

intro\_r 

Pau

30/08/2018

# ¿Qué es R?

**R es un lenguaje de programación con enfoque al análisis estadístico.**

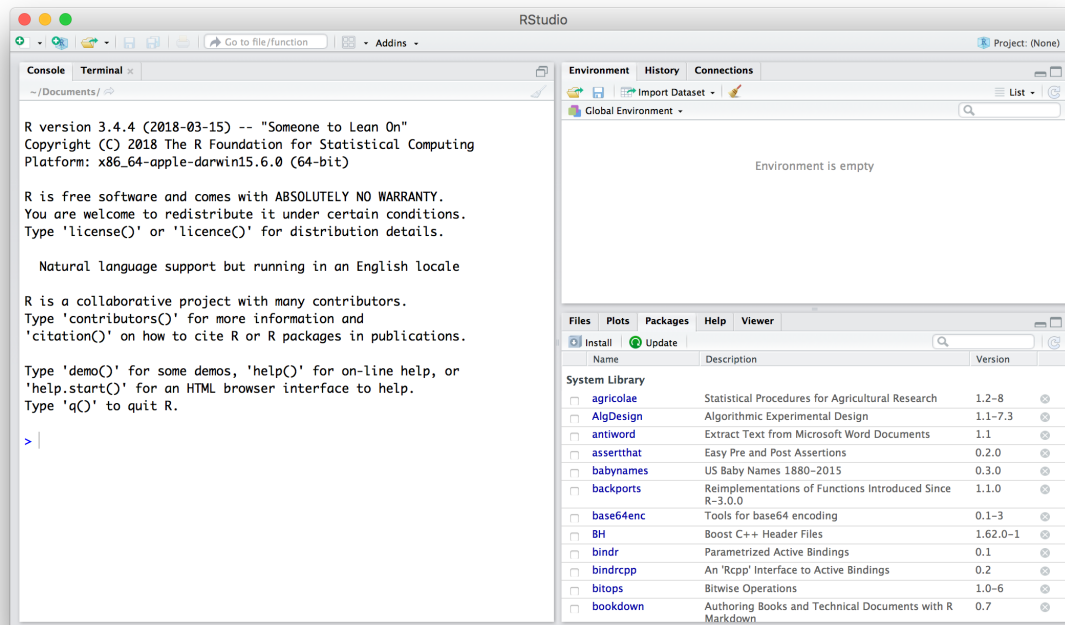
Además:

- Software libre (no dice qué podés o no hacer con el software)
- De código abierto (todo el código de R se inspecciona)
- Funcionalidad adicional está en paquetes que la comunidad contribuye

# ¿Qué es un IDE?

- acrónimo de Integrated Development Environment (Entorno de Desarrollo Integrado) → RStudio es una aplicación que nos entrega herramientas para hacer más fácil el desarrollo de proyectos usando R.

Se ve más o menos así:



# Instalación R

Para poder instalar R y RStudio, se siguen los siguientes pasos:

- Descargan R desde <https://cran.r-project.org/>. Eligen la opción que corresponda, según su sistema operativo.
- Instalan R en su PC, como cualquier programa.
- Una vez que R ha quedado correctamente instalado, descarguen RStudio desde la opción, es decir, "RStudio Desktop Open Source License" (gratis).
- Instalan RStudio en su PC, como cualquier programa.

# Utilización de R

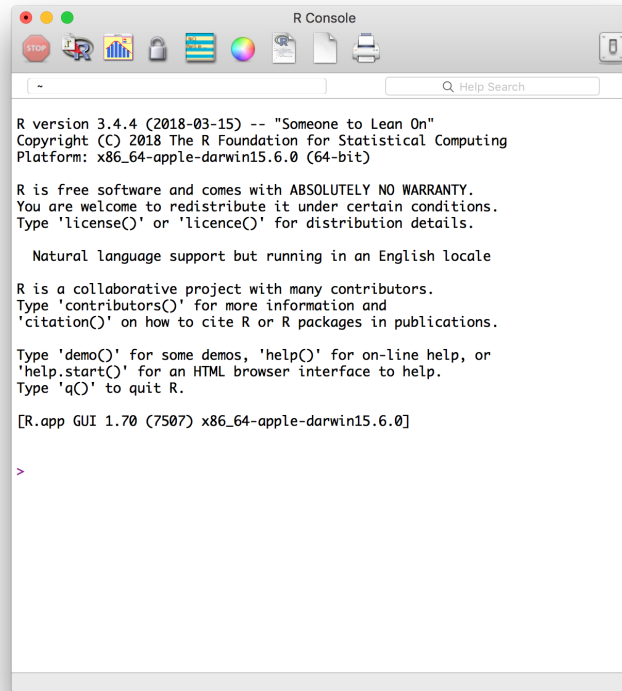
The image displays the RStudio environment with four main panels:

- Código (Code Editor):** Contains the following R code:

```
1 library(ggplot2)
2 ggplot(mpg, aes(x = displ, y = hwy)) +
3   geom_point(aes(color = class))
4
```
- Consola (Console):** Shows the execution of the code from the editor:

```
> library(ggplot2)
> ggplot(mpg, aes(x = displ, y = hwy)) +
+   geom_point(aes(color = class))
>
```
- Workspace:** Labeled "Environment is empty", indicating no objects are currently loaded in the workspace.
- Visualizador (Plots):** Displays a scatter plot of highway mileage (hwy) versus engine displacement (displ). The points are colored according to the vehicle class. The legend on the right lists the classes: 2seater, compact, midsize, minivan, pickup, subcompact, and suv.

Si solo abris R en la PC, lo que verás es una consola. Cuando trabajamos en ella escribimos directamente el código que queremos que se ejecute. El signo `>` nos indica que R está listo y esperando que escribamos algo.



```
R Console

R version 3.4.4 (2018-03-15) -- "Someone to Lean On"
Copyright (C) 2018 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin15.6.0 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

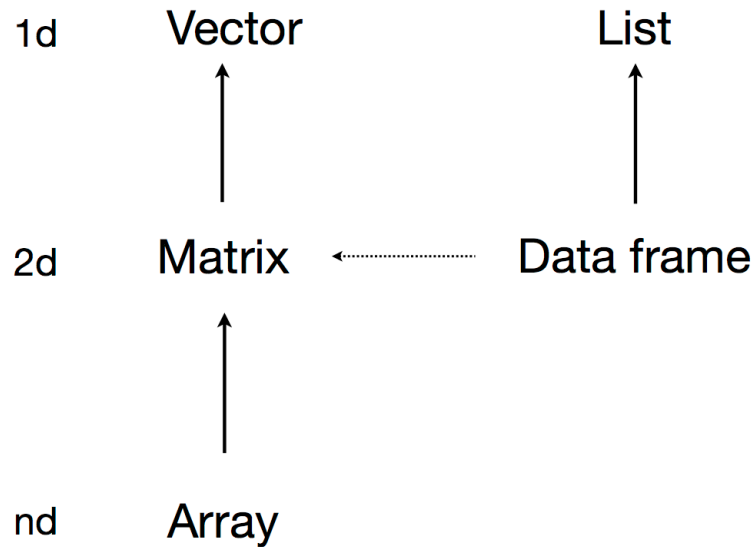
[R.app GUI 1.70 (7507) x86_64-apple-darwin15.6.0]

>
```

# Tipos de Datos

En R existen cinco tipos de datos básicos:

1) Vector, 2) Matriz, 3) Factor, 4) Data frame, 5) Lista



Same types

Different types

# Vectores

Un vector es un arreglo de una dimensión.

En R existen tres clases principales de vectores y se crean con la función combine `c()`:

- Numérico

```
num_vec <- c(-1, 2.5, 3, 4, 5.1)
```

- Character

```
cha_vec <- c("Mon", "Tue", "Wed", "Thu", "Sat", "Sun")
```

- Lógico

```
boo_vec <- c(TRUE, FALSE, FALSE, TRUE, TRUE, FALSE)
```



La función `class()` nos dice cuál es la clase o tipo del vector.

```
class(num_vec)
```

```
## [1] "numeric"
```

Otra función importante es `length()` que nos dice cuál es la longitud del vector.

```
length(num_vec)
```

```
## [1] 5
```

# Ejemplo: Ganancias - Ruleta y poker

Mis ganancias de poker por día de la semana son:

```
poker_gan <- c(150, 178, -6, 166, -80, -119, -142)
poker_gan
```

```
## [1] 150 178 -6 166 -80 -119 -142
```

# Nombres de vectores

La función `names()` nos permite nombrar los elementos de cada vector.

Por ejemplo, a cada elemento de las ganancias de poker del ejercicio anterior, asignaremos el nombre del día de la semana en que se obtuvieron.

```
dias <- c("Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun")
names(poker_gan) <- dias
poker_gan
```

```
## Mon Tue Wed Thu Fri Sat Sun
## 150 178 -6 166 -80 -119 -142
```

# Selección de elementos en vectores

La selección de elementos de un vector se realiza indicando las posiciones a seleccionar entre [ ].

Estas posiciones pueden indicarse por medio de un vector numérico o de caracteres si los elementos del vector están nombrados.

- Vector numérico:

```
poker_gan[ c(1, 5) ]
```

```
## Mon Fri  
## 150 -80
```

El uso de dos puntos: permiten crear un vector de secuencias numéricas:

```
poker_gan[ 1:3 ]
```

```
## Mon Tue Wed  
## 150 178 -6
```

- Nombres:

```
poker_gan[ c("Mon", "Tue")]
```

```
## Mon Tue
```

```
## 150 178
```

# Operaciones con vectores

```
ruleta_gan <- c(-48, 151, 198, -16, 134, -153, 126)
dias <- c("Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun")
names(ruleta_gan) <- dias

poker_gan + ruleta_gan
```

```
## Mon Tue Wed Thu Fri Sat Sun
## 102 329 192 150 54 -272 -16
```

En R para cualquier operación (+, -, \*, /) de vectores, las operaciones son elemento a elemento (element wise).

Por ejemplo, al sumar vectores:

la primera posición del primer vector se suma con la primera posición del segundo vector,

la segunda posición del primer vector se suma con la segunda posición del segundo vector

y así sucesivamente.

# Comparación de elementos

La comparación de elementos se realiza con los siguientes comandos:

- `>` mayor a
- `>=` mayor o igual
- `<` menor a
- `<=` menor o igual a
- `==` igual a
- `!=` distinto de
- `%in%` contenido en Este tipo de operaciones regresan un vector lógico dependiendo si la condición se cumple o no.

```
poker_gan
```

```
## Mon Tue Wed Thu Fri Sat Sun  
## 150 178 -6 166 -80 -119 -142
```

```
poker_pos <- poker_gan >= 0  
poker_pos
```

```
## Mon Tue Wed Thu Fri Sat Sun  
## TRUE TRUE FALSE TRUE FALSE FALSE FALSE
```

# Matrices

Una matriz es un arreglo de dos dimensiones en el que todos los elementos son del mismo tipo, por ejemplo: numéricos

La función `matrix()` permite crear la matriz de un vector especificando las dimensiones, por ejemplo:

```
matrix(data = 1:9, nrow = 3, ncol = 3)
```

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

En el siguiente vector se presentan los ingresos totales y de lanzamiento de cada película de la saga Harry Potter.



# Ejemplo: Box Office Mojo: Harry Potter

```
sales_hp <- c(497066400, 426630300, 401608200, 399302200, 377314200,  
             359788300, 357233500, 328833900, 141823200, 189432500,  
             142414700, 135197600, 99635700, 92756000, 134119300,  
             138752100)  
sales_mat <- matrix(sales_hp, nrow = 8)  
sales_mat
```

```
##           [,1]      [,2]  
## [1,] 497066400 141823200  
## [2,] 426630300 189432500  
## [3,] 401608200 142414700  
## [4,] 399302200 135197600  
## [5,] 377314200  99635700  
## [6,] 359788300  92756000  
## [7,] 357233500 134119300  
## [8,] 328833900 138752100
```

La función `dim()` regresa la dimensión de la matriz (renglones y columnas).

```
dim(sales_mat)
```

```
## [1] 8 2
```

La función `nrow()` regresa el número de renglones de la matriz y `ncol()` el número de columnas.

```
nrow(sales_mat)
```

```
## [1] 8
```

```
ncol(sales_mat)
```

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

# Nombres de matrices

En R es posible agregar nombres a los renglones y columnas de una matriz con las funciones `colnames()` y `rownames()`. Considerando los siete títulos de la saga, asignamos los títulos de las películas a los renglones con la función `rownames()`:

```
titles_hp <- c(
  "1. HP and the Sorcerer's Stone",
  "8. HP and the Deathly Hallows Part 2",
  "4. HP and the Goblet of Fire",
  "2. HP and the Chamber of Secrets",
  "5. HP and the Order of the Phoenix",
  "6. HP and the Half-Blood Prince",
  "3. HP and the Prisoner of Azkaban",
  "7. HP and the Deathly Hallows Part 1")
rownames(sales_mat) <- titles_hp
sales_hp <- c("total", "release_date")
colnames(sales_mat) <- sales_hp
sales_mat
```

|   | total     | release_date |
|---|-----------|--------------|
| ## 1. HP and the Sorcerer's Stone       | 497066400 | 141823200    |
| ## 8. HP and the Deathly Hallows Part 2 | 426630300 | 189432500    |
| ## 4. HP and the Goblet of Fire         | 401608200 | 142414700    |
| ## 2. HP and the Chamber of Secrets     | 399302200 | 135197600    |
| ## 5. HP and the Order of the Phoenix   | 377314200 | 99635700     |
| ## 6. HP and the Half-Blood Prince      | 359788300 | 92756000     |
| ## 3. HP and the Prisoner of Azkaban    | 357233500 | 134119300    |

# Selección de elementos en una matriz

Al igual que un vector, los elementos de una matriz pueden seleccionarse con un vector de posiciones o un vector de nombres. Pero, en este se define la posición de ambas dimensiones, renglones y columnas [ , ].

Por ejemplo, si queremos obtener una submatriz para las primeras tres películas de las ventas:

```
sales_mat[c(1, 4, 7), 1:2]
```

```
##                total release_date
## 1. HP and the Sorcerer's Stone 497066400 141823200
## 2. HP and the Chamber of Secrets 399302200 135197600
## 3. HP and the Prisoner of Azkaban 357233500 134119300
```

# Operaciones en matrices

Al igual que los vectores, las operaciones son elemento a elemento o element wise.

Siguiendo con el ejemplo de ingresos, para facilitar la lectura de los datos dividimos entre un millón cada valor.

```
sales_mat_mill <- sales_mat/1e6  
sales_mat_mill
```

```
##               total release_date  
## 1. HP and the Sorcerer's Stone    497.0664    141.8232  
## 8. HP and the Deathly Hallows Part 2 426.6303    189.4325  
## 4. HP and the Goblet of Fire      401.6082    142.4147  
## 2. HP and the Chamber of Secrets  399.3022    135.1976  
## 5. HP and the Order of the Phoenix 377.3142     99.6357  
## 6. HP and the Half-Blood Prince   359.7883     92.7560  
## 3. HP and the Prisoner of Azkaban  357.2335    134.1193  
## 7. HP and the Deathly Hallows Part 1 328.8339    138.7521
```

Lo mismo sucede con un vector. Supongamos que el siguiente vector contiene el número de cines en los que se exhibió cada película.

```
theaters_vec <- c(3672, 4375, 3858, 3682, 4285, 4325, 3855, 4125)  
theaters_vec
```

# Variable categórica nominal

Un ejemplo de variable categórica nominal es el sexo de una persona: femenino (F) o masculino (M)

En R un factor se define con la función `factor()`.

```
sex_vec <- c("F", "M", "M", "F", "M")
sex_fct <- factor(sex_vec)
sex_fct
```

```
## [1] F M M F M
## Levels: F M
```

En automático define los niveles del factor y los ordena en orden alfabético. Si se desea cambiar esto el argumento `levels = c()` permite asignar un vector de niveles específico.

```
sex_lev_fct <- factor(sex_vec, levels = c("M", "F"))
sex_lev_fct
```

```
## [1] F M M F M
## Levels: M F
```

# Variable categórica ordinal

Una variable categórica ordinal como el nombre lo dice tiene orden en los niveles del factor.

Para dar orden a los niveles en R se modifica el argumento `ordered = TRUE` de la función `factor()`.

Se tiene el siguiente vector de temperaturas y se desea crear un factor ordenado de menor temperatura a mayor temperatura.

```
temp_vec <- c("High", "Low", "Medium", "Low",  
             "Low", "Medium", "High", "Low",  
             "Medium", "Low", "Low")  
temp_fct <- factor(temp_vec,  
                  levels = c("Low", "Medium", "High"),  
                  ordered = T)  
temp_fct
```

```
## [1] High Low Medium Low Low Medium High Low Medium Low  
## [11] Low  
## Levels: Low < Medium < High
```

Ahora los niveles tiene una jerarquía.

Una forma de modificar las etiquetas de los niveles es reasignando un vector.

# Dataframe

- Un dataframe es un objeto de dos dimensiones en R. Puede verse como un arreglo de vectores de la misma dimensión, similar a una matriz.
- La ventaja de un dataframe, es que a diferencia de una matriz, los vectores o columnas pueden ser de diferentes tipos.

Puedo crear dataframes con la función `data.frame()`.



Una forma de crear un dataframe es asignando vectores.

```
muestra_df <- data.frame(secuencia = 1:5,  
                          aleatorio = rnorm(5),  
                          letras = c("a", "b", "c", "d", "e"))  
muestra_df
```

```
##   secuencia aleatorio letras  
## 1         1 -1.2205469      a  
## 2         2  1.4234340      b  
## 3         3 -1.1243452      c  
## 4         4 -0.4649135      d  
## 5         5 -2.3252691      e
```

O bien, se puede transformar una matriz con la misma función. Tomemos los datos de los ingresos de las películas de la saga de HP y hagamos una matriz.

```
sales_df <- data.frame(sales_mat)
sales_df
```

```
##               total release_date
## 1. HP and the Sorcerer's Stone    497066400    141823200
## 8. HP and the Deathly Hallows Part 2 426630300    189432500
## 4. HP and the Goblet of Fire      401608200    142414700
## 2. HP and the Chamber of Secrets  399302200    135197600
## 5. HP and the Order of the Phoenix 377314200     99635700
## 6. HP and the Half-Blood Prince   359788300     92756000
## 3. HP and the Prisoner of Azkaban  357233500    134119300
## 7. HP and the Deathly Hallows Part 1 328833900    138752100
```

# Nombres de dimensiones

Al igual que matrices, las funciones `rownames()` y `colnames()` permiten nombrar los renglones y columnas del objeto.

```
colnames(sales_df) <- c("total_grosses", "opening_grosses")
sales_df
```

|   | total_grosses | opening_grosses |
|---|---------------|-----------------|
| ##                                      |               |                 |
| ## 1. HP and the Sorcerer's Stone       | 497066400     | 141823200       |
| ## 8. HP and the Deathly Hallows Part 2 | 426630300     | 189432500       |
| ## 4. HP and the Goblet of Fire         | 401608200     | 142414700       |
| ## 2. HP and the Chamber of Secrets     | 399302200     | 135197600       |
| ## 5. HP and the Order of the Phoenix   | 377314200     | 99635700        |
| ## 6. HP and the Half-Blood Prince      | 359788300     | 92756000        |
| ## 3. HP and the Prisoner of Azkaban    | 357233500     | 134119300       |
| ## 7. HP and the Deathly Hallows Part 1 | 328833900     | 138752100       |

# Selección de elementos

Para dataframes, además de seleccionar posiciones de renglones y columnas con [ , ], se puede usar el signo \$.

```
sales_df$total_grosses
```

```
## [1] 497066400 426630300 401608200 399302200 377314200 359788300 357233500  
## [8] 328833900
```

# Funciones útiles para data frames

Existen algunas que ayudan a tratar dataframes.

`head()`:

```
head(sales_df)
```

```
##                                total_grosses opening_grosses
## 1. HP and the Sorcerer's Stone      497066400      141823200
## 8. HP and the Deathly Hallows Part 2 426630300      189432500
## 4. HP and the Goblet of Fire        401608200      142414700
## 2. HP and the Chamber of Secrets    399302200      135197600
## 5. HP and the Order of the Phoenix  377314200       99635700
## 6. HP and the Half-Blood Prince     359788300       92756000
```

`str()`

```
str(sales_df)
```

```
## 'data.frame':    8 obs. of  2 variables:
## $ total_grosses : num  4.97e+08 4.27e+08 4.02e+08 3.99e+08 3.77e+08 ...
## $ opening_grosses: num  1.42e+08 1.89e+08 1.42e+08 1.35e+08 9.96e+07 ...
```

dim(), nrow() y ncol()

```
dim(sales_df)
```

```
## [1] 8 2
```

```
nrow(sales_df)
```

```
## [1] 8
```

```
ncol(sales_df)
```

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

# Ejemplo:

```
datos <- read.csv('data/datos.csv')
```

```
str(datos)
```

```
## 'data.frame':    1704 obs. of  6 variables:
## $ pais          : Factor w/ 142 levels "Afghanistan",...: 1 1 1 1 1 1 1 1 1 1 ...
## $ continente    : Factor w/ 5 levels "Africa","Americas",...: 3 3 3 3 3 3 3 3 3 3 ...
## $ anioo         : int  1952 1957 1962 1967 1972 1977 1982 1987 1992 1997 ...
## $ esperanza_vida: num  28.8 30.3 32 34 36.1 ...
## $ poblacion     : int  8425333 9240934 10267083 11537966 13079460 14880372 12881816 1386...
## $ pbi_per_capita: num  779 821 853 836 740 ...
```

```
head(datos)
```

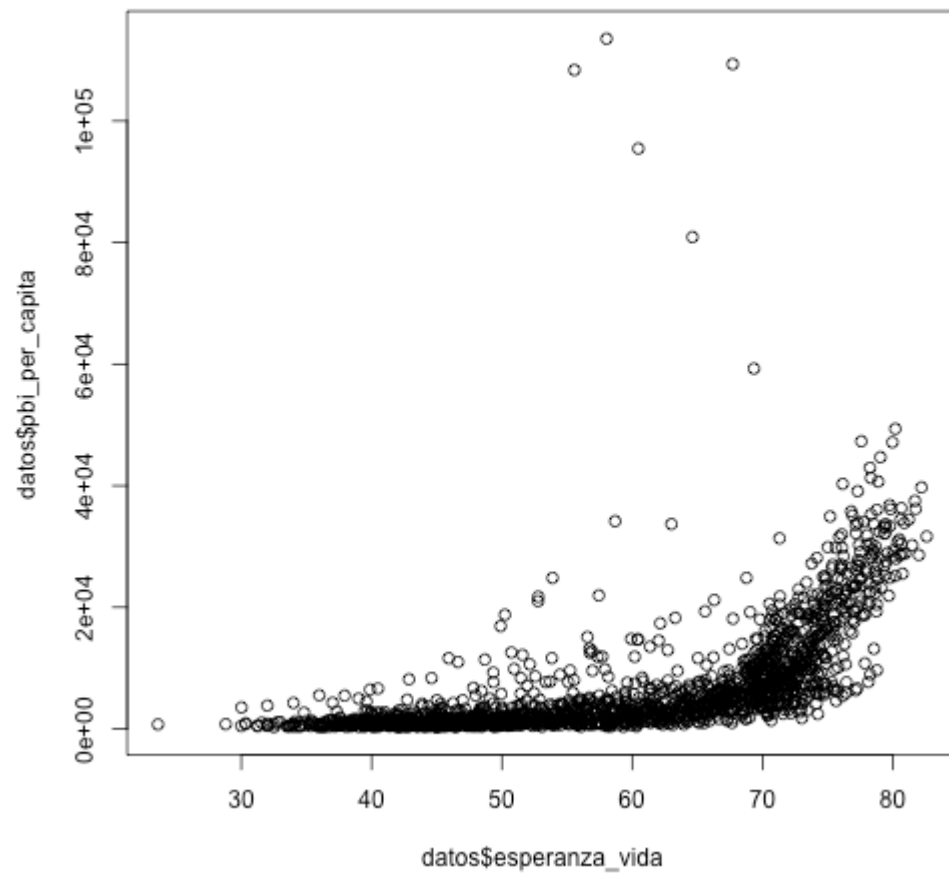
```
##      pais continente anioo esperanza_vida poblacion pbi_per_capita
## 1 Afghanistan      Asia  1952         28.801    8425333      779.4453
## 2 Afghanistan      Asia  1957         30.332    9240934      820.8530
## 3 Afghanistan      Asia  1962         31.997   10267083      853.1007
## 4 Afghanistan      Asia  1967         34.020   11537966      836.1971
## 5 Afghanistan      Asia  1972         36.088   13079460      739.9811
## 6 Afghanistan      Asia  1977         38.438   14880372      786.1134
```



```
table(datos$continente)
```

```
##  
## Africa Americas Asia Europe Oceania  
##      624      300    396    360      24
```

```
plot(datos$esperanza_vida, datos$pbi_per_capita)
```



# Listas

Una lista en R es un objeto que permite una estructura de datos complicada, una súper estructura. Esto porque permite reunir diferentes tipos de objetos:

- Vectores
- Matrices
- Dataframes
- Listas

Es decir, ¡puede almacenar cualquier cosa!

# Crear una lista

La función `list()` permite crear una lista.

```
ejem_list <- list(  
  vector = 1:10,  
  matriz = matrix(1:9, nrow = 3),  
  dataframe = mtcars[1:5,]  
)  
ejem_list
```

```
## $vector  
## [1] 1 2 3 4 5 6 7 8 9 10  
##  
## $matriz  
##      [,1] [,2] [,3]  
## [1,] 1    4    7  
## [2,] 2    5    8  
## [3,] 3    6    9  
##  
## $dataframe  
##           mpg  cyl  disp  hp  drat    wt  qsec  vs  am  gear  carb  
## Mazda RX4      21.0   6  160 110  3.90  2.620 16.46  0  1    4    4  
## Mazda RX4 Wag  21.0   6  160 110  3.90  2.875 17.02  0  1    4    4  
## Datsun 710      22.8   4  108  93  3.85  2.320 18.61  1  1    4    1  
## Hornet 4 Drive  21.4   6  258 110  3.08  3.215 19.44  1  0    3    1  
## Hornet Sportabout 18.7   8  360 175  3.15  3.440 17.02  0  0    3    2
```

# Selección de elementos en una lista

La selección de elementos de una lista puede realizarse de tres maneras:

1) `[ ]`

```
```r
ejem_list[1]

```

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

2) `[[ ]]`

```
```r
ejem_list[[1]]

```

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

3) \$

```
```r  
ejem_list$vector
```

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