**Syntax - Helpful conventions for wrangling**

dplyr::tbl_df(iris)

Converts data to tbl class. tbl’s are easier to examine than data frames. R displays only the data that fits onscreen:

```
Source: local data frame [150 x 5]
          Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1         5.1       3.5          1.4          0.2         setosa
2         4.9       3.0          1.4          0.2         setosa
3         4.7       3.2          1.3          0.2         setosa
4         4.6       3.1          1.5          0.2         setosa
5         5.0       3.6          1.4          0.2         setosa
... Variables not shown: Petal.Width (dbl), Species (fctr)
```

dplyr::glimpse(iris)

Information dense summary of tbl data.

tidy::View(iris)

View data set in spreadsheet-like display (note capital V).

dplyr::%>%%

Passes object on left hand side as first argument (or . argument) of function on right hand side.

```
x %>% f(y) is the same as f(x, y)  
y %>% f(x, .., z) is the same as f(x, y, z)
```

"Piping" with %>% makes code more readable, e.g.

```
iris %>%  
  group_by(Species) %>%  
  summarise(avg = mean(Sepal.Width)) %>%  
  arrange(avg)
```

**Reshaping Data - Change the layout of a data set**

tidy::gather(cases, "year", "n", 2:4)

Gather columns into rows.

tidy::spread(pollution, size, amount)

Spread rows into columns.

tidy::separate(storms, date, c("y", "m", "d"))

Separate one column into several.

tidy::unite(data, col, ..., sep)

Unite several columns into one.

dplyr::data_frame(a = 1:3, b = 4:6)

Combine vectors into data frame (optimized).

dplyr::arrange(mtcars, mpg)

Order rows by values of a column (low to high).

dplyr::arrange(mtcars, desc(mpg))

Order rows by values of a column (high to low).

dplyr::rename(tb, y = year)

Rename the columns of a data frame.

**Data Wrangling with dplyr and tidyr**

**Cheat Sheet**

- **Subset Observations (Rows)**
  - dplyr::filter(iris, Sepal.Length > 7)
    - Extract rows that meet logical criteria.
  - dplyr::distinct(iris)
    - Remove duplicate rows.
  - dplyr::sample_frac(iris, 0.5, replace = TRUE)
    - Randomly select fraction of rows.
  - dplyr::sample_n(iris, 10, replace = TRUE)
    - Randomly select n rows.
  - dplyr::slice(iris, 10:15)
    - Select rows by position.
  - dplyr::top_n(storms, 2, date)
    - Select and order top n entries (by group if grouped data).

- **Subset Variables (Columns)**
  - dplyr::select(iris, Sepal.Width, Petal.Length, Species)
    - Select columns by name or helper function.

  **Helper functions for select**
  
  - ?select
  - select(iris, contains(""))
    - Select columns whose name contains a character string.
  - select(iris, ends_with("Length"))
    - Select columns whose name ends with a character string.
  - select(iris, everything())
    - Select every column.
  - select(iris, matches("t").")
    - Select columns whose name matches a regular expression.
  - select(iris, num_range("x", 1:5))
    - Select columns named x1, x2, x3, x4, x5.
  - select(iris, one_of(iris))
    - Select columns whose names are in a group of names.
  - select(iris, starts_with("Sepal"))
    - Select columns whose name starts with a character string.
  - select(iris, Sepal.Length:Petal.Width)
    - Select all columns between Sepal.Length and Petal.Width (inclusive).
  - select(iris, Species)
    - Select all columns except Species.
**Summarise Data**

- `dplyr::summarise(iris, avg = mean(Sepal.Length))`
  - Summarise data into single row of values.
- `dplyr::summarise_each(iris, funs(mean))`
  - Apply summary function to each column.
- `dplyr::count(iris, Species, wt = Sepal.Length)`
  - Count number of rows with each unique value of Species.
- `dplyr::transmute(iris, sepal = Sepal.Length + Sepal.Width)`
  - Compute new variables by group.

**Make New Variables**

- `dplyr::mutate(iris, sepal = Sepal.Length + Sepal.Width)`
  - Compute new variables by group.
- `dplyr::cumall(iris)`
  - Cumulative all.
- `dplyr::cumif(iris)`
  - Cumulative any.
- `dplyr::cumsum(iris)`
  - Cumulative sum.
- `dplyr::cumprod(iris)`
  - Cumulative product.

**Group Data**

- `dplyr::group_by(iris, Species)`
  - Group data into rows with the same value of Species.
- `dplyr::ungroup(iris)`
  - Remove grouping information from data frame.

**Combine Data Sets**

- `dplyr::bind_rows(y, z)`
  - Append z to y as new rows.
- `dplyr::bind_cols(y, z)`
  - Append z to y as new columns.

**Set Operations**

- `y %>% union(y, z)`
  - Rows that appear in both y and z.
- `y %>% intersect(y, z)`
  - Rows that appear in either or both y and z.
- `y %>% setdiff(y, z)`
  - Rows that appear in only one of y and z.

**Filtering Joins**

- `dplyr::left_join(a, b, by = "x1")` 
  - Join matching rows from b to a.
- `dplyr::right_join(a, b, by = "x1")` 
  - Join matching rows from a to b.
- `dplyr::inner_join(a, b, by = "x1")` 
  - Join data. Retain only rows in both sets.
- `dplyr::outer_join(a, b, by = "x1")` 
  - Join data. Retain all values, all rows.