Clase 4

Operaciones en R

Miriam Lerma

Febrero 2021

1. Intro

- Operaciones sencillas.
- Ejercicios usando operaciones.
- Datos ordenados.
- Ejercicios con funciones de tidyverse.

Ustedes

- Conocimientos básicos de R (saben abrirlo, cargar paquetes y datos)
- Quieren hacer algunas operaciones matemáticas usando R.
- Quieren tener un dataframe que les sirva para hacer análisis y gráficos.

Créditos

-Material basado en el libro:
R4DS, editado por Riva Quiroga

-Y materiales de RLadies
Zero to Hero

-Presentaciones de tidyverse:
María Paula Caldas
RLadiesBuenosAires

-Imágenes adicionales:
Unsplash
Portada por StellrWeb
Allison Horst

1. CRAN

Los datos de pinguinos solo están en github, aun no en CRAN

```
install.packages(remotes)
remotes::install_github("cienciadedatos/datos")
```

Les va a aparecer:
Downloading GitHub repo cienciadedatos/datos@HEAD
Enter one or more numbers, or an empty line to skip updates:
Pueden darle 1 si quieren, pero tarda un poquito.

1. CRAN

Carguen los datos de pinguinos 🐧, los vamos a usar mas adelante.

```
library(datos)
Pingus<-datos::pinguinos
```

Otra opción es usar read_csv desde su computadora.

1. CRAN

CRAN es un acrónimo de **C**omprehensive **R** **A**rchive **N**etwork.

- CRAN tiene hasta Noviembre de 2020, 16mil paquetes.
- Cualquiera puede crear su paquete y someterlo a CRAN.
- Cada paquete pasa por varias pruebas, y si pasa todas las pruebas queda disponible en el Archive.

2. Operaciones

2.1. Operaciones basicas

Sumas

```
15+6
```

```
## [1] 21
```

Restas

```
4-6
```

```
## [1] -2
```

2.1. Operaciones basicas

Divisiones

```
1700/8
```

```
## [1] 212.5
```

Multiplicaciones

```
20*20
```

```
## [1] 400
```

2.2. Operaciones usando objetos

```
Personas<-5+6
Pizzas<-5*12
```

Cuantos pedazos le toca a cada quien?

```
Personas/Pizzas
```

```
## [1] 0.1833333
```

2.3. Funciones comunes

Media

```
Temperatura<-c(34,45,67,20)
mean(Temperatura)
```

```
## [1] 41.5
```

Mediana

```
median(Temperatura)
```

```
## [1] 39.5
```

Desviacion estandar

```
sd(Temperatura)
```

```
## [1] 19.84103
```

2.3. Funciones comunes

Rangos

```
range(Temperatura)
```

```
## [1] 20 67
```

Minimo

```
min(Temperatura)
```

```
## [1] 20
```

Maximo

```
max(Temperatura)
```

```
## [1] 67
```

2.4. Buscar ayuda

```
mean(1,3,6,9,12)
```

```
## [1] 1
```

Porque me da 1? no me parece correcto.

Pregúntale a R

```
?mean
```

Las instrucciones aparecerán en la esquina inferior derecha, en la pestaña **Help**.

Hay que poner la c de ***concatenar***

```
mean(c(1,3,6,9,12))
```

```
[1] 6.2
```

Ya tiene más sentido.

2.4. Buscar ayuda

Parte de la fortalezas de R y porque es tan usado es que hay muchos sitios para pedir ayuda.

Lo primero que hay que hacer es tener paciencia y revisar que no hayamos escrito algo mal.

Si no funciona, otra opción es literalmente copiar el error que nos aparece y escribirlo en google.

Fuentes confiables son:

- stackoverflow
  Seguro lo vamos a usar en algún momento durante la clase.

- twitter
  Hashtag: #rstatsES (en español) o #rstats (en inglés).

Ejercicios 

Realizar operaciones en R usando objetos

2.5. Ejercicios

Tengo tres gatos y cada uno come 2 latas de comida, cuantas latas tengo que comprar?

```
Gatitos<-3
Latas<-2
Gatitos*Latas
```

Tengo 4 perros y ya se cuantos gatos, cuantas mascotas tengo en total?

```
Perritos<-4
Mascotas<-Perritos+Gatitos
Mascotas
```

2.5 Ejercicios

Quiero calcular cuantos kilos de croquetas debo comprar si mis perritos (que son 3) se comen 0.5 Kg al día y quiero ir al super cada 15 días por lo de la pandemia?

```
#Perritos<-
#Croquetas<-
#Dias<-
```

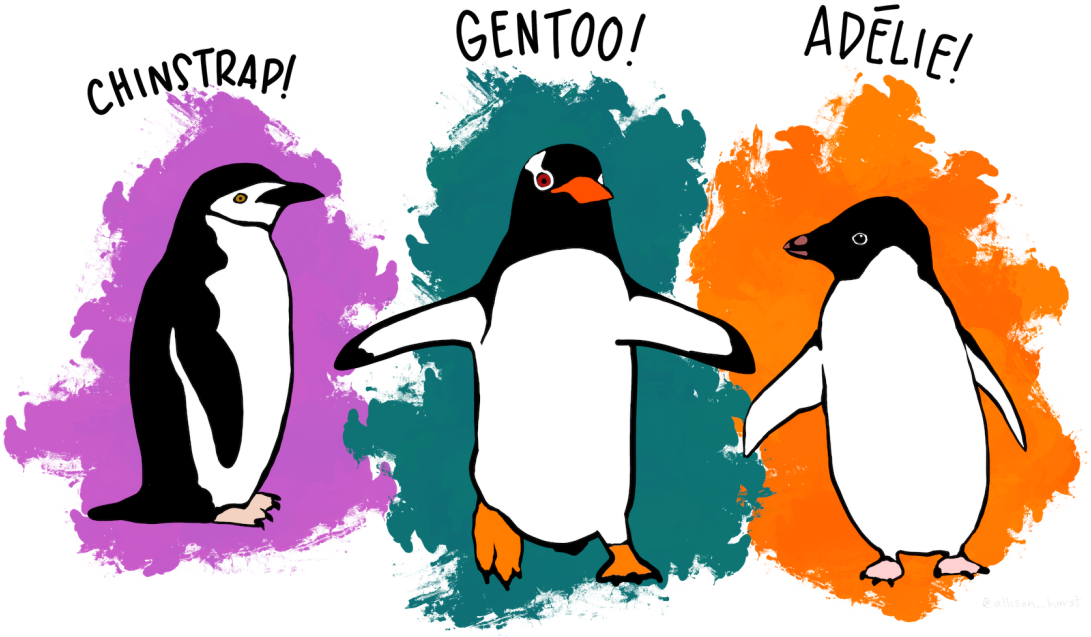
```
#(Perritos*Croquetas)*Dias
```

Hasta le podríamos agregar el precio de las croquetas, si quisiéramos.

2.6. Datos pinguinos.

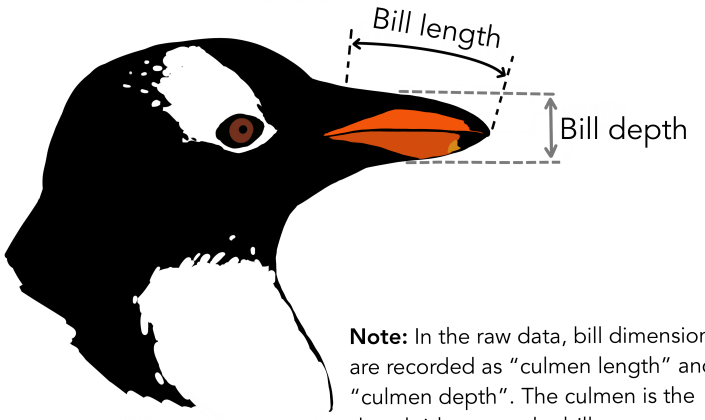
Inspeccionen los datos de pinguinos.

```
library(datos)
Pingus<-pinguinos
```



2.6. Datos pinguinos.

Incluye datos de hembras y machos, de tres especies diferentes en tres islas diferentes. Incluye datos de medidas como masa corporal, longitud del pico, largo de la aleta.



Note: In the raw data, bill dimensions are recorded as "culmen length" and "culmen depth". The culmen is the dorsal ridge atop the bill.

2.6. Datos pinguinos.

Como calculo el **rango** de la masa corporal de los pinguinos?
Cuidado con los NAs!, na.rm me sirve para ignorarlos

```
range(Pingus$masa_corporal_g,
      na.rm=TRUE)
```

Cuanto pesan en **promedio** los pinguinos en kilos?
Guardarlo en un objeto y transformar de kilos a gramos.

```
PromedioPesoPingus<-mean(Pingus$masa_corporal_g,na.rm=TRUE)
PromedioPesoPingus/1000
```

```
## [1] 4.201754
```

2.7. Ejercicios pinguinos.

- Como calculo el **promedio** del largo del pico de los pinguinos?

```
m**n(Pingus$largo_pico_mm,na.rm=TRUE)
```

- Como obtengo el **rango** de valores del largo del pico de los pinguinos?

```
r***e(Pingus$*****,na.rm=TRUE)
```

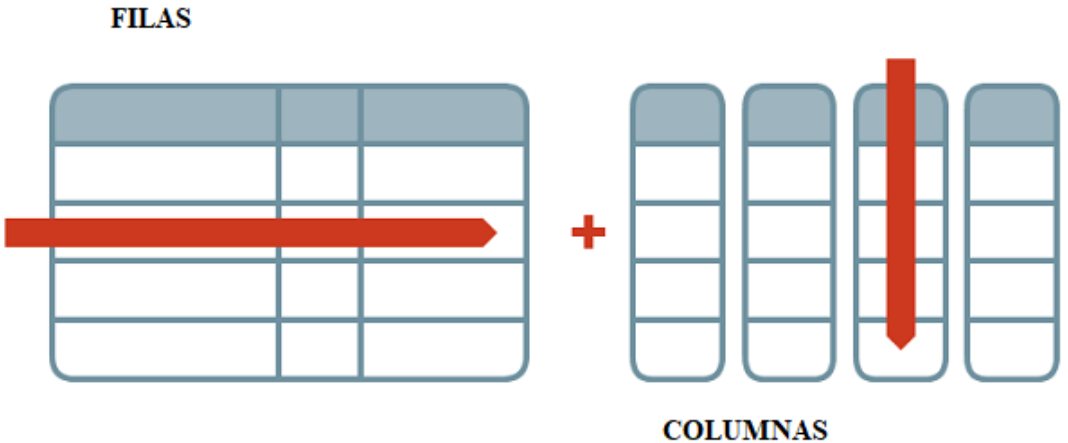
- Como obtengo el valor **mínimo** del largo del pico de los pinguinos?

```
m**(Pingus$*****,na.rm=TRUE)
```

3. Inspeccionar data frames

3.1. Inspeccionar data frames

- Ver mis primeras y ultimas filas
- Ver filas especificas
- Ver columnas especificas
- Extraer específicos de acuerdo a su columna y fila



Fuente: frases333

3.2. head() y tail()

Para ver mis primeras y ultimas filas

```
head(Pingus)
```

```
## # A tibble: 6 x 8
##   especie isla   largo_pico_mm alto_pico_mm largo_aleta_mm masa_corporal_g
##   <fct>   <fct>          <dbl>        <dbl>          <int>           <int>
## 1 Adelia  Torge~          39.1         18.7            181            3750
## 2 Adelia  Torge~          39.5         17.4            186            3800
## 3 Adelia  Torge~          40.3         18              195            3250
## 4 Adelia  Torge~          NA           NA               NA              NA
## 5 Adelia  Torge~          36.7         19.3            193            3450
## 6 Adelia  Torge~          39.3         20.6            190            3650
## # ... with 1 more variable: anio <int>
```

```
tail(Pingus)
```

```
## # A tibble: 6 x 8
##   especie isla  largo_pico_mm alto_pico_mm largo_aleta_mm masa_corporal_g se
##   <fct>   <fct>         <dbl>        <dbl>          <int>           <int> <f
## 1 Barbijo Dream          45.7         17              195            3650 ho
## 2 Barbijo Dream          55.8         19.8            207            4000 ma
```

3.3. Filas

Las filas se pone entre corchetes en la **primera** posición.

```
(Pingus[1,])
```

```
## # A tibble: 1 x 8
##   especie isla   largo_pico_mm alto_pico_mm largo_aleta_mm masa_corporal_g
##   <fct>   <fct>          <dbl>        <dbl>          <int>           <int>
## 1 Adelia  Torge~          39.1         18.7            181            3750
## # ... with 1 more variable: anio <int>
```

Muéstrame las primeras 3 filas.
El signo de : es como decir "de aquí hasta acá".

```
(Pingus[1:3,])
```

```
## # A tibble: 3 x 8
##   especie isla   largo_pico_mm alto_pico_mm largo_aleta_mm masa_corporal_g
##   <fct>   <fct>          <dbl>        <dbl>          <int>           <int>
## 1 Adelia  Torge~          39.1         18.7            181            3750
## 2 Adelia  Torge~          39.5         17.4            186            3800
## 3 Adelia  Torge~          40.3         18              195            3250
## # ... with 1 more variable: anio <int>
```

3.4. Columnas

Las columnas se pone entre corchetes en la **segunda** posición.

```
head(Pingus[,1])
```

```
## # A tibble: 6 x 1
##   especie
##   <fct>
## 1 Adelia
## 2 Adelia
## 3 Adelia
## 4 Adelia
## 5 Adelia
## 6 Adelia
```

Usando $ y el nombre de la columna.

```
head(Pingus$especie)
```

```
## [1] Adelia Adelia Adelia Adelia Adelia Adelia
## Levels: Adelia Barbijo Papúa
```

3.5. Columnas y filas

Muéstrame un dato especifico [fila, columna]

```
(Pingus[1,1])
```

```
## # A tibble: 1 x 1
##   especie
##   <fct>
## 1 Adelia
```

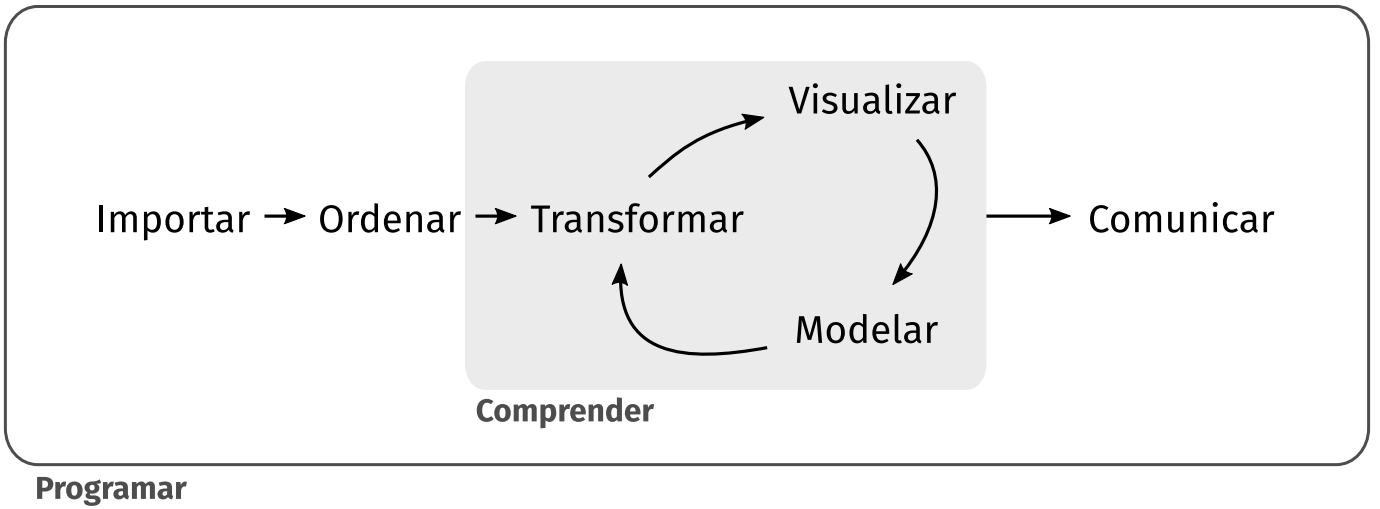
```
(Pingus[3,2])
```

```
## # A tibble: 1 x 1
##   isla
##   <fct>
## 1 Torgersen
```

4. Datos ordenados

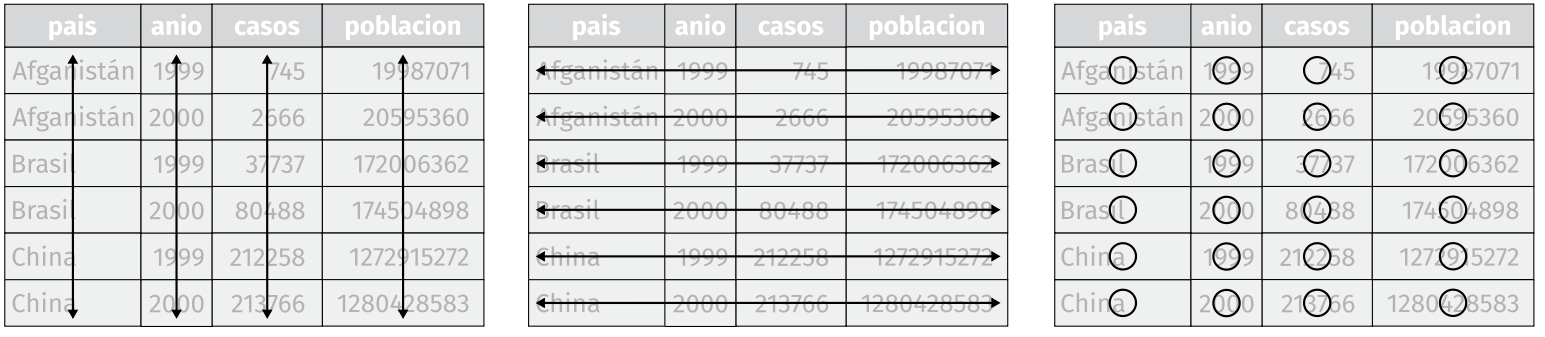
4.1. Ordenar datos

El proceso mas tardado es el de limpieza y preparación de los datos, y se realiza varias veces.



4.2. Datos ordenados

Los datos ordenados ("tidy"), es una estructuración de los conjuntos de datos para facilitar el análisis.



pais	anio	casos	poblacion
Afganistán	1999	745	19987071
Afganistán	2000	2666	20595360
Brasil	1999	37737	172006362
Brasil	2000	80488	174504898
China	1999	212258	1272915272
China	2000	213766	1280428583

variables

observaciones

valores

- Cada variable debe estar columna
- Cada observación es una fila
- Cada tipo de unidad de observación forma una tabla

4.3. Errores comunes

Se recomienda que al almacenar datos:

- Diferentes tipos de variables se almacenen en columnas separadas.

```
MiPrimerAcomodo<-'Miriam (Mujer)'
MiMejorAcomodo<-c('Miriam','Mujer')
```

Es importante que cuando recolecten datos por primera vez piensen en como van a estructurarlos.
Si no el proceso de ordenarlos puede ser tardado 🕐.

4.4. Ordenar

Hay muchas maneras de ordenar y manipular tus datos.
En ingles le dicen 'data wrangling' que significa como rodeo.

Hay muchas maneras de hacer las mismas operaciones. Hoy usaremos el paquete **tidyverse**.

```
library(tidyverse)
```

```
## Warning: package 'tibble' was built under R version 4.0.4
```

```
## Warning: package 'dplyr' was built under R version 4.0.4
```

4.5. Ordenar

Porque usar datos ordenados?

- Muchos comandos se basan en la suposición de que tus datos están ordenados.
- El formato es el esperado para análisis estadísticos.
- Tener datos ordenados ayudan a realizar los gráficos.
- Podemos compartirlos y la otra persona podría entender nuestra tabla mas rápido que en una tabla desordenada.

4.6. Tidyverse

Tidyverse engloba varios paquetes, la mayoría para específicamente para inspeccionar y manipular tus datos.

4.7. Pipe

Vamos a usar mucho el **pipe** un argumento que se usa para encadenar funciones.

En su teclado: strg+alt+M

```
%>%
```

4.8. Manipular

El paquete dplyr nos da una serie de herramientas para **manipular** datos

Las principales funciones, o **verbos** de dplyr, son:

- **count()** para contar
- **select()**, para seleccionar columnas
- **filter()**, para filtrar filas
- **mutate()**, para crear o modificar columnas
- **summarise()**, para resumir información de las columnas

Hoy veremos **count** y **select**, la próxima clase los otros.

4.9. count()

¿Cuantos datos tengo?

```
Pingus %>%
  count()
```

¿Cuantos datos tengo por especie?

```
Pingus %>%
  count(especie)
```

4.10. count()

¿Cuantos datos tengo por isla y por especie?

```
Pingus %>%
  count(isla,especie)
```

```
## # A tibble: 5 x 3
##   isla      especie     n
##   <fct>     <fct>   <int>
## 1 Biscoe    Adelia     44
## 2 Biscoe    Papúa     124
## 3 Dream     Adelia     56
## 4 Dream     Barbijo    68
## 5 Torgersen Adelia     52
```

El argumento **arrange()** nos ayuda ordenarlos

```
Pingus %>%
  count(anio) %>%
  arrange((n))
```

4.11. select()

Con la función **select()**, podemos elegir: Que columnas quiero ver.

```
Pingus %>%
  select(especie) %>%
  head()
```

```
## # A tibble: 6 x 1
##   especie
##   <fct>
## 1 Adelia
## 2 Adelia
## 3 Adelia
## 4 Adelia
## 5 Adelia
## 6 Adelia
```

4.12. select(-)

Que columnas no quiero ver.
El signo - es como decir "menos ese"

```
Pingus %>%
  select(-sexo) %>%
  head()
```

```
## # A tibble: 6 x 7
##   especie isla   largo_pico_mm alto_pico_mm largo_aleta_mm masa_corporal_g
##   <fct>   <fct>          <dbl>        <dbl>          <int>           <int>
## 1 Adelia  Torge~          39.1         18.7            181            3750
## 2 Adelia  Torge~          39.5         17.4            186            3800
## 3 Adelia  Torge~          40.3         18              195            3250
## 4 Adelia  Torge~          NA           NA               NA              NA
## 5 Adelia  Torge~          36.7         19.3            193            3450
## 6 Adelia  Torge~          39.3         20.6            190            3650
```

4.13. select(-)

El signo de admiración ! es como decir "diferente a".

```
Pingus %>%
  select(!sexo) %>%
  head()
```

```
## # A tibble: 6 x 7
##   especie isla   largo_pico_mm alto_pico_mm largo_aleta_mm masa_corporal_g
##   <fct>   <fct>          <dbl>        <dbl>          <int>           <int>
## 1 Adelia  Torge~          39.1         18.7            181            3750
## 2 Adelia  Torge~          39.5         17.4            186            3800
## 3 Adelia  Torge~          40.3         18              195            3250
## 4 Adelia  Torge~          NA           NA               NA              NA
## 5 Adelia  Torge~          36.7         19.3            193            3450
## 6 Adelia  Torge~          39.3         20.6            190            3650
```

4.14. select(:)

Seleccionar columnas que están en medio de varias columnas.
El signo de : es como decir "de aquí hasta acá".

```
Pingus %>%
  select(largo_pico_mm:masa_corporal_g) %>%
  head()
```

```
## # A tibble: 6 x 4
##   largo_pico_mm alto_pico_mm largo_aleta_mm masa_corporal_g
##           <dbl>        <dbl>          <int>           <int>
## 1          39.1         18.7            181            3750
## 2          39.5         17.4            186            3800
## 3          40.3         18              195            3250
## 4          NA           NA               NA              NA
## 5          36.7         19.3            193            3450
## 6          39.3         20.6            190            3650
```

4.15. select(:)

También se pueden usar cadenas de caracteres (strings).
Tanto usando como termina el nombre de la columna.

```
Pingus %>%
  select(ends_with("mm"))
```

```
## # A tibble: 344 x 3
##    largo_pico_mm alto_pico_mm largo_aleta_mm
##            <dbl>        <dbl>          <int>
##  1          39.1         18.7            181
##  2          39.5         17.4            186
##  3          40.3         18              195
##  4          NA           NA               NA
##  5          36.7         19.3            193
##  6          39.3         20.6            190
##  7          38.9         17.8            181
##  8          39.2         19.6            195
##  9          34.1         18.1            193
## 10          42           20.2            190
## # ... with 334 more rows
```

4.16. select(:)

También se pueden usar cadenas de caracteres (strings).
Como seleccionando como inicia el nombre de la columna.

```
Pingus %>%
  select(starts_with("masa"))
```

```
## # A tibble: 344 x 1
##    masa_corporal_g
##              <int>
##  1            3750
##  2            3800
##  3            3250
##  4              NA
##  5            3450
##  6            3650
##  7            3625
##  8            4675
##  9            3475
## 10            4250
## # ... with 334 more rows
```

Ejercicios

```
Pingus %>%
  count()
```

```
Pingus %>%
  count(especie)
```

```
Pingus %>%
  select(especie) %>%
  head()
```

```
Pingus %>%
  select(-sexo) %>%
  head()
```

```
PingusSinSexo<-Pingus %>%
  select(-sexo) %>%
  head()
```

Recapitulando

Esta clase:

- Operaciones matemáticas simples.
- Funciones básicas (mean, median, sd, range).
- Funciones tidyverse (count, select).

Siguiente clase:

- Funciones tidyverse (filter, group_by, summarize)
- Unir dataframes (join)
- Exportar dataframes (write_csv)

Contacto

Para dudas, comentarios y sugerencias:
Escríbeme a miriamjlerma@gmail.com

Este material esta accesible
y se encuentra en mi github y mi página