# Task:-

## 1. Use the state data (the state of your choice) generated in Stage II to fit a distribution (not Poisson) to the number of COVID-19 cases. (25 points)

- **[A]** `Graphically plot the distribution and describe the distribution statistics. If using discrete values, calculate the Probability Mass Function for the individual values or range (if using histogram) and plot that.`
- **[B]** `Describe the type of distribution and its statistics (eg., center, variance, skewness, kurtosis) in the report and the notebook.`
- **[C]** `Compare the distribution and its statistics to 5 other states of your choosing. Describe if the distributions look different and what does that imply.`

## 2. Model a Poisson distribution of COVID-19 cases and deaths of a state and compare to other five states. Describe how the Poisson modeling is different from the first modeling you did. (25 points)

- Example, number of new cases and deaths per 100,000 population.
- Hint - the parameter for a Poisson's distribution will be its mean value. Then for the minimum and maximum range of covid cases you are calculating probability mass function to observe the probability at different points.

## 3. Perform correlation between Enrichment data valiables and COVID-19 cases to observe any patterns. (20 points)

- You can compare either within your chosen specific state or among different states with the different enrichment variables. Within the state you can compare the county based covid data to enrichment data for correlation. Between states you would need to aggregate to state level data and then perform correlation. Both covid and enrichment data will need to be normalized for population. For number of covid cases you can use a measure of center value (median or mean) to compare the number of cases.
- **[A]** `Formulate hypothesis between Enrichment data and number of cases to be compared against states. Choose 3 different variables to compare against. (30 points)`
- For example: Does higher employment data lead to higher covid case numbers or more rapid increase in covid cases.

---

# 1. Use the state data (the state of your choice) generated in Stage II to fit a distribution (not Poisson) to the number of COVID-19 cases. (25 points)

First we import necessary library

In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import scipy.stats as stats
```

Import the super_covid19 dataset that we created in Phase-2

In [9]:
```
super_covid19=pd.read_csv('super_covid19_dataframe.csv')
print("(Rows, Columns)=",super_covid19.shape)
super_covid19.head(5)
```

```
(Rows, Columns)= (3142, 2535)
```

Out[9]:

| | countyFIPS | County Name | State | StateFIPS | 2020-01-22_cases | 2020-01-23_cases | 2020-01-24_cases | 2020-01-25_cases | 2020-01-26_cases | 202 0 27_cas |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 1001 | Autauga County | AL | 1 | 0 | 0 | 0 | 0 | 0 | |
| **1** | 1003 | Baldwin County | AL | 1 | 0 | 0 | 0 | 0 | 0 | |
| **2** | 1005 | Barbour County | AL | 1 | 0 | 0 | 0 | 0 | 0 | |
| **3** | 1007 | Bibb County | AL | 1 | 0 | 0 | 0 | 0 | 0 | |
| **4** | 1009 | Blount County | AL | 1 | 0 | 0 | 0 | 0 | 0 | |

5 rows × 2535 columns

Write code for filtering the dataset for below task:-

- filter the data for state data (StateFIPS=4 , StateName= 'Arizona')
- filter the data for COVID-19 cases only
- filter the data for date starts from 2020.6.1(Monday) - 2021.1.3(Sunday) from Stage II

In [11]:
```
state_data = super_covid19[super_covid19['StateFIPS'] == 4]
state_data
```

Out[11]:

| | countyFIPS | County Name | State | StateFIPS | 2020-01-22_cases | 2020-01-23_cases | 2020-01-24_cases | 2020-01-25_cases | 2020-01-26_cases | 27 |
|---|---|---|---|---|---|---|---|---|---|---|
| **96** | 4001 | Apache County | AZ | 4 | 0 | 0 | 0 | 0 | 0 | |
| **97** | 4003 | Cochise County | AZ | 4 | 0 | 0 | 0 | 0 | 0 | |
| **98** | 4005 | Coconino County | AZ | 4 | 0 | 0 | 0 | 0 | 0 | |
| **99** | 4007 | Gila County | AZ | 4 | 0 | 0 | 0 | 0 | 0 | |
| **100** | 4009 | Graham County | AZ | 4 | 0 | 0 | 0 | 0 | 0 | |
| **101** | 4011 | Greenlee County | AZ | 4 | 0 | 0 | 0 | 0 | 0 | |
| **102** | 4012 | La Paz County | AZ | 4 | 0 | 0 | 0 | 0 | 0 | |
| **103** | 4013 | Maricopa County | AZ | 4 | 0 | 0 | 0 | 0 | 1 | |
| **104** | 4015 | Mohave County | AZ | 4 | 0 | 0 | 0 | 0 | 0 | |
| **105** | 4017 | Navajo County | AZ | 4 | 0 | 0 | 0 | 0 | 0 | |
| **106** | 4019 | Pima County | AZ | 4 | 0 | 0 | 0 | 0 | 0 | |
| **107** | 4021 | Pinal County | AZ | 4 | 0 | 0 | 0 | 0 | 0 | |
| **108** | 4023 | Santa Cruz County | AZ | 4 | 0 | 0 | 0 | 0 | 0 | |
| **109** | 4025 | Yavapai County | AZ | 4 | 0 | 0 | 0 | 0 | 0 | |
| **110** | 4027 | Yuma County | AZ | 4 | 0 | 0 | 0 | 0 | 0 | |

15 rows × 2535 columns

In [92]:

```
state_data = super_covid19[super_covid19['StateFIPS'] == 4]
#print(state_data)
state_data_filter = [col for col in state_data.columns
                     if ('_cases' in col)  and '2020-06-01' <= col.split('_')[0] <= '
df=pd.concat([state_data['countyFIPS'],state_data[state_data_filter]],axis=1)
df.index=df.countyFIPS
del df['countyFIPS']
df  #dataframe with countyFIPS and its corresponding covid cases
```

Out[92]:

| countyFIPS | 2020-06-01_cases | 2020-06-02_cases | 2020-06-03_cases | 2020-06-04_cases | 2020-06-05_cases | 2020-06-06_cases | 2020-06-07_cases | 2020-06-08_cases | 202 0 09_cas |
|---|---|---|---|---|---|---|---|---|---|
| **4001** | 1526 | 1569 | 1586 | 1637 | 1656 | 1692 | 1727 | 1732 | 17 |
| **4003** | 76 | 89 | 94 | 105 | 120 | 122 | 141 | 149 | 1 |
| **4005** | 1155 | 1173 | 1186 | 1221 | 1248 | 1267 | 1282 | 1289 | 12 |
| **4007** | 31 | 35 | 37 | 39 | 42 | 43 | 45 | 46 | |
| **4009** | 32 | 37 | 38 | 38 | 39 | 39 | 40 | 41 | |
| **4011** | 6 | 8 | 8 | 8 | 9 | 9 | 10 | 10 | |
| **4012** | 79 | 91 | 91 | 110 | 149 | 158 | 181 | 183 | 1 |
| **4013** | 9937 | 10536 | 11068 | 11229 | 12091 | 12761 | 13498 | 14003 | 143 |
| **4015** | 403 | 409 | 422 | 428 | 447 | 485 | 500 | 512 | 5 |
| **4017** | 1873 | 1957 | 1994 | 2042 | 2104 | 2152 | 2198 | 2229 | 22 |
| **4019** | 2382 | 2496 | 2627 | 2669 | 2883 | 2950 | 3098 | 3154 | 31 |
| **4021** | 865 | 909 | 940 | 948 | 1018 | 1067 | 1112 | 1127 | 11 |
| **4023** | 330 | 365 | 438 | 462 | 503 | 530 | 599 | 615 | 6 |
| **4025** | 297 | 300 | 304 | 307 | 314 | 326 | 327 | 330 | 3 |
| **4027** | 1131 | 1275 | 1387 | 1510 | 1708 | 1850 | 2131 | 2257 | 23 |

15 rows × 217 columns

Code to display countyFIPS in column and covid cases in each rows according to it's date

In [93]:
```
df=df.T
df
```

Out[93]:

| countyFIPS | 4001 | 4003 | 4005 | 4007 | 4009 | 4011 | 4012 | 4013 | 4015 | 4017 | 4019 | 4021 | 4 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2020-06-01_cases | 1526 | 76 | 1155 | 31 | 32 | 6 | 79 | 9937 | 403 | 1873 | 2382 | 865 | |
| 2020-06-02_cases | 1569 | 89 | 1173 | 35 | 37 | 8 | 91 | 10536 | 409 | 1957 | 2496 | 909 | |
| 2020-06-03_cases | 1586 | 94 | 1186 | 37 | 38 | 8 | 91 | 11068 | 422 | 1994 | 2627 | 940 | |
| 2020-06-04_cases | 1637 | 105 | 1221 | 39 | 38 | 8 | 110 | 11229 | 428 | 2042 | 2669 | 948 | |
| 2020-06-05_cases | 1656 | 120 | 1248 | 42 | 39 | 9 | 149 | 12091 | 447 | 2104 | 2883 | 1018 | |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | |
| 2020-12-30_cases | 7336 | 7324 | 10982 | 4419 | 3180 | 392 | 1387 | 314464 | 12108 | 11196 | 68437 | 27191 | 5 |
| 2020-12-31_cases | 7438 | 7480 | 11126 | 4517 | 3228 | 400 | 1464 | 318827 | 12573 | 11296 | 69522 | 27649 | 6 |
| 2021-01-01_cases | 7594 | 7585 | 11344 | 4552 | 3287 | 403 | 1523 | 325404 | 12892 | 11559 | 70563 | 28137 | 6 |
| 2021-01-02_cases | 7661 | 7673 | 11437 | 4596 | 3370 | 409 | 1525 | 331233 | 13057 | 11582 | 71894 | 28482 | 6 |
| 2021-01-03_cases | 7758 | 7873 | 11691 | 4684 | 3414 | 415 | 1609 | 342994 | 13590 | 11865 | 74108 | 29093 | 6 |

217 rows × 15 columns

Code to convert above county wise data into state wise.

In [94]:
```
df['state_cases'] = df.sum(axis=1) # axis =1 means rows
df_state_cases=df.state_cases
df_state_cases
```

Out[94]:
```
2020-06-01_cases     20123
2020-06-02_cases     21249
2020-06-03_cases     22220
2020-06-04_cases     22753
2020-06-05_cases     24331
                     ...
2020-12-30_cases    512489
2020-12-31_cases    520207
2021-01-01_cases    530267
2021-01-02_cases    539148
2021-01-03_cases    556384
Name: state_cases, Length: 217, dtype: int64
```

Convert the above state wise COVID data into array for future calculation

In [66]:
```
arizona_cases_array = df_state_cases.values # Convert to array
print(arizona_cases_array)
len(arizona_cases_array)
```

```
[ 20123  21249  22220  22753  24331  25451  26889  27677  28296  29852
  31263  32918  34458  35691  36705  39097  40924  43443  46688  49798
  52390  54586  58353  59973  63030  66458  70049  73907  74532  79204
  84092  87424  91857  94553  98084 101441 105094 108614 112671 116892
 119930 122467 123824 128097 131354 134611 138523 141265 143619 145182
 148674 150609 152944 156293 160041 162003 163825 165934 168266 170798
 174010 177000 178467 179490 180498 182194 183645 185050 186103 186923
 187521 188736 189442 190791 191721 192654 193537 194005 194917 195557
 196280 196898 197872 198075 198386 199252 199459 200139 200658 201285
 201661 201835 202342 202861 203952 204679 205513 205765 205954 206044
 206541 206993 207521 208126 208512 208725 209209 209904 211625 212912
 213544 214015 214401 214841 215283 215849 216365 216820 217230 217506
 218183 218507 219212 219763 220398 220752 221070 221931 222538 223400
 224084 224970 225575 226050 226731 227635 228710 229434 230406 231147
 231863 232937 233912 234906 235881 236771 238158 238963 240119 241165
 242480 244045 245946 247473 248139 249818 250632 252767 254763 257384
 259264 259681 263133 265163 266562 269577 273053 275435 276912 279896
 283102 287225 291696 295334 299665 302324 306867 310849 314324 318638
 322774 325995 326817 337139 340979 346421 352101 358900 364276 365842
 378157 382601 387529 394512 402588 409264 419930 424371 429219 435036
 442671 448231 453597 453597 453597 453597 453597 453597 453597 453597
 504423 507222 512489 520207 530267 539148 556384]
```

Out[66]:
```
217
```

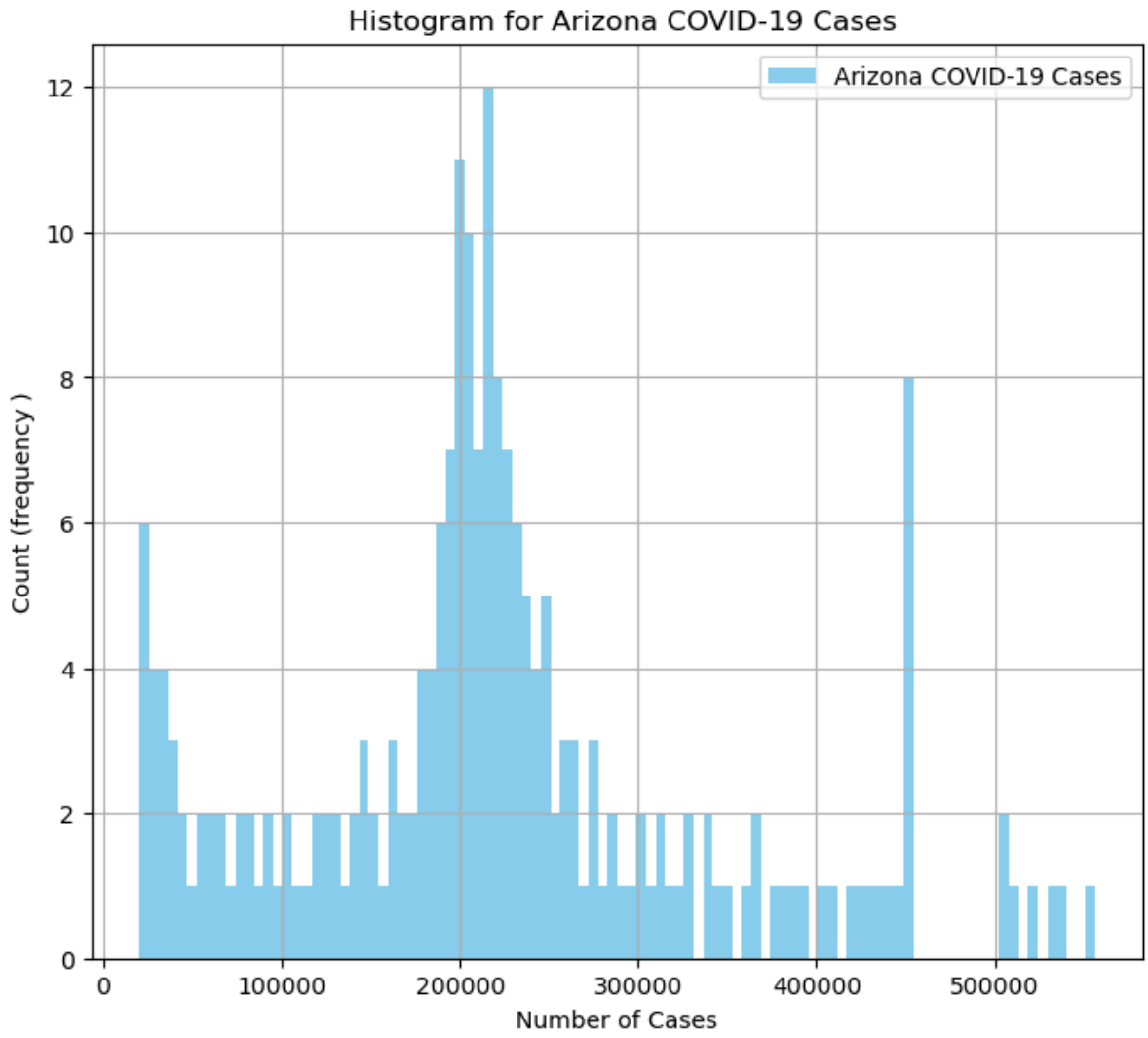
## [A] Graphically plot the distribution and describe the distribution statistics. If using discrete values, calculate the Probability Mass Function for the individual values or range (if using histogram) and plot that.

Designed the histogram which show the graph of covid cases and it's count(frequency)

- The histogram can give the shape of the data, the center, and the spread of the data.

In [110…

```
plt.figure(figsize=(8, 7))
plt.hist(arizona_cases_array, bins=100, density=False, color='skyblue', label='Ariz


# Add labels and legend
plt.title(' Histogram for Arizona COVID-19 Cases')
plt.xlabel('Number of Cases')
plt.ylabel('Count (frequency )')
plt.legend()
plt.grid(True)

# Show the plot
plt.show()
```

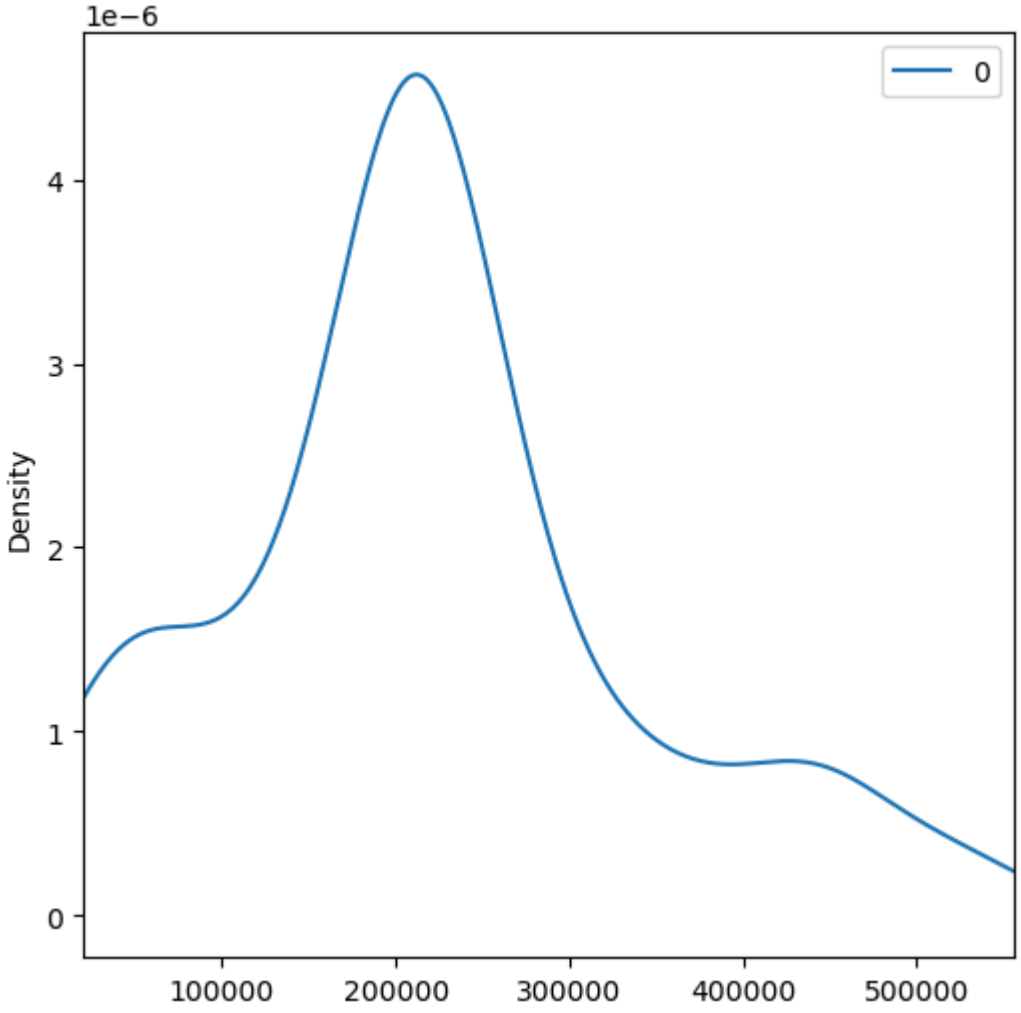



Probability Density Plot:-

- We have data of daily counts of new cases, which are discrete values. So we can say that our data is discrete.
- Density curve used to visualize the smoothed approximation of the distribution of discrete values. This creates a continuous-looking curve that can help understand the overall pattern of the data.
- So, the data is discrete, but the density curve is a continuous representation of its distribution.

In [53]:
```
pd.DataFrame(arizona_cases_array).plot(kind="density",  # Plot the distribution
                                       figsize=(6,6),xlim=(df_state_cases.values.min(),df_s
```

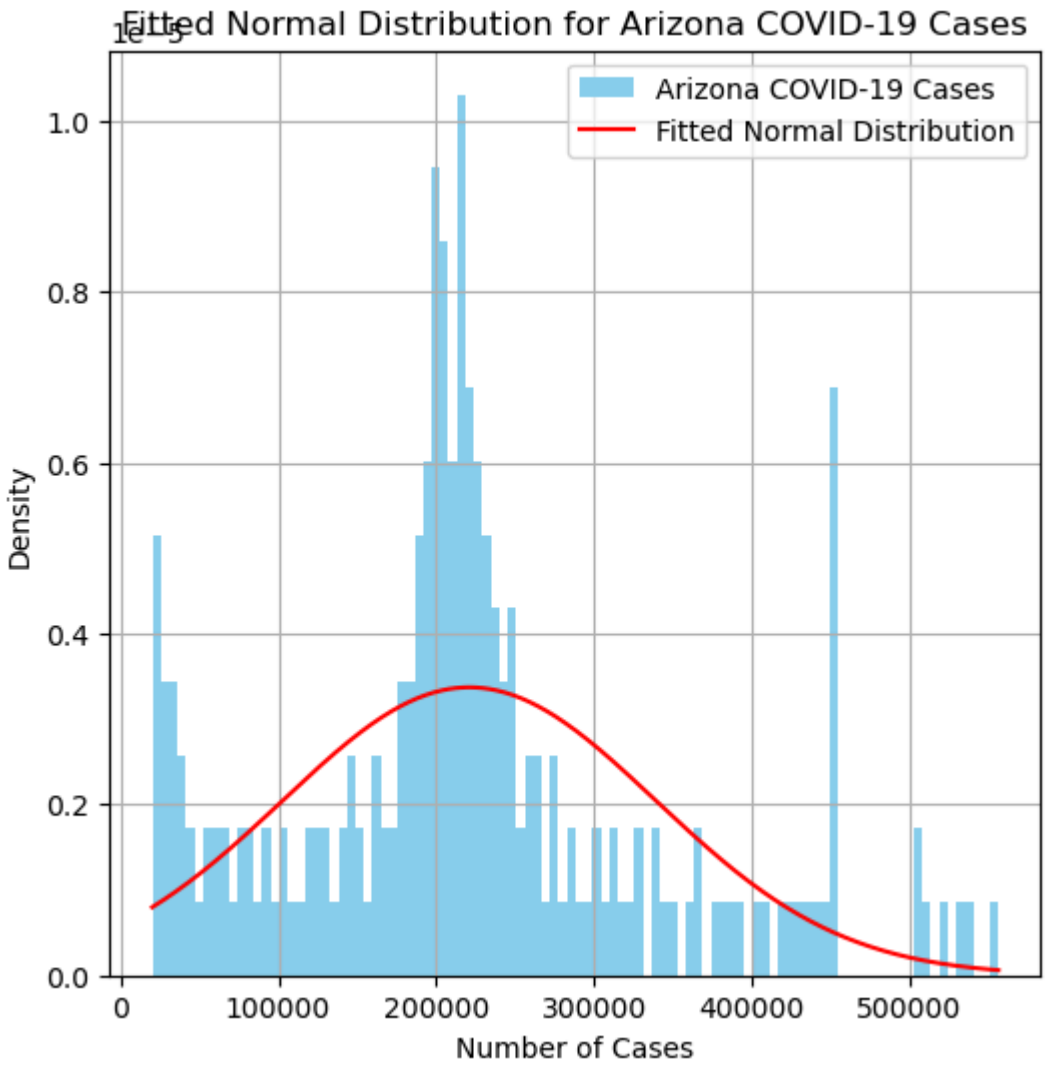
Out[53]:
```
<AxesSubplot:ylabel='Density'>
```




Graphically plot the distribution and try to fit a normal distribution to the number of COVID-19 cases.

In [109…

```
min_cases = df_state_cases.values.min()
max_cases = df_state_cases.values.max()
histogram_range = max_cases - min_cases

mean, std_dev = stats.norm.fit(arizona_cases_array) # Fit a normal distribution
# Create a range of values for plotting the fitted distribution
x = np.linspace(min(arizona_cases_array), max(arizona_cases_array), 500)
# Plot the histogram and fitted distribution

plt.figure(figsize=(6,6))
plt.hist(arizona_cases_array, bins=100, density=True, color='skyblue', label='Arizo
plt.plot(x, stats.norm.pdf(x, mean, std_dev), 'r-', label='Fitted Normal Distributi

plt.title('Fitted Normal Distribution for Arizona COVID-19 Cases')
plt.xlabel('Number of Cases')
plt.ylabel('Density')
plt.legend()
plt.grid(True)
plt.show()

# Calculate additional statistics
variance = np.var(arizona_cases_array)
skewness = stats.skew(arizona_cases_array)
kurtosis = stats.kurtosis(arizona_cases_array)
```




Calculate the Probability Mass Function for the individual values and plot it.

- Calculate PMF: The PMF for each distinct case count is calculated as:
- $PMF(x)$=(Frequency of $x$) / (Total Number of Observations)

In [100…
```
df_state_cases=pd.DataFrame(df_state_cases) # Conver to dataframe
case_counts=df_state_cases.state_cases.value_counts()
case_counts
```

Out[100]:
```
453597    8
20123     1
235881    1
226050    1
226731    1
         ..
190791    1
191721    1
192654    1
193537    1
556384    1
Name: state_cases, Length: 210, dtype: int64
```

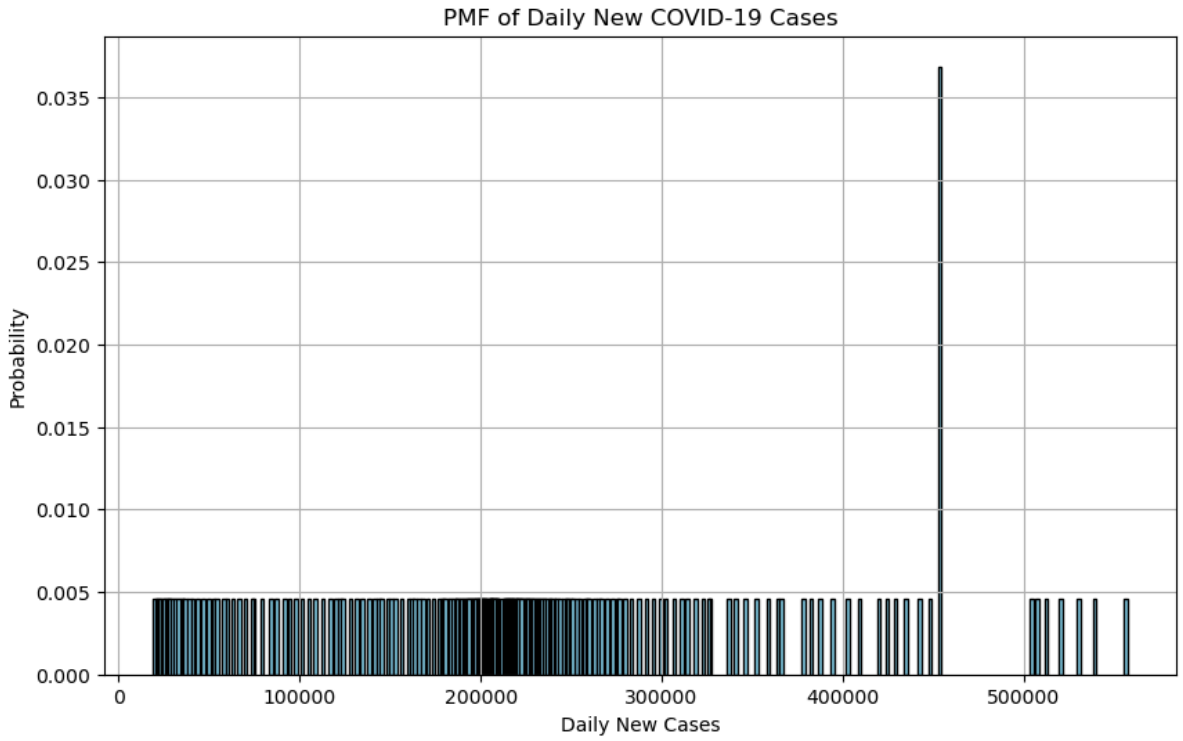
Calculate the PMF

In [101…
```
total_cases=len(df_state_cases) #217
# Calculate the PMF (probability of each case count)
pmf = case_counts / total_cases
pmf
```

Out[101]:
```
453597    0.036866
20123     0.004608
235881    0.004608
226050    0.004608
226731    0.004608
            ...
190791    0.004608
191721    0.004608
192654    0.004608
193537    0.004608
556384    0.004608
Name: state_cases, Length: 210, dtype: float64
```

Plot the PMF for the individual values.

In [108…
```
plt.figure(figsize=(10,6))
plt.bar(pmf.index, pmf.values, width=2000,color='skyblue', edgecolor='black')
plt.title('PMF of Daily New COVID-19 Cases')
plt.xlabel('Daily New Cases')
plt.ylabel('Probability')
plt.grid(True)
plt.show()
```




## [B] Describe the type of distribution and its statistics (eg., center, variance, skewness, kurtosis) in the report and the notebook.

We can discribe the above distribution as `right-skewed continuous distribution`.

Distribution statistics:-

In [112…
```
print('mean:',mean)
print('std_dev:',std_dev)
print('variance:',variance)
print('skewness:',skewness)
print('kurtosis:',kurtosis)
print('histogram_range:',histogram_range)
```

```
mean: 221013.68202764977
std_dev: 118309.84827542547
variance: 13997220198.954193
skewness: 0.5844429590341947
kurtosis: 0.2790115895818883
histogram_range: 536261
```

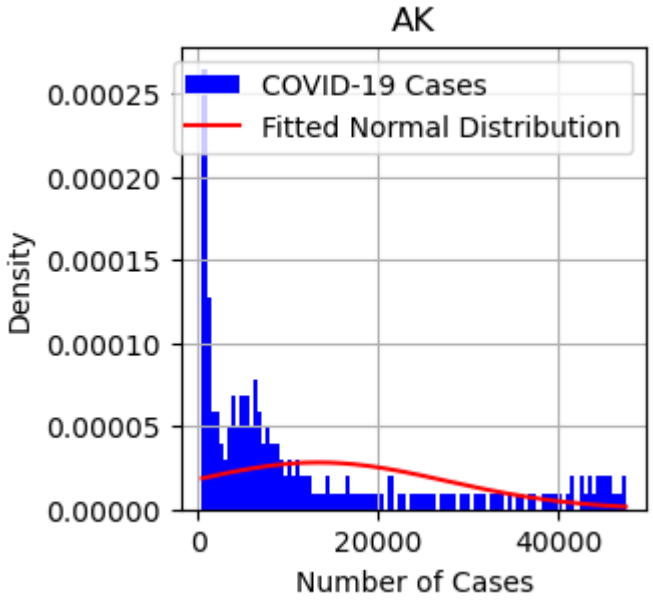
## [C]Compare the distribution and its statistics to 5 other states of your choosing. Describe if the distributions look different and what does that imply.

Compare the distribution and its statistics to 5 other states of your choosing.

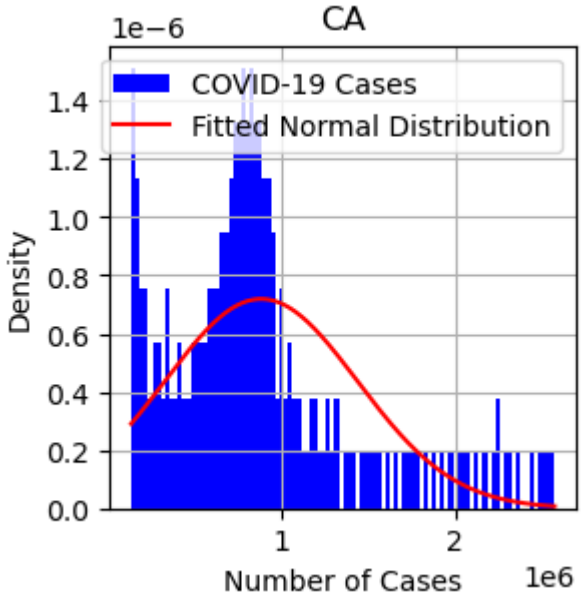
In [113…

```
def data_five_states(n,state_name):
    state_data = super_covid19[super_covid19['StateFIPS'] == n]
    state_data_filter = [col for col in state_data.columns
                  if ('_cases' in col)  and '2020-06-01' <= col.split('_')[0] <= '
    df_2020=state_data.drop(['County Name','State','StateFIPS','population'],axis=1
    df_2020=df_2020[state_data_filter]
    df=df_2020.T
    df['row_sum'] = df.sum(axis=1) # axis =1 means rows
    df_state_cases=df.row_sum
    state_cases_array = df_state_cases.values
    show_graph(state_cases_array,state_name)

def show_graph(state_cases_array,state_name):
    min_cases = state_cases_array.min()
    max_cases = state_cases_array.max()
    histogram_range = max_cases - min_cases

    mean, std_dev = stats.norm.fit(state_cases_array) # Fit a normal distributio
    # Create a range of values for plotting the fitted distribution
    x = np.linspace(min(state_cases_array), max(state_cases_array), 500)
    # Plot the histogram and fitted distribution

    plt.figure(figsize=(3, 3))
    plt.hist(state_cases_array, bins=100, density=True, color='b', label='COVID-19
    plt.plot(x, stats.norm.pdf(x, mean, std_dev), 'r-', label='Fitted Normal Distri
    plt.title(state_name)
    plt.xlabel('Number of Cases')
    plt.ylabel('Density')
    plt.legend()
    plt.grid(True)
    plt.show()
    # Calculate additional statistics
    variance = np.var(state_cases_array)
    skewness = stats.skew(state_cases_array)
    kurtosis = stats.kurtosis(state_cases_array)
    print_statistics(mean,std_dev,variance,skewness,kurtosis,histogram_range)

def print_statistics(mean,std_dev,variance,skewness,kurtosis,histogram_range):
    print('mean:',mean)
    print('std_dev:',std_dev)
    print('variance:',variance)
    print('skewness:',skewness)
    print('kurtosis:',kurtosis)
    print('histogram_range:',histogram_range)


statefips_list = [2, 6, 12, 13, 36]
for statefips in statefips_list:
```

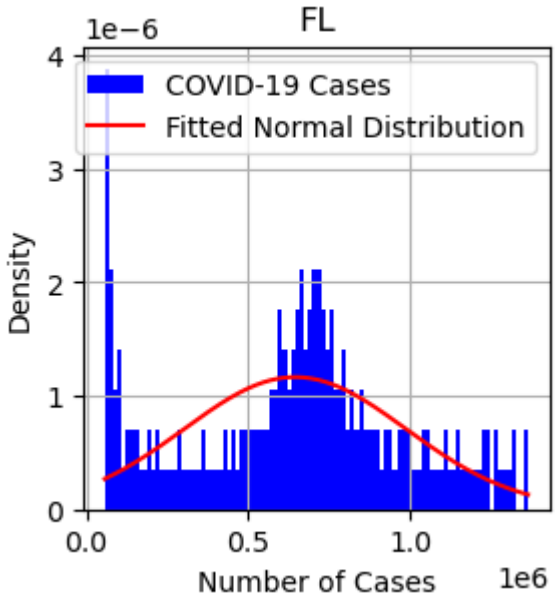
```
    state_name=super_covid19[super_covid19['StateFIPS'] == statefips].State.iloc[0]
    data_five_states(statefips,state_name)
```



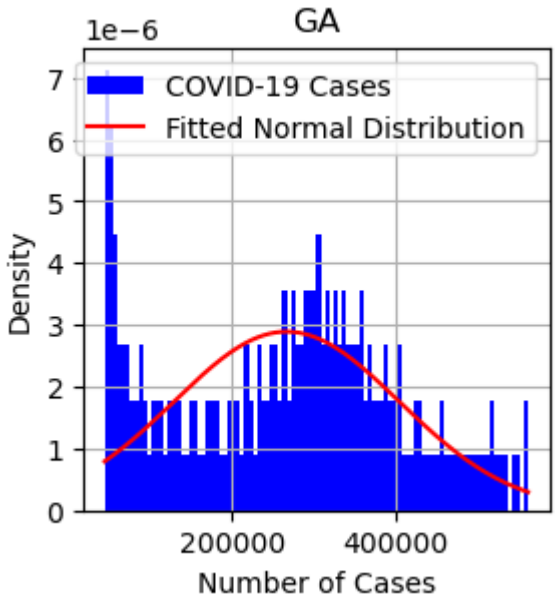

```
mean: 13352.52534562212
std_dev: 14189.386673045165
variance: 201338694.15719172
skewness: 1.1518453268963726
kurtosis: -0.034628467461078394
histogram_range: 46972
```



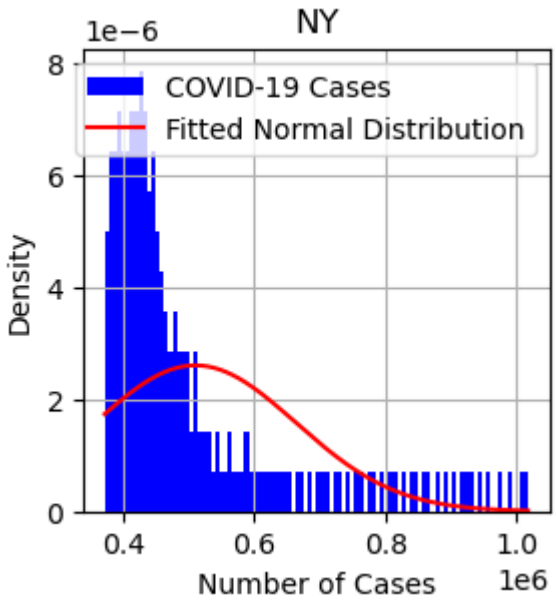

```
mean: 883644.9631336406
std_dev: 554753.6895998059
variance: 307751656124.5977
skewness: 1.1765055504542024
kurtosis: 1.1150816591833488
histogram_range: 2435364
```




```
mean: 644027.9539170507
std_dev: 342517.7879546051
variance: 117318435065.31584
skewness: -0.04895640315905616
kurtosis: -0.6818726374534347
histogram_range: 1305626
```




```
mean: 266488.7050691244
std_dev: 138047.2567307992
variance: 19057045090.89919
skewness: 0.0317078788159161
kurtosis: -0.8734370934793061
histogram_range: 517121
```




```
mean: 509919.83410138247
std_dev: 153409.94335022426
variance: 23534610718.71902
skewness: 1.681217112753293
kurtosis: 1.911882408874856
histogram_range: 645442
```

Explanation of the distributions look different and what does that imply.:-

- CA and NY have more extreme data distributions with higher skewness and kurtosis, meaning they have more outliers and greater variability.
- FL and GA have more symmetric, less peaked distributions, suggesting more even spread across their data.
- AK has the smallest variability and skewness, indicating more uniform data with less outlier presence.
- States like California and Florida experienced large and fluctuating COVID-19 outbreaks, with periods of extreme case surges, as reflected in their high means, variability, and positive skewness.
- Arizona had a moderate outbreak, with some periods of increased cases, but it was more stable compared to highly impacted states like CA.
- Alaska had a much smaller outbreak, with more consistent and lower case numbers, indicated by its low mean and low variability.
- States like New York and Georgia also experienced notable outbreaks, with NY showing more extreme events, while GA had a more stable case count.

# 2. Model a Poisson distribution of COVID-19 cases and deaths of a state and compare to other five states. Describe how the Poisson modeling is different from the first modeling you did. (25 points)

- Example, number of new cases and deaths per 100,000 population.
- Hint - the parameter for a Poisson's distribution will be its mean value. Then for the minimum and maximum range of covid cases you are calculating probability mass

function to observe the probability at different points.

# Cases:-

- Model a Poisson distribution of COVID-19 cases of a state.

In [118…

```
# show the COVID cases data for Arizona state
state_data = super_covid19[super_covid19['StateFIPS'] == 4]
#print(state_data)
state_data_filter = [col for col in state_data.columns
                     if ('_cases' in col)  and '2020-06-01' <= col.split('_')[0] <= '
df_2020=state_data.drop(['County Name','State','StateFIPS','population'],axis=1)
df_2020=df_2020[state_data_filter]
df_2020.head(5)
```

Out[118]:

| | 2020-06-01_cases | 2020-06-02_cases | 2020-06-03_cases | 2020-06-04_cases | 2020-06-05_cases | 2020-06-06_cases | 2020-06-07_cases | 2020-06-08_cases | 2020-06-09_cases | 10_ |
|---|---|---|---|---|---|---|---|---|---|---|
| **96** | 1526 | 1569 | 1586 | 1637 | 1656 | 1692 | 1727 | 1732 | 1747 | |
| **97** | 76 | 89 | 94 | 105 | 120 | 122 | 141 | 149 | 156 | |
| **98** | 1155 | 1173 | 1186 | 1221 | 1248 | 1267 | 1282 | 1289 | 1289 | |
| **99** | 31 | 35 | 37 | 39 | 42 | 43 | 45 | 46 | 47 | |
| **100** | 32 | 37 | 38 | 38 | 39 | 39 | 40 | 41 | 42 | |

5 rows × 217 columns

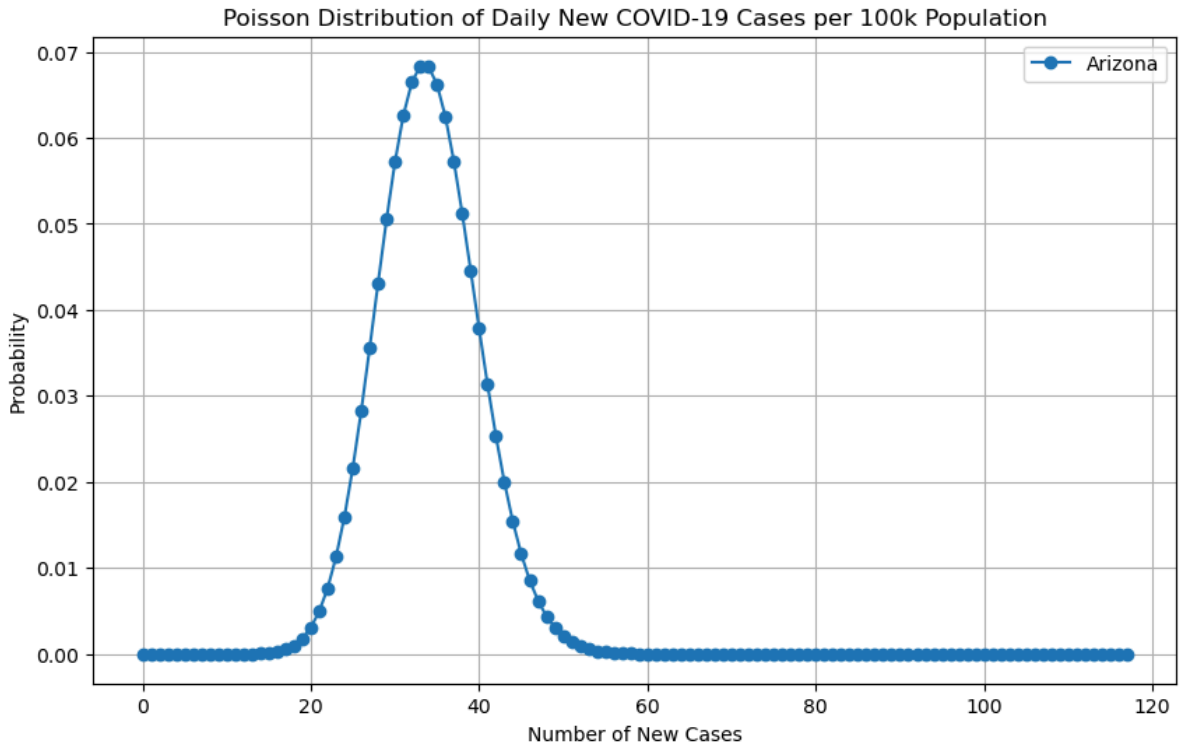
Poisson distribution of `Arizona` state with number of new cases per 100,000 population.

In [123…

```
from scipy.stats import poisson
df_sum=df_2020.sum(axis=0)
#print(df_sum)

for i in range(1,len(df_sum)):
    df_sum[i-1]=df_sum[i]-df_sum[i-1] # calculate new cases each day

df_sum[len(df_sum)-1]=0
#print(df_sum)

population = state_data['population'].sum()
print(population)
lamda=(df_sum.mean()/population)*100000
print(lamda)

plt.figure(figsize=(10, 6))
x_values = np.arange(0,int(lamda * 3.5))
pmf = poisson.pmf(x_values, lamda)
plt.plot(x_values, pmf,marker='o',label='Arizona')

# Customize the plot
plt.title('Poisson Distribution of Daily New COVID-19 Cases per 100k Population')
plt.xlabel('Number of New Cases')
plt.ylabel('Probability')
plt.grid(True)
```

```
plt.legend()
plt.show()
```

```
7278717
33.95170945547502
```

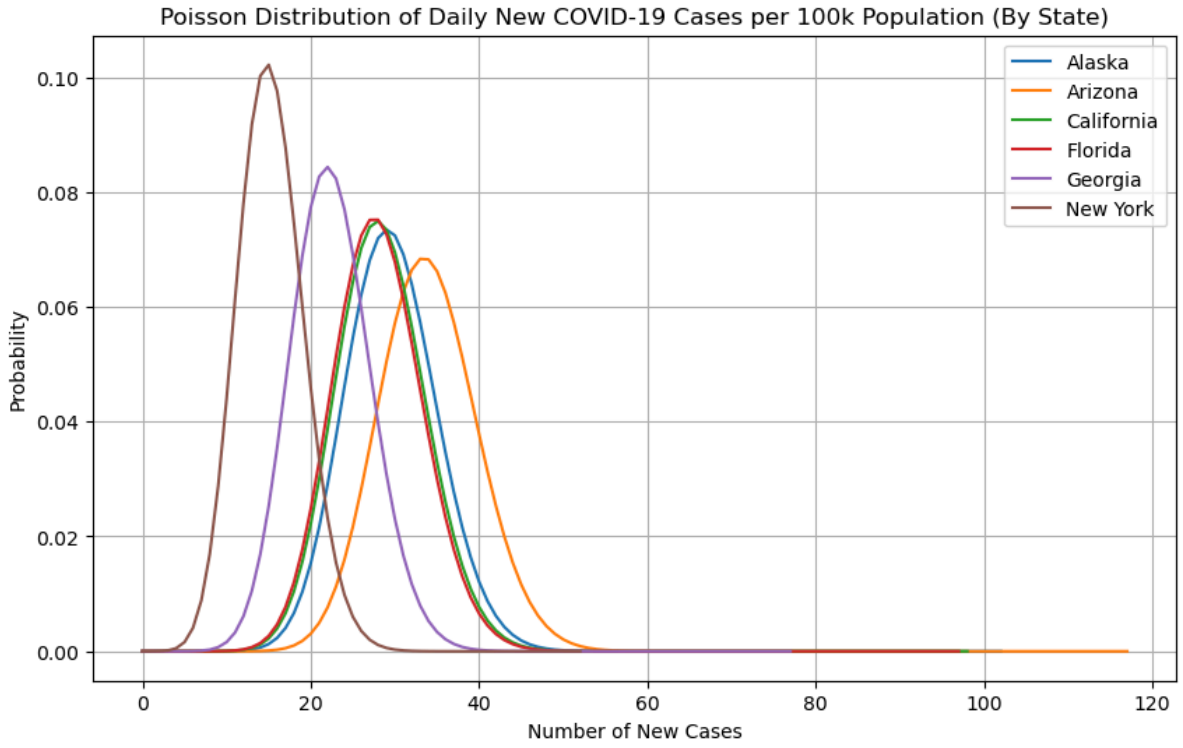



Compare this distribution to other 5 states.

In [128…

```
statefips_list = [2, 4, 6, 12, 13, 36]  # State FIPS codes
state_names = ['Alaska','Arizona', 'California', 'Florida', 'Georgia', 'New York']

plt.figure(figsize=(10, 6))


for statefips, state_name in zip(statefips_list, state_names):
    # Filter the data
    state_data = super_covid19[super_covid19['StateFIPS'] == statefips]
    state_data_filter = [col for col in state_data.columns
                         if ('_cases' in col) and '2020-06-01' <= col.split('_')[0]
    df_2020 = state_data.drop(['County Name', 'State', 'StateFIPS', 'population'],
    df_2020 = df_2020[state_data_filter]

    # Sum up the cases across counties for each state
    df_sum = df_2020.sum(axis=0)

    # Calculate daily new cases
    for i in range(1, len(df_sum)):
        df_sum[i-1] = df_sum[i] - df_sum[i-1]
    df_sum[len(df_sum)-1] = 0

    # Total population of the state
    population = state_data['population'].sum()

    # Calculate lambda (mean daily new cases per 100k population)
    lamda = (df_sum.mean() / population) * 100000

    # Plot the Poisson distribution for this state
    x_values = np.arange(0, int(lamda * 3.5))  # Set the x-axis values (0 to 3.5 *
    pmf = poisson.pmf(x_values, lamda)  # Calculate Poisson PMF
    plt.plot(x_values, pmf, label=state_name)  # Plot the PMF with a label for the
```

```
plt.title('Poisson Distribution of Daily New COVID-19 Cases per 100k Population (By
plt.xlabel('Number of New Cases')
plt.ylabel('Probability')
plt.grid(True)
plt.legend()
plt.show()
```




- New York and Georgia have sharper, narrower distributions, indicating more predictable daily new cases.
- Arizona, California, and Florida have wider, more variable distributions.
- Alaska's distribution is narrow but has a slightly higher peak number of cases compared to New York and Georgia.

# Deaths:-

- Model a Poisson distribution of COVID-19 deaths of a state.

In [129…

```
# show the COVID deaths data for Arizona state
state_data = super_covid19[super_covid19['StateFIPS'] == 4]
#print(state_data)
state_data_filter = [col for col in state_data.columns
                     if ('_deaths' in col)  and '2020-06-01' <= col.split('_')[0] <=
df_2020=state_data.drop(['County Name','State','StateFIPS','population'],axis=1)
df_2020=df_2020[state_data_filter]
df_2020.head(5)
```

Out[129]:

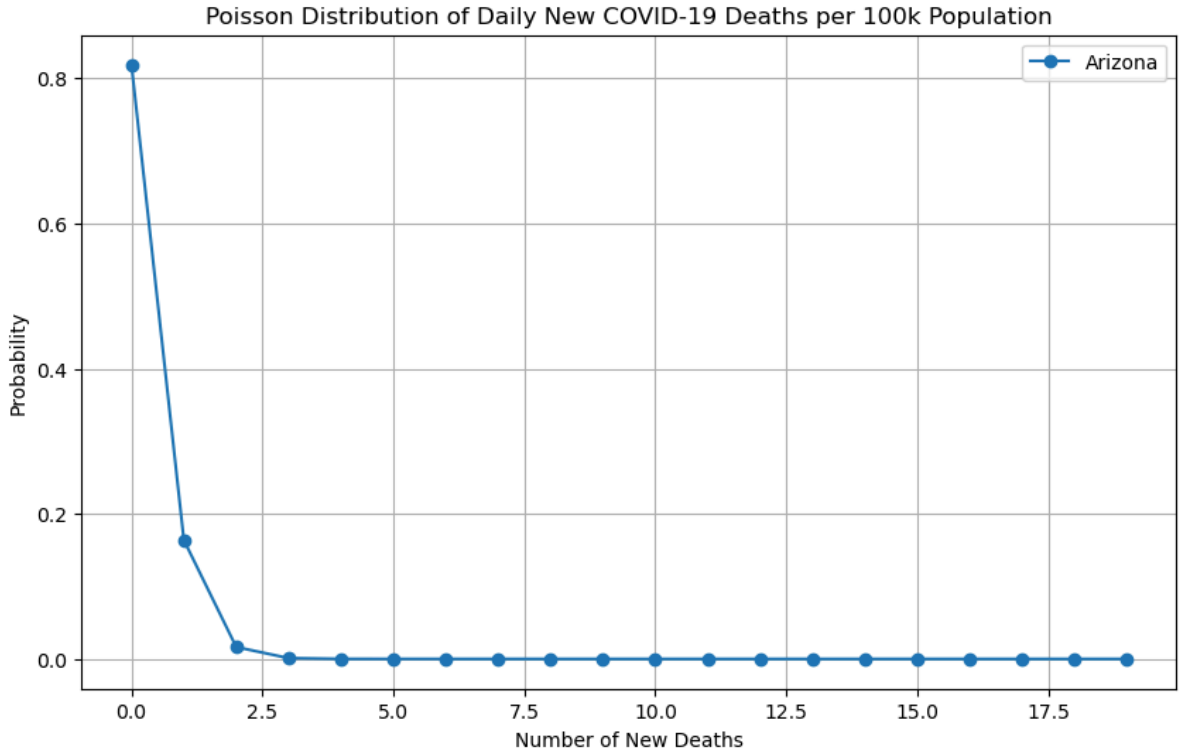
| | 2020-06-01_deaths | 2020-06-02_deaths | 2020-06-03_deaths | 2020-06-04_deaths | 2020-06-05_deaths | 2020-06-06_deaths | 2020-06-07_deaths | 2020-06-08_deaths | 2 09 |
|---|---|---|---|---|---|---|---|---|---|
| **96** | 39 | 41 | 45 | 45 | 46 | 49 | 49 | 49 | |
| **97** | 3 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | |
| **98** | 81 | 81 | 85 | 85 | 85 | 85 | 85 | 85 | |
| **99** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| **100** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

5 rows × 217 columns

In [154…

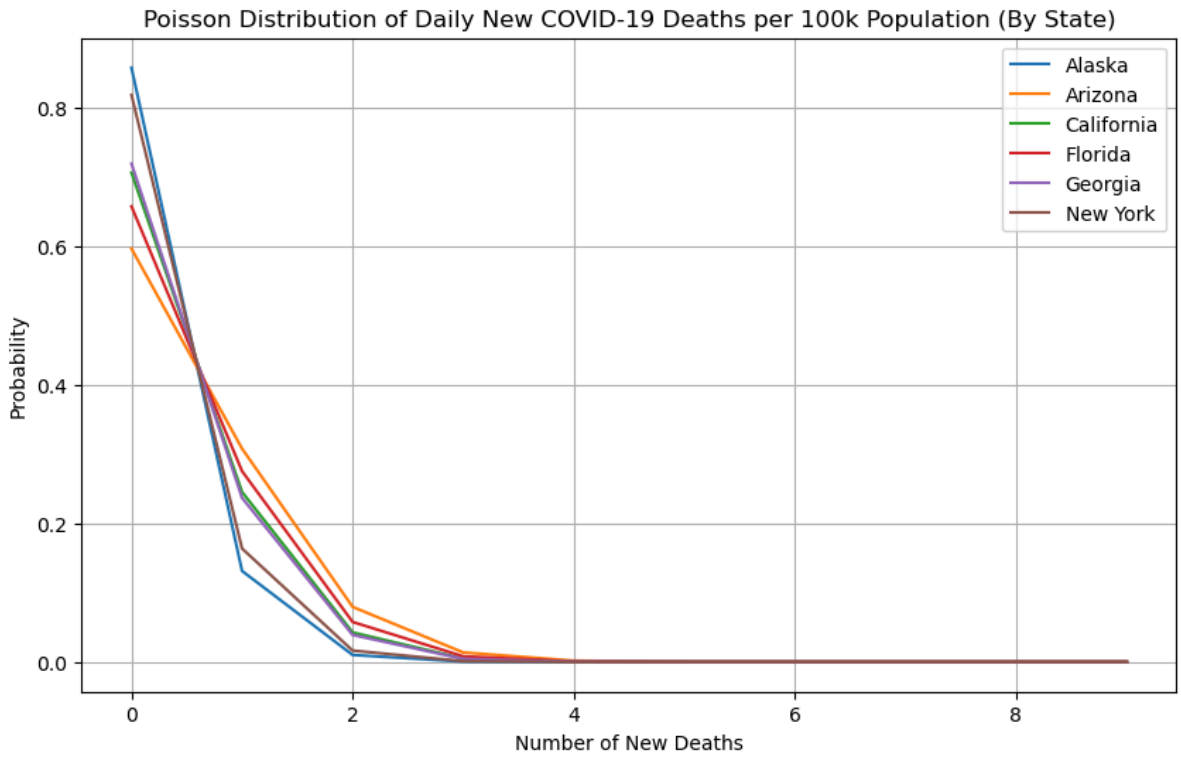
```
df_sum=df_2020.sum(axis=0)
#print(df_sum)

for i in range(1,len(df_sum)):
    df_sum[i-1]=df_sum[i]-df_sum[i-1] # calculate new deaths each day

df_sum[len(df_sum)-1]=0
print(df_sum)

population = state_data['population'].sum()
print(population)
lamda=(df_sum.mean()/population)*100000
print(lamda)

plt.figure(figsize=(10, 6))
x_values = np.arange(0, 20)
#print(x_values)
pmf = poisson.pmf(x_values, lamda)
plt.plot(x_values, pmf,marker='o',label='Arizona')

# Customize the plot
plt.title('Poisson Distribution of Daily New COVID-19 Deaths per 100k Population')
plt.xlabel('Number of New Deaths')
plt.ylabel('Probability')
plt.grid(True)
plt.legend()
plt.show()
```

```
2020-06-01_deaths     58
2020-06-02_deaths     82
2020-06-03_deaths    107
2020-06-04_deaths     79
2020-06-05_deaths     87
                    ...
2020-12-30_deaths    291
2020-12-31_deaths    165
2021-01-01_deaths    214
2021-01-02_deaths    153
2021-01-03_deaths      0
Length: 217, dtype: int64
19453561
0.19995628312739624
```




In [152…

```
statefips_list = [2, 4, 6, 12, 13, 36]  # State FIPS codes
state_names = ['Alaska','Arizona', 'California', 'Florida', 'Georgia', 'New York']

plt.figure(figsize=(10, 6))


for statefips, state_name in zip(statefips_list, state_names):
    # Filter the data
    state_data = super_covid19[super_covid19['StateFIPS'] == statefips]
    state_data_filter = [col for col in state_data.columns
                         if ('_deaths' in col) and '2020-06-01' <= col.split('_')[0
    df_2020 = state_data.drop(['County Name', 'State', 'StateFIPS', 'population'],
    df_2020 = df_2020[state_data_filter]

    # Sum up the Deaths across counties for each state
    df_sum = df_2020.sum(axis=0)

    # Calculate daily new Deaths
    for i in range(1, len(df_sum)):
        df_sum[i-1] = df_sum[i] - df_sum[i-1]
    df_sum[len(df_sum)-1] = 0

    # Total population of the state
    population = state_data['population'].sum()

    # Calculate lambda (mean daily new Deaths per 100k population)
    lamda = (df_sum.mean() / population) * 100000

    # Plot the Poisson distribution for this state
    x_values = np.arange(0, 10)  # Set the x-axis values (0 to 3.5 * mean)
    pmf = poisson.pmf(x_values, lamda)  # Calculate Poisson PMF
    plt.plot(x_values, pmf, label=state_name)  # Plot the PMF with a label for the

plt.title('Poisson Distribution of Daily New COVID-19 Deaths per 100k Population (B
plt.xlabel('Number of New Deaths')
plt.ylabel('Probability')
plt.grid(True)
plt.legend()
plt.show()
```




The Poisson distribution graph illustrates the probability of daily new COVID-19 deaths per 100k population for six states: Alaska, Arizona, California, Florida, Georgia, and New York. Here's a breakdown of the comparison among these states:

- All states show a high probability at zero daily new deaths per 100k population, which reflects days when no new deaths occurred. Alaska has the highest peak, indicating a greater number of days with no new deaths compared to other states.
- As the number of new deaths increases, the probability decreases rapidly for all states. This steep drop shows that higher numbers of new deaths per day are less likely. The decline is slightly less sharp for states like Florida and Arizona, indicating that they had more days with small but non-zero new deaths.
- **Alaska** consistently shows the highest probability at zero, suggesting the fewest number of days with daily new deaths.
- **New York** and **California** have relatively similar curves, with a higher likelihood of small numbers of daily deaths but still following the overall steep drop-off.
- **Arizona** and **Florida** tend to have a wider spread, meaning they experienced a larger variety of days with different death counts compared to the other states.
- **Georgia** has a distribution similar to New York and California but sits somewhat between Alaska and Florida in terms of spread and probabilities.

## Explanation of how the Poisson modeling is different from the first modeling I did.

## 1. Type of Distribution:

- **Poisson Distribution**:
  - **Discrete**: The Poisson distribution models the probability of a number of discrete events occurring in a fixed interval of time or space.
  - Example: Counting the number of daily new COVID-19 deaths per 100k population.

- **Continuous Distribution**:
  - **Continuous**: A right-skewed continuous distribution models data that takes on a range of continuous values, such as waiting times, income levels, or life spans.
  - Example: Modeling the time between COVID-19 deaths or the length of hospital stays.

## 2. Shape and Skewness:

- **Poisson Distribution**:
  - For smaller values of the mean (λ), the Poisson distribution is **right-skewed**, but it becomes more symmetric as λ increases.
  - The distribution shows distinct "steps" as it represents discrete probabilities for each possible count.
- **Continuous Distribution**:
  - Always skewed to the right (tail to the right), meaning that most data points are concentrated on the left with a long tail on the right.

## Conclusion:

- **Poisson Distribution** is best for modeling **discrete events or counts**, like the number of new COVID-19 deaths per day.
- A **right-skewed continuous distribution** is suited for modeling **continuous data** that is heavily concentrated on smaller values but with a few extreme cases, like wait times or income levels.

# 3. Perform correlation between Enrichment data valiables and COVID-19 cases to observe any patterns. (20 points)

Load merge enrichment data

In [32]:
```
merge_data=pd.read_csv('MergeData.csv',low_memory=False)
merge_data.head(5)
```

Out[32]:

| | countyFIPS | County Name | State | StateFIPS | population | 2020-01-22_cases | 2020-01-23_cases | 2020-01-24_cases | 2020-01-25_cases | 2 |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 1001 | Autauga County | AL | 1.0 | 55869.0 | 0.0 | 0.0 | 0.0 | 0.0 | |
| **1** | 1003 | Baldwin County | AL | 1.0 | 223234.0 | 0.0 | 0.0 | 0.0 | 0.0 | |
| **2** | 1005 | Barbour County | AL | 1.0 | 24686.0 | 0.0 | 0.0 | 0.0 | 0.0 | |
| **3** | 1007 | Bibb County | AL | 1.0 | 22394.0 | 0.0 | 0.0 | 0.0 | 0.0 | |
| **4** | 1009 | Blount County | AL | 1.0 | 57826.0 | 0.0 | 0.0 | 0.0 | 0.0 | |

5 rows × 1052 columns

Select state specific data so that within that state I can compare the county based covid data to enrichment data for correlation.

In [4]:
```
state_merge_data=merge_data[merge_data['StateFIPS'] == 4]
col = [col for col in state_merge_data.columns
                  if ('Margin of Error' in col) ]
state_merge_data=state_merge_data.drop(col,axis=1)
state_merge_data.head(5)
```

Out[4]:

| | countyFIPS | County Name | State | StateFIPS | population | 2020-01-22_cases | 2020-01-23_cases | 2020-01-24_cases | 2020-01-25_cases |
|---|---|---|---|---|---|---|---|---|---|
| **96** | 4001 | Apache County | AZ | 4.0 | 71887.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| **97** | 4003 | Cochise County | AZ | 4.0 | 125922.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| **98** | 4005 | Coconino County | AZ | 4.0 | 143476.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| **99** | 4007 | Gila County | AZ | 4.0 | 54018.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| **100** | 4009 | Graham County | AZ | 4.0 | 38837.0 | 0.0 | 0.0 | 0.0 | 0.0 |

5 rows × 874 columns

- Calculate total cases and deaths
- Measure of center value (median or mean) to compare the number of cases and deaths.

In [5]:
```
dates_col = [col for col in state_merge_data.columns
                  if ('_cases' in col or '_deaths' in col) ]

cases_col = [col for col in state_merge_data.columns
                  if ('_cases' in col) ]
deaths_col = [col for col in state_merge_data.columns
                  if ('_deaths' in col) ]

df=state_merge_data.drop(dates_col,axis=1)
df['cases_total'] = state_merge_data.loc[:, cases_col].sum(axis=1)
df['cases_mean'] = state_merge_data.loc[:, cases_col].mean(axis=1)
df['deaths_total'] = state_merge_data.loc[:, deaths_col].sum(axis=1)
df['deaths_mean'] = state_merge_data.loc[:, deaths_col].mean(axis=1)
df.head(5)
```

Out[5]:

| | countyFIPS | County Name | State | StateFIPS | population | Geographic Area Name | Estimate!!SEX AND AGE!!Total population | Estimate!! AND AGE!!T population!!M |
|---|---|---|---|---|---|---|---|---|
| **96** | 4001 | Apache County | AZ | 4.0 | 71887.0 | Apache County, Arizona | 71714.0 | 353 |
| **97** | 4003 | Cochise County | AZ | 4.0 | 125922.0 | Cochise County, Arizona | 126442.0 | 643 |
| **98** | 4005 | Coconino County | AZ | 4.0 | 143476.0 | Coconino County, Arizona | 142254.0 | 701 |
| **99** | 4007 | Gila County | AZ | 4.0 | 54018.0 | Gila County, Arizona | 53846.0 | 266 |
| **100** | 4009 | Graham County | AZ | 4.0 | 38837.0 | Graham County, Arizona | 38304.0 | 204 |

5 rows × 188 columns

Select only necessary columns to perform correlation.

In [6]:
```
df_copy=df.copy()
df_copy=df_copy.drop(['countyFIPS','County Name','State','StateFIPS','Geographic Ar
df_copy
```

Out[6]:

| | population | Estimate!!SEX AND AGE!!Total population | Estimate!!SEX AND AGE!!Total population!!Male | Estimate!!SEX AND AGE!!Total population!!Female | Estimate!!SEX AND AGE!!Total population!!Sex ratio (males per 100 females) | Estima AND AG population |
|---|---|---|---|---|---|---|
| **96** | 71887.0 | 71714.0 | 35388.0 | 36326.0 | 97.4 | |
| **97** | 125922.0 | 126442.0 | 64371.0 | 62071.0 | 103.7 | |
| **98** | 143476.0 | 142254.0 | 70124.0 | 72130.0 | 97.2 | |
| **99** | 54018.0 | 53846.0 | 26671.0 | 27175.0 | 98.1 | |
| **100** | 38837.0 | 38304.0 | 20458.0 | 17846.0 | 114.6 | |
| **101** | 9498.0 | 9465.0 | 5062.0 | 4403.0 | 115.0 | |
| **102** | 21108.0 | 21035.0 | 10750.0 | 10285.0 | 104.5 | |
| **103** | 4485414.0 | 4412779.0 | 2181967.0 | 2230812.0 | 97.8 | 2 |
| **104** | 212181.0 | 210998.0 | 106442.0 | 104556.0 | 101.8 | |
| **105** | 110924.0 | 110271.0 | 55277.0 | 54994.0 | 100.5 | |
| **106** | 1047279.0 | 1038476.0 | 510875.0 | 527601.0 | 96.8 | |
| **107** | 462789.0 | 447559.0 | 232422.0 | 215137.0 | 108.0 | |
| **108** | 46498.0 | 46594.0 | 22436.0 | 24158.0 | 92.9 | |
| **109** | 235099.0 | 232396.0 | 113499.0 | 118897.0 | 95.5 | |
| **110** | 213787.0 | 211931.0 | 109237.0 | 102694.0 | 106.4 | |

15 rows × 183 columns

Find a columns with null values in all rows

In [7]:
```
columns_with_X = df.columns[df.isin(['(X)']).any()]
columns_with_X
```

Out[7]:
```
Index(['Percent!!SEX AND AGE!!Total population!!Sex ratio (males per 100 female
s)',
       'Percent!!SEX AND AGE!!Total population!!Median age (years)',
       'Percent!!SEX AND AGE!!Total population!!18 years and over!!Sex ratio (male
s per 100 females)',
       'Percent!!SEX AND AGE!!Total population!!65 years and over!!Sex ratio (male
s per 100 females)',
       'Percent!!Total housing units'],
      dtype='object')
```

Drop that null columns

In [8]:
```
df_copy = df_copy.drop(columns=columns_with_X)
df_copy
```

Out[8]:

| | population | Estimate!!SEX AND AGE!!Total population | Estimate!!SEX AND AGE!!Total population!!Male | Estimate!!SEX AND AGE!!Total population!!Female | Estimate!!SEX AND AGE!!Total population!!Sex ratio (males per 100 females) | Estima AND AG population |
|---|---|---|---|---|---|---|
| **96** | 71887.0 | 71714.0 | 35388.0 | 36326.0 | 97.4 | |
| **97** | 125922.0 | 126442.0 | 64371.0 | 62071.0 | 103.7 | |
| **98** | 143476.0 | 142254.0 | 70124.0 | 72130.0 | 97.2 | |
| **99** | 54018.0 | 53846.0 | 26671.0 | 27175.0 | 98.1 | |
| **100** | 38837.0 | 38304.0 | 20458.0 | 17846.0 | 114.6 | |
| **101** | 9498.0 | 9465.0 | 5062.0 | 4403.0 | 115.0 | |
| **102** | 21108.0 | 21035.0 | 10750.0 | 10285.0 | 104.5 | |
| **103** | 4485414.0 | 4412779.0 | 2181967.0 | 2230812.0 | 97.8 | 2 |
| **104** | 212181.0 | 210998.0 | 106442.0 | 104556.0 | 101.8 | |
| **105** | 110924.0 | 110271.0 | 55277.0 | 54994.0 | 100.5 | |
| **106** | 1047279.0 | 1038476.0 | 510875.0 | 527601.0 | 96.8 | |
| **107** | 462789.0 | 447559.0 | 232422.0 | 215137.0 | 108.0 | |
| **108** | 46498.0 | 46594.0 | 22436.0 | 24158.0 | 92.9 | |
| **109** | 235099.0 | 232396.0 | 113499.0 | 118897.0 | 95.5 | |
| **110** | 213787.0 | 211931.0 | 109237.0 | 102694.0 | 106.4 | |

15 rows × 178 columns

Find the unique data type values of columns

- Result: `float` and `object` type
- Reason: To find correlation between columns we need that column values in int or float datatype

In [9]:
```
df_copy.dtypes.unique()
```

Out[9]:
```
array([dtype('float64'), dtype('O')], dtype=object)
```

In [10]:
```
# Display the list of columns with dtype 'object'
object_columns = df.select_dtypes(include=['O']).columns
print(object_columns)
```

```
Index(['County Name', 'State', 'Geographic Area Name',
       'Estimate!!SEX AND AGE!!Total population!!65 years and over!!Sex ratio (mal
es per 100 females)',
       'Percent!!SEX AND AGE!!Total population!!Sex ratio (males per 100 female
s)',
       'Percent!!SEX AND AGE!!Total population!!Median age (years)',
       'Percent!!SEX AND AGE!!Total population!!18 years and over!!Sex ratio (male
s per 100 females)',
       'Percent!!SEX AND AGE!!Total population!!65 years and over!!Sex ratio (male
s per 100 females)',
       'Percent!!Total housing units'],
      dtype='object')
```

Convert that columns datatype into float datatype

In [13]:
```
df_copy = df_copy.astype(float)
df_copy.dtypes.unique()
```

Out[13]:
```
array([dtype('float64')], dtype=object)
```

Now normalized the data of each county for population.

In [14]:
```
# Normalization
# Apply the division to all rows in one operation
df_copy = (df_copy.div(df_copy['population'], axis=0)) * 100000
df_copy
```

Out[14]:

| | population | Estimate!!SEX AND AGE!!Total population | Estimate!!SEX AND AGE!!Total population!!Male | Estimate!!SEX AND AGE!!Total population!!Female | Estimate!!SEX AND AGE!!Total population!!Sex ratio (males per 100 females) | Estim AND AG populatior |
|---|---|---|---|---|---|---|
| **96** | 100000.0 | 99759.344527 | 49227.259449 | 50532.085078 | 135.490422 | 657( |
| **97** | 100000.0 | 100412.954051 | 51119.740792 | 49293.213259 | 82.352567 | 580 |
| **98** | 100000.0 | 99148.289609 | 48875.073183 | 50273.216426 | 67.746522 | 531 |
| **99** | 100000.0 | 99681.587619 | 49374.282647 | 50307.304972 | 181.606131 | 532 |
| **100** | 100000.0 | 98627.597394 | 52676.571311 | 45951.026083 | 295.079435 | 711 |
| **101** | 100000.0 | 99652.558433 | 53295.430617 | 46357.127816 | 1210.781217 | 772 |
| **102** | 100000.0 | 99654.159560 | 50928.557893 | 48725.601668 | 495.072958 | 445 |
| **103** | 100000.0 | 98380.640003 | 48645.832915 | 49734.807088 | 2.180401 | 618 |
| **104** | 100000.0 | 99442.457147 | 50165.660450 | 49276.796697 | 47.977906 | 434 |
| **105** | 100000.0 | 99411.308644 | 49833.219141 | 49578.089503 | 90.602575 | 668 |
| **106** | 100000.0 | 99159.440798 | 48781.174835 | 50378.265964 | 9.243000 | 550 |
| **107** | 100000.0 | 96709.083405 | 50222.023428 | 46487.059978 | 23.336769 | 544 |
| **108** | 100000.0 | 100206.460493 | 48251.537701 | 51954.922792 | 199.793540 | 672 |
| **109** | 100000.0 | 98850.271588 | 48277.108792 | 50573.162795 | 40.621185 | 399 |
| **110** | 100000.0 | 99131.846183 | 51096.184520 | 48035.661663 | 49.769163 | 700 |

15 rows × 178 columns

Correlation Matrix

In [21]:
```
correlation_matrix = df_copy.corr()
print(correlation_matrix)
```

```
                                                                population  \
population                                                             NaN
Estimate!!SEX AND AGE!!Total population                                NaN
Estimate!!SEX AND AGE!!Total population!!Male                          NaN
Estimate!!SEX AND AGE!!Total population!!Female                        NaN
Estimate!!SEX AND AGE!!Total population!!Sex ra...                     NaN
...                                                                    ...
Percent!!CITIZEN, VOTING AGE POPULATION!!Citize...                     NaN
cases_total                                                            NaN
cases_mean                                                             NaN
deaths_total                                                           NaN
deaths_mean                                                            NaN

                                                                Estimate!!SEX AND AGE!!Total p
opulation  \
population
NaN
Estimate!!SEX AND AGE!!Total population
1.000000
Estimate!!SEX AND AGE!!Total population!!Male
0.022752
Estimate!!SEX AND AGE!!Total population!!Female
0.482862
Estimate!!SEX AND AGE!!Total population!!Sex ra...
0.254575
...
...
Percent!!CITIZEN, VOTING AGE POPULATION!!Citize...
0.290879
cases_total
0.211395
cases_mean
0.211395
deaths_total
0.295284
deaths_mean
0.295284

                                                                Estimate!!SEX AND AGE!!Total p
opulation!!Male  \
population
NaN
Estimate!!SEX AND AGE!!Total population
0.022752
Estimate!!SEX AND AGE!!Total population!!Male
1.000000
Estimate!!SEX AND AGE!!Total population!!Female
-0.864484
Estimate!!SEX AND AGE!!Total population!!Sex ra...
0.676753
...
...
Percent!!CITIZEN, VOTING AGE POPULATION!!Citize...
0.645588
cases_total
-0.295363
cases_mean
-0.295363
deaths_total
-0.293287
deaths_mean
-0.293287

                                                                Estimate!!SEX AND AGE!!Total p
```

```
opulation!!Female  \
population
NaN
Estimate!!SEX AND AGE!!Total population
0.482862
Estimate!!SEX AND AGE!!Total population!!Male
-0.864484
Estimate!!SEX AND AGE!!Total population!!Female
1.000000
Estimate!!SEX AND AGE!!Total population!!Sex ra...
-0.464786
...
...
Percent!!CITIZEN, VOTING AGE POPULATION!!Citize...
-0.419235
cases_total
0.365003
cases_mean
0.365003
deaths_total
0.405363
deaths_mean
0.405363

                                                    Estimate!!SEX AND AGE!!Total p
opulation!!Sex ratio (males per 100 females)  \
population
NaN
Estimate!!SEX AND AGE!!Total population
0.254575
Estimate!!SEX AND AGE!!Total population!!Male
0.676753
Estimate!!SEX AND AGE!!Total population!!Female
-0.464786
Estimate!!SEX AND AGE!!Total population!!Sex ra...
1.000000
...
...
Percent!!CITIZEN, VOTING AGE POPULATION!!Citize...
0.997142
cases_total
-0.288390
cases_mean
-0.288390
deaths_total
-0.308418
deaths_mean
-0.308418

                                                    Estimate!!SEX AND AGE!!Total p
opulation!!Under 5 years  \
population
NaN
Estimate!!SEX AND AGE!!Total population
0.120002
Estimate!!SEX AND AGE!!Total population!!Male
0.463971
Estimate!!SEX AND AGE!!Total population!!Female
-0.346067
Estimate!!SEX AND AGE!!Total population!!Sex ra...
0.413570
...
...
Percent!!CITIZEN, VOTING AGE POPULATION!!Citize...
```

```
0.393426
cases_total
0.435499
cases_mean
0.435499
deaths_total
0.254372
deaths_mean
0.254372

                                                    Estimate!!SEX AND AGE!!Total p
opulation!!5 to 9 years  \
population
NaN
Estimate!!SEX AND AGE!!Total population
0.181750
Estimate!!SEX AND AGE!!Total population!!Male
0.199811
Estimate!!SEX AND AGE!!Total population!!Female
-0.083637
Estimate!!SEX AND AGE!!Total population!!Sex ra...
0.256220
...
...
Percent!!CITIZEN, VOTING AGE POPULATION!!Citize...
0.255776
cases_total
0.568369
cases_mean
0.568369
deaths_total
0.378042
deaths_mean
0.378042

                                                    Estimate!!SEX AND AGE!!Total p
opulation!!10 to 14 years  \
population
NaN
Estimate!!SEX AND AGE!!Total population
-0.027316
Estimate!!SEX AND AGE!!Total population!!Male
0.274546
Estimate!!SEX AND AGE!!Total population!!Female
-0.254216
Estimate!!SEX AND AGE!!Total population!!Sex ra...
0.230265
...
...
Percent!!CITIZEN, VOTING AGE POPULATION!!Citize...
0.210830
cases_total
0.522020
cases_mean
0.522020
deaths_total
0.404692
deaths_mean
0.404692

                                                    Estimate!!SEX AND AGE!!Total p
opulation!!15 to 19 years  \
population
NaN
```

```
Estimate!!SEX AND AGE!!Total population
0.041541
Estimate!!SEX AND AGE!!Total population!!Male
-0.030834
Estimate!!SEX AND AGE!!Total population!!Female
0.047895
Estimate!!SEX AND AGE!!Total population!!Sex ra...
0.043047
...
...
Percent!!CITIZEN, VOTING AGE POPULATION!!Citize...
0.029067
cases_total
0.312292
cases_mean
0.312292
deaths_total
0.265935
deaths_mean
0.265935

                                                    Estimate!!SEX AND AGE!!Total p
opulation!!20 to 24 years  \
population
NaN
Estimate!!SEX AND AGE!!Total population
-0.001291
Estimate!!SEX AND AGE!!Total population!!Male
0.067463
Estimate!!SEX AND AGE!!Total population!!Female
-0.059742
Estimate!!SEX AND AGE!!Total population!!Sex ra...
0.110446
...
...
Percent!!CITIZEN, VOTING AGE POPULATION!!Citize...
0.082153
cases_total
0.112875
cases_mean
0.112875
deaths_total
0.023846
deaths_mean
0.023846

                                                    ...  \
population                                          ...
Estimate!!SEX AND AGE!!Total population             ...
Estimate!!SEX AND AGE!!Total population!!Male       ...
Estimate!!SEX AND AGE!!Total population!!Female     ...
Estimate!!SEX AND AGE!!Total population!!Sex ra...  ...
...                                                 ...
Percent!!CITIZEN, VOTING AGE POPULATION!!Citize...  ...
cases_total                                         ...
cases_mean                                          ...
deaths_total                                        ...
deaths_mean                                         ...

                                                    Percent!!HISPANIC OR LATINO AN
D RACE!!Total population!!Not Hispanic or Latino!!Two or more races  \
population
NaN
Estimate!!SEX AND AGE!!Total population
```

```
0.367970
Estimate!!SEX AND AGE!!Total population!!Male
0.602238
Estimate!!SEX AND AGE!!Total population!!Female
-0.342503
Estimate!!SEX AND AGE!!Total population!!Sex ra...
0.877439
...
...
Percent!!CITIZEN, VOTING AGE POPULATION!!Citize...
0.896353
cases_total
-0.280269
cases_mean
-0.280269
deaths_total
-0.210446
deaths_mean
-0.210446
```

```
                                                    Percent!!HISPANIC OR LATINO AN
D RACE!!Total population!!Not Hispanic or Latino!!Two or more races!!Two races inc
luding Some other race  \
population
NaN
Estimate!!SEX AND AGE!!Total population
0.410201
Estimate!!SEX AND AGE!!Total population!!Male
0.038254
Estimate!!SEX AND AGE!!Total population!!Female
0.172738
Estimate!!SEX AND AGE!!Total population!!Sex ra...
0.055657
...
...
Percent!!CITIZEN, VOTING AGE POPULATION!!Citize...
0.099370
cases_total
-0.228188
cases_mean
-0.228188
deaths_total
-0.047799
deaths_mean
-0.047799
```

```
                                                    Percent!!HISPANIC OR LATINO AN
D RACE!!Total population!!Not Hispanic or Latino!!Two or more races!!Two races exc
luding Some other race, and Three or more races  \
population
NaN
Estimate!!SEX AND AGE!!Total population
0.352532
Estimate!!SEX AND AGE!!Total population!!Male
0.615469
Estimate!!SEX AND AGE!!Total population!!Female
-0.361854
Estimate!!SEX AND AGE!!Total population!!Sex ra...
0.898693
...
...
Percent!!CITIZEN, VOTING AGE POPULATION!!Citize...
0.915450
cases_total
```

```
-0.275817
cases_mean
-0.275817
deaths_total
-0.214330
deaths_mean
-0.214330

                                                    Percent!!CITIZEN, VOTING AGE P
OPULATION!!Citizen, 18 and over population  \
population
NaN
Estimate!!SEX AND AGE!!Total population
0.070884
Estimate!!SEX AND AGE!!Total population!!Male
-0.074849
Estimate!!SEX AND AGE!!Total population!!Female
0.101202
Estimate!!SEX AND AGE!!Total population!!Sex ra...
-0.028075
...
...
Percent!!CITIZEN, VOTING AGE POPULATION!!Citize...
-0.024880
cases_total
-0.650993
cases_mean
-0.650993
deaths_total
-0.214958
deaths_mean
-0.214958

                                                    Percent!!CITIZEN, VOTING AGE P
OPULATION!!Citizen, 18 and over population!!Male  \
population
NaN
Estimate!!SEX AND AGE!!Total population
0.269547
Estimate!!SEX AND AGE!!Total population!!Male
0.668608
Estimate!!SEX AND AGE!!Total population!!Female
-0.450124
Estimate!!SEX AND AGE!!Total population!!Sex ra...
0.999407
...
...
Percent!!CITIZEN, VOTING AGE POPULATION!!Citize...
0.998993
cases_total
-0.273476
cases_mean
-0.273476
deaths_total
-0.298168
deaths_mean
-0.298168

                                                    Percent!!CITIZEN, VOTING AGE P
OPULATION!!Citizen, 18 and over population!!Female  \
population
NaN
Estimate!!SEX AND AGE!!Total population
0.290879
```

```
Estimate!!SEX AND AGE!!Total population!!Male
0.645588
Estimate!!SEX AND AGE!!Total population!!Female
-0.419235
Estimate!!SEX AND AGE!!Total population!!Sex ra...
0.997142
...
...
Percent!!CITIZEN, VOTING AGE POPULATION!!Citize...
1.000000
cases_total
-0.255953
cases_mean
-0.255953
deaths_total
-0.280298
deaths_mean
-0.280298
```

| | cases_total | cases_mean | \ |
|---|---|---|---|
| population | NaN | NaN | |
| Estimate!!SEX AND AGE!!Total population | 0.211395 | 0.211395 | |
| Estimate!!SEX AND AGE!!Total population!!Male | -0.295363 | -0.295363 | |
| Estimate!!SEX AND AGE!!Total population!!Female | 0.365003 | 0.365003 | |
| Estimate!!SEX AND AGE!!Total population!!Sex ra... | -0.288390 | -0.288390 | |
| ... | ... | ... | |
| Percent!!CITIZEN, VOTING AGE POPULATION!!Citize... | -0.255953 | -0.255953 | |
| cases_total | 1.000000 | 1.000000 | |
| cases_mean | 1.000000 | 1.000000 | |
| deaths_total | 0.836768 | 0.836768 | |
| deaths_mean | 0.836768 | 0.836768 | |

| | deaths_total | deaths_mean |
|---|---|---|
| population | NaN | NaN |
| Estimate!!SEX AND AGE!!Total population | 0.295284 | 0.295284 |
| Estimate!!SEX AND AGE!!Total population!!Male | -0.293287 | -0.293287 |
| Estimate!!SEX AND AGE!!Total population!!Female | 0.405363 | 0.405363 |
| Estimate!!SEX AND AGE!!Total population!!Sex ra... | -0.308418 | -0.308418 |
| ... | ... | ... |
| Percent!!CITIZEN, VOTING AGE POPULATION!!Citize... | -0.280298 | -0.280298 |
| cases_total | 0.836768 | 0.836768 |
| cases_mean | 0.836768 | 0.836768 |
| deaths_total | 1.000000 | 1.000000 |
| deaths_mean | 1.000000 | 1.000000 |

```
[178 rows x 178 columns]
```

Display the correlation for enrichment data and total cases and total deaths

In [33]:
```
# Access the columns 'cases_total' and 'deaths_total' as a DataFrame
#result = correlation_matrix[['cases_total', 'deaths_total','cases_mean','deaths_me
result = correlation_matrix[['cases_total', 'deaths_total']]

# Sort by the values of 'cases_total' and 'deaths_total'
#result = result.sort_values(by=['cases_total', 'deaths_total','cases_mean','deaths
result = result.sort_values(by=['cases_total', 'deaths_total'], ascending=False)

print(result)

# Save the result to a CSV file
result.to_csv('correlation_matrix.csv')
```

| | cases_total | deaths_total |
|---|---|---|
| cases_total | 1.000000 | 0.836768 |
| cases_mean | 1.000000 | 0.836768 |
| deaths_mean | 0.836768 | 1.000000 |
| deaths_total | 0.836768 | 1.000000 |
| Estimate!!RACE!!Total population!!Two or more r... | 0.663001 | 0.751393 |
| ... | ... | ... |
| Percent!!CITIZEN, VOTING AGE POPULATION!!Citize... | -0.650993 | -0.214958 |
| Estimate!!SEX AND AGE!!Total population!!18 yea... | -0.656103 | -0.492122 |
| Estimate!!CITIZEN, VOTING AGE POPULATION!!Citiz... | -0.736402 | -0.345004 |
| Estimate!!HISPANIC OR LATINO AND RACE!!Total po... | -0.813538 | -0.593272 |
| population | NaN | NaN |

[178 rows x 2 columns]

Select the useful rows of coorelation matrix which help me to formulate hypothesis.

In [34]:
```
result.iloc[0:20]
```

Out[34]:

| | cases_total | deaths_total |
|---|---|---|
| **cases_total** | 1.000000 | 0.836768 |
| **cases_mean** | 1.000000 | 0.836768 |
| **deaths_mean** | 0.836768 | 1.000000 |
| **deaths_total** | 0.836768 | 1.000000 |
| **Estimate!!RACE!!Total population!!Two or more races!!Black or African American and American Indian and Alaska Native** | 0.663001 | 0.751393 |
| **Estimate!!SEX AND AGE!!Total population!!5 to 9 years** | 0.568369 | 0.378042 |
| **Estimate!!SEX AND AGE!!Total population!!Under 18 years** | 0.548336 | 0.386383 |
| **Estimate!!SEX AND AGE!!Total population!!10 to 14 years** | 0.522020 | 0.404692 |
| **Percent!!RACE!!Total population!!Two or more races!!Black or African American and American Indian and Alaska Native** | 0.511534 | 0.620658 |
| **Estimate!!RACE!!Total population!!One race!!Some other race** | 0.500443 | 0.100857 |
| **Estimate!!Race alone or in combination with one or more other races!!Total population!!Some other race** | 0.481616 | 0.009574 |
| **Percent!!HISPANIC OR LATINO AND RACE!!Total population!!Not Hispanic or Latino!!Some other race alone** | 0.464210 | 0.167502 |
| **Estimate!!RACE!!Total population!!One race!!American Indian and Alaska Native** | 0.458928 | 0.787929 |
| **Estimate!!HISPANIC OR LATINO AND RACE!!Total population!!Not Hispanic or Latino!!American Indian and Alaska Native alone** | 0.457466 | 0.788589 |
| **Estimate!!Race alone or in combination with one or more other races!!Total population!!American Indian and Alaska Native** | 0.456579 | 0.788379 |
| **Estimate!!RACE!!Total population!!One race!!American Indian and Alaska Native!!Navajo tribal grouping** | 0.445483 | 0.748762 |
| **Estimate!!SEX AND AGE!!Total population!!Under 5 years** | 0.435499 | 0.254372 |
| **Estimate!!SEX AND AGE!!Total population!!25 to 34 years** | 0.369831 | 0.250296 |
| **Estimate!!RACE!!Total population!!One race!!American Indian and Alaska Native!!Sioux tribal grouping** | 0.368379 | 0.644891 |
| **Estimate!!SEX AND AGE!!Total population!!Female** | 0.365003 | 0.405363 |

## [A] Formulate hypothesis between Enrichment data and number of cases to be compared against states. Choose 3 different variables to compare against. (30 points)

Based on the correlation data, we can formulate hypotheses around the relationship between different enrichment variables and the number of COVID-19 cases and deaths, all normalized for population.

## Hypotheses:

1. **Age Groups and COVID-19 Outcomes**:

- **Hypothesis**: Counties with a larger proportion of children (5-9 years old and under 18 years) will have fewer COVID-19 deaths per 100k population.
- **Reasoning**: Younger populations are less likely to experience severe symptoms, which may contribute to lower mortality rates.

2. **Under 5 Years Population and COVID-19 Outcomes**:

- **Hypothesis**: Counties with a higher percentage of the population under 5 years old will have fewer cases and deaths per 100k population.
- **Reasoning**: Infants and young children are generally at lower risk of severe COVID-19 outcomes compared to older adults, though they may contribute to transmission.

3. **Female Population and COVID-19 Outcomes**:

- **Hypothesis**: Counties with a higher proportion of females will have similar or slightly lower COVID-19 death rates compared to males.
- **Reasoning**: Globally, men have shown a slightly higher COVID-19 death rate, though female populations may exhibit different outcomes based on cultural and socioeconomic factors.