ST558_HW3_Importing_Data

Charles Lane

Task 1 - Conceptual Questions

1 - *If your working directory is myfolder/homework/, what relative path would you specify to get the file located at myfolder/MyData.csv?*

From the R Console, we use the getwd() and setwd() commands to get and set the working directory, respectively. Setting a working directory of a different folder/object with the same root path as the current directory is accomplished with:

'setwd("../MyData.csv")'

From the Terminal, commands of simply 'cd ..' will change the directory to a level higher. 'cd ./DIRECTORY' would move the active directory to a location within the current directory.

2 - *What are the major benefits of using R Projects?*

R projects establish a self-contained organizational location. This helps sharing configuration between different parties as all folder or file references can be local.

3 - *What is git and what is github?*

Git is a version control and history software which retains all versions of code and allows for review/comparison of updates to code. Git can therefore be used to collaborate as well. Several different entities may develop on a main branch of code while enabling review by all parties prior to commmitting updates. Github is a website or front-end environment for git, including visualization of code projects ("Repos"), and their associated components (branches, descriptive files, etc.)

4 - *What are the two main differences between a tibble and a data.frame?*

A tibble's print results are more organized and display only the top rows/columns. Also, as stated in the notes - they do not coerce down to a vector when you subset to only one column using [

5 - *Rewrite the following nested function call using baseR's chaining operator:*

6 - *What is meant by long format data and wide format data? Which do we generally prefer for statistical analysis?*

Wide format data is generally used more often for presentation and includes multiple observations in one row. Long format limits the data table to one row per observation. While wide format may be easier to consume visually, statistical analysis is more easily performed on long format because it requires less manipulation of data to separate observations.

Task 2 - Reading Delimited Data

Task 2.1 - Glass Data

First, need to activate the necessary packages for this session.

```
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 beco
```

```
library(readr)
```

Read-in data using the 'readr' package from tidyverse.

```
#reading in data
glass_data <- read_csv(
  "./glass.data",
  col_names = c("ID","RI", "Na", "Mg", "Al", "Si",
                "K", "Ca", "Ba", "Fe", "Type_of_glass"),
  show_col_types = TRUE)
```

```
Rows: 214 Columns: 11
-- Column specification --------------------------------------------------------
Delimiter: ","
dbl (11): ID, RI, Na, Mg, Al, Si, K, Ca, Ba, Fe, Type_of_glass

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.
```

```
glass_data
```

```
# A tibble: 214 x 11
      ID    RI    Na    Mg    Al    Si     K    Ca    Ba    Fe Type_of_glass
   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>         <dbl>
 1     1  1.52  13.6  4.49  1.1   71.8  0.06  8.75     0  0                 1
 2     2  1.52  13.9  3.6   1.36  72.7  0.48  7.83     0  0                 1
 3     3  1.52  13.5  3.55  1.54  73.0  0.39  7.78     0  0                 1
 4     4  1.52  13.2  3.69  1.29  72.6  0.57  8.22     0  0                 1
 5     5  1.52  13.3  3.62  1.24  73.1  0.55  8.07     0  0                 1
 6     6  1.52  12.8  3.61  1.62  73.0  0.64  8.07     0  0.26              1
 7     7  1.52  13.3  3.6   1.14  73.1  0.58  8.17     0  0                 1
 8     8  1.52  13.2  3.61  1.05  73.2  0.57  8.24     0  0                 1
 9     9  1.52  14.0  3.58  1.37  72.1  0.56  8.3      0  0                 1
10    10  1.52  13    3.6   1.36  73.0  0.57  8.4      0  0.11              1
# i 204 more rows
```

Now update "Type_of_glass" variable to be a string with descriptive values in place of values of 1-7.

```
glass_data |>
  mutate(
    Type_of_glass = ifelse(
      Type_of_glass == 1,"building_windows_float_processed",
        ifelse(
          Type_of_glass == 2, "building_windows_non_float_processed",
            ifelse(
              Type_of_glass == 3, "vehicle_windows_float_processed",
                ifelse(
                  Type_of_glass == 4,
                    "vehicle_windows_non_float_processed",
                    ifelse(
                      Type_of_glass == 5, "containers",
                        ifelse(
                          Type_of_glass == 6, "tableware",
                            ifelse(
                              Type_of_glass == 7, "headlamps", "ERROR")
                            )
                        )
                    )
                )
            )
        )
      )
```

```
# A tibble: 214 x 11
      ID    RI    Na    Mg    Al    Si     K    Ca    Ba    Fe Type_of_glass
   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <chr>
 1     1  1.52  13.6  4.49  1.1   71.8  0.06  8.75     0  0    building_windows~
 2     2  1.52  13.9  3.6   1.36  72.7  0.48  7.83     0  0    building_windows~
 3     3  1.52  13.5  3.55  1.54  73.0  0.39  7.78     0  0    building_windows~
 4     4  1.52  13.2  3.69  1.29  72.6  0.57  8.22     0  0    building_windows~
 5     5  1.52  13.3  3.62  1.24  73.1  0.55  8.07     0  0    building_windows~
 6     6  1.52  12.8  3.61  1.62  73.0  0.64  8.07     0  0.26 building_windows~
 7     7  1.52  13.3  3.6   1.14  73.1  0.58  8.17     0  0    building_windows~
 8     8  1.52  13.2  3.61  1.05  73.2  0.57  8.24     0  0    building_windows~
 9     9  1.52  14.0  3.58  1.37  72.1  0.56  8.3      0  0    building_windows~
10    10  1.52  13    3.6   1.36  73.0  0.57  8.4      0  0.11 building_windows~
# i 204 more rows
```

The preceding chain only updated the 'Type_of_glass' values, but the chain can be extended to include filtering and selection.

```
glass_data |>
  mutate(
    Type_of_glass = ifelse(
      Type_of_glass == 1,"building_windows_float_processed",
        ifelse(
          Type_of_glass == 2, "building_windows_non_float_processed",
            ifelse(
              Type_of_glass == 3, "vehicle_windows_float_processed",
                ifelse(
                  Type_of_glass == 4,
                    "vehicle_windows_non_float_processed",
                    ifelse(
                      Type_of_glass == 5, "containers",
                        ifelse(
                          Type_of_glass == 6, "tableware",
                            ifelse(
                              Type_of_glass == 7, "headlamps", "ERROR")
                            )
                        )
                    )
                )
            )
        )
      ) |>
  filter(
    Fe < 0.2,
    Type_of_glass == "tableware" | Type_of_glass == "headlamps")
```

```
# A tibble: 38 x 11
      ID    RI    Na    Mg    Al    Si     K    Ca    Ba    Fe Type_of_glass
   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <chr>
 1   177  1.52  14    2.39  1.56  72.4  0     9.57  0        0 tableware
 2   178  1.52  13.8  2.41  1.19  72.8  0     9.77  0        0 tableware
 3   179  1.52  14.5  2.24  1.62  72.4  0     9.26  0        0 tableware
 4   180  1.52  14.1  2.19  1.66  72.7  0     9.32  0        0 tableware
 5   181  1.51  14.4  1.74  1.54  74.6  0     7.59  0        0 tableware
 6   182  1.52  15.0  0.78  1.74  72.5  0     9.95  0        0 tableware
 7   183  1.52  14.2  0     2.09  72.7  0    10.9   0        0 tableware
 8   184  1.52  14.6  0     0.56  73.5  0    11.2   0        0 tableware
 9   185  1.51  17.4  0     0.34  75.4  0     6.65  0        0 tableware
10   186  1.51  13.7  3.2   1.81  72.8  1.76  5.43  1.19     0 headlamps
# i 28 more rows
```

Task 2.2 - Yeast Data

Pull in the yeast data

```
yeast_data <- read_fwf(
  "./yeast.data",
  col_positions = fwf_widths(c(12, 6, 6, 6, 6, 6, 6, 6, 6, 3),
                             c("seq_name","mcg", "gvh", "alm", "mit",
                               "erl", "pox", "vac", "nuc", "class")),
  show_col_types = TRUE)
```

```
Rows: 1484 Columns: 10
-- Column specification --------------------------------------------------------

chr (2): seq_name, class
dbl (8): mcg, gvh, alm, mit, erl, pox, vac, nuc

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.
```

```
yeast_data
```

```
# A tibble: 1,484 x 10
   seq_name     mcg   gvh   alm   mit   erl   pox   vac   nuc class
   <chr>      <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <chr>
 1 ADT1_YEAST  0.58  0.61  0.47  0.13   0.5   0    0.48  0.22 MIT
 2 ADT2_YEAST  0.43  0.67  0.48  0.27   0.5   0    0.53  0.22 MIT
 3 ADT3_YEAST  0.64  0.62  0.49  0.15   0.5   0    0.53  0.22 MIT
 4 AAR2_YEAST  0.58  0.44  0.57  0.13   0.5   0    0.54  0.22 NUC
 5 AATM_YEAST  0.42  0.44  0.48  0.54   0.5   0    0.48  0.22 MIT
 6 AATC_YEAST  0.51  0.4   0.56  0.17   0.5   0.5  0.49  0.22 CYT
 7 ABC1_YEAST  0.5   0.54  0.48  0.65   0.5   0    0.53  0.22 MIT
 8 BAF1_YEAST  0.48  0.45  0.59  0.2    0.5   0    0.58  0.34 NUC
 9 ABF2_YEAST  0.55  0.5   0.66  0.36   0.5   0    0.49  0.22 MIT
10 ABP1_YEAST  0.4   0.39  0.6   0.15   0.5   0    0.58  0.3  CYT
# i 1,474 more rows
```

Start a chain removing the 'seq_name' and 'nuc' columns.

```
yeast_data |>
  select(-seq_name & -nuc)
```

```
# A tibble: 1,484 x 8
     mcg   gvh   alm   mit   erl   pox   vac class
   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <chr>
 1  0.58  0.61  0.47  0.13   0.5   0    0.48 MIT
 2  0.43  0.67  0.48  0.27   0.5   0    0.53 MIT
 3  0.64  0.62  0.49  0.15   0.5   0    0.53 MIT
 4  0.58  0.44  0.57  0.13   0.5   0    0.54 NUC
 5  0.42  0.44  0.48  0.54   0.5   0    0.48 MIT
 6  0.51  0.4   0.56  0.17   0.5   0.5  0.49 CYT
 7  0.5   0.54  0.48  0.65   0.5   0    0.53 MIT
 8  0.48  0.45  0.59  0.2    0.5   0    0.58 NUC
 9  0.55  0.5   0.66  0.36   0.5   0    0.49 MIT
10  0.4   0.39  0.6   0.15   0.5   0    0.58 CYT
# i 1,474 more rows
```

Continue the chain to determine mean and median for all numeric variables, grouped by 'class'.

```
yeast_data |>
  select(-seq_name & -nuc) |>
  group_by(class) |>
  mutate(across(where(is.numeric), list(mean = mean, median = median), .names = "{.col}_{.fn
```

```
# A tibble: 1,484 x 22
# Groups:   class [10]
     mcg   gvh   alm   mit   erl   pox   vac class mcg_mean mcg_median gvh_mean
   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <chr>    <dbl>      <dbl>    <dbl>
 1  0.58  0.61  0.47  0.13   0.5   0    0.48 MIT      0.521       0.51    0.533
 2  0.43  0.67  0.48  0.27   0.5   0    0.53 MIT      0.521       0.51    0.533
 3  0.64  0.62  0.49  0.15   0.5   0    0.53 MIT      0.521       0.51    0.533
 4  0.58  0.44  0.57  0.13   0.5   0    0.54 NUC      0.452       0.45    0.456
 5  0.42  0.44  0.48  0.54   0.5   0    0.48 MIT      0.521       0.51    0.533
 6  0.51  0.4   0.56  0.17   0.5   0.5  0.49 CYT      0.481       0.48    0.470
 7  0.5   0.54  0.48  0.65   0.5   0    0.53 MIT      0.521       0.51    0.533
 8  0.48  0.45  0.59  0.2    0.5   0    0.58 NUC      0.452       0.45    0.456
 9  0.55  0.5   0.66  0.36   0.5   0    0.49 MIT      0.521       0.51    0.533
10  0.4   0.39  0.6   0.15   0.5   0    0.58 CYT      0.481       0.48    0.470
# i 1,474 more rows
```

```
# i 11 more variables: gvh_median <dbl>, alm_mean <dbl>, alm_median <dbl>,
#   mit_mean <dbl>, mit_median <dbl>, erl_mean <dbl>, erl_median <dbl>,
#   pox_mean <dbl>, pox_median <dbl>, vac_mean <dbl>, vac_median <dbl>
```

Task 2.3 - Combining Excel and Delimited Data

Read in data from the white wine excel spreadsheet. First activate the 'readxl' package for this session.

```
library("readxl")
```

Then, read in the excel sheet from the RProject session folder via the 'readxl' package.

```
white_wine <- read_excel(
  "./white-wine.xlsx",
  sheet = "white-wine",
  col_names = TRUE
)
white_wine
```

```
# A tibble: 4,898 x 12
   `fixed acidity` `volatile acidity` `citric acid` `residual sugar` chlorides
             <dbl>              <dbl>         <dbl>             <dbl>     <dbl>
 1               7               0.27          0.36              20.7     0.045
 2              63               0.3           0.34               1.6     0.049
 3              81               0.28          0.4                6.9     0.05
 4              72               0.23          0.32               8.5     0.058
 5              72               0.23          0.32               8.5     0.058
 6              81               0.28          0.4                6.9     0.05
 7              62               0.32          0.16               7       0.045
 8               7               0.27          0.36              20.7     0.045
 9              63               0.3           0.34               1.6     0.049
10              81               0.22          0.43               1.5     0.044
# i 4,888 more rows
# i 7 more variables: `free sulfur dioxide` <dbl>,
#   `total sulfur dioxide` <dbl>, density <dbl>, pH <dbl>, sulphates <dbl>,
#   alcohol <dbl>, quality <dbl>
```

Rename the column names (i.e. remove spaces) by using the variables names from the second excel sheet.

```
ww_col_names <- data.frame(
  read_excel(
    "./white-wine.xlsx",
    sheet = "variables",
    col_names = TRUE
  )
)
ww_col_names
```

```
             Variables
1        fixed_acidity
2     volatile_acidity
3          citric_acid
4       residual_sugar
5            chlorides
6   free_sulfur_dioxide
7  total_sulfur_dioxide
8              density
9                   pH
10           sulphates
11             alcohol
12             quality
```

Overwrite the existing column names with the 'ww_col_names' tibble from the excel sheet.

```
colnames(white_wine) <- ww_col_names[[1]]
white_wine
```

```
# A tibble: 4,898 x 12
   fixed_acidity volatile_acidity citric_acid residual_sugar chlorides
           <dbl>            <dbl>       <dbl>          <dbl>     <dbl>
 1             7             0.27        0.36           20.7     0.045
 2            63             0.3         0.34            1.6     0.049
 3            81             0.28        0.4             6.9     0.05
 4            72             0.23        0.32            8.5     0.058
 5            72             0.23        0.32            8.5     0.058
 6            81             0.28        0.4             6.9     0.05
 7            62             0.32        0.16            7       0.045
 8             7             0.27        0.36           20.7     0.045
 9            63             0.3         0.34            1.6     0.049
10            81             0.22        0.43            1.5     0.044
```

```
# i 4,888 more rows
# i 7 more variables: free_sulfur_dioxide <dbl>, total_sulfur_dioxide <dbl>,
#   density <dbl>, pH <dbl>, sulphates <dbl>, alcohol <dbl>, quality <dbl>
```

Add a column indicating that these observations are associated with white wines.

```
mut_white_wine <- white_wine |>
  mutate(wine_type = "white")
print(mut_white_wine)
```

```
# A tibble: 4,898 x 13
   fixed_acidity volatile_acidity citric_acid residual_sugar chlorides
           <dbl>            <dbl>       <dbl>          <dbl>     <dbl>
 1             7             0.27        0.36           20.7     0.045
 2            63             0.3         0.34            1.6     0.049
 3            81             0.28        0.4             6.9     0.05
 4            72             0.23        0.32            8.5     0.058
 5            72             0.23        0.32            8.5     0.058
 6            81             0.28        0.4             6.9     0.05
 7            62             0.32        0.16            7       0.045
 8             7             0.27        0.36           20.7     0.045
 9            63             0.3         0.34            1.6     0.049
10            81             0.22        0.43            1.5     0.044
# i 4,888 more rows
# i 8 more variables: free_sulfur_dioxide <dbl>, total_sulfur_dioxide <dbl>,
#   density <dbl>, pH <dbl>, sulphates <dbl>, alcohol <dbl>, quality <dbl>,
#   wine_type <chr>
```

Pull in the red wine data by using readr package. First enable the package.

```
library(readr)
```

Then pull in the data from a .csv file.

```
red_wine <- read_delim(
  "./red-wine.csv",
  delim = ";",
  show_col_types = TRUE
)
```

```
Rows: 1599 Columns: 12
-- Column specification --------------------------------------------------------
Delimiter: ";"
dbl (12): fixed acidity, volatile acidity, citric acid, residual sugar, chlo...

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.
```

```
red_wine
```

```
# A tibble: 1,599 x 12
   `fixed acidity` `volatile acidity` `citric acid` `residual sugar` chlorides
             <dbl>              <dbl>         <dbl>            <dbl>     <dbl>
 1             7.4               0.7           0                1.9     0.076
 2             7.8               0.88          0                2.6     0.098
 3             7.8               0.76          0.04             2.3     0.092
 4            11.2               0.28          0.56             1.9     0.075
 5             7.4               0.7           0                1.9     0.076
 6             7.4               0.66          0                1.8     0.075
 7             7.9               0.6           0.06             1.6     0.069
 8             7.3               0.65          0                1.2     0.065
 9             7.8               0.58          0.02             2       0.073
10             7.5               0.5           0.36             6.1     0.071
# i 1,589 more rows
# i 7 more variables: `free sulfur dioxide` <dbl>,
#   `total sulfur dioxide` <dbl>, density <dbl>, pH <dbl>, sulphates <dbl>,
#   alcohol <dbl>, quality <dbl>
```

Update the column names to be the same as those for the white wines.

```
colnames(red_wine) <- ww_col_names[[1]]
red_wine
```

```
# A tibble: 1,599 x 12
   fixed_acidity volatile_acidity citric_acid residual_sugar chlorides
           <dbl>            <dbl>       <dbl>          <dbl>     <dbl>
 1           7.4              0.7         0              1.9     0.076
 2           7.8              0.88        0              2.6     0.098
 3           7.8              0.76        0.04           2.3     0.092
 4          11.2              0.28        0.56           1.9     0.075
 5           7.4              0.7         0              1.9     0.076
```

```
 6           7.4             0.66        0               1.8     0.075
 7           7.9             0.6         0.06            1.6     0.069
 8           7.3             0.65        0               1.2     0.065
 9           7.8             0.58        0.02            2       0.073
10           7.5             0.5         0.36            6.1     0.071
# i 1,589 more rows
# i 7 more variables: free_sulfur_dioxide <dbl>, total_sulfur_dioxide <dbl>,
#   density <dbl>, pH <dbl>, sulphates <dbl>, alcohol <dbl>, quality <dbl>
```

Add a column indicating that these observations are associated with white wines.

```
mut_red_wine <- red_wine |>
  mutate(wine_type = "red")
mut_red_wine
```

```
# A tibble: 1,599 x 13
   fixed_acidity volatile_acidity citric_acid residual_sugar chlorides
           <dbl>            <dbl>       <dbl>          <dbl>     <dbl>
 1           7.4              0.7         0               1.9     0.076
 2           7.8              0.88        0               2.6     0.098
 3           7.8              0.76        0.04            2.3     0.092
 4          11.2              0.28        0.56            1.9     0.075
 5           7.4              0.7         0               1.9     0.076
 6           7.4              0.66        0               1.8     0.075
 7           7.9              0.6         0.06            1.6     0.069
 8           7.3              0.65        0               1.2     0.065
 9           7.8              0.58        0.02            2       0.073
10           7.5              0.5         0.36            6.1     0.071
# i 1,589 more rows
# i 8 more variables: free_sulfur_dioxide <dbl>, total_sulfur_dioxide <dbl>,
#   density <dbl>, pH <dbl>, sulphates <dbl>, alcohol <dbl>, quality <dbl>,
#   wine_type <chr>
```

Now combine the two datasets

```
wine_dataset <- bind_rows(mut_white_wine, mut_red_wine)
wine_dataset
```

```
# A tibble: 6,497 x 13
   fixed_acidity volatile_acidity citric_acid residual_sugar chlorides
           <dbl>            <dbl>       <dbl>          <dbl>     <dbl>
```

```
 1             7             0.27        0.36           20.7     0.045
 2            63             0.3         0.34            1.6     0.049
 3            81             0.28        0.4             6.9     0.05
 4            72             0.23        0.32            8.5     0.058
 5            72             0.23        0.32            8.5     0.058
 6            81             0.28        0.4             6.9     0.05
 7            62             0.32        0.16            7       0.045
 8             7             0.27        0.36           20.7     0.045
 9            63             0.3         0.34            1.6     0.049
10            81             0.22        0.43            1.5     0.044
# i 6,487 more rows
# i 8 more variables: free_sulfur_dioxide <dbl>, total_sulfur_dioxide <dbl>,
#   density <dbl>, pH <dbl>, sulphates <dbl>, alcohol <dbl>, quality <dbl>,
#   wine_type <chr>
```

Filter combined dataset to only see high quality (> 6.5) wines and wines that have an alcohol value <132.

```
wine_dataset |>
  filter(quality > 6.5, alcohol < 132)
```

```
# A tibble: 1,206 x 13
   fixed_acidity volatile_acidity citric_acid residual_sugar chlorides
           <dbl>            <dbl>       <dbl>          <dbl>     <dbl>
 1            66             0.16        0.4             1.5     0.044
 2            66             0.17        0.38            1.5     0.032
 3            62             0.66        0.48            1.2     0.029
 4            62             0.66        0.48            1.2     0.029
 5            64             0.31        0.38            2.9     0.038
 6            68             0.26        0.42            1.7     0.049
 7            72             0.32        0.36            2       0.033
 8            74             0.18        0.31            1.4     0.058
 9            66             0.25        0.29            1.1     0.068
10            62             0.16        0.33            1.1     0.057
# i 1,196 more rows
# i 8 more variables: free_sulfur_dioxide <dbl>, total_sulfur_dioxide <dbl>,
#   density <dbl>, pH <dbl>, sulphates <dbl>, alcohol <dbl>, quality <dbl>,
#   wine_type <chr>
```

Continue to sort the rows based on quality.

```
wine_dataset |>
  filter(quality > 6.5, alcohol < 132) |>
  arrange(quality)
```

```
# A tibble: 1,206 x 13
   fixed_acidity volatile_acidity citric_acid residual_sugar chlorides
           <dbl>            <dbl>       <dbl>          <dbl>     <dbl>
 1            66             0.16        0.4             1.5     0.044
 2            66             0.17        0.38            1.5     0.032
 3            64             0.31        0.38            2.9     0.038
 4            72             0.32        0.36            2       0.033
 5            74             0.18        0.31            1.4     0.058
 6            66             0.25        0.29            1.1     0.068
 7            62             0.16        0.33            1.1     0.057
 8            64             0.26        0.24            6.4     0.04
 9            71             0.18        0.36            1.4     0.043
10             7             0.32        0.34            1.3     0.042
# i 1,196 more rows
# i 8 more variables: free_sulfur_dioxide <dbl>, total_sulfur_dioxide <dbl>,
#   density <dbl>, pH <dbl>, sulphates <dbl>, alcohol <dbl>, quality <dbl>,
#   wine_type <chr>
```

Continue the chain to only show variables acid, alcohol, type, and variable.

```
wine_dataset |>
  filter(quality > 6.5, alcohol < 132) |>
  arrange(quality) |>
  select(citric_acid, alcohol, wine_type, quality)
```

```
# A tibble: 1,206 x 4
   citric_acid alcohol wine_type quality
         <dbl>   <dbl> <chr>       <dbl>
 1         0.4     124 white           7
 2         0.38    114 white           7
 3         0.38     11 white           7
 4         0.36    123 white           7
 5         0.31     10 white           7
 6         0.29     11 white           7
 7         0.33    109 white           7
 8         0.24    126 white           7
 9         0.36    127 white           7
```

```
10        0.34      12 white           7
# i 1,196 more rows
```

Now add the mean and standard deviation of the alcohol variable, grouped by quality.

```
wine_dataset |>
  group_by(quality) |>
  mutate(mean_alcohol = mean(alcohol)) |>
  mutate(sd_alcohol = sd(alcohol))
```

```
# A tibble: 6,497 x 15
# Groups:   quality [7]
   fixed_acidity volatile_acidity citric_acid residual_sugar chlorides
           <dbl>            <dbl>       <dbl>          <dbl>     <dbl>
 1             7             0.27        0.36           20.7     0.045
 2            63             0.3         0.34            1.6     0.049
 3            81             0.28        0.4             6.9     0.05
 4            72             0.23        0.32            8.5     0.058
 5            72             0.23        0.32            8.5     0.058
 6            81             0.28        0.4             6.9     0.05
 7            62             0.32        0.16            7       0.045
 8             7             0.27        0.36           20.7     0.045
 9            63             0.3         0.34            1.6     0.049
10            81             0.22        0.43            1.5     0.044
# i 6,487 more rows
# i 10 more variables: free_sulfur_dioxide <dbl>, total_sulfur_dioxide <dbl>,
#   density <dbl>, pH <dbl>, sulphates <dbl>, alcohol <dbl>, quality <dbl>,
#   wine_type <chr>, mean_alcohol <dbl>, sd_alcohol <dbl>
```

Task 3 - Database Practice

First, invoke the RSQLite package for this session.

```
library(RSQLite)
```

Then, connect to the lahman database.

```
con <- dbConnect(RSQLite::SQLite(), "./lahman.db")
```

Look at the tables in the database.

```
dbListTables(con)
```

```
 [1] "AllstarFull"         "Appearances"         "AwardsManagers"
 [4] "AwardsPlayers"       "AwardsShareManagers" "AwardsSharePlayers"
 [7] "Batting"             "BattingPost"         "CollegePlaying"
[10] "Fielding"            "FieldingOF"          "FieldingOFsplit"
[13] "FieldingPost"        "HallOfFame"          "HomeGames"
[16] "LahmanData"          "Managers"            "ManagersHalf"
[19] "Parks"               "People"              "Pitching"
[22] "PitchingPost"        "Salaries"            "Schools"
[25] "SeriesPost"          "Teams"               "TeamsFranchises"
[28] "TeamsHalf"           "battingLabels"       "fieldingLabels"
[31] "pitchingLabels"
```

Now let's get all data from the 'Teams' table in the year 2015.

```
library(dplyr)
tbl(con, "Teams") |>
  filter(yearID == 2015)
```

```
# Source:   SQL [?? x 48]
# Database: sqlite 3.45.2 [/Users/charleslane/hello/ST558_HW3/lahman.db]
   yearID lgID  teamID franchID divID  Rank     G Ghome     W     L DivWin WCWin
    <int> <chr> <chr>  <chr>    <chr> <int> <int> <int> <int> <int> <chr>  <chr>
 1   2015 NL    ARI    ARI      W         3   162    81    79    83 N      N
 2   2015 NL    ATL    ATL      E         4   162    81    67    95 N      N
 3   2015 AL    BAL    BAL      E         3   162    78    81    81 N      N
 4   2015 AL    BOS    BOS      E         5   162    81    78    84 N      N
 5   2015 AL    CHA    CHW      C         4   162    81    76    86 N      N
 6   2015 NL    CHN    CHC      C         3   162    81    97    65 N      Y
 7   2015 NL    CIN    CIN      C         5   162    81    64    98 N      N
 8   2015 AL    CLE    CLE      C         3   161    80    81    80 N      N
 9   2015 NL    COL    COL      W         5   162    81    68    94 N      N
10   2015 AL    DET    DET      C         5   161    81    74    87 N      N
# i more rows
# i 36 more variables: LgWin <chr>, WSWin <chr>, R <int>, AB <int>, H <int>,
#   X2B <int>, X3B <int>, HR <int>, BB <int>, SO <int>, SB <int>, CS <int>,
#   HBP <int>, SF <int>, RA <int>, ER <int>, ERA <dbl>, CG <int>, SHO <int>,
#   SV <int>, IPouts <int>, HA <int>, HRA <int>, BBA <int>, SOA <int>, E <int>,
#   DP <int>, FP <dbl>, name <chr>, park <chr>, attendance <int>, BPF <int>,
#   PPF <int>, teamIDBR <chr>, teamIDlahman45 <chr>, teamIDretro <chr>
```

Perform the same activity using SQL statements w/in the tbl function.

```
tbl(con, sql(
  "SELECT `Teams`.*
  FROM `Teams`
  WHERE (`yearID` = 2015.0)"
))
```

```
# Source:   SQL [?? x 48]
# Database: sqlite 3.45.2 [/Users/charleslane/hello/ST558_HW3/lahman.db]
   yearID lgID  teamID franchID divID  Rank     G Ghome     W     L DivWin WCWin
    <int> <chr> <chr>  <chr>    <chr> <int> <int> <int> <int> <int> <chr>  <chr>
 1   2015 NL    ARI    ARI      W         3   162    81    79    83 N      N
 2   2015 NL    ATL    ATL      E         4   162    81    67    95 N      N
 3   2015 AL    BAL    BAL      E         3   162    78    81    81 N      N
 4   2015 AL    BOS    BOS      E         5   162    81    78    84 N      N
 5   2015 AL    CHA    CHW      C         4   162    81    76    86 N      N
 6   2015 NL    CHN    CHC      C         3   162    81    97    65 N      Y
 7   2015 NL    CIN    CIN      C         5   162    81    64    98 N      N
 8   2015 AL    CLE    CLE      C         3   161    80    81    80 N      N
 9   2015 NL    COL    COL      W         5   162    81    68    94 N      N
10   2015 AL    DET    DET      C         5   161    81    74    87 N      N
# i more rows
# i 36 more variables: LgWin <chr>, WSWin <chr>, R <int>, AB <int>, H <int>,
#   X2B <int>, X3B <int>, HR <int>, BB <int>, SO <int>, SB <int>, CS <int>,
#   HBP <int>, SF <int>, RA <int>, ER <int>, ERA <dbl>, CG <int>, SHO <int>,
#   SV <int>, IPouts <int>, HA <int>, HRA <int>, BBA <int>, SOA <int>, E <int>,
#   DP <int>, FP <dbl>, name <chr>, park <chr>, attendance <int>, BPF <int>,
#   PPF <int>, teamIDBR <chr>, teamIDlahman45 <chr>, teamIDretro <chr>
```

Task 3.1 - Hall of Fame Data Organization

```
HOF_Players <- tbl(con, "HallOfFame") |>
  filter(inducted == "Y", category == "Player") |>
  select(playerID, yearID, category) |>
  collect()
HOF_Players
```

```
# A tibble: 270 x 3
   playerID  yearID category
```

```
   <chr>      <int> <chr>
 1 cobbty01    1936 Player
 2 ruthba01    1936 Player
 3 wagneho01   1936 Player
 4 mathech01   1936 Player
 5 johnswa01   1936 Player
 6 lajoina01   1937 Player
 7 speaktr01   1937 Player
 8 youngcy01   1937 Player
 9 alexape01   1938 Player
10 sislege01   1939 Player
# i 260 more rows
```

Combine the table above with first and last names from the People table.

```
inner_join(tbl(con, "HallOfFame") |>
             filter(inducted == "Y", category == "Player") |>
             select(playerID, yearID, category),
           tbl(con, "People") |>
             select(playerID, nameFirst, nameLast),
           by = join_by(playerID == playerID)) |>
  collect()
```

```
# A tibble: 270 x 5
   playerID  yearID category nameFirst nameLast
   <chr>      <int> <chr>    <chr>     <chr>
 1 cobbty01    1936 Player   Ty        Cobb
 2 ruthba01    1936 Player   Babe      Ruth
 3 wagneho01   1936 Player   Honus     Wagner
 4 mathech01   1936 Player   Christy   Mathewson
 5 johnswa01   1936 Player   Walter    Johnson
 6 lajoina01   1937 Player   Nap       Lajoie
 7 speaktr01   1937 Player   Tris      Speaker
 8 youngcy01   1937 Player   Cy        Young
 9 alexape01   1938 Player   Pete      Alexander
10 sislege01   1939 Player   George    Sisler
# i 260 more rows
```

Task 3.2 - Hall of Fame Managers' Win Percentage

Generate a dataset of the Hall of Fame managers and their win/loss records.

```r
tbl(con, "Managers") |>
  select(playerID, G, W, L) |>
  group_by(playerID) |>
  summarize(G_managed = sum(G, na.rm = TRUE),
            Total_W = sum(W, na.rm = TRUE),
            Total_L = sum(L, na.rm = TRUE)) |>
  collect() |>
  mutate(Win_Loss_Pc = Total_W/G_managed) |>
  arrange(desc(Win_Loss_Pc)) |>
  collect()
```

```
# A tibble: 749 x 5
   playerID  G_managed Total_W Total_L Win_Loss_Pc
   <chr>         <int>   <int>   <int>       <dbl>
 1 bensove01         1       1       0           1
 2 burwebi01         1       1       0           1
 3 cohenan01         1       1       0           1
 4 ebeldi99          3       3       0           1
 5 falkbi01          1       1       0           1
 6 hardeme01         3       3       0           1
 7 simmote01         1       1       0           1
 8 steinte01         2       2       0           1
 9 sukefcl01         2       2       0           1
10 tamarjo01         1       1       0           1
# i 739 more rows
```

Task 3.3 Determine Managers in the Hall of Fame

Determine people who were managers and are in the hall of fame, regardless of whether they were inducted as managers or not.

Approach: 1) Use the Hall of Fame table with similar transformation as above (successfully inducted into HoF) with the exception to *not* filter on category of 'player'.

2) Inner Join the resulting HoF table with the Managers manipulated table from above (including win percentage for managers). Join on 'playerID'.

NOTE - Since the HoF table was generated with a join, and the dplyr 'inner_join()' function only joins 2 tables, a "nested" inner join or an inner join of 'Managers' and (inner join of Hall of Fame and People) will be used.

3) The result will be a table with 9 variables: playerID, yearID, category, nameFirst, nameLast, G_managed, Total_W, Total_L, Win_Loss_Pc.

```
inner_join(tbl(con, "Managers") |>
             select(playerID, G, W, L) |>
             group_by(playerID) |>
             summarize(G_managed = sum(G, na.rm = TRUE),
                       Total_W = sum(W, na.rm = TRUE),
                       Total_L = sum(L, na.rm = TRUE)) |>
             collect() |>
             mutate(Win_Loss_Pc = Total_W/G_managed) |>
             arrange(desc(Win_Loss_Pc)),
           inner_join(tbl(con, "HallOfFame") |>
                        filter(inducted == "Y") |>
                        select(playerID, yearID, category),
                      tbl(con, "People") |>
                        select(playerID, nameFirst, nameLast),
                      by = join_by(playerID == playerID)) |>
             collect(),
           by = join_by(playerID == playerID))
```

```
# A tibble: 97 x 9
   playerID  G_managed Total_W Total_L Win_Loss_Pc yearID category     nameFirst
   <chr>         <int>   <int>   <int>       <dbl>  <int> <chr>        <chr>
 1 simmote01         1       1       0       1       2020 Player       Ted
 2 wrighge01        85      59      25       0.694   1937 Pioneer/Exe~ George
 3 spaldal01       126      78      47       0.619   1939 Pioneer/Exe~ Al
 4 mccarjo99      3487    2125    1333       0.609   1957 Manager      Joe
 5 comisch01      1410     840     541       0.596   1939 Pioneer/Exe~ Charlie
 6 southbi01      1770    1044     704       0.590   2008 Manager      Billy
 7 seleefr99      2180    1284     862       0.589   1999 Manager      Frank
 8 chancfr01      1622     946     648       0.583   1946 Player       Frank
 9 weaveea99      2541    1480    1060       0.582   1996 Manager      Earl
10 lopezal01      2425    1410    1004       0.581   1977 Manager      Al
# i 87 more rows
# i 1 more variable: nameLast <chr>
```