Covid_World_Demographics

Table of Contents

Importing the libraries and data

In [1]:
```
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import numpy as np # linear algebra
import matplotlib as mpl # Data Visualization
import matplotlib.pyplot as plt # data Visualization
import seaborn as sns # data Visualization

# If using notebook, you need this to display plots

%matplotlib inline

# Or plt.show()for other editors or if you are using scripts (.py)
```

In [2]:
```
#Import the DATASET
covid_lat_long = pd.read_csv('COVID-19_geo_timeseries_ver_0311.csv')
```

EDA

In [3]:
```
# Exploring the first five rows of the dataset
covid_lat_long.head()
```

Out[3]:

	province	country	latitude	longitude	confirmed_cases	deaths	recovered	update_time	data_sc
0	Hubei	China	35.86166	104.195397	729	39.0	NaN	2020-01-24 14:55:00	
1	Guangdong	China	35.86166	104.195397	53	0.0	NaN	2020-01-24 14:55:00	
2	Zhejiang	China	35.86166	104.195397	43	0.0	NaN	2020-01-24 14:55:00	
3	Beijing	China	35.86166	104.195397	36	0.0	NaN	2020-01-24 14:55:00	
4	Chongqing	China	35.86166	104.195397	27	0.0	NaN	2020-01-24 14:55:00	

In [4]:
```
# Exploring the last five rows of the dataset
covid_lat_long.tail()
```

Out[4]:

	province	country	latitude	longitude	confirmed_cases	deaths	recovered	update_time	dat
24786	Nebraska	US	37.090240	-95.712891	5	0.0	0.0	3/11/2020 23:13	
24787	NaN	Reunion	NaN	NaN	1	0.0	0.0	3/11/2020 23:13	
24788	NaN	Cote d'Ivoire	NaN	NaN	1	0.0	0.0	3/11/2020 23:33	
24789	NaN	Greece	39.074208	21.824312	99	1.0	0.0	3/11/2020 23:53	
24790	California	US	37.090240	-95.712891	177	3.0	2.0	3/11/2020 23:53	

Checking the info of the data, to view the data types, columns, number of entries

In [5]:
```
# checking the info about the dataset
covid_lat_long.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 24791 entries, 0 to 24790
Data columns (total 12 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   province         22483 non-null  object 
 1   country          24791 non-null  object 
 2   latitude         24698 non-null  float64
 3   longitude        24698 non-null  float64
 4   confirmed_cases  24791 non-null  int64  
 5   deaths           24785 non-null  float64
 6   recovered        4935 non-null   float64
 7   update_time      24791 non-null  object 
 8   data_source      24791 non-null  object 
 9   country_code     24722 non-null  object 
 10  region           24722 non-null  object 
 11  country_flag     24722 non-null  object 
dtypes: float64(4), int64(1), object(7)
memory usage: 2.3+ MB
```

In [6]:
```
# checking the info again to verify the datatype modifies
covid_lat_long.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 24791 entries, 0 to 24790
Data columns (total 12 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   province         22483 non-null  object 
 1   country          24791 non-null  object 
 2   latitude         24698 non-null  float64
```

```
 3   longitude        24698 non-null  float64
 4   confirmed_cases  24791 non-null  int64
 5   deaths           24785 non-null  float64
 6   recovered        4935 non-null   float64
 7   update_time      24791 non-null  object
 8   data_source      24791 non-null  object
 9   country_code     24722 non-null  object
 10  region           24722 non-null  object
 11  country_flag     24722 non-null  object
dtypes: float64(4), int64(1), object(7)
memory usage: 2.3+ MB
```

So, we have 2205720 data points and 9 features.

In [7]:
```
# include paramater gives us option to show all features

covid_lat_long.describe(include='all').transpose()
```

Out[7]:

	count	unique	top	freq	mean	std
province	22483	252	Hunan	634	NaN	NaN
country	24791	140	China	21057	NaN	NaN
latitude	24698.0	NaN	NaN	NaN	34.902915	9.030786
longitude	24698.0	NaN	NaN	NaN	89.982701	47.900761
confirmed_cases	24791.0	NaN	NaN	NaN	506.282038	3203.5785
deaths	24785.0	NaN	NaN	NaN	12.104095	111.38341
recovered	4935.0	NaN	NaN	NaN	201.01155	2179.79852
update_time	24791	1824	2/1/2020 19:43	67	NaN	NaN
data_source	24791	3	dxy	18393	NaN	NaN
country_code	24722	112	CHN	21057	NaN	NaN
region	24722	5	Asia	22174	NaN	NaN
country_flag	24722	112	https://www.countryflags.io/cn/flat/64.png	21057	NaN	NaN

In [8]:
```
# checking the numbers of unique values in each column
covid_lat_long.nunique()
```

Out[8]:
```
province            252
country             140
latitude            113
longitude           113
confirmed_cases     689
deaths              102
recovered           516
update_time        1824
data_source           3
country_code        112
region                5
country_flag        112
dtype: int64
```

Handling missing values

In [9]:
```
covid_lat_long.isnull().sum()
```

Out[9]:
```
province            2308
country                0
latitude              93
longitude             93
confirmed_cases        0
deaths                 6
recovered          19856
update_time            0
data_source            0
country_code          69
region                69
country_flag          69
dtype: int64
```

In [10]:
```
# filling all missing values with zero(0)
covid_lat_long_filled = covid_lat_long.fillna(0)
```

In [11]:
```
covid_lat_long_filled.isnull().sum()
```

Out[11]:
```
province           0
country            0
latitude           0
longitude          0
confirmed_cases    0
deaths             0
recovered          0
update_time        0
data_source        0
country_code       0
region             0
country_flag       0
dtype: int64
```

In [12]:
```
# counting the number of times a case is recorded for each country
covid_lat_long['country'].value_counts()
```

Out[12]:
```
China                          21057
US                              1003
Australia                        253
Canada                           167
France                            79
                                ...
Iran (Islamic Republic of)         1
Czechia                            1
Congo (Kinshasa)                   1
Macao SAR                          1
Cote d'Ivoire                      1
Name: country, Length: 140, dtype: int64
```

In [13]:
```
# counting the number of times a case is recorded for each region
covid_lat_long['region'].value_counts()
```

Out[13]:
```
Asia        22174
```

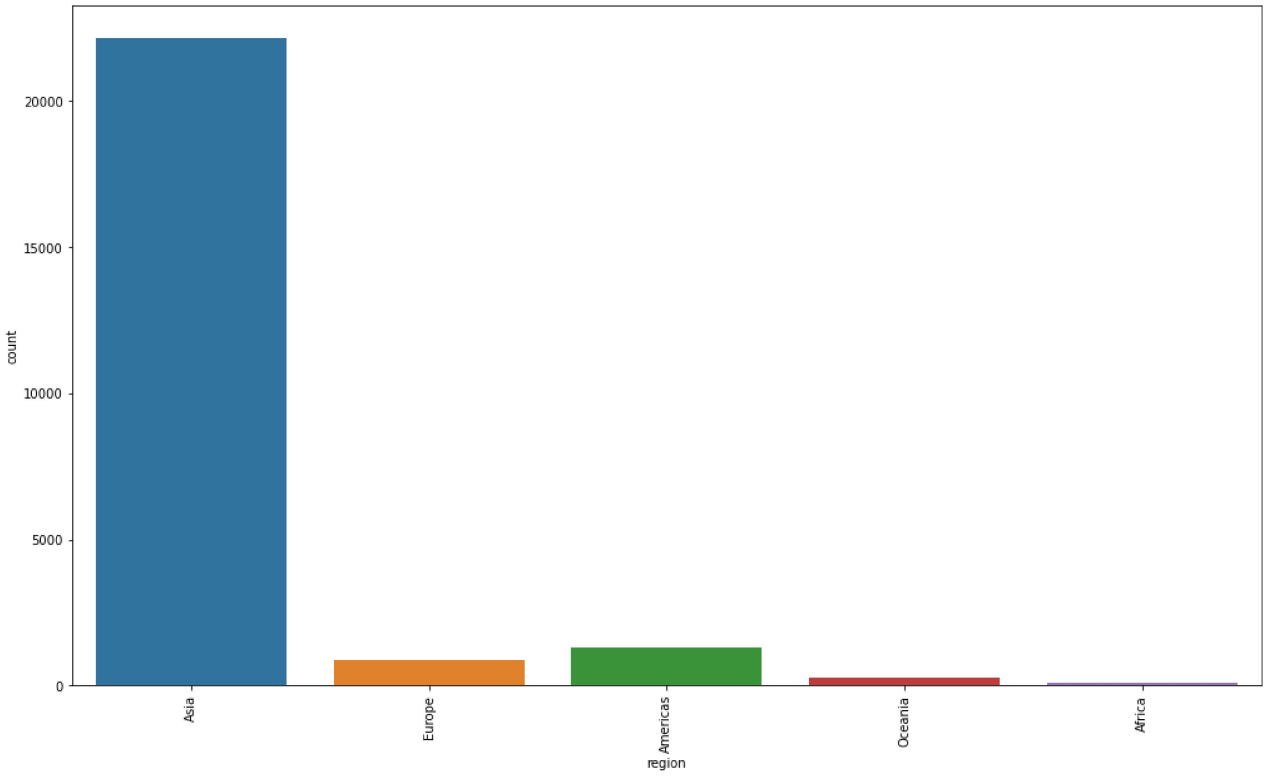
```
Americas     1297
Europe        878
Oceania       266
Africa        107
Name: region, dtype: int64
```

Visualization

Countplot Visualization

In [14]:
```
# function to plot a count plot; specifying the figure size, defining the plot paramete
def countplot(x):
    plt.rcParams['figure.figsize']=(17,10)
    sns.countplot(data=covid_lat_long, x=x)
    plt.xticks(rotation=90);
    return
```

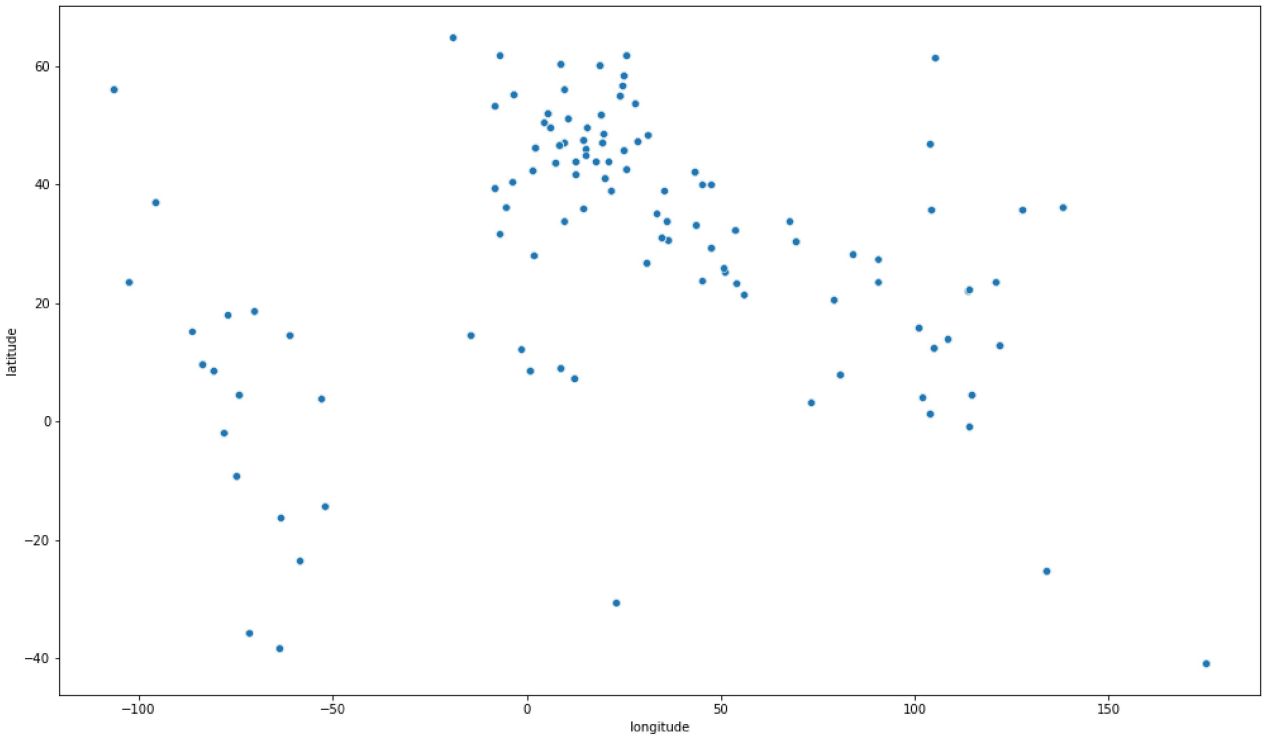
In [15]:
```
# calling the function to plot state_code countplot
countplot(covid_lat_long['region'])
```



Geographical Visualization

In [16]:
```
# function to plot a scatter plot; specifying the figure size, defining the plot
def covid(hue=None):
    plt.rcParams['figure.figsize']=(17,10)
    sns.scatterplot(data = covid_lat_long, x='longitude', y='latitude', hue = hue, size
    return
```

In [17]:
```
# scatter plot of all the location(lat-long)
covid()
```



Grouping the columns by 'country','latitude','longitude','country_code' to sum the cases, recovery and death that have the same lat, long, country and country_code

In [18]:
```
# Let's group covid_lat_long_geo by 'country','country_code'

covid_lat_long_geo = covid_lat_long.groupby(['country','country_code']).sum()
```

In [19]:
```
covid_lat_long_geo
```

Out[19]:

country	country_code	latitude	longitude	confirmed_cases	deaths	recovered
Afghanistan	**AFG**	576.964870	1151.069201	33	0.0	0.0
Albania	**ALB**	123.459996	60.504993	24	1.0	0.0
Algeria	**DZA**	448.542176	26.554016	151	0.0	0.0
Andorra	**AND**	425.462450	16.015540	10	0.0	0.0
Argentina	**ARG**	-345.744873	-572.550048	73	4.0	0.0
...	**...**	...	...	...	...	...
United Arab Emirates	**ARE**	1007.235268	2315.456174	717	0.0	144.0
United Kingdom	**GBR**	0.000000	0.000000	459	8.0	19.0
United States	**USA**	1075.616960	-2775.673839	173	0.0	0.0

		latitude	longitude	confirmed_cases	deaths	recovered
country	**country_code**					
Viet Nam	**VNM**	0.000000	0.000000	31	0.0	16.0
Vietnam	**VNM**	1082.490948	8337.344323	750	0.0	374.0

124 rows × 5 columns

In [20]:
```
# reeting the grouped indexed columns
covid_lat_long_geo_reset = covid_lat_long_geo.reset_index()
```

In [21]:
```
covid_lat_long_geo_reset
```

Out[21]:

	country	country_code	latitude	longitude	confirmed_cases	deaths	recovered
0	Afghanistan	AFG	576.964870	1151.069201	33	0.0	0.0
1	Albania	ALB	123.459996	60.504993	24	1.0	0.0
2	Algeria	DZA	448.542176	26.554016	151	0.0	0.0
3	Andorra	AND	425.462450	16.015540	10	0.0	0.0
4	Argentina	ARG	-345.744873	-572.550048	73	4.0	0.0
...	...	...	...	...	...	...	...
119	United Arab Emirates	ARE	1007.235268	2315.456174	717	0.0	144.0
120	United Kingdom	GBR	0.000000	0.000000	459	8.0	19.0
121	United States	USA	1075.616960	-2775.673839	173	0.0	0.0
122	Viet Nam	VNM	0.000000	0.000000	31	0.0	16.0
123	Vietnam	VNM	1082.490948	8337.344323	750	0.0	374.0

124 rows × 7 columns

In [22]:
```
covid_lat_long_geo_reset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 124 entries, 0 to 123
Data columns (total 7 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   country          124 non-null    object 
 1   country_code     124 non-null    object 
 2   latitude         124 non-null    float64
 3   longitude        124 non-null    float64
 4   confirmed_cases  124 non-null    int64  
 5   deaths           124 non-null    float64
 6   recovered        124 non-null    float64
dtypes: float64(4), int64(1), object(2)
memory usage: 6.9+ KB
```

In [23]:
```
covid_lat_long_geo_reset['country'].value_counts()
```

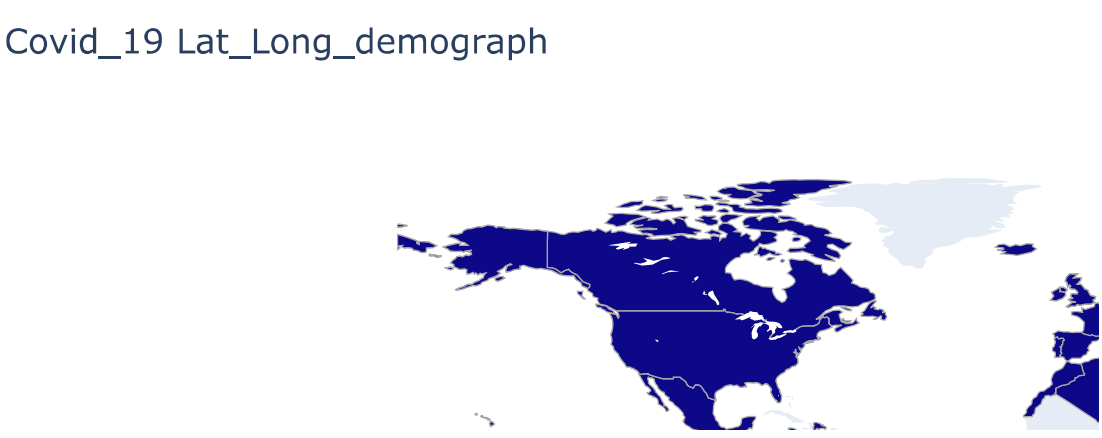
Out[23]:
```
Moldova             1
South Korea         1
Australia           1
Singapore           1
Chile               1
                   ..
Austria             1
Finland             1
Congo (Kinshasa)    1
Belarus             1
Nepal               1
Name: country, Length: 124, dtype: int64
```

Visualizing geo location of cases and death recorded

In [24]:
```
import plotly.graph_objects as go

fig = go.Figure(data=go.Choropleth(
    locations = covid_lat_long_geo_reset['country_code'],
    z = covid_lat_long_geo_reset['confirmed_cases'],
    text = covid_lat_long_geo_reset['country'],
    colorscale = 'Blues',
    autocolorscale=True,
    reversescale=False,
    marker_line_color='darkgray',
    marker_line_width=0.5,
    colorbar_tickprefix = '',
    colorbar_title = 'Covid_19<br>confirmed_cases count',
))

fig.update_layout(
    title_text='Covid_19 Lat_Long_demograph',
    geo=dict(
        showframe=False,
        showcoastlines=False,
        projection_type='equirectangular'
    )
    )
fig.show()
```



In [25]:

```
fig = go.Figure(data=go.Choropleth(
    locations = covid_lat_long_geo_reset['country_code'],
    z = covid_lat_long_geo_reset['recovered'],
    text = covid_lat_long_geo_reset['country'],
    colorscale = 'Blues',
    autocolorscale=True,
    reversescale=False,
    marker_line_color='darkgray',
    marker_line_width=0.5,
    colorbar_tickprefix = '',
    colorbar_title = 'Covid_19<br>recovered_count',
))

fig.update_layout(
    title_text='Covid_19 Lat_Long_demograph',
    geo=dict(
        showframe=False,
        showcoastlines=False,
        projection_type='equirectangular'
    )
    )
fig.show()
```

Covid_19 Lat_Long_demograph

In [26]:

```
fig = go.Figure(data=go.Choropleth(
    locations = covid_lat_long_geo_reset['country_code'],
    z = covid_lat_long_geo_reset['deaths'],
    text = covid_lat_long_geo_reset['country'],
    colorscale = 'Blues',
    autocolorscale=True,
    reversescale=False,
    marker_line_color='darkgray',
    marker_line_width=0.5,
    colorbar_tickprefix = '',
    colorbar_title = 'Covid_19<br>death recorded count',
))

fig.update_layout(
    title_text='Covid_19 Lat_Long_demoograph',
    geo=dict(
        showframe=False,
        showcoastlines=False,
        projection_type='equirectangular'
    )
    )
fig.show()
```

Covid_19 Lat_Long_demoograph

Go To Top