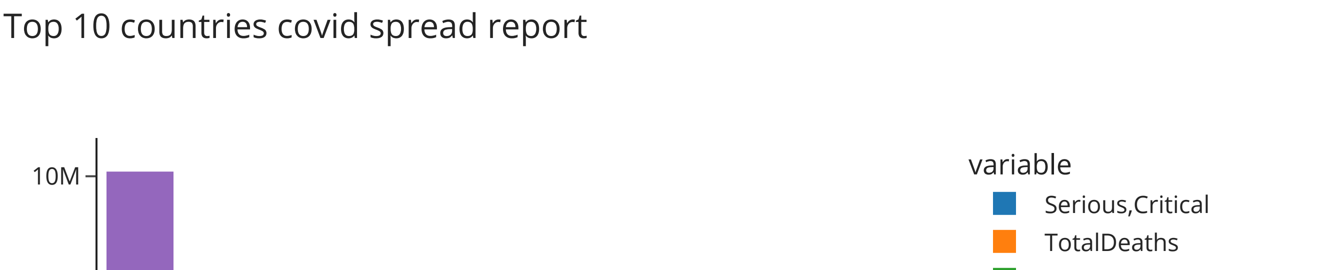
In [26]:

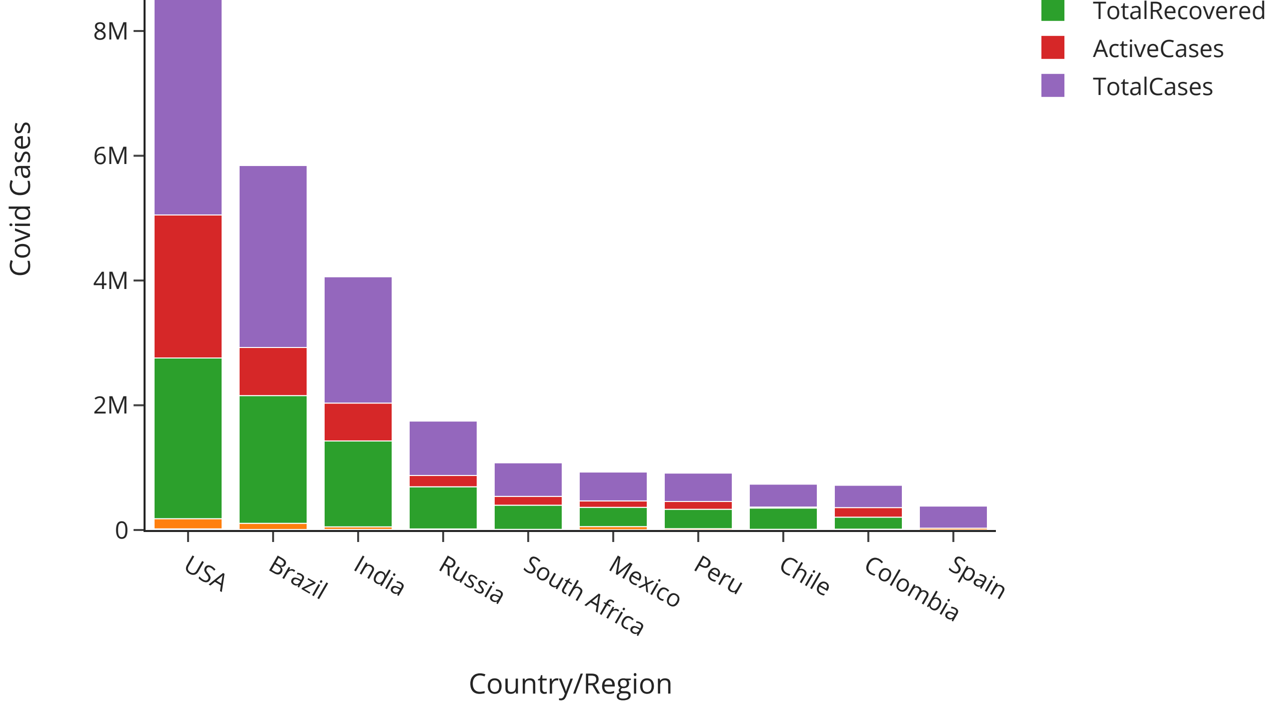
```
fig=px.bar(worldmeter_data.head(10),y='Country/Region',x='TotalCases',color='TotalCases'
fig.update_layout(template="plotly_dark",title_text="Top 10 countries with Total confirm
fig.show()
```



In [27]:

```
px.bar(worldmeter_data.sort_values(by="TotalCases", ascending=False).head(10),x='Country
       labels={"value": "Covid Cases"})\
    .show()
```





At one Glance data visualization is possbile with stacked bar graphs

Finding the percentage of tests covered for each country

We can get the percentage by total tests / population

```
In [28]: worldmeter_data["PercentageofTotalTests"] = (worldmeter_data['TotalTests']/worldmeter_da
```

```
In [29]: worldmeter_data['PercentageofTotalTests'].idxmax()

Out[29]: 91
```

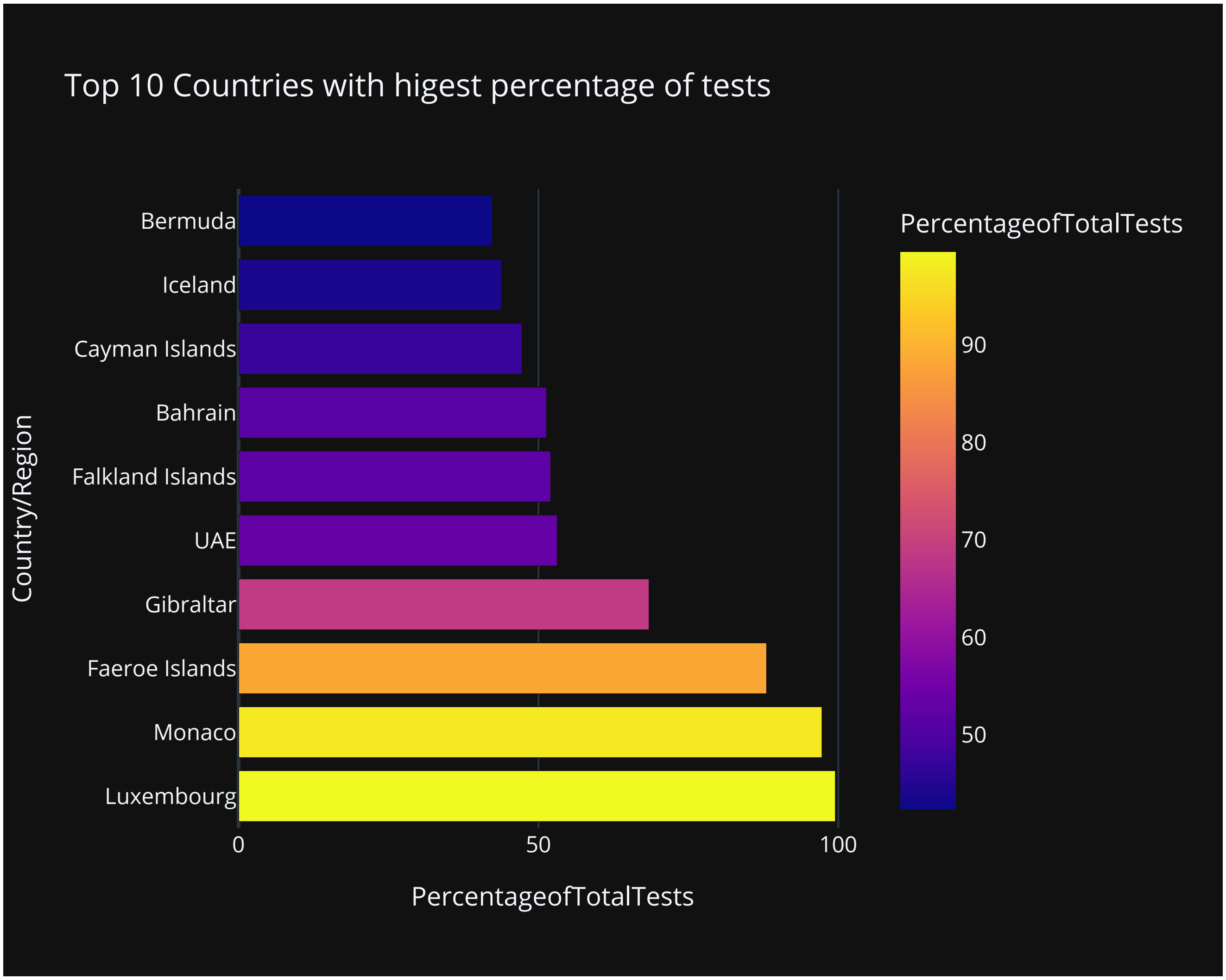
```
In [30]: worldmeter_data.iloc[worldmeter_data['PercentageofTotalTests'].idxmax(), :]
```

```
Out[30]: Country/Region            Luxembourg
         Continent                     Europe
         Population                  626952.0
         TotalCases                      7073
         NewCases                         NaN
         TotalDeaths                    119.0
         NewDeaths                        NaN
         TotalRecovered                5750.0
         NewRecovered                     NaN
         ActiveCases                   1204.0
         Serious,Critical                 9.0
         Tot Cases/1M pop             11282.0
         Deaths/1M pop                  190.0
         TotalTests                  623994.0
         Tests/1M pop                995282.0
         WHO Region                    Europe
         PercentageofTotalTests     99.528194
         Name: 91, dtype: object
```

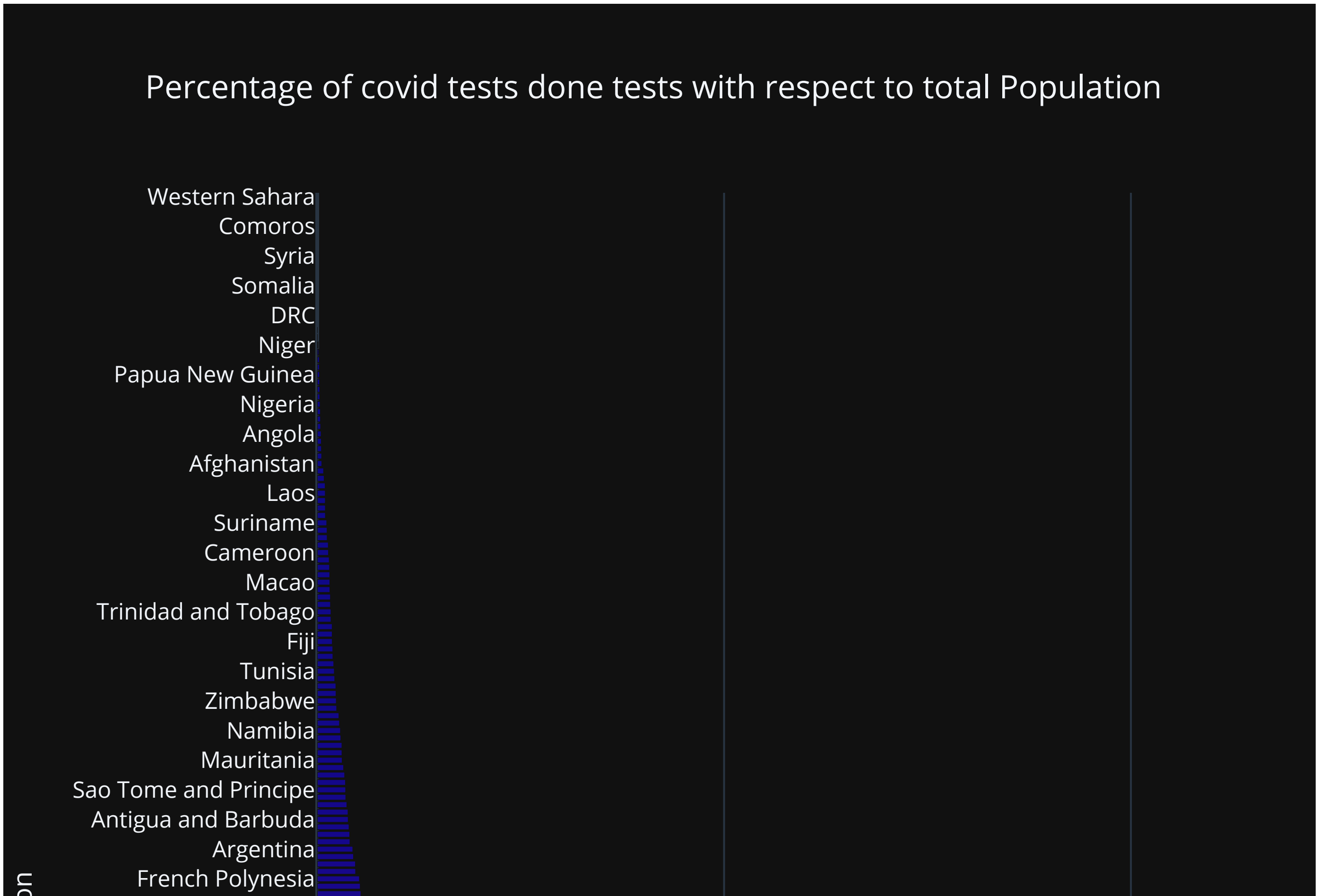
Luxembourg is the country where almost everyone went through covid 19 test

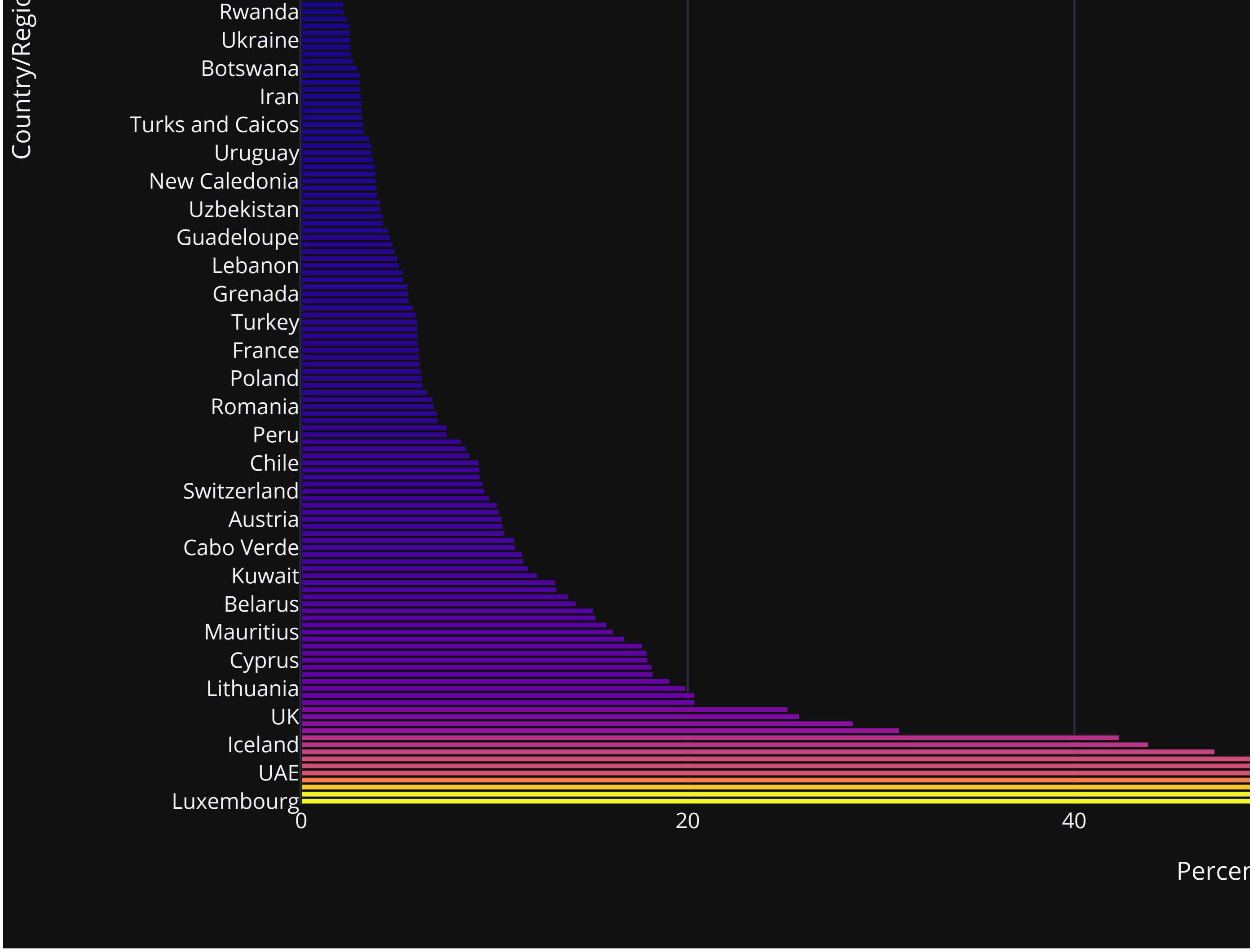
Now we plot the grapth for percentage of total tests conducted

```
In [31]: fig=px.bar(worldmeter_data.sort_values(by = 'PercentageofTotalTests', ascending=False).h
         fig.show()
```



```
In [32]: px.bar(worldmeter_data.sort_values(by = 'PercentageofTotalTests', ascending=False),color
```





if percentage of total tests are very less there should be chance that more number of total cases, active cases in a conutry

Understanding the data with ratios

Critical/Serious covid case to deaths

```
In [33]: worldmeter_data["Critical_to_death_Percentage"] = worldmeter_data['Serious,Critical']/wo
```

```
In [34]: worldmeter_data["Critical_to_death_Percentage"].max()
```

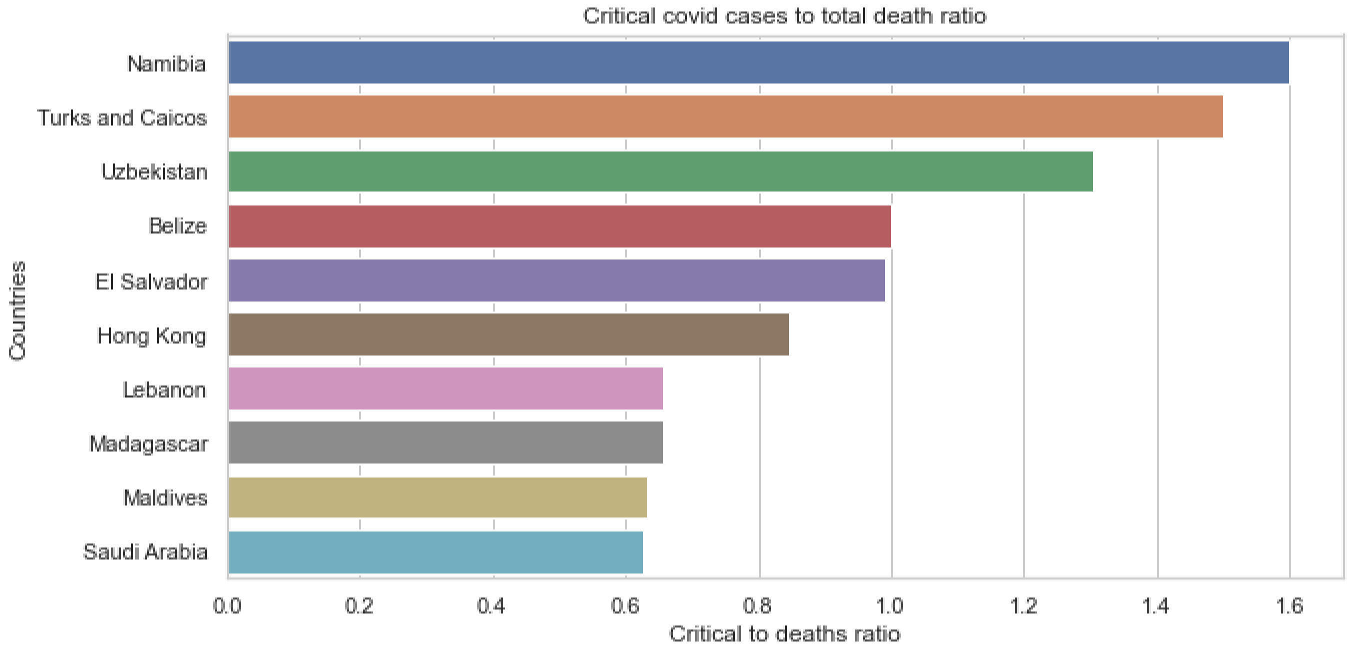
```
Out[34]: 1.6
```

```
In [35]: worldmeter_data["Critical_to_death_Percentage"]
```

```
Out[35]: 0      0.112381
         1      0.084323
         2      0.214804
         3      0.157470
         4      0.056122
                  ...
         204         NaN
         205         NaN
         206         NaN
         207         NaN
         208         NaN
         Name: Critical_to_death_Percentage, Length: 209, dtype: float64
```

```
In [36]: sns.set_theme(style="whitegrid")
```

```
plt.figure(figsize = (10, 5))
sns.barplot(data = worldmeter_data.sort_values(by = "Critical_to_death_Percentage", asce
    .set(title = "Critical covid cases to total death ratio")
plt.xlabel("Critical to deaths ratio")
plt.ylabel("Countries")
plt.show()
```



In most instances critical cases are converted into deaths for the countries Namibia, Turk, uzbekisthan compared to others`

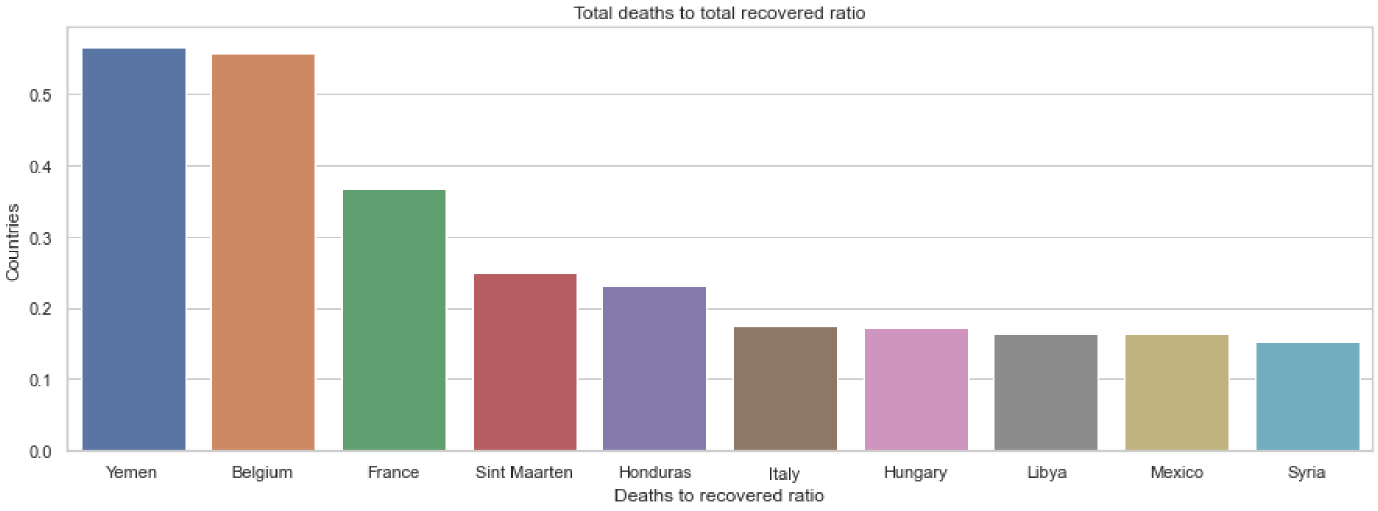
Deaths to recovered ratio

In [37]:

```
worldmeter_data['Deaths_to_recovered_ratio'] = worldmeter_data['TotalDeaths']/worldmeter
```

In [38]:

```
sns.set_theme(style="whitegrid")
plt.figure(figsize = (15, 5))
sns.barplot(data = worldmeter_data.sort_values(by = "Deaths_to_recovered_ratio", ascendi
    .set(title = "Total deaths to total recovered ratio")
plt.xlabel("Deaths to recovered ratio")
plt.ylabel("Countries")
plt.show()
```



More than 50% of chance that the perople from Yemen Belgium can dead when compared with the recovered cases

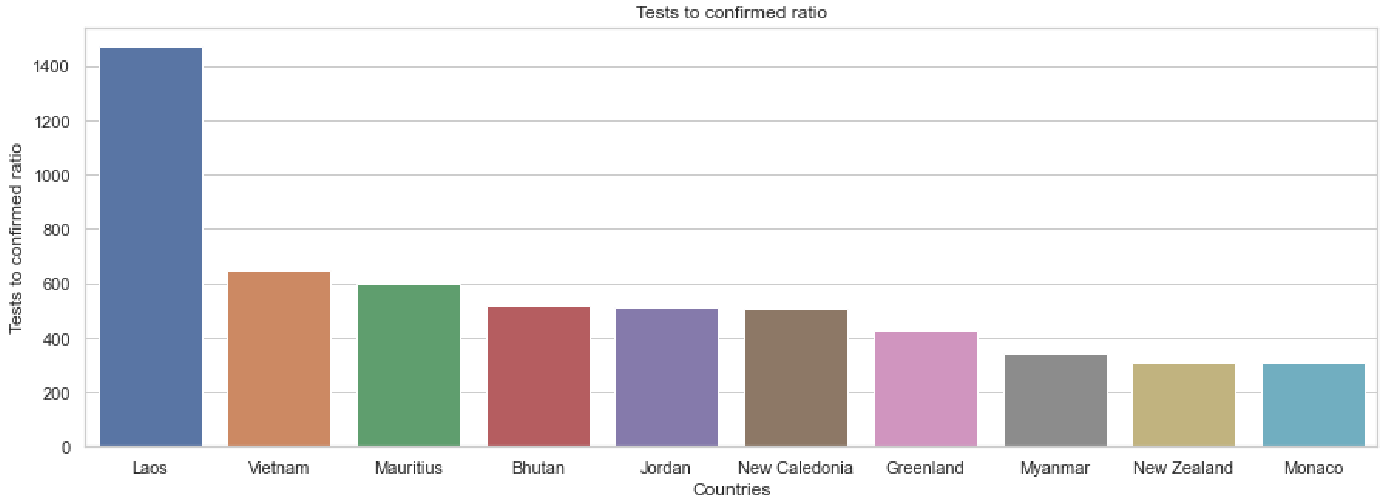
Tests to confirmed ratio

In [39]:
```
worldmeter_data['Tests_to_confirmed_ratio'] = worldmeter_data['TotalTests']/worldmeter_d
```

In [40]:
```
(worldmeter_data['TotalTests']/worldmeter_data['TotalCases']).max()
```

Out[40]:
```
1468.7
```

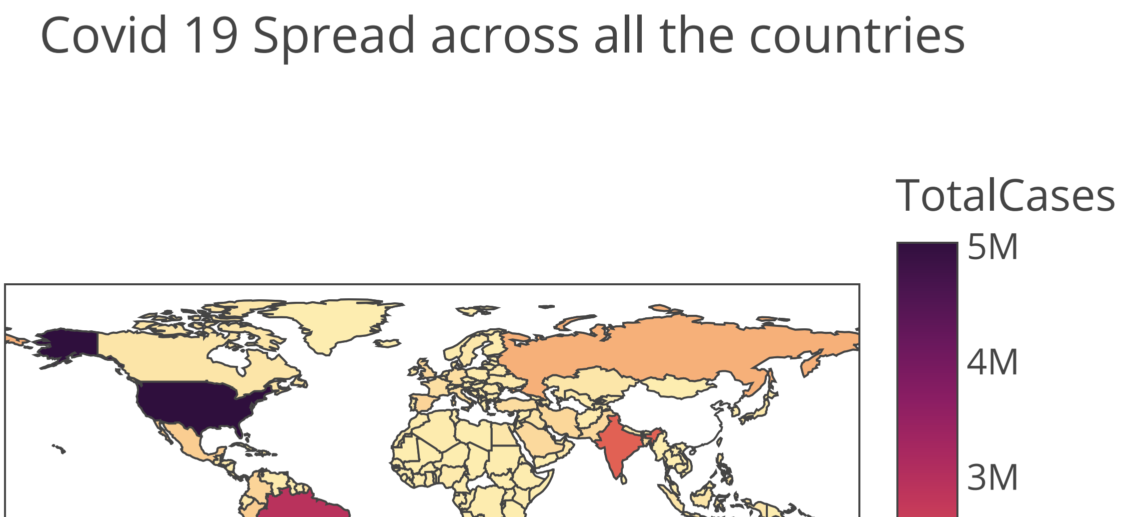
In [41]:
```
sns.set_theme(style="whitegrid")
plt.figure(figsize = (15, 5))
sns.barplot(data = worldmeter_data.sort_values(by = "Tests_to_confirmed_ratio", ascendin
     .set(title = "Tests to confirmed ratio")
plt.ylabel("Tests to confirmed ratio")
plt.xlabel("Countries")
plt.show()
```

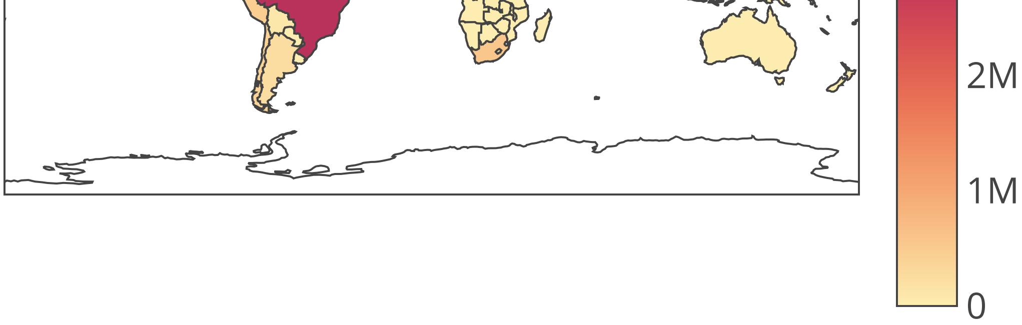


In [42]:
```
worldmeter_data.columns
```

Out[42]:
```
Index(['Country/Region', 'Continent', 'Population', 'TotalCases', 'NewCases',
       'TotalDeaths', 'NewDeaths', 'TotalRecovered', 'NewRecovered',
       'ActiveCases', 'Serious,Critical', 'Tot Cases/1M pop', 'Deaths/1M pop',
       'TotalTests', 'Tests/1M pop', 'WHO Region', 'PercentageofTotalTests',
       'Critical_to_death_Percentage', 'Deaths_to_recovered_ratio',
       'Tests_to_confirmed_ratio'],
      dtype='object')
```

In [43]:
```
px.choropleth(data_frame=worldmeter_data, locations='Country/Region', locationmode="coun
              color_continuous_scale = px.colors.sequential.matter,
              title='Covid 19 Spread across all the countries').show()
```





We can get data for each country as follows

In [44]:

```
worldmeter_data.head()
```

Out[44]:

	Country/Region	Continent	Population	TotalCases	NewCases	TotalDeaths	NewDeaths	TotalRecovered	Ne
0	USA	North America	3.311981e+08	5032179	NaN	162804.0	NaN	2576668.0	
1	Brazil	South America	2.127107e+08	2917562	NaN	98644.0	NaN	2047660.0	
2	India	Asia	1.381345e+09	2025409	NaN	41638.0	NaN	1377384.0	
3	Russia	Europe	1.459409e+08	871894	NaN	14606.0	NaN	676357.0	
4	South Africa	Africa	5.938157e+07	538184	NaN	9604.0	NaN	387316.0	

In [45]:

```
def country_wisegeoMap(world_data, country):
    for c in country:
        df = world_data[world_data['Country/Region'] == c]
        f = px.choropleth(data_frame=df, locations='Country/Region', locationmode="count
              color_continuous_scale = px.colors.sequential.matter,
              title=f'Covid 19 Spread in {c}')
        f.show()
```

In [46]:

```
worldmeter_data['Country/Region'].unique().tolist()
```

Out[46]:

```
['USA',
 'Brazil',
 'India',
 'Russia',
 'South Africa',
 'Mexico',
 'Peru',
 'Chile',
 'Colombia',
 'Spain',
 'Iran',
 'UK',
 'Saudi Arabia',
 'Pakistan',
 'Bangladesh',
 'Italy',
 'Turkey',
 'Argentina',
 'Germany',
 'France',
 'Iraq',
 'Philippines',
 'Indonesia',
 'Canada',
 'Qatar',
 'Kazakhstan',
 'Egypt',
 'Ecuador',
 'Bolivia',
 'Sweden',
 'Oman',
 'Israel',
 'Ukraine',
 'Dominican Republic',
 'Panama',
 'Belgium',
 'Kuwait',
 'Belarus',
 'UAE',
 'Romania',
 'Netherlands',
 'Singapore',
 'Guatemala',
 'Portugal',
 'Poland',
 'Nigeria',
 'Honduras',
 'Bahrain',
 'Japan',
 'Armenia',
 'Ghana',
 'Kyrgyzstan',
 'Afghanistan',
 'Switzerland',
 'Algeria',
 'Azerbaijan',
 'Morocco',
 'Uzbekistan',
 'Serbia',
 'Moldova',
 'Ireland',
 'Kenya',
 'Venezuela',
 'Nepal',
 'Austria',
 'Costa Rica',
 'Ethiopia',
 'Australia',
 'El Salvador',
 'Czechia',
 'Cameroon',
 'Ivory Coast',
 'S. Korea',
 'Denmark',
 'Palestine',
 'Bosnia and Herzegovina',
 'Bulgaria',
 'Madagascar',
 'Sudan',
 'North Macedonia',
 'Senegal',
'Norway',
'DRC',
'Malaysia',
'French Guiana',
'Gabon',
'Tajikistan',
'Guinea',
'Haiti',
'Finland',
'Zambia',
'Luxembourg',
'Mauritania',
'Paraguay',
'Albania',
'Lebanon',
'Croatia',
'Djibouti',
'Greece',
'Libya',
'Equatorial Guinea',
'Maldives',
'CAR',
'Hungary',
'Malawi',
'Zimbabwe',
'Nicaragua',
'Hong Kong',
'Congo',
'Montenegro',
'Thailand',
'Somalia',
'Mayotte',
'Eswatini',
'Sri Lanka',
'Cuba',
'Cabo Verde',
'Namibia',
'Mali',
'Slovakia',
'South Sudan',
'Slovenia',
'Lithuania',
'Estonia',
'Mozambique',
'Rwanda',
'Suriname',
'Guinea-Bissau',
'Benin',
'Iceland',
'Sierra Leone',
'Yemen',
'Tunisia',
'New Zealand',
'Angola',
'Uruguay',
'Latvia',
'Jordan',
'Liberia',
'Uganda',
'Cyprus',
'Georgia',
'Burkina Faso',
'Niger',
'Togo',
'Syria',
'Jamaica',
'Malta',
'Andorra',
'Chad',
'Gambia',
'Sao Tome and Principe',
'Botswana',
'Bahamas',
'Vietnam',
'Lesotho',
'Diamond Princess',
'San Marino',
'Réunion',
'Channel Islands',
'Guyana',
'Tanzania',
'Taiwan',
'Comoros',
'Burundi',
'Myanmar',
'Mauritius',
'Isle of Man',
'Mongolia',
'Eritrea',
'Guadeloupe',
'Martinique',
'Faeroe Islands',
'Aruba',
'Cambodia',
'Trinidad and Tobago',
'Cayman Islands',
'Gibraltar',
'Papua New Guinea',
'Sint Maarten',
'Bermuda',
'Brunei ',
'Barbados',
'Turks and Caicos',
'Seychelles',
'Monaco',
'Bhutan',
'Antigua and Barbuda',
'Liechtenstein',
'Belize',
'French Polynesia',
'St. Vincent Grenadines',
'Saint Martin',
'Macao',
'Curaçao',
'Fiji',
'Saint Lucia',
'Timor-Leste',
'Grenada',
'New Caledonia',
'Laos',
'Dominica',
'Saint Kitts and Nevis',
'Greenland',
'Montserrat',
'Caribbean Netherlands',
'Falkland Islands',
'Vatican City',
'Western Sahara']
```

In [47]:

```
country_wisegeoMap(worldmeter_data, ['India', "USA"])
```

Covid 19 Spread in India

Covid 19 Spread in USA

If we need the data for each date, we can get it as follows

In [48]:
```
from plotly.subplots import make_subplots
import plotly.graph_objects as go
```

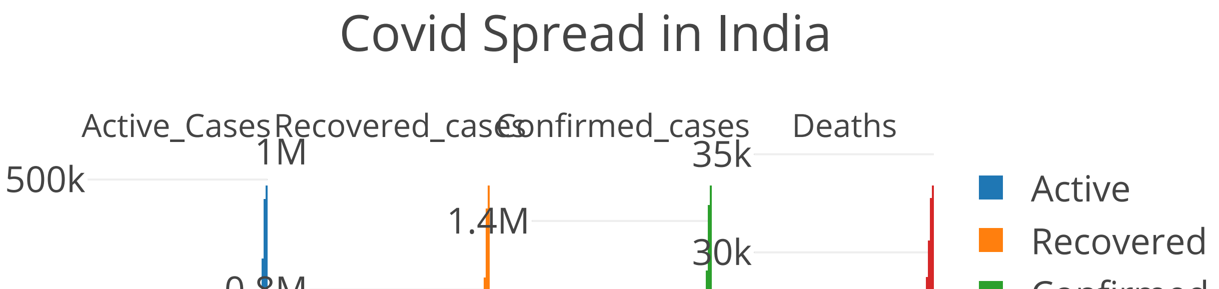
In [49]:
```
fully_grouped_data.head()
```

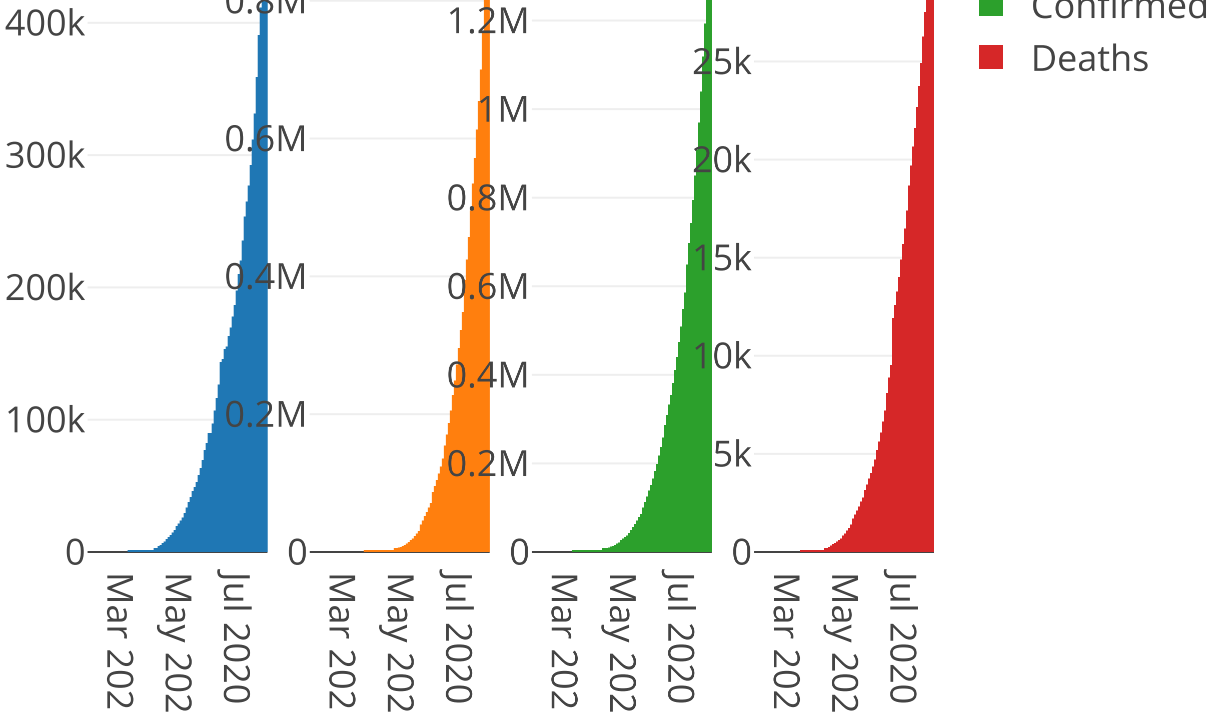
Out[49]:

	Date	Country/Region	Confirmed	Deaths	Recovered	Active	New cases	New deaths	New recovered	WHO Region
0	2020-01-22	Afghanistan	0	0	0	0	0	0	0	Eastern Mediterranean
1	2020-01-22	Albania	0	0	0	0	0	0	0	Europe
2	2020-01-22	Algeria	0	0	0	0	0	0	0	Africa
3	2020-01-22	Andorra	0	0	0	0	0	0	0	Europe
4	2020-01-22	Angola	0	0	0	0	0	0	0	Africa

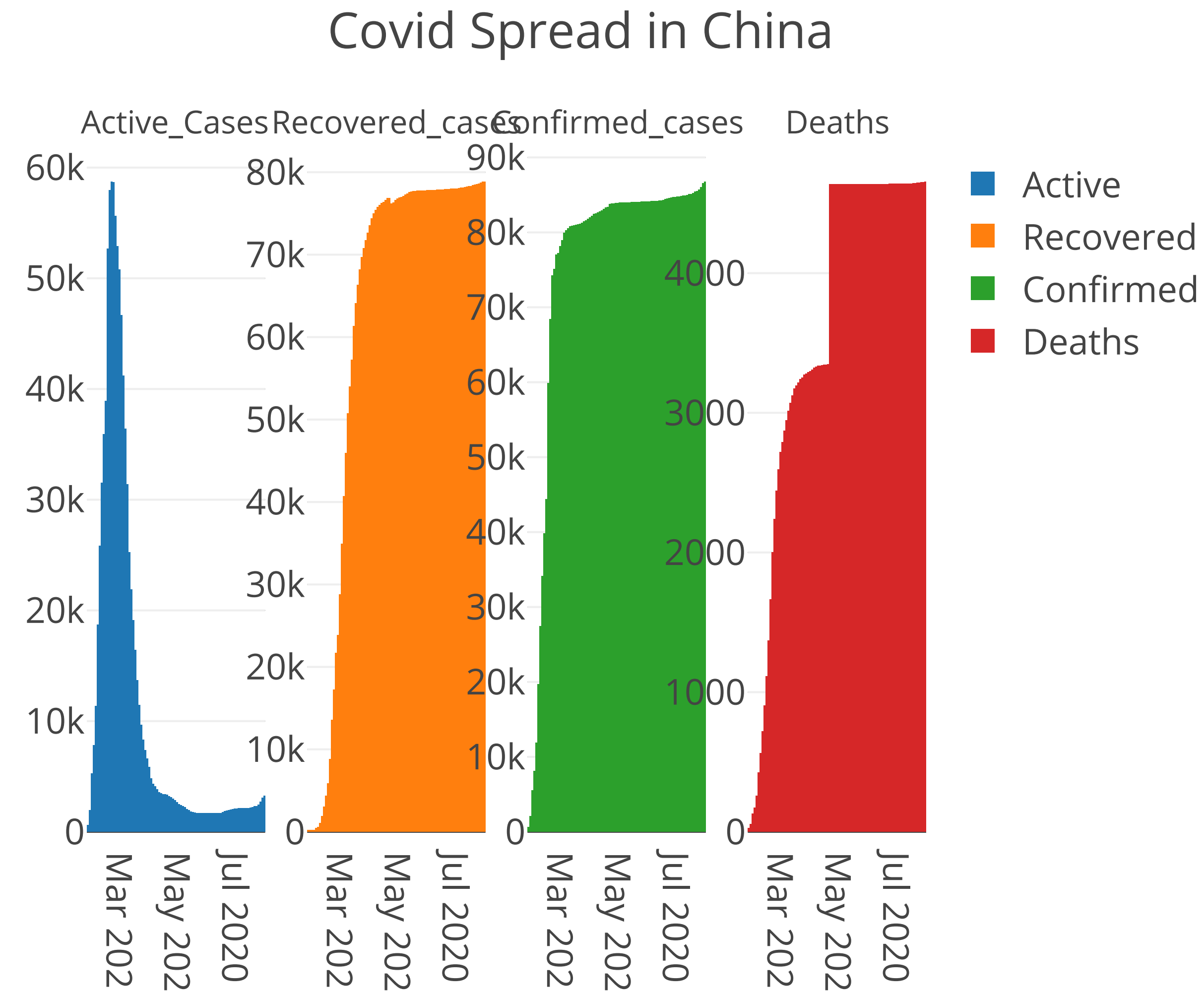
In [50]:
```
def country_visualization(fully_grouped_data,country):

    grouped_df=fully_grouped_data[fully_grouped_data['Country/Region']==country]
    grouped_df=grouped_df.loc[:,['Date',"Active", 'Recovered',"Confirmed", 'Deaths']]
    plot = make_subplots(rows=1, cols=4,subplot_titles=("Active_Cases", 'Recovered_cases
    plot.add_trace(
        go.Bar(name="Active",x=grouped_df['Date'],y=grouped_df['Active']),
        row=1, col=1
    )

    plot.add_trace(
        go.Bar(name="Recovered",x=grouped_df['Date'],y=grouped_df['Recovered']),
        row=1, col=2
    )
    plot.add_trace(
        go.Bar(name="Confirmed",x=grouped_df['Date'],y=grouped_df['Confirmed']),
        row=1, col=3
    )

    plot.add_trace(
        go.Bar(name="Deaths",x=grouped_df['Date'],y=grouped_df['Deaths']),
        row=1, col=4
    )

    plot.update_layout(title_text=f"Covid Spread in {country}",template="presentation")
    plot.show()
```

In [51]:
```
country_visualization(fully_grouped_data,'India')
```





```
In [52]: country_visualization(fully_grouped_data,'China')
```


Covid Spread in China
Active_Cases
Recovered_cases
Confirmed_cases
Deaths
60k
50k
40k
30k
20k
10k
80k
70k
60k
50k
40k
30k
20k
10k
90k
80k
70k
60k
50k
40k
30k
20k
10k
Mar 202
May 202
Jul 2020
🟦 Active
🟧 Recovered
🟩 Confirmed
🟥 Deaths