

Tab 1

```
> my_vector_sequence[my_vector_sequence <= 10]
```

Data frame

Sets data into labels

```
Sex <- c("male", "female", "male")
```

```
Length <- c(3.2, 3.7, 3.4)
```

```
Weight <- c(2.9, 4.0, 3.1)
```

Tibble

Frames the data

names() - states the name of the data or the ability to rename

dim() - lets you see the dimensions

str() - lets you see the data

Control+return (runs a command)

To get a specific data from a data set

Names of data collumb[#.#] first number is row, second is column

To use this data for calculations

Do what you want (mean, range, etc) name of data set, \$ and data point)

```
mean(hbirds$Length)
```

Data frames are a spreadsheet organized by vectors

Used to import files

```
hot_springs <- read_csv("hsprings.csv")
```

If the data is in a folder within the folder, you must specify by putting the name

```
fish <- read_csv("data/Gaeta_etal_CLC_data.csv")
```

- Plus option makes the <- command

Changing the class of a column..

See what the current class is (class(hot_spring\$spring)

Have it change to the factor you want (hot_spring\$spring <-
as.factor(hot_springs\$spring))

Lab 4 (applying syntax using ggplot2)

##Resources

- [R for Data Science 2e](https://r4ds.hadley.nz/data-visualize.html)
- [ggplot2 cheatsheet](https://posit.co/resources/cheatsheets/)

A giant lost of commands for this

Libraries

```
``{r}  
library(tidyverse)  
library(palmerpenguins)  
``
```

Gg plots help make quick and fast plots and graphs

Not all plot types work with all data

- + `discrete` quantitative data that only contains integers
- + `continuous` quantitative data that can take any numerical value
- + `categorical` qualitative data that can take on a limited number of values

The syntax we will be using is

Plot we use is in layers

In short, `**plot= data + geom_ + aesthetics**`.

If we have both categorica (island, species, sex) with continuous (numbers)

If something is considered an integer, it'll be a considered a number (bad for things like year)

First we make a graph for what we want,

```
ggplot(data=penguins, #this specifies the data
```

Then we specify the x and y

```
mapping=aes(x=species, y=body_mass_g)) #map the aesthetics  
Combine both for the graph
```

Notice that we have a nice background, labeled axes, and even a value range of our variables on the y-axis- but no plot. This is because we need to tell ggplot how we want our data represented. This is called the geometry or `geom()`. There are many types of `geom`, see the ggplot [cheatsheet](https://posit.co/resources/cheatsheets/).

To turn it into a boxplot you use the initial code and then `+geom_boxplot()`

```
ggplot(data=penguins,  
mapping=aes(x=species, y=flipper_length_mm))+  
geom_boxplot()
```

Always start with names(dataset)

Coding for a scatter plot

```
ggplot(data=penguins,  
  mapping=aes(x=body_mass_g, y=flipper_length_mm))+  
  geom_point(na.rm=T)+  
  geom_smooth(method=lm,se=T)
```

Basically reading as I want to plot the data which is penguins by mapping it to look like aesthetics (aes) into the x point being species, y-int being flipper length. Plus I want this to be in a box plot format.

Now we make it pretty! ✨

Simply add color=species next to the intercepts for a already assigned color

method=lm (linear model)

```
ggplot(data=penguins,  
  mapping=aes(x=bill_length_mm, y=bill_depth_mm))+  
  geom_point(na.rm = T)+  
  geom_smooth(method=lm,se=F)+  
  labs(title="Bill Length (mm) vs Bill Depth (mm)")
```

Bar Plots

Automatically makes a count the y int when making the x

FOR EVERY PLOT IN CLASS AND EXAM.

MUST SHOW TITLE + X AXIS + AND Y AXIS AND AN ATTEMPT TO SHOW COLOR

Options + Control + I makes a new data chunk

lab 5 and 6

Shift command m for a pipe

Paste a line to your repository to make sure it is public and professionally presented. A clear readme. well organized without duplicate or other mislabeled files. contains all of your labs and homework

Data transformations

`distinct()` - row wise, used to find duplicate rows,

- See all unique categories in the rows. (finding species)
- Add more data to find unique matches (what island is each species in? Species, island)
- Adding **(, .keep_all=T)** keeps all variables that `distinct` didn't look at (still get all the data from the points you took out with `distinct`)

`mutate()` - looks at columns, makes a new one from them based on commands

- Added to the end of the data frame
- Ex, make a new column where grams turns into kilograms
**(penguins %>%
select(species, island, body_mass_g) %>%
arrange(body_mass_g)**
 - `penguins %>%`
 - `mutate(body_mass_kg=body_mass_g/1000)`
 - `select()`
 - `arrange()`
 - Can use across multiple columns with `across`
 - `mutate(across(ends_with("mm"), ~./10))`
 - Be sure to convert data to its now right label with `select()`
 - `select(species,`
 - `bill_length_cm=bill_length_mm,`
 - `bill_depth_cm=bill_depth_mm,`
 - `flipper_length_cm=flipper_length_mm)`
 - “`~./10`” is saying that whatever follows `~` is the formula thats being used for the colu and `.` shows the column being used is the current one

`clean_names()` - from `janitor` package so we don't need to rename a bunch of data

- Removes underscores, spaces, just saves time
- Use whenever you import data
- Doesn't change the data itself
 - Use `mutate` again for that
 - `mammals %>%
mutate(across(c("order", "family"), tolower)) #specific columns`
 - Do this to change all columns to lowercase but be careful with numeric
 - `mammals %>%
mutate(across(everything(), tolower))`

- Use the where() function to only use the everything function on certain data points
- mutate(across(where(is.character), tolower))

if_else() : if something is true, it'll do something, if not, it won't

- Use this because many people used -999 for unknown observations instead of NA
- Blank expressions will automatically be NA in R
- Change with this and mutate
- If blank is this value, change to NA and leave alone if not
- mammals %>%

```
select(genus, species, newborn) %>%
```

```
mutate(newborn_new=ifelse(newborn==-999.00,NA, newborn))
```


lab 7 and 8 (midterm week)

Filter is used for determining size, species, any restrictions (bigger than or smaller than blank)
Arrange puts it in a certain order

```
msleep %>%  
  filter(bodywt>150) %>%  
Heps put the data in one word
```

```
  summarize(mean_bodywt_lg=mean(bodywt))  
Wraps it into a single number
```

Wrap any function to get clean data

```
  msleep %>%  
    filter(bodywt>150) %>%  
      summarize(mean_bodywt_lg=mean(bodywt),  
                min_bodywt_lg=min(bodywt),  
                max_bodywt_lg=max(bodywt),  
                sd_bodywt_lg=sd(bodywt),  
                total=n())
```

Summarize is used to find mean, min, or at least confirming which vocab represents it

n_distinct() displays the number of distinct observations? (only the unique genus(species, or data)

Distinct shows two things as separate even if they have the same name

Indistinct gives count

Indistinct with a pull gives the vectors

Midterm Vocab

Lab 4:

Data on making each type of graph

Lab 5;

Making data frames with only specific data

Part one is using select and where

Part two uses filter and arrange

Lab 6

“Lm” makes a linear model