BIS 15L R Markdown Cheatsheet

Lab2

2_1

Is integer?:

is.integer(my_numeric)

Create new object as integer:

my_integer <- as.integer(my_numeric)

Check for NA:

is.na(my_missing)

anyNA(my_missing)

Calculate without NA:

```{r}

mean(herbivores$mean.hra.m2, na.rm = T)

```

## 2_2

Generate a sequence of number:

**my_vector_sequence <- c(1:100)**

Pull out a vector:

**Days_of_the_week[3]**

# Lab 3

## 3_1:

Build data frame:

Combine vectors

```{r}
**hbirds <- data.frame(Sex, Length, Weight)**
**hbirds**
```

Column names:

names(hbirds)
```

Dimension of the frame:

dim(hbirds)

Structure of the data frame:

str(hbirds)

Rename:

```{r}
hbirds <- data.frame(sex = Sex, length_in = Length, weight_oz = Weight)
#renaming will become more helpful in later labs
names(hbirds)
```

OR

```{r}
superhero_info <- rename(superhero_info, gender = "Gender", eye_color = "Eye color", race = "Race", hair_color = "Hair color", height = "Height", pulisher = "Publisher", skin_color = "Skin color", alignment = "Alignment", weight = "Weight")
superhero_info
```

Select data:

First row:

hbirds[1,]

Third column:

```{r}
**hbirds[,3]**
```

Select value using $ sign:

```{r}
**w <- hbirds$weight_oz**
**mean(w)**
```

Adding a new column:

```{r}
**hbirds<- rbind(hbirds, new_bird)**
hbirds
```

Writing a csv file:

```{r}
**write.csv(hbirds, "hbirds_data.csv", row.names = FALSE)**
```

3_2:

Change a column to factor and show the level:

```{r}
**hot_springs$scientist <- as.factor(hot_springs$scientist)**
**levels(hot_springs$scientist)**
```

Summary function:

```{r}
**summary(fish)**
```

Glimpse function:

```{r}
**glimpse(fish)**
```

Number of rows:

```{r}
**nrow(fish) #the number of rows or observations**
```

Number of columns:

```{r}
**ncol(fish) #the number of columns or variables**
```

Head function:

Give the first n row of the data frame

```{r}
**head(fish, n = 10)**
```

Tail function:

```{r}
**tail(fish, n = 10)**
```

Table function:

Produces fast counts of the number of observations in a variable

```{r}
**table(fish$lakeid)**
```

Filter function:

Pulling out observations that meet specific criteria in a variable

little_fish <- filter(fish, length<=100)

Lab 4

4_1:

Data structure:

```{r}
glimpse(fish)
```

```{r}
str(fish)
```

```{r}
summary(fish)
```

```{r}
names(fish) (Column names)
```

dplyr:

The first package that we will use that is part of the tidyverse is `dplyr`. `dplyr` is used to transform data frames by extracting, rearranging, and summarizing data such that they are focused on a question of interest. This is very helpful, especially when wrangling large data, and makes dplyr one of most frequently used packages in the tidyverse. The two functions we will use most are `select()` and `filter()`.

Select:

select(fish, "lakeid", "scalelength")

To select a range of column:

select(fish, fish_id:length)

To select everything except:

select(fish, -fish_id, -annnumber, -length, -radii_length_mm)

To contain certain characters:

select(fish, contains("length"))

To start with certain characters:

select(fish, starts_with("radii"))

More of them:

matches() = Select columns that match a regular expression
one_of() = Select columns names that are from a group of names
ends_with() = Select columns that end with a character string

Regex:

column contains a letter, followed by a subsequent string
select(fish, matches("a.+er")) # names start with a end with er

Select based on class of data:

select_if(fish, is.numeric)

"Not" a class of data:

select_if(fish, ~!is.numeric(.))

HW:

Change the class of the variables `taxon` and `order` to factors and display their levels.

```{r}
**homerange$taxon <- as.factor(homerange$taxon)**
homerange$taxon
```

Lab 5

5_1:

Pipes:

shortcut: shift + command + M

```{r}
fish %>%
  select(lakeid, scalelength) %>%
  filter(lakeid == "AL")
```

List all of the superheros that are not human:

```{r}
superhero_info %>%
  filter(race != "Human")
```

Arrange:

arrange(scalelength) (Ascending)
arrange(desc(scalelength)) (Descending)

Mutate:

Mutate allows us to create a new column from existing columns in a data frame

mutate(length_mm = length*10)

mutate_all():

mutate_all(tolower) (Mutate all observations to lowercase)

Specify specific columns:

mutate(across(c("order", "family"), tolower))

Ifelse:

With `ifelse()`, you first specify a logical statement, afterwards what needs to happen if the statement returns `TRUE`, and lastly what needs to happen if it's `FALSE`.

mutate(newborn_new = ifelse(newborn == -999.00, NA, newborn))%>%

Within the parentheses, first comes the condition, next comes what to replace when the condition is met, and the last comes what happens if the condition doesn't meet.

5_2:

Know the data, how do we take out NA, spaces...:

superhero_info <- readr::read_csv("data/heroes_information.csv", *na = c("", "-99", "-"))*

Janitor: help cleans the data, especially renaming columns.

library("janitor")
superhero_powers <- janitor::clean_names(superhero_powers)

Tabyl:

tabyl(superhero_info, alignment) # show both counts and percentages
Within the parentheses, the first argument is the data frame, second is the column.

HW5:

Filter certain row(s) with all TRUE variables.

```{r}
superhero_powers %>%
  filter(hero_names == "Anti-Spawn") %>%
  select_if(all_vars(.=="TRUE"))
```

Lab 6

Warmup

Skip 2 rows:

ecosphere <- read_csv("data/ecs21351-sup-0003-SupplementS1.csv", skip=2)

6_1

Skimr:

library("skimr")

Get rid of NA:

filter(!is.na(vore))

Skim:

skim(msleep24)

Histograms:

hist(msleep24$sleep_total_24)

Tabyl use for multi variable:

tabyl(vore, order)

Summarize:

```{r}
msleep %>%
  filter(bodywt > 200) %>%
  summarize(mean_sleep_lg = mean(sleep_total),
            min_sleep_lg = min(sleep_total),
            max_sleep_lg = max(sleep_total),
            total = n()) (Total number of observations)
```

top_n():

Filter out the top n values.

n_distinct():

Presenting the number of distinct **observations** (NOT individuals)
summarize(**n_genera=n_distinct(genus)**)

Ex. number of distinct genera over 100 in body weight.

```{r}
msleep %>%
 filter(bodywt > 100) %>%
 **summarize(n_genera=n_distinct(genus))**
```

Ex. number of genera are represented in the msleep data frame.

```{r}
msleep %>%
 **summarize(n_genera=n_distinct(genus))**
```

OR

```{r}
**n_distinct(msleep$genus)**
```

First:

first() (returns first value in a column)

Last:

last() (returns last value in a column)

Group by:

group_by(vore)

6_2

Count:

An easy way of determining how many observations you have within a column.

```{r}
penguins %>%
 **count(island, sort = T) #sort=T sorts the column in descending order**
```

Count with combination of columns.

```{r}
superhero_powers %>%
 **count(accelerated_healing & durability & super_strength)**
```

Across multiple variables:

```
penguins %>%
 **count(island, species, sort = T) # sort=T will arrange in descending order**
```

Use of tabyl for two variables:

```{r}
penguins %>%
 **tabyl(species, island)** %>%
 adorn_percentages() %>%
 adorn_pct_formatting(digits = 2) # 2 decimal places
```

Find Specific strings:

filter(stringr::str_detect(asfis_species_name, "Sardina"))

Lab 7

7_1

Across:

```{r}
penguins %>%
 summarize(across(c(species, island, sex), n_distinct))
```

```{r}
penguins %>%
 summarize(across(contains("mm"), mean, na.rm=T))
```

```{r}
penguins %>%
 summarize(across(!c(species, island, sex),
                  n_distinct))
```

7_2

Dealing with NA:

Original method:

```{r}
msleep %>%
 summarize(number_nas = sum(is.na(msleep)))
```

Replacing values with NA:

```{r}
amniota_tidy <- amniota %>%
 na_if("-999")
```

Change values to NA:

```{r}
msleep %>%
 mutate(conservation_modified = na_if(conservation, "domesticated"))%>%
```

```
 amniota
 %>%
                    naniar::replace_with_na_all(condition = ~.x == -999)
```

Naniar:

```{r}
```

naniar::miss_var_summary(amniota_tidy) #how many NAs with percentages

```

```{r}
amniota %>% **summarize(number_nas = sum(is.na(amniota))) # how many NAs**
```


```{r}
amniota %>%
 **naniar::replace_with_na_all(condition = ~.x == -999)**
```

```{r}
amniota_tidy %>%
 select(genus, species, female_maturity_d) %>%
 **mutate(female_maturity_d2=ifelse(female_maturity_d<0, NA, female_maturity_d))**%>%
 arrange(female_maturity_d2)
```

```{r}
amphibio %>%
 select(fos, ter, arb, aqu) %>%
 **summarise_all(~(sum(is.na(.)))) # calculate the number of NAs in each column**
```

fos <int>	ter <int>	arb <int>	aqu <int>
6053	1104	4347	2810

1 row
```

Lab 8

8_1

Here(): trace root directory:

heartrate <- read_csv(here("data2", "heartrate.csv"))

Pivot_longer:

```{r}
heartrate %>%
  pivot_longer(-patient, # patient will not pivot
               names_to =  "drug", # make new colunm name
               values_to =  "heartrate"
               )
```

slice_max():

```{r}
mean_entero %>%
  pivot_wider(names_from=site,
              values_from=mean_enterococci_cfu_100ml) %>%
  filter(year==2018) %>%
  pivot_longer(-year,
               names_to = "site",
               values_to = "mean_enterococci_cfu_100ml") %>%
  slice_max(mean_enterococci_cfu_100ml, n=3) # select top 3 greatest value
```

year <chr>	site <chr>	mean_enterococci_cfu_100ml <dbl>
2018	South Maroubra Rockpool	112.18750
2018	Little Bay Beach	59.06250
2018	Bronte Beach	43.41667

3 rows

A range of columns:

```{r}
billboard2 <-
  billboard %>%
  pivot_longer(wk1:wk76, # a range of columns
               names_to = "week",
               values_to = "rank",
               values_drop_na = TRUE #this will drop the NA's
               )
billboard2
```

By a prefix:

```{r}
billboard %>%
  pivot_longer(
  cols = starts_with("wk"),
  names_to = "week",
  names_prefix = "wk",
  values_to = "rank",
  values_drop_na = TRUE)
```

More than one variable in a column name:

```{r}
qpcr_untidy %>%
  pivot_longer(
    exp1_rep1:exp3_rep3,
    names_to= c("experiment", "replicate"),
    names_sep="_",
    values_to="mRNA_expression"
  )
```

More than one value or observation in a row:

```{r}
length_data %>%
  transform(length = str_split(length, ";")) %>%
  unnest(length)
```

8_2

Separate:

```{r}
heartrate2 %>%
  separate(patient, into= c("patient", "sex"), sep = "_")
```

```{r}
sydneybeaches_long %>%
  separate(date, into=c("day", "month", "year"), sep="/")
```

```

| site <chr> | day <chr> | month <chr> | year <chr> | enterococci_cfu_100ml <dbl> |
|---|---|---|---|---|
| Clovelly Beach | 02 | 01 | 2013 | 19 |
| Clovelly Beach | 06 | 01 | 2013 | 3 |
| Clovelly Beach | 12 | 01 | 2013 | 2 |
| Clovelly Beach | 18 | 01 | 2013 | 13 |
| Clovelly Beach | 30 | 01 | 2013 | 8 |
| Clovelly Beach | 05 | 02 | 2013 | 7 |
| Clovelly Beach | 11 | 02 | 2013 | 11 |
| Clovelly Beach | 23 | 02 | 2013 | 97 |
| Clovelly Beach | 07 | 03 | 2013 | 3 |
| Clovelly Beach | 25 | 03 | 2013 | 0 |

1-10 of 3,690 rows Previous 1 2 3 4 5 6 ... 100 Next

## Unite:

```{r}
heartrate3 %>%
  unite(patient_sex, "patient", "sex", sep = " ")
```

Pivot_wide:

```{r}
tb_data %>%
  pivot_wider(names_from = "key", #the observations under key will become new columns
              values_from = "value")
```

```{r}
gapminder %>%
  select(country, year, pop) %>%
  filter(year==1952 | year==2007) %>%
  pivot_wider(names_from = year,
              names_prefix = "yr_", # set name prefix for new columns
              values_from = pop)
```
```

country <fctr>	yr_1952 <int>	yr_2007 <int>
Afghanistan	8425333	31889923
Albania	1282697	3600523
Algeria	9279525	33333216
Angola	4232095	12420476
Argentina	17876956	40301927
Australia	8691212	20434176
Austria	6927772	8199783
Bahrain	120447	708573
Bangladesh	46886859	150448339
Belgium	8730405	10392226

```{r}
gapminder %>%
 select(country, year, pop) %>%
 filter(year==1952 | year==2007) %>%
 pivot_wider(names_from = year,
        values_from = pop) %>%
 mutate(delta= `2007`-`1952`) %>%  # if prefix not added before number use ``
 arrange(desc(delta))
```

With different variables:

```{r}
edu_level %>%
pivot_wider(names_from = "education_level", #new column names come from the education_level column
        values_from = c(mean_income, n)) #values come from two separate columns
```

Lab 9:

9_1:

Geom_bar:

```{r}
life_history %>%
  ggplot(aes(x=order))+
  geom_bar()+
```

Geom_col:

```{r}
life_history %>%
  count(order, sort=T) %>%
  ggplot(aes(x=order, y=n))+
  geom_col()
```

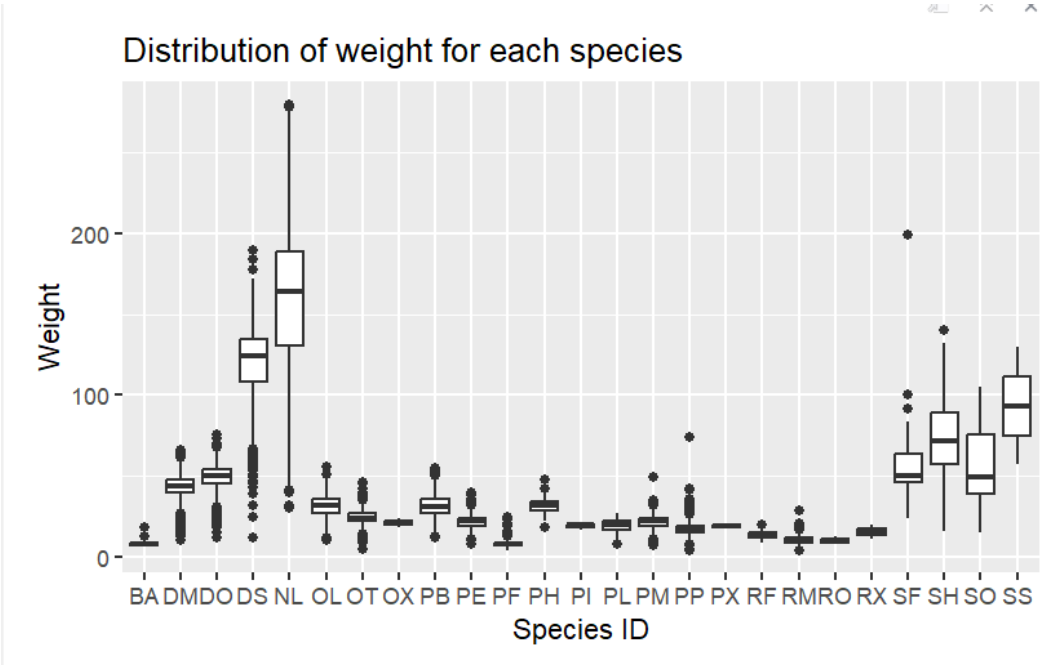
9_2:

Geom_boxplot:

```{r}
life_history %>%
  ggplot(aes(x=order, y=mass))+
  geom_boxplot(na.rm = T)
```

```{r}
deserts %>%
```

```
  filter(weight!="NA") %>%  # remove NA values
  ggplot(aes(x=species_id, y=weight)) +
  geom_boxplot()+
  labs(title = "Distribution of weight for each species",
       x = "Species ID",
       y = "Weight")
```



Lab 10

10_1:

Clean names when load the data:

```
life_history <- read_csv("data/mammal_lifehistories_v2.csv", na="-999") %>%
clean_names()
```

Flip:

coord_flip()

Scientific notation:

options(scipen = 999)

Scale y by log10:

scale_y_log10()

Order the graph by length of bar:

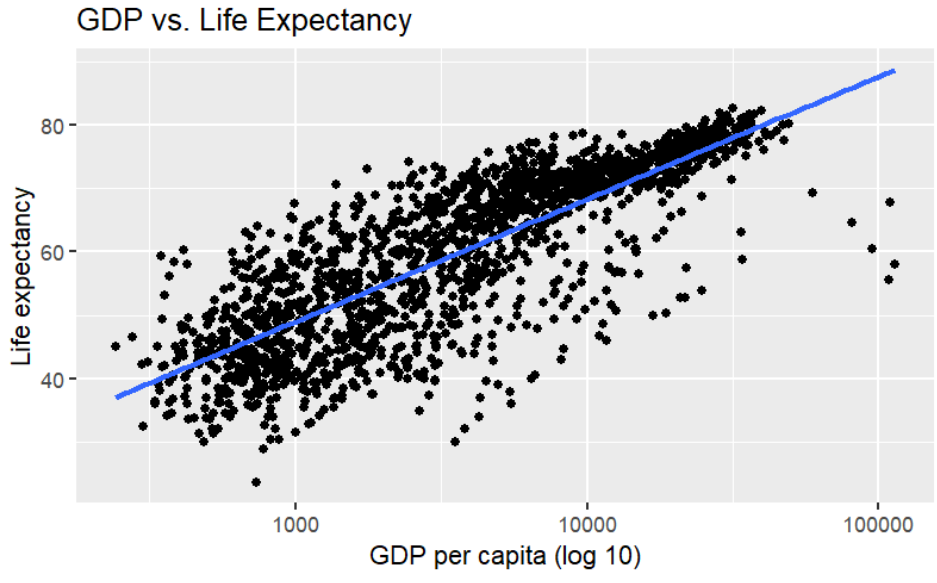
ggplot(aes(x=reorder(order, mean_mass), y=mean_mass))

Geom_point:

```{r}
life_history %>%
  ggplot(aes(x=gestation, y=wean_mass))+
  geom_point(na.rm = T)
```

Geom_smooth:

```{r}
gapminder %>%
  ggplot(aes(x=gdpPercap, y=lifeExp))+
  geom_point()+
  scale_x_log10()+ # balance the plot visualization
  geom_smooth(method=lm, se=F)+ # add regression line
  labs(title = "GDP vs. Life Expectancy",
       x = "GDP per capita (log 10)",
       y = "Life expectancy")
```



Labels:

Labs:

```
labs(title = "Elephant Age vs. Height",
     x = "Age",
     y = "Height")
```

Theme:

```
theme(plot.title = element_text(size = rel(1.25), hjust = 0.5))
```

Fill:

```{r}
elephants %>%
  ggplot(aes(x=sex, fill=sex))+
  geom_bar()
```

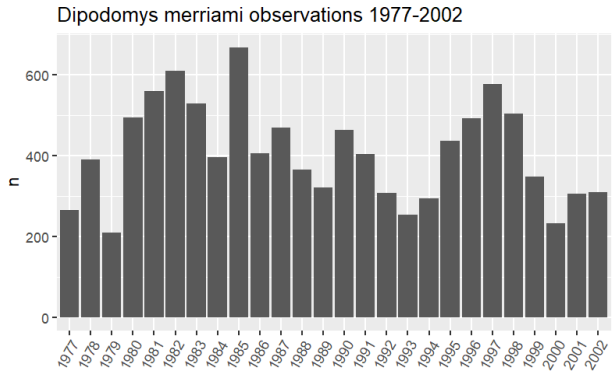
Size:

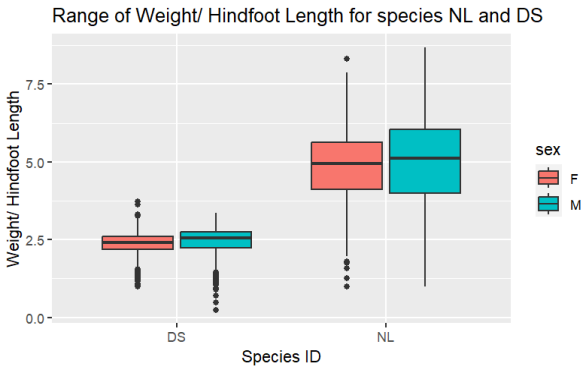
Size of points relative to a continuous variable:

```{r}
life_history %>%
  ggplot(aes(x=gestation, y=log10(mass), size=mass))+
  geom_point(na.rm = T) # remove warning
```

```{r}
deserts %>%
  filter(species_id=="DM") %>%
  group_by(year) %>%
  summarize(n_samples=n()) %>%  # count frequency of samples of DM each year
```

```
 ggplot(aes(x=as.factor(year), y=n_samples)) + geom_col()+
 theme(axis.text.x = element_text(angle = 60, hjust = 1)) +
 labs(title = "Dipodomys merriami observations 1977-2002",
     x = NULL,
     y= "n")
```






```{r}
deserts %>%
 filter(species_id=="NL" | species_id=="DS") %>%
 filter(weight!="NA" & hindfoot_length!="NA" & sex!="NA") %>%
 mutate(ratio=weight/hindfoot_length) %>%  # create new column of ratio
 select(species_id, sex, weight, hindfoot_length, ratio) %>%
 ggplot(aes(x=species_id, y=ratio, fill=sex)) + geom_boxplot()+
 labs(title = "Range of Weight/ Hindfoot Length for species NL and DS",
     x = "Species ID",
     y = "Weight/ Hindfoot Length")
```

10_2:

Size of Geom_point:

geom_point(size=2)

Maps Shapes in Geom_point:

geom_point(aes(shape=thermoregulation, color=thermoregulation), size=1.75)

Position = “dodge”:

Compare side by side:

ggplot(aes(x = taxon, fill = trophic.guild)) + geom_bar(position = "dodge")

X-axis aes:

theme(axis.text.x = element_text(angle = 60, hjust = 1))

Scale to percentage:

scale_y_continuous(labels = scales::percent)

Group:

Same as “fill” but doesn’t add color:

```{r}
homerange %>%
  ggplot(aes(x = class, y = log10.mass, group = taxon)) +
  geom_boxplot()
```

Lab 11:

11_1

Geom_line:

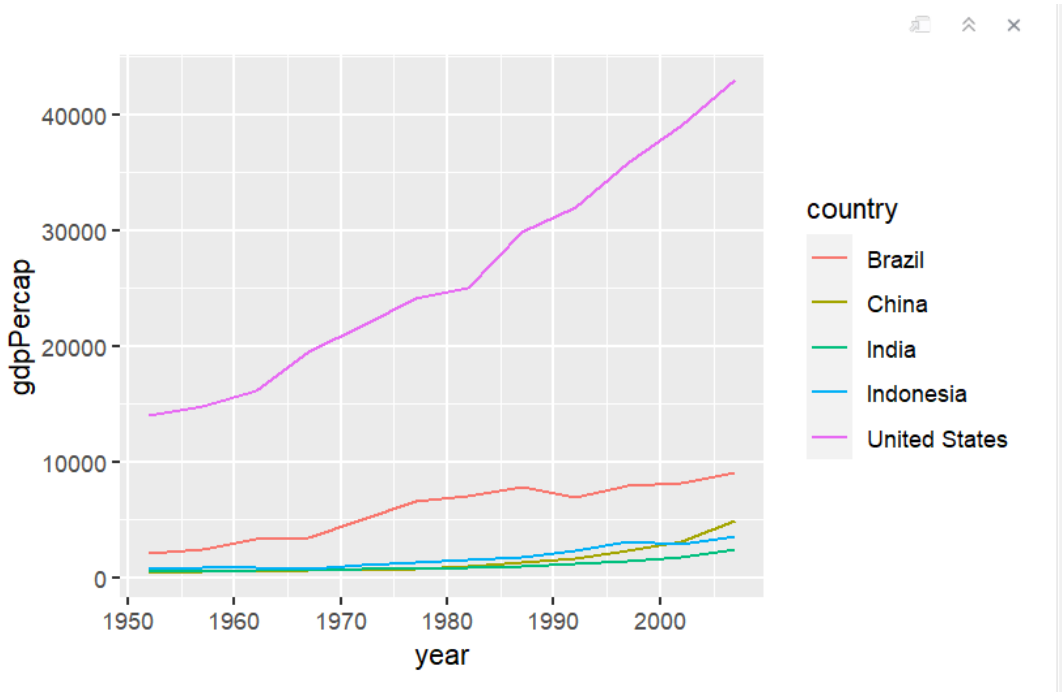
Factor the x-axis to makes it all shown:

```{r}
deserts2 <- deserts %>% mutate(year=as_factor(year))
```

Geom_line:

```{r}
deserts2 %>%
  ggplot(aes(x=year, y=n, group=species_id, color=species_id))+
  geom_line()+
  geom_point(shape=9)
```

```{r}
gapminder %>%
  filter(country=="China" | country=="India" | country=="United States" | country=="Indonesia" | country=="Brazil") %>%  # combine variables while filtering
  select(country, year, pop) %>%
  ggplot(aes(x=year, y=pop, color=country))+
  geom_line()
```



Geom_histogram:

```{r}
homerange %>%
  ggplot(aes(x = log10.mass)) +
  geom_histogram(alpha = 0.4, color = "black", fill = "deepskyblue4", bins=40)+
  labs(title = "Distribution of Body Mass")
```

Alpha:

Alpha = 0.4 (transparency = 0.4)

Color available:

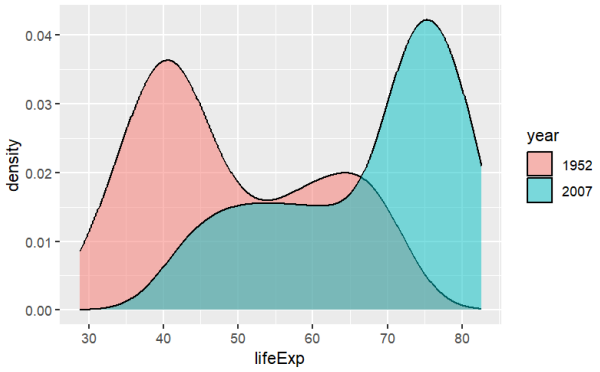
```{r}
grDevices::colors()
```

```

Geom_density: # find the distribution of the variables

```{r}
homerange %>%
  ggplot(aes(x = log10.mass)) +
  geom_density(fill="deepskyblue4", alpha  =0.4, color = "black")+
  labs(title = "Distribution of Body Mass")
```

```{r}
gapminder %>%
  filter(year==1952 | year==2007) %>%
  mutate(year=as.factor(year)) %>%  # turn year into factor
  ggplot(aes(x=lifeExp, group=year, fill=year))+
  geom_density(alpha=0.5) # alpha -> adjust transparency of the color
```



Case_when:

Put observations within certain range into new variables:

```{r}
homerange <- homerange %>%
        mutate(mass_category = case_when(log10.mass <= 1.75 ~ "small",
                                         log10.mass > 1.75 & log10.mass <= 2.75 ~ "medium",

```
                                    log10.mass > 2.75 ~ "large"))
```

Quartiles:

```{r}
library(gtools)
quartiles <- quantcut(homerange$log10.hra)
table(quartiles)
```

11_2

Themes:

```{r}
p+theme_linedraw()+
  theme(axis.text.x = element_text(angle = 60, hjust=1)) # hjust-> space between names on x axis, element_text()-> for adjusting text style on x axis
```

Legend:

```{r}
p+theme_linedraw()+
  theme(legend.position = "bottom",
        axis.text.x = element_text(angle = 60, hjust=1))
```

All ggthemes:

```
ls("package:ggthemes")[grepl("theme_", ls("package:ggthemes"))]
```

R Color Brewer:

+`scale_colour_brewer()` is for points
+`scale_fill_brewer()` is for fills

display.brewer.pal(4,"GnBu")

scale_fill_brewer(palette = "Paired")

Website: http://colorbrewer2.org/#type=sequential&scheme=BuGn&n=3

Manually setting color:

colors <- paletteer::palettes_d_names # Get the names of the color.

my_palette <- paletteer_d("ggprism::flames") # Store the targeted color.

scale_fill_manual(values=my_palette) for bar plots

scale_color_manual(values=my_palette) for point plots

Adjusting x, y axis limits:

xlim(0, 4) +
ylim(1, 6)

Faceting:

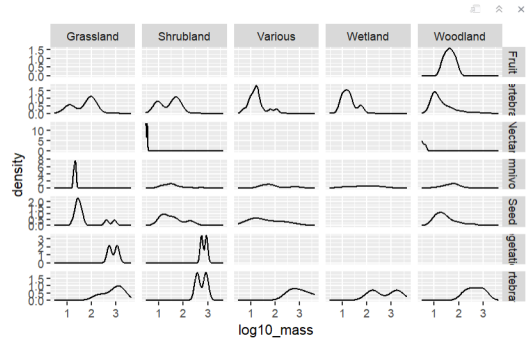
facet_wrap(~migratory_strategy, ncol=4)+

facet_grid(migratory_strategy~.)+

facet_grid(.~migratory_strategy)+

facet_grid(diet~habitat, scales = "free_y") **# row~col**

```{r}
ecosphere %>%
  ggplot(aes(x=log10_mass))+
  geom_density()+
  facet_grid(diet~habitat, scales = "free_y")
```




```{r}
ecosphere %>%
  ggplot(aes(x=log10_mass))+
  geom_density()+
  **facet_wrap(~habitat, scales = "free_y")** # wrap
```