A Deep Dive into COVID-19 Death Trends in the USA

SHIBBIR AHMED ARIF

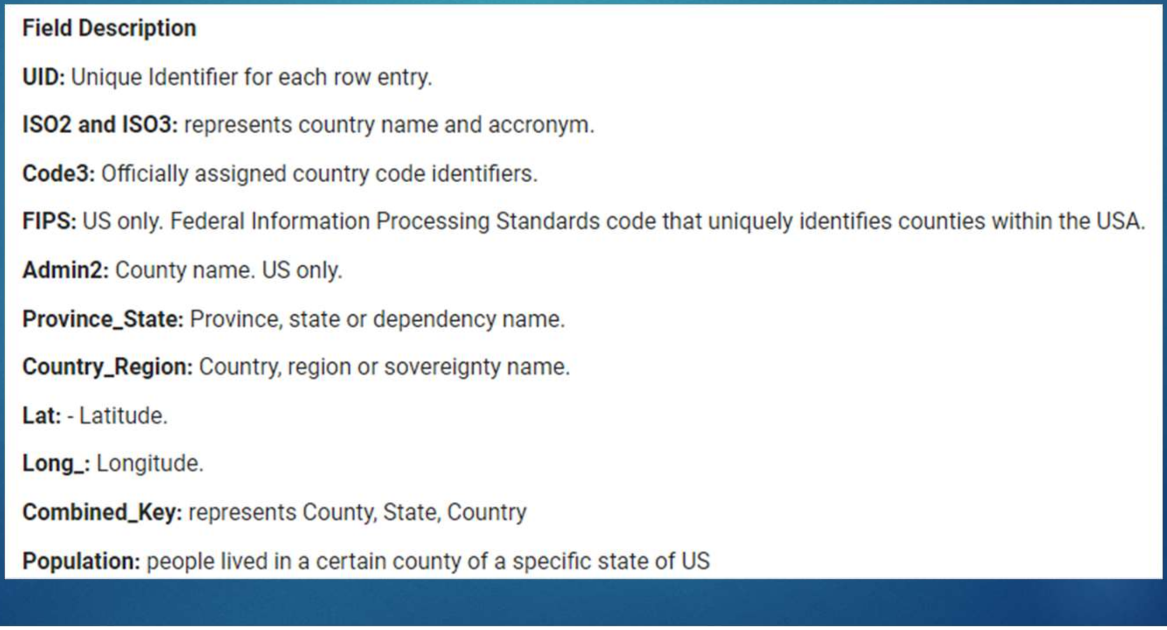
MS – DATA SCIENCE

Project Outline

- Data Collection
- Data Load
- Data Exploration
- Data Cleaning
- Data Analysis
- Map Visualization
- Aggregation Visualization
- Interactive Visualization
- Conclusion

Data Collection

- This is a daily reports based CSSE COVID-19 dataset published by John Hopkins University. This dataset contains daily death cases records between 2020 - 2023.
- Data Source: https://github.com/CSSEGISandData/COVID-19/tree/master/csse_covid_19_data/csse_covid_19_daily_reports_us



Field Description

UID: Unique Identifier for each row entry.

ISO2 and ISO3: represents country name and accronym.

Code3: Officially assigned country code identifiers.

FIPS: US only. Federal Information Processing Standards code that uniquely identifies counties within the USA.

Admin2: County name. US only.

Province_State: Province, state or dependency name.

Country_Region: Country, region or sovereignty name.

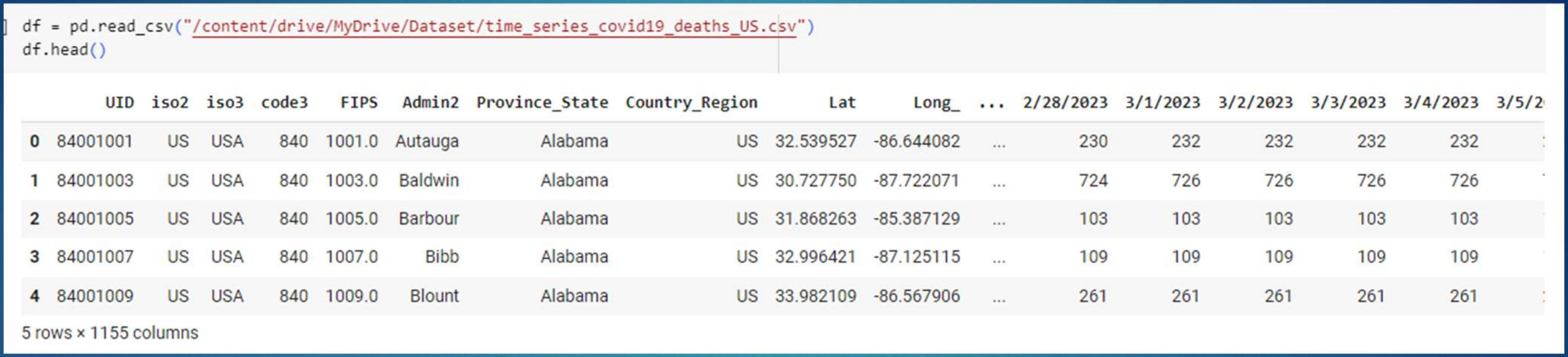
Lat: - Latitude.

Long_: Longitude.

Combined_Key: represents County, State, Country

Population: people lived in a certain county of a specific state of US

Data Load



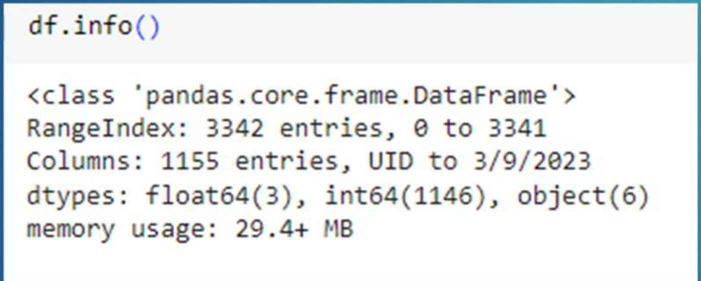
```
df = pd.read_csv("/content/drive/MyDrive/Dataset/time_series_covid19_deaths_US.csv")
df.head()
```

	UID	iso2	iso3	code3	FIPS	Admin2	Province_State	Country_Region	Lat	Long_	...	2/28/2023	3/1/2023	3/2/2023	3/3/2023	3/4/2023	3/5/2
0	84001001	US	USA	840	1001.0	Autauga	Alabama	US	32.539527	-86.644082	...	230	232	232	232	232	
1	84001003	US	USA	840	1003.0	Baldwin	Alabama	US	30.727750	-87.722071	...	724	726	726	726	726	
2	84001005	US	USA	840	1005.0	Barbour	Alabama	US	31.868263	-85.387129	...	103	103	103	103	103	
3	84001007	US	USA	840	1007.0	Bibb	Alabama	US	32.996421	-87.125115	...	109	109	109	109	109	
4	84001009	US	USA	840	1009.0	Blount	Alabama	US	33.982109	-86.567906	...	261	261	261	261	261	

5 rows × 1155 columns

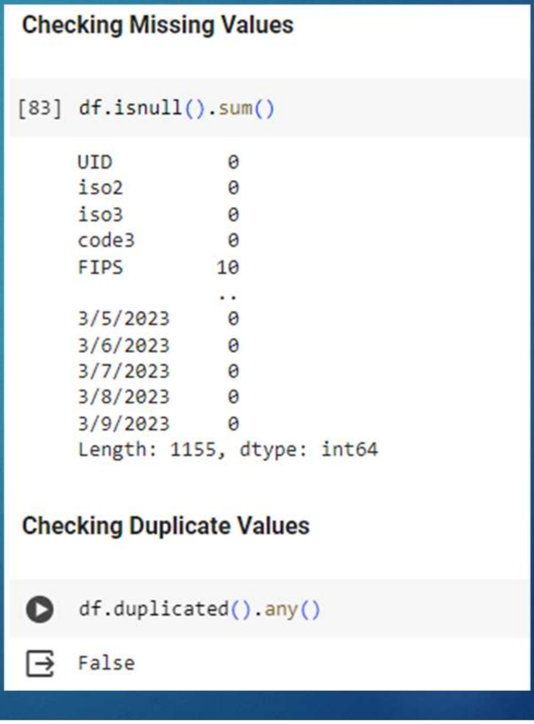
Data Exploration

- Data Shape: 3342 rows and 1155 columns



```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3342 entries, 0 to 3341
Columns: 1155 entries, UID to 3/9/2023
dtypes: float64(3), int64(1146), object(6)
memory usage: 29.4+ MB
```



Checking Missing Values

```
[83] df.isnull().sum()
```

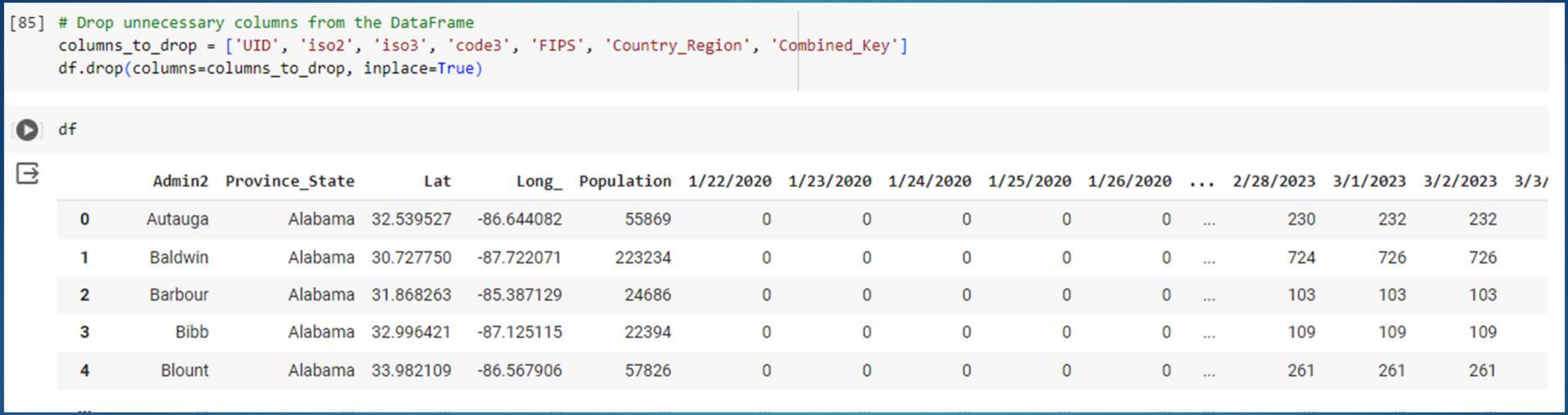
```
UID          0
iso2         0
iso3         0
code3        0
FIPS        10
            ..
3/5/2023     0
3/6/2023     0
3/7/2023     0
3/8/2023     0
3/9/2023     0
Length: 1155, dtype: int64
```

Checking Duplicate Values

```
df.duplicated().any()
```

```
False
```

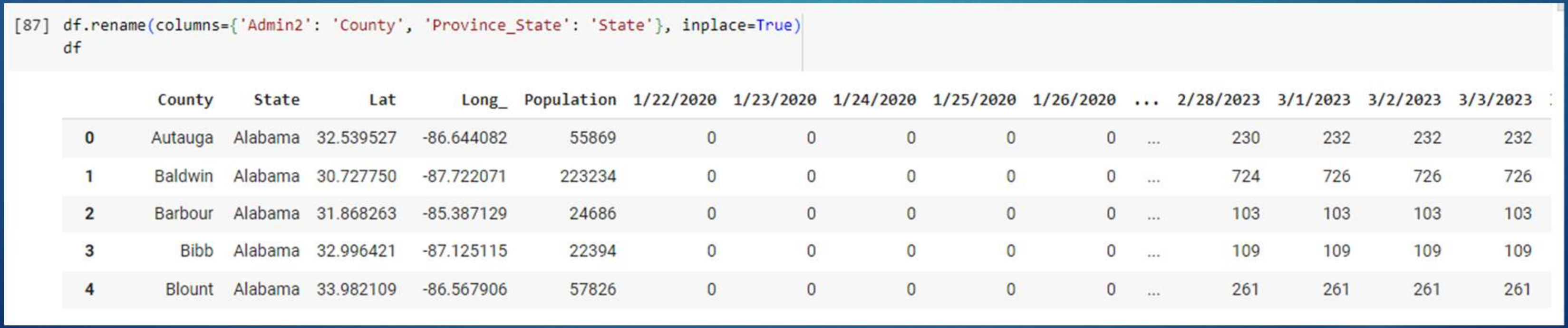
Data Cleaning



```
[85] # Drop unnecessary columns from the DataFrame
     columns_to_drop = ['UID', 'iso2', 'iso3', 'code3', 'FIPS', 'Country_Region', 'Combined_Key']
     df.drop(columns=columns_to_drop, inplace=True)
```

```
df
```

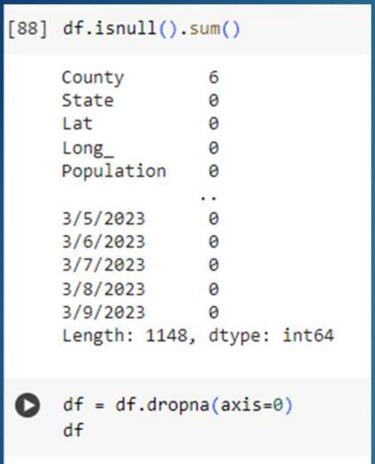
	Admin2	Province_State	Lat	Long_	Population	1/22/2020	1/23/2020	1/24/2020	1/25/2020	1/26/2020	...	2/28/2023	3/1/2023	3/2/2023	3/3/
0	Autauga	Alabama	32.539527	-86.644082	55869	0	0	0	0	0	...	230	232	232	
1	Baldwin	Alabama	30.727750	-87.722071	223234	0	0	0	0	0	...	724	726	726	
2	Barbour	Alabama	31.868263	-85.387129	24686	0	0	0	0	0	...	103	103	103	
3	Bibb	Alabama	32.996421	-87.125115	22394	0	0	0	0	0	...	109	109	109	
4	Blount	Alabama	33.982109	-86.567906	57826	0	0	0	0	0	...	261	261	261	



```
[87] df.rename(columns={'Admin2': 'County', 'Province_State': 'State'}, inplace=True)
     df
```

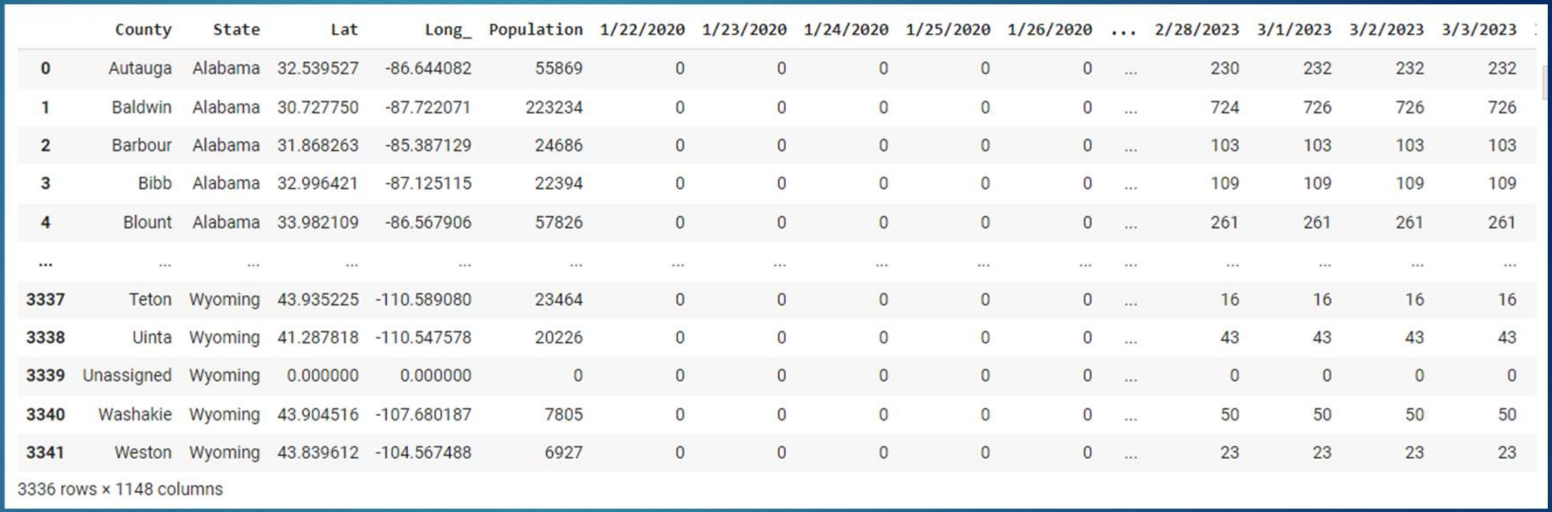
	County	State	Lat	Long_	Population	1/22/2020	1/23/2020	1/24/2020	1/25/2020	1/26/2020	...	2/28/2023	3/1/2023	3/2/2023	3/3/2023
0	Autauga	Alabama	32.539527	-86.644082	55869	0	0	0	0	0	...	230	232	232	232
1	Baldwin	Alabama	30.727750	-87.722071	223234	0	0	0	0	0	...	724	726	726	726
2	Barbour	Alabama	31.868263	-85.387129	24686	0	0	0	0	0	...	103	103	103	103
3	Bibb	Alabama	32.996421	-87.125115	22394	0	0	0	0	0	...	109	109	109	109
4	Blount	Alabama	33.982109	-86.567906	57826	0	0	0	0	0	...	261	261	261	261

Data Cleaning – Cont.



```
[88] df.isnull().sum()

     County        6
     State         0
     Lat           0
     Long_         0
     Population    0
                  ..
     3/5/2023      0
     3/6/2023      0
     3/7/2023      0
     3/8/2023      0
     3/9/2023      0
     Length: 1148, dtype: int64
```

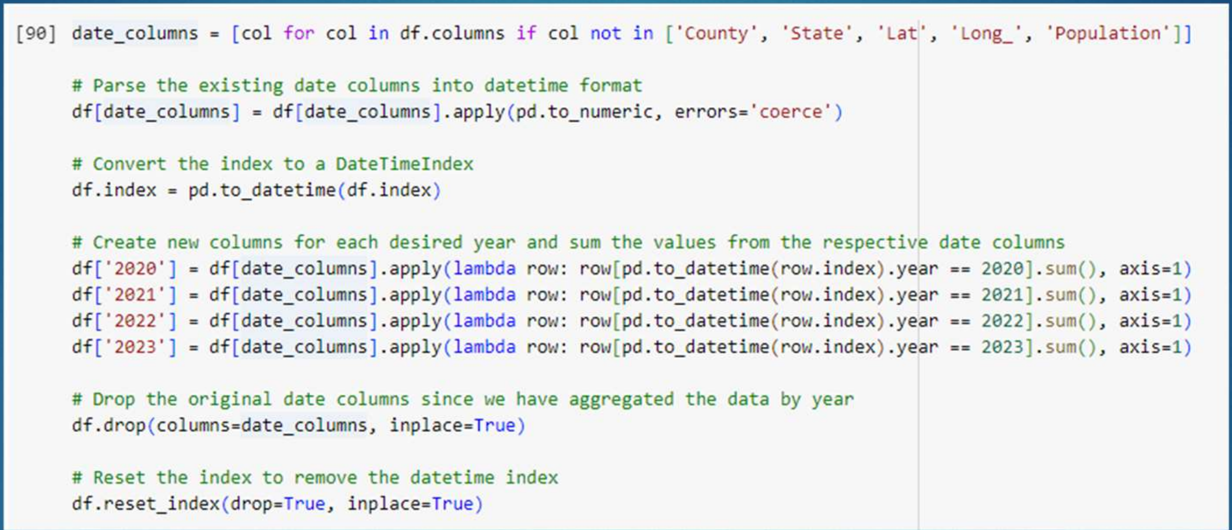
```
df = df.dropna(axis=0)
df
```



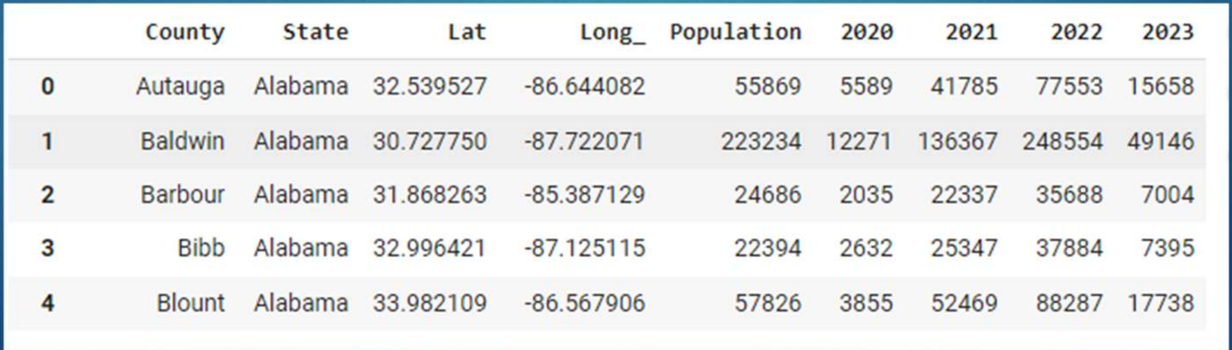
	County	State	Lat	Long_	Population	1/22/2020	1/23/2020	1/24/2020	1/25/2020	1/26/2020	...	2/28/2023	3/1/2023	3/2/2023	3/3/2023
0	Autauga	Alabama	32.539527	-86.644082	55869	0	0	0	0	0	...	230	232	232	232
1	Baldwin	Alabama	30.727750	-87.722071	223234	0	0	0	0	0	...	724	726	726	726
2	Barbour	Alabama	31.868263	-85.387129	24686	0	0	0	0	0	...	103	103	103	103
3	Bibb	Alabama	32.996421	-87.125115	22394	0	0	0	0	0	...	109	109	109	109
4	Blount	Alabama	33.982109	-86.567906	57826	0	0	0	0	0	...	261	261	261	261
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
3337	Teton	Wyoming	43.935225	-110.589080	23464	0	0	0	0	0	...	16	16	16	16
3338	Uinta	Wyoming	41.287818	-110.547578	20226	0	0	0	0	0	...	43	43	43	43
3339	Unassigned	Wyoming	0.000000	0.000000	0	0	0	0	0	0	...	0	0	0	0
3340	Washakie	Wyoming	43.904516	-107.680187	7805	0	0	0	0	0	...	50	50	50	50
3341	Weston	Wyoming	43.839612	-104.567488	6927	0	0	0	0	0	...	23	23	23	23

3336 rows × 1148 columns

Data Wrangling

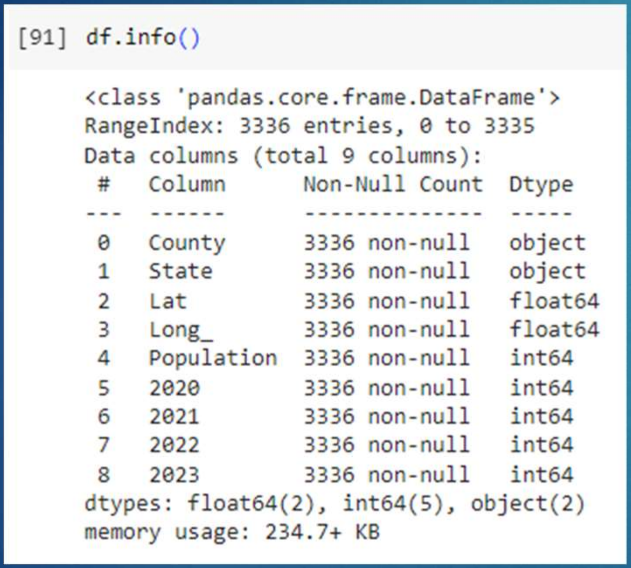


```
[90] date_columns = [col for col in df.columns if col not in ['County', 'State', 'Lat', 'Long_', 'Population']]

     # Parse the existing date columns into datetime format
     df[date_columns] = df[date_columns].apply(pd.to_numeric, errors='coerce')

     # Convert the index to a DateTimeIndex
     df.index = pd.to_datetime(df.index)

     # Create new columns for each desired year and sum the values from the respective date columns
     df['2020'] = df[date_columns].apply(lambda row: row[pd.to_datetime(row.index).year == 2020].sum(), axis=1)
     df['2021'] = df[date_columns].apply(lambda row: row[pd.to_datetime(row.index).year == 2021].sum(), axis=1)
     df['2022'] = df[date_columns].apply(lambda row: row[pd.to_datetime(row.index).year == 2022].sum(), axis=1)
     df['2023'] = df[date_columns].apply(lambda row: row[pd.to_datetime(row.index).year == 2023].sum(), axis=1)

     # Drop the original date columns since we have aggregated the data by year
     df.drop(columns=date_columns, inplace=True)

     # Reset the index to remove the datetime index
     df.reset_index(drop=True, inplace=True)
```

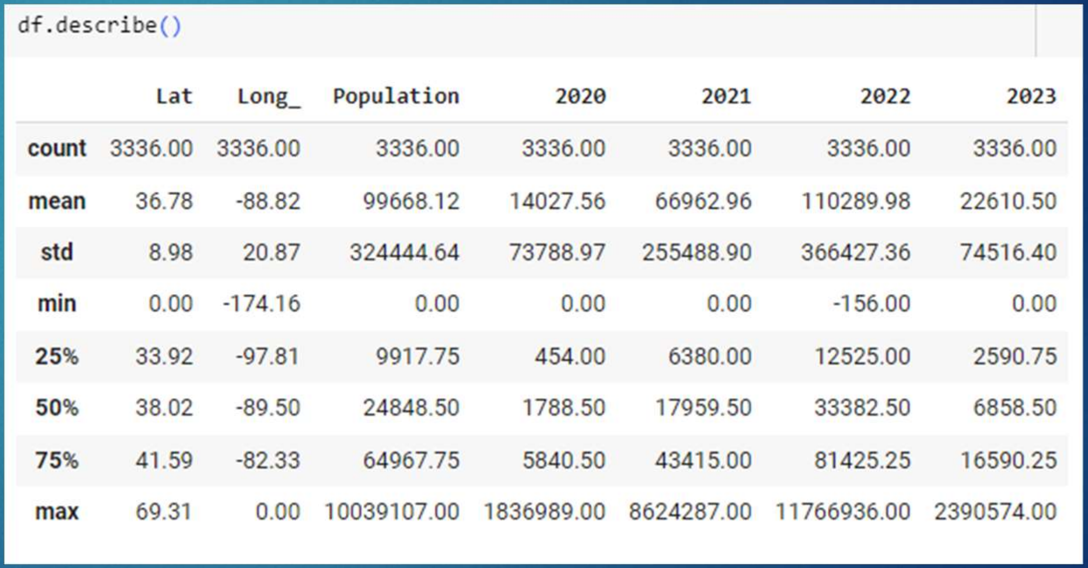


	County	State	Lat	Long_	Population	2020	2021	2022	2023
0	Autauga	Alabama	32.539527	-86.644082	55869	5589	41785	77553	15658
1	Baldwin	Alabama	30.727750	-87.722071	223234	12271	136367	248554	49146
2	Barbour	Alabama	31.868263	-85.387129	24686	2035	22337	35688	7004
3	Bibb	Alabama	32.996421	-87.125115	22394	2632	25347	37884	7395
4	Blount	Alabama	33.982109	-86.567906	57826	3855	52469	88287	17738

Descriptive Statistics



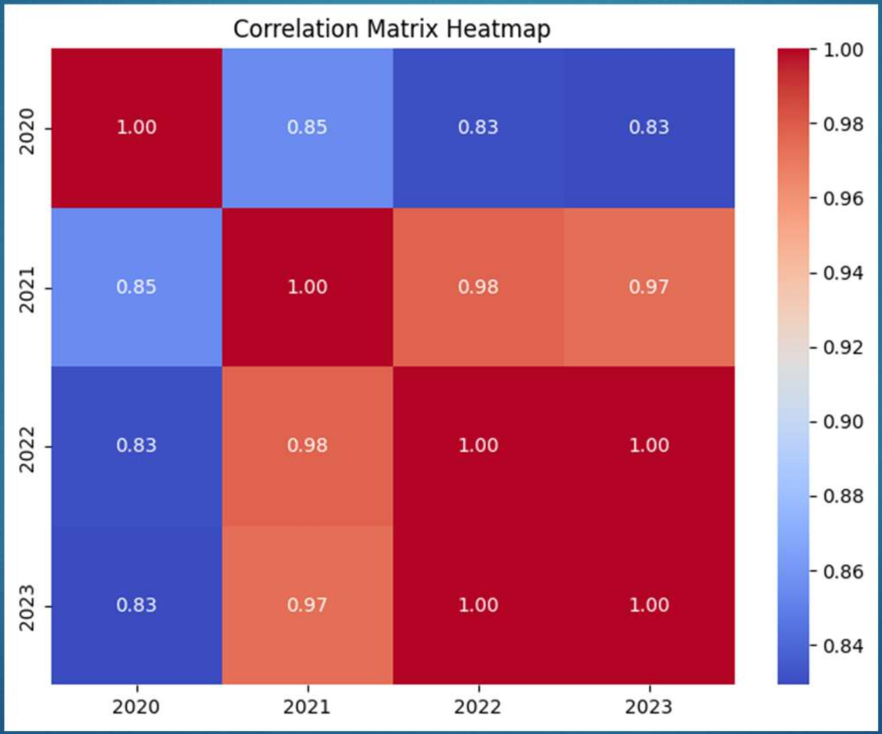
```
[91] df.info()

     <class 'pandas.core.frame.DataFrame'>
     RangeIndex: 3336 entries, 0 to 3335
     Data columns (total 9 columns):
      #   Column      Non-Null Count  Dtype
     ---  ------      --------------  -----
      0   County      3336 non-null   object
      1   State       3336 non-null   object
      2   Lat         3336 non-null   float64
      3   Long_       3336 non-null   float64
      4   Population  3336 non-null   int64
      5   2020        3336 non-null   int64
      6   2021        3336 non-null   int64
      7   2022        3336 non-null   int64
      8   2023        3336 non-null   int64
     dtypes: float64(2), int64(5), object(2)
     memory usage: 234.7+ KB
```



```
df.describe()
```

	Lat	Long_	Population	2020	2021	2022	2023
count	3336.00	3336.00	3336.00	3336.00	3336.00	3336.00	3336.00
mean	36.78	-88.82	99668.12	14027.56	66962.96	110289.98	22610.50
std	8.98	20.87	324444.64	73788.97	255488.90	366427.36	74516.40
min	0.00	-174.16	0.00	0.00	0.00	-156.00	0.00
25%	33.92	-97.81	9917.75	454.00	6380.00	12525.00	2590.75
50%	38.02	-89.50	24848.50	1788.50	17959.50	33382.50	6858.50
75%	41.59	-82.33	64967.75	5840.50	43415.00	81425.25	16590.25
max	69.31	0.00	10039107.00	1836989.00	8624287.00	11766936.00	2390574.00

Correlation Matrix

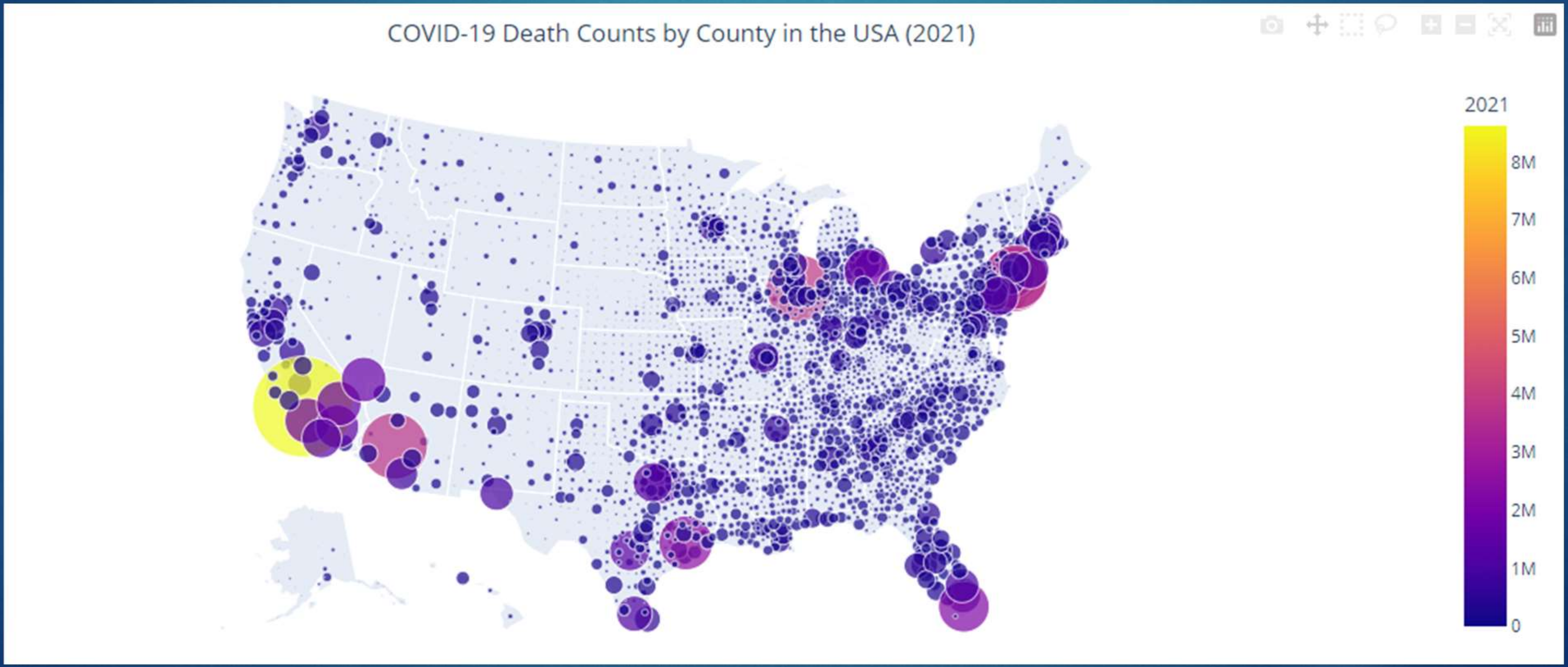


Correlation Matrix Heatmap

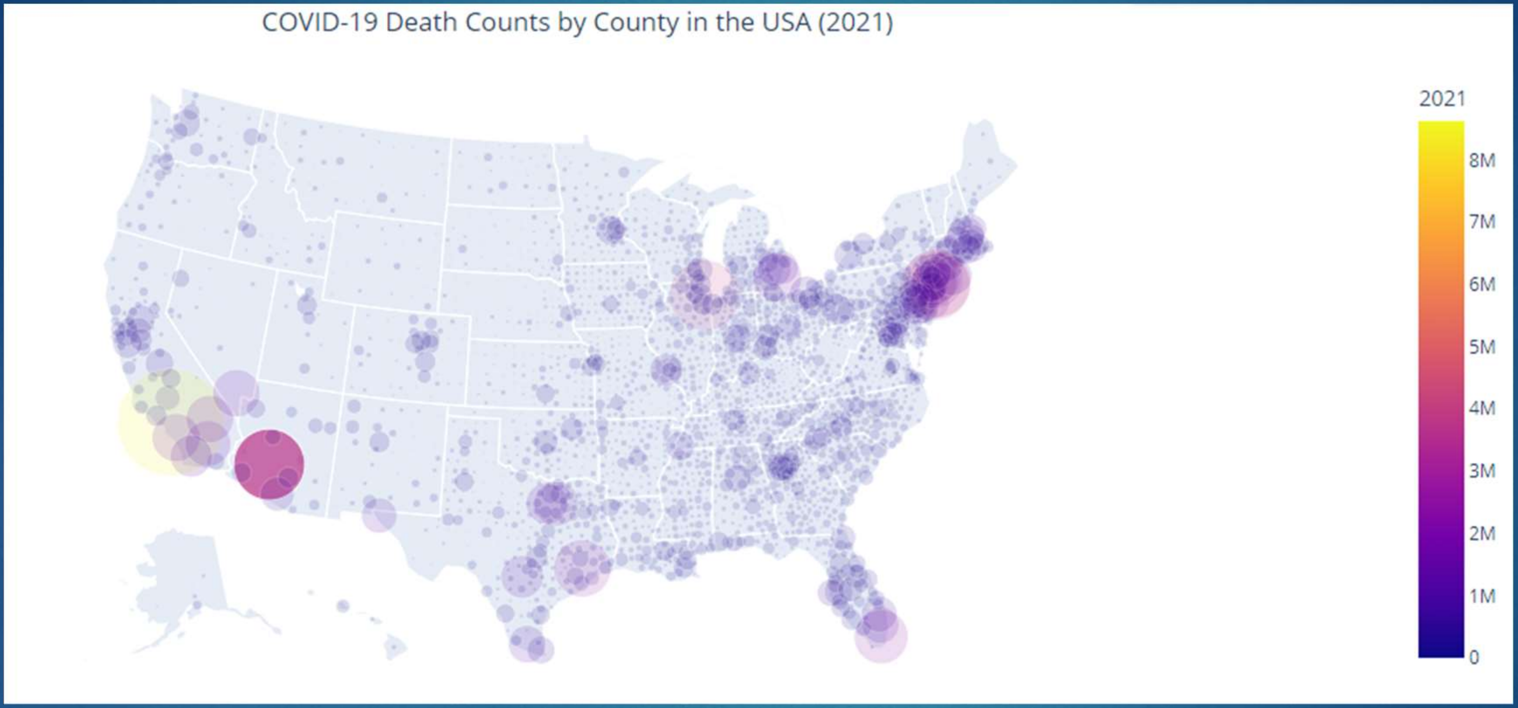
	2020	2021	2022	2023
2020	1.00	0.85	0.83	0.83
2021	0.85	1.00	0.98	0.97
2022	0.83	0.98	1.00	1.00
2023	0.83	0.97	1.00	1.00

Interactive Map Visualization 1

Question: How do COVID-19 death counts vary across different counties in the USA in 2021?



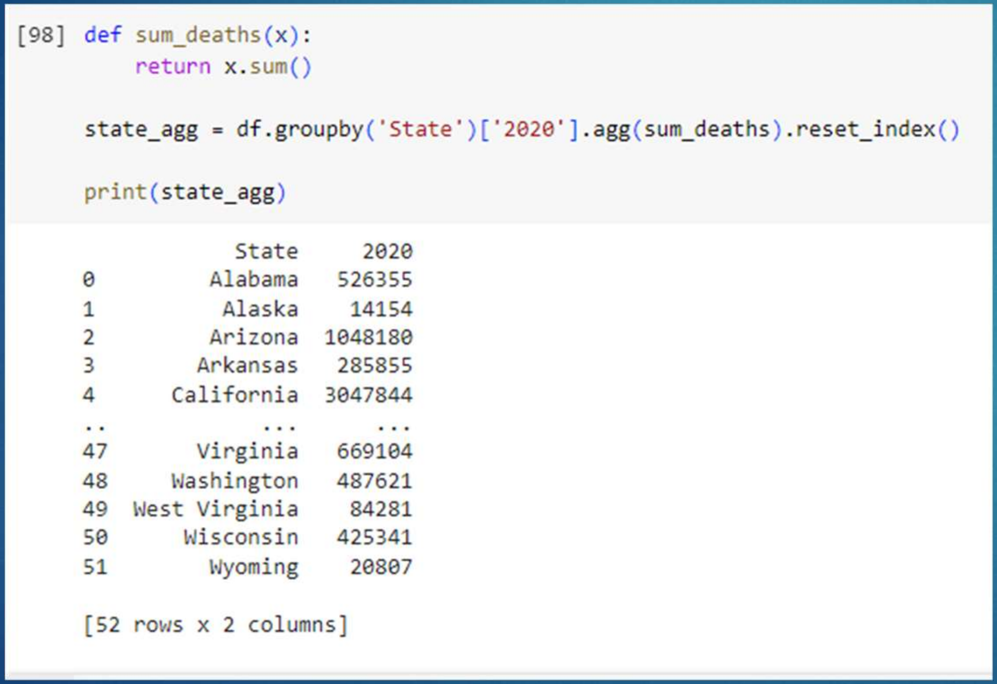
Continue



Each county is represented by a bubble on the map, where the size and color of the bubble indicate the death count for that county. This visualization provides a clear understanding of the distribution of COVID-19 fatalities across different counties in the USA during the year 2021.

Interactive Aggregation Visualization 1

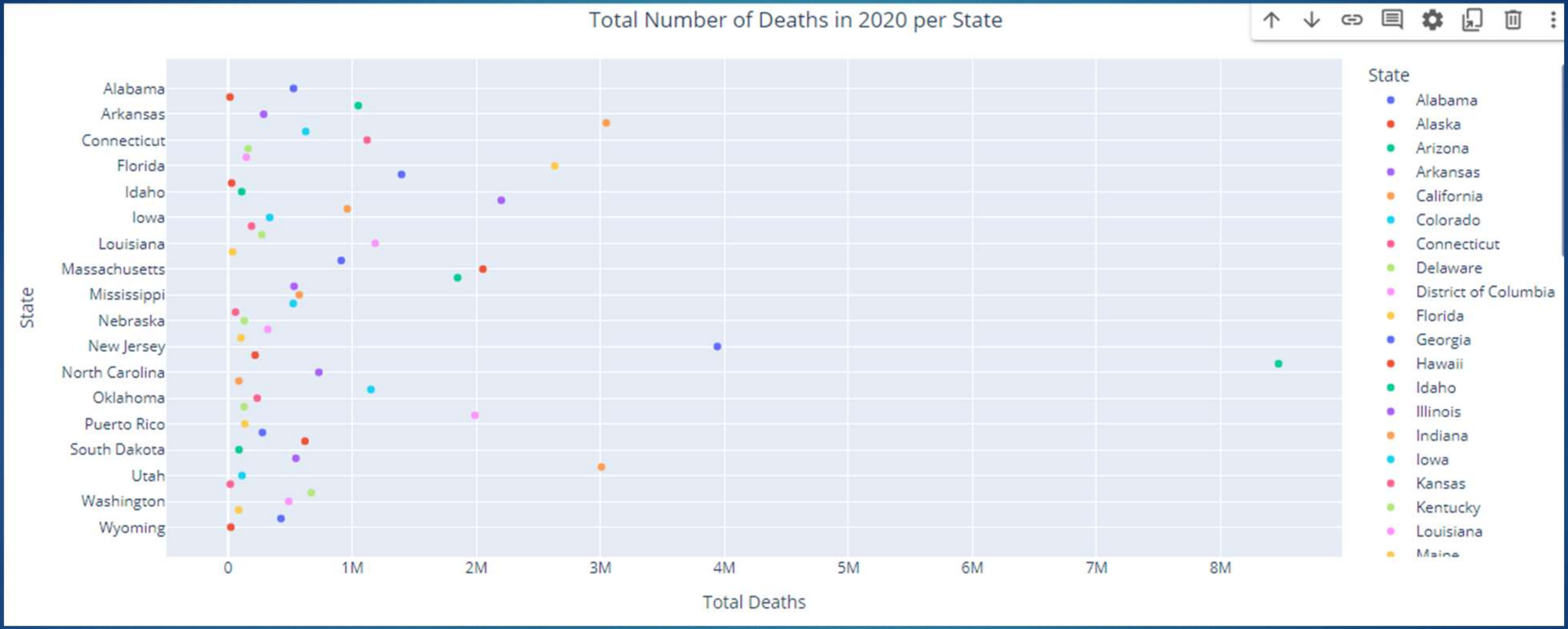
Question: How many people died in 2020 in each state?



```
[98] def sum_deaths(x):
         return x.sum()

     state_agg = df.groupby('State')['2020'].agg(sum_deaths).reset_index()

     print(state_agg)
```

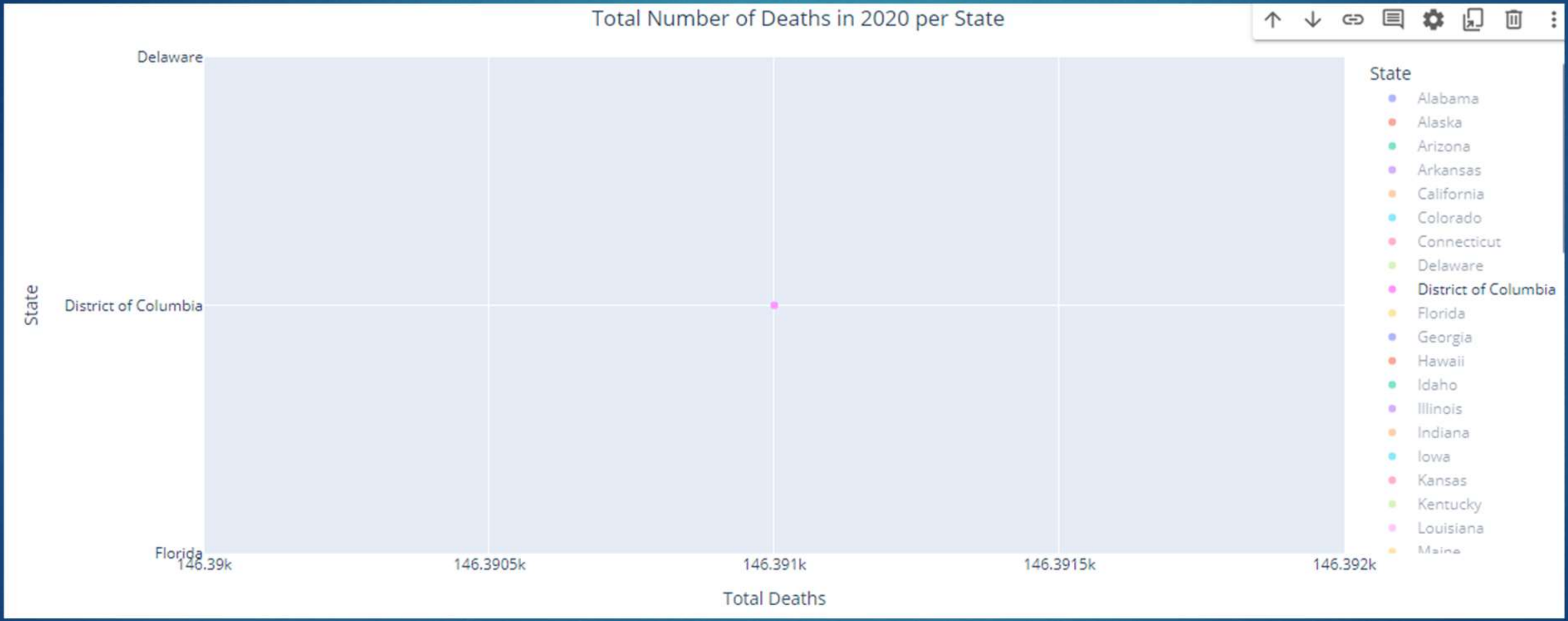
```
            State     2020
0         Alabama   526355
1          Alaska    14154
2         Arizona  1048180
3        Arkansas   285855
4      California  3047844
..            ...      ...
47       Virginia   669104
48     Washington   487621
49  West Virginia    84281
50      Wisconsin   425341
51        Wyoming    20807

[52 rows x 2 columns]
```

Continue



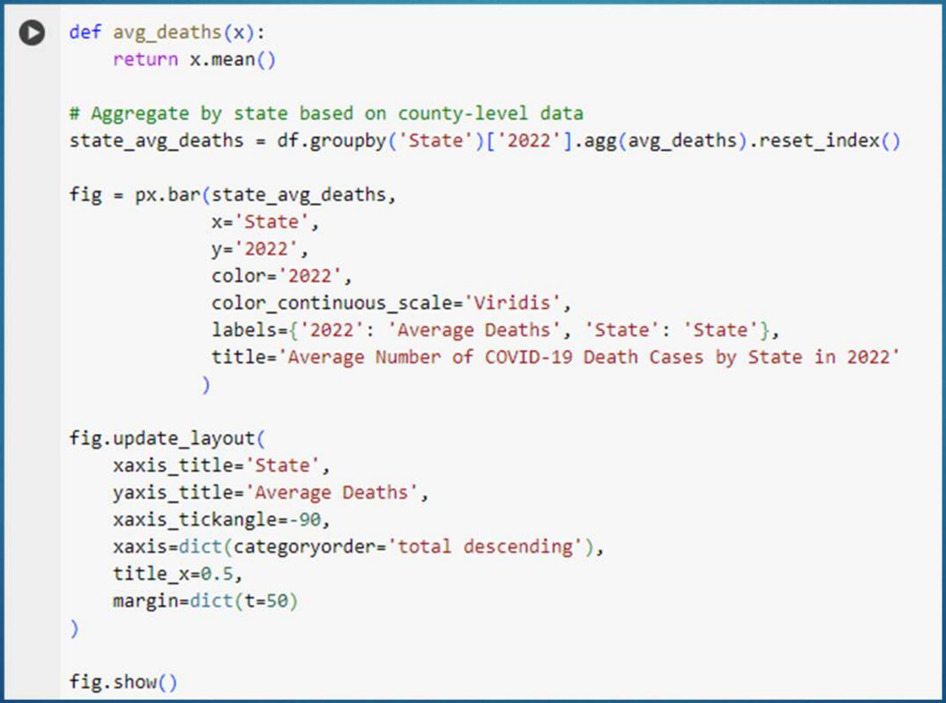
From this figure, we see that most number of people died in 2020 in the state of New York, New Jersey, California and Texas.

Continue


Total Number of Deaths in 2020 per State
Delaware
District of Columbia
Florida
State
146.39k
146.3905k
146.391k
146.3915k
146.392k
Total Deaths
State
Alabama
Alaska
Arizona
Arkansas
California
Colorado
Connecticut
Delaware
District of Columbia
Florida
Georgia
Hawaii
Idaho
Illinois
Indiana
Iowa
Kansas
Kentucky
Louisiana
Maine

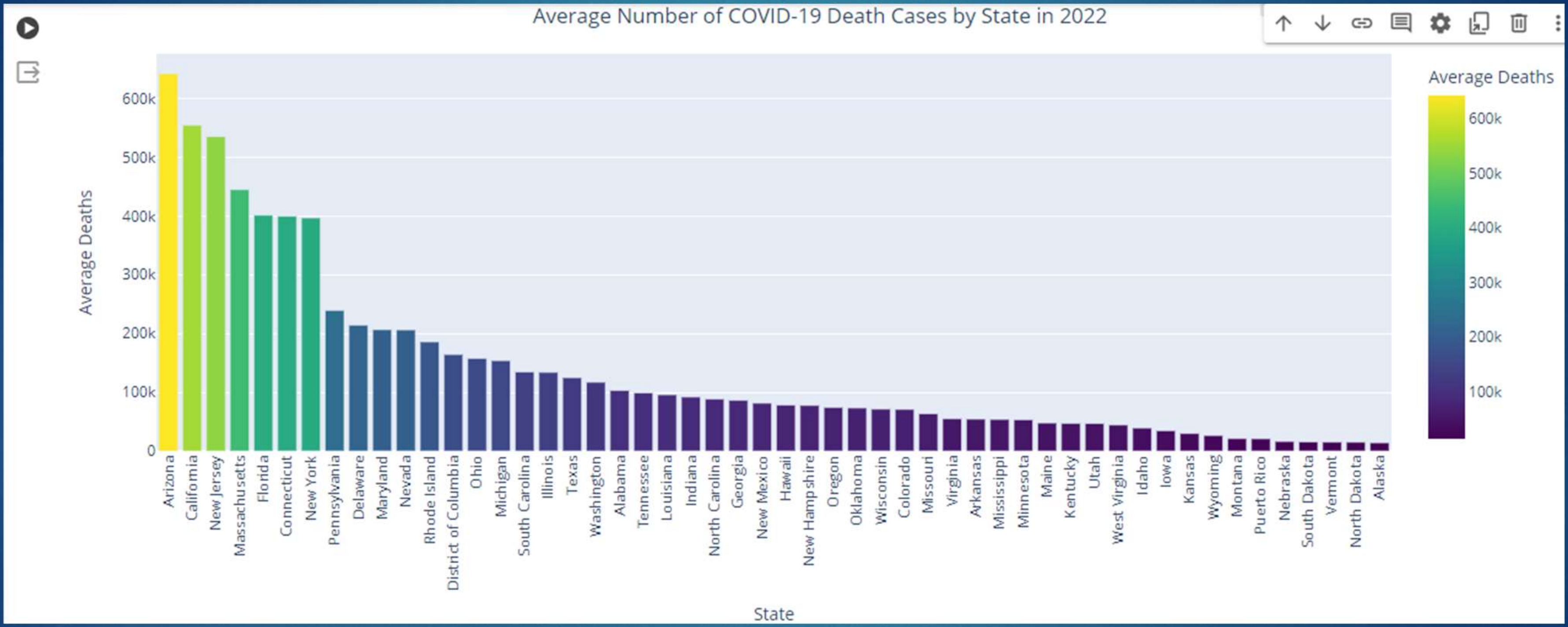
Aggregation Visualization 2

Question: What is the average number of COVID-19 deaths cases per US State in 2022?



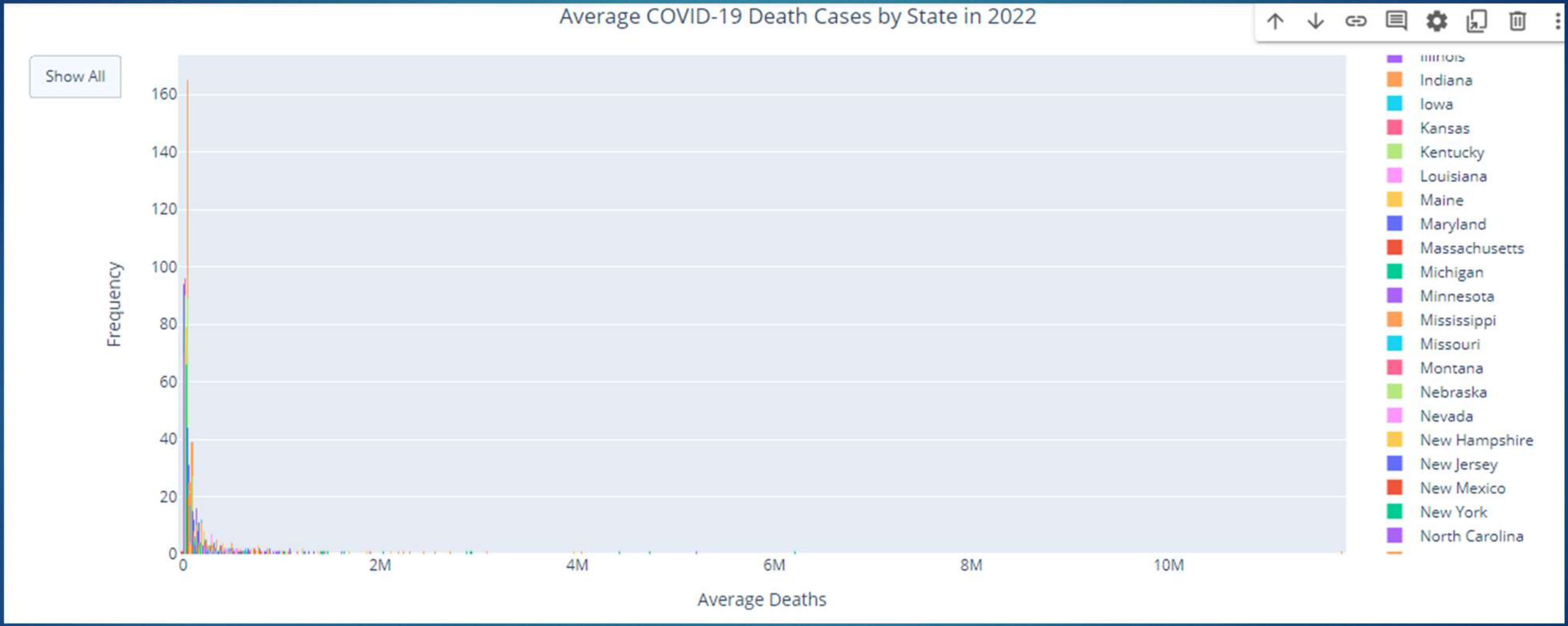
```
def avg_deaths(x):
    return x.mean()

# Aggregate by state based on county-level data
state_avg_deaths = df.groupby('State')['2022'].agg(avg_deaths).reset_index()

fig = px.bar(state_avg_deaths,
             x='State',
             y='2022',
             color='2022',
             color_continuous_scale='Viridis',
             labels={'2022': 'Average Deaths', 'State': 'State'},
             title='Average Number of COVID-19 Death Cases by State in 2022'
            )

fig.update_layout(
    xaxis_title='State',
    yaxis_title='Average Deaths',
    xaxis_tickangle=-90,
    xaxis=dict(categoryorder='total descending'),
    title_x=0.5,
    margin=dict(t=50)
)

fig.show()
```

Continue

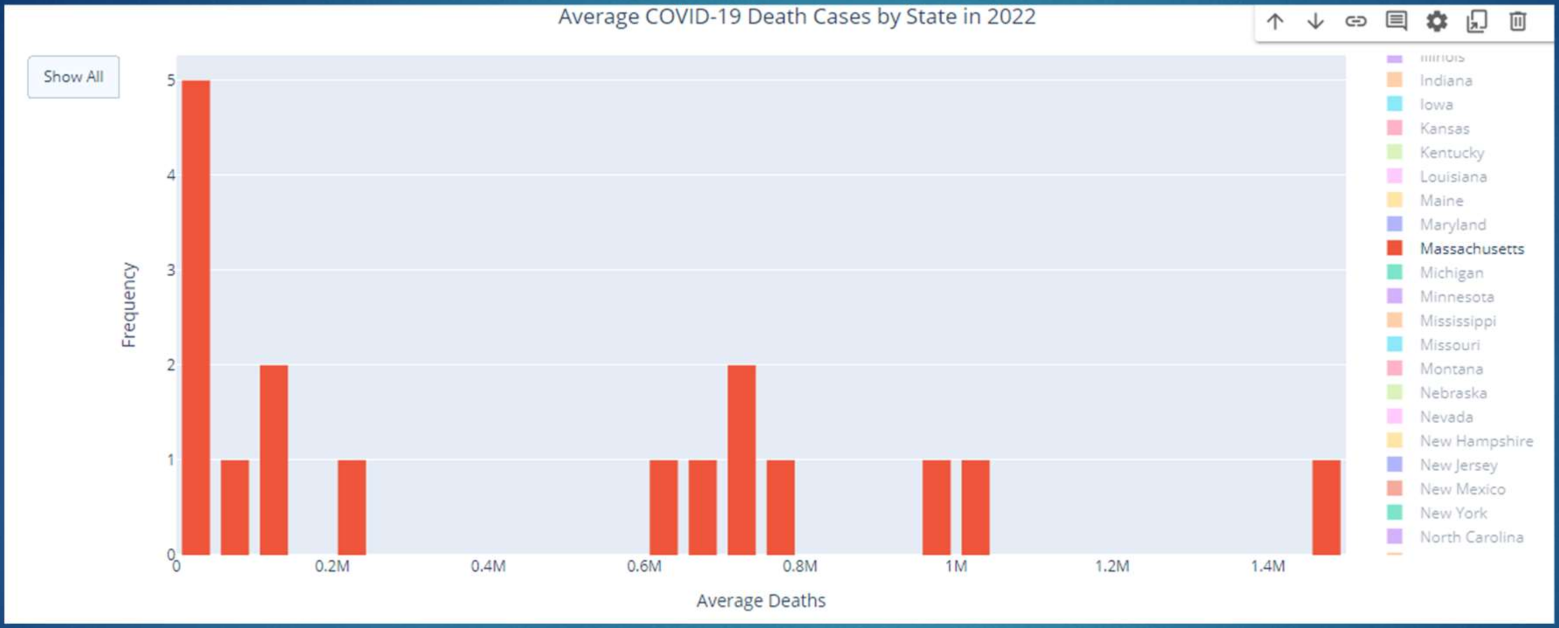


From the figure, we see that the highest average number of COVID-19 deaths cases in 2022 are in the state of Arizona, California, New Jersey.

Interactive Visualization 3



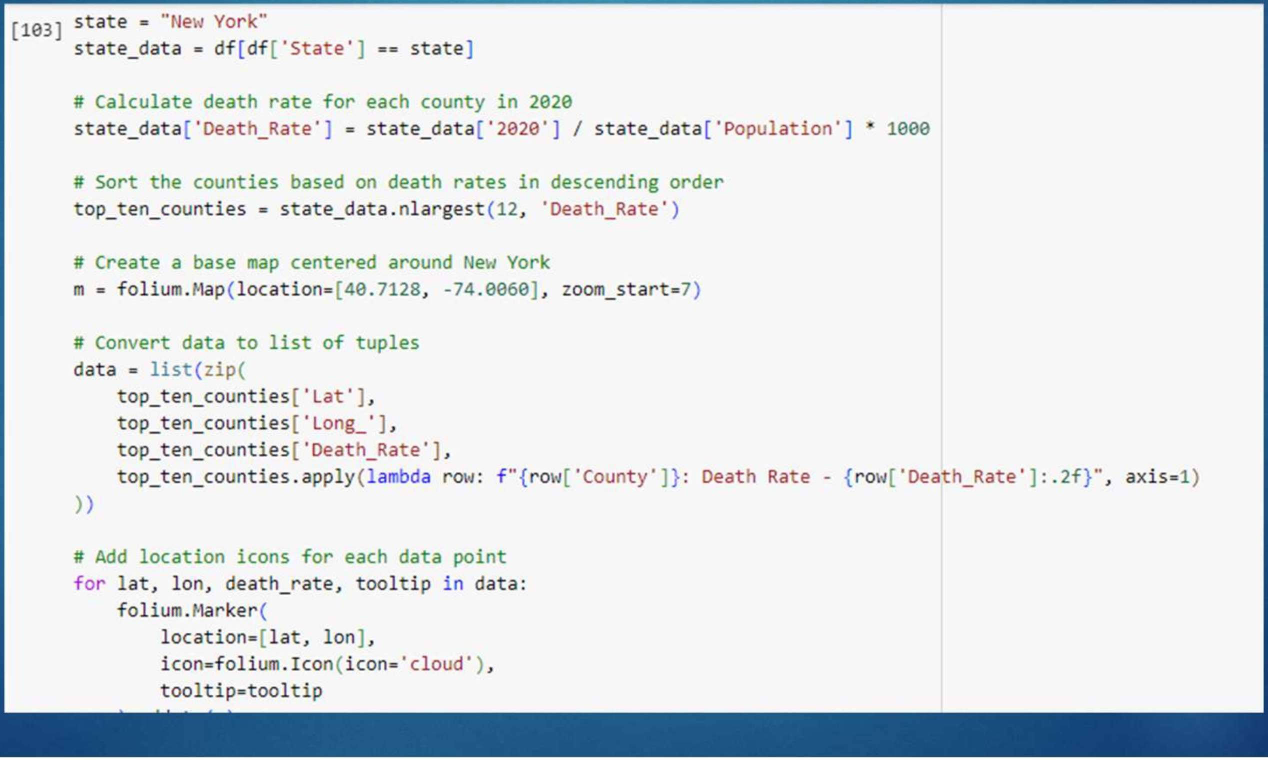
Continue



After double clicking on any state, we see that each bar in the histogram represents the distribution of average COVID-19 deaths cases across counties for that specific state.

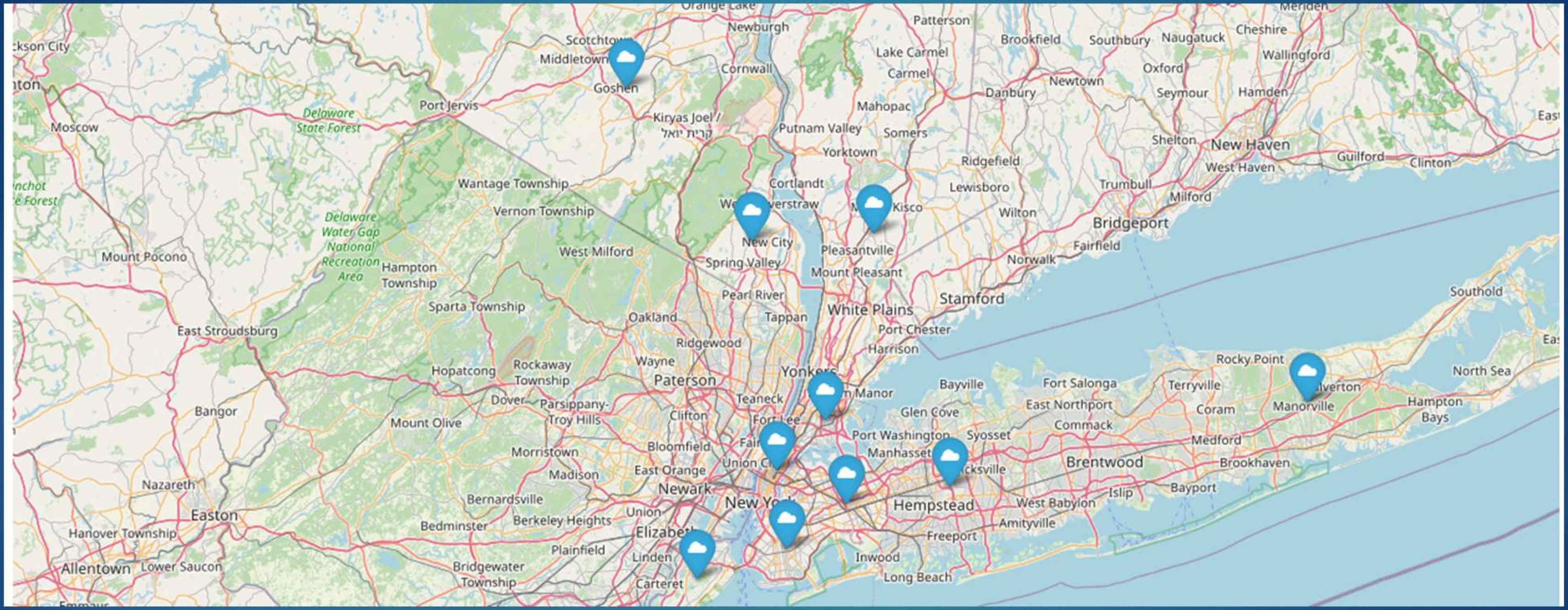
Map Visualization 2

Question: Find top ten counties of a particular state having the highest death rates in 2020?



```
[103] state = "New York"
      state_data = df[df['State'] == state]

      # Calculate death rate for each county in 2020
      state_data['Death_Rate'] = state_data['2020'] / state_data['Population'] * 1000

      # Sort the counties based on death rates in descending order
      top_ten_counties = state_data.nlargest(12, 'Death_Rate')

      # Create a base map centered around New York
      m = folium.Map(location=[40.7128, -74.0060], zoom_start=7)

      # Convert data to list of tuples
      data = list(zip(
          top_ten_counties['Lat'],
          top_ten_counties['Long_'],
          top_ten_counties['Death_Rate'],
          top_ten_counties.apply(lambda row: f"{row['County']}: Death Rate - {row['Death_Rate']:.2f}", axis=1)
      ))

      # Add location icons for each data point
      for lat, lon, death_rate, tooltip in data:
          folium.Marker(
              location=[lat, lon],
              icon=folium.Icon(icon='cloud'),
              tooltip=tooltip
```

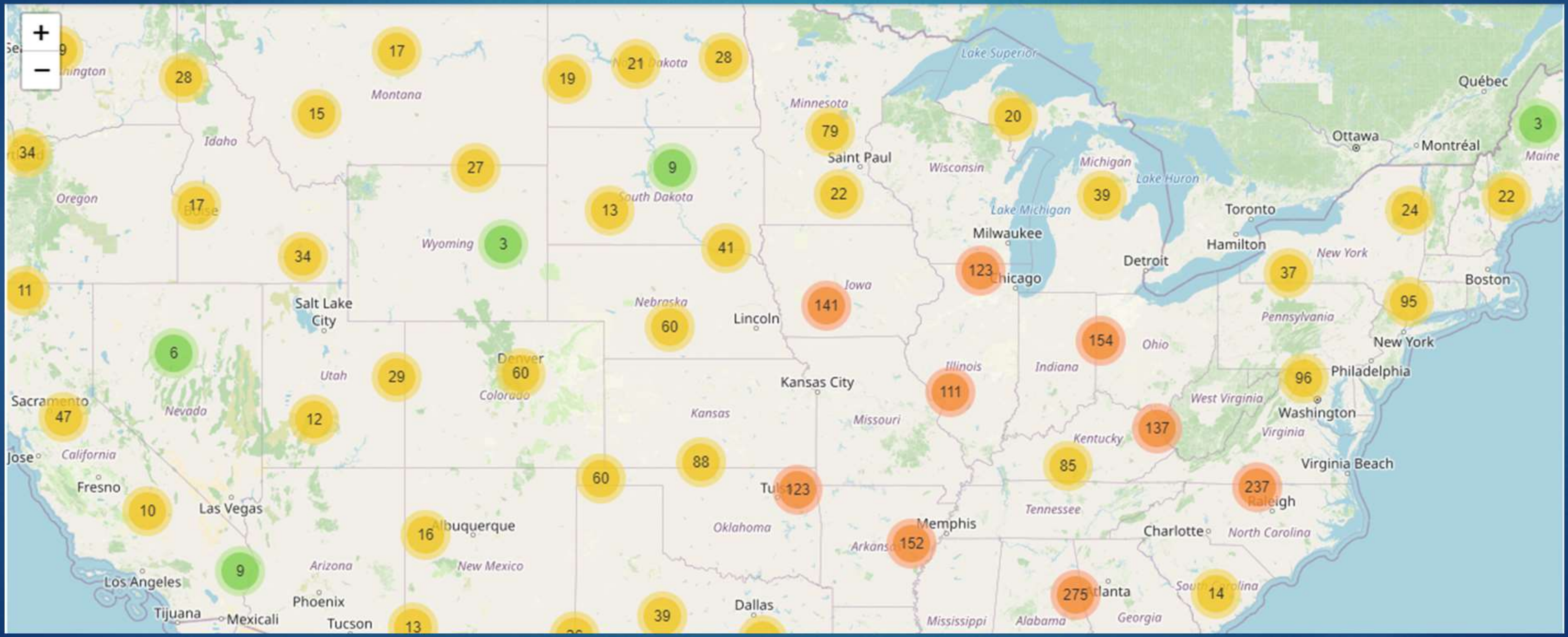
Continue



In this map, we can see that the map shows top 10 counties of New York state having highest covid19 death rates in 2020.

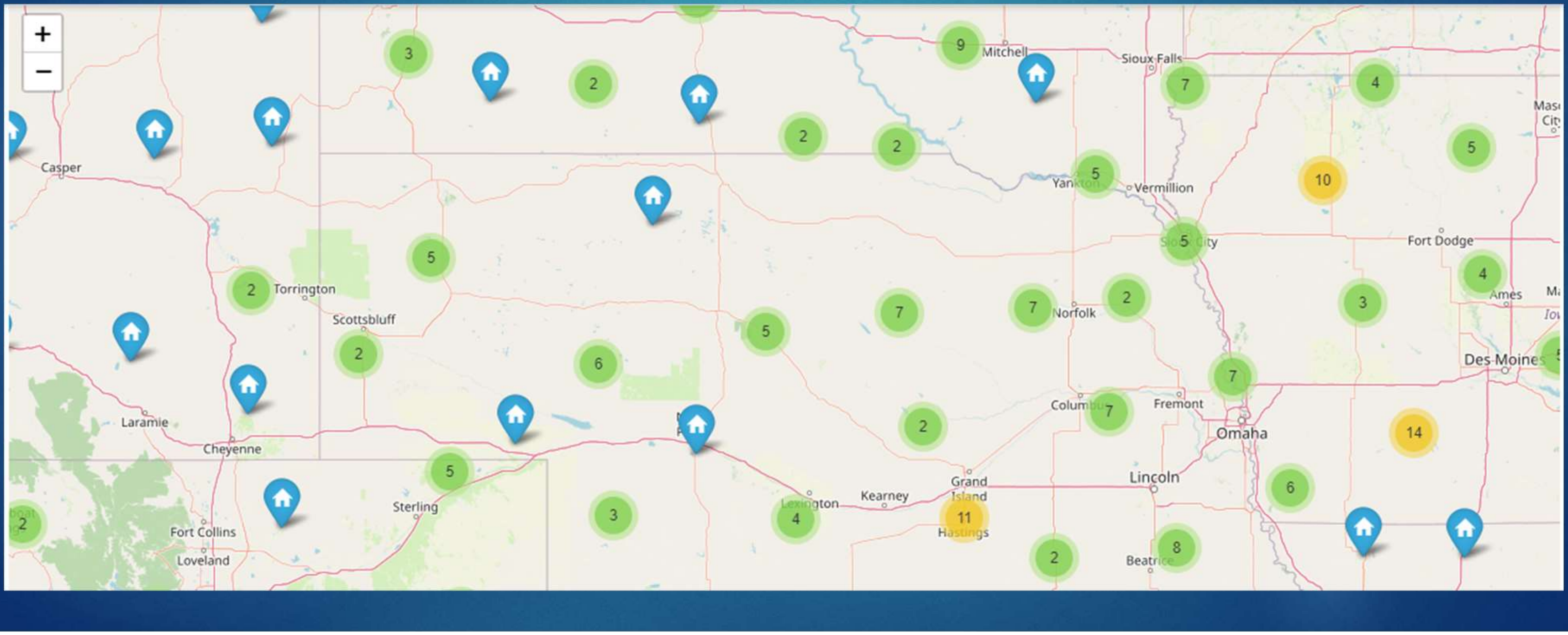
Interactive Visualization 4

Question: What is the number of death cases per city in each state?



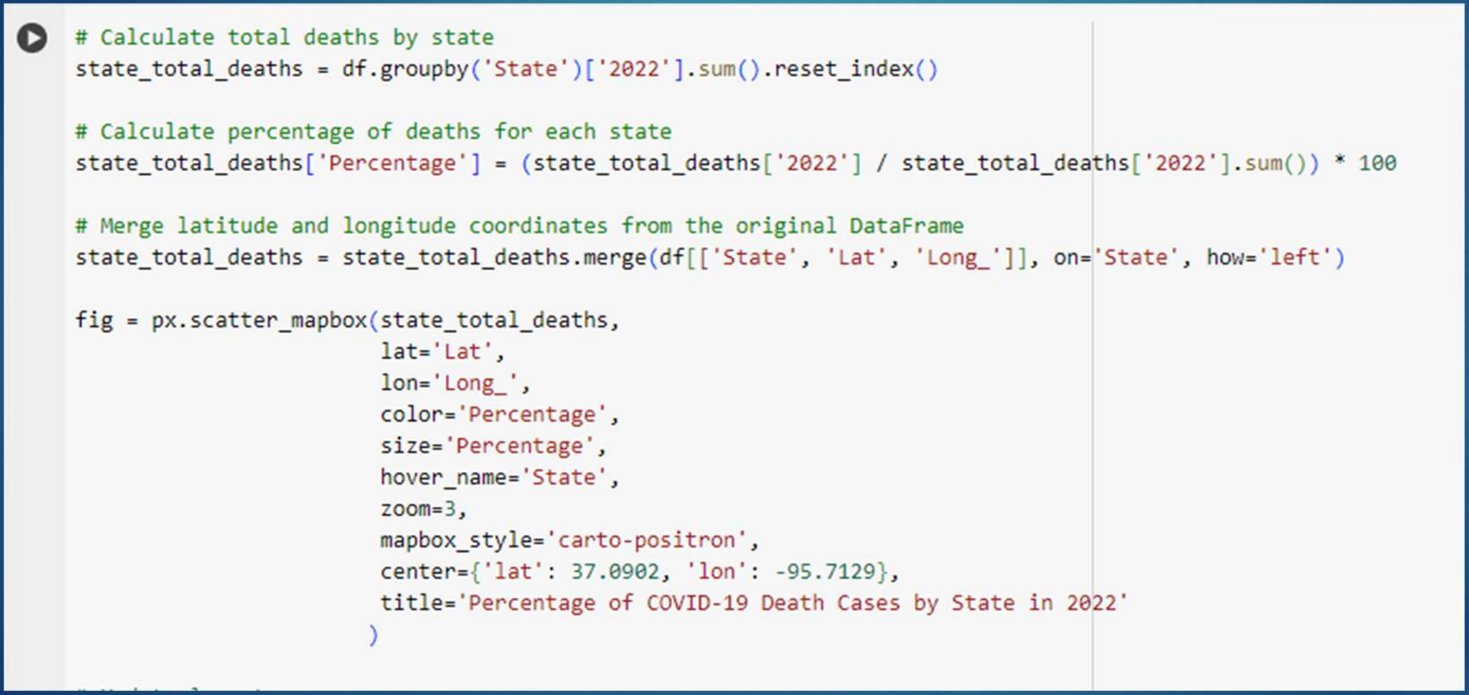
Continue

From this map, if we click on any state then we can see the number of death cases of different cities for that state.



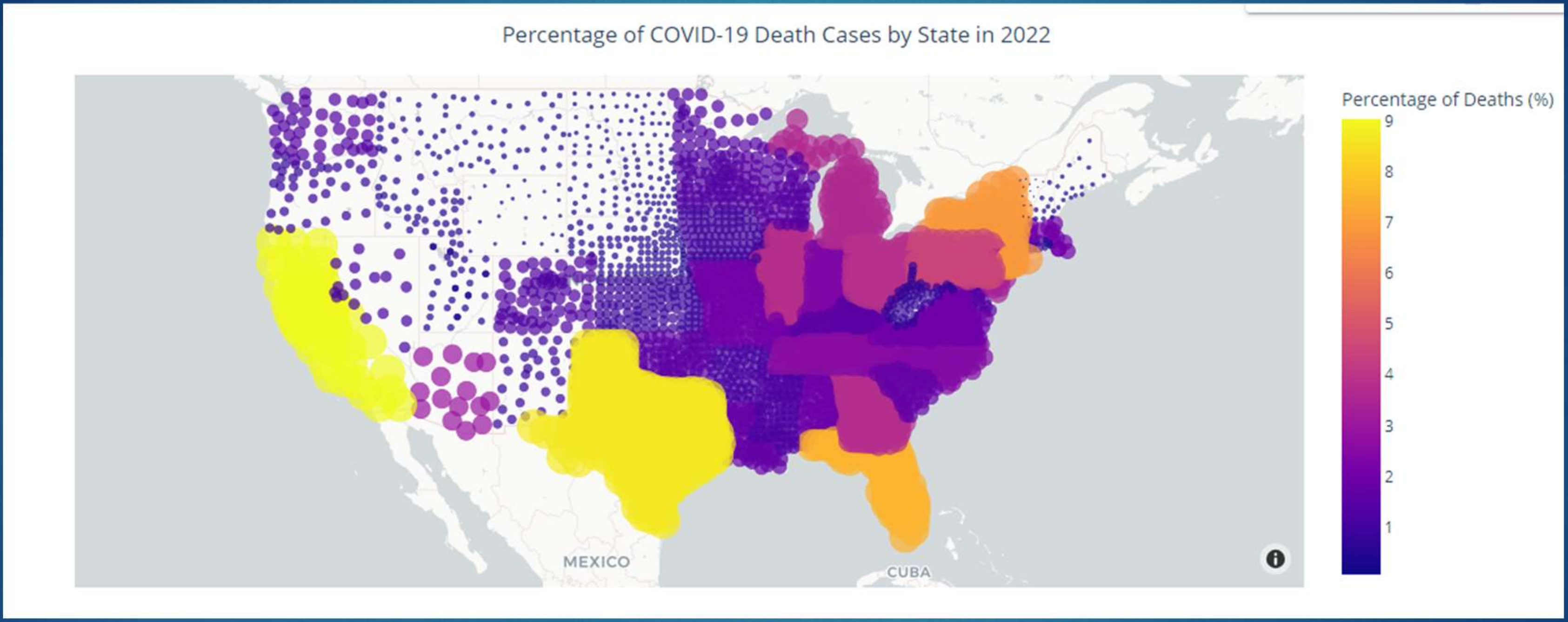
Map Visualization 3

Question: What is the percentage of deaths for each state in a specific year?



```
# Calculate total deaths by state
state_total_deaths = df.groupby('State')['2022'].sum().reset_index()

# Calculate percentage of deaths for each state
state_total_deaths['Percentage'] = (state_total_deaths['2022'] / state_total_deaths['2022'].sum()) * 100

# Merge latitude and longitude coordinates from the original DataFrame
state_total_deaths = state_total_deaths.merge(df[['State', 'Lat', 'Long_']], on='State', how='left')

fig = px.scatter_mapbox(state_total_deaths,
                        lat='Lat',
                        lon='Long_',
                        color='Percentage',
                        size='Percentage',
                        hover_name='State',
                        zoom=3,
                        mapbox_style='carto-positron',
                        center={'lat': 37.0902, 'lon': -95.7129},
                        title='Percentage of COVID-19 Death Cases by State in 2022'
                       )
```

Continue



We can see the percentage of COVID-19 death cases by state in 2022 in this figure.

Conclusion

- COVID-19 death counts varied significantly across US counties in 2021 due to factors like population density and healthcare infrastructure.
- In 2020, COVID-19 deaths varied by state, reflecting differences in virus spread and public health responses.
- The average number of COVID-19 deaths per US state in 2022 indicates ongoing pandemic impact.
- Top ten counties within a state with highest death rates in 2020 faced challenges in containment efforts.
- Analyzing death cases per city within each state offers insights into localized transmission patterns.
- Calculating the percentage of deaths for each state in a specific year informs resource allocation and targeted interventions.

Thank You!