In the following section, I load the COVID-19 dataset and filter it to include only the columns related to cases and deaths. The dataset is in wide format, so I use pd.melt() to reshape it into a long format, making it easier to perform time-based analysis. Then I converted the 'date' column to datetime format.

In [21]:
```
import pandas as pd

# Load the dataset
file_path = '/Users/amnasohail/Desktop/405Stage2indv/final_merged_data.csv'
state_data = pd.read_csv(file_path)

# Filter columns for cases and deaths
# Only keep columns that contain '_cases' or '_deaths' in their names, and exclude non-relevant columns
cases_columns = [col for col in state_data.columns if '_cases' in col]
deaths_columns = [col for col in state_data.columns if '_deaths' in col]

# Filter for case and death columns and necessary identifiers
filtered_cases = ['County Name', 'State', 'population'] + cases_columns
filtered_deaths = ['County Name', 'State', 'population'] + deaths_columns

filtered_cases_data = state_data[filtered_cases]
filtered_deaths_data = state_data[filtered_deaths]

# Melt the data to long format for cases
data_cases_long = pd.melt(
    filtered_cases_data,
    id_vars=['County Name', 'State', 'population'],
    value_vars=cases_columns,  # Only melt the case columns
    var_name='date',
    value_name='cases'
)

# Melt the data to long format for deaths
data_deaths_long = pd.melt(
    filtered_deaths_data,
    id_vars=['County Name', 'State', 'population'],
    value_vars=deaths_columns,  # Only melt the death columns
    var_name='date',
    value_name='deaths'
)

# Clean the 'date' column by removing '_cases' or '_deaths' and convert to datetime
data_cases_long['date'] = pd.to_datetime(data_cases_long['date'].str.replace('_cases', ''), format='%Y-%m-%d
data_deaths_long['date'] = pd.to_datetime(data_deaths_long['date'].str.replace('_deaths', ''), format='%Y-%m

# Group by State and Date, and calculate weekly sums
weekly_cases = data_cases_long.groupby(['State', pd.Grouper(key='date', freq='W')]).sum().reset_index()
weekly_deaths = data_deaths_long.groupby(['State', pd.Grouper(key='date', freq='W')]).sum().reset_index()

# List of states to filter
selected_states = ['AL', 'CA', 'WA', 'SC', 'MI', 'MA']

# Filter weekly cases for the selected states
weekly_cases_filtered = weekly_cases[weekly_cases['State'].isin(selected_states)]

# Filter weekly deaths for the selected states
weekly_deaths_filtered = weekly_deaths[weekly_deaths['State'].isin(selected_states)]

# Display the filtered weekly data for cases and deaths
print(weekly_cases_filtered.head())
print(weekly_deaths_filtered.head())
```

```
    State       date                                        County Name  \
182    AL 2020-01-26  Autauga County Baldwin County Barbour County B...
183    AL 2020-02-02  Autauga County Baldwin County Barbour County B...
184    AL 2020-02-09  Autauga County Baldwin County Barbour County B...
185    AL 2020-02-16  Autauga County Baldwin County Barbour County B...
186    AL 2020-02-23  Autauga County Baldwin County Barbour County B...

     population  cases
182    24515925      0
183    34322295      0
184    34322295      0
185    34322295      0
186    34322295      0
    State       date                                        County Name  \
182    AL 2020-01-26  Autauga County Baldwin County Barbour County B...
183    AL 2020-02-02  Autauga County Baldwin County Barbour County B...
184    AL 2020-02-09  Autauga County Baldwin County Barbour County B...
185    AL 2020-02-16  Autauga County Baldwin County Barbour County B...
186    AL 2020-02-23  Autauga County Baldwin County Barbour County B...

     population  deaths
182    24515925       0
183    34322295       0
184    34322295       0
185    34322295       0
186    34322295       0
```

In [35]:

```
# Filter the data for Alabama
alabama_cases = weekly_cases_filtered[weekly_cases_filtered['State'] == 'AL'].copy()
alabama_deaths = weekly_deaths_filtered[weekly_deaths_filtered['State'] == 'AL'].copy()

# Normalize cases and deaths per 100,000 people for Alabama using .loc to avoid SettingWithCopyWarning
alabama_cases.loc[:, 'cases_per_100k'] = (alabama_cases['cases'] / alabama_cases['population']) * 100000
alabama_deaths.loc[:, 'deaths_per_100k'] = (alabama_deaths['deaths'] / alabama_deaths['population']) * 10000

# Calculate weekly mean, median, and mode for cases and deaths in Alabama

# For cases
weekly_mean_cases_al = alabama_cases_stats['mean_cases'].mean()
weekly_median_cases_al = alabama_cases_stats['median_cases'].median()
weekly_mode_cases_al = alabama_cases_stats['mode_cases'].mode()[0]

# For deaths
weekly_mean_deaths_al = alabama_deaths_stats['mean_deaths'].mean()
weekly_median_deaths_al = alabama_deaths_stats['median_deaths'].median()
weekly_mode_deaths_al = alabama_deaths_stats['mode_deaths'].mode()[0]

# Display the weekly statistics for Alabama in a formatted manner
print(f"Weekly Mean of Cases in Alabama: {weekly_mean_cases_al}")
print(f"Weekly Median of Cases in Alabama: {weekly_median_cases_al}")
print(f"Weekly Mode of Cases in Alabama: {weekly_mode_cases_al}")
print()

print(f"Weekly Mean of Deaths in Alabama: {weekly_mean_deaths_al}")
print(f"Weekly Median of Deaths in Alabama: {weekly_median_deaths_al}")
print(f"Weekly Mode of Deaths in Alabama: {weekly_mode_deaths_al}")
print()
```

```
Weekly Mean of Cases in Alabama: 17380.16780405117
Weekly Median of Cases in Alabama: 16602.770881143
Weekly Mode of Cases in Alabama: 33854.23964219176

Weekly Mean of Deaths in Alabama: 258.0357283298914
Weekly Median of Deaths in Alabama: 309.48105305895194
Weekly Mode of Deaths in Alabama: 0.0
```

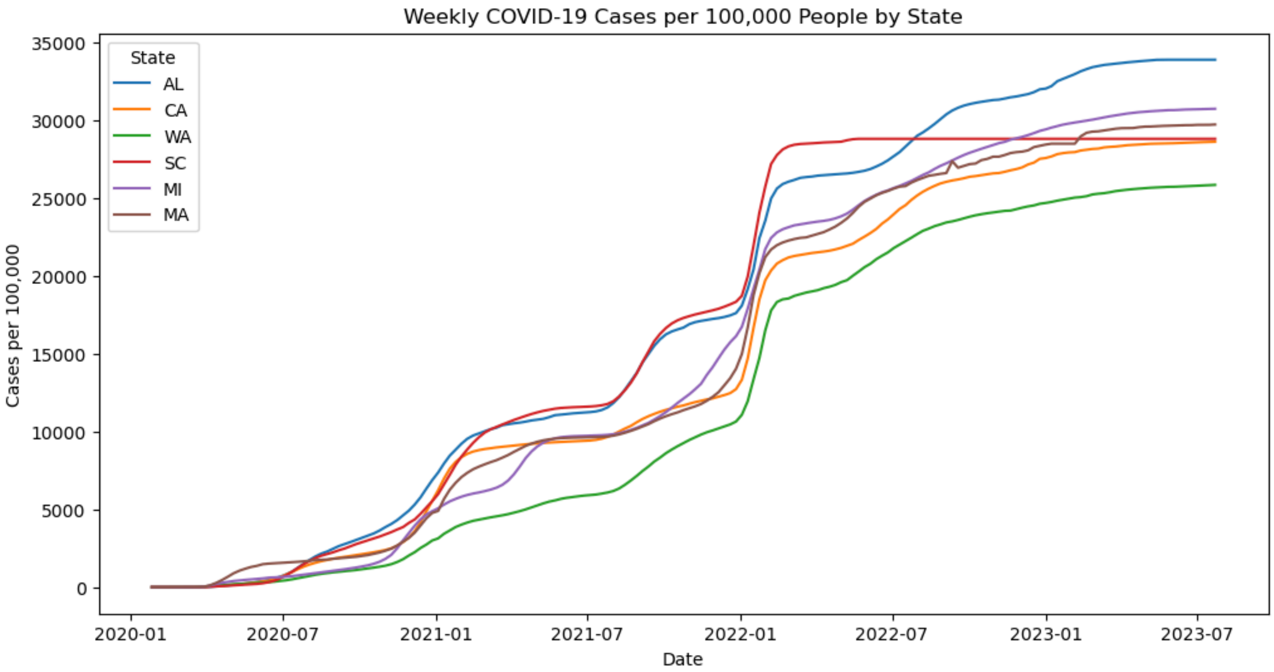
In [50]:

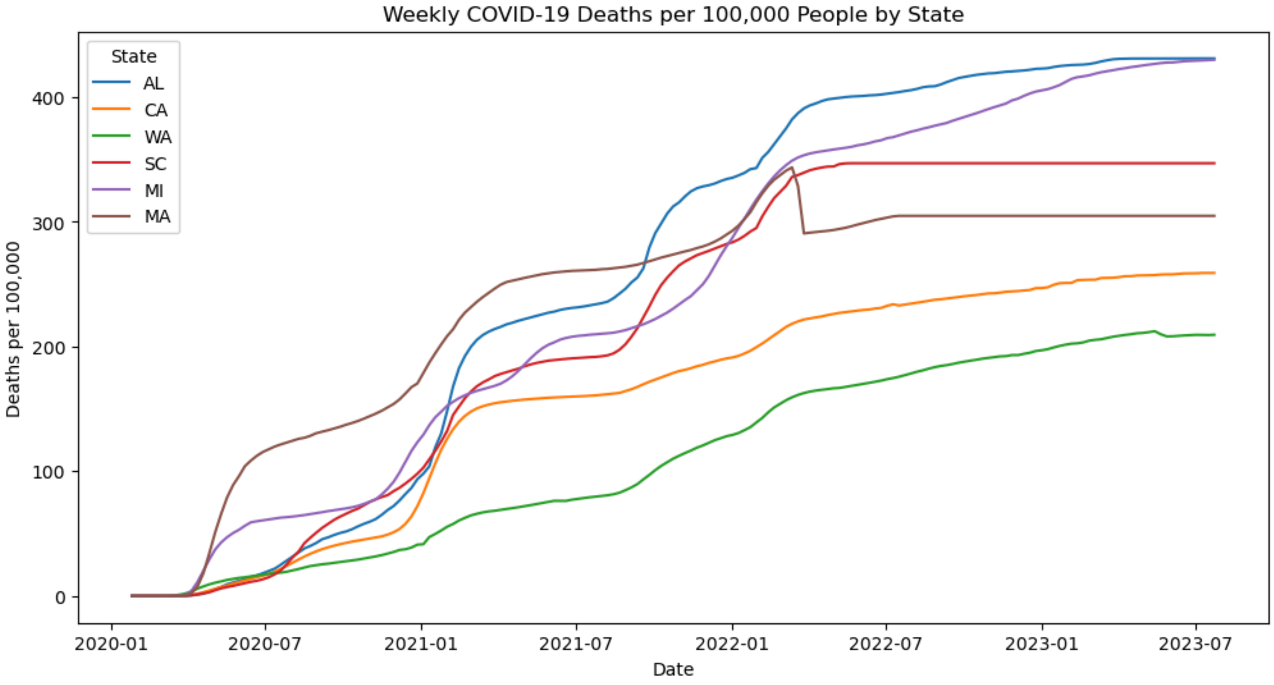
```
# List of states
other_states = ['CA', 'WA', 'SC', 'MI', 'MA']

for state in other_states:
    print(f"Weekly statistics for {state}:\n")

    # Filter data for each state
    state_cases = weekly_cases_filtered[weekly_cases_filtered['State'] == state].copy()
    state_deaths = weekly_deaths_filtered[weekly_deaths_filtered['State'] == state].copy()

    # Normalize cases and deaths per 100,000 people for the state
    state_cases['cases_per_100k'] = (state_cases['cases'] / state_cases['population']) * 100000
    state_deaths['deaths_per_100k'] = (state_deaths['deaths'] / state_deaths['population']) * 100000

    # Calculate weekly mean, median, and mode for cases
    weekly_mean_cases = state_cases.groupby(pd.Grouper(key='date', freq='W'))['cases_per_100k'].mean().mean(
    weekly_median_cases = state_cases.groupby(pd.Grouper(key='date', freq='W'))['cases_per_100k'].median().m
    weekly_mode_cases = state_cases.groupby(pd.Grouper(key='date', freq='W'))['cases_per_100k'].apply(lambda

    # Calculate weekly mean, median, and mode for deaths
    weekly_mean_deaths = state_deaths.groupby(pd.Grouper(key='date', freq='W'))['deaths_per_100k'].mean().me
    weekly_median_deaths = state_deaths.groupby(pd.Grouper(key='date', freq='W'))['deaths_per_100k'].median(
    weekly_mode_deaths = state_deaths.groupby(pd.Grouper(key='date', freq='W'))['deaths_per_100k'].apply(lam

    # Display the weekly statistics for the states
    print(f"Weekly Mean of Cases in {state}: {weekly_mean_cases}")
    print(f"Weekly Median of Cases in {state}: {weekly_median_cases}")
    print(f"Weekly Mode of Cases in {state}: {weekly_mode_cases}\n")

    print(f"Weekly Mean of Deaths in {state}: {weekly_mean_deaths}")
    print(f"Weekly Median of Deaths in {state}: {weekly_median_deaths}")
    print(f"Weekly Mode of Deaths in {state}: {weekly_mode_deaths}\n")
```

```
Weekly statistics for CA:

Weekly Mean of Cases in CA: 14396.782024823066
Weekly Median of Cases in CA: 11624.479196873188
Weekly Mode of Cases in CA: 0.0

Weekly Mean of Deaths in CA: 157.67247795422975
Weekly Median of Deaths in CA: 177.96301376701297
Weekly Mode of Deaths in CA: 0.0

Weekly statistics for WA:

Weekly Mean of Cases in WA: 12098.968116695483
Weekly Median of Cases in WA: 9151.312340923803
Weekly Mode of Cases in WA: 0.013132160885254724

Weekly Mean of Deaths in WA: 110.55408939904117
Weekly Median of Deaths in WA: 108.64518103818776
Weekly Mode of Deaths in WA: 0.0

Weekly statistics for SC:

Weekly Mean of Cases in SC: 16654.692497998785
Weekly Median of Cases in SC: 17205.989967314446
Weekly Mode of Cases in SC: 28777.011113843186

Weekly Mean of Deaths in SC: 218.0548464755134
Weekly Median of Deaths in SC: 257.80223954952635
Weekly Mode of Deaths in SC: 347.0575370859597

Weekly statistics for MI:

Weekly Mean of Cases in MI: 15102.57338205475
Weekly Median of Cases in MI: 11967.98738296028
Weekly Mode of Cases in MI: 0.0

Weekly Mean of Deaths in MI: 242.4092866468423
Weekly Median of Deaths in MI: 228.80858025989272
Weekly Mode of Deaths in MI: 0.0

Weekly statistics for MA:

Weekly Mean of Cases in MA: 14902.080249095174
Weekly Median of Cases in MA: 11291.142813327133
Weekly Mode of Cases in MA: 28464.652100985666

Weekly Mean of Deaths in MA: 233.3014360274787
Weekly Median of Deaths in MA: 273.1943471251093
Weekly Mode of Deaths in MA: 304.8384599905143
```

In [48]:

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

# list of states including Alabama
selected_states = ['AL', 'CA', 'WA', 'SC', 'MI', 'MA']

# empty lists to store the data for cases and deaths
consolidated_cases_data = []
consolidated_deaths_data = []

for state in selected_states:
    # Filter data for each state
    state_cases = weekly_cases_filtered[weekly_cases_filtered['State'] == state].copy()
    state_deaths = weekly_deaths_filtered[weekly_deaths_filtered['State'] == state].copy()

    # Normalize cases and deaths per 100,000 people for the state
    state_cases['cases_per_100k'] = (state_cases['cases'] / state_cases['population']) * 100000
    state_deaths['deaths_per_100k'] = (state_deaths['deaths'] / state_deaths['population']) * 100000

    consolidated_cases_data.append(state_cases)
    consolidated_deaths_data.append(state_deaths)

# Combine all the states' data into a single DataFrame for cases and deaths
consolidated_cases_data = pd.concat(consolidated_cases_data)
consolidated_deaths_data = pd.concat(consolidated_deaths_data)

# Plot Weekly COVID-19 Cases per 100,000 People for Each State
plt.figure(figsize=(12, 6))
sns.lineplot(data=consolidated_cases_data, x='date', y='cases_per_100k', hue='State')
plt.title('Weekly COVID-19 Cases per 100,000 People by State')
plt.xlabel('Date')
plt.ylabel('Cases per 100,000')
plt.legend(title='State')
plt.show()

# Plot Weekly COVID-19 Deaths per 100,000 People for Each State
plt.figure(figsize=(12, 6))
sns.lineplot(data=consolidated_deaths_data, x='date', y='deaths_per_100k', hue='State')
plt.title('Weekly COVID-19 Deaths per 100,000 People by State')
plt.xlabel('Date')
plt.ylabel('Deaths per 100,000')
plt.legend(title='State')
plt.show()
```





Analysis of Weekly COVID-19 Trends Across States

Some states, such as South Carolina and Michigan, experienced higher peaks in cases,but Washington and Massachusetts had lower but more sustained case numbers. The timing and the magnitude of deaths peaks was different across states, which I think shows differences in healthcare systems, population density, and public health responses.

Why the Rates Differ Across States

factors that could have contributed to the differences in COVID-19 rates across these states:
-Population Density: States with larger urban populations, such as California and Massachusetts, may have seen more rapid transmission due to closer physical contact.
-Healthcare Capacity: States with stronger healthcare infrastructure, like Massachusetts, may have been better equipped to handle severe cases, leading to lower death rates even with high case numbers.
-Public Health Measures: Differences in state policies (e.g., mask mandates, lockdowns) affected the spread of the virus. For example, stricter measures in Washington likely helped control the spread earlier in the pandemic.

```
In [51]: # Find the peak week for cases for each state
         peak_cases_week = consolidated_cases_data.loc[consolidated_cases_data.groupby('State')['cases_per_100k'].idx

         # Find the peak week for deaths for each state
         peak_deaths_week = consolidated_deaths_data.loc[consolidated_deaths_data.groupby('State')['deaths_per_100k']

         # Display the peak weeks for cases and deaths
         print("Peak weeks for cases:")
         print(peak_cases_week[['State', 'date', 'cases_per_100k']])

         print("\nPeak weeks for deaths:")
         print(peak_deaths_week[['State', 'date', 'deaths_per_100k']])
```

```
Peak weeks for cases:
     State       date  cases_per_100k
355     AL 2023-05-21    33854.239642
909     CA 2023-07-23    28589.035058
3639    MA 2023-07-23    29695.115320
4185    MI 2023-07-23    30704.008564
7401    SC 2022-05-22    28777.011114
8735    WA 2023-07-23    25819.287846

Peak weeks for deaths:
     State       date  deaths_per_100k
355     AL 2023-05-21       431.107535
909     CA 2023-07-23       259.048953
3569    MA 2022-03-13       343.706778
4185    MI 2023-07-23       429.996573
7401    SC 2022-05-22       347.057537
8726    WA 2023-05-14       212.325155
```

```
The peaks in COVID-19 cases and deaths across the states are probably related to factors like population
density, public health measures, and vaccination rates. States like California, Massachusetts, and Michigan
had case peaks in July 2023, most likely because of  summer travel and gatherings. South Carolina peaked
earlier in May 2022, reflecting looser restrictions. Death peaks usually followed case peaks by a few
weeks. States like Alabama and South Carolina, with earlier peaks, probably struggled with lower
vaccination rates and healthcare capacity. Overall, these patterns match national trends, with spikes
occurring during periods of increased travels and social interaction.
```

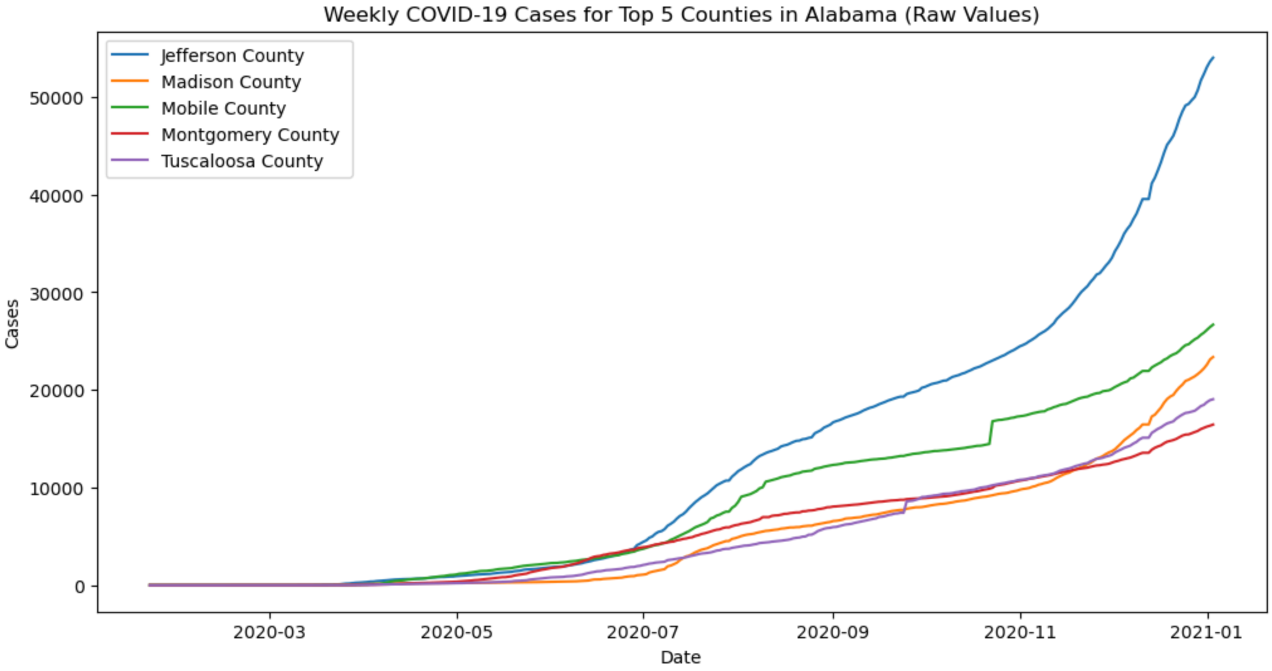
In [69]:

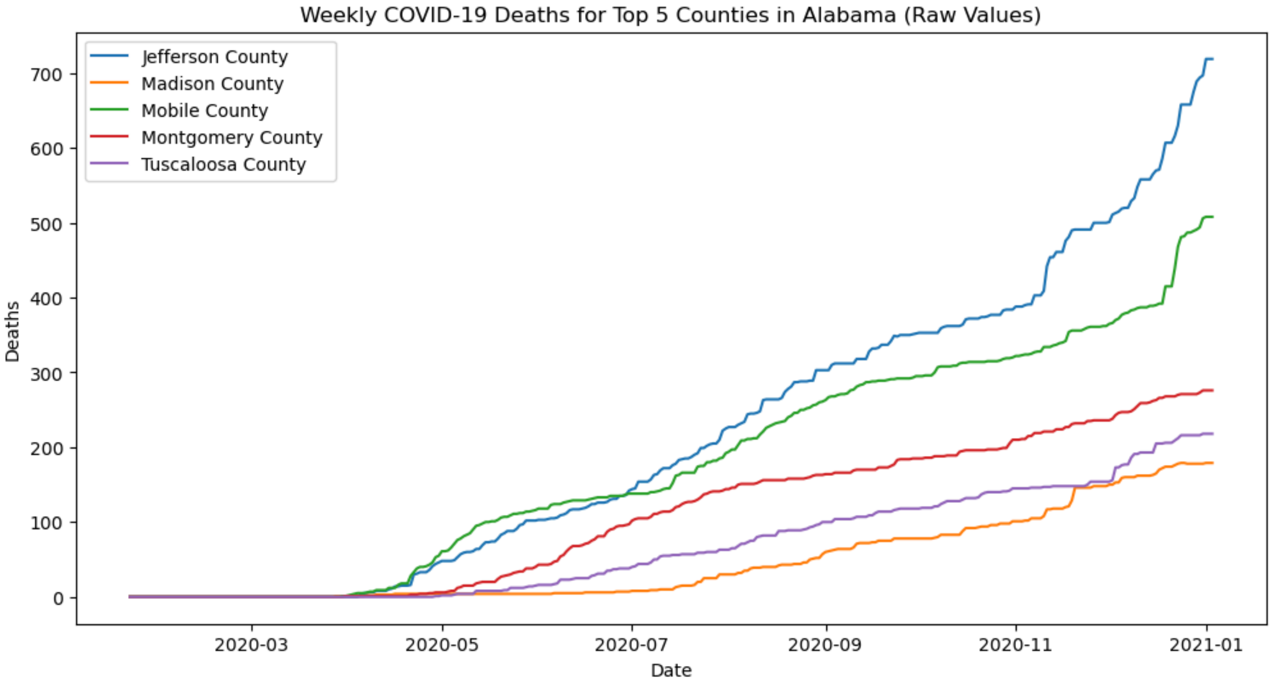
```
# Define the end date
end_date = '2021-01-03'

# Filter the data for Alabama
alabama_cases = data_cases_long[data_cases_long['State'] == 'AL']
alabama_deaths = data_deaths_long[data_deaths_long['State'] == 'AL']

# Filter the data for dates up to and including the end_date
alabama_cases_until_date = alabama_cases[alabama_cases['date'] <= end_date]
alabama_deaths_until_date = alabama_deaths[alabama_deaths['date'] <= end_date]

# Group by county and sum up the cases and deaths up to the end date
total_cases = alabama_cases_until_date.groupby('County Name').agg({
    'cases': 'sum',
    'population': 'max'
}).reset_index()

total_deaths = alabama_deaths_until_date.groupby('County Name').agg({
    'deaths': 'sum',
    'population': 'max'
}).reset_index()

# Sort and get the highest case and death rates by county
top_case_counties = total_cases.nlargest(5, 'cases')[['County Name', 'cases']]
top_death_counties = total_deaths.nlargest(5, 'deaths')[['County Name', 'deaths']]

# Print top counties by total cases and deaths until the end date
print(f"Top 5 Counties by Number of Cases up to {end_date}")
print(top_case_counties)

print(f"\nTop 5 Counties by Number of Deaths up to {end_date}")
print(top_death_counties)

# Get the list of top 5 county names
top_5_county_names = top_case_counties['County Name'].tolist()
```

```
Top 5 Counties by Number of Cases up to 2021-01-03
           County Name    cases
36   Jefferson County   4399139
48      Mobile County   2857206
50  Montgomery County   1884843
44     Madison County   1764606
62  Tuscaloosa County   1720640

Top 5 Counties by Number of Deaths up to 2021-01-03
           County Name  deaths
36   Jefferson County    75338
48      Mobile County    62330
50  Montgomery County    38269
62  Tuscaloosa County    23862
61  Tallapoosa County    19576
```

In [63]:

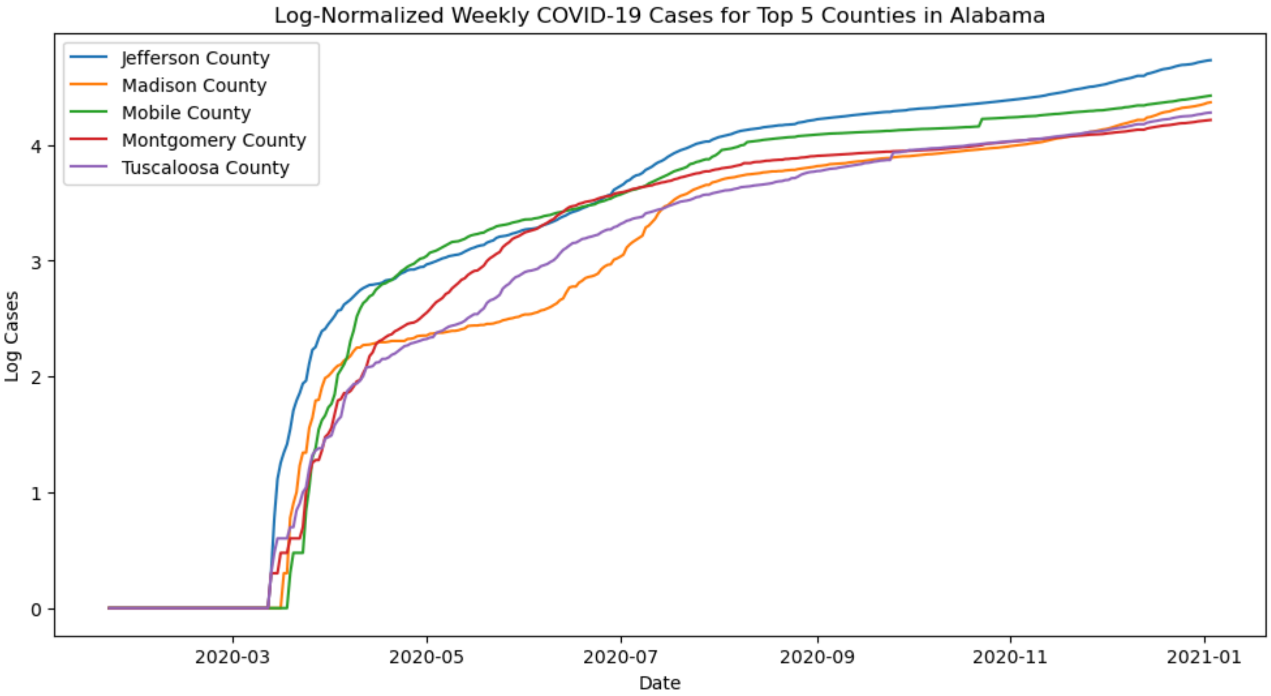
```
# Filter data for the top 5 counties
top_5_county_cases = alabama_cases_until_date[alabama_cases_until_date['County Name'].isin(top_5_county_name
top_5_county_deaths = alabama_deaths_until_date[alabama_deaths_until_date['County Name'].isin(top_5_county_n
In [64]: # Plot weekly cases for the top 5 counties
         plt.figure(figsize=(12, 6))
         sns.lineplot(data=top_5_county_cases, x='date', y='cases', hue='County Name')
         plt.title(f'Weekly COVID-19 Cases for Top 5 Counties in Alabama (Raw Values)')
         plt.xlabel('Date')
         plt.ylabel('Cases')
         plt.legend()
         plt.show()

         # Plot weekly deaths for the top 5 counties
         plt.figure(figsize=(12, 6))
         sns.lineplot(data=top_5_county_deaths, x='date', y='deaths', hue='County Name')
         plt.title(f'Weekly COVID-19 Deaths for Top 5 Counties in Alabama (Raw Values)')
         plt.xlabel('Date')
         plt.ylabel('Deaths')
         plt.legend()
         plt.show()
```

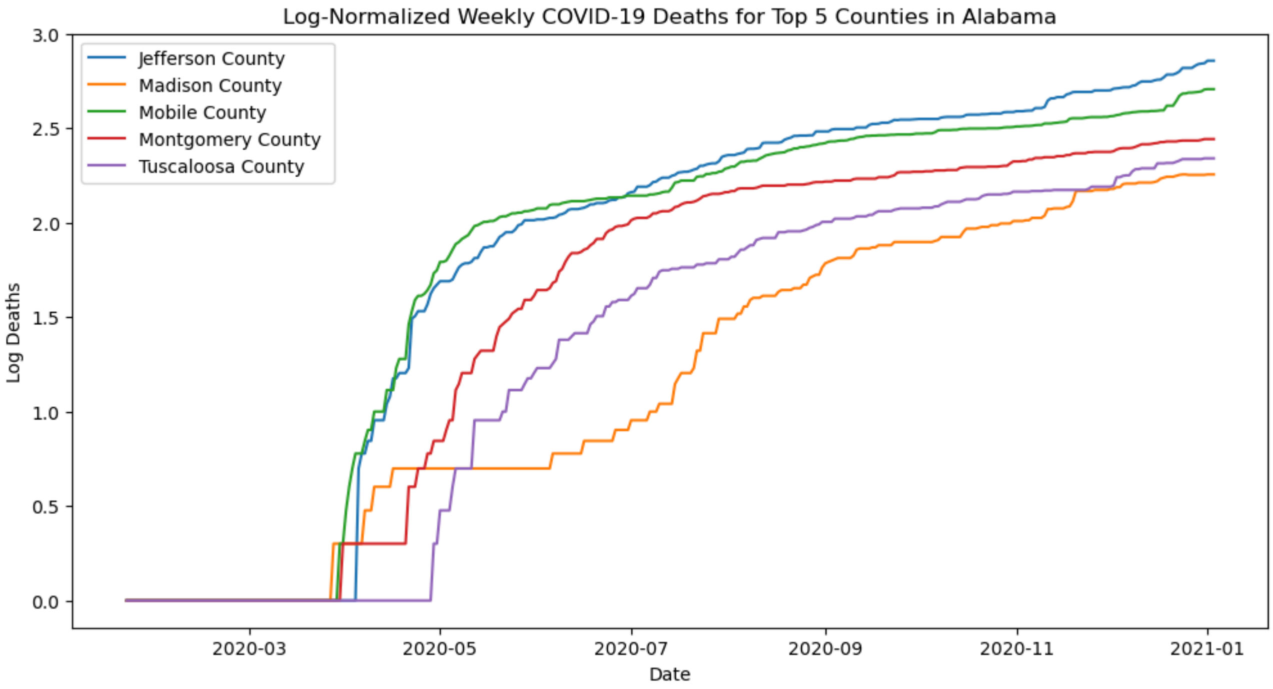




In [67]:

```
import numpy as np
# Log-normalize the cases and deaths using .loc[] to avoid the SettingWithCopyWarning
top_5_county_cases.loc[:, 'log_cases'] = np.log10(top_5_county_cases['cases'] + 1)
top_5_county_deaths.loc[:, 'log_deaths'] = np.log10(top_5_county_deaths['deaths'] + 1)

# Plot log-normalized weekly cases for the top 5 counties
plt.figure(figsize=(12, 6))
sns.lineplot(data=top_5_county_cases, x='date', y='log_cases', hue='County Name')
plt.title(f'Log-Normalized Weekly COVID-19 Cases for Top 5 Counties in Alabama')
plt.xlabel('Date')
plt.ylabel('Log Cases')
plt.legend()
plt.show()

# Plot log-normalized weekly deaths for the top 5 counties
plt.figure(figsize=(12, 6))
sns.lineplot(data=top_5_county_deaths, x='date', y='log_deaths', hue='County Name')
plt.title(f'Log-Normalized Weekly COVID-19 Deaths for Top 5 Counties in Alabama')
plt.xlabel('Date')
plt.ylabel('Log Deaths')
plt.legend()
plt.show()
```





The COVID-19 peaks in Alabama's top counties, like Jefferson and Mobile, were mainly becayse of their larger populations and urban settings, which made it easier for the virus to spread. The spikes in cases and deaths around May 2023 suggest that public health measures and vaccines might have taken longer to have an effect in these areas. While all the counties followed the same general trend as the state, the number of cases and deaths varied depending on how big and crowded the counties are. Overall, these counties’ trends fit in with what was happening across Alabama at the time.