Importing the libraries

In [146…

```
import pandas as pd
import numpy as np
from scipy import stats
import plotly.express as px
import plotly.offline as pyo
pyo.init_notebook_mode()
```

Reading the Dataset

From the feeback given for Stage 1 of project, (cases/deaths against dates), this is the updated data

In [146…

```
superdata = pd.read_csv("covid19_superdata.csv")
superdata
```

Out[1467]:

	countyFIPS	County Name	State	StateFIPS	date	cases	deaths	population
0	1001	autauga county	AL	1	2020-01-22	0	0	55869
1	1001	autauga county	AL	1	2020-01-23	0	0	55869
2	1001	autauga county	AL	1	2020-01-24	0	0	55869
3	1001	autauga county	AL	1	2020-01-25	0	0	55869
4	1001	autauga county	AL	1	2020-01-26	0	0	55869
...	...	...	...	...	...	...	...	...
3487424	56045	weston county	WY	56	2023-02-01	1885	22	6927
3487425	56045	weston county	WY	56	2023-02-02	1880	22	6927
3487426	56045	weston county	WY	56	2023-02-03	1880	22	6927
3487427	56045	weston county	WY	56	2023-02-04	1880	22	6927
3487428	56045	weston county	WY	56	2023-02-05	1880	22	6927

3487429 rows × 8 columns

Generate weekly statistics (mean, median, mode) for number of new cases and deaths across a specific state.

In [146…

```
#Filtering data based on a specific state
IL_data = superdata[superdata['State']=='IL']
IL_data
```

Out[1468]:

	countyFIPS	County Name	State	StateFIPS	date	cases	deaths	population
658823	17001	adams county	IL	17	2020-01-22	0	0	65435
658824	17001	adams county	IL	17	2020-01-23	0	0	65435
658825	17001	adams county	IL	17	2020-01-24	0	0	65435
658826	17001	adams county	IL	17	2020-01-25	0	0	65435
658827	17001	adams county	IL	17	2020-01-26	0	0	65435
...	...	...	...	...	...	...	...	...
772140	17203	woodford county	IL	17	2023-02-01	12249	118	38459
772141	17203	woodford county	IL	17	2023-02-02	12249	118	38459
772142	17203	woodford county	IL	17	2023-02-03	12249	118	38459
772143	17203	woodford county	IL	17	2023-02-04	12249	118	38459
772144	17203	woodford county	IL	17	2023-02-05	12249	118	38459

113322 rows × 8 columns

In [146…

```
#Grouping the data based on date and state
IL_data=IL_data[['date', 'State', 'cases','deaths', 'population']].groupby(['date','St
IL_data
```

Out[1469]:

	date	State	cases	deaths	population
0	2020-01-22	IL	0	0	12671821
1	2020-01-23	IL	0	0	12671821
2	2020-01-24	IL	1	0	12671821
3	2020-01-25	IL	1	0	12671821
4	2020-01-26	IL	1	0	12671821
...	...	...	...	...	...
1106	2023-02-01	IL	3125454	27998	12671821
1107	2023-02-02	IL	3125454	27998	12671821
1108	2023-02-03	IL	3125454	27998	12671821
1109	2023-02-04	IL	3125454	27998	12671821
1110	2023-02-05	IL	3125454	27998	12671821

1111 rows × 5 columns

In [147…

```
#converting the date column to datetime format, easier to use this way.
IL_data['date'] = pd.to_datetime(IL_data['date'])

#Aggregating and grouping the data by weekly intervals.
IL_weekly_data = IL_data.groupby(['State', pd.Grouper(key="date", freq="W-Wed")]).agg(
    "cases": "sum",
    "deaths": "sum",
```

```
    "population": "first"
}).reset_index()
IL_weekly_data
```

Out[1470]:

	State	date	cases	deaths	population
0	IL	2020-01-22	0	0	12671821
1	IL	2020-01-29	6	0	12671821
2	IL	2020-02-05	13	0	12671821
3	IL	2020-02-12	14	0	12671821
4	IL	2020-02-19	14	0	12671821
...	...	...	...	...	...
155	IL	2023-01-11	21860791	195933	12671821
156	IL	2023-01-18	21869246	195958	12671821
157	IL	2023-01-25	21878178	195986	12671821
158	IL	2023-02-01	21878178	195986	12671821
159	IL	2023-02-08	12501816	111992	12671821

160 rows × 5 columns

In [147…

```
#Adding new columns to see the no. of new cases and deaths each week.
IL_weekly_data = IL_weekly_data.assign(new_cases = IL_weekly_data.cases.diff())
IL_weekly_data = IL_weekly_data.assign(new_deaths = IL_weekly_data.deaths.diff())

#Handling Nan values converting values to int
IL_weekly_data["new_cases"] = IL_weekly_data["new_cases"].fillna(0).astype(int)
IL_weekly_data["new_deaths"] = IL_weekly_data["new_deaths"].fillna(0).astype(int)

#Handling negative values
IL_weekly_data.loc[IL_weekly_data['new_cases'] < 0, 'new_cases'] = 0
IL_weekly_data.loc[IL_weekly_data['new_deaths'] < 0, 'new_deaths'] = 0
IL_weekly_data
```

Out[1471]:

	State	date	cases	deaths	population	new_cases	new_deaths
0	IL	2020-01-22	0	0	12671821	0	0
1	IL	2020-01-29	6	0	12671821	6	0
2	IL	2020-02-05	13	0	12671821	7	0
3	IL	2020-02-12	14	0	12671821	1	0
4	IL	2020-02-19	14	0	12671821	0	0
...	...	...	...	...	...	...	...
155	IL	2023-01-11	21860791	195933	12671821	5387	33
156	IL	2023-01-18	21869246	195958	12671821	8455	25
157	IL	2023-01-25	21878178	195986	12671821	8932	28
158	IL	2023-02-01	21878178	195986	12671821	0	0
159	IL	2023-02-08	12501816	111992	12671821	0	0

160 rows × 7 columns

In [147…

```
#Weekly statistics
mean_cases = IL_weekly_data["new_cases"].mean()
median_cases = IL_weekly_data["new_cases"].median()
mode_cases = stats.mode(IL_weekly_data["new_cases"], keepdims=True).mode[0]

mean_deaths = IL_weekly_data["new_deaths"].mean()
median_deaths = IL_weekly_data["new_deaths"].median()
mode_deaths = stats.mode(IL_weekly_data["new_deaths"], keepdims=True).mode[0]

print("Mean of new cases", mean_cases)
print("Median of new cases", median_cases)
print("Mode of new cases", mode_cases)
print("Mean of new deaths", mean_deaths)
print("Median of new deaths", median_deaths)
print("Median of new deaths", mode_deaths)
```

```
Mean of new cases 151863.30625
Median of new cases 87391.5
Mode of new cases 0
Mean of new deaths 1632.81875
Median of new deaths 991.5
Median of new deaths 0
```

Compare the data against 3 other states. Normalize by population, use a normalization factor which is able to identify cases and deaths, for example try per 10,000 or 100,000 (this depends on the population). Plot the values across the weeks in a line plot for the 3 states in a single graph. Describe why the rates differ across these states in the notebook. Identify the peaks, are they consistent with the US pattern?

In [147…

```
OH_data = superdata[superdata['State']=='OH']
```

```
OH_data
```

Out[1473]:

	countyFIPS	County Name	State	StateFIPS	date	cases	deaths	population
2266440	39001	adams county	OH	39	2020-01-22	0	0	27698
2266441	39001	adams county	OH	39	2020-01-23	0	0	27698
2266442	39001	adams county	OH	39	2020-01-24	0	0	27698
2266443	39001	adams county	OH	39	2020-01-25	0	0	27698
2266444	39001	adams county	OH	39	2020-01-26	0	0	27698
...	...	...	...	...	...	...	...	...
2364203	39175	wyandot county	OH	39	2023-02-01	6612	111	21772
2364204	39175	wyandot county	OH	39	2023-02-02	6614	111	21772
2364205	39175	wyandot county	OH	39	2023-02-03	6614	111	21772
2364206	39175	wyandot county	OH	39	2023-02-04	6614	111	21772
2364207	39175	wyandot county	OH	39	2023-02-05	6614	111	21772

97768 rows × 8 columns

In [147…

```
CA_data = superdata[superdata['State']=='CA']
CA_data
```

Out[1474]:

	countyFIPS	County Name	State	StateFIPS	date	cases	deaths	population
205535	6001	alameda county	CA	6	2020-01-22	4	0	1671329
205536	6001	alameda county	CA	6	2020-01-23	4	0	1671329
205537	6001	alameda county	CA	6	2020-01-24	4	0	1671329
205538	6001	alameda county	CA	6	2020-01-25	5	0	1671329
205539	6001	alameda county	CA	6	2020-01-26	5	0	1671329
...	...	...	...	...	...	...	...	...
269968	6115	yuba county	CA	6	2023-02-01	19903	122	78668
269969	6115	yuba county	CA	6	2023-02-02	19903	122	78668
269970	6115	yuba county	CA	6	2023-02-03	19903	122	78668
269971	6115	yuba county	CA	6	2023-02-04	19903	122	78668
269972	6115	yuba county	CA	6	2023-02-05	19903	122	78668

64438 rows × 8 columns

In [147…

```
CO_data = superdata[superdata['State']=='CO']
CO_data
```

Out[1475]:

	countyFIPS	County Name	State	StateFIPS	date	cases	deaths	population
269973	8001	adams county	CO	8	2020-01-22	0	0	517421
269974	8001	adams county	CO	8	2020-01-23	0	0	517421
269975	8001	adams county	CO	8	2020-01-24	0	0	517421
269976	8001	adams county	CO	8	2020-01-25	0	0	517421
269977	8001	adams county	CO	8	2020-01-26	0	0	517421
...	...	...	...	...	...	...	...	...
341072	8125	yuma county	CO	8	2023-02-01	2315	27	10019
341073	8125	yuma county	CO	8	2023-02-02	2315	27	10019
341074	8125	yuma county	CO	8	2023-02-03	2315	27	10019
341075	8125	yuma county	CO	8	2023-02-04	2315	27	10019
341076	8125	yuma county	CO	8	2023-02-05	2315	27	10019

71104 rows × 8 columns

In [147...

```
OH_data=OH_data[['date', 'State', 'cases', 'deaths', 'population']].groupby(['date','S
CA_data=CA_data[['date', 'State', 'cases', 'deaths', 'population']].groupby(['date','S
CO_data=CO_data[['date', 'State', 'cases', 'deaths', 'population']].groupby(['date','S
```

In [147...

```
OH_data['date'] = pd.to_datetime(OH_data['date'])
OH_weekly_data = OH_data.groupby(['State', pd.Grouper(key="date", freq="W-Wed")]).agg(
    "cases": "sum",
    "deaths": "sum",
    "population": "first"
}).reset_index()

CA_data['date'] = pd.to_datetime(CA_data['date'])
CA_weekly_data = CA_data.groupby(['State', pd.Grouper(key="date", freq="W-Wed")]).agg(
    "cases": "sum",
    "deaths": "sum",
    "population": "first"
}).reset_index()

CO_data['date'] = pd.to_datetime(CO_data['date'])
CO_weekly_data = CO_data.groupby(['State', pd.Grouper(key="date", freq="W-Wed")]).agg(
    "cases": "sum",
    "deaths": "sum",
    "population": "first"
}).reset_index()
```

In [147...

```
OH_weekly_data = OH_weekly_data.assign(new_cases = OH_weekly_data.cases.diff())
OH_weekly_data = OH_weekly_data.assign(new_deaths = OH_weekly_data.deaths.diff())
OH_weekly_data["new_cases"] = OH_weekly_data["new_cases"].fillna(0).astype(int)
OH_weekly_data["new_deaths"] = OH_weekly_data["new_deaths"].fillna(0).astype(int)
OH_weekly_data.loc[OH_weekly_data['new_cases'] < 0, 'new_cases'] = 0
OH_weekly_data.loc[OH_weekly_data['new_deaths'] < 0, 'new_deaths'] = 0

CA_weekly_data = CA_weekly_data.assign(new_cases = CA_weekly_data.cases.diff())
CA_weekly_data = CA_weekly_data.assign(new_deaths = CA_weekly_data.deaths.diff())
```

```
CA_weekly_data["new_cases"] = CA_weekly_data["new_cases"].fillna(0).astype(int)
CA_weekly_data["new_deaths"] = CA_weekly_data["new_deaths"].fillna(0).astype(int)
CA_weekly_data.loc[CA_weekly_data['new_cases'] < 0, 'new_cases'] = 0
CA_weekly_data.loc[CA_weekly_data['new_deaths'] < 0, 'new_deaths'] = 0

CO_weekly_data = CO_weekly_data.assign(new_cases = CO_weekly_data.cases.diff())
CO_weekly_data = CO_weekly_data.assign(new_deaths = CO_weekly_data.deaths.diff())
CO_weekly_data["new_cases"] = CO_weekly_data["new_cases"].fillna(0).astype(int)
CO_weekly_data["new_deaths"] = CO_weekly_data["new_deaths"].fillna(0).astype(int)
CO_weekly_data.loc[CO_weekly_data['new_cases'] < 0, 'new_cases'] = 0
CO_weekly_data.loc[CO_weekly_data['new_deaths'] < 0, 'new_deaths'] = 0
```

In [147…

```
OH_weekly_data
```

Out[1479]:

	State	date	cases	deaths	population	new_cases	new_deaths
0	OH	2020-01-22	0	0	11689100	0	0
1	OH	2020-01-29	0	0	11689100	0	0
2	OH	2020-02-05	0	0	11689100	0	0
3	OH	2020-02-12	0	0	11689100	0	0
4	OH	2020-02-19	0	0	11689100	0	0
…	…	…	…	…	…	…	…
155	OH	2023-01-11	23250241	287196	11689100	97265	735
156	OH	2023-01-18	23321557	287931	11689100	71316	735
157	OH	2023-01-25	23377284	288701	11689100	55727	770
158	OH	2023-02-01	23434369	289443	11689100	57085	742
159	OH	2023-02-08	13424108	165832	11689100	0	0

160 rows × 7 columns

In [148…

```
CA_weekly_data
```

Out[1480]:

	State	date	cases	deaths	population	new_cases	new_deaths
0	CA	2020-01-22	722	0	39512223	0	0
1	CA	2020-01-29	5291	0	39512223	4569	0
2	CA	2020-02-05	2544	0	39512223	0	0
3	CA	2020-02-12	573	6	39512223	0	6
4	CA	2020-02-19	968	9	39512223	395	3
...	...	...	...	...	...	...	...
155	CA	2023-01-11	76645756	687113	39512223	370624	3623
156	CA	2023-01-18	76954237	691670	39512223	308481	4557
157	CA	2023-01-25	77108590	694000	39512223	154353	2330
158	CA	2023-02-01	77214261	694406	39512223	105671	406
159	CA	2023-02-08	44133840	396824	39512223	0	0

160 rows × 7 columns

In [148…

```
CO_weekly_data
```

Out[1481]:

	State	date	cases	deaths	population	new_cases	new_deaths
0	CO	2020-01-22	0	0	5758736	0	0
1	CO	2020-01-29	0	0	5758736	0	0
2	CO	2020-02-05	0	0	5758736	0	0
3	CO	2020-02-12	0	0	5758736	0	0
4	CO	2020-02-19	0	0	5758736	0	0
...	...	...	...	...	...	...	...
155	CO	2023-01-11	12048323	96675	5758736	33740	243
156	CO	2023-01-18	12070959	96958	5758736	22636	283
157	CO	2023-01-25	12090515	97250	5758736	19556	292
158	CO	2023-02-01	12106525	97469	5758736	16010	219
159	CO	2023-02-08	6928060	55748	5758736	0	0

160 rows × 7 columns

In [148…

```
#Normalizing the values per 100k population
IL_weekly_data["new_cases_norm_100K"] = IL_weekly_data["new_cases"] / IL_weekly_data['
IL_weekly_data["new_deaths_norm_100K"] = IL_weekly_data["new_deaths"] / IL_weekly_data
IL_weekly_data.loc[IL_weekly_data['new_cases_norm_100K'] < 0, 'new_cases_norm_100K'] =
IL_weekly_data.loc[IL_weekly_data['new_deaths_norm_100K'] < 0, 'new_deaths_norm_100K']

OH_weekly_data["new_cases_norm_100K"] = OH_weekly_data["new_cases"] / OH_weekly_data['
OH_weekly_data["new_deaths_norm_100K"] = OH_weekly_data["new_deaths"] / OH_weekly_data
OH_weekly_data.loc[OH_weekly_data['new_cases_norm_100K'] < 0, 'new_cases_norm_100K'] =
```

```
OH_weekly_data.loc[OH_weekly_data['new_deaths_norm_100K'] < 0, 'new_deaths_norm_100K']

CA_weekly_data["new_cases_norm_100K"] = CA_weekly_data["new_cases"] / CA_weekly_data['
CA_weekly_data["new_deaths_norm_100K"] = CA_weekly_data["new_deaths"] / CA_weekly_data
CA_weekly_data.loc[CA_weekly_data['new_cases_norm_100K'] < 0, 'new_cases_norm_100K'] =
CA_weekly_data.loc[CA_weekly_data['new_deaths_norm_100K'] < 0, 'new_deaths_norm_100K']

CO_weekly_data["new_cases_norm_100K"] = CO_weekly_data["new_cases"] / CO_weekly_data['
CO_weekly_data["new_deaths_norm_100K"] = CO_weekly_data["new_deaths"] / CO_weekly_data
CO_weekly_data.loc[CO_weekly_data['new_cases_norm_100K'] < 0, 'new_cases_norm_100K'] =
CO_weekly_data.loc[CO_weekly_data['new_deaths_norm_100K'] < 0, 'new_deaths_norm_100K']
```

In [148…

```
IL_weekly_data
```

Out[1483]:

	State	date	cases	deaths	population	new_cases	new_deaths	new_cases_norm_100K	ne
0	IL	2020-01-22	0	0	12671821	0	0	0.000000	
1	IL	2020-01-29	6	0	12671821	6	0	0.047349	
2	IL	2020-02-05	13	0	12671821	7	0	0.055241	
3	IL	2020-02-12	14	0	12671821	1	0	0.007892	
4	IL	2020-02-19	14	0	12671821	0	0	0.000000	
...	...	...	...	...	...	...	...	...	
155	IL	2023-01-11	21860791	195933	12671821	5387	33	42.511648	
156	IL	2023-01-18	21869246	195958	12671821	8455	25	66.722849	
157	IL	2023-01-25	21878178	195986	12671821	8932	28	70.487107	
158	IL	2023-02-01	21878178	195986	12671821	0	0	0.000000	
159	IL	2023-02-08	12501816	111992	12671821	0	0	0.000000	

160 rows × 9 columns

In [148…

```
CA_weekly_data
```

Out[1484]:

	State	date	cases	deaths	population	new_cases	new_deaths	new_cases_norm_100K	ne
0	CA	2020-01-22	722	0	39512223	0	0	0.000000	
1	CA	2020-01-29	5291	0	39512223	4569	0	11.563510	
2	CA	2020-02-05	2544	0	39512223	0	0	0.000000	
3	CA	2020-02-12	573	6	39512223	0	6	0.000000	
4	CA	2020-02-19	968	9	39512223	395	3	0.999691	
...	...	...	...	...	...	...	...	...	
155	CA	2023-01-11	76645756	687113	39512223	370624	3623	937.998351	
156	CA	2023-01-18	76954237	691670	39512223	308481	4557	780.722968	
157	CA	2023-01-25	77108590	694000	39512223	154353	2330	390.646206	
158	CA	2023-02-01	77214261	694406	39512223	105671	406	267.438762	
159	CA	2023-02-08	44133840	396824	39512223	0	0	0.000000	

160 rows × 9 columns

In [148…

```
OH_weekly_data
```

Out[1485]:

	State	date	cases	deaths	population	new_cases	new_deaths	new_cases_norm_100K	ne
0	OH	2020-01-22	0	0	11689100	0	0	0.000000	
1	OH	2020-01-29	0	0	11689100	0	0	0.000000	
2	OH	2020-02-05	0	0	11689100	0	0	0.000000	
3	OH	2020-02-12	0	0	11689100	0	0	0.000000	
4	OH	2020-02-19	0	0	11689100	0	0	0.000000	
...	...	...	...	...	...	...	...	...	
155	OH	2023-01-11	23250241	287196	11689100	97265	735	832.099991	
156	OH	2023-01-18	23321557	287931	11689100	71316	735	610.106852	
157	OH	2023-01-25	23377284	288701	11689100	55727	770	476.743291	
158	OH	2023-02-01	23434369	289443	11689100	57085	742	488.360952	
159	OH	2023-02-08	13424108	165832	11689100	0	0	0.000000	

160 rows × 9 columns

In [148…

```
CO_weekly_data
```

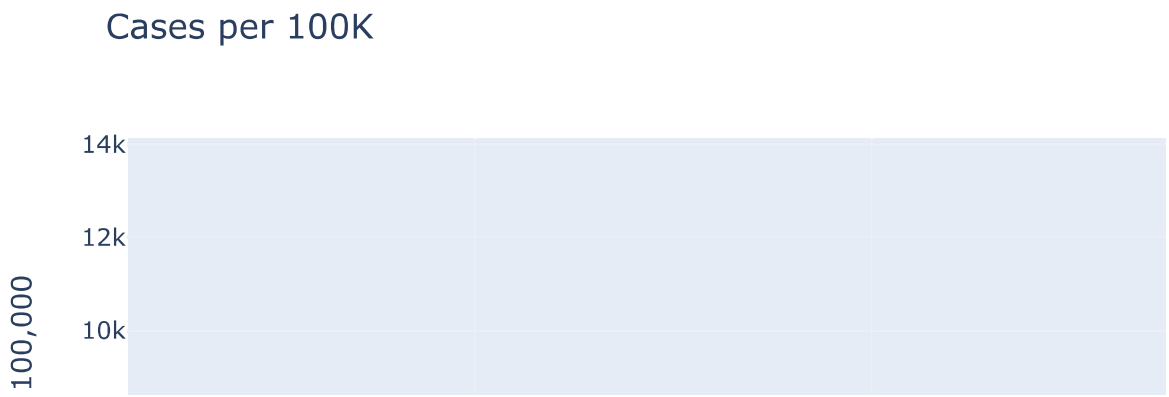
Out[1486]:

	State	date	cases	deaths	population	new_cases	new_deaths	new_cases_norm_100K	ne
0	CO	2020-01-22	0	0	5758736	0	0	0.000000	
1	CO	2020-01-29	0	0	5758736	0	0	0.000000	
2	CO	2020-02-05	0	0	5758736	0	0	0.000000	
3	CO	2020-02-12	0	0	5758736	0	0	0.000000	
4	CO	2020-02-19	0	0	5758736	0	0	0.000000	
...	...	...	...	...	...	...	...	...	
155	CO	2023-01-11	12048323	96675	5758736	33740	243	585.892460	
156	CO	2023-01-18	12070959	96958	5758736	22636	283	393.072369	
157	CO	2023-01-25	12090515	97250	5758736	19556	292	339.588410	
158	CO	2023-02-01	12106525	97469	5758736	16010	219	278.012397	
159	CO	2023-02-08	6928060	55748	5758736	0	0	0.000000	

160 rows × 9 columns

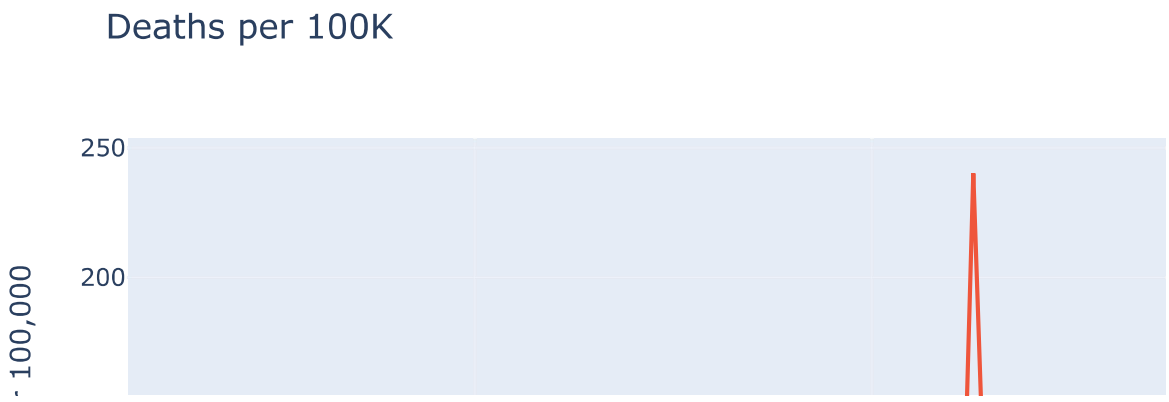
In [148…

```
weekly_data = [IL_weekly_data, OH_weekly_data, CA_weekly_data, CO_weekly_data]

# Getting all states weekly data into one dataframe
all_states_weekly_data = pd.concat(weekly_data)

# Plot the no. of new cases per 100K population
cases_weekly_norm = px.line(all_states_weekly_data, x='date', y='new_cases_norm_100K',
cases_weekly_norm.update_layout(xaxis_title='Weekly cases', yaxis_title=f'Number of ca
cases_weekly_norm.show()
```



In [148…

```
# Plotting the no. of new deaths per 100K population
deaths_weekly_norm = px.line(all_states_weekly_data, x='date', y='new_deaths_norm_100k
deaths_weekly_norm.update_layout(xaxis_title='Weekly deaths', yaxis_title=f'Number of
deaths_weekly_norm.show()
```



After identifying the peaks and comparing them with the US pattern, we observe that they are consistent at most of the plot i.e., during jan 2021 the cases and deaths are almost similar and also in jan 2022 they are similar.

Identify 3 counties within a state of your choice with high cases and death rates.

In [148…

```
NY_weekly_data = superdata[superdata['State']=='NY']
NY_weekly_data
```

Out[1489]:

	countyFIPS	County Name	State	StateFIPS	date	cases	deaths	population
2027575	36001	albany county	NY	36	2020-01-22	0	0	305506
2027576	36001	albany county	NY	36	2020-01-23	0	0	305506
2027577	36001	albany county	NY	36	2020-01-24	0	0	305506
2027578	36001	albany county	NY	36	2020-01-25	0	0	305506
2027579	36001	albany county	NY	36	2020-01-26	0	0	305506
...	...	...	...	...	...	...	...	...
2096452	36123	yates county	NY	36	2023-02-01	4427	43	24913
2096453	36123	yates county	NY	36	2023-02-02	4427	43	24913
2096454	36123	yates county	NY	36	2023-02-03	4427	43	24913
2096455	36123	yates county	NY	36	2023-02-04	4427	43	24913
2096456	36123	yates county	NY	36	2023-02-05	4427	43	24913

68882 rows × 8 columns

In [149...

```
NY_weekly_data = NY_weekly_data.assign(new_cases = NY_weekly_data.cases.diff())
NY_weekly_data = NY_weekly_data.assign(new_deaths = NY_weekly_data.deaths.diff())
```

In [149...

```
NY_weekly_data=NY_weekly_data.groupby('County Name').agg({
    'new_cases': 'sum',
    'new_deaths': 'sum',
    'population': 'first'
}).reset_index()

NY_weekly_data["new_cases"] = NY_weekly_data["new_cases"].fillna(0).astype(int)
NY_weekly_data["new_deaths"] = NY_weekly_data["new_deaths"].fillna(0).astype(int)
NY_weekly_data

NY_weekly_data.loc[NY_weekly_data['new_cases'] < 0, 'new_cases'] = 0
NY_weekly_data.loc[NY_weekly_data['new_deaths'] < 0, 'new_deaths'] = 0
NY_weekly_data
```

Out[1491]:

	County Name	new_cases	new_deaths	population
0	albany county	78356	608	305506
1	allegany county	0	0	46091
2	bronx county	497834	8297	1418207
3	broome county	0	0	190488
4	cattaraugus county	0	0	76117
...	...	...	...	...
57	washington county	0	0	61204
58	wayne county	6091	67	89918
59	westchester county	311770	2768	967506
60	wyoming county	0	0	39859
61	yates county	0	0	24913

62 rows × 4 columns

In [149…

```
NY_weekly_data[['County Name', 'new_cases', 'new_deaths', 'population']]
```

Out[1492]:

	County Name	new_cases	new_deaths	population
0	albany county	78356	608	305506
1	allegany county	0	0	46091
2	bronx county	497834	8297	1418207
3	broome county	0	0	190488
4	cattaraugus county	0	0	76117
...	...	...	...	...
57	washington county	0	0	61204
58	wayne county	6091	67	89918
59	westchester county	311770	2768	967506
60	wyoming county	0	0	39859
61	yates county	0	0	24913

62 rows × 4 columns

In [149…

```
top_3_counties = NY_weekly_data.sort_values(by=['new_cases'], ascending=False).head(3)
top_3_counties

top_3 = top_3_counties['County Name'].to_list()
print("The Top three counties with highest no. of cases and deaths are: ", top_3)
```

```
The Top three counties with highest no. of cases and deaths are:  ['kings county', 'q
ueens county', 'suffolk county']
```

This is for plotting State pattern to compare with Counties pattern

In [149…
```
NY_state_pattern_data = superdata[superdata['State']=='NY']
NY_state_pattern_data
```

Out[1494]:

	countyFIPS	County Name	State	StateFIPS	date	cases	deaths	population
2027575	36001	albany county	NY	36	2020-01-22	0	0	305506
2027576	36001	albany county	NY	36	2020-01-23	0	0	305506
2027577	36001	albany county	NY	36	2020-01-24	0	0	305506
2027578	36001	albany county	NY	36	2020-01-25	0	0	305506
2027579	36001	albany county	NY	36	2020-01-26	0	0	305506
...	...	...	...	...	...	...	...	...
2096452	36123	yates county	NY	36	2023-02-01	4427	43	24913
2096453	36123	yates county	NY	36	2023-02-02	4427	43	24913
2096454	36123	yates county	NY	36	2023-02-03	4427	43	24913
2096455	36123	yates county	NY	36	2023-02-04	4427	43	24913
2096456	36123	yates county	NY	36	2023-02-05	4427	43	24913

68882 rows × 8 columns

In [149…
```
#Grouping the data based on date and state
NY_state_pattern_data=NY_state_pattern_data[['date', 'State', 'cases','deaths', 'popul
NY_state_pattern_data
```

Out[1495]:

	date	State	cases	deaths	population
0	2020-01-22	NY	0	0	19453561
1	2020-01-23	NY	0	0	19453561
2	2020-01-24	NY	0	0	19453561
3	2020-01-25	NY	0	0	19453561
4	2020-01-26	NY	0	0	19453561
...	...	...	...	...	...
1106	2023-02-01	NY	6567845	75336	19453561
1107	2023-02-02	NY	6567845	75602	19453561
1108	2023-02-03	NY	6567845	75602	19453561
1109	2023-02-04	NY	6567845	75602	19453561
1110	2023-02-05	NY	6567845	75602	19453561

1111 rows × 5 columns

In [149…
```
#converting the date column to datetime format, easier to use this way.
NY_state_pattern_data['date'] = pd.to_datetime(NY_state_pattern_data['date'])

#Aggregating and grouping the data by weekly intervals.
NY_weekly_pattern = NY_state_pattern_data.groupby(['State', pd.Grouper(key="date", fre
    "cases": "sum",
    "deaths": "sum",
    "population": "first"
}).reset_index()
NY_weekly_pattern
```

Out[1496]:

	State	date	cases	deaths	population
0	NY	2020-01-22	0	0	19453561
1	NY	2020-01-29	0	0	19453561
2	NY	2020-02-05	0	0	19453561
3	NY	2020-02-12	0	0	19453561
4	NY	2020-02-19	0	0	19453561
…	…	…	…	…	…
155	NY	2023-01-11	45340093	521498	19453561
156	NY	2023-01-18	45520292	523754	19453561
157	NY	2023-01-25	45718345	525272	19453561
158	NY	2023-02-01	45854147	527091	19453561
159	NY	2023-02-08	26271380	302408	19453561

160 rows × 5 columns

In [149…
```
#Adding new columns to see the no. of new cases and deaths each week.
NY_weekly_pattern = NY_weekly_pattern.assign(new_cases = NY_weekly_pattern.cases.diff(
NY_weekly_pattern = NY_weekly_pattern.assign(new_deaths = NY_weekly_pattern.deaths.dif

#Handling Nan values converting values to int
NY_weekly_pattern["new_cases"] = NY_weekly_pattern["new_cases"].fillna(0).astype(int)
NY_weekly_pattern["new_deaths"] = NY_weekly_pattern["new_deaths"].fillna(0).astype(int

#Handling negative values
NY_weekly_pattern.loc[NY_weekly_pattern['new_cases'] < 0, 'new_cases'] = 0
NY_weekly_pattern.loc[NY_weekly_pattern['new_deaths'] < 0, 'new_deaths'] = 0
NY_weekly_pattern
```

Out[1497]:

	State	date	cases	deaths	population	new_cases	new_deaths
0	NY	2020-01-22	0	0	19453561	0	0
1	NY	2020-01-29	0	0	19453561	0	0
2	NY	2020-02-05	0	0	19453561	0	0
3	NY	2020-02-12	0	0	19453561	0	0
4	NY	2020-02-19	0	0	19453561	0	0
...	...	...	...	...	...	...	...
155	NY	2023-01-11	45340093	521498	19453561	266995	2119
156	NY	2023-01-18	45520292	523754	19453561	180199	2256
157	NY	2023-01-25	45718345	525272	19453561	198053	1518
158	NY	2023-02-01	45854147	527091	19453561	135802	1819
159	NY	2023-02-08	26271380	302408	19453561	0	0

160 rows × 7 columns

In [149…

```
eps = 1e-8 #adding a small constant to avoid warning
NY_weekly_pattern['new_cases_log_norm'] = np.log(NY_weekly_pattern['new_cases'] + eps)
NY_weekly_pattern['new_deaths_log_norm'] = np.log(NY_weekly_pattern['new_deaths'] + ep

NY_weekly_pattern.loc[NY_weekly_pattern['new_cases_log_norm'] < 0, 'new_cases_log_norm
NY_weekly_pattern.loc[NY_weekly_pattern['new_deaths_log_norm'] < 0, 'new_deaths_log_no
NY_weekly_pattern
```

Out[1498]:

	State	date	cases	deaths	population	new_cases	new_deaths	new_cases_log_norm	new
0	NY	2020-01-22	0	0	19453561	0	0	0.000000	
1	NY	2020-01-29	0	0	19453561	0	0	0.000000	
2	NY	2020-02-05	0	0	19453561	0	0	0.000000	
3	NY	2020-02-12	0	0	19453561	0	0	0.000000	
4	NY	2020-02-19	0	0	19453561	0	0	0.000000	
...	...	...	...	...	...	...	...	...	
155	NY	2023-01-11	45340093	521498	19453561	266995	2119	12.494985	
156	NY	2023-01-18	45520292	523754	19453561	180199	2256	12.101817	
157	NY	2023-01-25	45718345	525272	19453561	198053	1518	12.196290	
158	NY	2023-02-01	45854147	527091	19453561	135802	1819	11.818953	
159	NY	2023-02-08	26271380	302408	19453561	0	0	0.000000	

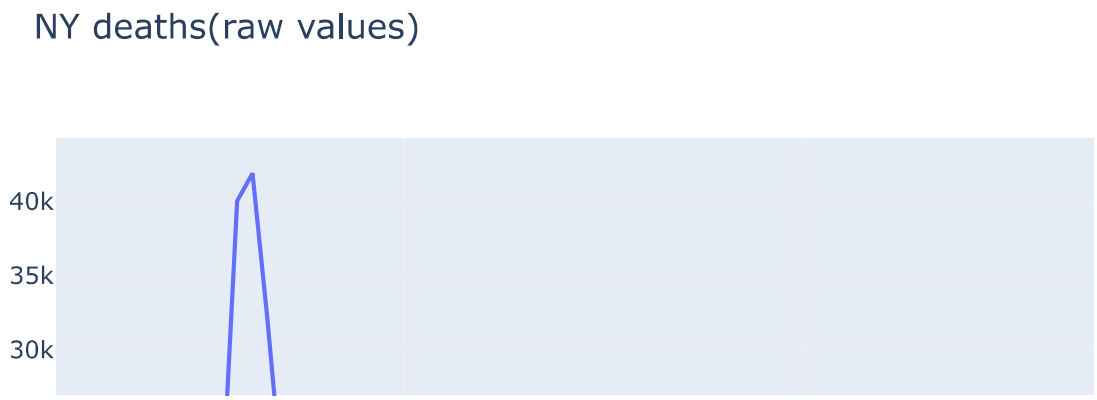
160 rows × 9 columns

In [149…

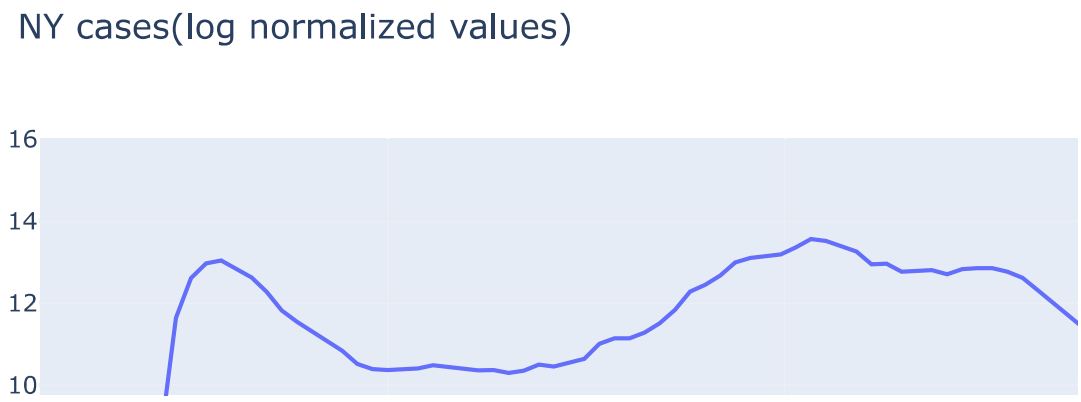
```
cases_raw_values_NY = px.line(NY_weekly_pattern, x = 'date', y = 'new_cases', color='S
cases_raw_values_NY.show()
```

In [150…

```
deaths_raw_values_NY = px.line(NY_weekly_pattern, x = 'date', y = 'new_deaths', color=
deaths_raw_values_NY.show()
```



In [150…
```
cases_log_values_NY = px.line(NY_weekly_pattern, x = 'date', y = ['new_cases_log_norm'
cases_log_values_NY.show()
```



In [150…
```
deaths_log_values_NY = px.line(NY_weekly_pattern, x = 'date', y = ['new_deaths_log_nor
deaths_log_values_NY.show()
```

Plot weekly trends (new cases and deaths) for the top 3 infected counties. Show plots by raw values and log normalized values. Describe what is causing them and what were the peaks. Do the counties follow state pattern.

In [150…

```
NY_weekly_data = superdata[superdata['State']=='NY']
counties_data = NY_weekly_data.loc[NY_weekly_data['County Name'].isin(top_3)][['County
counties_data
```

Out[1503]:

	County Name	date	cases	deaths	population
2053128	kings county	2020-01-22	0	0	2559903
2053129	kings county	2020-01-23	0	0	2559903
2053130	kings county	2020-01-24	0	0	2559903
2053131	kings county	2020-01-25	0	0	2559903
2053132	kings county	2020-01-26	0	0	2559903
...	...	...	...	...	...
2085342	suffolk county	2023-02-01	565311	4907	1476601
2085343	suffolk county	2023-02-02	565311	4922	1476601
2085344	suffolk county	2023-02-03	565311	4922	1476601
2085345	suffolk county	2023-02-04	565311	4922	1476601
2085346	suffolk county	2023-02-05	565311	4922	1476601

3333 rows × 5 columns

In [150…

```
counties_data['date'] = pd.to_datetime(counties_data['date'])
counties_weekly = counties_data.groupby(['County Name', pd.Grouper(key="date", freq="W
    "cases": "sum",
    "deaths": "sum",
    "population": "first"
}).reset_index()
counties_weekly
```

Out[1504]:

	County Name	date	cases	deaths	population
0	kings county	2020-01-22	0	0	2559903
1	kings county	2020-01-29	0	0	2559903
2	kings county	2020-02-05	0	0	2559903
3	kings county	2020-02-12	0	0	2559903
4	kings county	2020-02-19	0	0	2559903
...	...	...	...	...	...
475	suffolk county	2023-01-11	3908765	33925	1476601
476	suffolk county	2023-01-18	3925012	34055	1476601
477	suffolk county	2023-01-25	3939130	34223	1476601
478	suffolk county	2023-02-01	3948759	34335	1476601
479	suffolk county	2023-02-08	2261244	19688	1476601

480 rows × 5 columns

In [150…

```
counties_weekly = counties_weekly.assign(new_cases = counties_weekly.cases.diff())
counties_weekly = counties_weekly.assign(new_deaths = counties_weekly.deaths.diff())
```

```
counties_weekly["new_cases"] = counties_weekly["new_cases"].fillna(0).astype(int)
counties_weekly["new_deaths"] = counties_weekly["new_deaths"].fillna(0).astype(int)

counties_weekly.loc[counties_weekly['new_cases'] < 0, 'new_cases'] = 0
counties_weekly.loc[counties_weekly['new_deaths'] < 0, 'new_deaths'] = 0

eps = 1e-8 #adding a small constant to avoid warning
counties_weekly['new_cases_log_norm'] = np.log(counties_weekly['new_cases'] + eps)
counties_weekly['new_deaths_log_norm'] = np.log(counties_weekly['new_deaths'] + eps)

counties_weekly.loc[counties_weekly['new_cases_log_norm'] < 0, 'new_cases_log_norm'] =
counties_weekly.loc[counties_weekly['new_deaths_log_norm'] < 0, 'new_deaths_log_norm']
counties_weekly
```

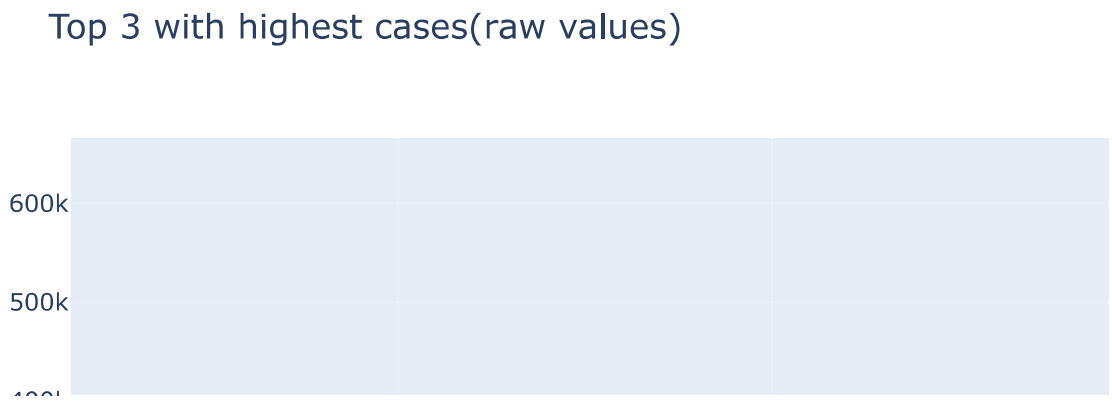
Out[1505]:

	County Name	date	cases	deaths	population	new_cases	new_deaths	new_cases_log_norm	new
0	kings county	2020-01-22	0	0	2559903	0	0	0.000000	
1	kings county	2020-01-29	0	0	2559903	0	0	0.000000	
2	kings county	2020-02-05	0	0	2559903	0	0	0.000000	
3	kings county	2020-02-12	0	0	2559903	0	0	0.000000	
4	kings county	2020-02-19	0	0	2559903	0	0	0.000000	
...	...	...	...	...	...	...	...	...	
475	suffolk county	2023-01-11	3908765	33925	1476601	28952	182	10.273395	
476	suffolk county	2023-01-18	3925012	34055	1476601	16247	130	9.695664	
477	suffolk county	2023-01-25	3939130	34223	1476601	14118	168	9.555206	
478	suffolk county	2023-02-01	3948759	34335	1476601	9629	112	9.172535	
479	suffolk county	2023-02-08	2261244	19688	1476601	0	0	0.000000	

480 rows × 9 columns

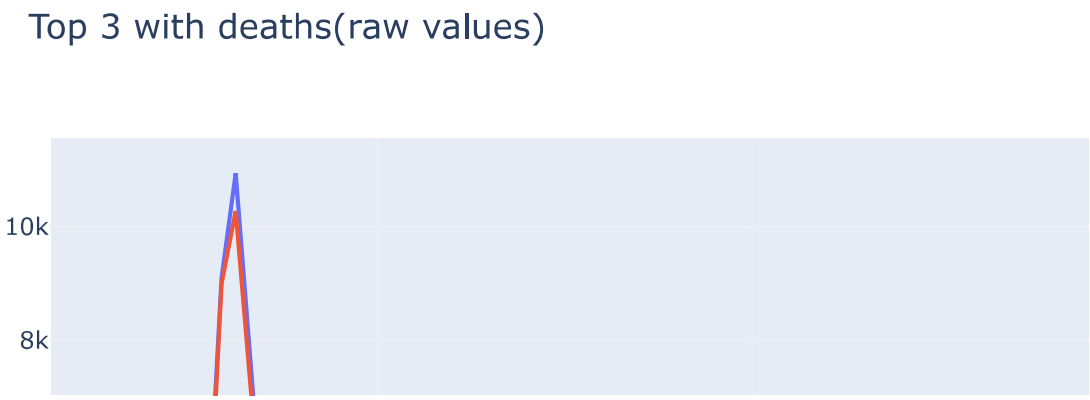
In [150…

```
cases_raw_values = px.line(counties_weekly, x = 'date', y = 'new_cases', color='County
cases_raw_values.show()
```

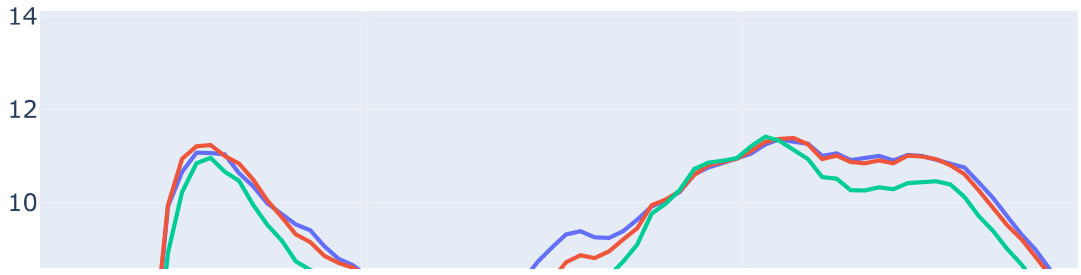


In [150…

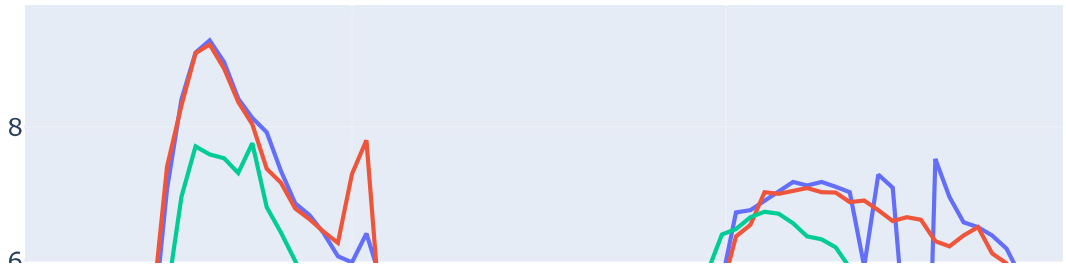
```
deaths_raw_values = px.line(counties_weekly, x = 'date', y = 'new_deaths', color='Cour
deaths_raw_values.show()
```



In [150…
```
cases_log_values = px.line(counties_weekly, x = 'date', y = ['new_cases_log_norm'], co
cases_log_values.show()
```



In [150…
```
deaths_log_values = px.line(counties_weekly, x = 'date', y = ['new_deaths_log_norm'],
deaths_log_values.show()
```



From the plots we understand that, the cases and deaths plot of log normalized values looks more symmetrical and peaks are more clear allowing us to understand the distribution more clear. When compared to all of NY plot above, these counties follow the same pattern.