In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
pd.options.display.float_format = '{:,.2f}'.format
```

```
/opt/conda/lib/python3.10/site-packages/scipy/__init__.py:146: UserWar
ning: A NumPy version >=1.16.5 and <1.23.0 is required for this versio
n of SciPy (detected version 1.23.5
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversio
n}"
```

In [2]:
```
df=pd.read_csv("/kaggle/input/corona-virus-report/worldometer_data.csv")
df
```

Out[2]:

	Country/Region	Continent	Population	TotalCases	NewCases	TotalDeaths	New
0	USA	North America	331,198,130.00	5032179	NaN	162,804.00	NaN
1	Brazil	South America	212,710,692.00	2917562	NaN	98,644.00	NaN
2	India	Asia	1,381,344,997.00	2025409	NaN	41,638.00	NaN
3	Russia	Europe	145,940,924.00	871894	NaN	14,606.00	NaN
4	South Africa	Africa	59,381,566.00	538184	NaN	9,604.00	NaN
...	...	...	...	...	...	...	...
204	Montserrat	North America	4,992.00	13	NaN	1.00	NaN
205	Caribbean Netherlands	North America	26,247.00	13	NaN	NaN	NaN
206	Falkland Islands	South America	3,489.00	13	NaN	NaN	NaN
207	Vatican City	Europe	801.00	12	NaN	NaN	NaN
208	Western Sahara	Africa	598,682.00	10	NaN	1.00	NaN

209 rows × 16 columns

In [3]:
```
df.shape
```

Out[3]:
```
(209, 16)
```

In [4]:
```
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 209 entries, 0 to 208
Data columns (total 16 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   Country/Region    209 non-null    object 
 1   Continent         208 non-null    object 
 2   Population        208 non-null    float64
 3   TotalCases        209 non-null    int64  
 4   NewCases          4 non-null      float64
 5   TotalDeaths       188 non-null    float64
 6   NewDeaths         3 non-null      float64
 7   TotalRecovered    205 non-null    float64
 8   NewRecovered      3 non-null      float64
 9   ActiveCases       205 non-null    float64
 10  Serious,Critical  122 non-null    float64
 11  Tot Cases/1M pop  208 non-null    float64
 12  Deaths/1M pop     187 non-null    float64
 13  TotalTests        191 non-null    float64
 14  Tests/1M pop      191 non-null    float64
 15  WHO Region        184 non-null    object 
dtypes: float64(12), int64(1), object(3)
memory usage: 26.2+ KB
None
```

As you can see there are multiple null values in DataFrame but before that lets run and study some statistical summary

In [5]:
```
df.describe()
```

Out[5]:

	Population	TotalCases	NewCases	TotalDeaths	NewDeaths	TotalRecovered
count	208.00	209.00	4.00	188.00	3.00	205.00
mean	30,415,486.97	91,718.50	1,980.50	3,792.59	300.00	58,878.98
std	104,766,099.48	432,586.68	3,129.61	15,487.18	451.20	256,698.41
min	801.00	10.00	20.00	1.00	1.00	7.00
25%	966,314.00	712.00	27.50	22.00	40.50	334.00
50%	7,041,972.50	4,491.00	656.00	113.00	80.00	2,178.00
75%	25,756,135.50	36,896.00	2,609.00	786.00	449.50	20,553.00
max	1,381,344,997.00	5,032,179.00	6,590.00	162,804.00	819.00	2,576,668.00

lets count null values

In [6]:
```
df.isnull().sum()
```

Out[6]:
```
Country/Region        0
Continent             1
Population            1
TotalCases            0
NewCases            205
TotalDeaths          21
NewDeaths           206
TotalRecovered        4
NewRecovered        206
ActiveCases           4
Serious,Critical     87
Tot Cases/1M pop      1
Deaths/1M pop        22
TotalTests           18
Tests/1M pop         18
WHO Region           25
dtype: int64
```

In such DataFrame if data is null then we have to keep it blank as it might be possible that that's true.

Let's talk about the "TotalRecovered" column. we can assume that 4 null values mean no recovery at that point in time

formatting column name for ease of use

In [7]:
```
lowercase_columns = [column.lower() for column in df.columns]
df.columns = lowercase_columns
df.head()
```

Out[7]:

	country/region	continent	population	totalcases	newcases	totaldeaths	newdeaths
0	USA	North America	331,198,130.00	5032179	NaN	162,804.00	NaN
1	Brazil	South America	212,710,692.00	2917562	NaN	98,644.00	NaN
2	India	Asia	1,381,344,997.00	2025409	NaN	41,638.00	NaN
3	Russia	Europe	145,940,924.00	871894	NaN	14,606.00	NaN
4	South Africa	Africa	59,381,566.00	538184	NaN	9,604.00	NaN

What is the total population covered by the dataset, and which country/region and continent has the highest population?

In [8]:
```
df['population'].sum()
```

Out[8]:
```
6326421290.0
```

In [9]:
```
df[['country/region','population']].sort_values(by='population',ascending
=False)
```

Out[9]:

	country/region	population
2	India	1,381,344,997.00
0	USA	331,198,130.00
22	Indonesia	273,808,365.00
13	Pakistan	221,295,851.00
1	Brazil	212,710,692.00
...	...	...
205	Caribbean Netherlands	26,247.00
204	Montserrat	4,992.00
206	Falkland Islands	3,489.00
207	Vatican City	801.00
156	Diamond Princess	NaN

209 rows × 2 columns

In [10]:
```
df.groupby('continent')['population'].sum().sort_values(ascending=False)
```

Out[10]:
```
continent
Asia                3,173,656,415.00
Africa              1,343,515,489.00
Europe                747,677,546.00
North America         589,503,467.00
South America         431,110,464.00
Australia/Oceania      40,957,909.00
Name: population, dtype: float64
```

How many countries/regions are missing data for the "Continent" column, and can you identify them?

In [11]:
```
df[df['continent'].isna()]['country/region'].count()
```

Out[11]:
```
1
```

Calculate the total number of active cases, and what is the average number of active cases per country/region?

In [12]:
```
df['activecases'].sum()
```

Out[12]:
```
5671187.0
```

In [13]:
```
df.groupby('country/region')['activecases'].mean().sort_values(ascending=
False)
```

Out[13]:
```
country/region
USA                2,292,707.00
Brazil               771,258.00
India                606,387.00
Russia               180,931.00
Colombia             153,416.00
                        ...
Falkland Islands           0.00
Netherlands                 NaN
Spain                       NaN
Sweden                      NaN
UK                          NaN
Name: activecases, Length: 209, dtype: float64
```

How does the number of total tests conducted correlate with the number of total cases? Visualize the relationship.

In [14]:
```
df_corr=df[['totaltests','totalcases']].corr()
df_corr
```

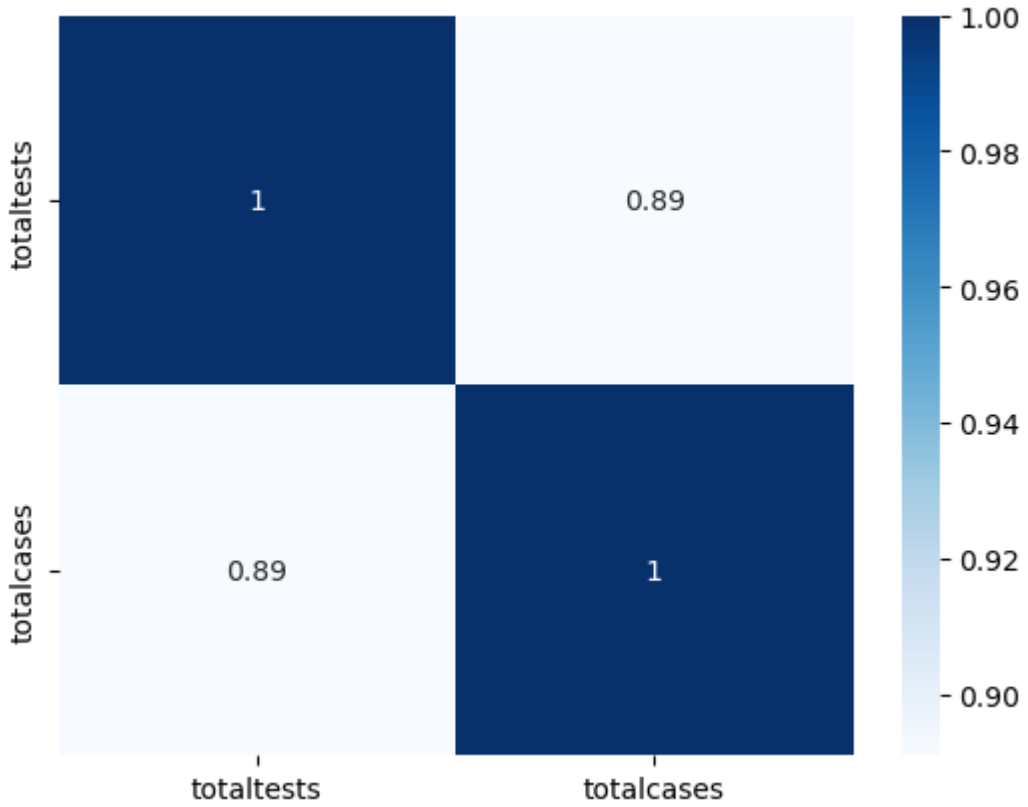
Out[14]:

	totaltests	totalcases
totaltests	1.00	0.89
totalcases	0.89	1.00

In [15]:
```
sns.heatmap(data=df_corr,annot=True,cmap='Blues')
```

Out[15]:

```
<Axes: >
```



What is the distribution of "Serious,Critical" cases by continent, and which continent has the highest average?

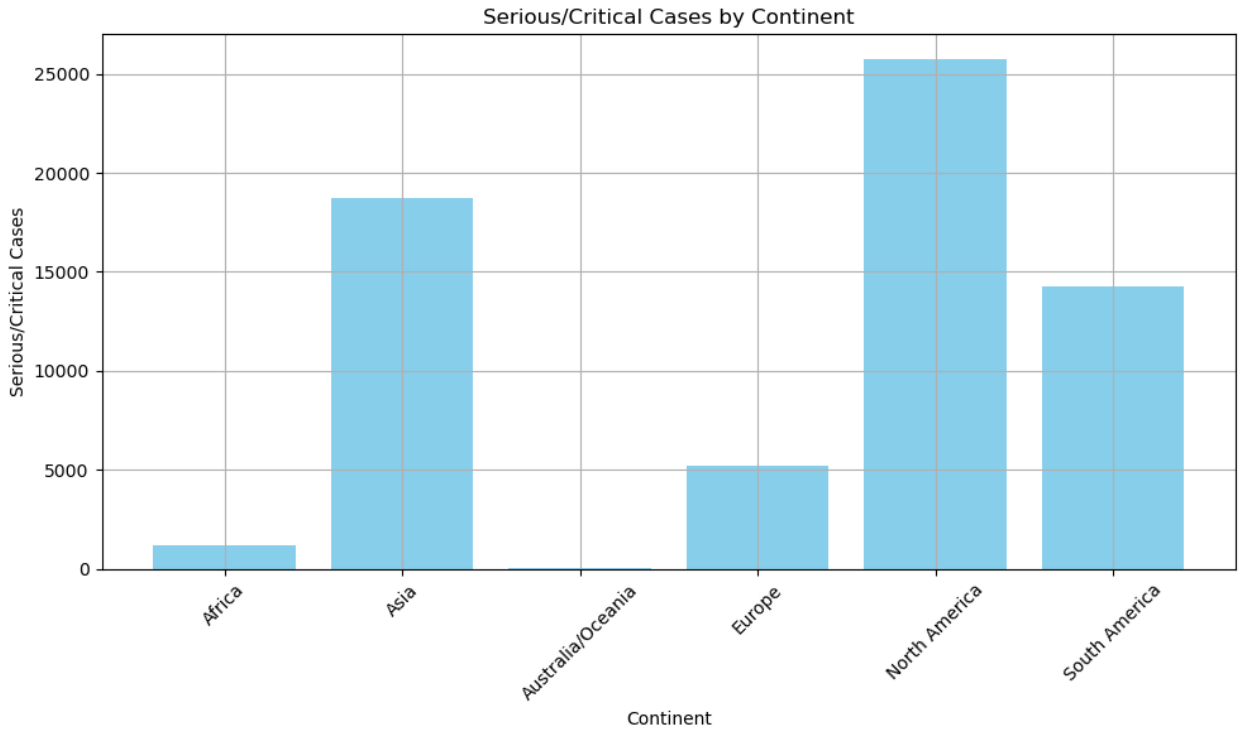
In [16]:
```
cases_dis=df.groupby('continent')['serious,critical'].sum().reset_index()
cases_dis
```

Out[16]:

	continent	serious,critical
0	Africa	1,187.00
1	Asia	18,749.00
2	Australia/Oceania	52.00
3	Europe	5,200.00
4	North America	25,709.00
5	South America	14,295.00

In [17]:

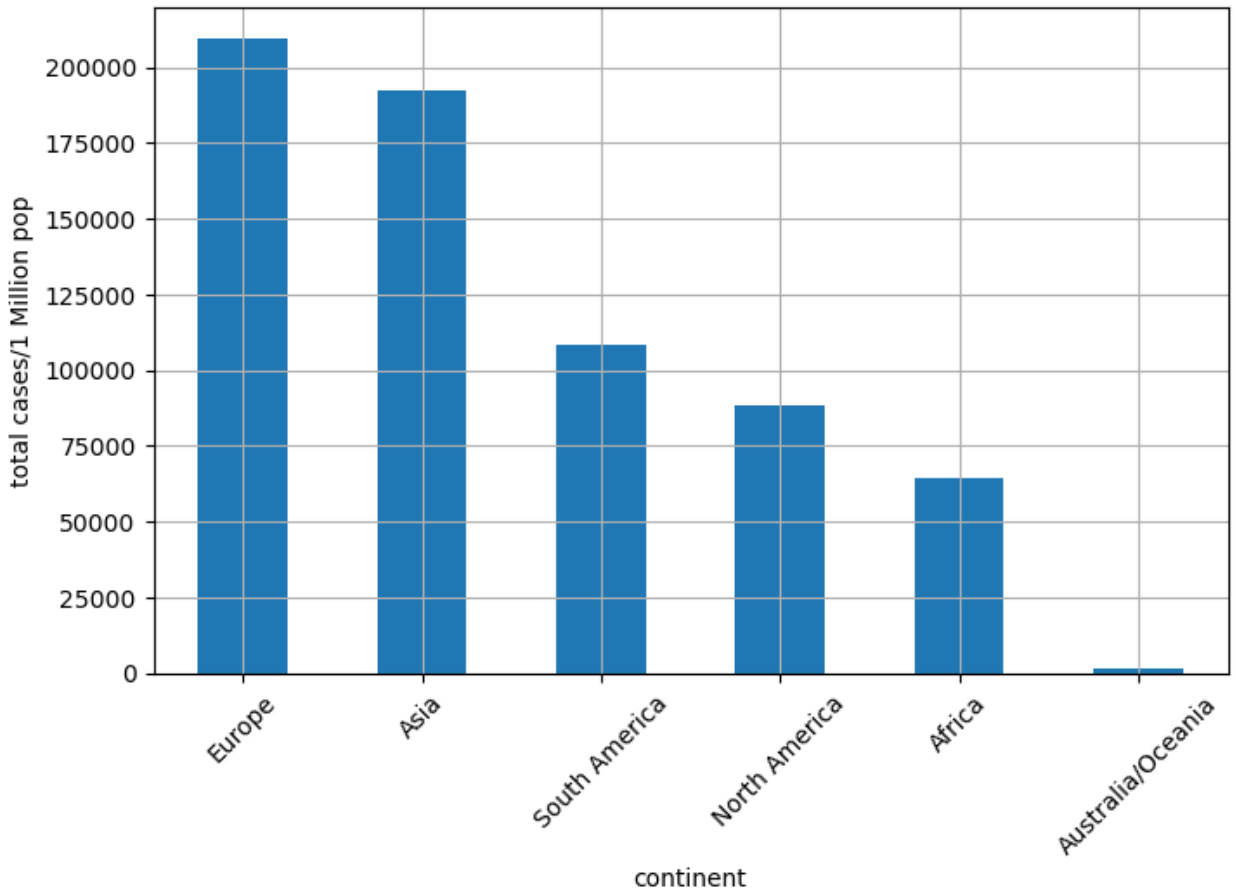
```
plt.figure(figsize=(10, 6))
plt.bar(cases_dis['continent'], cases_dis['serious,critical'], color='sky blue')
plt.xlabel('Continent')
plt.ylabel('Serious/Critical Cases')
plt.title('Serious/Critical Cases by Continent')
plt.xticks(rotation=45)
plt.grid()

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



Continents with the Highest Cases per Million Population:

In [18]:

```
plt.figure(figsize=(8,5))
df.groupby('continent')['tot cases/1m pop'].sum().sort_values(ascending=F
alse).plot(kind='bar')
plt.ylabel("total cases/1 Million pop")
plt.xticks(rotation=45)
plt.grid()
```



Identify the continents with the highest and lowest average cases per million population.

In [19]:
```
df.groupby('continent')['tot cases/1m pop'].mean().sort_values(ascending=
False)
```

Out[19]:
```
continent
South America       7,745.79
Europe              4,363.62
Asia                4,008.94
North America       2,529.91
Africa              1,130.81
Australia/Oceania     241.00
Name: tot cases/1m pop, dtype: float64
```

Calculate the percentage of serious/critical cases relative to the total cases for each country/region. and Identify countries/regions where a high percentage of cases are serious or critical.

In [20]:
```
df['serious % of total']=(df['serious,critical']/df['totalcases'])*100
x=df[['country/region','totalcases','serious % of total']].dropna()
x.sort_values(by='serious % of total',ascending=False)
```

Out[20]:

	country/region	totalcases	serious % of total
68	El Salvador	19126	2.66
189	Belize	86	2.33
183	Turks and Caicos	129	2.33
23	Canada	118561	1.91
192	Saint Martin	53	1.89
...	...	...	...
50	Ghana	39642	0.02
15	Italy	249204	0.02
45	Nigeria	45244	0.02
73	Denmark	14306	0.01
42	Guatemala	54339	0.01

122 rows × 3 columns

In [ ]:
```
```