Stats 102A - Homework 3 - Output File

Daren Sathasivam — 306229580

Homework questions and prompts copyright Miles Chen, Do not post, share, or distribute without permission.

To receive full credit the functions you write must pass all tests. We may conduct further tests that are not included on this page as well.

Academic Integrity Statement

By including this statement, I, **Daren Sathasivam**, declare that all of the work in this assignment is my own original work. At no time did I look at the code of other students nor did I search for code solutions online. I understand that plagiarism on any single part of this assignment will result in a 0 for the entire assignment and that I will be referred to the dean of students.

Part 1. Basic dplyr exercises

Install the package `fueleconomy` and load the dataset `vehicles`. Answer the following questions.

```
library(fueleconomy) # run install.packages("fueleconomy") if necessary
library(tidyverse) # run install.packages("tidyverse") if necessary
data(vehicles)
# head(vehicles)
```

a. How many unique vehicle makers (variable `make`) are included in the dataset?

```
# write your code here, the output displayed should answer the question.
# ?dplyr
n_distinct(vehicles$make)
```

```
## [1] 128
```

b. How many vehicles made in 2014 are represented in the dataset?

```
# write your code here, the output displayed should answer the question.
nrow(filter(vehicles, year == 2014))
```

```
## [1] 1214
```

c. For the year 2014, what was the average city mpg (gas mileage) for all compact cars? What was the average city mpg for midsize cars in 2014?

```
# write your code here, the output displayed should answer the question.
# unique(vehicles$class)
# Compacts
compact_avg <- vehicles %>%
  filter(year == 2014, class == "Compact Cars") %>%
  summarise(avg_cty = mean(cty))
compact_avg
```

```
## # A tibble: 1 x 1
```

```
##   avg_cty
##     <dbl>
## 1    23.9
```

```
# Midsize
midsize_avg <- vehicles %>%
  filter(year == 2014, class == "Midsize Cars") %>%
  summarise(avg_cty = mean(cty))
midsize_avg
```

```
## # A tibble: 1 x 1
##   avg_cty
##     <dbl>
## 1    23.5
```

d. For the year 2014, compare makers of midsize cars. Find the average city mpg of midsize cars for each manufacturer. For example, in 2014, Acura has 5 midsize cars with an average city mpg of 20.6, while Audi has 12 midsize cars with an average city mpg of 19.08.

Produce a table showing the city mpg for 2014 midsize cars for the 27 manufacturers represented in the table. Arrange the results in descending order, so that the manufacturer with the highest average mpg will be listed first.

```
# write your code here, the output displayed should answer the question.
# unique(vehicles$make)
# nrow(filter(vehicles, year == 2014, make == "Acura", class == "Midsize Cars"))
avg_2014_midsize <- vehicles %>%
  filter(year == 2014, class == "Midsize Cars") %>%
  group_by(make) %>%
  summarise(avg_city = mean(cty, na.rm = TRUE)) %>%
  arrange(desc(avg_city))
avg_2014_midsize
```

```
## # A tibble: 27 x 2
##    make       avg_city
##    <chr>         <dbl>
##  1 Nissan         38.9
##  2 Toyota         32.4
##  3 Ford           30.4
##  4 Honda          29.2
##  5 Mazda          27.6
##  6 Hyundai        27.4
##  7 Kia            26.4
##  8 Chevrolet      25.6
##  9 Lincoln        25.2
## 10 Volkswagen     24.7
## # i 17 more rows
```

e. Finally, for the years 1994, 1999, 2004, 2009, and 2014, find the average city mpg of midsize cars for each manufacturer for each year. Use tidyr to transform the resulting output so each manufacturer has one row, and five columns (a column for each year). Print out all the rows of the resulting tibble. You can use `print(tibble, n = 40)` to print 40 rows of a tibble.

```
#             make     1994     1999     2004     2009     2014
# 1          Acura       NA 16.50000 17.33333 17.00000 20.60000
# 2           Audi       NA 15.25000 16.20000 15.83333 19.08333
avg_years_midsize <- vehicles %>%
  filter(year %in% c(1994, 1999, 2004, 2009, 2014), class == "Midsize Cars") %>%
```

```
  group_by(year, make) %>%
  summarise(avg_cty = mean(cty, na.rm = TRUE)) %>%
  ungroup() %>%
  pivot_wider(names_from = year, values_from = avg_cty)
```

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

```
# avg_years_midsize
print(avg_years_midsize, n = 40)
```

```
## # A tibble: 36 x 6
##    make          `1994` `1999` `2004` `2009` `2014`
##    <chr>          <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
##  1 BMW             13.7   14.5   16.9   15.8   20.2
##  2 Buick           18.2   16.5   17.3   16.5   20.2
##  3 Cadillac        15     15.3   15.8   16.2   16.3
##  4 Chevrolet       16     17.2   18.9   20.7   25.6
##  5 Dodge           18.8   18.8   18.8   18.7   21.1
##  6 Ford            16.6   16.2   NA     18.8   30.4
##  7 Hyundai         17.5   18.2   18.8   24.5   27.4
##  8 Infiniti        14.5   17.7   16.5   17.7   19.3
##  9 Lexus           16     16.7   16.7   17.4   21.1
## 10 Lincoln         16     NA     16.2   17.5   25.2
## 11 Mazda           19.2   19.8   18.8   19     27.6
## 12 Mercury         17     16.5   17.3   18.8   NA
## 13 Nissan          17.3   18.5   18.4   24.4   38.9
## 14 Oldsmobile      17.8   17     NA     NA     NA
## 15 Plymouth        18.8   20     NA     NA     NA
## 16 Pontiac         16     16.7   16.7   NA     NA
## 17 Rolls-Royce      9     10.5   11     11     13
## 18 Saab            17     17.3   17.5   17.5   NA
## 19 Toyota          18     19     25     28.4   32.4
## 20 Volvo           17     17.3   17.8   15.3   19
## 21 Acura           NA     16.5   17.3   17     20.6
## 22 Audi            NA     15.2   16.2   15.8   19.1
## 23 Bentley         NA     10.5    9     10     11.5
## 24 Chrysler        NA     17     19.3   18.7   20
## 25 Daewoo          NA     18     17     NA     NA
## 26 Honda           NA     20.2   20     NA     29.2
## 27 Jaguar          NA     14.7   15.8   15.5   16.4
## 28 Mercedes-Benz   NA     18.2   15     16.2   20
## 29 Mitsubishi      NA     18     16.8   18     NA
## 30 Volkswagen      NA     17     19     19     24.7
## 31 Kia             NA     NA     19.7   22.2   26.4
## 32 Saturn          NA     NA     19.5   22.6   NA
## 33 Suzuki          NA     NA     17     NA     NA
## 34 Ferrari         NA     NA     NA      9.5   11
## 35 Maserati        NA     NA     NA     NA     15
## 36 Subaru          NA     NA     NA     NA     21
```

Part 2. More dplyr

Make sure your final output shows the desired average number of days between visits.

```r
load("dr4.Rdata")
# head(dr4)

# Get IDs of people who visit more than once
visits_more_than_once <- dr4 %>%
  mutate(total_visits = rowSums(!is.na(select(dr4, starts_with("visit"))), na.rm = TRUE)) %>%
  rowwise() %>%
  filter(total_visits > 1)
# visits_more_than_once # 130 x 7 --> 130 visited more than once

# Pivot data
visits_long <- visits_more_than_once %>%
  pivot_longer(cols = starts_with("visit"),
               names_to = "visit_num",
               values_to = "date",
               values_drop_na = TRUE
  ) %>%
  arrange(id, date)
# visits_long

# Get time diff between dates
visits_diff <- visits_long %>%
  group_by(id) %>%
  mutate(time_diff = as.numeric(difftime(date, lag(date), units = "days"))) %>%
  ungroup() %>%
  filter(!is.na(time_diff))
# visits_diff


# Calculations
total_days <- sum(visits_diff$time_diff, na.rm = TRUE)
total_intervals <- nrow(visits_diff)
total_average = total_days / total_intervals
total_average
```

```
## [1] 22.4902
```

Part 3. Scrape baseball-reference.com with rvest

```r
library(rvest)
library(polite)

# Open a polite session.
session <- bow("http://www.baseball-reference.com/teams/")

# Scrape the content of the page and store it in an object teampage.
# There's no need to open another session.
teampage <- session %>%
  scrape(content = "text/html; charset=UTF-8")

# Now that the page content has been scraped, you do not need to request it
# again. Use the object teampage and html_nodes() to extract the desired nodes,
# for example, you'll want to extract the team names among other values.
```

```
teamnames2 <- teampage %>% html_nodes("#teams_active .left a")
# teamnames2

# Write a loop to visit each of the active franchise team pages.
# To change what page you are visiting, use nod("url of updated location")

# Initializers
team_urls <- teampage %>%
  html_nodes("#teams_active .left a") %>%
  html_attr("href")
# team_urls
teamnames <- teampage %>%
  html_nodes("#teams_active .left a") %>%
  html_text()
# teamnames
names(team_urls) <- team_names
```

```
## Error in eval(expr, envir, enclos): object 'team_names' not found
```

```
# print(team_urls)
team_urls["Los Angeles Dodgers"]
```

```
## [1] NA
```

```
team_data <- list()
# seq_along(team_urls)
for(team_name in seq_along(team_urls)) {
  team_page <- session %>%
    nod(path = team_urls[team_name]) %>%
    scrape(content = "text/html; charset=UTF-8")

  franchise_history <- team_page %>%
    html_node("#franchise_years") %>%
    html_table(fill = TRUE) %>%
    mutate(
      GB = suppressWarnings(as.numeric(as.character(GB))),  # Convert GB to numeric, handling NA values
      current_name = teamnames[team_name]  # Add current team name
    )
  team_data[[teamnames[team_name]]] <- franchise_history
}
# Combine all the data into a single table called baseball that contains all
# of the teams' franchise histories
baseball <- bind_rows(team_data)
# at the end, be sure to print out the dimensions of your baseball table
dim(baseball) # Correct 2804 x 22 dim
```

```
## [1] 2804   22
```

```
# also print the first few rows of the table
print(baseball, n = 10)
```

```
## # A tibble: 2,804 x 22
##     Year Tm         Lg        G     W     L  Ties `W-L%` `pythW-L%` Finish    GB
##    <int> <chr>      <chr> <int> <int> <int> <int>  <dbl>      <dbl> <chr>  <dbl>
##  1  2023 Arizona D~ NL W~   162    84    78     0  0.519      0.491 2nd o~  16
##  2  2022 Arizona D~ NL W~   162    74    88     0  0.457      0.476 4th o~  37
```

```
##  3  2021 Arizona D~ NL W~   162    52   110     0  0.321      0.377 5th o~  55
##  4  2020 Arizona D~ NL W~    60    25    35     0  0.417      0.458 5th o~  18
##  5  2019 Arizona D~ NL W~   162    85    77     0  0.525      0.541 2nd o~  21
##  6  2018 Arizona D~ NL W~   162    82    80     0  0.506      0.533 3rd o~   9.5
##  7  2017 Arizona D~ NL W~   162    93    69     0  0.574      0.594 2nd o~  11
##  8  2016 Arizona D~ NL W~   162    69    93     0  0.426      0.424 4th o~  22
##  9  2015 Arizona D~ NL W~   162    79    83     0  0.488      0.504 3rd o~  13
## 10  2014 Arizona D~ NL W~   162    64    98     0  0.395      0.415 5th o~  30
## # i 2,794 more rows
## # i 11 more variables: Playoffs <chr>, R <int>, RA <int>, Attendance <chr>,
## #   BatAge <dbl>, PAge <dbl>, `#Bat` <int>, `#P` <int>, `Top Player` <chr>,
## #   Managers <chr>, current_name <chr>
```

Some light text clean up

```
## [1] "Lengths (21, 20) differ (comparison on first 20 components)"
## [2] "13 element mismatches"
```

```
## [1] TRUE
```

Part 4. dplyr to summarize the baseball data

```r
# Enter your r code here
# Your final line of code here should print the summary table in the report
# Be sure to print all 30 rows
# All requested columns must appear in the output to receive full credit.
baseball_summary <- baseball %>%
  mutate(year = as.numeric(Year)) %>%
  filter(year >= 2001 & year <= 2023) %>%
  group_by(current_name) %>%
  summarise(
    TW = sum(W, na.rm = TRUE),
    TL = sum(L, na.rm = TRUE),
    TR = sum(R, na.rm = TRUE),
    TRA = sum(RA, na.rm = TRUE),
    TWP = TW / (TW + TL)
  ) %>%
  arrange(desc(TWP))
# head(baseball_summary) check NY Yankees: 2105W, 1516L, 0.5813TWP
baseball_summary
```

```
## # A tibble: 30 x 6
##    current_name            TW    TL    TR   TRA   TWP
##    <chr>                <int> <int> <int> <int> <dbl>
##  1 New York Yankees      2105  1516 18378 15772 0.581
##  2 Los Angeles Dodgers   2055  1569 16625 14241 0.567
##  3 St. Louis Cardinals   2001  1620 16964 15219 0.553
##  4 Boston Red Sox        1979  1644 18585 16615 0.546
##  5 Atlanta Braves        1968  1652 16845 15255 0.544
##  6 Los Angeles Angels    1889  1735 16689 16139 0.521
##  7 San Francisco Giants  1875  1746 15665 15325 0.518
##  8 Oakland Athletics     1870  1753 16365 15677 0.516
##  9 Houston Astros        1869  1754 16530 15725 0.516
## 10 Cleveland Guardians   1864  1758 16849 16171 0.515
## # i 20 more rows
```

5. Regular expressions to extract values in the Managers Column

```r
# enter your r code here
# your final line of code here should print the first 15 rows of
# the summary table in the report
# All requested columns must appear in the output to receive full credit.
# regex to get: first initial, last name w/ possible whitespace, win-loss record
regex <- "([A-Z])\\.([a-zA-Z]+(?:\\s[a-zA-Z]+)?)\\s?\\((\\d+)-(\\d+)\\)"
# Extract manager data
manager_data <- baseball %>%
  filter(!is.na(Managers)) %>%
  mutate(manager_records = str_match_all(Managers, regex)) %>%
  select(current_name, manager_records)
expand_manager_data <- manager_data %>%
  unnest(manager_records) %>%
  transmute(
    Manager = paste0(manager_records[, 2], ".", manager_records[, 3]),
    Wins = as.numeric(manager_records[, 4]),
    Losses = as.numeric(manager_records[, 5])
  )
# Create manager summary
manager_summary <- expand_manager_data %>%
  group_by(Manager) %>%
  summarise(
    TG = sum(Wins + Losses, na.rm = TRUE),
    TW = sum(Wins, na.rm = TRUE),
    TL = sum(Losses, na.rm = TRUE),
    TWP = TW / TG
  ) %>%
  arrange(desc(TG))
print(manager_summary, n = 15)
```

```
## # A tibble: 591 x 5
##    Manager       TG    TW    TL   TWP
##    <chr>      <dbl> <dbl> <dbl> <dbl>
##  1 C.Mack      7679  3731  3948 0.486
##  2 T.La Russa  5383  2884  2499 0.536
##  3 D.Baker     4824  2602  2222 0.539
##  4 B.Cox       4505  2504  2001 0.556
##  5 B.Harris    4377  2158  2219 0.493
##  6 J.McGraw    4373  2583  1790 0.591
##  7 J.Torre     4323  2326  1997 0.538
##  8 B.Bochy     4194  2093  2101 0.499
##  9 S.Anderson  4028  2194  1834 0.545
## 10 G.Mauch     3939  1902  2037 0.483
## 11 C.Stengel   3747  1905  1842 0.508
## 12 L.Durocher  3717  2008  1709 0.540
## 13 W.Alston    3653  2040  1613 0.558
## 14 T.Francona  3622  1950  1672 0.538
## 15 L.Piniella  3548  1835  1713 0.517
## # i 576 more rows
```