Week 2, Class 4: Practice Exercises - ANSWER KEY

Summary Statistics

2024-12-31

1 Non-AI Exercises

1.1 1. Understanding Measures of Central Tendency

1.1.1 1.1 Multiple Choice: Mean vs Median

When should you use the median instead of the mean?

a) When you want the most precise calculation
b) When the data has extreme outliers or is skewed
c) When all values are the same
d) When you have categorical data

Answer: **b) When the data has extreme outliers or is skewed**

Explanation: The median is robust to outliers because it represents the middle value when data is ordered. Unlike the mean, which is pulled toward extreme values, the median remains stable. For example, if incomes are $30k, $35k, $40k, $45k, and $1 million, the mean ($230k) is misleading while the median ($40k) better represents the typical value.

1.1.2 1.2 Fill in the Blanks: Central Tendency

Complete these statements about measures of center:

1. The **mean** is the sum of all values divided by the count
2. The **median** is the middle value when data is ordered
3. The **mode** is the most frequently occurring value
4. When data is skewed right, the mean is typically **greater** than the median

5. The **mean** is most affected by outliers

Explanation: In right-skewed data (like income), extreme high values pull the mean upward while the median stays centered, making mean > median. The mean incorporates every value in its calculation, making it sensitive to outliers.

1.1.3 1.3 Code Detective: Grouped Summaries

What does this code calculate?

```
approval %>%
  group_by(party_id) %>%
  summarise(
    avg_approval = mean(congress_approval, na.rm = TRUE),
    n = n()
  )
```

This code calculates: **The average congressional approval rating for each political party (Democrat, Republican, Independent) and counts how many respondents are in each party group.**

Explanation: `group_by(party_id)` splits the data by party affiliation, then `summarise()` calculates the mean approval within each group. The `n()` function counts observations per group, and `na.rm = TRUE` removes missing values before calculating the mean.

1.2 2. Measures of Spread

1.2.1 2.1 Match: Measures of Spread

Match each measure with its definition:

Measures: a) Range b) Variance c) Standard deviation d) Interquartile range (IQR)

Definitions: 1. Square root of the variance 2. Maximum value minus minimum value 3. Average squared deviation from the mean 4. Distance between 25th and 75th percentiles

Matches: a = **2**, b = **3**, c = **1**, d = **4**

1.2.2 2.2 Multiple Choice: Standard Deviation

If approval ratings have a mean of 45% and standard deviation of 10%, approximately what percentage of responses fall between 35% and 55%?

a) 50%
b) 68%
c) 95%
d) 99%

Answer: **b) 68%** (Hint: Think about the empirical rule)

Explanation: The empirical rule (68-95-99.7 rule) states that for normally distributed data, approximately 68% of observations fall within one standard deviation of the mean. Here, 45% ± 10% gives us the range 35% to 55%, which captures about 68% of the data.

1.2.3 2.3 True or False: Spread

Mark each statement as True (T) or False (F):

T Variance is always positive or zero **T** Standard deviation has the same units as the original data **T** A larger standard deviation means data is more spread out **T** The range is affected by outliers **T** IQR is more robust to outliers than standard deviation

Explanations: - Variance is squared deviations, so always 0 - SD is the square root of variance, returning to original units - Larger SD means values are further from the mean on average - Range uses min and max, both susceptible to outliers - IQR uses middle 50% of data, ignoring extremes

1.3 3. The `summarise()` Function

1.3.1 3.1 Fill in the Code: summarise()

Complete this code to calculate mean, median, and standard deviation:

```
approval %>%
  summarise(
    mean_approval = mean(congress_approval, na.rm = TRUE),
    median_approval = median(congress_approval, na.rm = TRUE),
    sd_approval = sd(congress_approval, na.rm = TRUE)
  )
```

Functions needed: **mean**, **median**, **sd**

1.3.2 3.2 Multiple Choice: NA Values

What happens when you run `mean(c(1, 2, NA, 4))`?

a) Returns 2.33 (ignores the NA)
b) Returns NA
c) Gives an error
d) Returns 2 (treats NA as 0)

Answer: **b) Returns NA**

Explanation: R propagates NA (missing) values through calculations by default. Any operation involving NA returns NA unless you explicitly tell R to remove them using `na.rm = TRUE`. This forces you to make conscious decisions about handling missing data.

1.4 4. Real-World Applications

1.4.1 4.1 Match: Statistical Concepts

Match polling concepts with statistical terms:

Polling Terms: a) Margin of error b) Sample size c) Confidence level d) Poll average

Statistical Terms: 1. Mean of multiple measurements 2. Number of observations (n) 3. Related to standard error 4. Probability of capturing true value

Matches: a = **3**, b = **2**, c = **4**, d = **1**

Explanation: Margin of error is typically 1.96 × standard error for 95% confidence. Sample size (n) is the count of respondents. Confidence level (e.g., 95%) is the probability the interval contains the true population value. Poll averages aggregate multiple polls using the mean.

1.4.2 4.2 Code Detective: Real Analysis

What question does this code answer?

```
approval %>%
  group_by(party_id, region) %>%
  summarise(
    mean_approval = mean(congress_approval, na.rm = TRUE),
    sd_approval = sd(congress_approval, na.rm = TRUE),
    count = n()
  )
```

This analysis shows: **How congressional approval ratings vary by both party affiliation AND geographic region, including the average approval, spread of opinions (standard deviation), and number of respondents in each party-region combination.**

Example interpretation: We might find that Democrats in the Northeast have higher approval (mean = 55%, sd = 12%) than Republicans in the South (mean = 35%, sd = 15%), showing both partisan and regional differences.

2 AI Exercises

For each AI exercise: - Work with Claude to analyze the data - Record your prompts and key findings

2.1 5. Congressional Approval Analysis

Dataset: congressional_approval.csv

Description: Survey data on congressional approval ratings with demographics.

Variables: - `respondent_id`: Unique identifier (int) - `age`: Respondent's age (int) - `education`: Education level (chr) - `party_id`: Democrat, Republican, Independent (chr) - `income_category`: Income bracket (chr) - `region`: Geographic region (chr) - `congress_approval`: Approval rating for Congress, 0-100 scale (dbl)

2.1.1 5.1 Initial Data Exploration

```r
# Load the dataset
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.2
v ggplot2   4.0.0     v tibble    3.3.0
v lubridate 1.9.4     v tidyr     1.3.1
v purrr     1.1.0
-- 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
approval <- read_csv("congressional_approval.csv")
```

```
Rows: 2000 Columns: 7
-- Column specification --------------------------------------------------------
Delimiter: ","
chr (4): education, party_id, income_category, region
dbl (3): respondent_id, age, congress_approval

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.
```

```r
# Examine the data
glimpse(approval)
```

```
Rows: 2,000
Columns: 7
$ respondent_id     <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 1~
$ age               <dbl> 31, 68, 41, 75, 19, 55, 27, 49, 57, 56, 55, 47, 18, ~
$ education         <chr> "Some College", "Bachelor's", "Some College", "Bache~
$ party_id          <chr> "Independent", "Republican", "Republican", "Democrat~
$ income_category   <chr> "$30k-$60k", "$30k-$60k", "$30k-$60k", "$30k-$60k", ~
$ region            <chr> "Midwest", "South", "South", "South", "West", "South~
$ congress_approval <dbl> 19.132044, 19.898068, 17.585641, 52.891775, 14.57116~
```

```r
summary(approval$congress_approval)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
   0.00   19.07   28.51   28.71   38.15   75.88 
```

Use Claude to calculate and interpret basic summary statistics for congressional approval ratings.

Prompt to Claude: I have a dataset called approval with congressional approval ratings on a 0-100 scale. Help me calculate comprehensive summary statistics including mean, median, mode, standard deviation, and percentiles. Also check for outliers and describe the distribution shape. Use tidyverse.

```r
# Comprehensive summary statistics
approval_summary <- approval %>%
    summarise(n = n(), mean_approval = mean(congress_approval,
        na.rm = TRUE), median_approval = median(congress_approval,
        na.rm = TRUE), sd_approval = sd(congress_approval, na.rm = TRUE),
        min_approval = min(congress_approval, na.rm = TRUE),
        max_approval = max(congress_approval, na.rm = TRUE),
        q1 = quantile(congress_approval, 0.25, na.rm = TRUE),
        q3 = quantile(congress_approval, 0.75, na.rm = TRUE),
        iqr = IQR(congress_approval, na.rm = TRUE), skewness = (mean_approval -
            median_approval)/sd_approval)

print(approval_summary)
```

```
# A tibble: 1 x 10
      n mean_approval median_approval sd_approval min_approval max_approval
  <int>         <dbl>           <dbl>       <dbl>        <dbl>        <dbl>
1  2000          28.7            28.5        13.5            0         75.9
# i 4 more variables: q1 <dbl>, q3 <dbl>, iqr <dbl>, skewness <dbl>
```

```r
# Check distribution
approval %>%
    count(cut(congress_approval, breaks = seq(0, 100, by = 10))) %>%
    print()
```

```
# A tibble: 9 x 2
  `cut(congress_approval, breaks = seq(0, 100, by = 10))`     n
  <fct>                                                   <int>
1 (0,10]                                                    129
2 (10,20]                                                   378
3 (20,30]                                                   542
4 (30,40]                                                   511
5 (40,50]                                                   277
6 (50,60]                                                   100
7 (60,70]                                                    26
8 (70,80]                                                     2
9 <NA>                                                       35
```

```r
# Identify potential outliers using IQR method
outlier_bounds <- approval_summary %>%
    mutate(lower_bound = q1 - 1.5 * iqr, upper_bound = q3 + 1.5 *
```

```
        iqr)

outliers <- approval %>%
    filter(congress_approval < outlier_bounds$lower_bound | congress_approval >
        outlier_bounds$upper_bound) %>%
    nrow()

print(paste("Number of potential outliers:", outliers))
```

```
[1] "Number of potential outliers: 2"
```

Interpretation: Congressional approval shows moderate ratings with substantial variation. The relationship between mean and median indicates the distribution shape, while the standard deviation reveals how polarized opinions are about Congress.

2.1.2 5.2 Approval by Party

Work with Claude to calculate mean and median approval ratings by party affiliation. What patterns do you observe? Are there differences in how spread out opinions are within each party?

Prompt to Claude: Using the approval dataset, calculate mean, median, and standard deviation of congress_approval by party_id. Also calculate the coefficient of variation (CV = sd/mean) to compare relative variability. Which party shows most consensus? Use tidyverse.

```
# Summary statistics by party
party_summary <- approval %>%
    group_by(party_id) %>%
    summarise(n = n(), mean_approval = mean(congress_approval,
        na.rm = TRUE), median_approval = median(congress_approval,
        na.rm = TRUE), sd_approval = sd(congress_approval, na.rm = TRUE),
        cv = sd_approval/mean_approval, min = min(congress_approval,
            na.rm = TRUE), max = max(congress_approval, na.rm = TRUE),
        range = max - min, .groups = "drop") %>%
    arrange(desc(mean_approval))

print(party_summary)
```

```
# A tibble: 3 x 9
  party_id        n mean_approval median_approval sd_approval    cv   min   max
```

```
  <chr>        <int>          <dbl>            <dbl>        <dbl> <dbl> <dbl> <dbl>
1 Democrat       682           36.5             37.4         12.2 0.333     0  73.4
2 Republican     697           27.1             26.9         12.1 0.448     0  75.9
3 Independent    621           22.0             21.5         12.2 0.553     0  64.6
# i 1 more variable: range <dbl>
```

```r
# Compare spreads
print("Party with highest consensus (lowest SD):")
```

```
[1] "Party with highest consensus (lowest SD):"
```

```r
party_summary %>%
    filter(sd_approval == min(sd_approval)) %>%
    select(party_id, sd_approval)
```

```
# A tibble: 1 x 2
  party_id   sd_approval
  <chr>            <dbl>
1 Republican        12.1
```

```r
print("Party with most disagreement (highest SD):")
```

```
[1] "Party with most disagreement (highest SD):"
```

```r
party_summary %>%
    filter(sd_approval == max(sd_approval)) %>%
    select(party_id, sd_approval)
```

```
# A tibble: 1 x 2
  party_id    sd_approval
  <chr>             <dbl>
1 Independent        12.2
```

Interpretation: Party differences in congressional approval reflect partisan polarization. The party of the current majority often shows higher approval from their base. Standard deviations reveal whether party members are unified or divided in their views.

2.1.3 5.3 Regional Variations

Ask Claude to help you explore how congressional approval varies by region. Which regions show the highest and lowest approval? Which have the most consensus (lowest standard deviation)?

Prompt to Claude: Analyze congressional approval by region. Calculate summary statistics, identify which regions have highest/lowest approval, and which show most/least consensus (using standard deviation). Also explore if regional differences are statistically meaningful. Use tidyverse.

```r
# Regional analysis
regional_summary <- approval %>%
    group_by(region) %>%
    summarise(n = n(), mean_approval = mean(congress_approval,
        na.rm = TRUE), median_approval = median(congress_approval,
        na.rm = TRUE), sd_approval = sd(congress_approval, na.rm = TRUE),
        se_approval = sd_approval/sqrt(n), ci_lower = mean_approval -
            1.96 * se_approval, ci_upper = mean_approval + 1.96 *
            se_approval, .groups = "drop") %>%
    arrange(desc(mean_approval))

print("Regional approval rankings:")
```

```
[1] "Regional approval rankings:"
```

```r
print(regional_summary)
```

```
# A tibble: 4 x 8
  region        n mean_approval median_approval sd_approval se_approval ci_lower
  <chr>     <int>         <dbl>           <dbl>       <dbl>       <dbl>    <dbl>
1 South       782          29.2            28.9        13.6       0.488     28.2
2 Midwest     424          28.9            28.5        12.9       0.629     27.6
3 Northeast   354          28.6            27.8        14.1       0.748     27.2
4 West        440          27.8            28.1        13.4       0.638     26.6
# i 1 more variable: ci_upper <dbl>
```

```r
# Identify extremes
print(paste("Highest approval region:", regional_summary$region[1],
    "with mean =", round(regional_summary$mean_approval[1], 1)))
```

```
[1] "Highest approval region: South with mean = 29.2"
```

```
print(paste("Most consensus region:", regional_summary$region[which.min(regional_summary$sd_a
    "with SD =", round(min(regional_summary$sd_approval), 1)))
```

```
[1] "Most consensus region: Midwest with SD = 12.9"
```

```
# Range of regional means
regional_range <- max(regional_summary$mean_approval) - min(regional_summary$mean_approval)
print(paste("Regional approval ranges from", round(min(regional_summary$mean_approval),
    1), "to", round(max(regional_summary$mean_approval), 1),
    "a difference of", round(regional_range, 1), "points"))
```

```
[1] "Regional approval ranges from 27.8 to 29.2 a difference of 1.3 points"
```

Interpretation: Regional variations in congressional approval often reflect local political cultures, economic conditions, and representation. Regions with more consensus (lower SD) may have more homogeneous political views.

2.1.4 5.4 Creating a Summary Report

Work with Claude to create a comprehensive summary table that shows approval statistics by both party and region. What insights emerge from this analysis?

Prompt to Claude: Create a comprehensive summary table showing congressional approval by party AND region combinations. Include mean, SD, and sample size. Identify interesting patterns like which party-region combination has highest/lowest approval. Format the output clearly. Use tidyverse.

```
# Comprehensive party-region analysis
party_region_summary <- approval %>%
    group_by(party_id, region) %>%
    summarise(n = n(), mean_approval = mean(congress_approval,
        na.rm = TRUE), sd_approval = sd(congress_approval, na.rm = TRUE),
        .groups = "drop") %>%
    arrange(party_id, desc(mean_approval))

# Display as a matrix
approval_matrix <- party_region_summary %>%
    select(party_id, region, mean_approval) %>%
```

```r
    pivot_wider(names_from = region, values_from = mean_approval)

print("Mean Approval by Party and Region:")
```

```
[1] "Mean Approval by Party and Region:"
```

```r
print(approval_matrix)
```

```
# A tibble: 3 x 5
  party_id    Northeast South Midwest  West
  <chr>           <dbl> <dbl>   <dbl> <dbl>
1 Democrat         38.0  36.7    36.4  35.0
2 Independent      21.3  22.4    22.7  21.2
3 Republican       27.0  27.0    27.7  26.8
```

```r
# Find extremes
extremes <- party_region_summary %>%
    mutate(combo = paste(party_id, "-", region)) %>%
    summarise(highest = combo[which.max(mean_approval)], highest_approval = max(mean_approva
        lowest = combo[which.min(mean_approval)], lowest_approval = min(mean_approval),
        most_consensus = combo[which.min(sd_approval)], consensus_sd = min(sd_approval),
        most_divided = combo[which.max(sd_approval)], divided_sd = max(sd_approval))

print("\nKey Findings:")
```

```
[1] "\nKey Findings:"
```

```r
print(paste("Highest approval:", extremes$highest, "at", round(extremes$highest_approval,
    1)))
```

```
[1] "Highest approval: Democrat - Northeast at 38"
```

```r
print(paste("Lowest approval:", extremes$lowest, "at", round(extremes$lowest_approval,
    1)))
```

```
[1] "Lowest approval: Independent - West at 21.2"
```

```
print(paste("Most unified group:", extremes$most_consensus, "SD =",
    round(extremes$consensus_sd, 1)))
```

```
[1] "Most unified group: Democrat - Midwest SD = 11.3"
```

Interpretation: The intersection of party and region reveals nuanced patterns in political attitudes. Some combinations (like Democrats in liberal regions) show high approval and consensus, while others show internal divisions.

2.2 6. Income and Wealth Distribution

Dataset: household_panel.csv

Description: Panel data tracking household economic conditions over time.

Variables: - `hh_id`: Unique household identifier (int) - `year`: Year of observation (int) - `head_age`: Age of household head (int) - `education_head`: Education of household head (chr) - `num_children`: Number of children (int) - `income`: Annual household income (dbl) - `has_employer_insurance`: Has employer insurance (lgl) - `self_reported_health`: Health status (chr) - `unexpected_expense`: Can handle $400 expense (lgl)

2.2.1 6.1 Loading and Understanding the Data

```
# Load the dataset
household <- read_csv("household_panel.csv")
```

```
Rows: 6000 Columns: 9
-- Column specification --------------------------------------------------------
Delimiter: ","
dbl (9): hh_id, year, head_age, education_head, num_children, income, has_em...

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.
```

```
# Check the structure
glimpse(household)
```

```
Rows: 6,000
Columns: 9
$ hh_id                   <dbl> 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3,~
$ year                    <dbl> 2019, 2020, 2021, 2022, 2023, 2024, 2019, 2020,~
$ head_age                <dbl> 58, 58, 58, 58, 58, 58, 26, 26, 26, 26, 26, 26,~
$ education_head          <dbl> 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4,~
$ num_children            <dbl> 1, 1, 0, 2, 0, 0, 0, 1, 0, 1, 3, 0, 2, 3, 2, 2,~
$ income                  <dbl> 47159, 114000, 178018, 60681, 42426, 40098, 931~
$ has_employer_insurance <dbl> 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1,~
$ self_reported_health   <dbl> 4, 1, 4, 3, 3, 3, 3, 2, 3, 2, 3, 2, 2, 4, 4, 3,~
$ unexpected_expense     <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,~
```

```r
# Basic summary of income
summary(household$income)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  13150   47688   64208   71502   87016  320524 
```

```r
# Panel structure
household %>%
    count(year) %>%
    print()
```

```
# A tibble: 6 x 2
   year     n
  <dbl> <int>
1  2019  1000
2  2020  1000
3  2021  1000
4  2022  1000
5  2023  1000
6  2024  1000
```

```r
household %>%
    group_by(hh_id) %>%
    count() %>%
    count(n, name = "households") %>%
    print()
```

```
# A tibble: 1,000 x 3
```

```
# Groups:   hh_id [1,000]
   hh_id     n households
   <dbl> <int>      <int>
 1     1     6          1
 2     2     6          1
 3     3     6          1
 4     4     6          1
 5     5     6          1
 6     6     6          1
 7     7     6          1
 8     8     6          1
 9     9     6          1
10    10     6          1
# i 990 more rows
```

This is panel data (same households observed over multiple years). Ask Claude to help you understand the income and wealth distributions.

Prompt to Claude: I have household panel data with income information. Help me understand the income distribution including calculating percentiles (10th, 25th, 50th, 75th, 90th), identifying skewness, and checking how income varies over the years in the panel. Use tidyverse.

```r
# Income distribution analysis
income_stats <- household %>%
    summarise(mean_income = mean(income, na.rm = TRUE), median_income = median(income,
        na.rm = TRUE), sd_income = sd(income, na.rm = TRUE),
        p10 = quantile(income, 0.1, na.rm = TRUE), p25 = quantile(income,
            0.25, na.rm = TRUE), p50 = quantile(income, 0.5,
            na.rm = TRUE), p75 = quantile(income, 0.75, na.rm = TRUE),
        p90 = quantile(income, 0.9, na.rm = TRUE), p99 = quantile(income,
            0.99, na.rm = TRUE))

print(income_stats)
```

```
# A tibble: 1 x 9
  mean_income median_income sd_income    p10    p25   p50    p75     p90     p99
        <dbl>         <dbl>     <dbl>  <dbl>  <dbl> <dbl>  <dbl>   <dbl>   <dbl>
1      71502.         64208    34065. 36506. 47688. 64208 87016. 114505. 187858.
```

```r
# Calculate inequality measures
inequality <- income_stats %>%
    mutate(p90_p10_ratio = p90/p10, p90_p50_ratio = p90/p50,
        p50_p10_ratio = p50/p10, gini_approx = sd_income/(2 *
            mean_income)  # Rough approximation
)

print("Inequality Measures:")
```

```
[1] "Inequality Measures:"
```

```r
print(select(inequality, p90_p10_ratio, p90_p50_ratio, p50_p10_ratio))
```

```
# A tibble: 1 x 3
  p90_p10_ratio p90_p50_ratio p50_p10_ratio
          <dbl>         <dbl>         <dbl>
1          3.14          1.78          1.76
```

```r
# Income trends over time
income_by_year <- household %>%
    group_by(year) %>%
    summarise(mean_income = mean(income, na.rm = TRUE), median_income = median(income,
        na.rm = TRUE), p25 = quantile(income, 0.25, na.rm = TRUE),
        p75 = quantile(income, 0.75, na.rm = TRUE), .groups = "drop")

print("Income Trends Over Time:")
```

```
[1] "Income Trends Over Time:"
```

```r
print(income_by_year)
```

```
# A tibble: 6 x 5
   year mean_income median_income    p25    p75
  <dbl>       <dbl>         <dbl>  <dbl>  <dbl>
1  2019      72191.         64988  48206. 87015.
2  2020      71447.         63738. 47638. 87865
3  2021      70562.         64307  47325. 86654.
4  2022      71845.         63642. 46594. 87874.
5  2023      72192.         65658. 48606  87281.
6  2024      70773.         63094  47940. 85944.
```

Interpretation: Income distributions are typically right-skewed with a long tail of high earners. The percentile ratios help quantify inequality - a p90/p10 ratio of 10 means the 90th percentile earns 10 times the 10th percentile.

2.2.2 6.2 Income and Health

Work with Claude to explore the relationship between household income and self-reported health status. Do higher-income households report better health? Calculate appropriate summary statistics.

Prompt to Claude: Analyze the relationship between income and self_reported_health. Calculate mean and median income by health status, and explore what percentage of each income quartile reports good/excellent health. Use tidyverse.

```r
# Income by health status
health_income <- household %>%
    group_by(self_reported_health) %>%
    summarise(n = n(), mean_income = mean(income, na.rm = TRUE),
        median_income = median(income, na.rm = TRUE), sd_income = sd(income,
            na.rm = TRUE), pct_insured = mean(has_employer_insurance,
            na.rm = TRUE) * 100, .groups = "drop") %>%
    arrange(desc(mean_income))

print("Income by Health Status:")
```

```
[1] "Income by Health Status:"
```

```r
print(health_income)
```

```
# A tibble: 5 x 6
  self_reported_health     n mean_income median_income sd_income pct_insured
                 <dbl> <int>       <dbl>         <dbl>     <dbl>       <dbl>
1                    2   943      72501.         64183    34401.        68.8
2                    4  1790      71950.         64678.   34866.        70.2
3                    5   563      71696.         64635    33767.        70.5
4                    3  2413      70827.         63640    33130.        70.0
5                    1   291      70722.         62915    36251.        71.8
```

```r
# Create income quartiles and examine health
income_health_quartiles <- household %>%
    mutate(income_quartile = ntile(income, 4)) %>%
    group_by(income_quartile) %>%
    summarise(n = n(), income_range = paste(round(min(income)),
        "-", round(max(income))), mean_income = mean(income),
        pct_excellent = mean(self_reported_health == "Excellent",
            na.rm = TRUE) * 100, pct_good_plus = mean(self_reported_health %in%
            c("Excellent", "Good"), na.rm = TRUE) * 100, pct_poor = mean(self_reported_healt
            "Poor", na.rm = TRUE) * 100, .groups = "drop")

print("Health by Income Quartile:")
```

```
[1] "Health by Income Quartile:"
```

```r
print(income_health_quartiles)
```

```
# A tibble: 4 x 7
  income_quartile     n income_range   mean_income pct_excellent pct_good_plus
            <int> <int> <chr>                <dbl>         <dbl>         <dbl>
1               1  1500 13150 - 47672       37491.             0             0
2               2  1500 47693 - 64193       55983.             0             0
3               3  1500 64223 - 87008       74751.             0             0
4               4  1500 87038 - 320524     117781.             0             0
# i 1 more variable: pct_poor <dbl>
```

```r
# Health-income gradient
print(paste("Correlation between income and health (numeric encoding):",
    round(cor(household$income, as.numeric(factor(household$self_reported_health)),
        use = "complete.obs"), 3)))
```

```
[1] "Correlation between income and health (numeric encoding): 0.003"
```

Interpretation: The income-health gradient is a well-documented phenomenon where higher income is associated with better health outcomes. This relationship reflects multiple factors including access to healthcare, nutrition, stress levels, and living conditions.

2.2.3 6.3 Financial Security

Ask Claude to help you analyze financial security using the unexpected_expense variable. What percentage of households can handle a $400 emergency expense? How does this vary by income and education?

Prompt to Claude: Analyze financial security using the unexpected_expense variable (ability to handle $400 emergency). Calculate overall percentage who can handle it, then break down by income quartiles and education levels. Identify which groups are most financially vulnerable. Use tidyverse.

```r
# Overall financial security
overall_security <- household %>%
    summarise(total_households = n(), can_handle_expense = sum(unexpected_expense,
        na.rm = TRUE), pct_secure = mean(unexpected_expense,
        na.rm = TRUE) * 100)

print(paste("Overall:", round(overall_security$pct_secure, 1),
    "% can handle a $400 emergency expense"))
```

```
[1] "Overall: 18.5 % can handle a $400 emergency expense"
```

```r
# By income quartiles
security_by_income <- household %>%
    mutate(income_quartile = ntile(income, 4)) %>%
    group_by(income_quartile) %>%
    summarise(n = n(), mean_income = mean(income), pct_can_handle = mean(unexpected_expense,
        na.rm = TRUE) * 100, .groups = "drop")

print("\nFinancial Security by Income Quartile:")
```

```
[1] "\nFinancial Security by Income Quartile:"
```

```r
print(security_by_income)
```

```
# A tibble: 4 x 4
  income_quartile     n mean_income pct_can_handle
            <int> <int>       <dbl>          <dbl>
1               1  1500      37491.           18.9
2               2  1500      55983.           19
3               3  1500      74751.           18.8
4               4  1500     117781.           17.3
```

```r
# By education
security_by_education <- household %>%
    group_by(education_head) %>%
    summarise(n = n(), mean_income = mean(income, na.rm = TRUE),
        pct_can_handle = mean(unexpected_expense, na.rm = TRUE) *
            100, pct_insured = mean(has_employer_insurance, na.rm = TRUE) *
            100, .groups = "drop") %>%
    arrange(desc(pct_can_handle))

print("\nFinancial Security by Education:")
```

```
[1] "\nFinancial Security by Education:"
```

```r
print(security_by_education)
```

```
# A tibble: 5 x 5
  education_head     n mean_income pct_can_handle pct_insured
           <dbl> <int>       <dbl>          <dbl>       <dbl>
1              4  1248      71233.           21.2        70.0
2              2  1242      70726.           18.1        72.3
3              5  1152      70807.           18.0        68.8
4              1  1170      73074.           17.9        68.5
5              3  1188      71720.           17.2        70.3
```

```r
# Identify vulnerable groups
vulnerable <- household %>%
    group_by(education_head, has_employer_insurance) %>%
    summarise(n = n(), pct_cannot_handle = mean(!unexpected_expense,
        na.rm = TRUE) * 100, mean_income = mean(income, na.rm = TRUE),
        .groups = "drop") %>%
    filter(pct_cannot_handle > 50) %>%
    arrange(desc(pct_cannot_handle))

print("\nMost Financially Vulnerable Groups (>50% cannot handle $400 expense):")
```

```
[1] "\nMost Financially Vulnerable Groups (>50% cannot handle $400 expense):"
```

```r
print(vulnerable)
```

```
# A tibble: 10 x 5
   education_head has_employer_insurance     n pct_cannot_handle mean_income
            <dbl>                  <dbl> <int>             <dbl>       <dbl>
 1              3                      0   353              83.3      71776.
 2              1                      1   801              83.1      71813.
 3              3                      1   835              82.6      71696.
 4              2                      1   898              82.6      70831.
 5              5                      1   792              82.1      71722.
 6              5                      0   360              81.9      68793.
 7              4                      0   374              79.9      69831.
 8              2                      0   344              79.9      70452.
 9              1                      0   369              79.7      75810.
10              4                      1   874              78.4      71834.
```

Interpretation: The ability to handle unexpected expenses is a key indicator of financial security. Large percentages of lower-income households lacking this cushion reveals financial fragility that can spiral into debt or deprivation when emergencies occur.

2.3 7. Voter Turnout Patterns

Dataset: voter_turnout_simple.csv

Description: State-level voter turnout data from recent elections.

Variables: - `state`: State name (chr) - `turnout_2020`: Turnout percentage in 2020 (dbl) - `turnout_2016`: Turnout percentage in 2016 (dbl) - `population_millions`: State population in millions (dbl)

2.3.1 7.1 Turnout Overview

```r
# Load the dataset
turnout <- read_csv("voter_turnout_simple.csv")
```

```
Rows: 10 Columns: 4
-- Column specification --------------------------------------------------------
Delimiter: ","
chr (1): state
dbl (3): turnout_2020, turnout_2016, population_millions

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.
```

```r
# Explore the data
glimpse(turnout)
```

```
Rows: 10
Columns: 4
$ state              <chr> "Alabama", "Alaska", "Arizona", "Arkansas", "Calif~
$ turnout_2020       <dbl> 63.1, 58.9, 60.0, 54.9, 64.5, 76.4, 65.2, 66.0, 66~
$ turnout_2016       <dbl> 59.0, 61.5, 56.0, 53.2, 58.4, 71.9, 65.7, 61.8, 65~
$ population_millions <dbl> 5.0, 0.7, 7.3, 3.0, 39.5, 5.8, 3.6, 1.0, 21.5, 10.7
```

```r
# Basic summary statistics for both years
turnout_summary <- turnout %>%
    summarise(n_states = n(), mean_2020 = mean(turnout_2020,
        na.rm = TRUE), median_2020 = median(turnout_2020, na.rm = TRUE),
        sd_2020 = sd(turnout_2020, na.rm = TRUE), mean_2016 = mean(turnout_2016,
            na.rm = TRUE), median_2016 = median(turnout_2016,
            na.rm = TRUE), sd_2016 = sd(turnout_2016, na.rm = TRUE))

print(turnout_summary)
```

```
# A tibble: 1 x 7
  n_states mean_2020 median_2020 sd_2020 mean_2016 median_2016 sd_2016
     <int>     <dbl>       <dbl>   <dbl>     <dbl>       <dbl>   <dbl>
1       10      64.1        64.8    5.72      61.2        60.4    5.36
```

Work with Claude to calculate summary statistics for voter turnout in both 2016 and 2020. What changed between elections?

Prompt to Claude: Using the turnout dataset, calculate comprehensive summary statistics for 2016 and 2020 turnout. Compare the two elections - did turnout increase overall? Was there more or less variation between states? Calculate the change for each state. Use tidyverse.

```r
# Comprehensive turnout comparison
turnout_comparison <- turnout %>%
    mutate(change = turnout_2020 - turnout_2016, pct_change = (turnout_2020 -
        turnout_2016)/turnout_2016 * 100)

# Summary of changes
change_summary <- turnout_comparison %>%
    summarise(mean_change = mean(change), median_change = median(change),
        sd_change = sd(change), min_change = min(change), max_change = max(change),
```

```r
        all_increased = all(change > 0), n_increased = sum(change >
            0), n_decreased = sum(change < 0))

print("Turnout Change from 2016 to 2020:")
```

```
[1] "Turnout Change from 2016 to 2020:"
```

```r
print(change_summary)
```

```
# A tibble: 1 x 8
  mean_change median_change sd_change min_change max_change all_increased
        <dbl>         <dbl>     <dbl>      <dbl>      <dbl> <lgl>
1        2.92          4.05      2.97       -2.6       6.80 FALSE
# i 2 more variables: n_increased <int>, n_decreased <int>
```

```r
# States with biggest changes
print("\nBiggest turnout increases:")
```

```
[1] "\nBiggest turnout increases:"
```

```r
turnout_comparison %>%
    arrange(desc(change)) %>%
    select(state, turnout_2016, turnout_2020, change) %>%
    head(3) %>%
    print()
```

```
# A tibble: 3 x 4
  state      turnout_2016 turnout_2020 change
  <chr>             <dbl>        <dbl>  <dbl>
1 Georgia            59.4         66.2   6.80
2 California         58.4         64.5   6.1
3 Colorado           71.9         76.4   4.5
```

```r
print("\nAny turnout decreases?")
```

```
[1] "\nAny turnout decreases?"
```

```r
turnout_comparison %>%
    filter(change < 0) %>%
    select(state, turnout_2016, turnout_2020, change) %>%
    print()
```

```
# A tibble: 2 x 4
  state        turnout_2016 turnout_2020 change
  <chr>               <dbl>        <dbl>  <dbl>
1 Alaska               61.5         58.9   -2.6
2 Connecticut          65.7         65.2   -0.5
```

```r
# Variability comparison
print(paste("\n2016 CV:", round(turnout_summary$sd_2016/turnout_summary$mean_2016,
    3)))
```

```
[1] "\n2016 CV: 0.088"
```

```r
print(paste("2020 CV:", round(turnout_summary$sd_2020/turnout_summary$mean_2020,
    3)))
```

```
[1] "2020 CV: 0.089"
```

Interpretation: The 2020 election saw historically high turnout, driven by high stakes, polarization, and expanded voting access (mail-in voting). Comparing variation shows whether turnout increases were uniform or concentrated in certain states.

2.3.2 7.2 State-Level Patterns

Ask Claude to help you identify which states had the highest and lowest turnout in both elections. How consistent are state rankings between 2016 and 2020?

Prompt to Claude: Identify the top 3 and bottom 3 states for turnout in each election. Then check how consistent state rankings are between 2016 and 2020 using correlation. Do high-turnout states stay high? Use tidyverse.

```r
# Rankings for each year
turnout_rankings <- turnout %>%
    mutate(rank_2020 = rank(desc(turnout_2020)), rank_2016 = rank(desc(turnout_2016))) %>%
    arrange(rank_2020)

print("2020 Turnout Rankings:")
```

```
[1] "2020 Turnout Rankings:"
```

```r
turnout_rankings %>%
    select(state, turnout_2020, rank_2020) %>%
    head(3) %>%
    print()
```

```
# A tibble: 3 x 3
  state    turnout_2020 rank_2020
  <chr>           <dbl>     <dbl>
1 Colorado         76.4       1
2 Florida          66.2       2.5
3 Georgia          66.2       2.5
```

```r
print("\n2020 Lowest Turnout:")
```

```
[1] "\n2020 Lowest Turnout:"
```

```r
turnout_rankings %>%
    select(state, turnout_2020, rank_2020) %>%
    tail(3) %>%
    print()
```

```
# A tibble: 3 x 3
  state    turnout_2020 rank_2020
  <chr>           <dbl>     <dbl>
1 Arizona          60           8
2 Alaska           58.9         9
3 Arkansas         54.9        10
```

```r
# Check consistency
rank_correlation <- cor(turnout_rankings$rank_2020, turnout_rankings$rank_2016)
print(paste("\nRank correlation between years:", round(rank_correlation,
    3)))
```

```
[1] "\nRank correlation between years: 0.742"
```

```r
# States with biggest rank changes
turnout_rankings %>%
    mutate(rank_change = rank_2016 - rank_2020) %>%
    arrange(desc(abs(rank_change))) %>%
    select(state, rank_2016, rank_2020, rank_change) %>%
    head(3) %>%
    print()
```

```
# A tibble: 3 x 4
  state       rank_2016 rank_2020 rank_change
  <chr>           <dbl>     <dbl>       <dbl>
1 Alaska              5       9          -4
2 Georgia             6       2.5         3.5
3 Connecticut         2       5          -3
```

```r
# Direct turnout correlation
turnout_correlation <- cor(turnout$turnout_2020, turnout$turnout_2016)
print(paste("Turnout correlation between years:", round(turnout_correlation,
    3)))
```

```
[1] "Turnout correlation between years: 0.858"
```

Interpretation: High correlation between years suggests persistent state-level factors affecting turnout (voting laws, civic culture, demographics). States with big rank changes may have had specific mobilization efforts or law changes.

2.3.3 7.3 Turnout Changes

Work with Claude to calculate how turnout changed from 2016 to 2020 for each state. Which states saw the biggest increases or decreases?

Prompt to Claude: Calculate turnout changes from 2016 to 2020 for each state. Create categories for the size of change (large increase, moderate increase, etc.). Also explore if turnout changes relate to state population size. Use tidyverse.

```r
# Categorize turnout changes
turnout_changes <- turnout %>%
    mutate(change = turnout_2020 - turnout_2016, change_category = cut(change,
        breaks = c(-Inf, 0, 2, 5, Inf), labels = c("Decrease",
            "Small increase (0-2%)", "Moderate increase (2-5%)",
```

```r
            "Large increase (>5%)"))) %>%
    arrange(desc(change))

# Distribution of changes
print("Distribution of Turnout Changes:")
```

```
[1] "Distribution of Turnout Changes:"
```

```r
turnout_changes %>%
    count(change_category) %>%
    print()
```

```
# A tibble: 4 x 2
  change_category              n
  <fct>                    <int>
1 Decrease                     2
2 Small increase (0-2%)        2
3 Moderate increase (2-5%)     4
4 Large increase (>5%)         2
```

```r
# States by change category
print("\nStates with Large Increases:")
```

```
[1] "\nStates with Large Increases:"
```

```r
turnout_changes %>%
    filter(change_category == "Large increase (>5%)") %>%
    select(state, change) %>%
    print()
```

```
# A tibble: 2 x 2
  state      change
  <chr>       <dbl>
1 Georgia      6.80
2 California   6.1
```

```r
# Relationship with population
pop_correlation <- cor(turnout_changes$change, turnout_changes$population_millions)
print(paste("\nCorrelation between turnout change and population:",
    round(pop_correlation, 3)))
```

```
[1] "\nCorrelation between turnout change and population: 0.397"
```

```r
# Summary by population size
turnout_changes %>%
    mutate(pop_category = ifelse(population_millions > median(population_millions),
        "Large states", "Small states")) %>%
    group_by(pop_category) %>%
    summarise(n = n(), mean_change = mean(change), median_change = median(change),
        .groups = "drop") %>%
    print()
```

```
# A tibble: 2 x 4
  pop_category     n mean_change median_change
  <chr>        <int>       <dbl>         <dbl>
1 Large states     5        4.46          4.5
2 Small states     5        1.38          1.70
```

Interpretation: Turnout changes reflect various factors including competitiveness, mobilization efforts, voting law changes, and demographic shifts. The relationship (or lack thereof) with population size indicates whether changes were driven by state-specific or national factors.

2.4 8. Legislative Productivity

Dataset: rollcalls.csv

Description: Individual member votes on congressional bills.

Variables: - `congress`: Congress number (int) - `bill_id`: Unique bill identifier (int) - `member_id`: Member identifier (int) - `party`: Political party (chr) - `ideology`: Member ideology score (dbl) - `bill_ideology`: Bill ideology score (dbl) - `vote`: Vote choice (chr) - `district_partisanship`: District lean (dbl)

2.4.1 8.1 Understanding Voting Patterns

```r
# Load the dataset
rollcalls <- read_csv("rollcalls.csv")
```

```
Rows: 3000 Columns: 8
-- Column specification --------------------------------------------------------
Delimiter: ","
chr (1): party
dbl (7): congress, bill_id, member_id, ideology, bill_ideology, vote, distri...

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.
```

```r
# Look at the data
glimpse(rollcalls)
```

```
Rows: 3,000
Columns: 8
$ congress              <dbl> 114, 117, 116, 114, 114, 116, 116, 118, 114, 116~
$ bill_id               <dbl> 500, 238, 494, 352, 445, 177, 621, 40, 426, 165,~
$ member_id             <dbl> 66, 409, 327, 467, 532, 487, 290, 245, 464, 90, ~
$ party                 <chr> "Dem", "Rep", "Dem", "Dem", "Rep", "Rep", "Rep",~
$ ideology              <dbl> 0.20949025, 0.41277467, -0.27790813, 0.10842921,~
$ bill_ideology         <dbl> 0.43897533, -0.33229308, -0.27514849, -0.1348527~
$ vote                  <dbl> 0, 1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, ~
$ district_partisanship <dbl> 11.9, -20.8, 12.0, 15.8, -11.3, -25.6, 2.2, -13.~
```

```r
# Basic vote distribution
vote_summary <- rollcalls %>%
    count(vote) %>%
    mutate(pct = n/sum(n) * 100)

print("Overall voting patterns:")
```

```
[1] "Overall voting patterns:"
```

```r
print(vote_summary)
```

```
# A tibble: 2 x 3
   vote     n   pct
  <dbl> <int> <dbl>
1     0  1515  50.5
2     1  1485  49.5
```

```r
# Votes by congress
by_congress <- rollcalls %>%
    group_by(congress) %>%
    summarise(n_votes = n(), n_bills = n_distinct(bill_id), n_members = n_distinct(member_id)
        pct_yes = mean(vote == "Yes", na.rm = TRUE) * 100, .groups = "drop")

print("Voting by Congress:")
```

```
[1] "Voting by Congress:"
```

```r
print(by_congress)
```

```
# A tibble: 5 x 5
  congress n_votes n_bills n_members pct_yes
     <dbl>   <int>   <int>     <int>   <dbl>
1      114     626     419       379       0
2      115     596     428       355       0
3      116     590     425       354       0
4      117     603     419       362       0
5      118     585     400       360       0
```

Use Claude to calculate basic statistics about congressional voting patterns, including passage rates and participation.

Prompt to Claude: Analyze the rollcalls voting data. Calculate what percentage of votes are Yes/No/Abstain, how this varies by party, and explore the relationship between member ideology and voting patterns. Use tidyverse.

```r
# Voting patterns by party
party_voting <- rollcalls %>%
    group_by(party) %>%
    summarise(n_votes = n(), pct_yes = mean(vote == "Yes", na.rm = TRUE) *
        100, pct_no = mean(vote == "No", na.rm = TRUE) * 100,
        pct_abstain = mean(vote == "Abstain", na.rm = TRUE) *
            100, .groups = "drop")

print("Voting Patterns by Party:")
```

```
[1] "Voting Patterns by Party:"
```

```r
print(party_voting)
```

```
# A tibble: 2 x 5
  party n_votes pct_yes pct_no pct_abstain
  <chr>   <int>   <dbl>  <dbl>       <dbl>
1 Dem      1541       0      0           0
2 Rep      1459       0      0           0
```

```r
# Ideology and voting - using quantiles instead of
# equal-width bins
ideology_voting <- rollcalls %>%
    filter(vote %in% c("Yes", "No"), !is.na(ideology)) %>%
    mutate(vote_numeric = ifelse(vote == "Yes", 1, 0), ideology_quintile = ntile(ideology,
        5)) %>%
    group_by(ideology_quintile) %>%
    summarise(n = n(), mean_ideology = mean(ideology, na.rm = TRUE),
        pct_yes = mean(vote_numeric) * 100, .groups = "drop")

print("\nVoting by Ideology Quintile:")
```

```
[1] "\nVoting by Ideology Quintile:"
```

```r
print(ideology_voting)
```

```
# A tibble: 0 x 4
# i 4 variables: ideology_quintile <int>, n <int>, mean_ideology <dbl>,
#   pct_yes <dbl>
```

```r
# Member participation rates
member_participation <- rollcalls %>%
    group_by(member_id) %>%
    summarise(n_bills = n_distinct(bill_id), pct_abstain = mean(vote ==
        "Abstain") * 100, ideology = first(ideology), party = first(party),
        .groups = "drop")

participation_summary <- member_participation %>%
    summarise(mean_abstain = mean(pct_abstain), median_abstain = median(pct_abstain),
        high_abstainers = sum(pct_abstain > 10))

print("\nMember Participation:")
```

```
[1] "\nMember Participation:"
```

```
print(participation_summary)
```

```
# A tibble: 1 x 3
  mean_abstain median_abstain high_abstainers
         <dbl>          <dbl>           <int>
1            0              0               0
```

Interpretation: Voting patterns reveal party discipline, ideological consistency, and member engagement. High abstention rates might indicate strategic avoidance or absence.

2.4.2 8.2 Party Voting Patterns

Work with Claude to analyze how members vote based on their party affiliation. Do Democrats and Republicans vote differently on bills?

Prompt to Claude: Analyze party-line voting. For each bill, calculate what percentage of Democrats voted Yes vs Republicans. Identify bills with strongest party splits and those with bipartisan agreement. Use tidyverse.

```
# Party voting by bill
bill_party_voting <- rollcalls %>%
  filter(vote %in% c("Yes", "No")) %>%
  group_by(bill_id, party) %>%
  summarise(
    n_votes = n(),
    pct_yes = mean(vote == "Yes") * 100,
    .groups = 'drop'
  ) %>%
  pivot_wider(names_from = party, values_from = pct_yes, values_fill = 0)

# Check which columns exist and calculate party split accordingly
if (all(c("Democrat", "Republican") %in% names(bill_party_voting))) {
  bill_party_voting <- bill_party_voting %>%
    mutate(
      party_split = abs(Democrat - Republican),
      vote_type = case_when(
        party_split > 70 ~ "Strong party split",
        party_split > 30 ~ "Moderate party split",
        party_split <= 30 ~ "Bipartisan"
```

```r
      )
    )
} else {
  # If parties don't match expected names, create simplified version
  party_cols <- setdiff(names(bill_party_voting), "bill_id")
  if (length(party_cols) >= 2) {
    bill_party_voting <- bill_party_voting %>%
      mutate(
        party_split = abs(.[[party_cols[1]]] - .[[party_cols[2]]]),
        vote_type = case_when(
          party_split > 70 ~ "Strong party split",
          party_split > 30 ~ "Moderate party split",
          party_split <= 30 ~ "Bipartisan"
        )
      )
  } else {
    bill_party_voting <- bill_party_voting %>%
      mutate(
        party_split = 0,
        vote_type = "Single party data"
      )
  }
}

# Distribution of vote types
vote_type_summary <- bill_party_voting %>%
  count(vote_type) %>%
  mutate(pct = n / sum(n) * 100)

print("Distribution of Bill Vote Types:")
```

```
[1] "Distribution of Bill Vote Types:"
```

```r
print(vote_type_summary)
```

```
# A tibble: 0 x 3
# i 3 variables: vote_type <chr>, n <int>, pct <dbl>
```

```r
# Most partisan bills
print("\nMost Partisan Bills (biggest party splits):")
```

```
[1] "\nMost Partisan Bills (biggest party splits):"
```

```r
if (all(c("Democrat", "Republican") %in% names(bill_party_voting))) {
  bill_party_voting %>%
    arrange(desc(party_split)) %>%
    select(bill_id, Democrat, Republican, party_split) %>%
    head(5) %>%
    print()
} else {
  bill_party_voting %>%
    arrange(desc(party_split)) %>%
    head(5) %>%
    print()
}
```

```
# A tibble: 0 x 4
# i 4 variables: bill_id <dbl>, n_votes <int>, party_split <dbl>,
#   vote_type <chr>
```

```r
# Most bipartisan bills
print("\nMost Bipartisan Bills (smallest party splits):")
```

```
[1] "\nMost Bipartisan Bills (smallest party splits):"
```

```r
if (all(c("Democrat", "Republican") %in% names(bill_party_voting))) {
  bill_party_voting %>%
    filter(Democrat > 50 | Republican > 50) %>%  # At least one party mostly supports
    arrange(party_split) %>%
    select(bill_id, Democrat, Republican, party_split) %>%
    head(5) %>%
    print()
} else {
  bill_party_voting %>%
    arrange(party_split) %>%
    head(5) %>%
    print()
}
```

```
# A tibble: 0 x 4
# i 4 variables: bill_id <dbl>, n_votes <int>, party_split <dbl>,
#   vote_type <chr>
```

```r
# Party unity scores
party_unity <- rollcalls %>%
  filter(vote %in% c("Yes", "No")) %>%
  group_by(party) %>%
  mutate(party_position = ifelse(mean(vote == "Yes") > 0.5, "Yes", "No")) %>%
  ungroup() %>%
  mutate(votes_with_party = vote == party_position) %>%
  group_by(member_id, party) %>%
  summarise(
    unity_score = mean(votes_with_party) * 100,
    n_votes = n(),
    .groups = 'drop'
  )

unity_summary <- party_unity %>%
  group_by(party) %>%
  summarise(
    mean_unity = mean(unity_score),
    median_unity = median(unity_score),
    sd_unity = sd(unity_score),
    .groups = 'drop'
  )

print("\nParty Unity Scores:")
```

```
[1] "\nParty Unity Scores:"
```

```r
print(unity_summary)
```

```
# A tibble: 0 x 4
# i 4 variables: party <chr>, mean_unity <dbl>, median_unity <dbl>,
#   sd_unity <dbl>
```

Interpretation: Party-line voting has increased in recent decades. Bills with high party splits often involve ideological issues, while bipartisan bills typically address non-controversial or crisis issues.

2.4.3 8.3 Ideology and Voting

Ask Claude to help you explore how member ideology relates to voting patterns. Do more extreme members vote differently than moderates?

Prompt to Claude: Examine how member ideology affects voting behavior. Compare voting patterns of moderates (ideology near 0) versus extremists (far from 0). Also analyze how often members vote against bills that don't match their ideology. Use tidyverse.

```r
# Categorize members by ideology
member_ideology <- rollcalls %>%
    mutate(ideology_group = case_when(abs(ideology) < 0.2 ~ "Moderate",
        abs(ideology) < 0.5 ~ "Somewhat partisan", TRUE ~ "Highly partisan"),
        ideological_distance = abs(ideology - bill_ideology))

# Voting patterns by ideology group
ideology_patterns <- member_ideology %>%
    filter(vote %in% c("Yes", "No")) %>%
    group_by(ideology_group) %>%
    summarise(n_votes = n(), pct_yes = mean(vote == "Yes") *
        100, pct_no = mean(vote == "No") * 100, avg_ideology = mean(abs(ideology)),
        .groups = "drop") %>%
    arrange(avg_ideology)

print("Voting by Ideology Group:")
```

```
[1] "Voting by Ideology Group:"
```

```r
print(ideology_patterns)
```

```
# A tibble: 0 x 5
# i 5 variables: ideology_group <chr>, n_votes <int>, pct_yes <dbl>,
#   pct_no <dbl>, avg_ideology <dbl>
```

```r
# Ideological consistency
ideological_voting <- member_ideology %>%
    filter(vote %in% c("Yes", "No")) %>%
    mutate(ideologically_consistent = case_when(ideology > 0 &
        bill_ideology > 0 & vote == "Yes" ~ TRUE, ideology <
        0 & bill_ideology < 0 & vote == "Yes" ~ TRUE, ideology >
        0 & bill_ideology < 0 & vote == "No" ~ TRUE, ideology <
        0 & bill_ideology > 0 & vote == "No" ~ TRUE, TRUE ~ FALSE)) %>%
    group_by(ideology_group) %>%
    summarise(consistency_rate = mean(ideologically_consistent) *
        100, n_votes = n(), .groups = "drop")

print("\nIdeological Consistency by Member Type:")
```

```
[1] "\nIdeological Consistency by Member Type:"
```

```r
print(ideological_voting)
```

```
# A tibble: 0 x 3
# i 3 variables: ideology_group <chr>, consistency_rate <dbl>, n_votes <int>
```

```r
# Cross-party voting
cross_party <- rollcalls %>%
    filter(vote %in% c("Yes", "No")) %>%
    group_by(member_id, party, ideology) %>%
    summarise(n_votes = n(), .groups = "drop") %>%
    mutate(is_moderate = abs(ideology) < 0.3)

moderate_analysis <- cross_party %>%
    group_by(party, is_moderate) %>%
    summarise(n_members = n(), avg_ideology = mean(ideology),
        .groups = "drop")

print("\nModerate vs Partisan Members by Party:")
```

```
[1] "\nModerate vs Partisan Members by Party:"
```

```r
print(moderate_analysis)
```

```
# A tibble: 0 x 4
# i 4 variables: party <chr>, is_moderate <lgl>, n_members <int>,
#   avg_ideology <dbl>
```

Interpretation: Ideological extremity correlates with partisan voting behavior. Moderates are more likely to cross party lines and vote based on specific bill content rather than party position. This dynamic affects legislative outcomes and coalition building.