In [10]:

```
import pandas as pd
import os
import warnings
warnings.filterwarnings('ignore')

path = ".₩COVID-19-master₩csse_covid_19_data₩csse_covid_19_daily_reports"
file_list = os.listdir(path)
file_list_csv = [file for file in file_list if file.endswith(".csv")]


df=pd.read_csv(path+'/'+file_list_csv[0],encoding='ms949')
ObservationDate = []
date = file_list_csv[0]
date = date.split('.')[0]
date = date.replace('-', '/')
for i in range(len(df)):
    ObservationDate.append(date)
ObservationDate = list(ObservationDate)
print(ObservationDate)

for item in file_list_csv[1:160]:
    s = item.split('.')[0]
    s = s.replace('-', '/')
    a=pd.read_csv(path+'/'+item,encoding='ms949')
    a.columns = a.columns.str.replace('Country_', 'Country/')
    a.columns = a.columns.str.replace('Province_', 'Province/')
    a.columns = a.columns.str.replace('Last_', 'Last ')
    for i in range(1, len(a)+1):
        ObservationDate.append(s)
    df=pd.concat([df,a],axis=0)
```

```
['01/22/2020', '01/22/2020', '01/22/2020', '01/22/2020', '01/22/2020', '01/22/202
0', '01/22/2020', '01/22/2020', '01/22/2020', '01/22/2020', '01/22/2020', '01/22/2
020', '01/22/2020', '01/22/2020', '01/22/2020', '01/22/2020', '01/22/2020', '01/2
2/2020', '01/22/2020', '01/22/2020', '01/22/2020', '01/22/2020', '01/22/2020', '0
1/22/2020', '01/22/2020', '01/22/2020', '01/22/2020', '01/22/2020', '01/22/2020',
'01/22/2020', '01/22/2020', '01/22/2020', '01/22/2020', '01/22/2020', '01/22/202
0', '01/22/2020', '01/22/2020', '01/22/2020']
```

In [11]:

```
path = ".₩COVID-19-master₩csse_covid_19_data₩csse_covid_19_daily_reports"
file_list = os.listdir(path)
file_list_csv = [file for file in file_list if file.endswith(".csv")]
#print(file_list_csv)
print(len(file_list_csv))
print(file_list_csv.index('06-30-2020.csv'))
#os.getcwd()
```

```
171
160
```

In [12]:

```
df.head()
```

Out[12]:

	Active	Admin2	Case-Fatality_Ratio	Combined_Key	Confirmed	Country/Region	Deaths	FIPS
0	NaN	NaN	NaN	NaN	1.0	Mainland China	NaN	NaN
1	NaN	NaN	NaN	NaN	14.0	Mainland China	NaN	NaN
2	NaN	NaN	NaN	NaN	6.0	Mainland China	NaN	NaN
3	NaN	NaN	NaN	NaN	1.0	Mainland China	NaN	NaN
4	NaN	NaN	NaN	NaN	NaN	Mainland China	NaN	NaN

In [13]:

```
df.tail()
```

Out[13]:

	Active	Admin2	Case-Fatality_Ratio	Combined_Key	Confirmed	Country/Region	Deaths	FI
3780	1733.0	NaN	0.228833	West Bank and Gaza	2185.0	West Bank and Gaza	5.0	N
3781	1.0	NaN	10.000000	Western Sahara	10.0	Western Sahara	1.0	N
3782	392.0	NaN	26.950355	Yemen	1128.0	Yemen	304.0	N
3783	235.0	NaN	1.403061	Zambia	1568.0	Zambia	22.0	N
3784	415.0	NaN	1.219512	Zimbabwe	574.0	Zimbabwe	7.0	N

In [14]:

```
df.describe()
```

Out[14]:

	Active	Case-Fatality_Ratio	Confirmed	Deaths	FIPS	Incidenc
count	333446.000000	116899.000000	341044.000000	340622.000000	292260.000000	116277.
mean	697.709047	3.595770	1333.754316	79.127156	31216.426151	371.
std	9382.242074	19.544937	9826.974885	910.859839	16965.220763	648.
min	-793690.000000	0.000000	0.000000	0.000000	66.000000	0.
25%	2.000000	0.000000	5.000000	0.000000	18147.000000	68.
50%	21.000000	1.754386	31.000000	1.000000	29161.000000	175.
75%	143.000000	5.000000	195.000000	6.000000	46047.000000	421.
max	249015.000000	3766.666667	405843.000000	41128.000000	99999.000000	13133.

In [15]:

```
len(ObservationDate)
```

Out[15]:

341063

In [16]:

```
ObservationDate[344800:]
```

Out[16]:

[]

In [17]:

```
list(df.columns)
```

Out[17]:

```
['Active',
 'Admin2',
 'Case-Fatality_Ratio',
 'Combined_Key',
 'Confirmed',
 'Country/Region',
 'Deaths',
 'FIPS',
 'Incidence_Rate',
 'Last Update',
 'Lat',
 'Latitude',
 'Long_',
 'Longitude',
 'Province/State',
 'Recovered']
```

In [18]:

```
covid=df[['Province/State','Country/Region','Last Update','Confirmed',
          'Deaths','Recovered']]
```

In [19]:

```
covid[covid['Country/Region']=='Mainland China']
```

Out[19]:

	Province/State	Country/Region	Last Update	Confirmed	Deaths	Recovered
0	Anhui	Mainland China	1/22/2020 17:00	1.0	NaN	NaN
1	Beijing	Mainland China	1/22/2020 17:00	14.0	NaN	NaN
2	Chongqing	Mainland China	1/22/2020 17:00	6.0	NaN	NaN
3	Fujian	Mainland China	1/22/2020 17:00	1.0	NaN	NaN
4	Gansu	Mainland China	1/22/2020 17:00	NaN	NaN	NaN
...	...	...	...	...	...	...
178	Tibet	Mainland China	2020-02-12T06:43:02	1.0	0.0	1.0
206	Gansu	Mainland China	2020-03-11T02:18:28	0.0	0.0	0.0
207	Hebei	Mainland China	2020-03-11T02:18:29	0.0	0.0	0.0
211	Gansu	Mainland China	2020-03-11T02:18:28	0.0	0.0	0.0
212	Hebei	Mainland China	2020-03-11T02:18:29	0.0	0.0	0.0

1517 rows × 6 columns

In [20]:

```
covid[covid['Country/Region']=='China']
```

Out[20]:

	Province/State	Country/Region	Last Update	Confirmed	Deaths	Recovered
0	Hubei	China	2020-03-11T10:53:02	67773.0	3046.0	49134.0
7	Guangdong	China	2020-03-11T10:13:06	1356.0	8.0	1282.0
8	Henan	China	2020-03-11T08:13:09	1273.0	22.0	1249.0
9	Zhejiang	China	2020-03-11T09:33:12	1215.0	1.0	1195.0
10	Hunan	China	2020-03-11T02:18:14	1018.0	4.0	995.0
...	...	...	...	...	...	...
3543	Tianjin	China	2020-06-30 04:33:48	198.0	3.0	194.0
3544	Tibet	China	2020-06-30 04:33:48	1.0	0.0	1.0
3601	Xinjiang	China	2020-06-30 04:33:48	76.0	3.0	73.0
3609	Yunnan	China	2020-06-30 04:33:48	185.0	2.0	183.0
3614	Zhejiang	China	2020-06-30 04:33:48	1269.0	1.0	1267.0

3663 rows × 6 columns

In [21]:

```
covid['Country/Region'] = covid['Country/Region'].replace('China', 'Mainland China')
```

In [22]:

```
covid[covid['Country/Region']=='Mainland China']
```

Out[22]:

	Province/State	Country/Region	Last Update	Confirmed	Deaths	Recovered
0	Anhui	Mainland China	1/22/2020 17:00	1.0	NaN	NaN
1	Beijing	Mainland China	1/22/2020 17:00	14.0	NaN	NaN
2	Chongqing	Mainland China	1/22/2020 17:00	6.0	NaN	NaN
3	Fujian	Mainland China	1/22/2020 17:00	1.0	NaN	NaN
4	Gansu	Mainland China	1/22/2020 17:00	NaN	NaN	NaN
...	...	...	...	...	...	...
3543	Tianjin	Mainland China	2020-06-30 04:33:48	198.0	3.0	194.0
3544	Tibet	Mainland China	2020-06-30 04:33:48	1.0	0.0	1.0
3601	Xinjiang	Mainland China	2020-06-30 04:33:48	76.0	3.0	73.0
3609	Yunnan	Mainland China	2020-06-30 04:33:48	185.0	2.0	183.0
3614	Zhejiang	Mainland China	2020-06-30 04:33:48	1269.0	1.0	1267.0

5180 rows × 6 columns

In [23]:

```
covid.head()
```

Out[23]:

	Province/State	Country/Region	Last Update	Confirmed	Deaths	Recovered
0	Anhui	Mainland China	1/22/2020 17:00	1.0	NaN	NaN
1	Beijing	Mainland China	1/22/2020 17:00	14.0	NaN	NaN
2	Chongqing	Mainland China	1/22/2020 17:00	6.0	NaN	NaN
3	Fujian	Mainland China	1/22/2020 17:00	1.0	NaN	NaN
4	Gansu	Mainland China	1/22/2020 17:00	NaN	NaN	NaN

In [24]:

```
print("Size/Shape of the dataset: ", covid.shape)
print("Checking for null vales:₩n", covid.isnull().sum())
print("Checking Data-type of each column:₩n", covid.dtypes)
```

```
Size/Shape of the dataset:  (341063, 6)
Checking for null vales:
 Province/State    20888
Country/Region         0
Last Update            0
Confirmed             19
Deaths               441
Recovered            388
dtype: int64
Checking Data-type of each column:
 Province/State     object
Country/Region     object
Last Update        object
Confirmed         float64
Deaths            float64
Recovered         float64
dtype: object
```

In [25]:

```
covid['Deaths'] = covid['Deaths'].fillna(0)
covid['Recovered'] = covid['Recovered'].fillna(0)
covid['Confirmed'] = covid['Confirmed'].fillna(0)
```

In [26]:

```
covid.describe()
```

Out[26]:

	Confirmed	Deaths	Recovered
count	341063.000000	341063.000000	341063.000000
mean	1333.680015	79.024843	548.382604
std	9826.706201	910.275209	7993.529666
min	0.000000	0.000000	0.000000
25%	5.000000	0.000000	0.000000
50%	31.000000	1.000000	0.000000
75%	195.000000	6.000000	0.000000
max	405843.000000	41128.000000	705203.000000

In [27]:

```
s=len(covid)
len(list(covid.columns))
covid.shape
new = range(1, s+1)
covid['SNo']=list(new)
```

In [28]:

```
print(covid.shape); print(len(ObservationDate))
```

```
(341063, 7)
341063
```

In [29]:

```
covid['ObservationDate'] = ObservationDate
```

In [30]:

```
covid.head()
```

Out[30]:

	Province/State	Country/Region	Last Update	Confirmed	Deaths	Recovered	SNo	Observati
0	Anhui	Mainland China	1/22/2020 17:00	1.0	0.0	0.0	1	01/2
1	Beijing	Mainland China	1/22/2020 17:00	14.0	0.0	0.0	2	01/2
2	Chongqing	Mainland China	1/22/2020 17:00	6.0	0.0	0.0	3	01/2
3	Fujian	Mainland China	1/22/2020 17:00	1.0	0.0	0.0	4	01/2
4	Gansu	Mainland China	1/22/2020 17:00	0.0	0.0	0.0	5	01/2

In [31]:

```
#Dropping column as SNo is of no use, and "Province/State" contains too many missing values
covid.drop(["SNo"], 1, inplace=True)
```

In [32]:

```
#Converting "Observation Date" into Datetime format
covid["ObservationDate"]=pd.to_datetime(covid["ObservationDate"])
```

In [33]:

```
grouped_country = covid.groupby(["Country/Region", "ObservationDate"]).agg(
    {"Confirmed":'sum',"Recovered":'sum',"Deaths":'sum'})
grouped_country
```

Out[33]:

		Confirmed	Recovered	Deaths
Country/Region	**ObservationDate**			
Azerbaijan	2020-02-28	1.0	0.0	0.0
Afghanistan	2020-02-24	1.0	0.0	0.0
	2020-02-25	1.0	0.0	0.0
	2020-02-26	1.0	0.0	0.0
	2020-02-27	1.0	0.0	0.0
...	...	...	...	...
occupied Palestinian territory	2020-03-12	0.0	0.0	0.0
	2020-03-14	0.0	0.0	0.0
	2020-03-15	0.0	0.0	0.0
	2020-03-16	0.0	0.0	0.0
	2020-03-17	0.0	0.0	0.0

22269 rows × 3 columns

In [34]:

```
grouped_country["Active Cases"] = grouped_country[
    "Confirmed"]-grouped_country["Recovered"]-grouped_country["Deaths"]
```

In [35]:

```
import numpy as np
```

In [36]:

```
grouped_country["log_confirmed"] = np.log(grouped_country["Confirmed"])
grouped_country["log_active"] = np.log(grouped_country["Active Cases"])
```

In [37]:

```
grouped_country
```

Out[37]:

Country/Region	ObservationDate	Confirmed	Recovered	Deaths	Active Cases	log_confirmed	log_
Azerbaijan	2020-02-28	1.0	0.0	0.0	1.0	0.0	
Afghanistan	2020-02-24	1.0	0.0	0.0	1.0	0.0	
	2020-02-25	1.0	0.0	0.0	1.0	0.0	
	2020-02-26	1.0	0.0	0.0	1.0	0.0	
	2020-02-27	1.0	0.0	0.0	1.0	0.0	
...	...	...	...	...	...	...	
occupied Palestinian territory	2020-03-12	0.0	0.0	0.0	0.0	-inf	
	2020-03-14	0.0	0.0	0.0	0.0	-inf	
	2020-03-15	0.0	0.0	0.0	0.0	-inf	
	2020-03-16	0.0	0.0	0.0	0.0	-inf	
	2020-03-17	0.0	0.0	0.0	0.0	-inf	

22269 rows × 6 columns

In [38]:

```
grouped_country[1:100]
```

Out[38]:

Country/Region	ObservationDate	Confirmed	Recovered	Deaths	Active Cases	log_confirmed	log
	2020-02-24	1.0	0.0	0.0	1.0	0.000000	0
	2020-02-25	1.0	0.0	0.0	1.0	0.000000	0
	2020-02-26	1.0	0.0	0.0	1.0	0.000000	0
	2020-02-27	1.0	0.0	0.0	1.0	0.000000	0
	2020-02-28	1.0	0.0	0.0	1.0	0.000000	0
Afghanistan	...	...	...	...	...	...	
	2020-05-28	13036.0	1209.0	235.0	11592.0	9.475470	9
	2020-05-29	13659.0	1259.0	246.0	12154.0	9.522154	9
	2020-05-30	14525.0	1303.0	249.0	12973.0	9.583627	9
	2020-05-31	15205.0	1328.0	257.0	13620.0	9.629380	9
	2020-06-01	15750.0	1428.0	265.0	14057.0	9.664596	9

99 rows × 6 columns

Datewise analysis

In [39]:

```
#Grouping different types of cases as per the date cumulatively
datewise = covid.groupby(
    ["ObservationDate"]).agg({"Confirmed":'sum',"Recovered":'sum',"Deaths":'sum'})
datewise["Days Since"] = datewise.index - datewise.index.min()
datewise.head()
```

Out[39]:

ObservationDate	Confirmed	Recovered	Deaths	Days Since
2020-01-22	555.0	28.0	17.0	0 days
2020-01-23	653.0	30.0	18.0	1 days
2020-01-24	941.0	36.0	26.0	2 days
2020-01-25	1438.0	39.0	42.0	3 days
2020-01-26	2118.0	52.0	56.0	4 days

In [40]:

```
print("Basic Information")

print()

print("Total number of countries with Disease Spread: ", len(covid["Country/Region"].unique()))
print("Total number of Confirmed Cases around the World: ", datewise["Confirmed"].iloc[-1])
print("Total number of Recovered around the world: ",datewise["Recovered"].iloc[-1] )
print("Total number of Deaths Cases around the world: ", datewise["Deaths"].iloc[-1])
print("Total number of Active Cases around the world: ", (datewise["Confirmed"].iloc[-1] - datew
ise[
    "Recovered"].iloc[-1] - datewise["Deaths"].iloc[-1])) #Confirmed - Recovered - Deaths
print("Total number of Closed Cases around the world: ", datewise["Recovered"].iloc[-1]+datewise
["Deaths"].iloc[-1])

print()

print("Approximate number of Confirmed Cases per Day around the world: ",
     np.round(datewise["Confirmed"].iloc[-1]/datewise.shape[0]))
print("Approximate number of Recovered Cases per Day around the world: ",
     np.round(datewise["Recovered"].iloc[-1]/datewise.shape[0]))
print("Approximate number of Death Cases per Day around the world: ",
     np.round(datewise["Deaths"].iloc[-1]/datewise.shape[0]))

print()

print("Approximate number of Confirmed Cases per hour around the world: ",
     np.round(datewise["Confirmed"].iloc[-1]/(datewise.shape[0]*24)))
print("Approximate number of Recovered Cases per hour around the word: ",
     np.round(datewise["Recovered"].iloc[-1]/(datewise.shape[0]*24)))
print("Approximate number of Death Cases per hour around the world: ",
     np.round(datewise["Deaths"].iloc[-1]/(datewise.shape[0]*24)))

print()

print("Number of Confirmed Cases in last 24 hours: ", datewise["Confirmed"].iloc[-1]-datewise["C
onfirmed"].iloc[-2])
print("Number of Recovered Cases in last 24 hours: ", datewise["Recovered"].iloc[-1]-datewise["R
ecovered"].iloc[-2])
print("Number of Death Cases in last 24 hours: ", datewise["Deaths"].iloc[-1]-datewise["Deaths"]
.iloc[-2])
```

```
Basic Information

Total number of countries with Disease Spread:  237
Total number of Confirmed Cases around the World:  10302151.0
Total number of Recovered around the world:  5235813.0
Total number of Deaths Cases around the world:  505505.0
Total number of Active Cases around the world:  4560833.0
Total number of Closed Cases around the world:  5741318.0

Approximate number of Confirmed Cases per Day around the world:  64388.0
Approximate number of Recovered Cases per Day around the world:  32724.0
Approximate number of Death Cases per Day around the world:  3159.0

Approximate number of Confirmed Cases per hour around the world:  2683.0
Approximate number of Recovered Cases per hour around the word:  1363.0
Approximate number of Death Cases per hour around the world:  132.0

Number of Confirmed Cases in last 24 hours:  156360.0
Number of Recovered Cases in last 24 hours:  94914.0
Number of Death Cases in last 24 hours:  3612.0
```

In [41]:

```
datewise.shape
```

Out[41]:

```
(160, 4)
```

In [42]:

```
datewise.describe
```

Out[42]:

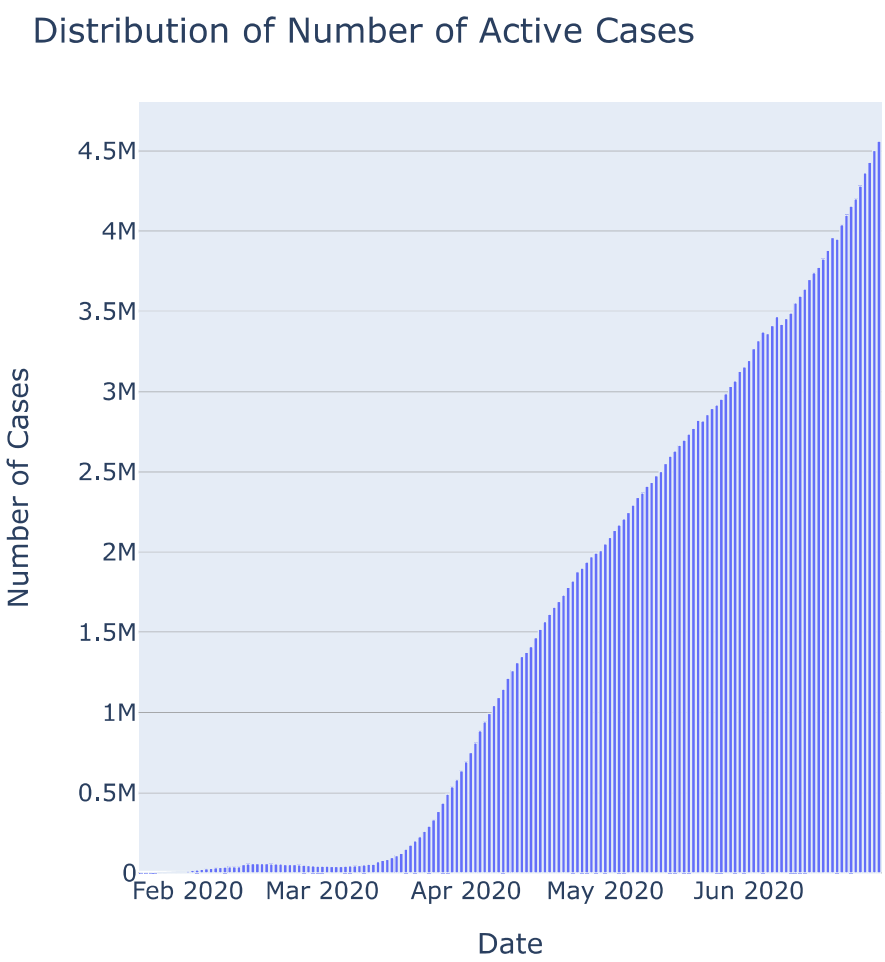
```
<bound method NDFrame.describe of                 Confirmed  Recovered    Deaths
Days Since
ObservationDate
2020-01-22              555.0       28.0      17.0     0 days
2020-01-23              653.0       30.0      18.0     1 days
2020-01-24              941.0       36.0      26.0     2 days
2020-01-25             1438.0       39.0      42.0     3 days
2020-01-26             2118.0       52.0      56.0     4 days
...                       ...        ...       ...        ...
2020-06-25          9609829.0  4838921.0  489312.0   155 days
2020-06-26          9801572.0  4945557.0  494181.0   156 days
2020-06-27          9979535.0  5051864.0  498710.0   157 days
2020-06-28         10145791.0  5140899.0  501893.0   158 days
2020-06-29         10302151.0  5235813.0  505505.0   159 days

[160 rows x 4 columns]>
```

In [43]:

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

In [44]:

```
fig = px.bar(x=datewise.index, y=datewise["Confirmed"]-datewise["Recovered"]-datewise["Deaths"])
fig.update_layout(title="Distribution of Number of Active Cases",
                  xaxis_title="Date", yaxis_title="Number of Cases",)
fig.show()
```

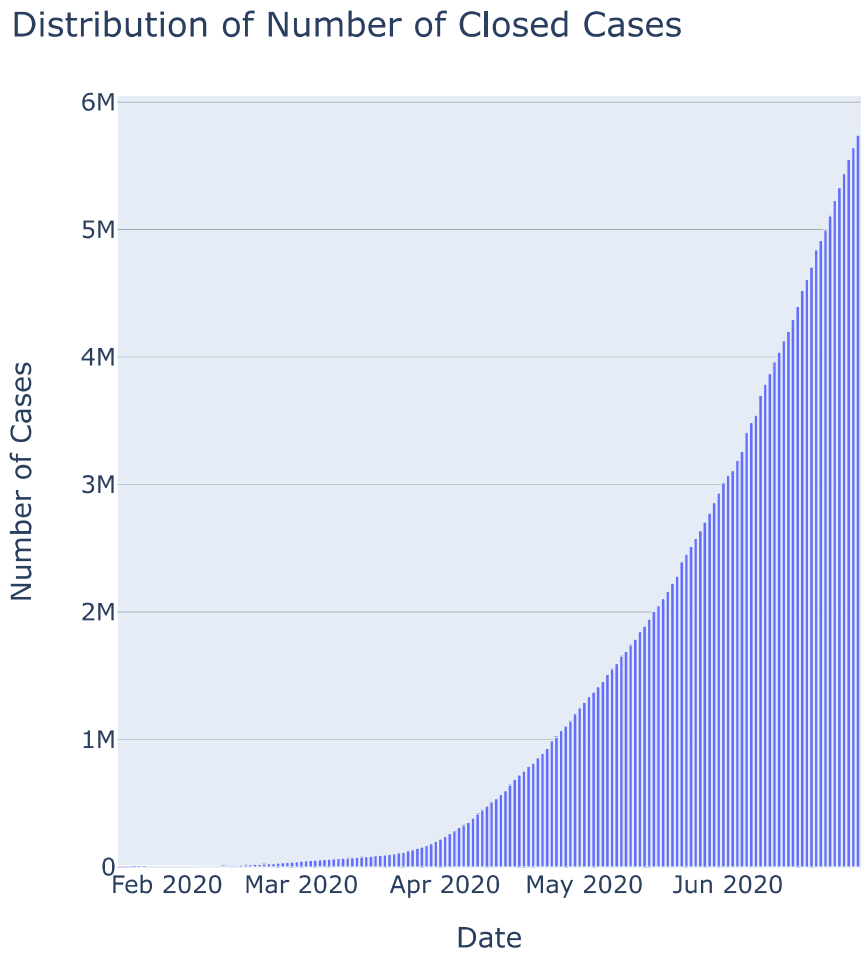


Active Cases = Number of Confirmed Cases - Number of Recovered Cases - Number of Death Cases

Increase in number of Active Cases is probably an indication of Recovered case or Death case number is dropping in comparison to number of Confirmed Cases drastically. Will look for the conclusive evidence for the same in the notebook ahead.

In [45]:

```
fig2 = px.bar(x=datewise.index, y=datewise["Recovered"]+datewise["Deaths"])
fig2.update_layout(title = "Distribution of Number of Closed Cases",
                   xaxis_title = "Date", yaxis_title = "Number of Cases")
fig2.show()
```



Closed Cases = Number of Recovered Cases + Number of Death Cases

Increase in number of Closed classes imply wither more patients are getting recovered from the disease or more people are dying because of COVID-19

In [46]:

```
datewise["WeekOfYear"]=datewise.index.weekofyear
datewise
```

Out[46]:

ObservationDate	Confirmed	Recovered	Deaths	Days Since	WeekOfYear
2020-01-22	555.0	28.0	17.0	0 days	4
2020-01-23	653.0	30.0	18.0	1 days	4
2020-01-24	941.0	36.0	26.0	2 days	4
2020-01-25	1438.0	39.0	42.0	3 days	4
2020-01-26	2118.0	52.0	56.0	4 days	4
...	...	...	...	...	...
2020-06-25	9609829.0	4838921.0	489312.0	155 days	26
2020-06-26	9801572.0	4945557.0	494181.0	156 days	26
2020-06-27	9979535.0	5051864.0	498710.0	157 days	26
2020-06-28	10145791.0	5140899.0	501893.0	158 days	26
2020-06-29	10302151.0	5235813.0	505505.0	159 days	27

160 rows × 5 columns

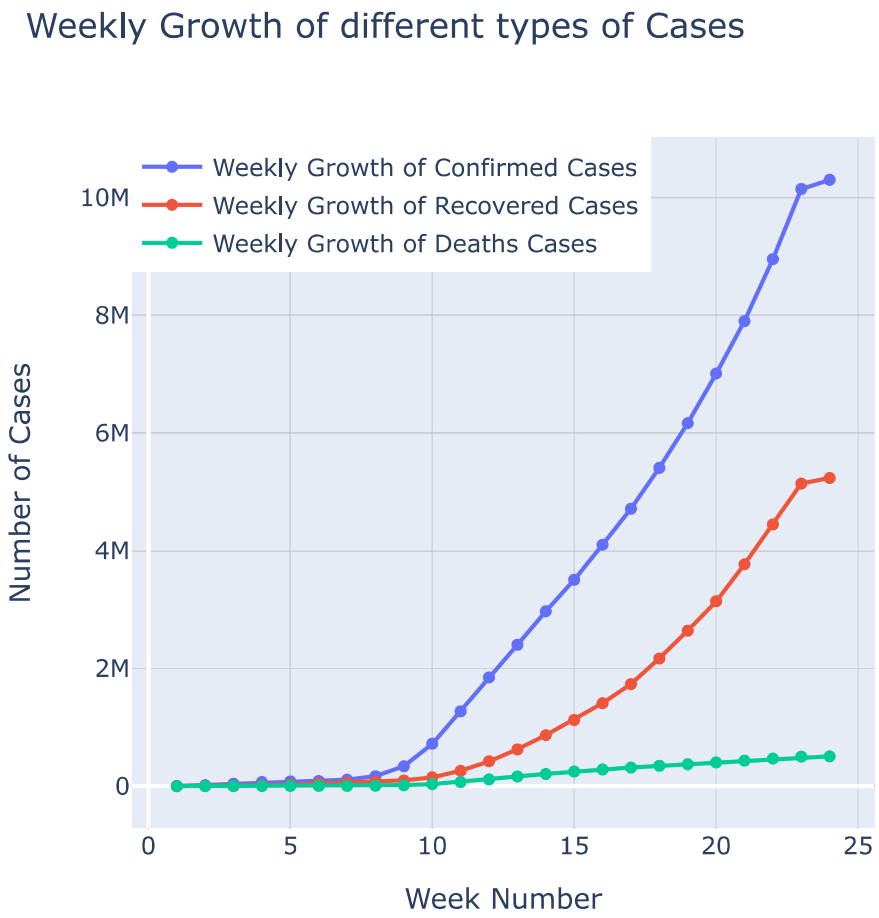
In [47]:

```
week_num=[]
weekwise_confirmed=[]
weekwise_recovered=[]
weekwise_deaths=[]
w=1
for i in list(datewise["WeekOfYear"].unique()):
    weekwise_confirmed.append(datewise[datewise["WeekOfYear"]==i]["Confirmed"].iloc[-1])
    weekwise_recovered.append(datewise[datewise["WeekOfYear"]==i]["Recovered"].iloc[-1])
    weekwise_deaths.append(datewise[datewise["WeekOfYear"]==i]["Deaths"].iloc[-1])
    week_num.append(w)
    w+=1
print(len(datewise["WeekOfYear"].unique()))
```

24

In [48]:

```
import plotly.graph_objects as go
fig=go.Figure()
fig.add_trace(go.Scatter(x=week_num, y=weekwise_confirmed,
                         mode="lines+markers",
                         name = "Weekly Growth of Confirmed Cases"))
fig.add_trace(go.Scatter(x=week_num, y=weekwise_recovered,
                         mode="lines+markers",
                         name='Weekly Growth of Recovered Cases'))
fig.add_trace(go.Scatter(x=week_num, y=weekwise_deaths,
                         mode='lines+markers',
                         name='Weekly Growth of Deaths Cases'))
fig.update_layout(title='Weekly Growth of different types of Cases',
                  xaxis_title="Week Number", yaxis_title="Number of Cases",
                   legend=dict(x=0, y=1, traceorder="normal"))
```



In [49]:

```
datewise
```

Out[49]:

ObservationDate	Confirmed	Recovered	Deaths	Days Since	WeekOfYear
2020-01-22	555.0	28.0	17.0	0 days	4
2020-01-23	653.0	30.0	18.0	1 days	4
2020-01-24	941.0	36.0	26.0	2 days	4
2020-01-25	1438.0	39.0	42.0	3 days	4
2020-01-26	2118.0	52.0	56.0	4 days	4
...	...	...	...	...	...
2020-06-25	9609829.0	4838921.0	489312.0	155 days	26
2020-06-26	9801572.0	4945557.0	494181.0	156 days	26
2020-06-27	9979535.0	5051864.0	498710.0	157 days	26
2020-06-28	10145791.0	5140899.0	501893.0	158 days	26
2020-06-29	10302151.0	5235813.0	505505.0	159 days	27

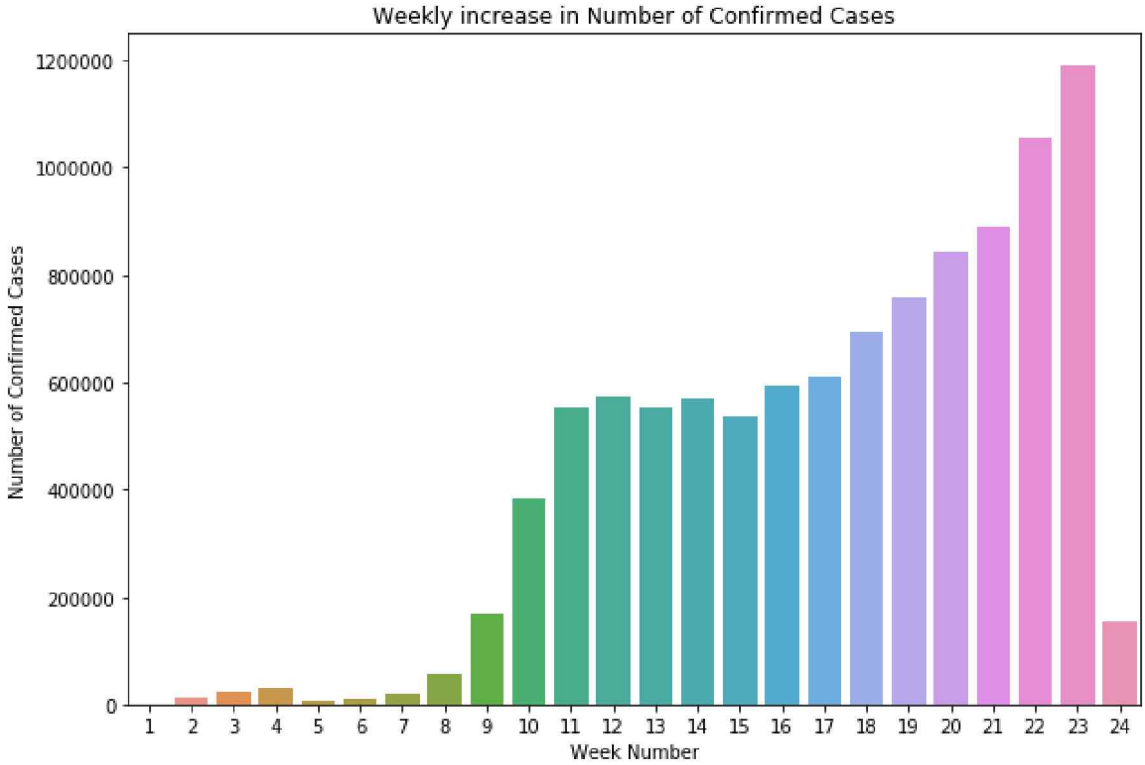
160 rows × 5 columns

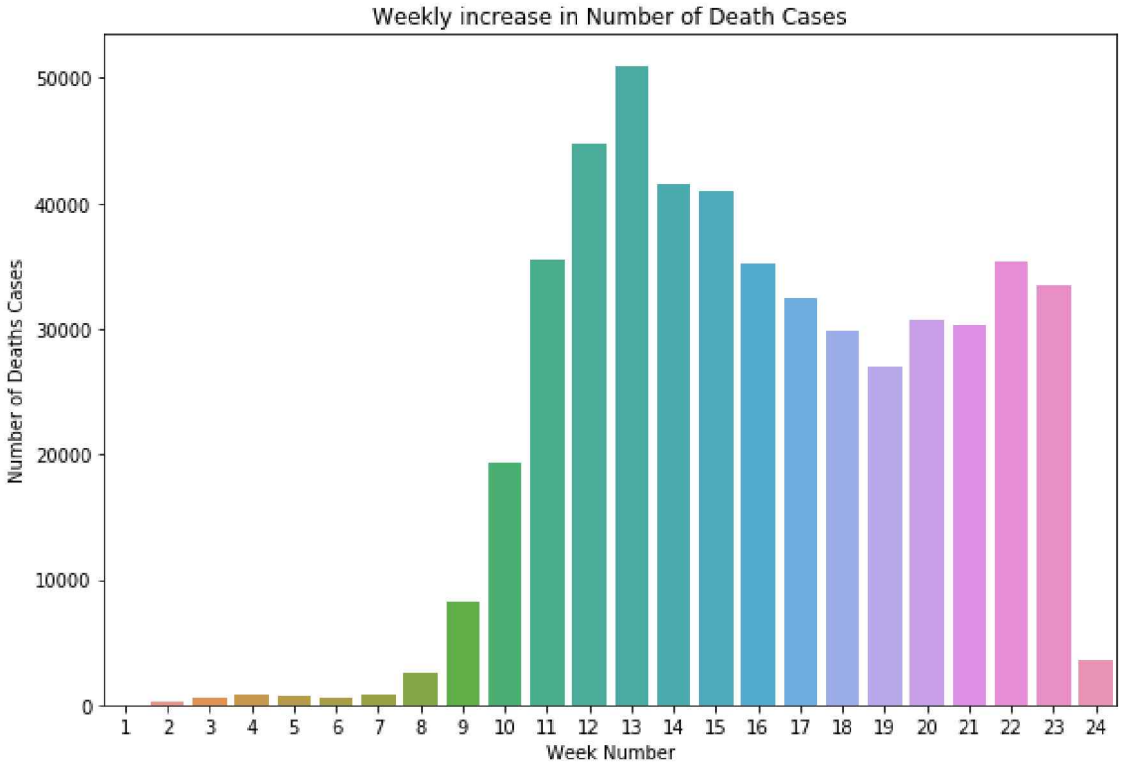
In [50]:

```
# View entire dataframe
pd.set_option('display.max_rows', None)
pd.set_option('display.max_columns', None)
pd.set_option('display.width', None)
pd.set_option('display.max_colwidth', -1)
```

In [52]:

```
import seaborn as sns
import matplotlib.pyplot as plt

fig,(ax1, ax2) = plt.subplots(2, 1, figsize=(10, 15))
sns.barplot(x=week_num, y=pd.Series(weekwise_confirmed).diff().fillna(0), ax=ax1)
sns.barplot(x=week_num, y=pd.Series(weekwise_deaths).diff().fillna(0), ax=ax2)
ax1.set_xlabel("Week Number")
ax2.set_xlabel("Week Number")
ax1.set_ylabel("Number of Confirmed Cases")
ax2.set_ylabel("Number of Deaths Cases")
ax1.set_title("Weekly increase in Number of Confirmed Cases")
ax2.set_title("Weekly increase in Number of Death Cases")

fig.show()
```


Weekly increase in Number of Confirmed Cases
Number of Confirmed Cases
0
200000
400000
600000
800000
1000000
1200000
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
Week Number


Weekly increase in Number of Death Cases
Number of Deaths Cases
0
10000
20000
30000
40000
50000
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
Week Number

In [48]:

```
datewise["WeekOfYear"].unique()[20]
datewise[datewise["WeekOfYear"]==24]
```

Out[48]:

ObservationDate	Confirmed	Recovered	Deaths	Days Since	WeekOfYear
2020-06-08	7119002.0	3293408.0	406543.0	138 days	24
2020-06-09	7242313.0	3375673.0	411436.0	139 days	24
2020-06-10	7360239.0	3454807.0	416201.0	140 days	24
2020-06-11	7514481.0	3540696.0	421458.0	141 days	24
2020-06-12	7632802.0	3613277.0	425394.0	142 days	24
2020-06-13	7766952.0	3698304.0	429736.0	143 days	24
2020-06-14	7900924.0	3769712.0	433066.0	144 days	24

In [49]:

```
datewise.tail()
```

Out[49]:

ObservationDate	Confirmed	Recovered	Deaths	Days Since	WeekOfYear
2020-06-25	9609829.0	4838921.0	489312.0	155 days	26
2020-06-26	9801572.0	4945557.0	494181.0	156 days	26
2020-06-27	9979535.0	5051864.0	498710.0	157 days	26
2020-06-28	10145791.0	5140899.0	501893.0	158 days	26
2020-06-29	10302151.0	5235813.0	505505.0	159 days	27

The death toll was low in 14th week, as it was expected to rise looking at the trend of infection of death trend of previous few weeks.

Number of Death cases were consistently dropping since 14th week, upto 19th week. After which it's again showing a spike for two consecutive weeks.

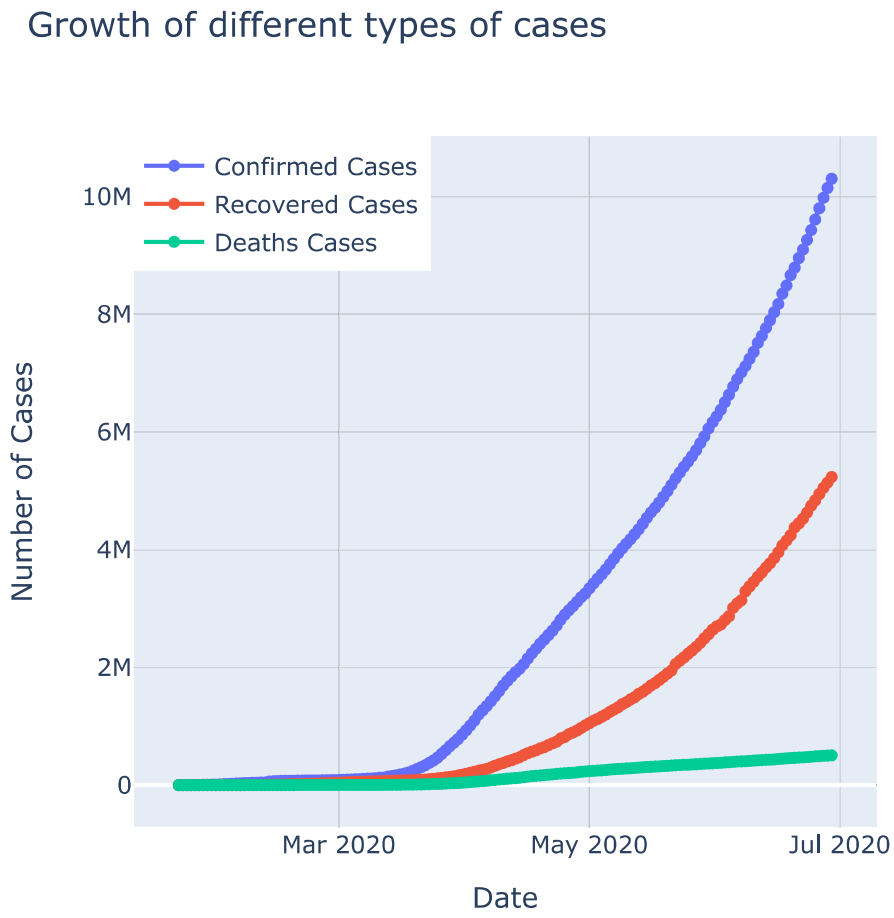
We are somehow able to reduce the Death Numbers or maybe able to control it somehow, but new infections are increasing with considerable speed recording almost 800k cases in 21st, week which is a staggering number.

The number infections are increasing every week.

Growth rate of Confirmed, Recovered and Death Cases

In [50]:

```
fig = go.Figure()
fig.add_trace(go.Scatter(x=datewise.index, y=datewise["Confirmed"],
                         mode = 'lines+markers',
                         name='Confirmed Cases'))
fig.add_trace(go.Scatter(x=datewise.index, y=datewise["Recovered"],
                         mode='lines+markers',
                         name="Recovered Cases"))
fig.add_trace(go.Scatter(x=datewise.index, y=datewise["Deaths"],
                         mode='lines+markers',
                         name='Deaths Cases'))
fig.update_layout(title='Growth of different types of cases',
                  xaxis_title='Date', yaxis_title='Number of Cases',
                  legend=dict(x=0, y=1))
fig.show()
```



Moratality and Recovery Rate analysis around the World

In [72]:

```
# Calculating the Mortality Rate and Recovery Rate
datewise["Mortality Rate"] = (datewise["Deaths"]/datewise["Confirmed"])*100
datewise["Recovery Rate"]= (datewise["Recovered"]/datewise["Confirmed"])*100
datewise["Active Cases"]= datewise["Confirmed"]-datewise["Recovered"]-datewise["Deaths"]
datewise["Closed Cases"]=datewise["Recovered"]+datewise["Deaths"]

datewise.head()
```

Out[72]:

ObservationDate	Confirmed	Recovered	Deaths	Days Since	WeekOfYear	Mortality Rate	Recovery Rate	A C
2020-01-22	555.0	28.0	17.0	0 days	4	3.063063	5.045045	5
2020-01-23	653.0	30.0	18.0	1 days	4	2.756508	4.594181	6
2020-01-24	941.0	36.0	26.0	2 days	4	2.763018	3.825717	8
2020-01-25	1438.0	39.0	42.0	3 days	4	2.920723	2.712100	13
2020-01-26	2118.0	52.0	56.0	4 days	4	2.644004	2.455146	20

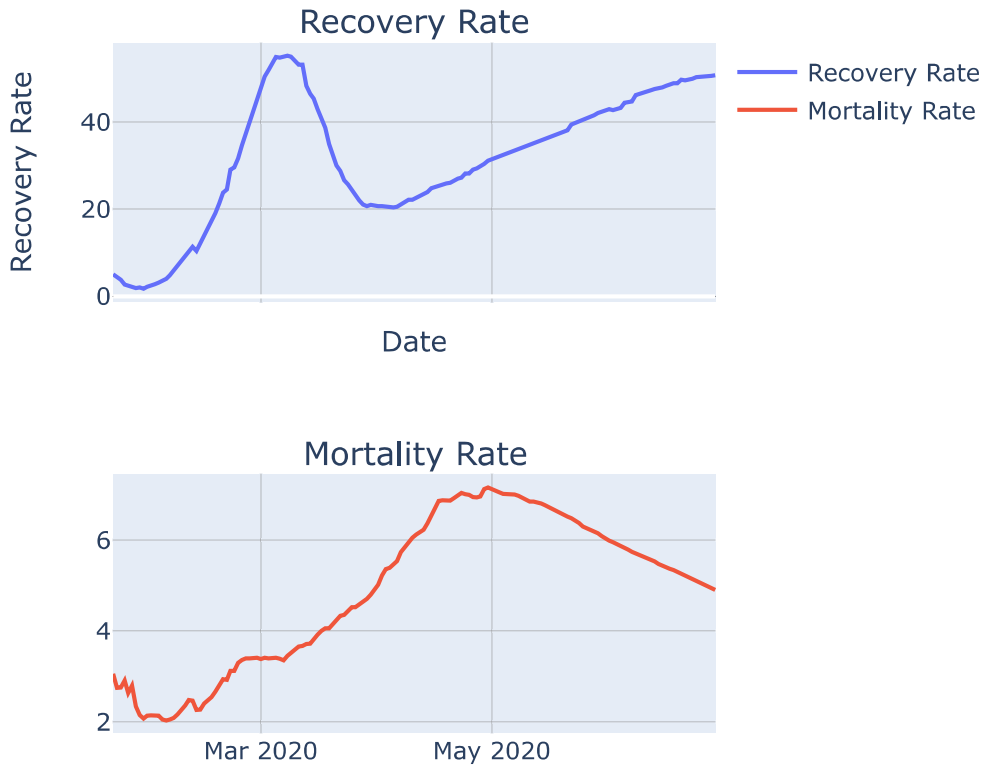
In [73]:

```
print("Average Mortality Rate", datewise['Mortality Rate'].mean())
print('Median Mortality Rate', datewise['Mortality Rate'].median())
print('Average Recovery Rate', datewise['Recovery Rate'].mean())
print('Median Recovery Rate', datewise['Recovery Rate'].median())
```

```
Average Mortality Rate 4.885801503323821
Median Mortality Rate 5.229317799345466
Average Recovery Rate 31.588583249886586
Median Recovery Rate 32.59076981198068
```

In [74]:

```
# Plotting Mortality and Recovery Rate

from plotly.subplots import make_subplots

fig = make_subplots(rows=2, cols=1,
                    shared_xaxes=True, shared_yaxes=True,
                    subplot_titles=("Recovery Rate", "Mortality Rate"))
fig.add_trace(go.Scatter(x=datewise.index, y=(datewise["Recovered"]/datewise["Confirmed"])*100,
                         name="Recovery Rate"),
              row=1, col=1)
fig.add_trace(go.Scatter(x=datewise.index, y=(datewise["Deaths"]/datewise["Confirmed"])*100,
                         name="Mortality Rate"),
              row=2, col=1)


fig.update_xaxes(title_text="Date", row=1, col=1)
fig.update_yaxes(title_text="Recovery Rate", row=1, col=1)
fig.update_xaxes(title_text="Date", row=1, col=2)
fig.update_yaxes(title_text="Mortality Rate", row=1, col=2)
fig.show()
```



Recovery rate = (Number of Recovered Cases / Number of Confirmed Cases) * 100

Recovery rate has stared to pick up again which is a good sign, another supportive reason to why number of Closed Cases are increasing.

Mortality rate = (Number of Death Cases / Number of Confirmed Cases) * 100

Mortality rate is showing a considerable for a pretty long time, which is positive sign.

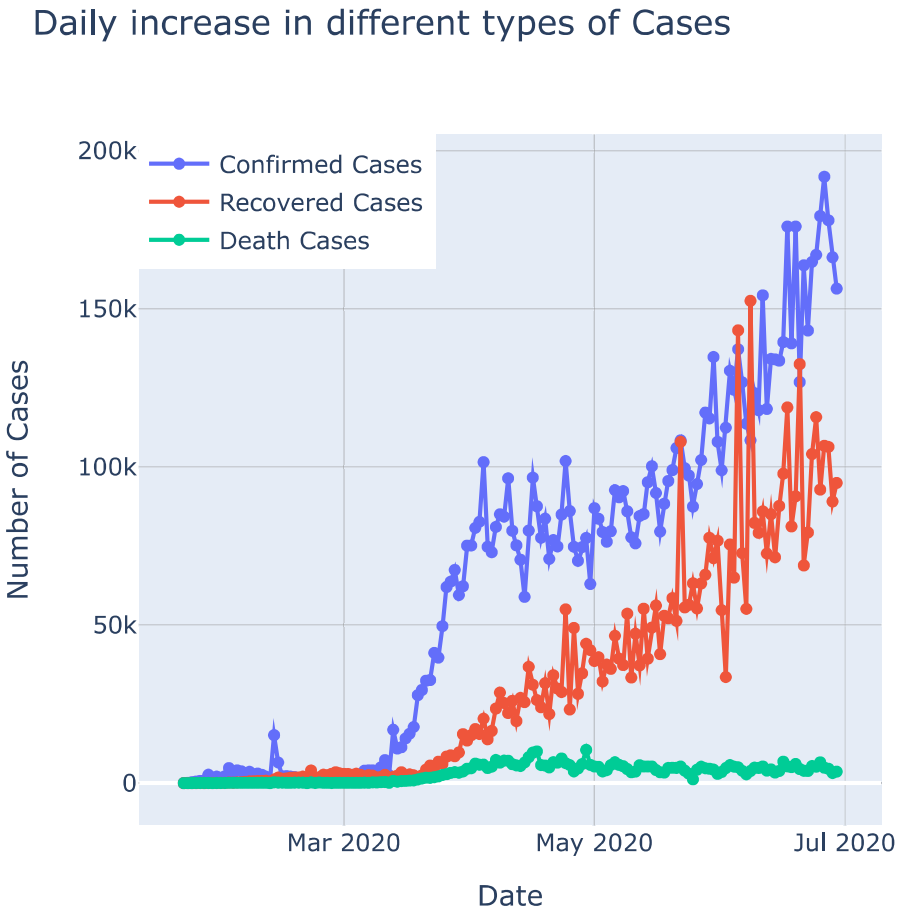
In [75]:

```
print('Average increase in number of Confirmed Cases every day: ', np.round(datewise["Confirmed"
].diff().fillna(0).mean()))
print('Average increase in number of Recovered Cases every day: ', np.round(datewise["Recovered"
].diff().fillna(0).mean()))
print('Average increase in number of Deaths Cases every day: ', np.round(datewise["Deaths"].diff
().fillna(0).mean()))
```

```
Average increase in number of Confirmed Cases every day:  64385.0
Average increase in number of Recovered Cases every day:  32724.0
Average increase in number of Deaths Cases every day:  3159.0
```

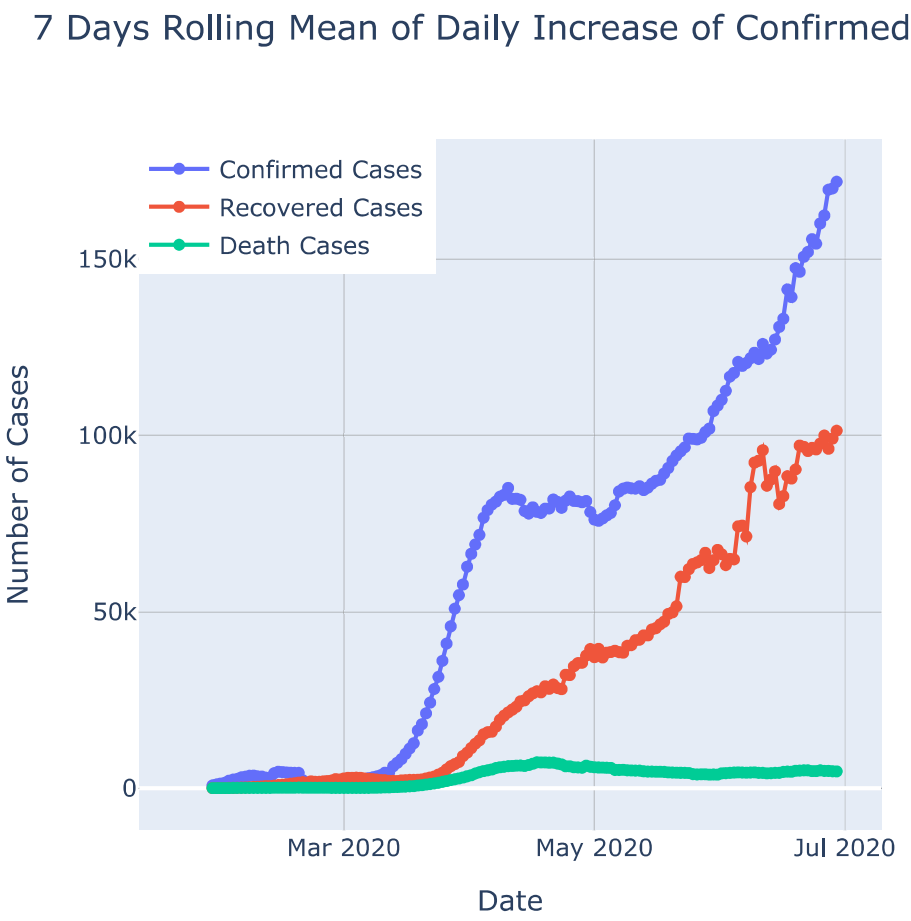
In [76]:

```
fig = go.Figure()
fig.add_trace(go.Scatter(x=datewise.index, y=datewise["Confirmed"].diff().fillna(0),
                         mode='lines+markers', name="Confirmed Cases"))
fig.add_trace(go.Scatter(x=datewise.index, y=datewise["Recovered"].diff().fillna(0),
                         mode='lines+markers', name='Recovered Cases'))
fig.add_trace(go.Scatter(x=datewise.index, y=datewise["Deaths"].diff().fillna(0),
                         mode='lines+markers', name='Death Cases'))
fig.update_layout(title='Daily increase in different types of Cases',
                  xaxis_title="Date", yaxis_title="Number of Cases",
                   legend=dict(x=0, y=1, traceorder="normal"))
fig.show()
```



In [77]:

```
fig = go.Figure()

fig.add_trace(go.Scatter(x=datewise.index, y=datewise["Confirmed"].diff().rolling(7).mean(),
                         mode='lines+markers', name="Confirmed Cases"))
fig.add_trace(go.Scatter(x=datewise.index, y=datewise["Recovered"].diff().rolling(7).mean(),
                         mode='lines+markers', name="Recovered Cases"))
fig.add_trace(go.Scatter(x=datewise.index, y=datewise["Deaths"].diff().rolling(7).mean(),
                         mode='lines+markers', name="Death Cases"))
fig.update_layout(title='7 Days Rolling Mean of Daily Increase of Confirmed, Recovered and Death
Cases',
                  xaxis_title="Date", yaxis_title="Number of Cases", legend=dict(x=0, y=1))
fig.show()
```



In [78]:

```
# Roling concept example
print(datewise["Confirmed"].diff().rolling(window=3).mean().head())
print(datewise["Confirmed"].diff().head())
print((98+288+497)/3)
```

```
ObservationDate
2020-01-22    NaN
2020-01-23    NaN
2020-01-24    NaN
2020-01-25     294.333333
2020-01-26     488.333333
Name: Confirmed, dtype: float64
ObservationDate
2020-01-22    NaN
2020-01-23     98.0
2020-01-24     288.0
2020-01-25     497.0
2020-01-26     680.0
Name: Confirmed, dtype: float64
294.3333333333333
```

Growth Factor

Growth factor is the factor by which a quantity multiplies itself over time. The formula used is:

Every day's new (Confirmed, Recovered, Deaths) / new (,,) on the previous day

A growth factor above 1 indicates an increase corresponding cases.

A growth factor above 1 but trending downward is a positive sign, whereas a growth factor constantly above 1 is the sign of exponential growth.

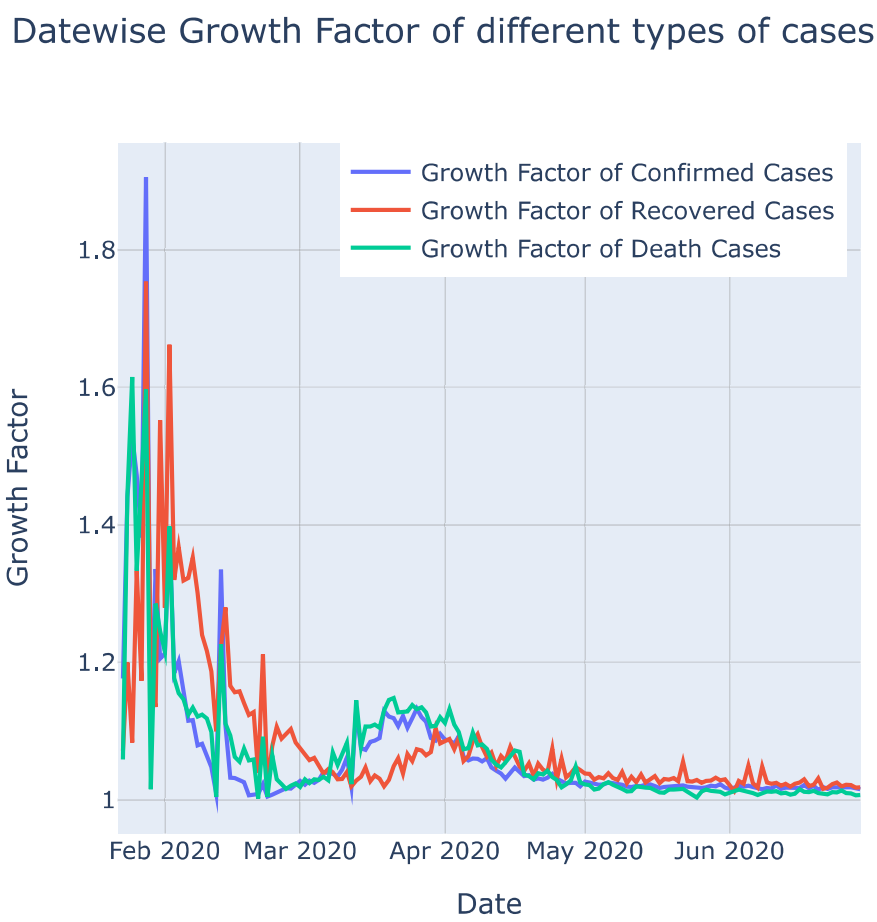
In [79]:

```
print('Average growth factor of number of Confirmed Cases: ',
     (datewise["Confirmed"]/datewise["Confirmed"].shift()).mean())
print('Median growth factor of number of Confirmed Cases: ',
     (datewise["Confirmed"]/datewise["Confirmed"].shift()).median())
print('Average growth factor of number of Recovered Cases: ',
     (datewise["Recovered"]/datewise["Recovered"].shift()).mean())
print('Median growth factor of number of Recovered Cases: ',
     (datewise["Recovered"]/datewise["Recovered"].shift()).median())
print('Average growth factor of number of Death Cases: ',
     (datewise["Deaths"]/datewise["Deaths"].shift()).mean())
print('Median growth factor of number of Death Cases: ',
     (datewise["Deaths"]/datewise["Deaths"].shift()).median())
```

```
Average growth factor of number of Confirmed Cases:  1.068274863415844
Median growth factor of number of Confirmed Cases:  1.0257975673393525
Average growth factor of number of Recovered Cases:  1.0839823834206643
Median growth factor of number of Recovered Cases:  1.043212890875988
Average growth factor of number of Death Cases:  1.0707693256475572
Median growth factor of number of Death Cases:  1.029746835443038
```

In [80]:

```
fig = go.Figure()
fig.add_trace(go.Scatter(x=datewise.index, y=datewise["Confirmed"]/datewise["Confirmed"].shift
(),
                         mode='lines', name='Growth Factor of Confirmed Cases'))
fig.add_trace(go.Scatter(x=datewise.index, y=datewise["Recovered"]/datewise["Recovered"].shift
(),
                         mode="lines", name='Growth Factor of Recovered Cases'))
fig.add_trace(go.Scatter(x=datewise.index, y=datewise["Deaths"]/datewise["Deaths"].shift(),
                         mode='lines', name='Growth Factor of Death Cases'))
fig.update_layout(title="Datewise Growth Factor of different types of cases",
                  xaxis_title='Date', yaxis_title='Growth Factor',
                  legend=dict(x=0.3, y=1, traceorder='normal'))
fig.show()
```



Growth Factor for Active and Close Cases

Growth factor is the factor by which a quentity multiplies itself over time. The formula used is :

Formula: Every day's new (Active and Closed Cases) / new (and) on the previous day.

A growth factor above 1 indicates an increase corresponding cases.

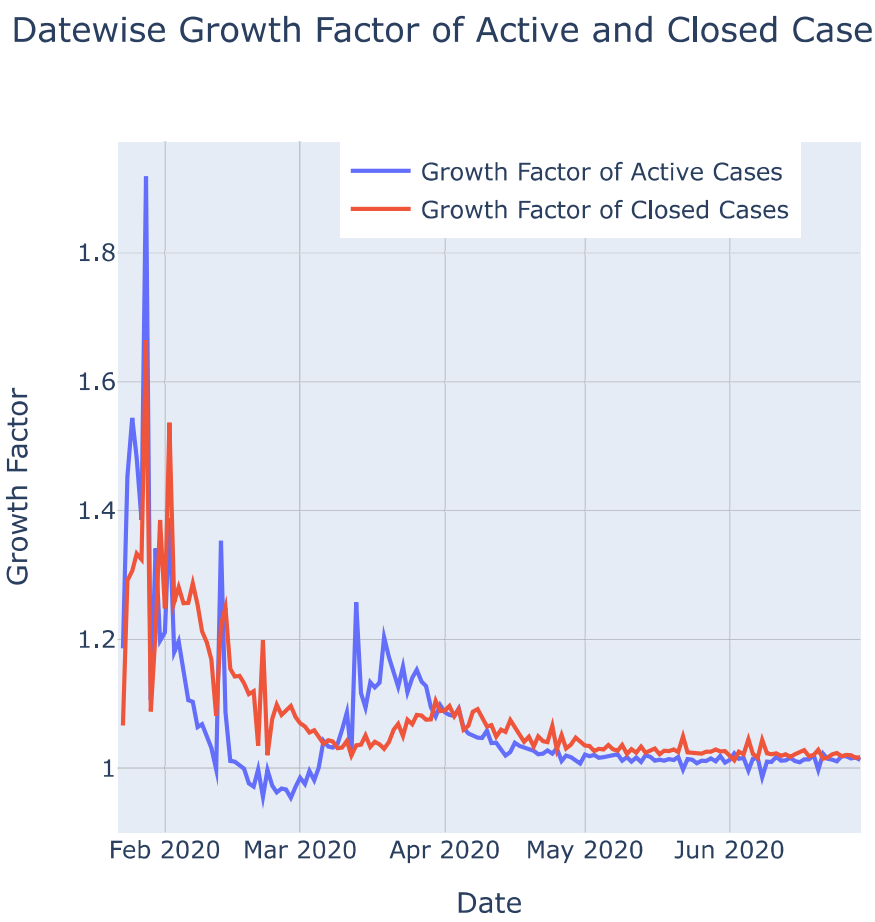
A growth factor above 1 but trending downward is a positive sign.

A growth factor constant at 1 indicates there is no change in any kind of cases.

A growth factor below 1 indicates real positive sign implying more patients are getting recovered or dying as compared to the Confirmed Cases.

In [81]:

```
fig = go.Figure()
fig.add_trace(go.Scatter(x=datewise.index,
            y=(datewise["Confirmed"]-datewise["Recovered"]-datewise["Deaths"])/
                        (datewise["Confirmed"]-datewise["Recovered"]-datewise["Deaths"]).shift
(),
                        mode='lines', name='Growth Factor of Active Cases'))
fig.add_trace(go.Scatter(x=datewise.index,
            y=(datewise["Recovered"]+datewise["Deaths"])/
                        (datewise["Recovered"]+datewise["Deaths"]).shift(),
                        mode='lines', name='Growth Factor of Closed Cases'))
fig.update_layout(title="Datewise Growth Factor of Active and Closed Cases",
                  xaxis_title="Date", yaxis_title="Growth Factor",
                  legend=dict(x=0.3, y=1))
fig.show()
```



Growth Factor constantly above 1 is an clear indication of Exponential increase in all form of cases.

Rate of Doubling for Confirmed Cases around the world

In [82]:

```
c=560
double_days=[]
cc=[]
while(1) :
    double_days.append(datewise[datewise["Confirmed"]<=c].iloc[[-1]]["Days Since"][0])
    cc.append(c)
    c=c*2
    if(c<datewise["Confirmed"].max()):
        continue
    else:
            break
```

In [83]:

```
print(cc)
```

```
[560, 1120, 2240, 4480, 8960, 17920, 35840, 71680, 143360, 286720, 573440, 114688
0, 2293760, 4587520, 9175040]
```

In [84]:

```
doubling_rate = pd.DataFrame(list(zip(cc, double_days)),
                             columns=['No. of cases', 'Days since first Case'])
doubling_rate['Number of days for doubling']=doubling_rate['Days since first Case'].diff().filln
a(doubling_rate['Days since first Case'])
doubling_rate
```

Out[84]:

	No. of cases	Days since first Case	Number of days for doubling
0	560	0 days	0 days
1	1120	2 days	2 days
2	2240	4 days	2 days
3	4480	5 days	1 days
4	8960	8 days	3 days
5	17920	11 days	3 days
6	35840	16 days	5 days
7	71680	25 days	9 days
8	143360	50 days	25 days
9	286720	58 days	8 days
10	573440	64 days	6 days
11	1146880	72 days	8 days
12	2293760	86 days	14 days
13	4587520	114 days	28 days
14	9175040	152 days	38 days

Doubling Rate is fluctuating very much, which ideally supposed to increase if we are successfully faltening the curve.

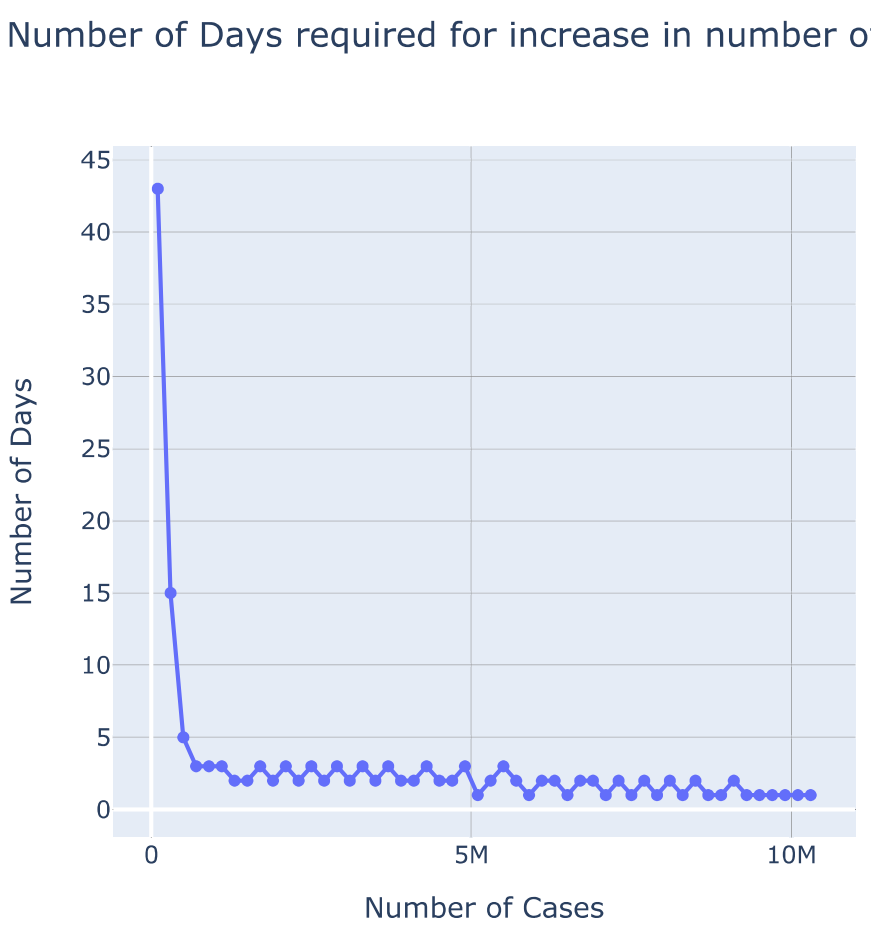
Number of days required for increase in Confirmed Cases by 200k.

In [85]:

```
c1 = 100000
days_200k = []
cc1 = []
while(1):
    days_200k.append(datewise[datewise["Confirmed"]<=c1].iloc[[-1]]["Days Since"][0])
    cc1.append(c1)
    c1 = c1+200000
    if(c1<datewise["Confirmed"].max()):
        continue
    else:
        break
```

In [86]:

```
rate_200k = pd.DataFrame(list(zip(cc1, days_200k)),
                         columns=["No. of Cases", "Days Since first Case"])
rate_200k["Days required for rise of 200k"]=rate_200k["Days Since first Case"].diff().fillna(rat
e_200k['Days Since first Case'].iloc[[0]][0])

fig=go.Figure()
fig.add_trace(go.Scatter(x=rate_200k["No. of Cases"],
                         y=rate_200k['Days required for rise of 200k'].dt.days,
                         mode='lines+markers',
                         name='Weekly Growth of Confirmed Cases'))
fig.update_layout(title='Number of Days required for increase in number of cases by 200k',
                  xaxis_title="Number of Cases", yaxis_title="Number of Days")
fig.show()
```



It's hardly taking a day or two for rise in cases by 200k, which is pretty much a clear indication that we are still not able to "Flatten the curve"

In [ ]:

In [ ]:

Export File

In [87]:

```
datewise.head()
```

Out[87]:

ObservationDate	Confirmed	Recovered	Deaths	Days Since	WeekOfYear	Mortality Rate	Recovery Rate	A C
2020-01-22	555.0	28.0	17.0	0 days	4	3.063063	5.045045	5
2020-01-23	653.0	30.0	18.0	1 days	4	2.756508	4.594181	6
2020-01-24	941.0	36.0	26.0	2 days	4	2.763018	3.825717	8
2020-01-25	1438.0	39.0	42.0	3 days	4	2.920723	2.712100	13
2020-01-26	2118.0	52.0	56.0	4 days	4	2.644004	2.455146	20

In [88]:

```
datewise.to_csv('./datewise.csv', sep=',', na_rep='NaN')
```

In [89]:

```
covid.head()
```

Out[89]:

	Province/State	Country/Region	Last Update	Confirmed	Deaths	Recovered	ObservationDate
0	Anhui	Mainland China	1/22/2020 17:00	1.0	0.0	0.0	2020-01-22
1	Beijing	Mainland China	1/22/2020 17:00	14.0	0.0	0.0	2020-01-22
2	Chongqing	Mainland China	1/22/2020 17:00	6.0	0.0	0.0	2020-01-22
3	Fujian	Mainland China	1/22/2020 17:00	1.0	0.0	0.0	2020-01-22
4	Gansu	Mainland China	1/22/2020 17:00	0.0	0.0	0.0	2020-01-22

In [90]:

```
covid.to_csv('./covid.csv', sep=',', na_rep='NaN')
```

In [91]:

```
grouped_country.to_csv('./grouped_country.csv', sep=',', na_rep='NaN')
```

In [ ]:

```

```