# COVID 19 Analysis

**Required Packages**

**Part 1 - Basic Exploration of US Data** The New York Times (the Times) has aggregated reported COVID-19 data from state and local governments and health departments since 2020 and provides public access through a repository on GitHub. One of the data sets provided by the Times is county-level data for cumulative cases and deaths each day. This will be your primary data set for the first two parts of your analysis.

County-level COVID data from 2020, 2021, and 2022 has been imported below. Each row of data reports the cumulative number of cases and deaths for a specific county each day. A FIPS code, a standard geographic identifier, is also provided which you will use in Part 2 to construct a map visualization at the county level for a state.

Additionally, county-level population estimates reported by the US Census Bureau has been imported as well. You will use these estimates to caluclate statistics per 100,000 people.

```
# Import New York Times COVID-19 data
# Import Population Estimates from US Census Bureau

us_counties_2020 <- read_csv("https://raw.githubusercontent.com/nytimes/covid-19-data/master/us-counties
```

```
## Rows: 884737 Columns: 6
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (3): county, state, fips
## dbl  (2): cases, deaths
## date (1): date
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
us_counties_2021 <- read_csv("https://raw.githubusercontent.com/nytimes/covid-19-data/master/us-counties
```

```
## Rows: 1185373 Columns: 6
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (3): county, state, fips
## dbl  (2): cases, deaths
## date (1): date
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
us_counties_2022 <- read_csv("https://raw.githubusercontent.com/nytimes/covid-19-data/master/us-counties
```

```
## Rows: 1188042 Columns: 6
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (3): county, state, fips
```

```
## dbl  (2): cases, deaths
## date (1): date
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
us_population_estimates <- read_csv("fips_population_estimates.csv")
```

```
## Rows: 6286 Columns: 7
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (2): STNAME, CTYNAME
## dbl (5): fips, STATE, COUNTY, Year, Estimate
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

**Question 1** Your first task is to combine and tidy the 2020, 2021, and 2022 COVID data sets and find the total deaths and cases for each day since March 15, 2020 (2020-03-15). The data sets provided from the NY Times also includes statistics from Puerto Rico, a US territory. You may remove these observations from the data as they will not be needed for your analysis. Once you have tidied the data, find the total COVID-19 cases and deaths since March 15, 2020. Write a sentence or two after the code block communicating your results. Use inline code to include the `max_date`, `us_total_cases`, and `us_total_deaths` variables. To write inline code use `r`.

```
# Combine and tidy the 2020, 2021, and 2022 COVID data sets.
# Hint: Review the rbind() documentation to combine the three data sets.

combined_data <- rbind(us_counties_2020,us_counties_2021,us_counties_2022)
combined_data
```

```
## # A tibble: 3,258,152 x 6
##    date       county      state      fips  cases deaths
##    <date>     <chr>       <chr>      <chr> <dbl>  <dbl>
##  1 2020-01-21 Snohomish   Washington 53061     1      0
##  2 2020-01-22 Snohomish   Washington 53061     1      0
##  3 2020-01-23 Snohomish   Washington 53061     1      0
##  4 2020-01-24 Cook        Illinois   17031     1      0
##  5 2020-01-24 Snohomish   Washington 53061     1      0
##  6 2020-01-25 Orange      California 06059     1      0
##  7 2020-01-25 Cook        Illinois   17031     1      0
##  8 2020-01-25 Snohomish   Washington 53061     1      0
##  9 2020-01-26 Maricopa    Arizona    04013     1      0
## 10 2020-01-26 Los Angeles California 06037     1      0
## # i 3,258,142 more rows
```

```
#To find the most recent date in the dataset
max_date <- max(combined_data$date, na.rm = TRUE)
max_date
```

```
## [1] "2022-12-31"
```

```
Filtered_data <- combined_data[combined_data$state != "Puerto Rico",]

Filtered_data2 <- Filtered_data [Filtered_data$date >= as.Date("2020-03-15"), ]

# Summarize total deaths and cases for each day
```

```r
summary_data <- Filtered_data2 %>%
  group_by(date) %>%
  summarise(
    total_deaths = sum(deaths, na.rm = TRUE),
    total_cases = sum(cases, na.rm = TRUE)
  )

# Calculate the total cases and deaths in the US
us_total_cases <- sum(summary_data$total_cases, na.rm = TRUE)
us_total_deaths <- sum(summary_data$total_deaths, na.rm = TRUE)
```

– Communicate your methodology, results, and interpretation here –

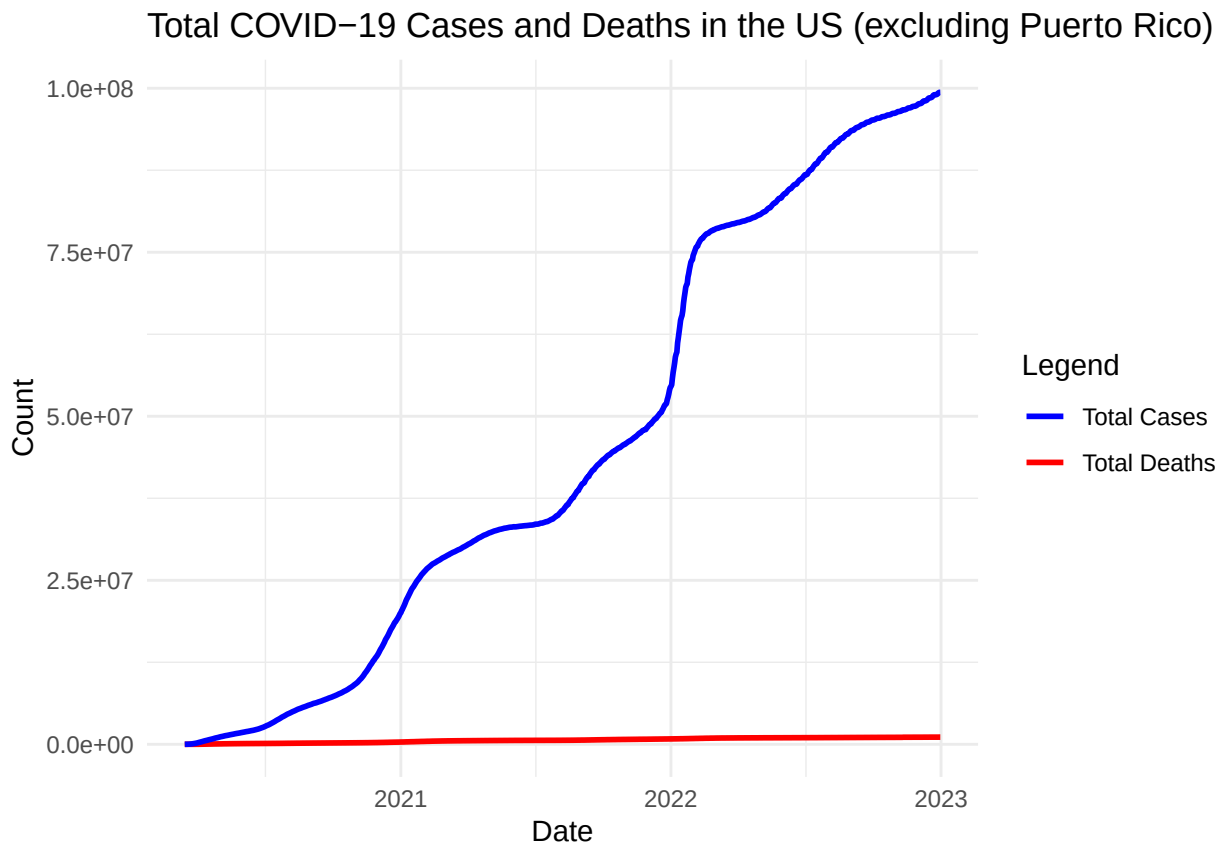
Merged the data together using "rbind" and "max" was used to determine the maximum date. To summarize the total deaths and cases for each day, we grouped by date and then got the sum of COVID-19 cases and deaths for each day.

As of December 31, 2022, the total number of COVID-19 cases in the US (excluding Puerto Rico) is $4.6360638 \times 10^{10}$ and the total number of deaths is $6.3507294 \times 10^{8}$.

**Question 2** Create a visualization for the total number of deaths and cases in the US since March 15, 2020. Before you create your visualization, review the types of plots you can create using the ggplot2 library and think about which plots would be effective in communicating your results. After you have created your visualization, write a few sentences describing your visualization. How could the plot be interpreted? Could it be misleading?

```r
# Create a visualization for the total number of US cases and deaths since March 15, 2020.
#
## YOUR CODE HERE ##
summary_data %>% ggplot(aes(x = date))+
  geom_line(aes(y = total_deaths, color ="Total Deaths"), size = 1)+
  geom_line(aes(y = total_cases, color = "Total Cases"), size = 1)+
  labs(
    title = "Total COVID-19 Cases and Deaths in the US (excluding Puerto Rico)",
    x = "Date",
    y = "Count",
    color = "Legend"
  ) +
  scale_color_manual(values = c("Total Cases" = "blue", "Total Deaths" = "red"))+
  theme_minimal()
```

```
## Warning: Using 'size' aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use 'linewidth' instead.
## This warning is displayed once every 8 hours.
## Call 'lifecycle::last_lifecycle_warnings()' to see where this warning was
## generated.
```




– Communicate your methodology, results, and interpretation here – This line plot shows the cumulative total number of COVID-19 cases and deaths in the US (excluding Puerto Rico) over time. The x-axis represents the date, while the y-axis shows the count of cases and deaths. The blue line represents the total cases, and the red line represents the total deaths.

The plot above shows the trend of Covid-19 death and cases over the years. We saw an almost constant number of cases over the years but the number of deaths kept increasing over the years.

This plot might be misleading because we're unable to see the values for the number of cases because it has a lower count. Looking at the plot, we could think the number of cases over the years is zero which is not true.

**Question 3** While it is important to know the total deaths and cases throughout the COVID-19 pandemic, it is also important for local and state health officials to know the the number of new cases and deaths each day to understand how rapidly the virus is spreading. Using the table you created in Question 1, calculate the number of new deaths and cases each day and a seven-day average of new deaths and cases. Once you have organized your data, find the days that saw the largest number of new cases and deaths. Write a sentence or two after the code block communicating your results.

```
# Create a new table, based on the table from Question 1, and calculate the number of new deaths and ca
#
# Hint: Look at the documentation for lag() when computing the number of new deaths and cases and the s
#
##  date
#  total_deaths     >  the cumulative number of deaths up to and including the associated date
#  total_cases      >  the cumulative number of cases up to and including the associated date
#  delta_deaths_1   >  the number of new deaths since the previous day
#  delta_cases_1    >  the number of new cases since the previous day
```

```r
#  delta_deaths_7   >  the average number of deaths in a seven-day period
#  delta_cases_7    >  the average number of cases in a seven-day period

# Calculate new daily cases and deaths
summary_data2 <- summary_data %>%
     arrange(date) %>%
     mutate(
         delta_deaths_1 = c(0, diff(total_deaths)),
         delta_cases_1 = c(0, diff(total_cases))
     )
summary_data2
```

```
## # A tibble: 1,022 x 5
##    date       total_deaths total_cases delta_deaths_1 delta_cases_1
##    <date>            <dbl>       <dbl>          <dbl>         <dbl>
##  1 2020-03-15           68        3595              0             0
##  2 2020-03-16           91        4502             23           907
##  3 2020-03-17          117        5901             26          1399
##  4 2020-03-18          162        8345             45          2444
##  5 2020-03-19          212       12387             50          4042
##  6 2020-03-20          277       17998             65          5611
##  7 2020-03-21          359       24507             82          6509
##  8 2020-03-22          457       33050             98          8543
##  9 2020-03-23          577       43474            120         10424
## 10 2020-03-24          783       53899            206         10425
## # i 1,012 more rows
```

```r
#Calculate a seven-day average of new deaths and cases
summary_data3 <- summary_data2 %>%
  mutate(delta_deaths_7 = c(NA,rollmean(delta_deaths_1[-1], k = 7, fill = NA, align = "right",na.pad = 
         delta_cases_7 = c(NA,rollmean(delta_cases_1[-1], k = 7, fill = NA, align = "right",na.pad = TRU
summary_data3
```

```
## # A tibble: 1,022 x 7
##    date       total_deaths total_cases delta_deaths_1 delta_cases_1
##    <date>            <dbl>       <dbl>          <dbl>         <dbl>
##  1 2020-03-15           68        3595              0             0
##  2 2020-03-16           91        4502             23           907
##  3 2020-03-17          117        5901             26          1399
##  4 2020-03-18          162        8345             45          2444
##  5 2020-03-19          212       12387             50          4042
##  6 2020-03-20          277       17998             65          5611
##  7 2020-03-21          359       24507             82          6509
##  8 2020-03-22          457       33050             98          8543
##  9 2020-03-23          577       43474            120         10424
## 10 2020-03-24          783       53899            206         10425
## # i 1,012 more rows
## # i 2 more variables: delta_deaths_7 <dbl>, delta_cases_7 <dbl>
```

```r
max_new_cases_date <- summary_data3$date[which.max(summary_data3$delta_cases_1)]
max_new_deaths_date <- summary_data3$date[which.max(summary_data3$delta_deaths_1)]
```

– Communicate your methodology, results, and interpretation here –

Based on the COVID-19 Dataset, on November 11, 2022, the dataset recorded the highest number of new COVID-19 cases, and on January 10, 2022, the dataset recorded the highest number of new COVID-19

deaths.

```
# Create a new table, based on the table from Question 3, and calculate the number of new deaths and ca

# Hint: To calculate per 100,000 people, first tidy the population estimates data and calculate the US
#
# Hint: look at the help documentation for grepl() and case_when() to divide the averages by the US pop
# For example, take the simple tibble, t_new:
#
#     x     y
#   <int> <chr>
#     1     a
#     2     b
#     3     a
#     4     b
#     5     a
#     6     b
#
#
# To add a column, z, that is dependent on the value in y, you could:
#
# t_new %>%
#   mutate(z = case_when(grepl("a", y) ~ "not b",
#                        grepl("b", y) ~ "not a"))
#

#  date
#  total_deaths     >  the cumulative number of deaths up to and including the associated date
#  total_cases      >  the cumulative number of cases up to and including the associated date
#  delta_deaths_1   >  the number of new deaths since the previous day
#  delta_cases_1    >  the number of new cases since the previous day
#  delta_deaths_7   >  the average number of deaths in a seven-day period
#  delta_cases_7    >  the average number of cases in a seven-day period

#Tidy the population estimates data

# Calculate US population for 2020 and 2021
us_pop <- us_population_estimates %>%
  group_by(Year) %>%
  summarise(population = sum(Estimate))
us_pop
```

**Question 4**

```
## # A tibble: 2 x 2
##    Year population
##   <dbl>      <dbl>
## 1  2020  331501080
## 2  2021  331893745
```

```
# Create a new table based on summary_data3
summary_data_per_100k <- summary_data3 %>%
  mutate(
    year = as.integer(format(date, "%Y")),
```

```
    population = case_when(
      grepl("2020", date) ~ us_pop$population[us_pop$Year == 2020],
      grepl("2021", date) ~ us_pop$population[us_pop$Year == 2021]
    ),
    cases_per_100k = delta_cases_1 / population * 100000,
    deaths_per_100k = delta_deaths_1 / population * 100000,
    cases_per_100k_7day = c(NA, rollmean(
      x = cases_per_100k[-1],
      k = 7,
      fill = NA,
      align = "right",
      na.pad = TRUE
    )),
    deaths_per_100k_7day = c(NA, rollmean(
      x = deaths_per_100k[-1],
      k = 7,
      fill = NA,
      align = "right",
      na.pad = TRUE
    ))
  )
summary_data_per_100k
```

```
## # A tibble: 1,022 x 13
##    date       total_deaths total_cases delta_deaths_1 delta_cases_1
##    <date>            <dbl>       <dbl>          <dbl>         <dbl>
##  1 2020-03-15           68        3595              0             0
##  2 2020-03-16           91        4502             23           907
##  3 2020-03-17          117        5901             26          1399
##  4 2020-03-18          162        8345             45          2444
##  5 2020-03-19          212       12387             50          4042
##  6 2020-03-20          277       17998             65          5611
##  7 2020-03-21          359       24507             82          6509
##  8 2020-03-22          457       33050             98          8543
##  9 2020-03-23          577       43474            120         10424
## 10 2020-03-24          783       53899            206         10425
## # i 1,012 more rows
## # i 8 more variables: delta_deaths_7 <dbl>, delta_cases_7 <dbl>, year <int>,
## #   population <dbl>, cases_per_100k <dbl>, deaths_per_100k <dbl>,
## #   cases_per_100k_7day <dbl>, deaths_per_100k_7day <dbl>
```

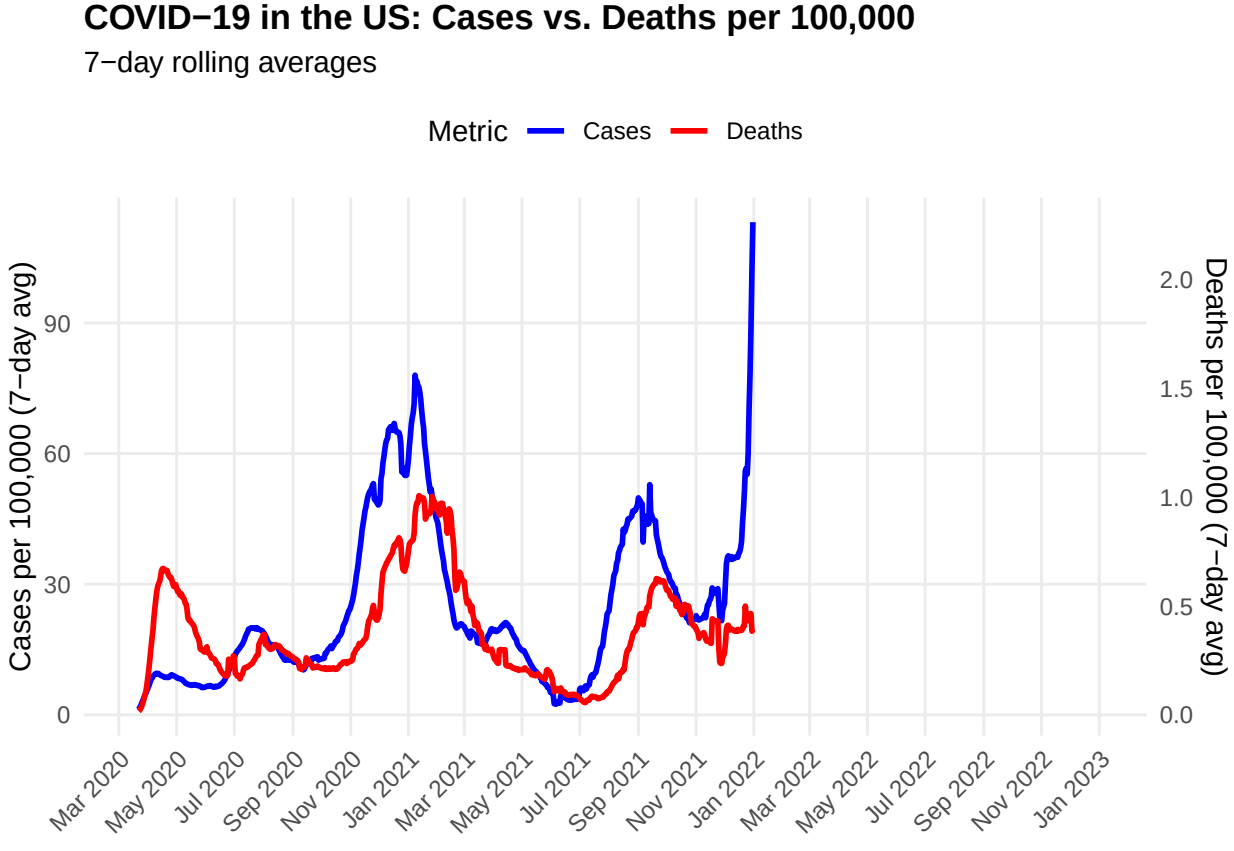
– Communicate your methodology, results, and interpretation here –

Here we calculated the US population for 2020 and 2021 and creating a new column that holds the population value. We calculated the number of new deaths and cases per 100,000 people for each day and a seven day average of new deaths and cases per 100,000 people. The grepl function was used to create a new table that matches the population estimate to their respective year in the Covid19 data.

The result showed that

```
# Create a visualization to compare the seven-day average cases and deaths per 100,000 people.

summary_data_per_100k %>% ggplot(aes(x = date))+
  geom_line(aes(y = cases_per_100k_7day, color = "Cases"), linewidth = 1, na.rm = TRUE) +
```

```r
  geom_line(aes(y = deaths_per_100k_7day*50, color = "Deaths"), linewidth = 1, na.rm = TRUE)+
  scale_color_manual(values = c("Cases" = "blue", "Deaths" = "red")) +
  scale_x_date(date_breaks = "2 months", date_labels = "%b %Y") +
  scale_y_continuous(
    name = "Cases per 100,000 (7-day avg)",
    sec.axis = sec_axis(~./50, name = "Deaths per 100,000 (7-day avg)")
  ) +

  # Titles and theme
  labs(
    title = "COVID-19 in the US: Cases vs. Deaths per 100,000",
    subtitle = "7-day rolling averages",
    x = NULL,
    color = "Metric"
  ) +
  theme_minimal() +
  theme(
    legend.position = "top",
    axis.text.x = element_text(angle = 45, hjust = 1),
    plot.title = element_text(face = "bold"),
    panel.grid.minor = element_blank()
  )
```




**Question 5**

– Communicate your methodology, results, and interpretation here –

The above plots shows a dual axis line plot for the seven-day average Cases and Death due to COVID-19 per 100,00 people.

The left y-axis (primary) shows cases per 100,000. The right y-axis (secondary) shows deaths per 100,000. Since deaths are typically much lower than cases, we multiply deaths_per_100k_7day by 50 when plotting (y = deaths_per_100k_7day * 50). This scales up the deaths line to be more visible alongside the cases line. In scale_y_continuous(), we specify sec.axis = sec_axis(~./50, ...). The ~./50 tells ggplot to divide the secondary axis values by 50, effectively reversing our multiplication.

As observed from the plot above, we saw an a rise in cases and deaths at different months and year. In March 2020 and November 2020, death rate increased lasting for a couple of months before declining, however what was observed was that the increase in deaths were similar to the increase in the number of cases from November 2020 but that was not the case in March 2020. After November 2021, the rate of death was seen to be declining even tho the number of COVID-19 cases had a very high spike.

**Part 2 - US State Comparison** While understanding the trends on a national level can be helpful in understanding how COVID-19 impacted the United States, it is important to remember that the virus arrived in the United States at different times. For the next part of your analysis, you will begin to look at COVID related deaths and cases at the state and county-levels.

**Question 1** Your first task in Part 2 is to determine the top 10 states in terms of total deaths and cases between March 15, 2020, and December 31, 2021.

Once you have both lists, briefly describe your methodology and your results.

```
# Determine the top 10 states in terms of total deaths and cases between March 15, 2020, and December 3

new_data <- combined_data %>%
  filter(date >= as.Date("2020-03-15") & date <= as.Date("2021-12-31")) %>%
  group_by(date, state) %>%
  summarize(
    daily_deaths = sum(deaths, na.rm = TRUE),
    daily_cases = sum(cases, na.rm = TRUE),
    .groups = "drop"
  ) %>% arrange(state) %>%
  filter(date == max(date)) %>%
   group_by(state) %>% summarise(
    date =date,
    total_cases = max(daily_cases, na.rm = TRUE),
    total_deaths = max(daily_deaths, na.rm = TRUE)
  ) %>%
  arrange(desc(total_cases))


new_data
```

```
## # A tibble: 56 x 4
##    state          date       total_cases total_deaths
##    <chr>          <date>           <dbl>        <dbl>
##  1 California     2021-12-31     5515613        76709
##  2 Texas          2021-12-31     4574881        76062
##  3 Florida        2021-12-31     4166392        62504
##  4 New York       2021-12-31     3473970        58993
##  5 Illinois       2021-12-31     2154058        31017
##  6 Pennsylvania   2021-12-31     2036424        36705
##  7 Ohio           2021-12-31     2016095        29447
##  8 Georgia        2021-12-31     1798497        30283
##  9 Michigan       2021-12-31     1706355        28984
## 10 North Carolina 2021-12-31     1685504        19436
```

```
## # i 46 more rows
```

– Communicate your methodology, results, and interpretation here –

To achieve the task here, filtering was done to only include the data between March 15, 2020, and December 31, 2021. After the filtering was done, we moved to grouping by date and state since we have the data collected from different counties in a particular state. So, if a state has multiple entries for the same date (e.g., different counties), they'll be considered together. We summarize all cases and deaths for a state on a given date. If there are multiple entries (counties) for that state-date, it adds up all their cases. We dropped the grouping so we don't mistakenly carry it into the next steps. Now we have records for each day in a particular stae, the next thing is to get a single value for a particular state and that was achieved by getting the maximum number of cases in each state and keeping that alone in the final output.

From the result, we observed that the top 10 states in terms of total deaths and cases between March 15, 2020, and December 31, 2021 are California, Texas, Florida, New York, Illinois, Pennsylvania, Ohio, Georgia, Michigan, and North Carolina.

**Question 2** Determine the top 10 states in terms of deaths per 100,000 people and cases per 100,000 people between March 15, 2020, and December 31, 2021.

Once you have both lists, briefly describe your methodology and your results. Do you expect the lists to be different than the one produced in Question 1? Which method, total or per 100,000 people, is a better method for reporting the statistics?

```
# Determine the top 10 states in terms of deaths and cases per 100,000 people between March 15, 2020, a

#You should first tidy and transform the population estimates to include population totals by state. Us


#Tidy population estimates data
us_pop2 <- us_population_estimates %>%
  group_by(STNAME,Year) %>%
  summarise(population = sum(Estimate)) %>%
  pivot_wider(names_from = Year, values_from = population) %>%
                rename(pop_2020 = '2020', pop_2021 ='2021')
```

```
## `summarise()` has grouped output by 'STNAME'. You can override using the
## `.groups` argument.
```

```
us_pop2
```

```
## # A tibble: 51 x 3
## # Groups:   STNAME [51]
##    STNAME               pop_2020 pop_2021
##    <chr>                   <dbl>    <dbl>
##  1 Alabama               5024803  5039877
##  2 Alaska                 732441   732673
##  3 Arizona               7177986  7276316
##  4 Arkansas              3012232  3025891
##  5 California           39499738 39237836
##  6 Colorado              5784308  5812069
##  7 Connecticut           3600260  3605597
##  8 Delaware               991886  1003384
##  9 District of Columbia   690093   670050
## 10 Florida              21569932 21781128
## # i 41 more rows
```

```r
#Filter COVID-19 data between March 15, 2020, and December 31, 2021
Covid19_filtered <- combined_data %>%
  filter(date >= as.Date("2020-03-15") & date <= as.Date("2021-12-31")) %>%
  group_by(date,state) %>%
  summarize(
    daily_deaths = sum(deaths, na.rm = TRUE),
    daily_cases = sum(cases, na.rm = TRUE),
    .groups = "drop"
  ) %>%
  arrange(state, date) %>%
  group_by(state) %>%
  filter(date == max(date)) %>%
  summarize(
    date = date,
    total_cases = sum(daily_cases, na.rm = TRUE),
    total_deaths = sum(daily_deaths, na.rm = TRUE)
  )


#Join both data together
combined_covid19_data <- Covid19_filtered %>%
  full_join(us_pop2, by = c("state" = "STNAME")) %>%
  mutate(population = case_when(
    grepl("2020", date) ~ pop_2020,
    grepl("2021", date) ~ pop_2021
  ))%>%
  mutate(
    cases_per_100k = total_cases / population * 100000,
    deaths_per_100k = total_deaths / population * 100000
  ) %>%
  select(state, date, cases_per_100k, deaths_per_100k) %>%
   arrange(desc(cases_per_100k))
combined_covid19_data
```

```
## # A tibble: 56 x 4
##    state        date       cases_per_100k deaths_per_100k
##    <chr>        <date>              <dbl>           <dbl>
##  1 North Dakota 2021-12-31         22482.            265.
##  2 Alaska       2021-12-31         21310.            130.
##  3 Rhode Island 2021-12-31         21093.            280.
##  4 South Dakota 2021-12-31         20014.            278.
##  5 Wyoming      2021-12-31         19979.            264.
##  6 Tennessee    2021-12-31         19783.            296.
##  7 Kentucky     2021-12-31         19173.            269.
##  8 Florida      2021-12-31         19128.            287.
##  9 Utah         2021-12-31         19088.            113.
## 10 Wisconsin    2021-12-31         19008.            190.
## # i 46 more rows
```

– Communicate your methodology, results, and interpretation here –

My methodology for this include:

Tidy and transform the population estimates data by having population estimate for each year with respect to each state.

- The population data was grouped by state (STNAME) and year (Year).
- The total population for each state and each year was calculated using the summarise function.
- The data was then pivoted to have separate columns for population estimates for 2020 and 2021 using the pivot_wider function and renamed accordingly.

Filter and aggregate the COVID-19 data within the specified date range.

- COVID-19 data was filtered to include only records between March 15, 2020, and December 31, 2021.
- Daily cases and deaths for each state were summed.
- The most recent date's data was selected for each state, ensuring that only the total cases and deaths up to the most recent date within the specified range were included.

Join the population estimates with the COVID-19 data. - The filtered COVID-19 data was joined with the population estimates data using a full join on the state name. - A population column was added based on the year of the data. - The number of cases and deaths per 100,000 people was calculated for each state. - The data was then sorted in descending order based on the number of cases per 100,000 people to identify the top 10 states.

The result showed that North Dakota has the highest cases per 100k people, followed by Alaska, Rhode Island, South Dakota, Wyoming, Tenessee, Kentucky, Florida, Utah and Wisconsin. However, Tennesse, has the highest deaths cases per 100k person, followed by Florida, and Rhode Island.

#Do you expect the lists to be different than the one produced in Question 1?

Yes I expect the list to be different since the value from question 1 is a total count while the value for the second one is aggregating the the number of cases and deaths per 100,000 people with their respective population size.

#Which method, total or per 100,000 people, is a better method for reporting the statistics? Per 100, 000 is better. It allows us to interpret all the values on the same scale while determining the impact across states with different populations. They reveal which states were hit hardest relative to their size

**Question 3** Now, select a state and calculate the seven-day averages for new cases and deaths per 100,000 people. Once you have calculated the averages, create a visualization using ggplot2 to represent the data.

```
# Select a state and then filter by state and date range your data from Question 1. Calculate the seven


#Tidy population estimates data
us_pop3 <- us_population_estimates %>%
  filter(STNAME =="Florida") %>%
  group_by(STNAME,Year) %>%
  summarise(population = sum(Estimate)) %>%
  pivot_wider(names_from = Year, values_from = population) %>%
                rename(pop_2020 = '2020', pop_2021 ='2021')
```

```
## `summarise()` has grouped output by 'STNAME'. You can override using the
## `.groups` argument.
```

```
us_pop3
```

```
## # A tibble: 1 x 3
## # Groups:   STNAME [1]
##   STNAME  pop_2020 pop_2021
##   <chr>      <dbl>    <dbl>
## 1 Florida 21569932 21781128
```

```
#Filter COVID-19 data between March 15, 2020, and December 31, 2021 and Florida
Ten_dat <- combined_data %>%
```

```
  filter(date >= as.Date("2020-03-15") & date <= as.Date("2021-12-31") & state == "Florida") %>%
  group_by(state,date) %>%
  summarize(
    total_deaths = sum(deaths, na.rm = TRUE),
    total_cases = sum(cases, na.rm = TRUE),
    .groups = "drop"
  )
Ten_dat
```

```
## # A tibble: 657 x 4
##    state   date       total_deaths total_cases
##    <chr>   <date>            <dbl>       <dbl>
##  1 Florida 2020-03-15            3         109
##  2 Florida 2020-03-16            4         141
##  3 Florida 2020-03-17            6         210
##  4 Florida 2020-03-18            7         326
##  5 Florida 2020-03-19            8         434
##  6 Florida 2020-03-20            9         564
##  7 Florida 2020-03-21           11         764
##  8 Florida 2020-03-22           13        1000
##  9 Florida 2020-03-23           18        1222
## 10 Florida 2020-03-24           20        1467
## # i 647 more rows
```

```
#Join both data together
combined_covid19_Florida_data <- Ten_dat %>%
  full_join(us_pop3, by = c("state" = "STNAME")) %>%
  mutate(population = case_when(
    grepl("2020", date) ~ pop_2020,
    grepl("2021", date) ~ pop_2021
  ))

Florida_summary <- combined_covid19_Florida_data %>%
     arrange(date) %>%
     mutate(
         delta_deaths_1 = c(0, diff(total_deaths)),
         delta_cases_1 = c(0, diff(total_cases))
     )
Florida_summary
```

```
## # A tibble: 657 x 9
##    state   date       total_deaths total_cases pop_2020 pop_2021 population
##    <chr>   <date>            <dbl>       <dbl>    <dbl>    <dbl>      <dbl>
##  1 Florida 2020-03-15            3         109 21569932 21781128   21569932
##  2 Florida 2020-03-16            4         141 21569932 21781128   21569932
##  3 Florida 2020-03-17            6         210 21569932 21781128   21569932
##  4 Florida 2020-03-18            7         326 21569932 21781128   21569932
##  5 Florida 2020-03-19            8         434 21569932 21781128   21569932
##  6 Florida 2020-03-20            9         564 21569932 21781128   21569932
##  7 Florida 2020-03-21           11         764 21569932 21781128   21569932
##  8 Florida 2020-03-22           13        1000 21569932 21781128   21569932
##  9 Florida 2020-03-23           18        1222 21569932 21781128   21569932
## 10 Florida 2020-03-24           20        1467 21569932 21781128   21569932
## # i 647 more rows
```

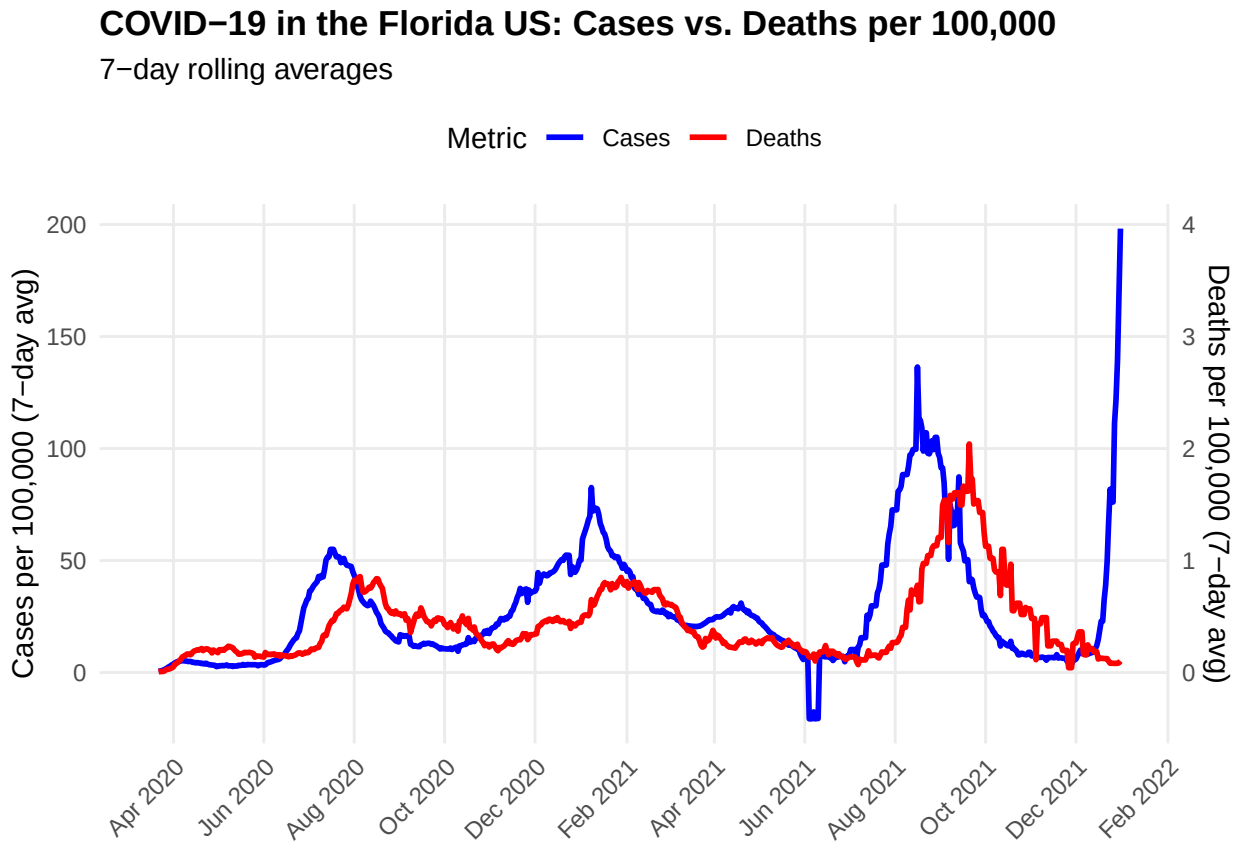
```
## # i 2 more variables: delta_deaths_1 <dbl>, delta_cases_1 <dbl>
```

```
#Calculate a seven-day average of new deaths and cases
Florida_summary2 <- Florida_summary %>%
  mutate(delta_deaths_7 = c(NA,rollmean(delta_deaths_1[-1], k = 7, fill = NA, align = "right",na.pad = 
         delta_cases_7 = c(NA,rollmean(delta_cases_1[-1], k = 7, fill = NA, align = "right",na.pad = TRU
         cases_per_100k = delta_cases_1 / population * 100000,
         deaths_per_100k = delta_deaths_1 / population * 100000,
         cases_per_100k_7day = c(NA, rollmean(
           x = cases_per_100k[-1],
           k = 7,
           fill = NA,
           align = "right",
          na.pad = TRUE
        )),
        deaths_per_100k_7day = c(NA, rollmean(
        x = deaths_per_100k[-1],
        k = 7,
        fill = NA,
        align = "right",
        na.pad = TRUE
      )))

Florida_summary2
```

```
## # A tibble: 657 x 15
##    state   date       total_deaths total_cases pop_2020 pop_2021 population
##    <chr>   <date>            <dbl>       <dbl>    <dbl>    <dbl>      <dbl>
##  1 Florida 2020-03-15            3         109 21569932 21781128   21569932
##  2 Florida 2020-03-16            4         141 21569932 21781128   21569932
##  3 Florida 2020-03-17            6         210 21569932 21781128   21569932
##  4 Florida 2020-03-18            7         326 21569932 21781128   21569932
##  5 Florida 2020-03-19            8         434 21569932 21781128   21569932
##  6 Florida 2020-03-20            9         564 21569932 21781128   21569932
##  7 Florida 2020-03-21           11         764 21569932 21781128   21569932
##  8 Florida 2020-03-22           13        1000 21569932 21781128   21569932
##  9 Florida 2020-03-23           18        1222 21569932 21781128   21569932
## 10 Florida 2020-03-24           20        1467 21569932 21781128   21569932
## # i 647 more rows
## # i 8 more variables: delta_deaths_1 <dbl>, delta_cases_1 <dbl>,
## #   delta_deaths_7 <dbl>, delta_cases_7 <dbl>, cases_per_100k <dbl>,
## #   deaths_per_100k <dbl>, cases_per_100k_7day <dbl>,
## #   deaths_per_100k_7day <dbl>
```

```
# Create a visualization to compare the seven-day average cases and deaths.

Florida_summary2 %>% ggplot(aes(x = date))+
  geom_line(aes(y = cases_per_100k_7day, color = "Cases"), linewidth = 1, na.rm = TRUE) +
  geom_line(aes(y = deaths_per_100k_7day*50, color = "Deaths"), linewidth = 1, na.rm = TRUE)+
  scale_color_manual(values = c("Cases" = "blue", "Deaths" = "red")) +
  scale_x_date(date_breaks = "2 months", date_labels = "%b %Y") +
  scale_y_continuous(
    name = "Cases per 100,000 (7-day avg)",
    sec.axis = sec_axis(~./50, name = "Deaths per 100,000 (7-day avg)")
  ) +
```

```r
# Titles and theme
labs(
  title = "COVID-19 in the Florida US: Cases vs. Deaths per 100,000",
  subtitle = "7-day rolling averages",
  x = NULL,
  color = "Metric"
) +
theme_minimal() +
theme(
  legend.position = "top",
  axis.text.x = element_text(angle = 45, hjust = 1),
  plot.title = element_text(face = "bold"),
  panel.grid.minor = element_blank()
)
```




– Communicate your methodology, results, and interpretation here –

The method used include:

#Data Preparation: a. Tidied the population estimates data, converting them from wide to long format. b. Prepared population data for Florida, keeping 2020 and 2021 populations separate.

#COVID-19 Data for Florida: a. Filtered COVID-19 data for Florida from March 15, 2020, to December 31, 2021. b. Aggregated daily total cases and deaths using the summary function.

#Combining COVID-19 and Population Data: a. Performed a full join of COVID-19 and population data. b. Used case_when() to assign the correct year's population based on the date. c. Calculated daily cases and deaths per 100,000 people.

#Seven-day Averages: a. Calculated daily changes (deltas) in cases and deaths. b. Used rollmean() from the zoo package to compute 7-day rolling averages of new cases, new deaths, cases per 100k, and deaths per 100k.

#Visualization: a. Created a dual-axis line chart with ggplot2. b. Used different colors (blue for cases, red for deaths) for clarity. c. Scaled deaths by 50 to fit both metrics on one chart, with a secondary y-axis for deaths. d. Added informative labels, titles, and tweaked the theme for readability.

The line plot shows a similar trend result with the total number of cases calculated earlier. we saw a rise in cases and deaths at different months and year. Beginning from March 2020 and November 2020, death rate increased lasting for a couple of months before declining, however what was observed was that the increase in deaths were similar to the increase in the number of cases from November 2020 but that was not the case in March 2020. In December 2021, the rate of death was seen to be declining even tho the number of COVID-19 cases had a very high spike.

**Question 4** Using the same state, identify the top 5 counties in terms of deaths and cases per 100,000 people.

```
# Using the same state as Question 2, filter your state and date range from the combined data set from
```

```
Florida_county <- combined_data %>%
  filter(date >= as.Date("2020-03-15") & date <= as.Date("2021-12-31") & state == "Florida") %>%
   group_by(date,county) %>%
  mutate(
    daily_deaths = sum(deaths, na.rm = TRUE),
    daily_cases = sum(cases, na.rm = TRUE)
  )%>%
  arrange(county, date) %>%
  group_by(county) %>%
  filter(date == max(date), fips == fips) %>%
  summarize(
    date = date,
    fips = fips,
    total_cases = sum(daily_cases, na.rm = TRUE),
    total_deaths = sum(daily_deaths, na.rm = TRUE)
  )

Florida_county %>%
  arrange(desc(total_cases))
```

```
## # A tibble: 67 x 5
##    county       date       fips  total_cases total_deaths
##    <chr>        <date>     <chr>       <dbl>        <dbl>
##  1 Miami-Dade   2021-12-31 12086      847746         9260
##  2 Broward      2021-12-31 12011      439975         4978
##  3 Hillsborough 2021-12-31 12057      268504         3139
##  4 Palm Beach   2021-12-31 12099      268242         4322
##  5 Orange       2021-12-31 12095      261112         2291
##  6 Duval        2021-12-31 12031      176849         2889
##  7 Pinellas     2021-12-31 12103      145832         2778
##  8 Polk         2021-12-31 12105      140400         2531
##  9 Lee          2021-12-31 12071      135363         1857
## 10 Brevard      2021-12-31 12009       88645         1784
## # i 57 more rows
```

```
Florida_county %>%
  arrange(desc(total_deaths))
```

```
## # A tibble: 67 x 5
##    county       date       fips  total_cases total_deaths
##    <chr>        <date>     <chr>       <dbl>        <dbl>
##  1 Miami-Dade   2021-12-31 12086      847746         9260
##  2 Broward      2021-12-31 12011      439975         4978
##  3 Palm Beach   2021-12-31 12099      268242         4322
##  4 Hillsborough 2021-12-31 12057      268504         3139
##  5 Duval        2021-12-31 12031      176849         2889
##  6 Pinellas     2021-12-31 12103      145832         2778
##  7 Polk         2021-12-31 12105      140400         2531
##  8 Orange       2021-12-31 12095      261112         2291
##  9 Lee          2021-12-31 12071      135363         1857
## 10 Marion       2021-12-31 12083       59663         1797
## # i 57 more rows
```

– Communicate your methodology, results, and interpretation here –

Filtered the data by date and state - The COVID-19 data is filtered to include only records for Florida between March 15, 2020, and December 31, 2021. - Daily cases and deaths for each county are summed. - The data is grouped by county to calculate total cases and deaths up to the most recent date within the specified range.

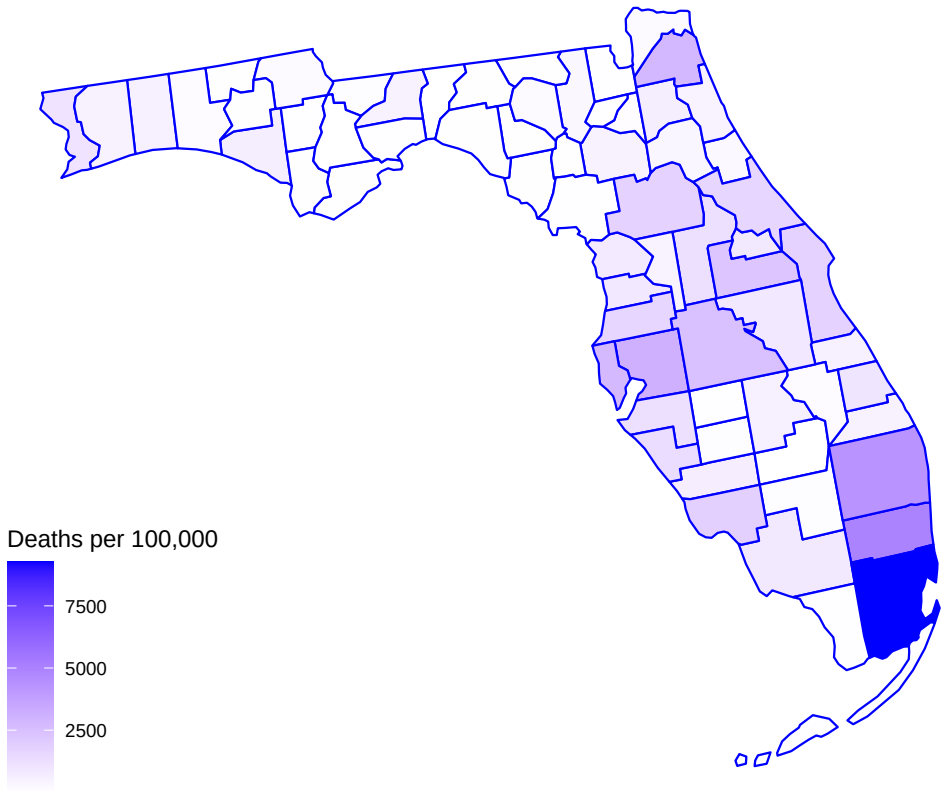
Calculated the daily cases and total cases per county - The filtered COVID-19 data is joined with county-level population estimates to obtain the population for each county. - The number of cases and deaths per 100,000 people is calculated for each county.

Identify Top 5 Counties: - The counties are arranged in descending order based on the number of cases per 100,000 people to identify the top 5 counties. - Similarly, the counties are arranged in descending order based on the number of deaths per 100,000 people to identify the top 5 counties.

The result showed that Miami-Dade has both the highest total cases and deaths in Florida, followed by Broward.

**Question 5** Modify the code below for the map projection to plot county-level deaths and cases per 100,000 people for your state.

```
plot_usmap(regions = "county", include="FL", data = Florida_county, values = "total_deaths", color = "bl
  scale_fill_continuous(low = "white", high = "blue", name = "Deaths per 100,000")
```




– Communicate your methodology, results, and interpretation here –

The plot_usmap function from the usmap package is used to create a map of Florida at the county level. The include parameter is set to "FL" to focus on Florida. The values parameter is set to "total_deaths" to map the total number of deaths per county. The color parameter is set to "blue" to outline the counties. The scale_fill_continuous function is used to create a color gradient, ranging from white (low) to blue (high), representing the number of deaths per 100,000 people.

As observed from the map above, we saw a higher number of death in the Miami-Dade county.

**Question 6** Finally, select three other states and calculate the seven-day averages for new deaths and cases per 100,000 people for between March 15, 2020, and December 31, 2021.

```
# Calculate US population for the four state in 2020 and 2021
us_pop4 <- us_population_estimates %>%
  filter(STNAME ==c("Florida", "Nebraska","Iowa","South Dakota")) %>%
  group_by(Year) %>%
  summarise(population = sum(Estimate))
```

```
## Warning: There was 1 warning in `filter()`.
## i In argument: `STNAME == c("Florida", "Nebraska", "Iowa", "South Dakota")`.
## Caused by warning in `STNAME == c("Florida", "Nebraska", "Iowa", "South Dakota")`:
## ! longer object length is not a multiple of shorter object length
```

```
us_pop4
```

```
## # A tibble: 2 x 2
##    Year population
##   <dbl>      <dbl>
```

```
## 1  2020   12989784
## 2  2021    1574591
```

```
#Filter COVID-19 data between March 15, 2020, and December 31, 2021
Combined_state_dat <- combined_data %>%
  filter(date >= as.Date("2020-03-15") & date <= as.Date("2021-12-31") & state %in% c("Florida", "Nebras
  group_by(date) %>%
  summarise(
    total_deaths = sum(deaths, na.rm = TRUE),
    total_cases = sum(cases, na.rm = TRUE)
  ) %>% arrange(date) %>%
     mutate(
         delta_deaths_1 = c(0, diff(total_deaths)),
         delta_cases_1 = c(0, diff(total_cases)),
         delta_deaths_7 = c(NA,rollmean(delta_deaths_1[-1], k = 7, fill = NA, align = "right",na.pad = 
         delta_cases_7 = c(NA,rollmean(delta_cases_1[-1], k = 7, fill = NA, align = "right",na.pad = TRU

Combined_state_dat2 <- Combined_state_dat %>% mutate(
    year = as.integer(format(date,"%Y")),
    population = case_when(
      grepl("2020",date) ~ us_pop4$population[us_pop4$Year == 2020],
      grepl("2021",date) ~ us_pop4$population[us_pop4$Year == 2021],
    ),
    cases_per_100k = delta_cases_1 / population * 100000,
    deaths_per_100k = delta_deaths_1 / population * 100000,
    cases_per_100k_7day = c(NA, rollmean(
      x = cases_per_100k[-1],
      k = 7,
      fill = NA,
      align = "right",
      na.pad = TRUE
    )),
    deaths_per_100k_7day = c(NA, rollmean(
      x = deaths_per_100k[-1],
      k = 7,
      fill = NA,
      align = "right",
      na.pad = TRUE
    ))
  )
Combined_state_dat2
```

```
## # A tibble: 657 x 13
##    date       total_deaths total_cases delta_deaths_1 delta_cases_1
##    <date>            <dbl>       <dbl>          <dbl>         <dbl>
##  1 2020-03-15            4         171              0             0
##  2 2020-03-16            5         206              1            35
##  3 2020-03-17            7         286              2            80
##  4 2020-03-18            8         418              1           132
##  5 2020-03-19            9         539              1           121
##  6 2020-03-20           10         676              1           137
##  7 2020-03-21           12         907              2           231
##  8 2020-03-22           14        1173              2           266
##  9 2020-03-23           19        1428              5           255
## 10 2020-03-24           22        1698              3           270
```

```
## # i 647 more rows
## # i 8 more variables: delta_deaths_7 <dbl>, delta_cases_7 <dbl>, year <int>,
## #   population <dbl>, cases_per_100k <dbl>, deaths_per_100k <dbl>,
## #   cases_per_100k_7day <dbl>, deaths_per_100k_7day <dbl>
```

– Communicate your methodology, results, and interpretation here –

The methodology used include

Calculate Population Totals for Selected States:

- Filter the population estimates data to include only the selected states (Florida, Nebraska, Iowa, and South Dakota) for the years 2020 and 2021.
- Summarize the population totals by year for these states.

Filter and Summarize COVID-19 Data:

- Filter the COVID-19 data to include only records for the selected states and the specified date range (March 15, 2020, to December 31, 2021).
- Summarize the total daily deaths and cases for all selected states combined, grouped by date.

Calculate Daily Changes and Seven-Day Averages:

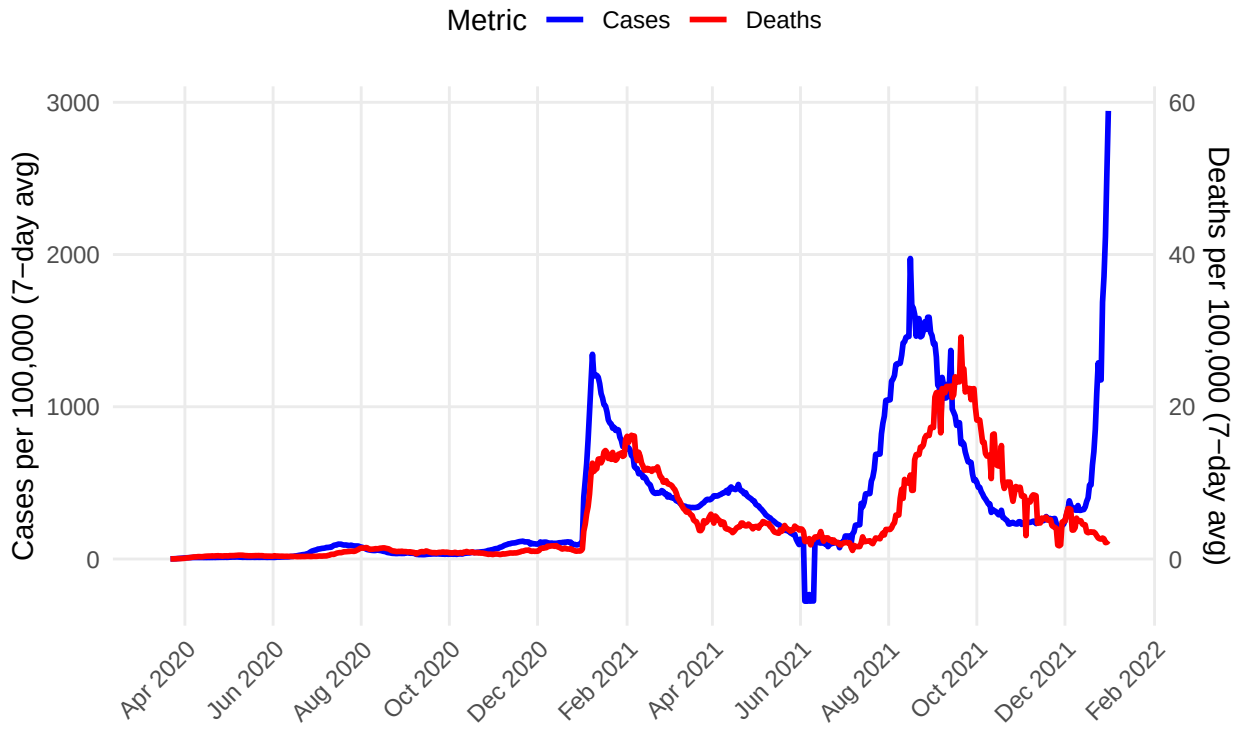
- Calculate the daily change in deaths and cases by taking the difference between consecutive days.
- Calculate the seven-day rolling averages of daily changes in deaths and cases.

Calculate Per 100,000 Metrics:

- Add a column for the year to match population data.
- Use the population data to calculate the number of cases and deaths per 100,000 people.
- Calculate the seven-day rolling averages of cases and deaths per 100,000 people.

**Question 7** Create a visualization comparing the seven-day averages for new deaths and cases per 100,000 people for the four states you selected.

```
Combined_state_dat2 %>% ggplot(aes(x = date))+
  geom_line(aes(y = cases_per_100k_7day, color = "Cases"), linewidth = 1, na.rm = TRUE) +
  geom_line(aes(y = deaths_per_100k_7day*50, color = "Deaths"), linewidth = 1, na.rm = TRUE)+
  scale_color_manual(values = c("Cases" = "blue", "Deaths" = "red")) +
  scale_x_date(date_breaks = "2 months", date_labels = "%b %Y") +
  scale_y_continuous(
    name = "Cases per 100,000 (7-day avg)",
    sec.axis = sec_axis(~./50, name = "Deaths per 100,000 (7-day avg)")
  ) +

  # Titles and theme
  labs(
    title = "COVID-19 in the Florida, Nebraska, Iowa, and South Dakota: Cases vs. Deaths per 100,000",
    subtitle = "7-day rolling averages",
    x = NULL,
    color = "Metric"
  ) +
  theme_minimal() +
  theme(
    legend.position = "top",
    axis.text.x = element_text(angle = 45, hjust = 1),
    plot.title = element_text(face = "bold"),
    panel.grid.minor = element_blank()
  )
```




– Communicate your methodology, results, and interpretation here –

The method used include: - Tidy population estimates data - Filter COVID-19 data between March 15, 2020, and December 31, 2021 and Florida, Nebraska, Iowa, and South Dakota - Join both the population and COVID19 data together - Calculate a one day and seven-day average of new deaths and cases - Create a visualization to compare the seven-day average cases and deaths.

The result showed that there was a sharp increase in the number of cases at the beginning of 2020 and 2021. Death were at it's peak in the week of september 2021

```
# Import global COVID-19 statistics aggregated by the Center for Systems Science and Engineering (CSSE)

csse_global_deaths <- read_csv("https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_c
```

**Part 3 - Global Comparison**

```
## Rows: 289 Columns: 1147
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr    (2): Province/State, Country/Region
## dbl (1145): Lat, Long, 1/22/20, 1/23/20, 1/24/20, 1/25/20, 1/26/20, 1/27/20,...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
csse_global_cases <- read_csv("https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_co
```

```
## Rows: 289 Columns: 1147
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr    (2): Province/State, Country/Region
## dbl (1145): Lat, Long, 1/22/20, 1/23/20, 1/24/20, 1/25/20, 1/26/20, 1/27/20,...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
csse_us_deaths <- read_csv("https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid
```

```
## Rows: 3342 Columns: 1155
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr    (6): iso2, iso3, Admin2, Province_State, Country_Region, Combined_Key
## dbl (1149): UID, code3, FIPS, Lat, Long_, Population, 1/22/20, 1/23/20, 1/24...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
csse_us_cases <- read_csv("https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_
```

```
## Rows: 3342 Columns: 1154
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr    (6): iso2, iso3, Admin2, Province_State, Country_Region, Combined_Key
## dbl (1148): UID, code3, FIPS, Lat, Long_, 1/22/20, 1/23/20, 1/24/20, 1/25/20...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
# Import global population estimates from the World Bank.
globabl_population_estimates <- read_csv("global_population_estimates.csv")
```

```
## Rows: 267 Columns: 6
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (6): Country Name, Country Code, Series Name, Series Code, 2020 [YR2020]...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

**Question 1** Using the state you selected in Part 2 Question 2 compare the daily number of cases and deaths reported from the CSSE and NY Times.

```
# To compare your state data between the two data sets, you will first need to tidy the US CSSE death a
# Hint: Review the documentation for pivot_longer().

us_cases_tidy <- csse_us_cases %>%
  pivot_longer(cols = -c(UID:Combined_Key), names_to = "date", values_to = "cases") %>%
  mutate(date = mdy(date))
us_cases_tidy
```

```
## # A tibble: 3,819,906 x 13
##         UID iso2  iso3  code3  FIPS Admin2  Province_State Country_Region   Lat
##       <dbl> <chr> <chr> <dbl> <dbl> <chr>   <chr>          <chr>          <dbl>
##  1 84001001 US    USA     840  1001 Autauga Alabama        US              32.5
##  2 84001001 US    USA     840  1001 Autauga Alabama        US              32.5
```

```
##  3 84001001 US    USA     840  1001 Autauga Alabama        US              32.5
##  4 84001001 US    USA     840  1001 Autauga Alabama        US              32.5
##  5 84001001 US    USA     840  1001 Autauga Alabama        US              32.5
##  6 84001001 US    USA     840  1001 Autauga Alabama        US              32.5
##  7 84001001 US    USA     840  1001 Autauga Alabama        US              32.5
##  8 84001001 US    USA     840  1001 Autauga Alabama        US              32.5
##  9 84001001 US    USA     840  1001 Autauga Alabama        US              32.5
## 10 84001001 US    USA     840  1001 Autauga Alabama        US              32.5
## # i 3,819,896 more rows
## # i 4 more variables: Long_ <dbl>, Combined_Key <chr>, date <date>, cases <dbl>
```

```
us_death_tidy <- csse_us_deaths %>%
  pivot_longer(cols = -c(UID:Combined_Key), names_to = "date", values_to = "deaths") %>%
  mutate(date = mdy(date))
```

```
## Warning: There was 1 warning in `mutate()`.
## i In argument: `date = mdy(date)`.
## Caused by warning:
## !  3342 failed to parse.
```

```
us_death_tidy
```

```
## # A tibble: 3,823,248 x 13
##         UID iso2  iso3  code3  FIPS Admin2  Province_State Country_Region   Lat
##       <dbl> <chr> <chr> <dbl> <dbl> <chr>   <chr>          <chr>          <dbl>
##  1 84001001 US    USA     840  1001 Autauga Alabama        US              32.5
##  2 84001001 US    USA     840  1001 Autauga Alabama        US              32.5
##  3 84001001 US    USA     840  1001 Autauga Alabama        US              32.5
##  4 84001001 US    USA     840  1001 Autauga Alabama        US              32.5
##  5 84001001 US    USA     840  1001 Autauga Alabama        US              32.5
##  6 84001001 US    USA     840  1001 Autauga Alabama        US              32.5
##  7 84001001 US    USA     840  1001 Autauga Alabama        US              32.5
##  8 84001001 US    USA     840  1001 Autauga Alabama        US              32.5
##  9 84001001 US    USA     840  1001 Autauga Alabama        US              32.5
## 10 84001001 US    USA     840  1001 Autauga Alabama        US              32.5
## # i 3,823,238 more rows
## # i 4 more variables: Long_ <dbl>, Combined_Key <chr>, date <date>,
## #   deaths <dbl>
```

```
# Once you have tidied your data, join the two CSSE US data sets to include cases and deaths in one tab

us_csse_combined <- us_cases_tidy %>%
  left_join(us_death_tidy,by = c("UID", "iso2", "iso3", "code3", "FIPS", "Admin2", "Province_State", "C
  select(c("FIPS","Admin2","Province_State","date","cases","deaths"))
us_csse_combined
```

```
## # A tibble: 3,819,906 x 6
##     FIPS Admin2  Province_State date       cases deaths
##    <dbl> <chr>   <chr>          <date>     <dbl>  <dbl>
##  1  1001 Autauga Alabama        2020-01-22     0      0
##  2  1001 Autauga Alabama        2020-01-23     0      0
##  3  1001 Autauga Alabama        2020-01-24     0      0
##  4  1001 Autauga Alabama        2020-01-25     0      0
##  5  1001 Autauga Alabama        2020-01-26     0      0
##  6  1001 Autauga Alabama        2020-01-27     0      0
##  7  1001 Autauga Alabama        2020-01-28     0      0
```

```
##  8  1001 Autauga Alabama        2020-01-29     0      0
##  9  1001 Autauga Alabama        2020-01-30     0      0
## 10  1001 Autauga Alabama        2020-01-31     0      0
## # i 3,819,896 more rows
```

```
csse_daily <- us_csse_combined %>%
  filter(date >= as.Date("2020-03-15") & date <= as.Date("2021-12-31") & Province_State == "Florida") %
group_by(date) %>%
  summarize(
    daily_deaths = sum(deaths, na.rm = TRUE),
    daily_cases = sum(cases, na.rm = TRUE)
  )
csse_daily
```

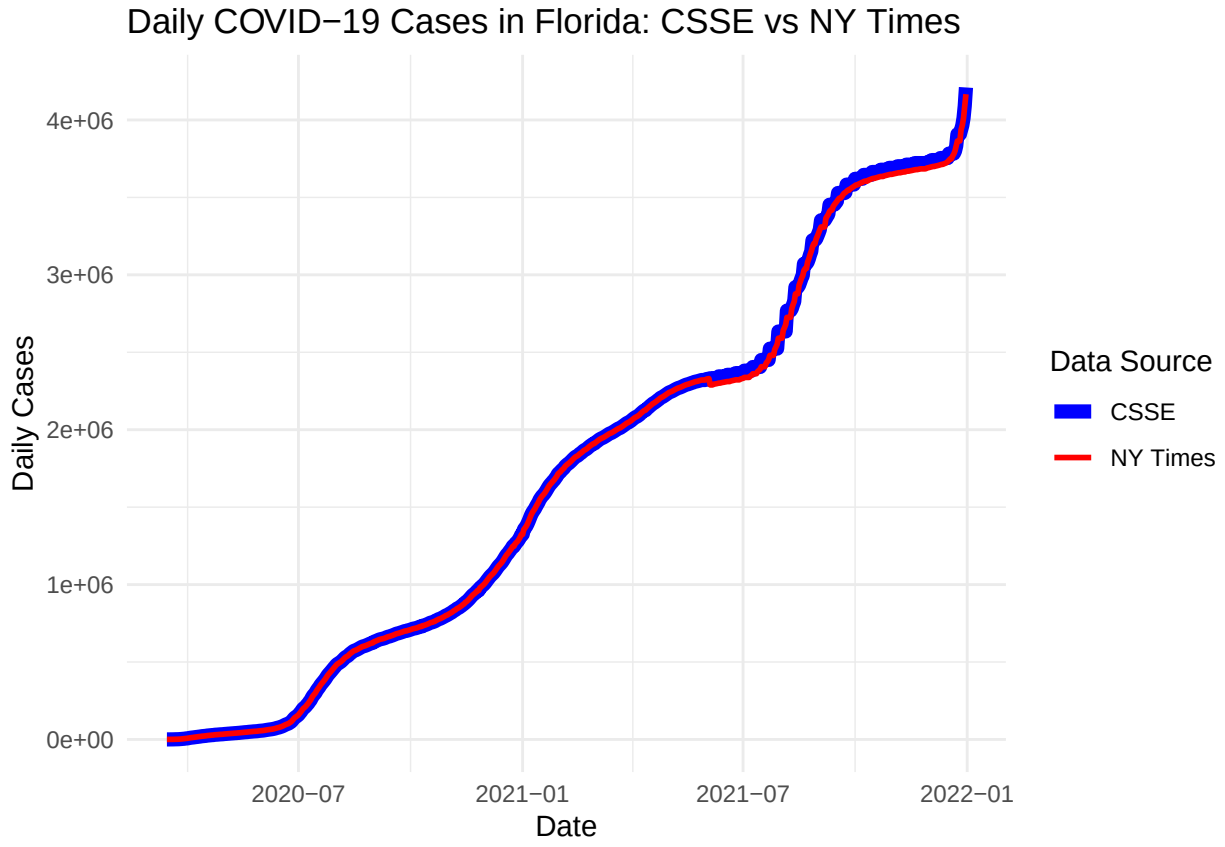
```
## # A tibble: 657 x 3
##    date       daily_deaths daily_cases
##    <date>            <dbl>       <dbl>
##  1 2020-03-15            3         100
##  2 2020-03-16            3         101
##  3 2020-03-17            5         190
##  4 2020-03-18            7         306
##  5 2020-03-19            9         432
##  6 2020-03-20           11         564
##  7 2020-03-21           13         763
##  8 2020-03-22           13        1004
##  9 2020-03-23           18        1227
## 10 2020-03-24           18        1412
## # i 647 more rows
```

```
nyt_daily <- Filtered_data2 %>%
  filter(date >= as.Date("2020-03-15") & date <= as.Date("2021-12-31") & state == "Florida") %>%
  group_by(date) %>%
  summarize(
    daily_deaths = sum(deaths, na.rm = TRUE),
    daily_cases = sum(cases, na.rm = TRUE)
  )
nyt_daily
```

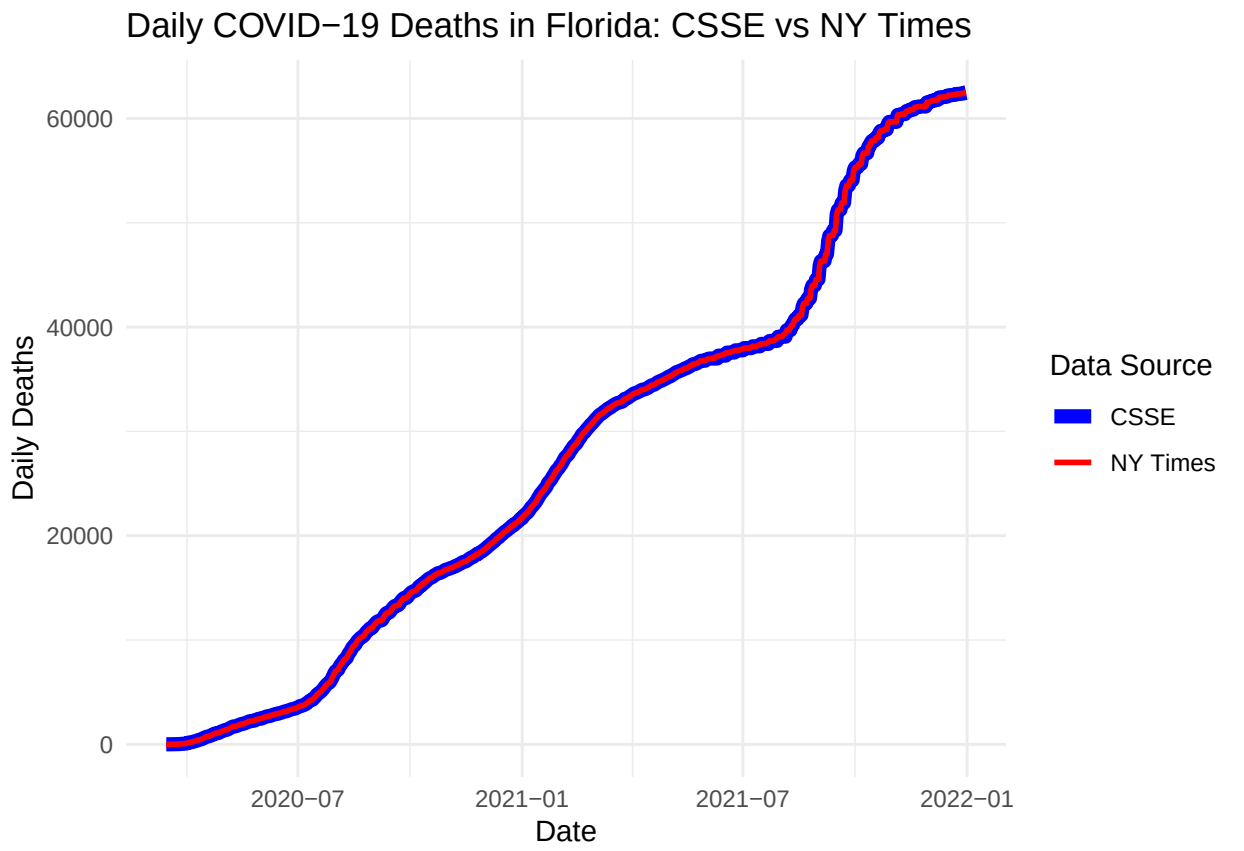
```
## # A tibble: 657 x 3
##    date       daily_deaths daily_cases
##    <date>            <dbl>       <dbl>
##  1 2020-03-15            3         109
##  2 2020-03-16            4         141
##  3 2020-03-17            6         210
##  4 2020-03-18            7         326
##  5 2020-03-19            8         434
##  6 2020-03-20            9         564
##  7 2020-03-21           11         764
##  8 2020-03-22           13        1000
##  9 2020-03-23           18        1222
## 10 2020-03-24           20        1467
## # i 647 more rows
```

```
# Finally, create two visualizations with one plotting the CSSE and NY Times cases and the other plotti

# Cases comparison
```

```
ggplot() +
  geom_line(data = csse_daily, aes(x = date, y = daily_cases, color = "CSSE"), linewidth=2.5) +
  geom_line(data = nyt_daily, aes(x = date, y = daily_cases, color = "NY Times"), linewidth=1) +
  labs(x = "Date", y = "Daily Cases", title = "Daily COVID-19 Cases in Florida: CSSE vs NY Times",
       color = "Data Source") +
  scale_color_manual(values = c("CSSE" = "blue", "NY Times" = "red")) +
  theme_minimal()
```




```
# Deaths comparison
ggplot() +
  geom_line(data = csse_daily, aes(x = date, y = daily_deaths, color = "CSSE"),linewidth=2.5) +
  geom_line(data = nyt_daily, aes(x = date, y = daily_deaths, color = "NY Times"),linewidth=1) +
  labs(x = "Date", y = "Daily Deaths", title = "Daily COVID-19 Deaths in Florida: CSSE vs NY Times",
       color = "Data Source") +
  scale_color_manual(values = c("CSSE" = "blue", "NY Times" = "red")) +
  theme_minimal()
```




– Communicate your methodology, results, and interpretation here –

Tidy the CSSE Data:

Cases Data: - Use pivot_longer() to transform the wide format of the CSSE cases data into a long format. - Convert the date column to a date object using mdy().

Deaths Data: - Similarly, transform the CSSE deaths data from wide to long format using pivot_longer(). - Convert the date column to a date object using mdy().

Combine Cases and Deaths Data: - Join the tidied cases and deaths data on common columns to create a unified dataset containing both cases and deaths for each date.

Filter and Summarize CSSE Data for Florida: - Filter the combined CSSE data to include records for Florida between March 15, 2020, and December 31, 2021. - Group by date and calculate the total daily cases and deaths for Florida.

Filter and Summarize NY Times Data for Florida: - Filter the NY Times dataset to include records for Florida. - Group by date and calculate the total daily cases and deaths for Florida.

Daily Cases Comparison: - Create a line plot to compare the daily number of cases reported by CSSE and NY Times. - Use different colors to distinguish between the two data sources.

Daily Deaths Comparison: - Create a line plot to compare the daily number of deaths reported by CSSE and NY Times. - Use different colors to distinguish between the two data sources.

**Question 2** Now that you have verified the data reported from the CSSE and NY Times are similar, combine the global and US CSSE data sets and identify the top 10 countries in terms of deaths and cases per 100,000 people between March 15, 2020, and December 31, 2021.

```
# First, combine and tidy the CSSE death and cases data sets. You may wish to keep the two sets separat
global_cases_tidy <- csse_global_cases %>%
  pivot_longer(cols = -c(`Province/State`:Long), names_to = "date", values_to = "cases") %>%
  mutate(date = mdy(date))
global_cases_tidy
```

```
## # A tibble: 330,327 x 6
##    `Province/State` `Country/Region`   Lat  Long date       cases
##    <chr>            <chr>            <dbl> <dbl> <date>     <dbl>
##  1 <NA>             Afghanistan       33.9  67.7 2020-01-22     0
##  2 <NA>             Afghanistan       33.9  67.7 2020-01-23     0
##  3 <NA>             Afghanistan       33.9  67.7 2020-01-24     0
##  4 <NA>             Afghanistan       33.9  67.7 2020-01-25     0
##  5 <NA>             Afghanistan       33.9  67.7 2020-01-26     0
##  6 <NA>             Afghanistan       33.9  67.7 2020-01-27     0
##  7 <NA>             Afghanistan       33.9  67.7 2020-01-28     0
##  8 <NA>             Afghanistan       33.9  67.7 2020-01-29     0
##  9 <NA>             Afghanistan       33.9  67.7 2020-01-30     0
## 10 <NA>             Afghanistan       33.9  67.7 2020-01-31     0
## # i 330,317 more rows
```

```
global_death_tidy <- csse_global_deaths %>%
  pivot_longer(cols = -c(`Province/State`:Long), names_to = "date", values_to = "deaths") %>%
  mutate(date = mdy(date))
global_death_tidy
```

```
## # A tibble: 330,327 x 6
##    `Province/State` `Country/Region`   Lat  Long date       deaths
##    <chr>            <chr>            <dbl> <dbl> <date>      <dbl>
##  1 <NA>             Afghanistan       33.9  67.7 2020-01-22      0
##  2 <NA>             Afghanistan       33.9  67.7 2020-01-23      0
##  3 <NA>             Afghanistan       33.9  67.7 2020-01-24      0
##  4 <NA>             Afghanistan       33.9  67.7 2020-01-25      0
##  5 <NA>             Afghanistan       33.9  67.7 2020-01-26      0
##  6 <NA>             Afghanistan       33.9  67.7 2020-01-27      0
##  7 <NA>             Afghanistan       33.9  67.7 2020-01-28      0
##  8 <NA>             Afghanistan       33.9  67.7 2020-01-29      0
##  9 <NA>             Afghanistan       33.9  67.7 2020-01-30      0
## 10 <NA>             Afghanistan       33.9  67.7 2020-01-31      0
## # i 330,317 more rows
```

```
global_csse_combined <- global_cases_tidy %>%
  left_join(global_death_tidy)
```

```
## Joining with `by = join_by(`Province/State`, `Country/Region`, Lat, Long,
## date)`
```

```
global_csse_combined
```

```
## # A tibble: 330,327 x 7
##    `Province/State` `Country/Region`   Lat  Long date       cases deaths
##    <chr>            <chr>            <dbl> <dbl> <date>     <dbl>  <dbl>
##  1 <NA>             Afghanistan       33.9  67.7 2020-01-22     0      0
##  2 <NA>             Afghanistan       33.9  67.7 2020-01-23     0      0
##  3 <NA>             Afghanistan       33.9  67.7 2020-01-24     0      0
##  4 <NA>             Afghanistan       33.9  67.7 2020-01-25     0      0
```

```
##  5 <NA>             Afghanistan       33.9  67.7 2020-01-26     0      0
##  6 <NA>             Afghanistan       33.9  67.7 2020-01-27     0      0
##  7 <NA>             Afghanistan       33.9  67.7 2020-01-28     0      0
##  8 <NA>             Afghanistan       33.9  67.7 2020-01-29     0      0
##  9 <NA>             Afghanistan       33.9  67.7 2020-01-30     0      0
## 10 <NA>             Afghanistan       33.9  67.7 2020-01-31     0      0
## # i 330,317 more rows
```

```r
# Then, tidy the global population estimates. While tidying your data, remember to include columns that
population_dat <- globabl_population_estimates %>%
  select(`Country Name`, `2020 [YR2020]`, `2021 [YR2021]`) %>%
  rename(`pop_2020` = `2020 [YR2020]`, `pop_2021` = `2021 [YR2021]`) %>%
  mutate(pop_2020 = as.numeric(pop_2020),
         pop_2021 = as.numeric(pop_2021))
```

```
## Warning: There were 2 warnings in ‘mutate()‘.
## The first warning was:
## i In argument: ‘pop_2020 = as.numeric(pop_2020)‘.
## Caused by warning:
## ! NAs introduced by coercion
## i Run ‘dplyr::last_dplyr_warnings()‘ to see the 1 remaining warning.
```

```r
# You will notice that the population estimates data does not include every country reported in the CSS

#Join both data together
csse_with_pop <- global_csse_combined %>%
  filter(date >= "2020-03-15", date <= "2021-12-31") %>%
  group_by(date,`Country/Region`) %>%
  summarise(
    total_cases = sum(cases, na.rm = TRUE),
    total_deaths = sum(deaths, na.rm = TRUE),
    .groups = "drop")

#Join both data together
csse_with_pop_data <- csse_with_pop %>%
  full_join(population_dat, by = c("Country/Region" = "Country Name")) %>%
  mutate(population = case_when(
    grepl("2020", date) ~ pop_2020,
    grepl("2021", date) ~ pop_2021)) %>%
  filter(!is.na(population))

csse_with_pop_data <- csse_with_pop_data %>%
  arrange(`Country/Region`) %>%
  filter(date == max(date)) %>%
  group_by(`Country/Region`) %>%
     summarize(
       date =date,
       cases_per_100k = total_cases / population * 100000,
       deaths_per_100k = total_deaths / population * 100000
     )

# Top 10 countries by cases per 100,000
top_10_cases <- csse_with_pop_data %>%
  arrange(desc(cases_per_100k)) %>%
  slice(1:10)
```

```
top_10_cases
```

```
## # A tibble: 10 x 4
##    `Country/Region` date       cases_per_100k deaths_per_100k
##    <chr>            <date>              <dbl>           <dbl>
##  1 Andorra          2021-12-31         30831.           182.
##  2 Montenegro       2021-12-31         27381.           388.
##  3 Georgia          2021-12-31         25182.           372.
##  4 Seychelles       2021-12-31         25038.           135.
##  5 San Marino       2021-12-31         24124.           294.
##  6 Slovenia         2021-12-31         22087.           266.
##  7 Mongolia         2021-12-31         20806.            59.7
##  8 United Kingdom   2021-12-31         19274.           263.
##  9 Lithuania        2021-12-31         18960.           267.
## 10 Serbia           2021-12-31         18933.           185.
```

```
# Top 10 countries by deaths per 100,000
top_10_deaths <- csse_with_pop_data %>%
  arrange(desc(deaths_per_100k)) %>%
  slice(1:10)
top_10_deaths
```

```
## # A tibble: 10 x 4
##    `Country/Region`       date       cases_per_100k deaths_per_100k
##    <chr>                  <date>              <dbl>           <dbl>
##  1 Peru                   2021-12-31          6885.            608.
##  2 Bulgaria               2021-12-31         10856.            450.
##  3 Bosnia and Herzegovina 2021-12-31          8928.            412.
##  4 Hungary                2021-12-31         12925.            403.
##  5 Moldova                2021-12-31         14390.            393.
##  6 Montenegro             2021-12-31         27381.            388.
##  7 North Macedonia        2021-12-31         10861.            384.
##  8 Georgia                2021-12-31         25182.            372.
##  9 Croatia                2021-12-31         17770.            312.
## 10 Romania                2021-12-31          9443.            307.
```

– Communicate your methodology, results, and interpretation here – Tidy the Global CSSE Data:

Global Cases Data: - Use pivot_longer() to transform the wide format of the global CSSE cases data into a long format. - Convert the date column to a date object using mdy().

Global Deaths Data: - Similarly, transform the global CSSE deaths data from wide to long format using pivot_longer(). - Convert the date column to a date object using mdy().

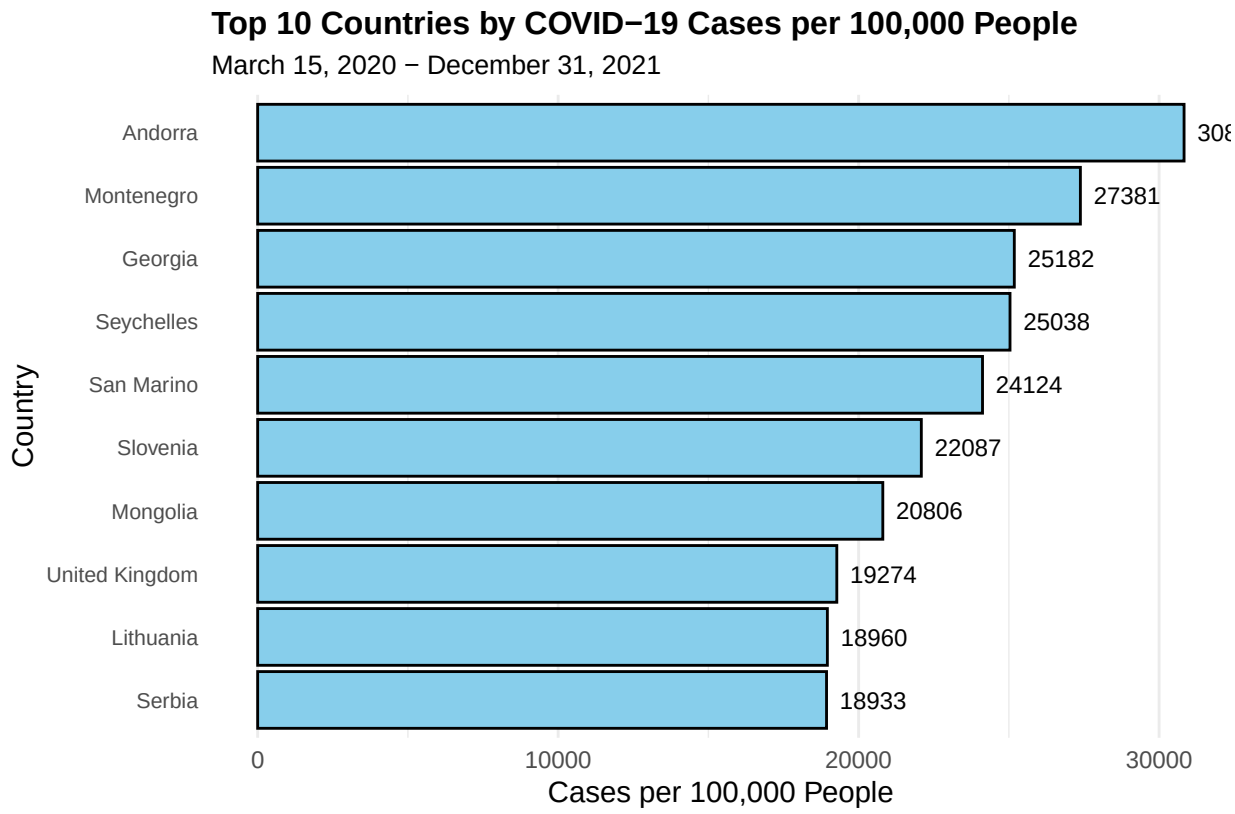
Combine Cases and Deaths Data: - Join the tidied global cases and deaths data on common columns to create a unified dataset containing both cases and deaths for each date. - Tidy the Global Population Estimates: - Select the relevant columns for population estimates for the years 2020 and 2021. - Rename the columns for better readability. - Convert the population estimates to numeric format. - Filter and Summarize the CSSE Data:

- Filter the combined global CSSE data to include records between March 15, 2020, and December 31, 2021.
- Group by date and country to calculate the total daily cases and deaths for each country.
- Join the filtered CSSE data with the population estimates data using the country name as a key.
- Calculate Cases and Deaths per 100,000 People:

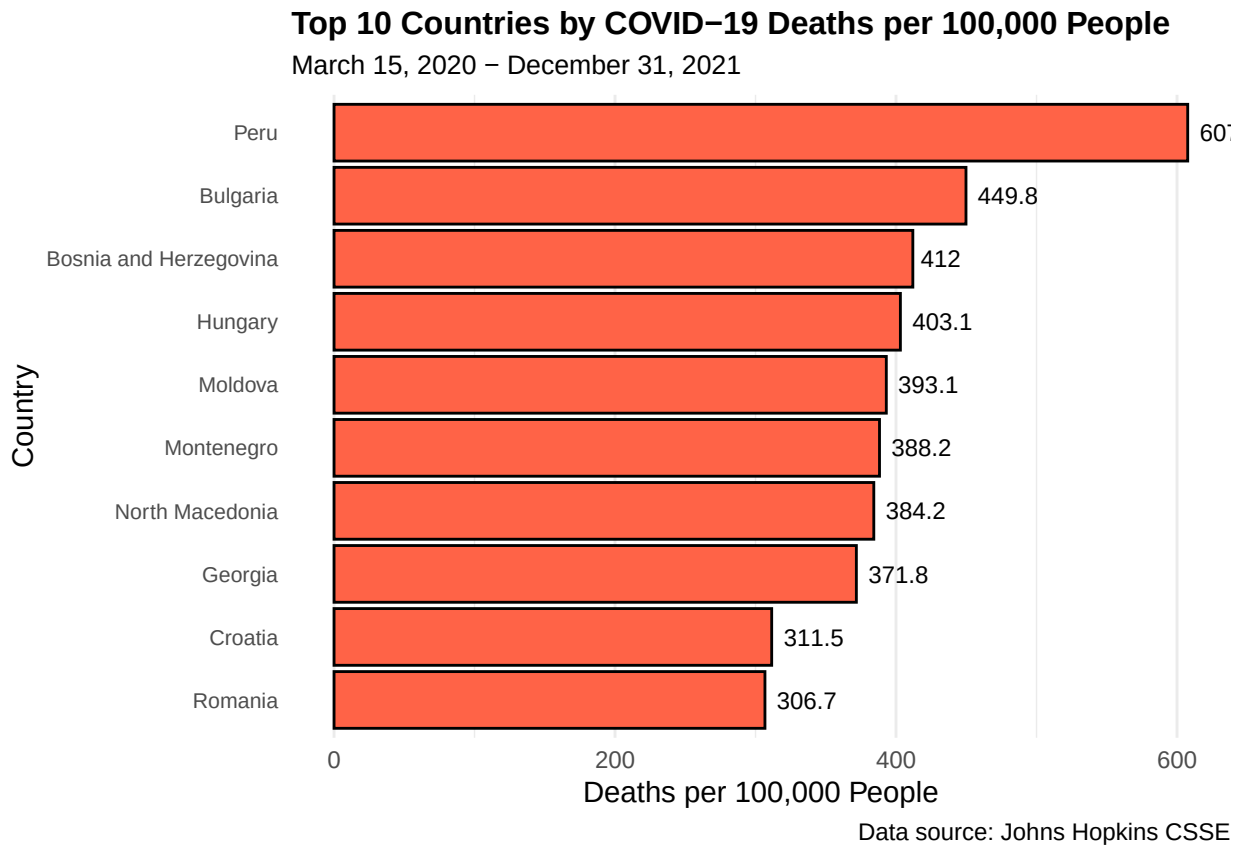
- Use case_when() to match the population for each country based on the year in the date column.
- Filter out rows where the population is missing.
- Calculate the cases and deaths per 100,000 people.
- Summarize the data to get the latest date's cases and deaths per 100,000 people for each country.
- Identify the Top 10 Countries:
- Sort the summarized data in descending order of cases per 100,000 people and select the top 10 countries.
- Sort the summarized data in descending order of deaths per 100,000 people and select the top 10 countries.

**Question 3** Construct a visualization plotting the 10 countries in terms of deaths and cases per 100,000 people between March 15, 2020, and December 31, 2021. In designing your visualization keep the number of data you will be plotting in mind. You may wish to create two separate visualizations, one for deaths and another for cases.

```
# Prepare data for visualization
top_10_cases_viz <- top_10_cases %>%
  mutate(Country = fct_reorder(`Country/Region`, cases_per_100k))

top_10_deaths_viz <- top_10_deaths %>%
  mutate(Country = fct_reorder(`Country/Region`, deaths_per_100k))

# Visualization 1: Top 10 countries by total cases per 100,000
cases_plot <- ggplot(top_10_cases_viz, aes(x = Country, y = cases_per_100k)) +
  geom_bar(stat = "identity", fill = "skyblue", color = "black") +
  coord_flip() +  # Flip coordinates for horizontal bars
  labs(
    title = "Top 10 Countries by COVID-19 Cases per 100,000 People",
    subtitle = "March 15, 2020 - December 31, 2021",
    x = "Country",
    y = "Cases per 100,000 People",
    caption = "Data source: Johns Hopkins CSSE"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 12, face = "bold"),
    plot.subtitle = element_text(size = 10),
    axis.text.y = element_text(size = 8),
    panel.grid.major.y = element_blank()
  ) +
  geom_text(aes(label = round(cases_per_100k, 0)), hjust = -0.2, size = 3)

print(cases_plot)
```




```
# Visualization 2: Top 10 countries by total deaths per 100,000
deaths_plot <- ggplot(top_10_deaths_viz, aes(x = Country, y = deaths_per_100k)) +
  geom_bar(stat = "identity", fill = "tomato", color = "black") +
  coord_flip() +  # Flip coordinates for horizontal bars
  labs(
    title = "Top 10 Countries by COVID-19 Deaths per 100,000 People",
    subtitle = "March 15, 2020 - December 31, 2021",
    x = "Country",
    y = "Deaths per 100,000 People",
    caption = "Data source: Johns Hopkins CSSE"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 12, face = "bold"),
    plot.subtitle = element_text(size = 10),
    axis.text.y = element_text(size = 8),
    panel.grid.major.y = element_blank()
  ) +
  geom_text(aes(label = round(deaths_per_100k, 1)), hjust = -0.2, size = 3)
print(deaths_plot)
```




– Communicate your methodology, results, and interpretation here –

Prepare Data for Visualization:

Top 10 Cases Data: - Reorder the countries based on the number of cases per 100,000 people using fct_reorder().

Top 10 Deaths Data: - Reorder the countries based on the number of deaths per 100,000 people using fct_reorder().

Create Visualizations:

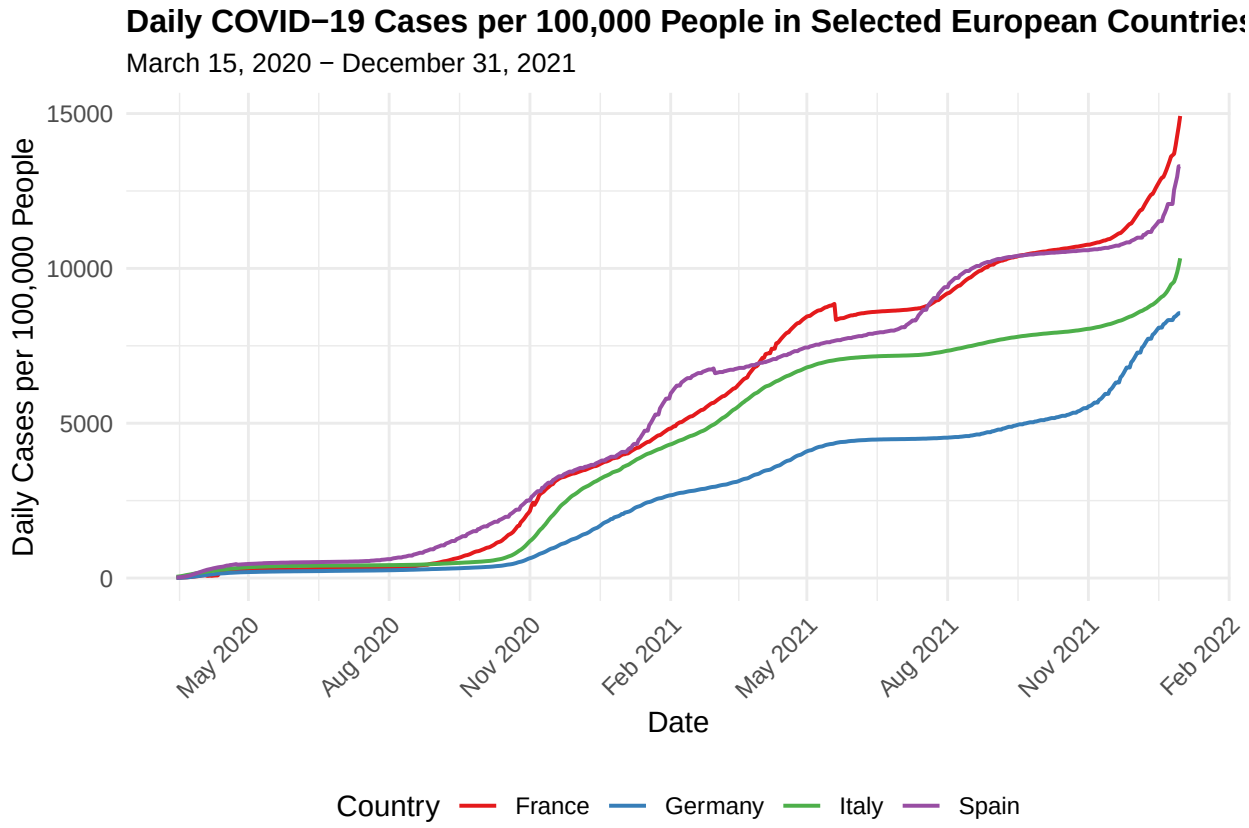
Visualization 1: Top 10 Countries by Cases per 100,000 People: - Use ggplot() to create a bar plot of the top 10 countries by cases per 100,000 people. - Use geom_bar() to create the bars, and set stat = "identity" to use the actual values. - Use coord_flip() to flip the coordinates, making the bars horizontal for better readability. - Add labels and titles using labs(). - Customize the theme using theme_minimal() and adjust various elements for better aesthetics. - Use geom_text() to add the exact numbers on the bars for clarity.

Visualization 2: Top 10 Countries by Deaths per 100,000 People: - Use similar steps as in the cases visualization but with deaths data. - Customize the fill color to distinguish it from the cases visualization.

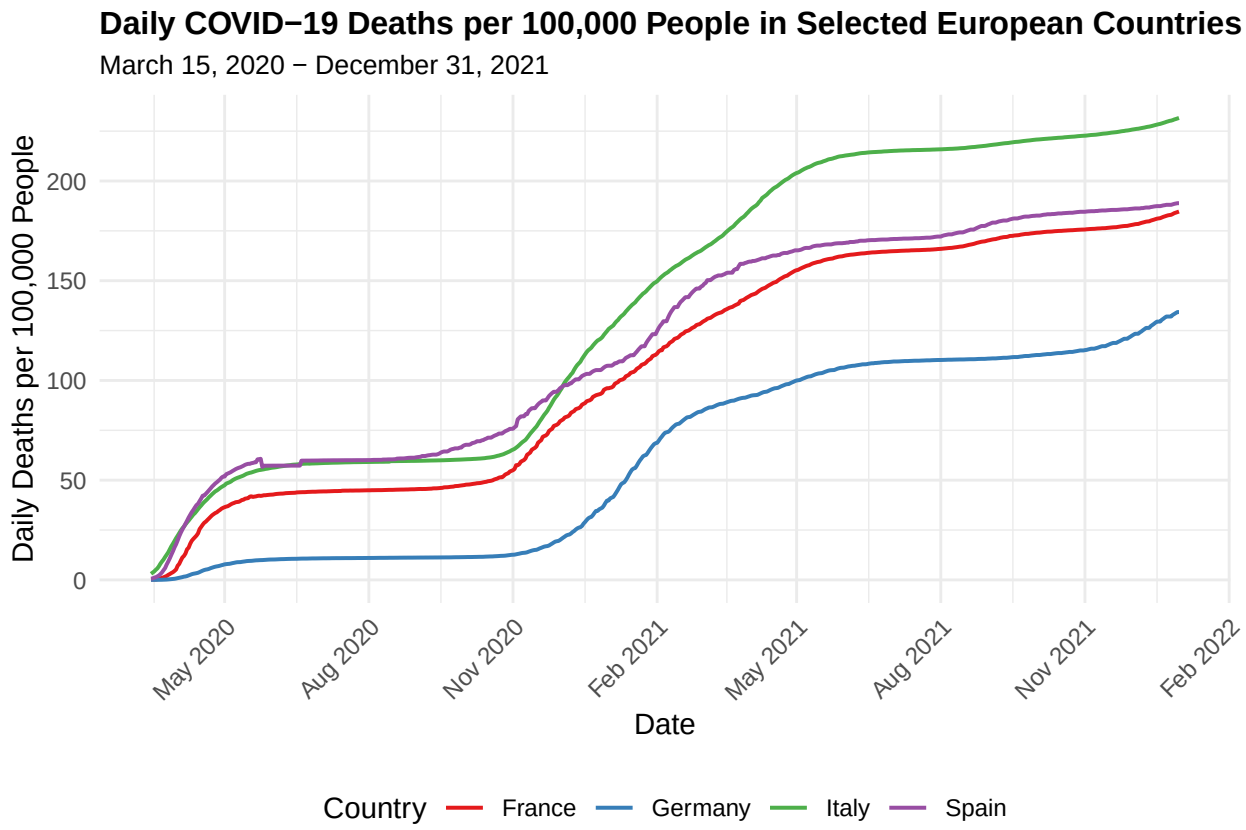
The result shows that Andorra has the highest number of cases per 100,000 people, folllowed by Montenegro, Georgia, Seychilles, San Marino, Slovenia, Mongonia, United Kingdom, Lithuania, and Serbia. For the death cases per 100,000 person, Peru has the highest number of death cases per 100,000 people, folllowed by Bulgaria, Bosnia & Herzegovina, Hungary, Moldova, Montenegro, North Macedonia, Georgia, Croatia and Romania.

**Question 4** Finally, select four countries from one continent and create visualizations for the daily number of confirmed cases per 100,000 and the daily number of deaths per 100,000 people between March 15, 2020, and December 31, 2021.

```r
# Filter and calculate daily metrics for selected countries
selected_countries <- c("Italy", "Spain", "France", "Germany")

#Join both data together
daily_metrics <- global_csse_combined %>%
  filter(date >= "2020-03-15", date <= "2021-12-31", `Country/Region` %in% selected_countries) %>%
  group_by(date,`Country/Region`) %>%
  summarise(
    total_cases = sum(cases, na.rm = TRUE),
    total_deaths = sum(deaths, na.rm = TRUE),
    .groups = "drop")

#Join both data together
daily_metrics <- daily_metrics %>%
  left_join(population_dat, by = c("Country/Region" = "Country Name")) %>%
  mutate(population = case_when(
    grepl("2020", date) ~ pop_2020,
    grepl("2021", date) ~ pop_2021)) %>%
  filter(!is.na(population))

daily_metrics <- daily_metrics %>%
  arrange(date) %>%
  group_by(`Country/Region`) %>%
     mutate(
       cases_per_100k = total_cases / population * 100000,
       deaths_per_100k = total_deaths / population * 100000
     )


# Visualization 1: Daily cases per 100,000
cases_plot <- ggplot(daily_metrics, aes(x = date, y = cases_per_100k, color = `Country/Region`)) +
  geom_line(size = 0.7) +
  scale_color_brewer(palette = "Set1") +
  labs(
    title = "Daily COVID-19 Cases per 100,000 People in Selected European Countries",
    subtitle = "March 15, 2020 - December 31, 2021",
    x = "Date",
    y = "Daily Cases per 100,000 People",
    color = "Country"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 12, face = "bold"),
    plot.subtitle = element_text(size = 10),
    legend.position = "bottom",
    axis.text.x = element_text(angle = 45, hjust = 1)
  ) +
  scale_x_date(date_breaks = "3 months", date_labels = "%b %Y")

print(cases_plot)
```




```r
# Visualization 2: Daily deaths per 100,000
deaths_plot <- ggplot(daily_metrics, aes(x = date, y = deaths_per_100k, color = `Country/Region`)) +
  geom_line(size = 0.7) +
  scale_color_brewer(palette = "Set1") +
  labs(
    title = "Daily COVID-19 Deaths per 100,000 People in Selected European Countries",
    subtitle = "March 15, 2020 - December 31, 2021",
    x = "Date",
    y = "Daily Deaths per 100,000 People",
    color = "Country"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 12, face = "bold"),
    plot.subtitle = element_text(size = 10),
    legend.position = "bottom",
    axis.text.x = element_text(angle = 45, hjust = 1)
  ) +
  scale_x_date(date_breaks = "3 months", date_labels = "%b %Y")

print(deaths_plot)
```




– Communicate your methodology, results, and interpretation here –

Filter and Calculate Daily Metrics for Selected Countries: - Select four European countries: Italy, Spain, France, and Germany. - Filter the combined global CSSE dataset to include only data from the selected countries within the specified date range. - Calculate the total daily cases and deaths for each country.

Join with Population Data: - Left join the daily metrics data with the population data to include population estimates for each country. - Create a population column based on the year (2020 or 2021) to match the date in the dataset. - Filter out rows with missing population data.

Calculate Per Capita Metrics: - For each country, calculate the daily cases and deaths per 100,000 people using the population data. - Ensure the data is arranged by date for accurate time series plotting.

Create Visualizations: Visualization 1 & 2: Daily Cases per 100,000 & Daily Deaths per 100,000: - Use ggplot() to create a line plot of daily cases per 100,000 people. - Color lines by country for differentiation. - Customize the plot with titles, labels, and themes for better readability. - Adjust the x-axis to display dates at three-month intervals.

We observe similar daily COVID-19 deaths cases per 100,000 people in the European Countries selected with Germany having the least death cases.