Case Study

Load in the important libraries.

```
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   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- 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 become errors
```

```
library(dplyr)
library(tidyr)
library(ggplot2)
library(lubridate)
```

Problem 1 – Data handling, analysis and plotting

The first problem of the case study builds on the data in the files p01-02_portfolio.csv and p01-02_rates.csv. One file contains membership information for a Group Life portfolio and one has information on the rates which should be charged.

```
# Load the CSV file
portfolio_data <- read_delim(
  "Case Study/data/p01-02_portfolio.csv",
  delim = ";",
  show_col_types = FALSE
)

# View the data
head(portfolio_data)
```

```
## # A tibble: 6 x 5
##   SchemeName Date.of.Birth Gender DeathSI Industry
##   <chr>      <chr>         <chr>  <chr>   <chr>
## 1 Scheme2    29.05.1949    F      <NA>    Government & Public Administration
## 2 Scheme2    07.09.1950    F      <NA>    Government & Public Administration
## 3 Scheme2    27.09.1956    F      <NA>    Government & Public Administration
## 4 Scheme2    18.02.1942    F      <NA>    Government & Public Administration
## 5 Scheme2    31.07.1951    F      <NA>    Government & Public Administration
## 6 Scheme2    10.07.1960    F      <NA>    Government & Public Administration
```

```r
# Load the CSV file
rates_data <- read_delim("Case Study/data/p01-02_rates.csv",
                         delim = ";",
                         show_col_types = FALSE)

# View the data
head(rates_data)
```

```
## # A tibble: 6 x 3
##     Age Gender  Rate
##   <dbl> <chr>  <dbl>
## 1    18 M       0.32
## 2    19 M       0.32
## 3    20 M       0.32
## 4    21 M       0.31
## 5    22 M       0.31
## 6    23 M       0.31
```

rates_data

```r
# Count occurrences of each combination of Gender and Age
duplicates <- rates_data %>%
  group_by(Gender, Age) %>%
  summarise(count = n()) %>%
  filter(count > 1)
```

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

```r
# Check if any duplicates exist
if (nrow(duplicates) == 0) {
  message("Sanity Check Passed: 'Gender' and 'Age' form a unique key.")
} else {
  message("Sanity Check Failed: There are duplicate combinations of 'Gender' and 'Age'.")
  print(duplicates)
}
```

```
## Sanity Check Passed: 'Gender' and 'Age' form a unique key.
```

Question a.

Read the data from the two files into R's memory. The rates are applicable to each individual in the portfolio, depending on that individual's age and gender. Combine the two datasets into a single table by looking up the rate for each line of the portfolio.

```r
# Step 1: Convert the Date.of.Birth column to Date format
portfolio_data$Date.of.Birth <- dmy(portfolio_data$Date.of.Birth)  # dmy is used for "day-month-year" f

# Step 2: Calculate the time difference in years
portfolio_data$age <- ceiling(interval(portfolio_data$Date.of.Birth, today()) / years(1))

# Step 3: View the updated data with age column
head(portfolio_data)
```

```
## # A tibble: 6 x 6
##   SchemeName Date.of.Birth Gender DeathSI Industry                        age
##   <chr>      <date>        <chr>  <chr>   <chr>                         <dbl>
## 1 Scheme2    1949-05-29    F      <NA>    Government & Public Administrat~    76
## 2 Scheme2    1950-09-07    F      <NA>    Government & Public Administrat~    75
## 3 Scheme2    1956-09-27    F      <NA>    Government & Public Administrat~    68
## 4 Scheme2    1942-02-18    F      <NA>    Government & Public Administrat~    83
## 5 Scheme2    1951-07-31    F      <NA>    Government & Public Administrat~    74
## 6 Scheme2    1960-07-10    F      <NA>    Government & Public Administrat~    65
```

```r
# Inner join the two datasets
combined_data <- inner_join(portfolio_data,
                            rates_data,
                            by = c("age" = "Age", "Gender" = "Gender"),
)

# Check if the row count of the joined data matches the original portfolio data
if (nrow(combined_data) == nrow(portfolio_data)) {
  message("Sanity Check Passed: The row count of combined_data matches portfolio_data.")
} else {
  message("Sanity Check Failed: The row count of combined_data does not match portfolio_data.")
  message("Rows in portfolio_data: ", nrow(portfolio_data))
  message("Rows in combined_data: ", nrow(combined_data))
}
```

```
## Sanity Check Failed: The row count of combined_data does not match portfolio_data.
```

```
## Rows in portfolio_data: 177922
```

```
## Rows in combined_data: 145607
```

This sanity check is expected to fail because the rates data is cutoff at 70. Do not consider people over 70 in this analysis.

```r
# Find rows in portfolio_data that do not have a match in rates_data
missing_matches <- anti_join(portfolio_data,
                             rates_data,
                             by = c("age" = "Age", "Gender" = "Gender"))

# Check how many rows are missing
nrow(missing_matches)
```

Investigate missing matches

```
## [1] 32315
```

Question b.

Group the Industry field into common-sense based groupings and determine the mean, standard deviation and quantiles of DeathSI for each of your industry groups.

```
industry_counts <- combined_data %>%
  count(Industry) %>%
  arrange(desc(n))

# View the result
print(industry_counts)
```

```
## # A tibble: 33 x 2
##    Industry                           n
##    <chr>                          <int>
##  1 <NA>                           87405
##  2 Government & Public Administration 29475
##  3 Other                          14130
##  4 Sporting Club                   2177
##  5 Ex-Services Club                1442
##  6 BSS-Business Services           1179
##  7 MAN-Manufacturing               1065
##  8 EDN-Education                    874
##  9 COM-Communication Serv.          817
## 10 FIN-Finance & Insurance          779
## # i 23 more rows
```

```
combined_data <- combined_data %>%
  mutate(
    Industry_Group = case_when(
      Industry %in% c("Government & Public Administration", "Ex-Services Club") ~ "Government and Publi
      Industry %in% c(
        "Sporting Club",
        "Golf Club",
        "Bowls Club",
        "Registered Club",
        "Surf Life Saving Club",
        "Workers Club",
        "Australian Rules Football Club",
        "Leagues Club",
        "Associated with Club Industry"
      ) ~ "Clubs and Associations",
      Industry %in% c(
        "BSS-Business Services",
        "FIN-Finance & Insurance",
        "Professional Services",
        "LAW-Solicitors/Barrister",
        "ENG-Engineers",
        "MGE-Medical Services Gen"
      ) ~ "Professional and Business Services",
      Industry %in% c(
        "MAN-Manufacturing",
        "CON-Construction",
```

```
        "ELE-Electricians",
        "VEH-Vehicle Industry",
        "WEO-Wholesale Trades"
      ) ~ "Manufacturing, Construction, and Trades",
      Industry %in% c(
        "EDN-Education",
        "HEA-Health Industry",
        "MGE-Medical Services Gen"
      ) ~ "Education and Health",
      Industry %in% c(
        "RTL-Retail Trade",
        "ACR-Accom. Cafes & Rests",
        "FOO-Food",
        "Hospitality"
      ) ~ "Retail, Hospitality, and Food",
      Industry %in% c("AGR-Farming/Agriculture", "EGW
-Electric/Gas/Water") ~ "Agriculture and Utilities",
Industry == "Other" ~ "Other",
TRUE ~ "Uncategorized"  # Catch any uncategorized industries
    )
  )


# View the newly grouped data
print(combined_data)
```

```
## # A tibble: 145,607 x 8
##    SchemeName Date.of.Birth Gender DeathSI Industry     age  Rate Industry_Group
##    <chr>      <date>        <chr>  <chr>   <chr>      <dbl> <dbl> <chr>
##  1 Scheme2    1956-09-27    F      <NA>    Governmen~    68  5.96 Government an~
##  2 Scheme2    1960-07-10    F      <NA>    Governmen~    65  4.35 Government an~
##  3 Scheme2    1954-12-24    F      <NA>    Governmen~    70  7.34 Government an~
##  4 Scheme2    1958-02-28    F      <NA>    Governmen~    67  5.36 Government an~
##  5 Scheme2    1968-09-12    F      <NA>    Governmen~    57  1.91 Government an~
##  6 Scheme2    1966-11-21    F      <NA>    Governmen~    58  2.11 Government an~
##  7 Scheme2    1957-03-05    F      <NA>    Governmen~    68  5.96 Government an~
##  8 Scheme2    1966-02-01    F      <NA>    Governmen~    59  2.34 Government an~
##  9 Scheme2    1975-02-10    F      <NA>    Governmen~    50  0.96 Government an~
## 10 Scheme2    1966-11-21    F      <NA>    Governmen~    58  2.11 Government an~
## # i 145,597 more rows
```

```
industry_counts <- combined_data %>%
  count(Industry_Group) %>%
  arrange(desc(n))

# View the result
print(industry_counts)
```

```
## # A tibble: 9 x 2
##   Industry_Group                              n
##   <chr>                                   <int>
## 1 Uncategorized                           88467
## 2 Government and Public Services          30917
```

```
## 3 Other                                     14130
## 4 Clubs and Associations                     4805
## 5 Manufacturing, Construction, and Trades    2376
## 6 Professional and Business Services         2364
## 7 Education and Health                       1242
## 8 Retail, Hospitality, and Food               974
## 9 Agriculture and Utilities                   332
```

1. ```
   # Check the type of DeathSI
   typeof(combined_data$DeathSI)
   ```

   ```
   ## [1] "character"
   ```

   ```
   # Count the number of NA values in DeathSI when it was character type
   na_count <- sum(is.na(combined_data$DeathSI))
   na_count
   ```

   ```
   ## [1] 14204
   ```

   ```
   # Count the number of "NA" string values in DeathSI when it was character type
   na_string_count <- sum(combined_data$DeathSI == "NA", na.rm = TRUE)
   na_string_count
   ```

   ```
   ## [1] 0
   ```

   ```
   # Remove apostrophes and convert the DeathSI column from character to numeric
   combined_data$DeathSI <- as.numeric(gsub("'", "", combined_data$DeathSI))

   # Check the type of DeathSI
   typeof(combined_data$DeathSI)
   ```

   ```
   ## [1] "double"
   ```

   ```
   # Count the number of NA values in DeathSI when it is the double type
   na_count <- sum(is.na(combined_data$DeathSI))
   na_count
   ```

   ```
   ## [1] 14204
   ```

```
# Calculate mean, standard deviation, and quantiles for each industry group
summary_stats <- combined_data %>%
  group_by(Industry_Group) %>%
  summarize(
    mean_value = mean(DeathSI, na.rm = TRUE),
    sd_value = sd(DeathSI, na.rm = TRUE),
    q25 = quantile(DeathSI, 0.25, na.rm = TRUE),
    median_value = median(DeathSI, na.rm = TRUE),
    q75 = quantile(DeathSI, 0.75, na.rm = TRUE)
  )

# View the result
print(summary_stats)
```
```

```
## # A tibble: 9 x 6
##   Industry_Group                     mean_value sd_value    q25 median_value    q75
##   <chr>                                   <dbl>    <dbl>  <dbl>        <dbl>  <dbl>
## 1 Agriculture and Utilities             174894.  224538. 3.39e4       142202 2.01e5
## 2 Clubs and Associations                279598.  184125. 2.00e5       220613 3.02e5
## 3 Education and Health                  350616.  234855. 1.79e5       320172 4.88e5
## 4 Government and Public Services        224447.   98490. 1.5 e5       220000 3    e5
## 5 Manufacturing, Construction, a~       303140.  236342. 1.35e5       263191 4.07e5
## 6 Other                                 262005.  114205. 2.08e5       245716 2.86e5
## 7 Professional and Business Serv~       449616.  323093. 2.52e5       383741 5.52e5
## 8 Retail, Hospitality, and Food         320963.  205977. 2.05e5       248697 3.89e5
## 9 Uncategorized                         228127.  214916. 9.07e4       170742 2.85e5
```

Question c.

The following code performs a Monte Carlo simulation on the data you have loaded and combined in Question a.:

```
set.seed(1234)
nsim <- 1000
res <- lapply(1:nsim, function(i,...) {
  x <- ifelse(
    runif(dim(combined_data)[1]) < combined_data$Rate / 1000,
    combined_data$DeathSI,
    0
  );
  list(cost = sum(x), count = length(x[x > 0]))
})
```

Apply this simulation to each scheme in the dataset you were provided, running 1000 simulations per scheme. Produce a plot of the simulated outcomes ("cost"). Your plot should show:

- a separate histogram per scheme;
- all 5 histograms below each other so that they can be easily compared;
- vertical lines in each graph indicating the median, mean and 99.5th percentile of each distribution.

Remove rows where DeathSI is NA for Monte Carlo simulation.

```
# Subset combined_data where DeathSI is not NA
combined_data_death_si_non_na <- subset(combined_data, !is.na(DeathSI))
```

```
monte_carlo_simulation <- function(data, nsim = 1000, seed = 1234) {
  # Set the seed for reproducibility
  set.seed(seed)

  # Perform the simulation
  res <- lapply(1:nsim, function(i, ...) {
    x <- ifelse(runif(dim(data)[1]) < data$Rate / 1000, data$DeathSI, 0)
```

```r
    # Return the cost and count as a list
    list(cost = sum(x), count = length(x[x > 0]))
  })

  # Return the result of the simulation
  return(res)
}

# Get the unique values in the SchemeName column
unique_schemes <- unique(combined_data$SchemeName)

# Print the unique values
unique_schemes
```

```
## [1] "Scheme2" "Scheme1" "Scheme3" "Scheme5"
```

```r
# Filter rows where SchemeName is "Scheme_1", "Scheme_2", "Scheme_3", "Scheme_5"
combined_data_scheme_1 <- subset(combined_data_death_si_non_na, SchemeName == "Scheme1")
combined_data_scheme_2 <- subset(combined_data_death_si_non_na, SchemeName == "Scheme2")
combined_data_scheme_3 <- subset(combined_data_death_si_non_na, SchemeName == "Scheme3")
combined_data_scheme_5 <- subset(combined_data_death_si_non_na, SchemeName == "Scheme5")

# Sanity check that each subset has more than 0 rows
check_scheme_1 <- nrow(combined_data_scheme_1) > 0
check_scheme_2 <- nrow(combined_data_scheme_2) > 0
check_scheme_3 <- nrow(combined_data_scheme_3) > 0
check_scheme_5 <- nrow(combined_data_scheme_5) > 0

# Print the results
cat("Scheme 1 has more than 0 rows:", check_scheme_1, "\n")
```

```
## Scheme 1 has more than 0 rows: TRUE
```

```r
cat("Scheme 2 has more than 0 rows:", check_scheme_2, "\n")
```

```
## Scheme 2 has more than 0 rows: TRUE
```

```r
cat("Scheme 3 has more than 0 rows:", check_scheme_3, "\n")
```

```
## Scheme 3 has more than 0 rows: TRUE
```

```r
cat("Scheme 5 has more than 0 rows:", check_scheme_5, "\n")
```

```
## Scheme 5 has more than 0 rows: TRUE
```

```r
# Perform a monte carlo simulation for each scheme
monte_carlo_scheme_1_result <- monte_carlo_simulation(combined_data_scheme_1)
monte_carlo_scheme_2_result <- monte_carlo_simulation(combined_data_scheme_2)
monte_carlo_scheme_3_result <- monte_carlo_simulation(combined_data_scheme_3)
monte_carlo_scheme_5_result <- monte_carlo_simulation(combined_data_scheme_5)
```