dplyr package notes:

* **filter**: return rows with matching conditions
  + *filter(flights, month == 1, day == 1)*
  + *filter(flights, month == 1 | day == 1)*
* **arrange**: arrange rows by variables. Make a copy of the original table, therefore is computationally extensive.
  + *arrange(flights, year, month, day)*
  + *arrange(flights, desc(arr\_delay))*
* **select**:
  + *select(flights, year, month, day)*
  + *select(flights, -(year:day))* # select all the columns except those between year and day
  + *select(iris,* ***starts\_with****("Petal"))*
  + *select(iris****, ends\_with****("Width"))*
  + *select(iris,* ***contains****("etal"))*
  + *select(iris,* ***matches****(".t."))* # regular expression
  + ***num\_range****("x", 1:5, width = 2)* # select x01 to x05
* **distinct**: select distinct/unique rows
  + *distinct(select(flights, tail\_num = tailnum))*
* **rename**:
  + *rename(flights, tail\_num = tailnum)*
* **mutate**: add new columns that are function of existing columns and preserves existing
  + *mutate(flights, speed = distance / air\_time \* 60)*
* **transmute**: add new columns and drop existing
  + *transmute(flights, speed = distance / air\_time \* 60)*
* **summarise**: summarize multiple values into a single value. When you apply summarise on grouped data with multiple grouped variables, each summary peels off one level of the grouping.
  + *summarise(flights, delay = mean(dep\_delay, na.rm = TRUE))*
* **group\_by()**
* **tally()**: counts observations by group
  + *planes %>% group\_by(type) %>% tally()* is same as *planes %>% group\_by(type) %>% summarise(n = n())*
* **sample\_n()** / **sample\_frac()**: randomly sample rows with fixed number or fixed fraction
  + *sample\_n(flights, 10)*
  + *sample\_frac(flights, 1, replace = TRUE)* # bootstrap sample
* aggregation function
  + **n()**: number of observations
  + **n\_distinct(x)**
  + **first(x)**, **last(x)**, **nth(x, n)**
* **%>%** pipe operator: x %>% f(y) turns into f(x, y), so you can chain multiple operations that read left-to-right, top-to-bottom
* Join operation:
  + **inner\_join(x, y, by)**
  + **left\_join(x, y, by)**
  + **semi\_join(x, y, by)**: include rows of x that match y
  + **anti\_join(x, y, by)**: include rows of x that don’t match y
* **do()** function: perform arbitrary computation, returning either a data frame or arbitrary objects which will be stored in a list. This is particularly useful when working with models: you can fit models per group with do() and then flexibly extract components with either another do() – one returned value, or summarise() – several returned values.
  + **.** is a noun representing the current goup
  + *models <- flights %>%*

*filter(hour >= 5, hour <= 20) %>%*

*group\_by(date) %>%*

*do(*

*mod = lm(dep\_delay ~ hour, data = .)*

*)*

*rsq <- function(x) summary(x)$r.squared*

*fit <- models %>% summarise(date = as.Date(date), rsq = rsq(mod))*

*coef\_df <- function(x) {*

*sc <- coef(summary(x))*

*colnames(sc) <- c("est", "se", "t", "P")*

*data.frame(coef = rownames(sc), sc)*

*}*

*models %>% do(coef\_df(.$mod))*