Covid 19 (Real Project)

In [1]:
```
import pandas as pd
import numpy as np
import math
import matplotlib.pyplot as plt
```

In [2]:
```
df=pd.read_csv('covid_19_data.csv',parse_dates=True,index_col=0)
```

In [3]:
```
df.head()#The head() method returns a specified number of rows, string fro
#The head() method returns the first 5 rows if a number is not specified.
```

Out[3]:

Date	State	Region	Confirmed	Deaths	Recovered
2020-04-29	NaN	Afghanistan	1939	60	252
2020-04-29	NaN	Albania	766	30	455
2020-04-29	NaN	Algeria	3848	444	1702
2020-04-29	NaN	Andorra	743	42	423
2020-04-29	NaN	Angola	27	2	7

In [4]:
```
df.tail()  #The tail function in Python displays the last five rows of the
#We can use this parameter to display the number of rows of our choice.
```

Out[4]:

Date	State	Region	Confirmed	Deaths	Recovered
2020-04-29	Wyoming	US	545	7	0
2020-04-29	Xinjiang	Mainland China	76	3	73
2020-04-29	Yukon	Canada	11	0	0
2020-04-29	Yunnan	Mainland China	185	2	181
2020-04-29	Zhejiang	Mainland China	1268	1	1263

In [5]:
```
df.shape
```

Out[5]: (321, 5)

In [6]:
```
1 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 321 entries, 2020-04-29 to 2020-04-29
Data columns (total 5 columns):
 #   Column     Non-Null Count  Dtype 
---  ------     --------------  ----- 
 0   State      140 non-null    object
 1   Region     321 non-null    object
 2   Confirmed  321 non-null    int64 
 3   Deaths     321 non-null    int64 
 4   Recovered  321 non-null    int64 
dtypes: int64(3), object(2)
memory usage: 15.0+ KB
```

In [7]:
```
1 df.count()
```

Out[7]:
```
State        140
Region       321
Confirmed    321
Deaths       321
Recovered    321
dtype: int64
```

In [8]:
```
1 df.describe()    # not normal distribution data
```

Out[8]:

	Confirmed	Deaths	Recovered
count	321.000000	321.000000	321.000000
mean	9949.800623	709.152648	3030.277259
std	31923.853086	3236.162817	14364.870365
min	0.000000	0.000000	0.000000
25%	104.000000	2.000000	2.000000
50%	653.000000	12.000000	73.000000
75%	4655.000000	144.000000	587.000000
max	299691.000000	27682.000000	132929.000000

In [9]:
```
1 df.isnull()
```

Out[9]:

Date	State	Region	Confirmed	Deaths	Recovered
2020-04-29	True	False	False	False	False
2020-04-29	True	False	False	False	False
2020-04-29	True	False	False	False	False
2020-04-29	True	False	False	False	False
2020-04-29	True	False	False	False	False
...	...	...	...	...	...
2020-04-29	False	False	False	False	False
2020-04-29	False	False	False	False	False
2020-04-29	False	False	False	False	False
2020-04-29	False	False	False	False	False
2020-04-29	False	False	False	False	False

321 rows × 5 columns

In [10]:
```
1 df.isnull().sum()
```

Out[10]:
```
State        181
Region         0
Confirmed      0
Deaths         0
Recovered      0
dtype: int64
```

Q1-Show the Number of Confirmed,Death and Recovered cases in each Region

In [11]:

```
ggroup = df['Region'].groupby([df['Deaths'],df['Recovered']])
ggroup.groups
```

Out[11]:
{(0, 0): [2020-04-29 00:00:00, 2020-04-29 00:00:00, 2020-04-29 00:00:00, 2020
-04-29 00:00:00, 2020-04-29 00:00:00, 2020-04-29 00:00:00, 2020-04-29 00:00:0
0, 2020-04-29 00:00:00, 2020-04-29 00:00:00, 2020-04-29 00:00:00], (0, 1): [2
020-04-29 00:00:00, 2020-04-29 00:00:00], (0, 2): [2020-04-29 00:00:00], (0,
3): [2020-04-29 00:00:00], (0, 4): [2020-04-29 00:00:00, 2020-04-29 00:00:0
0], (0, 5): [2020-04-29 00:00:00, 2020-04-29 00:00:00], (0, 6): [2020-04-29 0
0:00:00, 2020-04-29 00:00:00, 2020-04-29 00:00:00], (0, 7): [2020-04-29 00:0
0:00], (0, 8): [2020-04-29 00:00:00, 2020-04-29 00:00:00], (0, 10): [2020-04-
29 00:00:00, 2020-04-29 00:00:00], (0, 11): [2020-04-29 00:00:00, 2020-04-29
00:00:00], (0, 12): [2020-04-29 00:00:00, 2020-04-29 00:00:00], (0, 13): [202
0-04-29 00:00:00, 2020-04-29 00:00:00], (0, 15): [2020-04-29 00:00:00], (0, 1
6): [2020-04-29 00:00:00], (0, 17): [2020-04-29 00:00:00], (0, 18): [2020-04-
29 00:00:00], (0, 19): [2020-04-29 00:00:00], (0, 25): [2020-04-29 00:00:00],
(0, 34): [2020-04-29 00:00:00], (0, 50): [2020-04-29 00:00:00], (0, 52): [202
0-04-29 00:00:00], (0, 75): [2020-04-29 00:00:00], (0, 90): [2020-04-29 00:0
0:00], (0, 98): [2020-04-29 00:00:00], (0, 119): [2020-04-29 00:00:00], (0, 1
31): [2020-04-29 00:00:00], (0, 164): [2020-04-29 00:00:00], (0, 181): [2020-
04-29 00:00:00], (0, 222): [2020-04-29 00:00:00], (0, 300): [2020-04-29 00:0
0:00], (0, 648): [2020-04-29 00:00:00], (0, 20327): [2020-04-29 00:00:00],
(0, 120720): [2020-04-29 00:00:00], (1, 0): [2020-04-29 00:00:00], (1, 2): [2
020-04-29 00:00:00, 2020-04-29 00:00:00], (1, 3): [2020-04-29 00:00:00], (1,
4): [2020-04-29 00:00:00], (1, 5): [2020-04-29 00:00:00, 2020-04-29 00:00:0
0], (1, 6): [2020-04-29 00:00:00], (1, 8): [2020-04-29 00:00:00, 2020-04-29 0
0:00:00], (1, 9): [2020-04-29 00:00:00], (1, 10): [2020-04-29 00:00:00, 2020-
04-29 00:00:00], (1, 13): [2020-04-29 00:00:00], (1, 17): [2020-04-29 00:00:0
0], (1, 19): [2020-04-29 00:00:00], (1, 33): [2020-04-29 00:00:00], (1, 55):
[2020-04-29 00:00:00], (1, 93): [2020-04-29 00:00:00], (1, 99): [2020-04-29 0
0:00:00], (1, 124): [2020-04-29 00:00:00], (1, 150): [2020-04-29 00:00:00],
(1, 353): [2020-04-29 00:00:00], (1, 936): [2020-04-29 00:00:00], (1, 1263):
[2020-04-29 00:00:00], (2, 0): [2020-04-29 00:00:00, 2020-04-29 00:00:00],
(2, 7): [2020-04-29 00:00:00], (2, 9): [2020-04-29 00:00:00], (2, 18): [2020-
04-29 00:00:00], (2, 19): [2020-04-29 00:00:00], (2, 71): [2020-04-29 00:00:0
0], (2, 73): [2020-04-29 00:00:00], (2, 137): [2020-04-29 00:00:00], (2, 14
3): [2020-04-29 00:00:00], (2, 145): [2020-04-29 00:00:00], (2, 181): [2020-0
4-29 00:00:00], (2, 252): [2020-04-29 00:00:00], (2, 599): [2020-04-29 00:00:
00], (3, 0): [2020-04-29 00:00:00, 2020-04-29 00:00:00], (3, 7): [2020-04-29
00:00:00, 2020-04-29 00:00:00], (3, 11): [2020-04-29 00:00:00], (3, 21): [202
0-04-29 00:00:00], (3, 24): [2020-04-29 00:00:00], (3, 54): [2020-04-29 00:0
0:00], (3, 58): [2020-04-29 00:00:00], (3, 67): [2020-04-29 00:00:00], (3, 7
3): [2020-04-29 00:00:00], (3, 101): [2020-04-29 00:00:00], (3, 183): [2020-0
4-29 00:00:00], (3, 253): [2020-04-29 00:00:00], (3, 558): [2020-04-29 00:00:
00], (4, 0): [2020-04-29 00:00:00], (4, 5): [2020-04-29 00:00:00], (4, 12):
[2020-04-29 00:00:00], (4, 58): [2020-04-29 00:00:00], (4, 235): [2020-04-29
00:00:00], (4, 339): [2020-04-29 00:00:00], (4, 420): [2020-04-29 00:00:00],
(4, 830): [2020-04-29 00:00:00], (4, 1015): [2020-04-29 00:00:00], (5, 0): [2
020-04-29 00:00:00], (6, 0): [2020-04-29 00:00:00, 2020-04-29 00:00:00], (6,
8): [2020-04-29 00:00:00], (6, 27): [2020-04-29 00:00:00], (6, 48): [2020-04-
29 00:00:00], (6, 162): [2020-04-29 00:00:00], (6, 178): [2020-04-29 00:00:0
0], (6, 311): [2020-04-29 00:00:00], (6, 318): [2020-04-29 00:00:00], (6, 32
3): [2020-04-29 00:00:00], ...}

In [12]:
```
ggroup.sum()
```

Out[12]:
```
Deaths  Recovered
0       0            Papua New GuineaSouth SudanNetherlandsUSCanada...
        1                                              YemenMainland China
        2                                                        Holy See
        3                                                              UK
        4                    Saint Kitts and NevisSao Tome and Principe
                                                 ...
23477   0                                                              US
24087   48228                                                    France
24275   132929                                                    Spain
26097   0                                                              UK
27682   71252                                                     Italy
Name: Region, Length: 291, dtype: object
```

Q2- Remove all the Records where Confirmed cases is less than 10

In [13]:
```
cases_10 = df[df['Confirmed']==10]
cases_10.count()
```

Out[13]:
```
State        0
Region       3
Confirmed    3
Deaths       3
Recovered    3
dtype: int64
```

In [14]:
```
Assigne_confirmed_less_than_10 = df[df['Confirmed']>10].count()
Assigne_confirmed_less_than_10
```

Out[14]:
```
State        130
Region       301
Confirmed    301
Deaths       301
Recovered    301
dtype: int64
```

In [15]:
```
df.count()
```

Out[15]:
```
State        140
Region       321
Confirmed    321
Deaths       321
Recovered    321
dtype: int64
```

In [16]:
```
Remove_less_than_10 = df[df['Confirmed']>10]
Remove_less_than_10
```

Out[16]:

Date	State	Region	Confirmed	Deaths	Recovered
2020-04-29	NaN	Afghanistan	1939	60	252
2020-04-29	NaN	Albania	766	30	455
2020-04-29	NaN	Algeria	3848	444	1702
2020-04-29	NaN	Andorra	743	42	423
2020-04-29	NaN	Angola	27	2	7
...	...	...	...	...	...
2020-04-29	Wyoming	US	545	7	0
2020-04-29	Xinjiang	Mainland China	76	3	73
2020-04-29	Yukon	Canada	11	0	0
2020-04-29	Yunnan	Mainland China	185	2	181
2020-04-29	Zhejiang	Mainland China	1268	1	1263

301 rows × 5 columns

In [17]:
```
Remove_less_than_10.count()
```

Out[17]:
```
State        130
Region       301
Confirmed    301
Deaths       301
Recovered    301
dtype: int64
```

Q3-in which region , maximum number of confirmed cases were recomended?

In [18]:
```
Death_maximum_regions = df['Deaths'].groupby(df['Region'])
Death_maximum_regions.max()
#which indicates that maximum death record were in Algeria Region.
```

Out[18]:
```
Region
Afghanistan            60
Albania                30
Algeria               444
Andorra                42
Angola                  2
                      ...
West Bank and Gaza      2
Western Sahara          0
Yemen                   0
Zambia                  3
Zimbabwe                4
Name: Deaths, Length: 187, dtype: int64
```

Q4-in which region, minimum number of Deaths Cases where recorded?

In [19]:
```
Death_minumum_regions = df['Deaths'].groupby(df['Region'])
Death_minumum_regions.count()
# Which indicates that minmum deaths record were in Western Sahara and Yam
```

Out[19]:
```
Region
Afghanistan           1
Albania               1
Algeria               1
Andorra               1
Angola                1
                     ..
West Bank and Gaza    1
Western Sahara        1
Yemen                 1
Zambia                1
Zimbabwe              1
Name: Deaths, Length: 187, dtype: int64
```

Q5-How Many Confirmed , Deaths and Recovered cases were reported from India till April 2020?

In [20]:
```
1 df['Region'].unique()
```

Out[20]:
```
array(['Afghanistan', 'Albania', 'Algeria', 'Andorra', 'Angola',
       'Antigua and Barbuda', 'Argentina', 'Armenia', 'Austria',
       'Azerbaijan', 'Bahamas', 'Bahrain', 'Bangladesh', 'Barbados',
       'Belarus', 'Belgium', 'Belize', 'Benin', 'Bhutan', 'Bolivia',
       'Bosnia and Herzegovina', 'Botswana', 'Brazil', 'Brunei',
       'Bulgaria', 'Burkina Faso', 'Burma', 'Burundi', 'Cabo Verde',
       'Cambodia', 'Cameroon', 'Central African Republic', 'Chad',
       'Chile', 'Colombia', 'Congo (Brazzaville)', 'Congo (Kinshasa)',
       'Costa Rica', 'Croatia', 'Cuba', 'Cyprus', 'Czech Republic',
       'Denmark', 'Diamond Princess', 'Djibouti', 'Dominica',
       'Dominican Republic', 'Ecuador', 'Egypt', 'El Salvador',
       'Equatorial Guinea', 'Eritrea', 'Estonia', 'Eswatini', 'Ethiopia',
       'Fiji', 'Finland', 'France', 'Gabon', 'Gambia', 'Georgia',
       'Germany', 'Ghana', 'Greece', 'Grenada', 'Guatemala', 'Guinea',
       'Guinea-Bissau', 'Guyana', 'Haiti', 'Holy See', 'Honduras',
       'Hungary', 'Iceland', 'India', 'Indonesia', 'Iran', 'Iraq',
       'Ireland', 'Israel', 'Italy', 'Ivory Coast', 'Jamaica', 'Japan',
       'Jordan', 'Kazakhstan', 'Kenya', 'Kosovo', 'Kuwait', 'Kyrgyzstan',
       'Laos', 'Latvia', 'Lebanon', 'Liberia', 'Libya', 'Liechtenstein',
```

In [22]:
```
1 india_cases = df[df['Region']=='India']
2 india_cases
```

Out[22]:

Date	State	Region	Confirmed	Deaths	Recovered
2020-04-29	NaN	India	33062	1079	8437

Q6-sort the entire data with of recovered cases in descending sort.

In [50]:
```
1 Sort_Recovered = df.sort_values('Recovered',ascending = False)
2 Sort_Recovered
```

Out[50]:

Date	State	Region	Confirmed	Deaths	Recovered
2020-04-29	NaN	Spain	236899	24275	132929
2020-04-29	Recovered	US	0	0	120720
2020-04-29	NaN	Germany	161539	6467	120400
2020-04-29	NaN	Iran	93657	5957	73791
2020-04-29	NaN	Italy	203591	27682	71252
...	...	...	...	...	...
2020-04-29	Maryland	US	20849	1078	0
2020-04-29	Manitoba	Canada	275	6	0
2020-04-29	Louisiana	US	27660	1845	0
2020-04-29	Kentucky	US	4537	234	0
2020-04-29	Grand Princess	Canada	13	0	0

321 rows × 5 columns

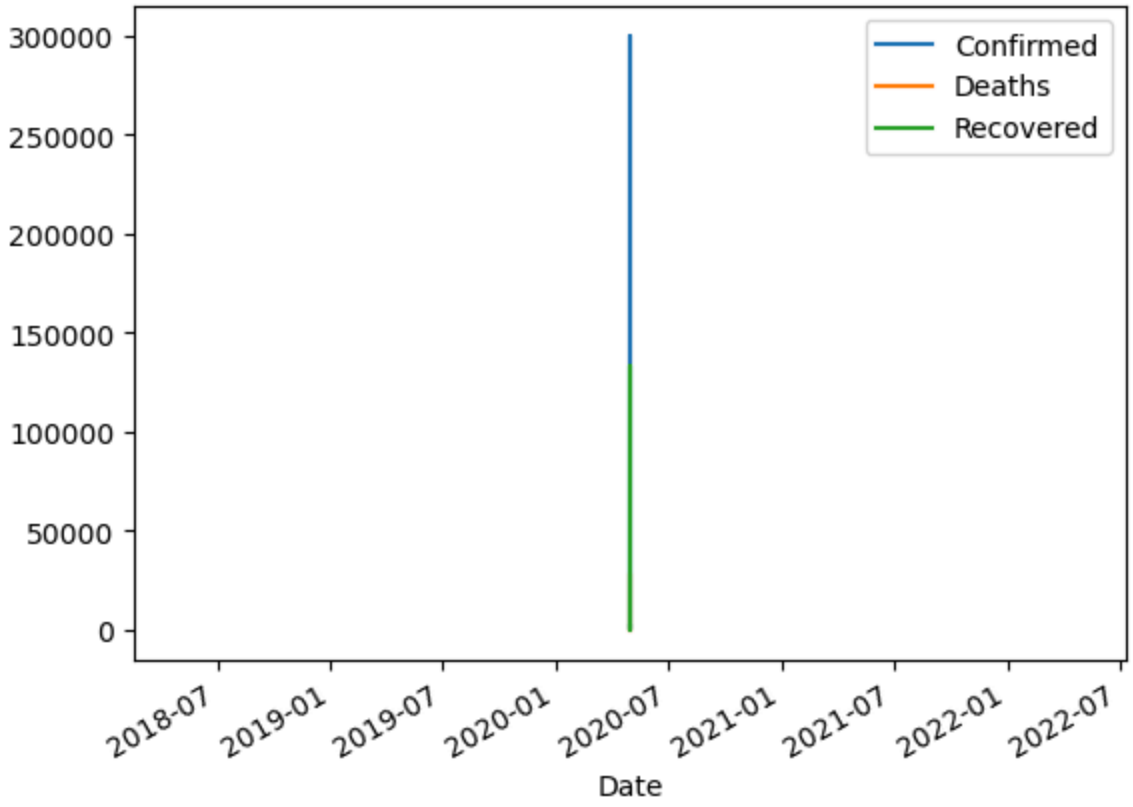
In [51]:
```
1 df.describe()
```

Out[51]:

	Confirmed	Deaths	Recovered
count	321.000000	321.000000	321.000000
mean	9949.800623	709.152648	3030.277259
std	31923.853086	3236.162817	14364.870365
min	0.000000	0.000000	0.000000
25%	104.000000	2.000000	2.000000
50%	653.000000	12.000000	73.000000
75%	4655.000000	144.000000	587.000000
max	299691.000000	27682.000000	132929.000000

In [49]:
```
1 df.plot()
```

Out[49]: <AxesSubplot:xlabel='Date'>



In [ ]:
```
1
```