In [1]:

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

In [2]:

```
files = os.listdir('/Users/haleigh/Desktop/Udemy Courses/Data Analysis Projects/COVID-19 Project/COVID-19 Files')
```

In [3]:

```
# print the files and show what .csv we are working with
files
```

Out[3]:

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

In [4]:

```
def read_data(path, filename):
    return pd.read_csv(path+'/'+filename)
```

In [6]:

```
path = '/Users/haleigh/Desktop/Udemy Courses/Data Analysis Projects/COVID-19 Project/COVID-19 Files'
world_data = read_data(path, 'worldometer_data.csv')
```

In [7]:

```
world_data.head()
```

Out[7]:

	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
0	USA	North America	3.311981e+08	5032179	NaN	162804.0	NaN	2576668.0	NaN	2292707.0	18296.0	15194.0	492.0	63139605.0	190640.0	Americas
1	Brazil	South America	2.127107e+08	2917562	NaN	98644.0	NaN	2047660.0	NaN	771258.0	8318.0	13716.0	464.0	13206188.0	62085.0	Americas
2	India	Asia	1.381345e+09	2025409	NaN	41638.0	NaN	1377384.0	NaN	606387.0	8944.0	1466.0	30.0	22149351.0	16035.0	South-EastAsia
3	Russia	Europe	1.459409e+08	871894	NaN	14606.0	NaN	676357.0	NaN	180931.0	2300.0	5974.0	100.0	29716907.0	203623.0	Europe
4	South Africa	Africa	5.938157e+07	538184	NaN	9604.0	NaN	387316.0	NaN	141264.0	539.0	9063.0	162.0	3149807.0	53044.0	Africa

In [8]:

```
day_wise = read_data(path, files[2])
```

In [9]:

```
group_data = read_data(path, files[1])
```

In [10]:

```
usa_data = read_data(path, files[4])
```

In [11]:

```python
province_data = read_data(path, files[3])
```

In [13]:

```python
# analysing trend of total cases, deaths, recovered, and active cases
world_data.columns
```

Out[13]:

```
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')
```

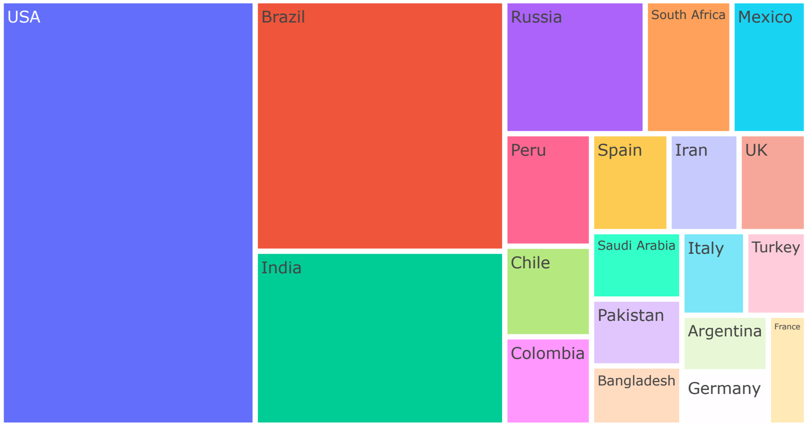
In [14]:

```python
import plotly.express as px
```

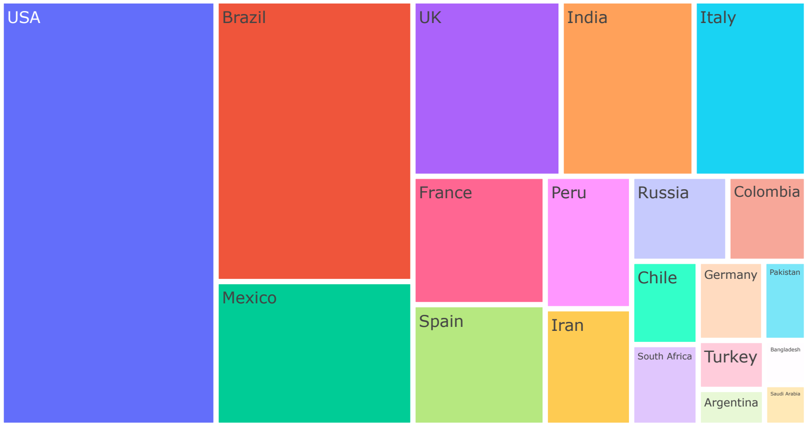
In [16]:

```python
columns =['TotalCases','TotalDeaths','TotalRecovered','ActiveCases']
for i in columns:
    # creating tree map
    fig = px.treemap(world_data.iloc[0:20], values = i, path = ['Country/Region'], title = 'Countries with respect to their {}'.format(i)) # first 20 rows
    fig.show()
```

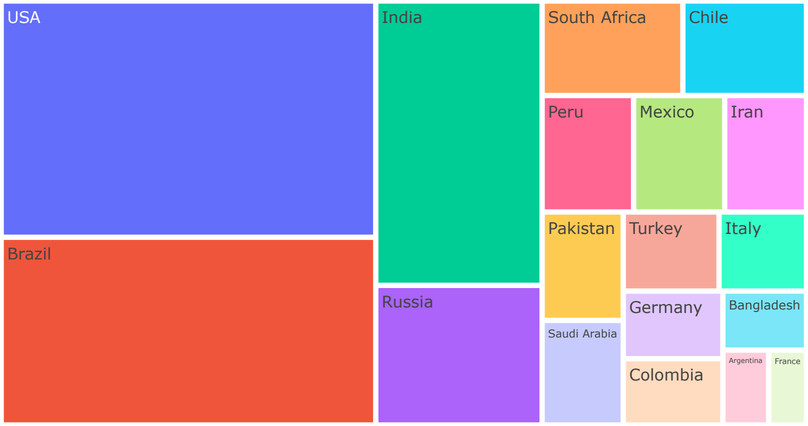
Countries with respect to their TotalCases



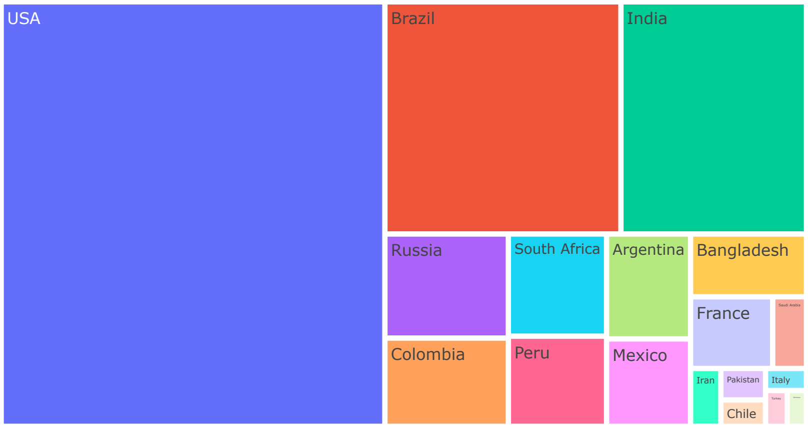
Countries with respect to their TotalDeaths



Countries with respect to their TotalRecovered



Countries with respect to their ActiveCases



In [17]:

```
day_wise.head()
```

Out[17]:

	Date	Confirmed	Deaths	Recovered	Active	New cases	New deaths	New recovered	Deaths / 100 Cases	Recovered / 100 Cases	Deaths / 100 Recovered	No. of countries
0	2020-01-22	555	17	28	510	0	0	0	3.06	5.05	60.71	6
1	2020-01-23	654	18	30	606	99	1	2	2.75	4.59	60.00	8
2	2020-01-24	941	26	36	879	287	8	6	2.76	3.83	72.22	9
3	2020-01-25	1434	42	39	1353	493	16	3	2.93	2.72	107.69	11
4	2020-01-26	2118	56	52	2010	684	14	13	2.64	2.46	107.69	13

In [18]:

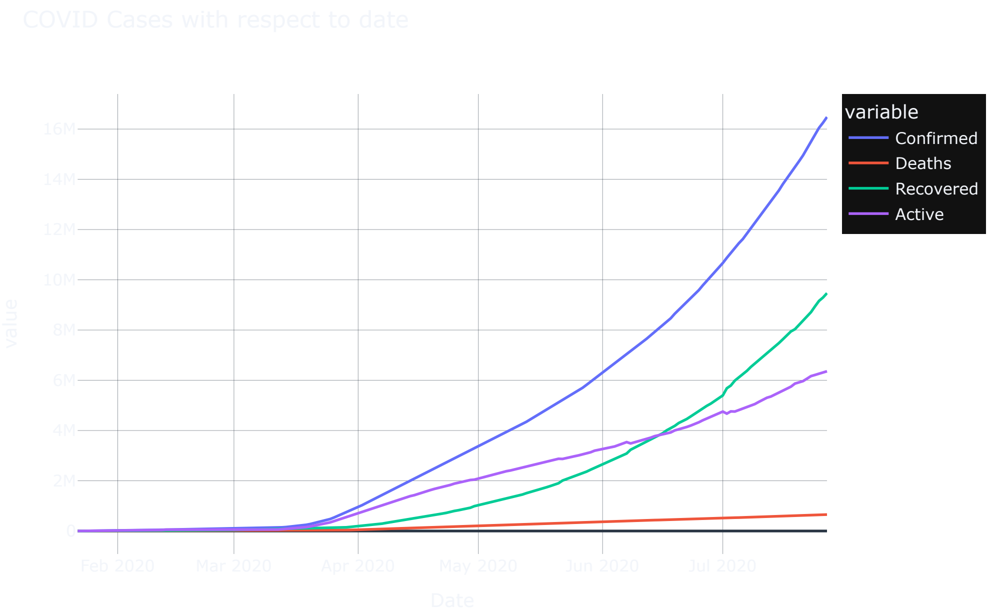
```
day_wise.columns
```

Out[18]:

```
Index(['Date', 'Confirmed', 'Deaths', 'Recovered', 'Active', 'New cases',
       'New deaths', 'New recovered', 'Deaths / 100 Cases',
       'Recovered / 100 Cases', 'Deaths / 100 Recovered', 'No. of countries'],
      dtype='object')
```

In [19]:

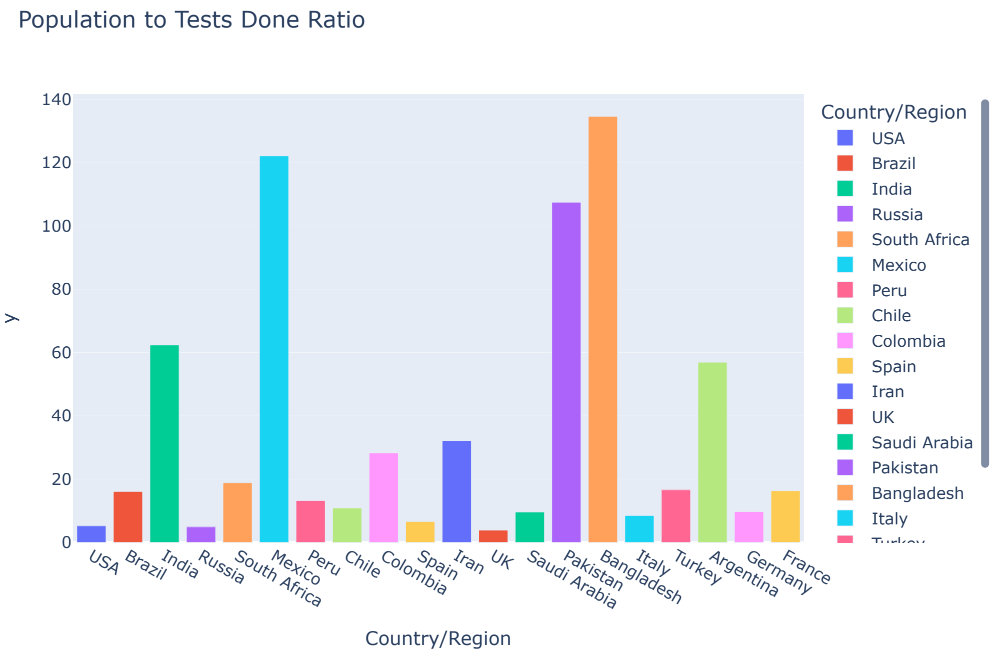
```
# what is the trend of confirmed deaths, recovered, and active cases?
px.line(day_wise, x='Date', y=['Confirmed','Deaths','Recovered','Active'], title = 'COVID Cases with respect to date', template = 'plotly_dark')
```



In [22]:

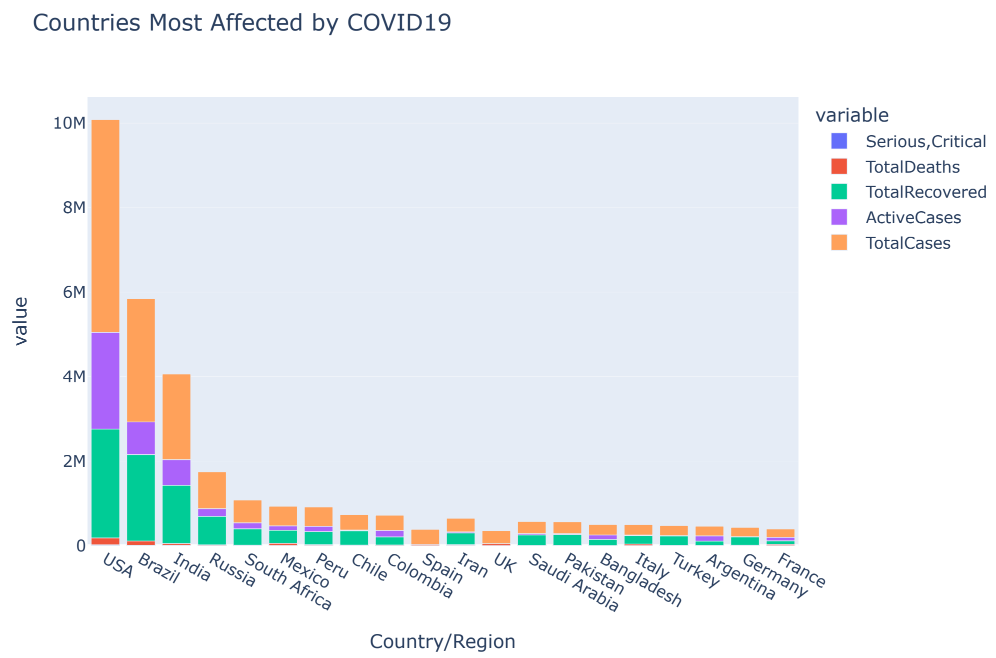
```
# visualize population of tests done ratio
# create the dones feature using population % total number of tests
population_test_ratio = world_data['Population']/world_data['TotalTests'].iloc[0:20]
```

```
In [23]:  fig = px.bar(world_data.iloc[0:20], x='Country/Region', y=population_test_ratio[0:20] ,color='Country/Region', title = 'Population to Tests Done Ratio')
          fig.show()
```



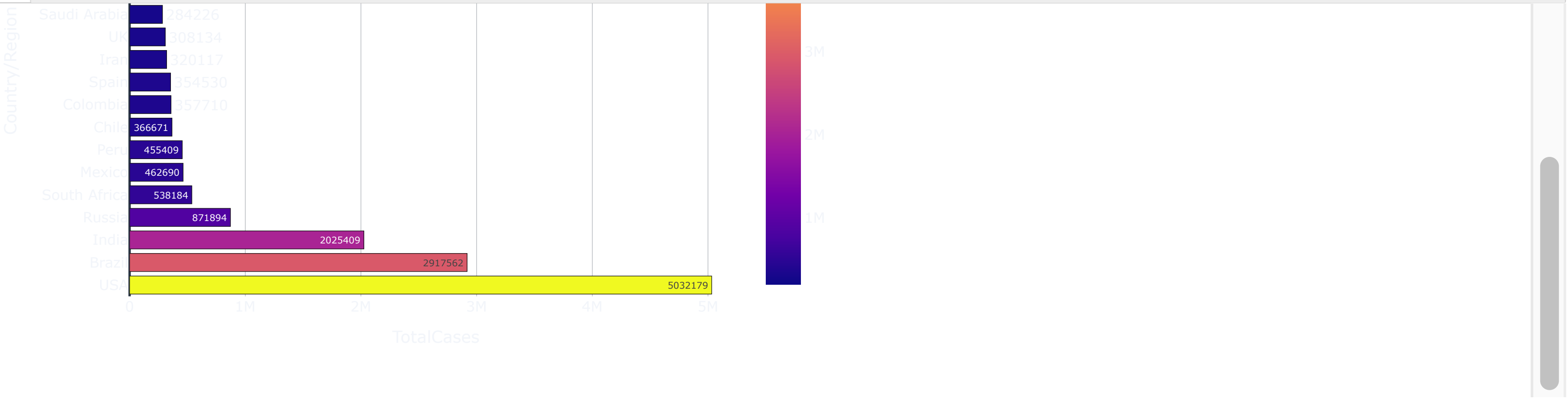
In [28]:

```
top 20 countries most affected by COVID19
g = px.bar(world_data.iloc[0:20], x='Country/Region', y=['Serious,Critical', 'TotalDeaths', 'TotalRecovered', 'ActiveCases', 'TotalCases'], title = 'Countries M
g.show()
```



In [29]:

```
# top 20 countries with maximum confirmed cases
fig = px.bar(world_data.iloc[0:20], y = 'Country/Region', x = 'TotalCases', color = 'TotalCases', text = 'TotalCases')
fig.update_layout(template = 'plotly_dark', title_text = 'Top 20 Countries: Maximum Confirmed Cases')
fig.show()
```



In [30]:

```
world_data.sort_values(by='TotalDeaths',ascending=False)
```

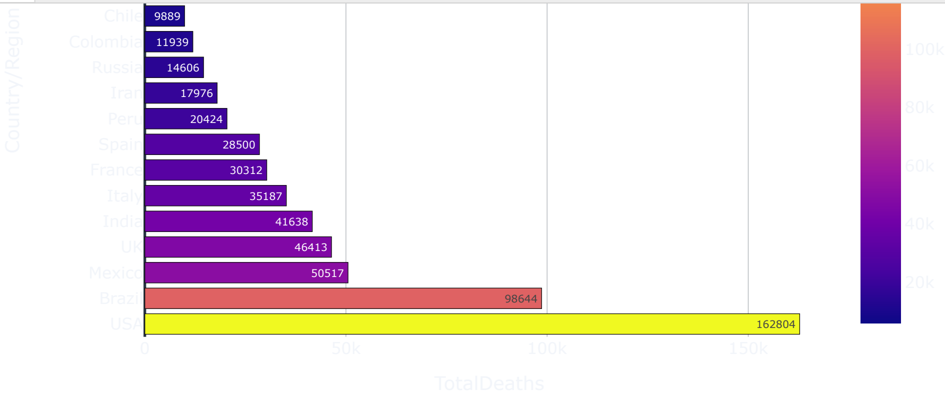
Out[30]:

	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
0	USA	North America	3.311981e+08	5032179	NaN	162804.0	NaN	2576668.0	NaN	2292707.0	18296.0	15194.0	492.0	63139605.0	190640.0	Americas
1	Brazil	South America	2.127107e+08	2917562	NaN	98644.0	NaN	2047660.0	NaN	771258.0	8318.0	13716.0	464.0	13206188.0	62085.0	Americas
5	Mexico	North America	1.290662e+08	462690	6590.0	50517.0	819.0	308848.0	4140.0	103325.0	3987.0	3585.0	391.0	1056915.0	8189.0	Americas
11	UK	Europe	6.792203e+07	308134	NaN	46413.0	NaN	NaN	NaN	NaN	73.0	4537.0	683.0	17515234.0	257873.0	Europe
2	India	Asia	1.381345e+09	2025409	NaN	41638.0	NaN	1377384.0	NaN	606387.0	8944.0	1466.0	30.0	22149351.0	16035.0	South-EastAsia
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
202	Saint Kitts and Nevis	North America	5.323700e+04	17	NaN	NaN	NaN	16.0	NaN	1.0	NaN	319.0	NaN	1146.0	21526.0	Americas
203	Greenland	North America	5.678000e+04	14	NaN	NaN	NaN	14.0	NaN	0.0	NaN	247.0	NaN	5977.0	105266.0	Europe
205	Caribbean Netherlands	North America	2.624700e+04	13	NaN	NaN	NaN	7.0	NaN	6.0	NaN	495.0	NaN	424.0	16154.0	NaN
206	Falkland Islands	South America	3.489000e+03	13	NaN	NaN	NaN	13.0	NaN	0.0	NaN	3726.0	NaN	1816.0	520493.0	NaN
207	Vatican City	Europe	8.010000e+02	12	NaN	NaN	NaN	12.0	NaN	0.0	NaN	14981.0	NaN	NaN	NaN	Europe

209 rows × 16 columns

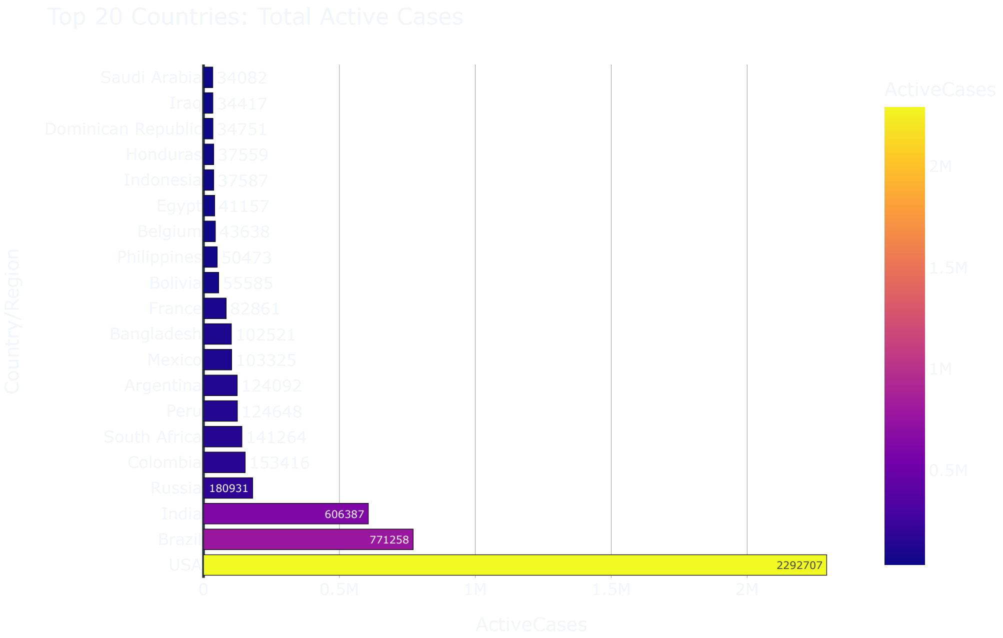
In [32]:

```
# top 20 countries with maximum total deaths
fig = px.bar(world_data.sort_values(by='TotalDeaths',ascending=False)[0:20],y = 'Country/Region', x = 'TotalDeaths', color = 'TotalDeaths', text = 'TotalDe
fig.update_layout(template = 'plotly_dark', title_text = 'Top 20 Countries: Total Deaths')
fig.show()
```



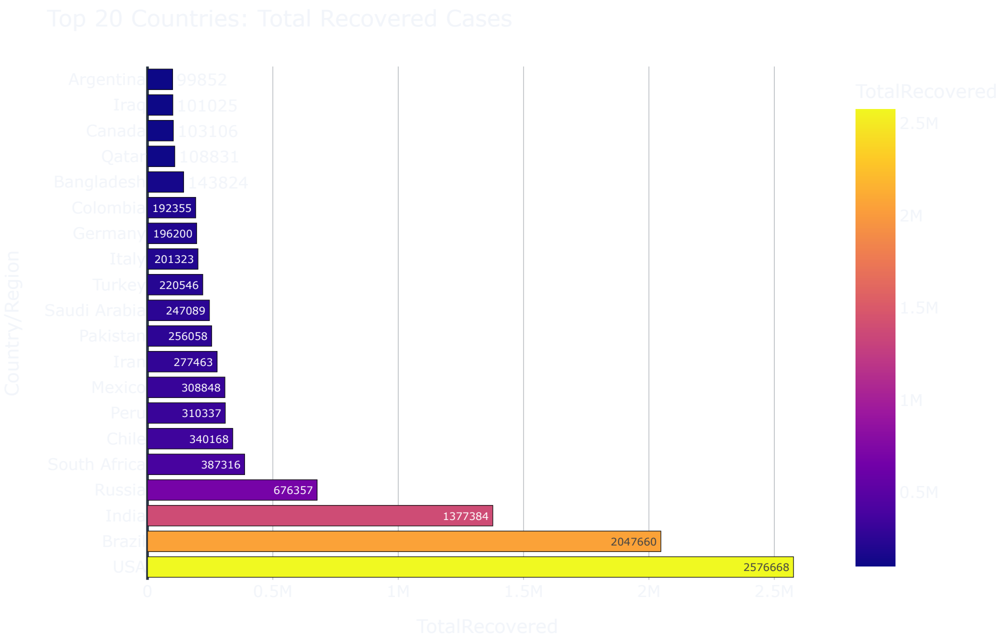
In [33]:

```
# top 20 countries with maximum active cases
fig = px.bar(world_data.sort_values(by='ActiveCases',ascending=False)[0:20],y = 'Country/Region', x = 'ActiveCases', color = 'ActiveCases', text = 'ActiveC
fig.update_layout(template = 'plotly_dark', title_text = 'Top 20 Countries: Total Active Cases')
fig.show()
```



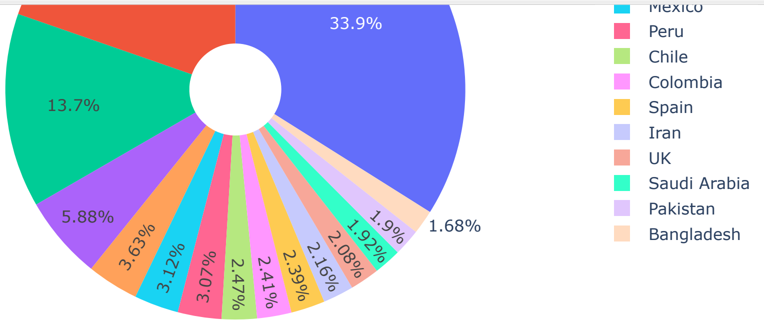
In [34]:

```
# top 20 countries with maximum recovered cases
fig = px.bar(world_data.sort_values(by='TotalRecovered',ascending=False)[0:20],y = 'Country/Region', x = 'TotalRecovered', color = 'TotalRecovered', text =
fig.update_layout(template = 'plotly_dark', title_text = 'Top 20 Countries: Total Recovered Cases')
fig.show()
```



In [36]:

```
# worst affected countries
labels = world_data[0:15]['Country/Region'].values
cases =['TotalCases','TotalDeaths','TotalRecovered','ActiveCases']
for i in cases:
    fig = px.pie(world_data[0:15], values = i, names = labels,hole = 0.2, title = "{} Most Affected Countries Recorded By WHO".format(i))
    fig.show()
```



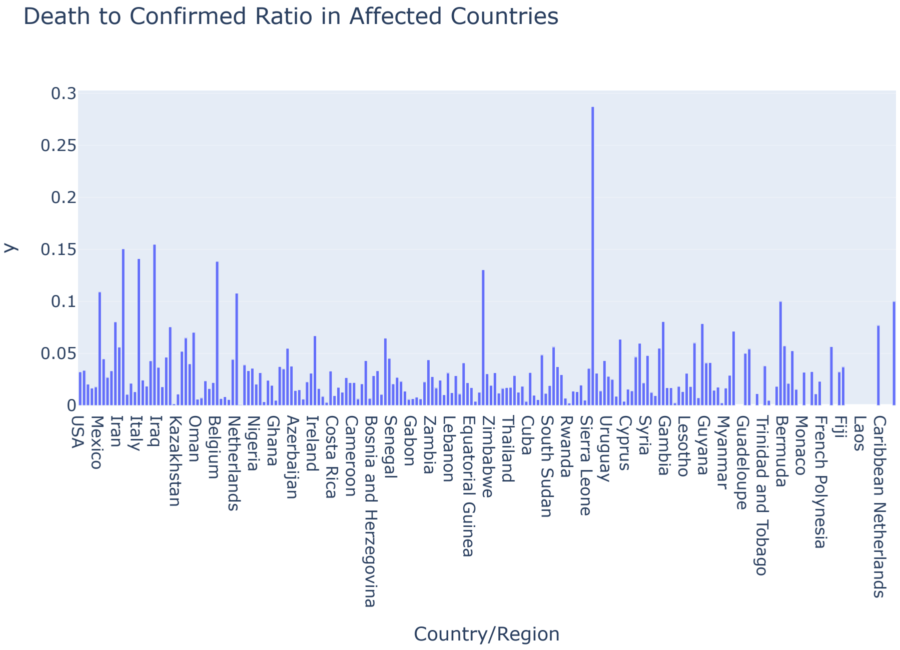
In [37]:

```
# deaths to confirmed ratio
deaths_to_confirmed = world_data['TotalDeaths']/world_data['TotalCases']
deaths_to_confirmed
```

Out[37]:

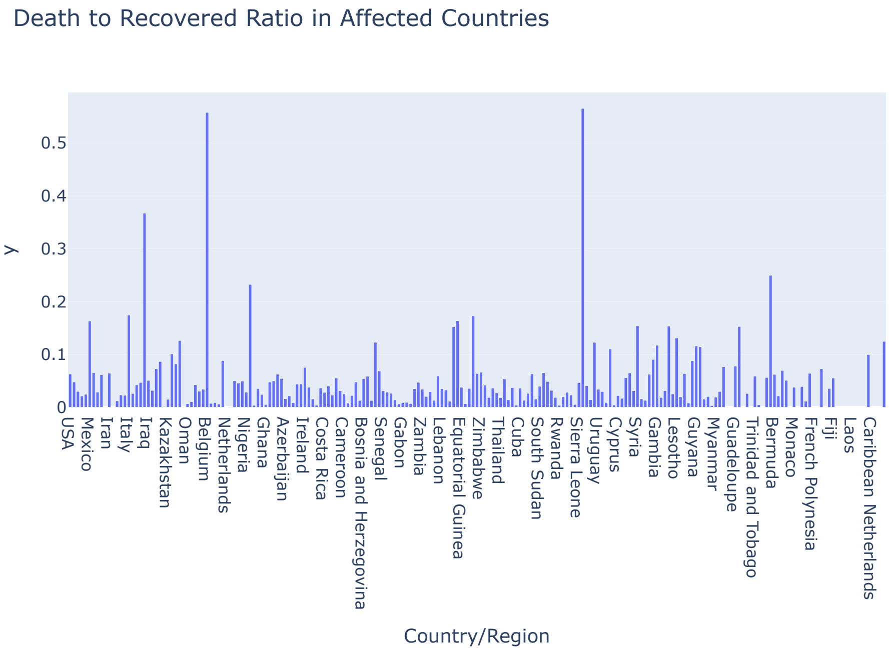
```
0      0.032353
1      0.033810
2      0.020558
3      0.016752
4      0.017845
         ...
204    0.076923
205         NaN
206         NaN
207         NaN
208    0.100000
Length: 209, dtype: float64
```

```
In [38]: px.bar(world_data, x='Country/Region', y=deaths_to_confirmed, title='Death to Confirmed Ratio in Affected Countries')
```



In [39]:

```python
# deaths to recovered ratio
deaths_to_recovered = world_data['TotalDeaths']/world_data['TotalRecovered']
px.bar(world_data, x='Country/Region', y=deaths_to_recovered, title='Death to Recovered Ratio in Affected Countries')
```

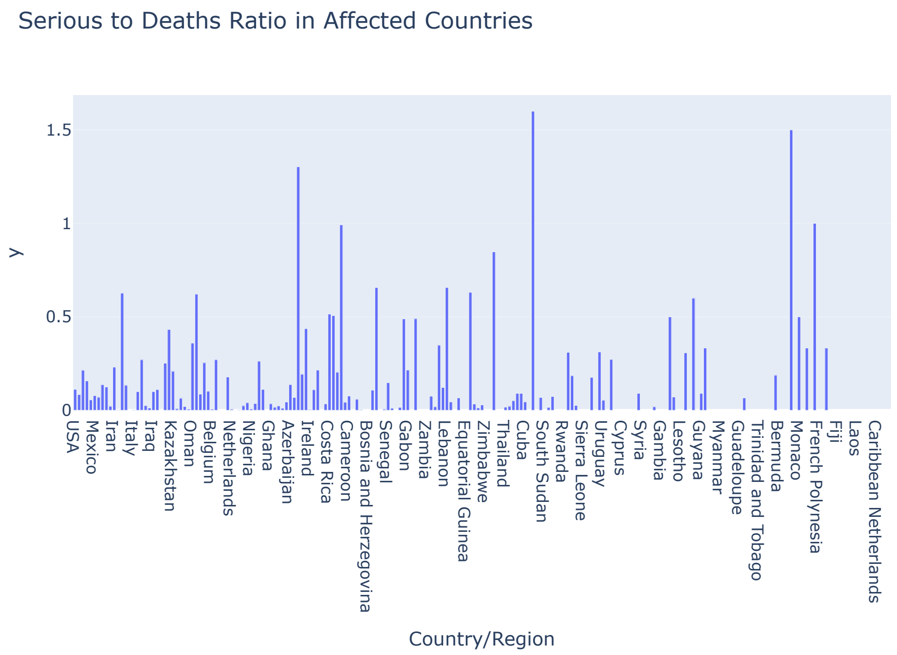


In [40]:

```python
# serious to deaths ratio
serious_to_deaths = world_data['Serious,Critical']/world_data['TotalDeaths']
```

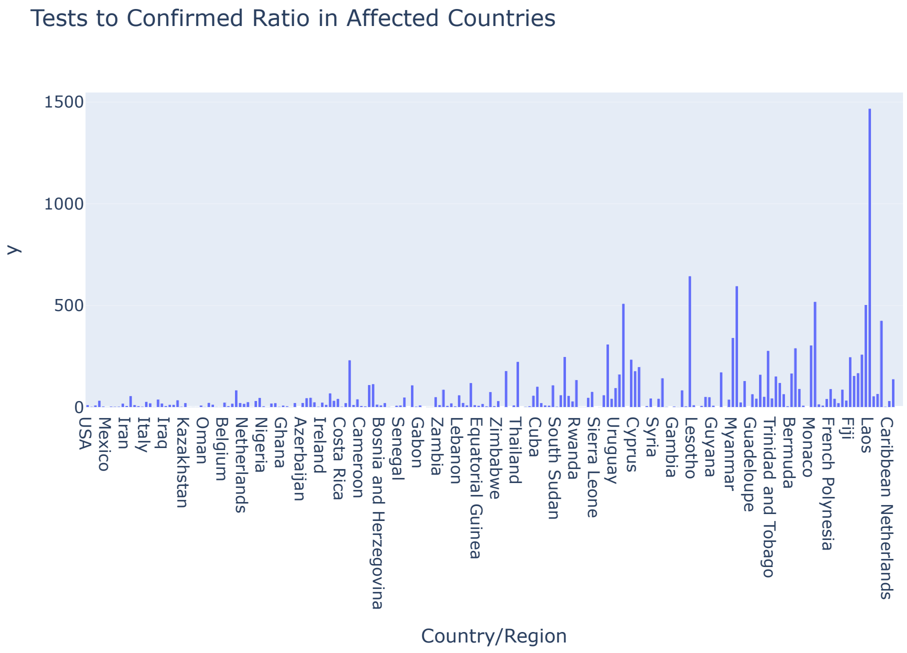
In [41]:

```python
px.bar(world_data, x='Country/Region', y=serious_to_deaths, title='Serious to Deaths Ratio in Affected Countries')
```



In [42]:

```
# tests to confirmed ratio
tests_to_confirmed = world_data['TotalTests']/world_data['TotalCases']
px.bar(world_data, x='Country/Region', y=tests_to_confirmed, title='Tests to Confirmed Ratio in Affected Countries')
```



In [43]:

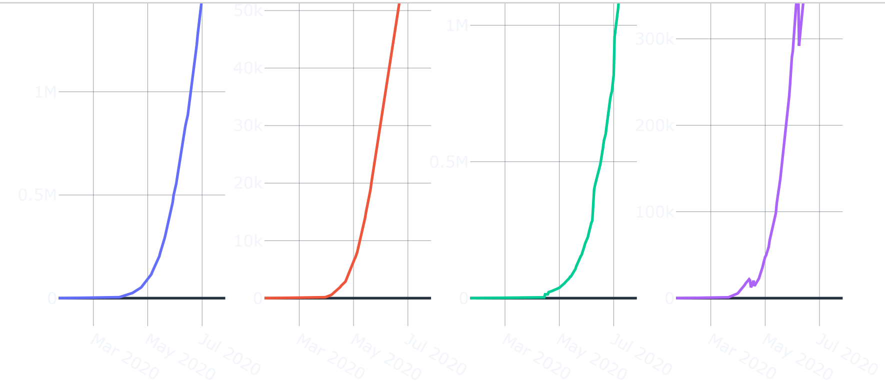
```
from plotly.subplots import make_subplots
import plotly.graph_objects as go
```

In [51]:

```python
# type in country to see visualization automated

def country_visualization(df, country):
    data = df[df['Country/Region'] == country]

    data2 = data.loc[:,['Date','Confirmed', 'Deaths', 'Recovered', 'Active']]

    fig = make_subplots(rows = 1, cols = 4, subplot_titles = ('confirmed','Active','Recovered','Deaths'))

    fig.add_trace(
    go.Scatter(name='Confirmed',x=data2['Date'], y=data2['Confirmed']), row = 1, col = 1
    )

    fig.add_trace(
    go.Scatter(name='Deaths',x=data2['Date'], y=data2['Deaths']), row = 1, col = 2
    )

    fig.add_trace(
    go.Scatter(name='Recovered',x=data2['Date'], y=data2['Recovered']),row = 1, col = 3
    )

    fig.add_trace(
    go.Scatter(name='Active',x=data2['Date'], y=data2['Active']), row = 1, col = 4
    )

    fig.update_layout(height=600, width = 800, title_text = 'Date vs. Recorded Cases of {}'.format(country), template = 'plotly_dark')
    fig.show()
```

In [56]:

```python
# enter any country below, using India, Brazil, and Germany as example
country_visualization(group_data, 'India')
country_visualization(group_data, 'Brazil')
country_visualization(group_data, 'Germany')
```


1M
0.5M
50k
40k
30k
20k
10k
1M
0.5M
300k
200k
100k
Mar 2020
May 2020
Jul 2020
Mar 2020
May 2020
Jul 2020
Mar 2020
May 2020
Jul 2020
Mar 2020
May 2020
Jul 2020