Final-project-260

Library

```
library(httr2)
library(janitor)
```

```

Attaching package: 'janitor'

The following objects are masked from 'package:stats':

    chisq.test, fisher.test
```

```
library(dplyr)
```

```

Attaching package: 'dplyr'

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

```
library(tidyr)
library(stringr)
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v forcats   1.0.0     v purrr     1.0.2
v ggplot2   3.5.1     v readr     2.1.5
v lubridate 1.9.4     v tibble    3.2.1

-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becom
```

```r
library(readxl)
library(jsonlite)
```

```

Attaching package: 'jsonlite'

The following object is masked from 'package:purrr':

    flatten
```

```r
library(ggplot2)
library(lubridate)
```

```r
census_key <- "9e178f97f6ffeb0a2cdd7608a4119c26733d2705"
```

```r
url <- "https://api.census.gov/data/2021/pep/population"
request <- request(url) |> req_url_query(get = I("POP_2020,POP_2021,NAME"),
                                          'for' = I("state:*"),
                                          key = census_key)
```

```r
# response <- request |> req_perform()
# status <- resp_check_status(response)
# type <- resp_content_type(response)
# population <- response |> resp_body_json(simplifyVector = TRUE)
# population <- population |>
#   row_to_names(1) |>
#   as_tibble() |>
#   select(-state) |>
#   rename(state_name = NAME) |>
#   pivot_longer(-state_name, names_to = "year", values_to = "population") |>
#   mutate(year = str_remove(year, "POP_")) |>
#   mutate(across(-state_name, as.numeric))
```

```r
file_path <- "newData.xlsx"
excel_data <- read_excel(file_path, sheet = "NST-EST2023-POP", skip = 3)
```

```
New names:
* `` -> `...1`
* `` -> `...2`
```

```r
colnames(excel_data) <- c("geographicArea", "2020", "2021", "2022", "2023", "Extra")

cleaned_data <- excel_data |>
  select(geographicArea, `2020`, `2021`, `2022`, `2023`) |>
  filter(!is.na(geographicArea)) |>
  filter(!str_detect(geographicArea, "United States|Region|Division|Northeast|Midwest|2|Cita
  filter(geographicArea != "South" & geographicArea != "West") |>
  mutate(geographicArea = str_remove(geographicArea, "^\\."))

# Reshape into long format
combined_population <- cleaned_data |>
  pivot_longer(cols = c("2020", "2021", "2022", "2023"), names_to = "year", values_to = "popu
  rename(state_name = `geographicArea`) |>
  mutate(year = as.numeric(year),
         population = as.numeric(population))
```

```r
# Calculate the growth rate for 2023
growth_rate_2023 <- combined_population |>
  filter(year == 2023) |>
  left_join(combined_population |>
              filter(year == 2022) |>
              select(state_name, population_2022 = population), by = "state_name") |>
  mutate(growth_rate_2023 = (population - population_2022) / population_2022)

# Estimate the population for 2024 based on the growth rate
estimated_population_2024 <- growth_rate_2023 |>
  mutate(population = round(population * (1 + growth_rate_2023))) |>
  mutate(year = 2024)

combined_population <- bind_rows(combined_population, estimated_population_2024)
combined_population <- combined_population |>
  select(state_name, year, population)
```

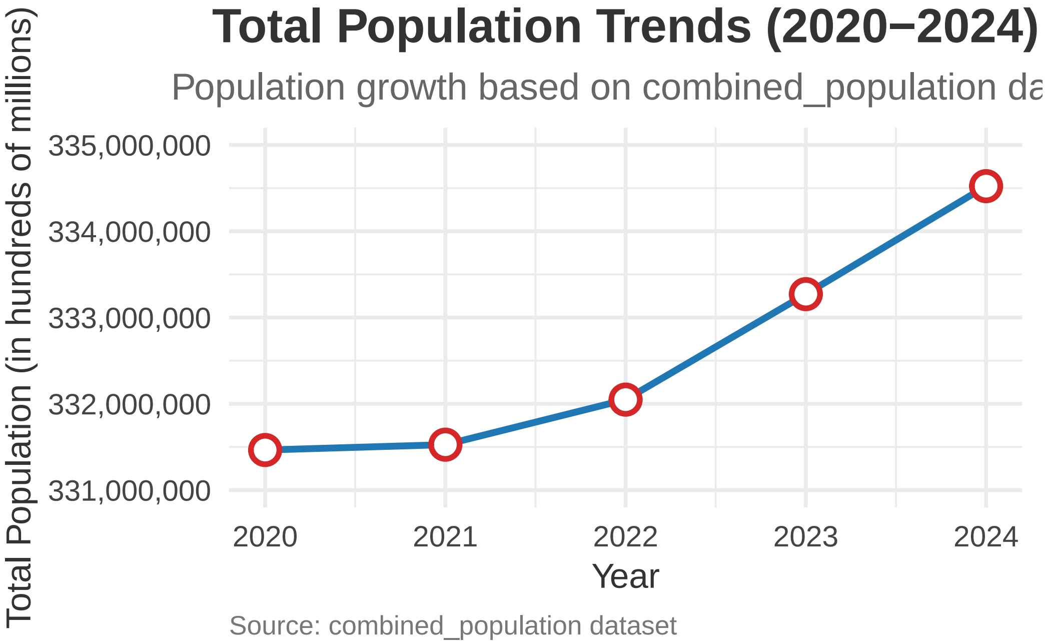
```
# Step 1: Summarize population data by year
yearly_population <- combined_population |>
  group_by(year) |>
  summarize(total_population = sum(population))

# Step 2: Calculate year-over-year changes and percent changes
yearly_population <- yearly_population |>
  mutate(
    Change = if_else(year == 2020, 0, total_population - lag(total_population)),
    Percent_Change = if_else(year == 2020, 0, (Change / lag(total_population)) * 100)
  )
yearly_population
```

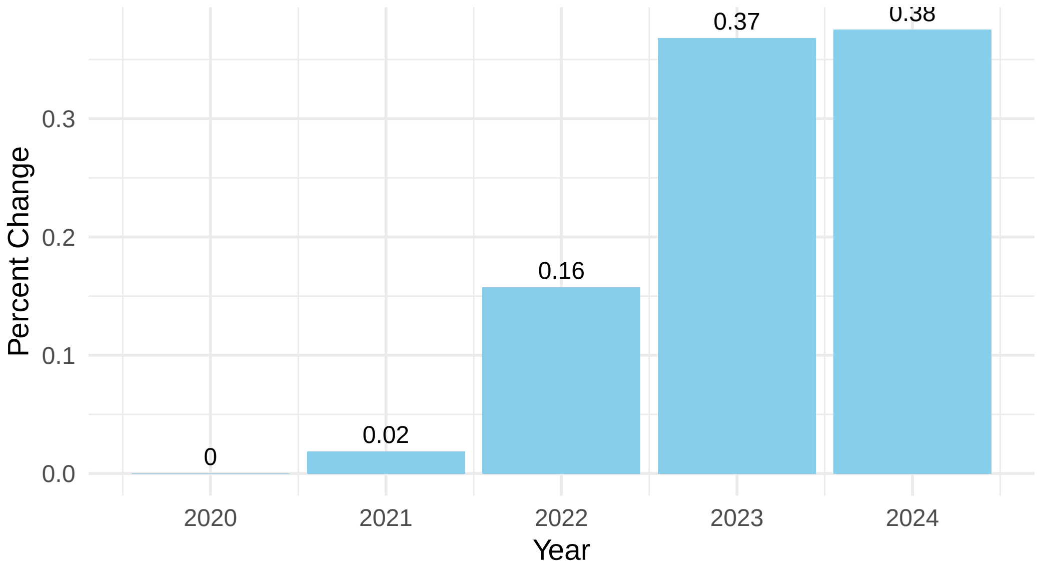
```
# A tibble: 5 x 4
   year total_population  Change Percent_Change
  <dbl>            <dbl>   <dbl>          <dbl>
1  2020        331464948       0         0
2  2021        331526933   61985         0.0187
3  2022        332048977  522044         0.157
4  2023        333271411 1222434         0.368
5  2024        334522730 1251319         0.375
```

```
# Step 3: Line plot for total population
ggplot(yearly_population, aes(x = year, y = total_population)) +
  geom_line(color = "#1f77b4", size = 1.2) +
  geom_point(size = 4, color = "#d62728", shape = 21, fill = "white", stroke = 1.5) +
  labs(
    title = "Total Population Trends (2020-2024)",
    subtitle = "Population growth based on combined_population data",
    x = "Year",
    y = "Total Population (in hundreds of millions)",
    caption = "Source: combined_population dataset"
  ) +
  theme_minimal(base_size = 15) +
  theme(
    plot.title = element_text(face = "bold", size = 18, hjust = 0.5, color = "#333333"),
    plot.subtitle = element_text(size = 14, hjust = 0.5, color = "#666666"),
    axis.title.x = element_text(size = 13, color = "#333333"),
    axis.title.y = element_text(size = 13, color = "#333333"),
    axis.text = element_text(size = 11, color = "#444444"),
    plot.caption = element_text(size = 10, hjust = 0, color = "#777777")
  ) +
  # Fine-tune y-axis scale to show readable numbers
  scale_y_continuous(labels = scales::comma, limits = c(3.31e8, 3.35e8))
```

```
Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
i Please use `linewidth` instead.
```



```
# Step 4: Bar plot for percent changes
ggplot(yearly_population, aes(x = year, y = Percent_Change)) +
  geom_bar(stat = "identity", fill = "skyblue") +
  geom_text(aes(label = round(Percent_Change, 2)), vjust = -0.5, size = 3) +
  labs(
    title = "Year-over-Year Percent Change in Population (2020-2024)",
    x = "Year",
    y = "Percent Change",
    caption = "Calculated from combined_population"
  ) +
  theme_minimal()
```


Year−over−Year Percent Change in Population (2020−2024)

Calculated from combined_population

```
# Using the graph, it looks like there are 3 periods: 2020 - 2021, 2021 - 2022, and 2022 - 20

get_cdc_data <- function(api){
  request(api) |>
    req_url_query("$limit" = 10000000) |>
    req_perform() |>
    resp_body_json(simplifyVector = TRUE)
}

deaths_raw <- get_cdc_data("https://data.cdc.gov/resource/r8kw-7aab.json")

deaths <- deaths_raw |>
  filter(!str_detect(state, "United States")) |>
  drop_na(year, state, covid_19_deaths) |>
  mutate(
    deaths = parse_number(covid_19_deaths),
    year = substr(as.character(start_date), 1, 4)
  ) |>
  group_by(state, year) |>
  summarise(
    total_deaths = sum(deaths, na.rm = TRUE),
    .groups = "drop"
  )
```

```
deaths_2020_table <- deaths |>
  filter(year == 2020) |>
  group_by(state) |>
  summarize(total_deaths = sum(total_deaths, na.rm = TRUE), .groups = 'drop') |>
  arrange(desc(total_deaths))

deaths_2020_table
```

```
# A tibble: 53 x 2
   state          total_deaths
   <chr>                 <dbl>
 1 California           102123
 2 Texas                101211
 3 New York City         66930
 4 Florida               65751
 5 Pennsylvania          55603
 6 New Jersey            54665
 7 Illinois              50374
 8 New York              48877
 9 Ohio                  45611
10 Michigan              37174
# i 43 more rows
```

```
tail(deaths_2020_table, n = 10)  # Show the last 10 rows
```

```
# A tibble: 10 x 2
   state                total_deaths
   <chr>                       <dbl>
 1 North Dakota                 4452
 2 Montana                      3760
 3 Delaware                     3204
 4 District of Columbia         2935
 5 New Hampshire                2427
 6 Wyoming                      1306
 7 Maine                        1282
 8 Hawaii                       1001
 9 Alaska                        678
10 Vermont                       388
```

```r
deaths_2021_table <- deaths |>
  filter(year == 2021) |>
  group_by(state) |>
  summarize(total_deaths = sum(total_deaths, na.rm = TRUE), .groups = 'drop') |>
  arrange(desc(total_deaths))

deaths_2021_table
```

```
# A tibble: 53 x 2
   state          total_deaths
   <chr>                 <dbl>
 1 Texas                145857
 2 California           143703
 3 Florida              116404
 4 Ohio                  61416
 5 Pennsylvania          61390
 6 Georgia               51755
 7 North Carolina        45512
 8 Michigan              44769
 9 New York              44175
10 Arizona               41964
# i 43 more rows
```

```r
tail(deaths_2021_table, n = 10)  # Show the last 10 rows
```

```
# A tibble: 10 x 2
   state                total_deaths
   <chr>                       <dbl>
 1 Delaware                     3717
 2 Rhode Island                 3537
 3 New Hampshire                3496
 4 South Dakota                 2848
 5 Wyoming                      2775
 6 North Dakota                 2691
 7 Alaska                       2400
 8 Hawaii                       2066
 9 District of Columbia         1994
10 Vermont                       752
```

```r
deaths_2022_2023_table <- deaths |>
  filter(year == 2022 | year == 2023) |>
  group_by(state) |>
  summarize(total_deaths = sum(total_deaths, na.rm = TRUE), .groups = 'drop') |>
  arrange(desc(total_deaths))

deaths_2022_2023_table
```

```
# A tibble: 53 x 2
   state          total_deaths
   <chr>                 <dbl>
 1 California            88402
 2 Texas                 70290
 3 Florida               67514
 4 Pennsylvania          46096
 5 Ohio                  45795
 6 New York              36989
 7 North Carolina        33819
 8 Illinois              33153
 9 Michigan              32052
10 Tennessee             28214
# i 43 more rows
```

```r
tail(deaths_2022_2023_table, n = 10)  # Show the last 10 rows
```

```
# A tibble: 10 x 2
   state                total_deaths
   <chr>                       <dbl>
 1 Delaware                     3161
 2 Rhode Island                 2673
 3 Montana                      2541
 4 Hawaii                       2361
 5 South Dakota                 2198
 6 North Dakota                 1777
 7 District of Columbia         1400
 8 Vermont                      1356
 9 Wyoming                      1114
10 Alaska                        953
```

```
api <- "https://data.cdc.gov/resource/pwn4-m3yp.json"
res <- request(api) |> req_url_query('$limit'=10000000000) |> req_perform()
cases <- res |>
    resp_body_json(simplifyDataFrame = TRUE) |>
    as.data.frame() |>
  select(state, date = end_date, case = new_cases) |>
  mutate(case = as.numeric(case))

cases_summary <- cases |>
  mutate(year = substr(date, 1, 4),
         period = case_when(
           year == 2020 ~ "2020",
           year == 2021 ~ "2021",
           year %in% c(2022, 2023) ~ "2022-2023"
         )) |>
  group_by(period) |>
  summarise(
    total_cases = sum(case, na.rm = TRUE),
    avg_daily_cases = mean(case, na.rm = TRUE),
    .groups = "drop"
)

cases_summary
```

```
# A tibble: 3 x 3
  period    total_cases avg_daily_cases
  <chr>           <dbl>           <dbl>
1 2020         19802808           6601.
2 2021         33816455          10839.
3 2022-2023    51038265          11981.
```

```
cases_2020_table <- cases |>
  filter(substr(date, 1, 4) == "2020") |>
  summarize(total_cases = sum(case, na.rm = TRUE), .groups = 'drop')

cases_2020_table
```

```
  total_cases
1    19802808
```

```r
cases_2021_table <- cases |>
  filter(substr(date, 1, 4) == "2021") |>
  summarize(total_cases = sum(case, na.rm = TRUE), .groups = 'drop')

cases_2021_table
```

```
  total_cases
1    33816455
```

```r
cases_2022_table <- cases |>
  filter(substr(date, 1, 4) == "2022") |>
  summarize(total_cases = sum(case, na.rm = TRUE), .groups = 'drop')

cases_2022_table
```

```
  total_cases
1    46928756
```

```r
cases_2023_table <- cases |>
  filter(substr(date, 1, 4) == "2023") |>
  summarize(total_cases = sum(case, na.rm = TRUE), .groups = 'drop')

cases_2023_table
```

```
  total_cases
1     4109509
```