Sociedad Ecuatoriana de Estadística

"Análisis de Encuestas por Muestreo con R"

Manipulación de datos con R. Paquete "dplyr"

Andrés Peña M.

agpena@colmex.mx

Mayo 2023

X Seminario Internacional de Estadística Aplicada

Tabla de contenidos

1. Data management en *R*

Cómo funciona *R*

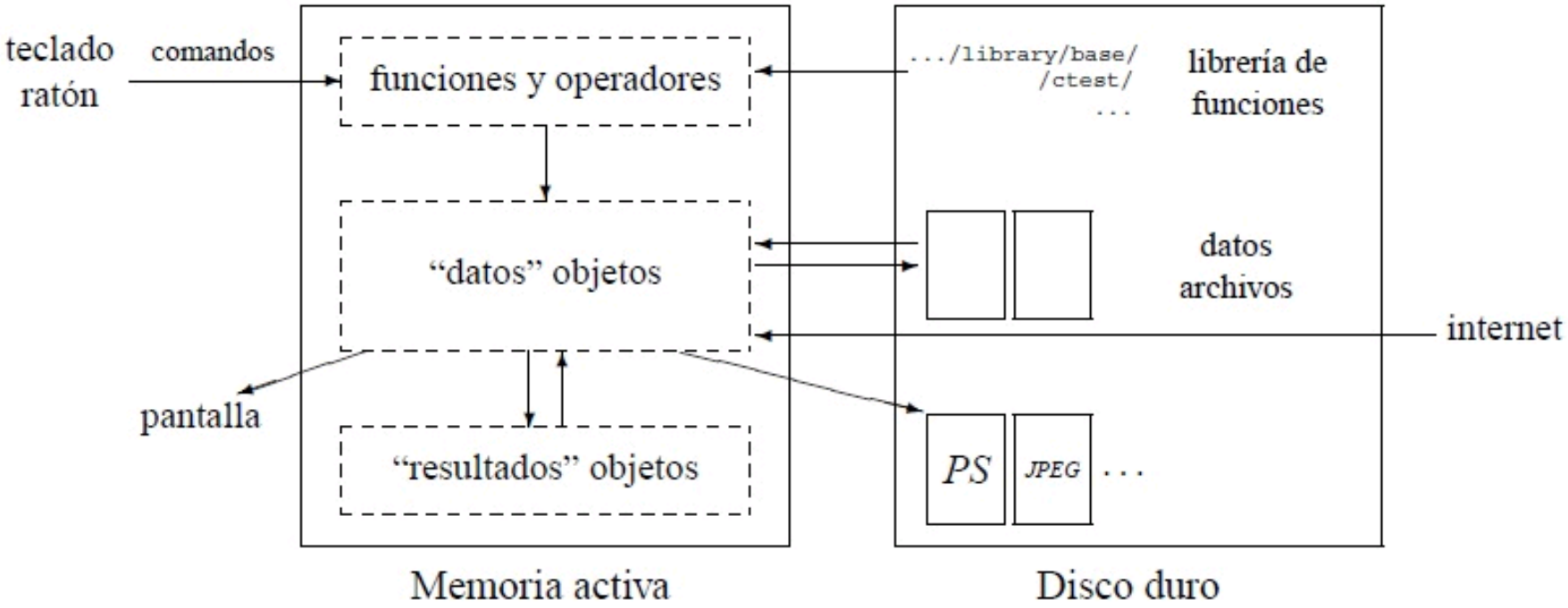


Gráfico. 1 : Funcionamiento de R

R como una calculadora

```
3*5
```

```
## [1] 15
```

```
12.3/2.6
```

```
## [1] 4.7
```

```
sqrt(16)
```

```
## [1] 4
```

R como una calculadora

```
3*5
```

```
## [1] 15
```

```
12.3/2.6
```

```
## [1] 4.7
```

```
sqrt(16)
```

```
## [1] 4
```

Además, cada operación anterior puede ser almacenada en un "objeto"

```
a<-3*5
b<-12.3/2.6
D<-sqrt(16)
```

Instalación de paquetes

- Además de las funciones básicas, *R* tiene un gran número de paquetes especializados.
- Los paquetes se instalan una sola vez y deben ser cargados en cada inicio de sesión.

Se utiliza la siguiente función:

```
install.packages("pckgname",dependencies = TRUE)
```

Una vez instalado, debemos cargarlo con el comando:

```
library(pckgname)
require(pckgname)
```

Estructuras de Datos

Las estructuras de datos en R se organizan por:

- Dimensionalidad; y
- Homogeneidad o heterogeneidad.

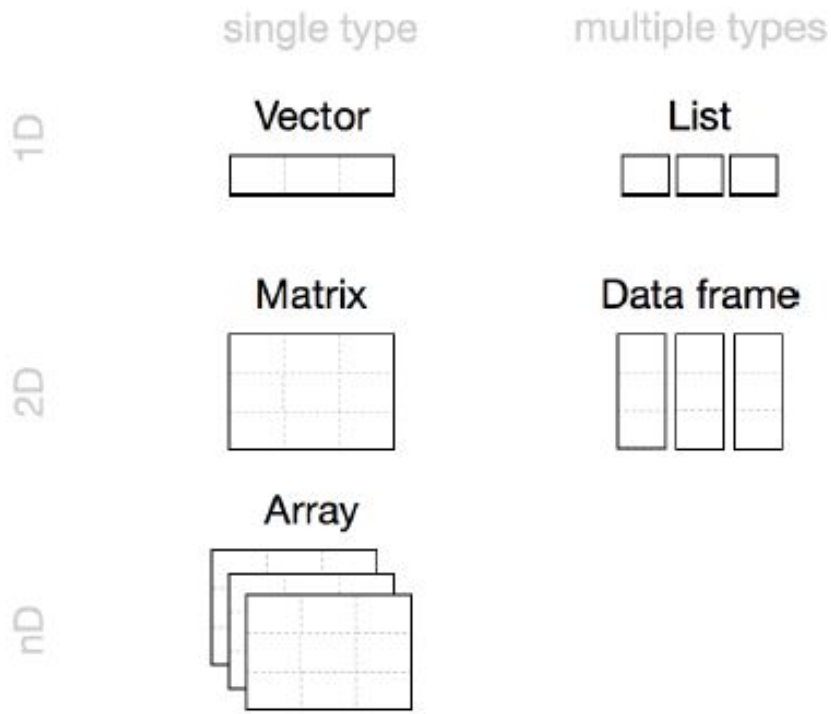


Gráfico. 2 : Estructuras de datos en R

Vectores

La estructura de datos básica de R son los vectores, estos se dividen en:

- Vectores atómicos; y
- Listas.

Propiedades:

Tipo: ¿Qué es?

```
typeof(x)
```

Longitud: Número de elementos.

```
length(x)
```

Vectores Atómicos

Los elementos de un vector atómico son del mismo tipo, a diferencia de los elementos de una lista que pueden ser de diferente tipo. Los tipos comunes son:

- double (numeric);
- integer;
- character;
- logical.

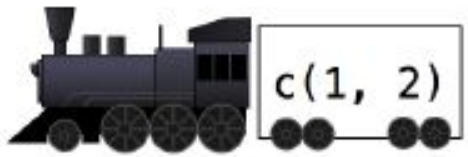


Gráfico. 3 : Función combinar

Un vector es creado mediante la función **c( )** (combinar).

```
vec <- c(1, 2)
vec

## [1] 1 2
```

Tipos de Vectores Atómicos

- Vector double:

```
dbl_vec <- c(3.5, 2, -1)
```

- Vector entero: Use el sufijo L para crear un vector entero;

```
int_vec <- c(3L, 7L, 1L)
```

- Vector caracter: Use " " para crear un vector caracter;

```
chr_vec <- c("R", "Users", "Group")
```

- Vector lógico: Use TRUE y FALSE o T y F para crear un vector lógico.

```
log_vec <- c(FALSE, TRUE, F, T)
```

Tipos de Vectores Atómicos

```
vec <- c(3.5, 2, -1)
is.atomic(vec)

## [1] TRUE
```

Para determinar el tipo de un vector vec utilizamos typeof(vec).

```
vec <- c("R", "Users", "Group")
typeof(vec)

## [1] "character"
```

Para verificar si un vector vec es de un tipo en específico, se utilizan las funciones "is":

```
is.character(vec)
is.double(vec)
is.integer(vec)
is.logical(vec)
```

Elementos de un vector atómico

La componente i de vec se obtiene mediante vec[i].

Componente 5 de vec:

```
vec <- c(6, 1, 3, 6, 10, 5)
vec[5]

## [1] 10
```

Para seleccionar varios elementos utilizamos vec[c(elementos)].

```
# elementos 2 y 4
vec[c(2, 4)]

## [1] 1 6
```

Para omitir el elemento i de vec se utiliza vec[-i].

```
vec[-5]
```

Generación de secuencias

El operador a:b genera el vector a, a+1, a+2, ..., b.

```
1:10
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

```
5:-5
```

```
##  [1]  5  4  3  2  1  0 -1 -2 -3 -4 -5
```

La función seq( ) genera secuencias controlando: inicio, fin y salto.

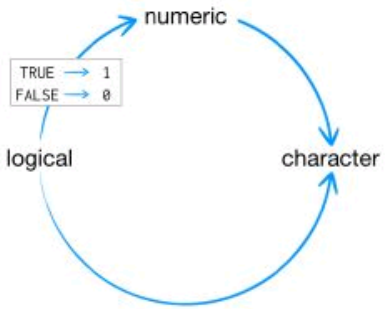
```
seq(from = 1, to = 10, by = 0.5)
```

```
##  [1]  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0
## [16]  8.5  9.0  9.5 10.0
```

SOCIEDAD ECUATORIANA ESTADÍSTICA

Coerción

Si combinamos tipos diferentes, serán coercionados al tipo más flexible dado por la jerarquía:



$$\text{character} \leq \text{double} \leq \text{integer} \leq \text{logical}$$

Gráfico. 4 : Coerción de vectores

Para coercionar un vector x a un determinado tipo, se utilizan las funciones "as"

```
as.character(x)
as.double(x)
as.integer(x)
as.logical(x)
as.numeric(x)
```

Listas

Una lista es un vector que puede contener elementos de cualquier tipo y de distinta longitud.

Gráfico. 5 : Lista en R

Para crear una lista se utiliza la función list() en lugar de c().

```
lst <- list(c(1, 2), c(TRUE), c("a", "b", "c"))
lst

## [[1]]
## [1] 1 2
##
## [[2]]
## [1] TRUE
##
## [[3]]
## [1] "a" "b" "c"
```

Matrices

Una matriz es un vector con el atributo dimensión dim. El atributo dim es un vector de longitud 2: c(nrow, ncol).

```
mtx <- matrix (1:12,nrow=3, ncol=4, byrow=FALSE)
# se construye por columnas por default (byrow=FALSE)
mtx
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    4    7   10
## [2,]    2    5    8   11
## [3,]    3    6    9   12
```

```
attributes(mtx)
```

```
## $dim
## [1] 3 4
```

Elementos de una matriz:

```
(mtx <- matrix (1:12, nrow=3, ncol=4, byrow=FALSE))
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    4    7   10
## [2,]    2    5    8   11
## [3,]    3    6    9   12
```

```
mtx[1,2] # componente 1, 2
```

```
## [1] 4
```

```
mtx[,3]  # columna 3
```

```
## [1] 7 8 9
```

```
mtx[1,]  # fila 1
```

```
## [1]  1  4  7 10
```

Elementos de una matriz:

```
(mtx <- matrix (1:12, nrow=3, ncol=4, byrow=FALSE))
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    4    7   10
## [2,]    2    5    8   11
## [3,]    3    6    9   12
```

```
mtx[,c(2,4)] # columnas 2 y 4
```

```
##      [,1] [,2]
## [1,]    4   10
## [2,]    5   11
## [3,]    6   12
```

```
mtx[c(1,3),]  # filas 1 y 3
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    4    7   10
## [2,]    3    6    9   12
```

Factores

Es la estructura de datos utilizada para almacenar variables categóricas. ¿Por qué utilizar factores?

- Un vector c(Femenino, Masculino) (factor) presenta mayor información que un vector c(1, 2).
- Un vector character c("Femenino", "Masculino") no pueden ser incluido en modelos de regresión, un factor c(Femenino, Masculino) si.

⇒

Género	Etiqueta
1	Femenino
2	Masculino
1	Femenino
2	Masculino

Gráfico. 6 : Ejemplo de factor en R

Factores

Si se dispone de un vector integer:

```
vec <- c(1, 2, 2, 1, 2, 1, 2)
vec

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

La función factor asigna labels a los levels (o categorías) de la variable. Los levels del vector vec son los valores 1, 2.

```
# Creación de un factor
fac <- factor(vec, levels=c(1,2), labels = c("Femenino",
                                             "Masculino"))
fac

## [1] Femenino  Masculino Masculino Femenino  Masculino Femenin
## Levels: Femenino Masculino
```

Factores

Para realizar conteos por categoría, se utiliza la función table().

```
fac <- factor(vec, levels=c(1,2), labels = c("Femenino",
                                             "Masculino"))
# frecuencias
table(fac)

## fac
##  Femenino Masculino
##         3         4

# porcentaje
prop.table(table(fac))

## fac
##  Femenino Masculino
##      0.43      0.57
```

Data Frame

Es una lista en la cual todos los elementos tienen la misma longitud. A diferencia de las matrices, pueden almacenar vectores atómicos de cualquier tipo. Presenta varios atributos adicionales class, rownames, names. Es la estructura de datos más utilizada para almacenar data tabulada.

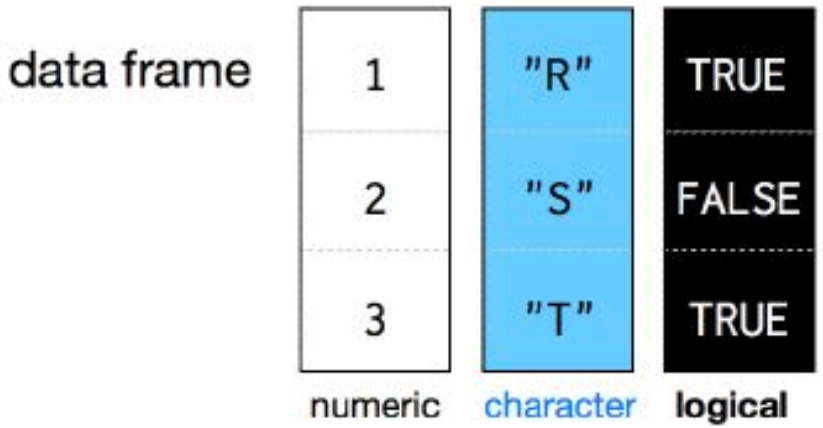


Gráfico. 7 : Data Frame en R

Data Frame

Para crear un data frame se utiliza la función data.frame(). Con los siguientes vectores atómicos:

```
dbl_vec <- c(1, 2, 3)
chr_vec <- c("R", "S", "T")
log_vec <- c(TRUE, FALSE, TRUE)
```

Creamos el data frame df:

```
df <- data.frame(dbl_vec, chr_vec, log_vec)
df
```

```
##   dbl_vec chr_vec log_vec
## 1       1       R    TRUE
## 2       2       S   FALSE
## 3       3       T    TRUE
```

Un data frame es una lista:

```
typeof(df) # Tipo de un data frame
```

Elementos de un data frame:

Mediante df[i, j] se obtiene la componente i, j del data frame.

John	1940	guitar
Paul	1941	bass
George	1943	guitar
Ringo	1940	drums

```
df[2, c(2,3)]
```

Gráfico. 8 : Extracción de elementos de un Data Frame

Elementos de un data frame:

```
nomb <- c("John", "Paul", "George", "Ringo")
nac <- c(1940, 1941, 1943, 1940)
instr <- c("guitar", "bass", "guitar", "drums")

df <- data.frame(nomb, nac, instr)
df[2, c(2,3)]

##    nac instr
## 2 1941  bass

print(df)

##     nomb  nac  instr
## 1   John 1940 guitar
## 2   Paul 1941   bass
## 3 George 1943 guitar
## 4  Ringo 1940  drums
```

Elementos de un data frame:

```
df[2, 2] # componente 2, 2

## [1] 1941

df[3, 1] # componente 3, 1

## [1] "George"

df[3, ] # fila 3

##     nomb  nac  instr
## 3 George 1943 guitar

df[c(1, 4), ] # filas 1, 4

##    nomb  nac  instr
## 1  John 1940 guitar
## 4 Ringo 1940  drums
```

Elementos de un data frame:

Importante: Filtrado o subsetting:

```
df[ , 3]=="guitar" # columna 3 de df igual a "guitar"

## [1]  TRUE FALSE  TRUE FALSE

f_guitar <- df[ , 3]=="guitar"
```

Filas donde la columna 3 es igual a "guitar"

```
df[f_guitar, ]

##      nomb  nac  instr
## 1    John 1940 guitar
## 3 George 1943 guitar
```

Elementos de un data frame:

Columna de nombre "nac"

```
df[ , "nac"] # equivalente a df[ , 2]

## [1] 1940 1941 1943 1940
```

Columnas de nombres "nomb y nac"

```
df[ , c("nomb", "nac")] # equivalente a df[ , c(1, 2)]

##     nomb  nac
## 1   John 1940
## 2   Paul 1941
## 3 George 1943
## 4  Ringo 1940
```

Lectura de datos

R puede acceder a información almacenada en distintos formatos:

- Archivos de excel .xls, .xlsx, .csv;
- Archivos de texto plano .txt;
- Archivos de spss .sav;
- Archivos de la web;
- Archivos de bases de datos, etc.

Gráfico. 9 : Varios formatos que lee el R

Directorio de trabajo

Working directory (wd). Es la dirección donde se almacenan, leen y escriben los archivos utilizados y generados mediante R. getwd() permite obtener el wd actual:

```
getwd()

## [1] "C:/Users/usuario/Desktop/RUGE/TIIE_RUGE_2018/TIIE_RUGE_u
```

setwd() permite setear un nuevo wd, por ejemplo:

```
setwd("C:/Users/Andres/Desktop/TIIE_RUGE_2018")
```

dir() enlista los nombres de los archivos en el wd actual.

```
dir()
```

¿Cómo leer archivos en R?

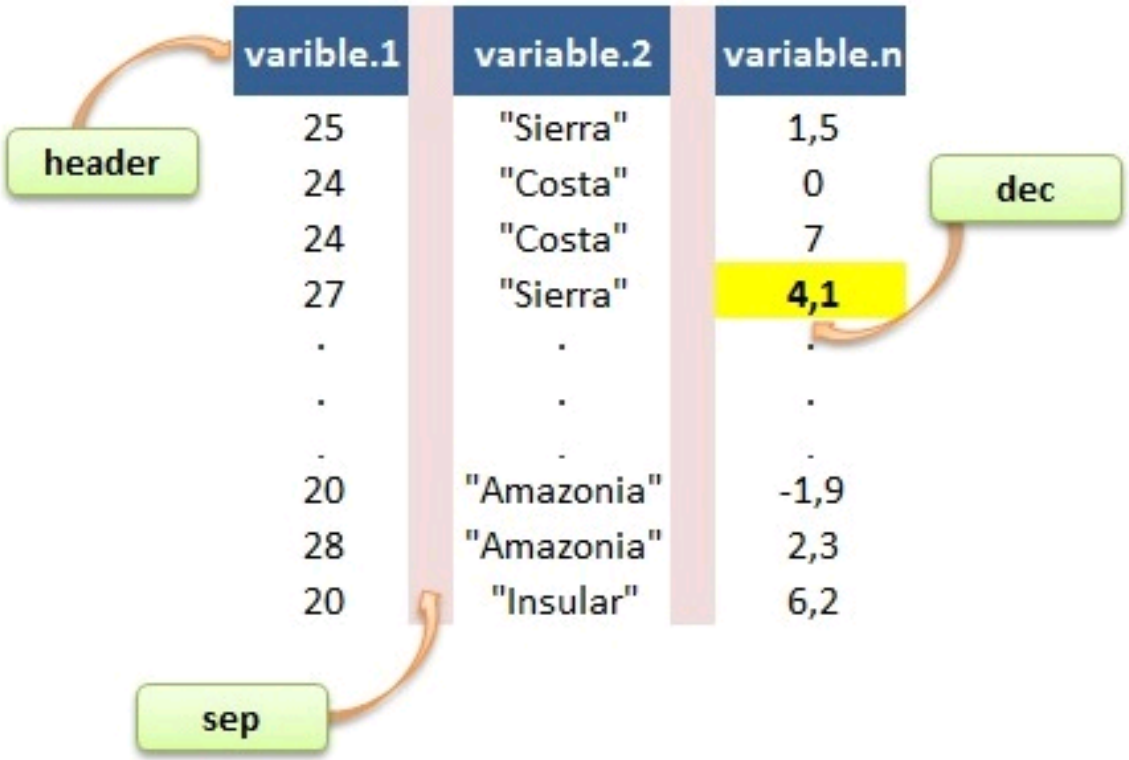


Gráfico. 10 : Cómo lee los archivos el R

Lectura de archivos .txt

Se requiere leer un archivo en formato txt: archivo.txt. Se utiliza la función read.table()

```
data_txt <- read.table(file = "archivo.txt",sep = "\t",
                       dec = ",", header = TRUE)
str(data_txt)
```

Parámetros:

- file: nombre del archivo (incluida extensión);
- sep: caracter utilizado para separar columnas (variables);
- dec: caracter utilizado para decimales;
- header: TRUE, si la primera fila contiene los nombres de las columnas.

str() describe la estructura de datos.

Lectura de archivos .csv

Se requiere leer un archivo en formato csv: archivo.csv, se utiliza las función read.csv que presenta por defecto los argumentos: sep = ",", dec = ".", , header = TRUE

```
data_csv <- read.csv(file = "archivo.csv")
```

read.csv2(): Se debe especificar los argumentos: sep, dec, header

```
data_csv2 <- read.csv2(file = "archivo.csv",sep = ",",
                       dec = ".", header = TRUE)
```

Lectura de archivos .xls, .xlsx

Se requiere leer un archivo en formato xls o xlsx: archivo.xlsx. Se utiliza la función read_excel() del paquete readxl

```
install.packages("readxl", dependencies = TRUE)
library(readxl)
ls("package:readxl")
```

Lectura del archivo archivo.xlsx:

```
data_xlsx <- read_excel("archivo.xlsx",sheet = "datos",
                        col_names = TRUE, na="")
read.table("clipboard",sep = "\t",header = FALSE)
```

Parámetros:

- sheet: Nombre de la hoja que contiene la data (recibe también el número de hoja);
- col_names: TRUE si la primera fila contiene los nombres de las columnas;
- na: los caracter que se coerciona a NA.

Lectura de archivos .sav

Se requiere leer archivos desde spss, es decir en formato .sav: archivo.sav. Se utiliza la función read.spss() del paquete foreign.

```
install.packages("foreign", dependencies = TRUE)
library(foreign)
ls("package:foreign")

data_sav <- read.spss(file="archivo.sav",
                      use.value.labels = TRUE,
                      to.data.frame = TRUE)
```

Parámetros:

- file: nombre del archivo (incluida extensión);
- use.value.labels: TRUE, si se consideran las etiquetas de las variables;
- to.data.frame: TRUE, para coercionar el archivo leído a data frame.

2. Paquete dplyr

El paquete dplyr y su gramática

- El paquete **dplyr** fue desarrollado por Hadley Wickham de RStudio.
- Una importante contribución del paquete dplyr es que proporciona una "gramática" (particularmente verbos) para la manipulación y operaciones con data frames.
- Algunas de los principales "verbos" del paquete
- El paquete **dplyr** son:
  - **select**: devuelve un conjunto de columnas
  - **filter**: devuelve filas según condiciones lógicas
  - **arrange**: reordena filas de un data frame
  - **mutate**: añade nuevas variables/columnas o transforma variables existentes
  - **summarise/summarize**: genera resúmenes estadísticos
  - **%>%**: el operador "pipe" es usado para conectar múltiples acciones en una única "pipeline" (tubería)

Gracias!!!