Chapter 4: Data Wrangling

- Data wrangling (also known as data munging), refers to the preprocessing of data to get it from its raw initial form into a form that is ready for the analysis we want to do.
- The R package `dplyr`, which is also part of the `tidyverse`, has many useful data wrangling tools.
- Hadley Wickham defined five verbs for common data wrangling steps and created R functions for each of these:
- `select`, `filter`, `mutate`, `arrange`, `summarize`
- Each of these takes a data frame as an argument and returns a data frame as its output.
- These tools were inspired by the abilities of the relational database querying syntax *SQL*, which we will study later.
- Using `dplyr` is in many ways an alternative to using *SQL*.

The `select` function

- Recall that the columns of a data table correspond to the variables (sometimes called the features).
- The `select` function selects a subset of the columns of the data table.
- This is useful when we measure or observe many variables, but we only want to analyze a few of them.

The `filter` function

- Recall that the rows of a data table correspond to the observations (also known as cases), which are the individuals in the data set.
- The `filter` function selects a subset of the rows of the data table.
- This is useful when we have data on many individuals, but we only want to examine a subset of them.
- We generally specify a logical condition on one (or more) of the columns, and this limits the rows to be only the ones that satisfy this condition.

Common Logical Conditions to Filter On

- == checks for which elements of a column are equal to some value (could be numerical or character).
- Note the double equals-sign is the equality check in R.
- A single equals-sign is an assignment operator that assigns a name to an object, although for clarity, the assignment operator <- is preferred for this task.
- Various forms of inequality can be checked with <= , >= , < , >.
- `%in%` checks which elements of a column are in a specified list.

Combining Logical Conditions

- It is possible to combine more than one logical condition with "AND" or "OR" statements:
  1. `&`: Element-wise Logical AND operator; returns TRUE elementwise if both elements are TRUE
  2. `&&`: Logical AND operator; returns TRUE if both statements are TRUE
  3. `|`: Elementwise Logical OR operator; returns TRUE elementwise if at least one of the elements is TRUE
  4. `||`: Logical OR operator; returns TRUE if at least one of the statements is TRUE
  5. `!`: Logical NOT; returns FALSE if statement is TRUE
- So note that `!=` checks for which elements of a column are NOT equal to some value (could be numerical or character).

The pipe operator

- We can often subset both the rows and the columns, and so we can use both `select` and `filter` on the data table at once.
- Longer expressions can be rendered easier to read with the *pipe operator*, which is: %>%
- This defines a *pipeline* where the object before the pipe operator is the data frame to be worked on, and the code after the pipe operator gives the manipulation that we want to do on the data frame.
- We can have multiple pipe operators in a single statement, which tells R:
  1. First do the first operation
  2. then on the resulting object, do the next operation, etc.

The `mutate` and `rename` function

- The `mutate` function can redefine or transform a variable in an existing data table, or add a new column that is based on other columns.
- When we add or change a column, it's typically recommended to save the altered data table with a new name, which allows the original data table to be preserved under the old name rather than rewritten.
- Don't overdo this though: If you are making several changes in sequence, don't need to create a newly named data table with each change (this would clutter up the workspace).
- Best practices in R suggest not using periods in the names of functions, data frames, and variables that you create; better to use underscores (_) instead.
- The `rename` function is an easy way to change the name of an existing column.

The `arrange` function

- The `arrange` function sorts the rows of a data value in ascending or descending order of some column(s).
- The simple R function `sort` sorts elements of vectors, but not rows of data frames.
- The `arrange` function works similarly to the `order` function in base R.
- If there are ties in values for a variable being sorted on (common with a categorical variable), you can specify multiple sorting conditions to break the ties.
- For character columns, the arranging is done based on alphabetical order.
- The `desc` function tells R to sort in descending order on that variable.

The `summarize` function

- The `summarize` function will reduce a data table to a row of summary statistics (or several rows if the `group_by` function is used.
- The summary statistics in the output could be different statistics for different variables.
- A descriptive name for each column of the output table should be assigned.
- The `group_by` function specifies a categorical variable, and the summary statistics would then be reported for each level of the categorical variable in separate rows of the output.
- The code `N = n()` will include a sample size (or the per-group sample sizes if `group_by` is used) and is generally recommended to include.

Extended example on Baseball Data

- Section 4.2 has a extended example of data wrangling with a baseball data set.
- Let's work through that example as well as some extra code.