RLadies meetup 2. Trabajando con Data frames

Leticia Vega-Alvarado
leticia.vega@icat.unam.mx

28 Feb 2020

Índice

1. Objetivo

Tener una visión general de qué es un *data frame*, algunas de sus características, cómo crearlo, cómo acceder a sus elementos y su utilidad. Así mismo, proporcionar algunas funciones para la lectura de *data frames* desde un archivo, añadir columnas y renglones, entre otras.

2. ¿Qué es un data frame?

Un *data frame*, es una lista de vectores (de la misma longitud), que R despliega como una tabla, es decir, como una estructura rectangular de datos, organizada en renglones y columnas. El *data frame* es muy similar a una matriz, la diferencia es que los renglones de un *data frame* pueden contener valores de diferentes tipos de datos, ya que cada columna puede ser de un tipo diferente.

3. Creación de un *data frame*

Podemos crear un *data frame* con la función `data.frame()`, por ejemplo:

```r
# Creando los vectores de valores cada vector será una columna del data frame
paciente <- c("Alfredo Aguirre","Guadalupe Colin", "Miguel Garza","Alicia Zapata")
edad <- c(50,25,32,76)
tiposangre <- c("O+","A+","O-","O+")
peso <- c(95.2,56.4,67,73.1)

# Creando el data frame
df_pacientes <- data.frame(paciente,tiposangre,peso,edad)
df_pacientes
```

```
##          paciente tiposangre peso edad
## 1 Alfredo Aguirre         O+ 95.2   50
## 2 Guadalupe Colin         A+ 56.4   25
## 3    Miguel Garza         O- 67.0   32
## 4   Alicia Zapata         O+ 73.1   76
```

3.1. Creando un *data frame* desde una lista

Podemos convertir una lista en un *data frame*, utilizando la función **`as.data.frame()`**, por ejemplo, utilicemos los vectores creados anteriormente para generar una lista con ellos.

```r
# Creando la lista y mostrando su contenido
lista_pacientes <- list(paciente=paciente,edad=edad,tiposangre=tiposangre,peso=peso)
lista_pacientes
```

```
## $paciente
## [1] "Alfredo Aguirre" "Guadalupe Colin" "Miguel Garza"    "Alicia Zapata"
##
## $edad
## [1] 50 25 32 76
##
## $tiposangre
## [1] "O+" "A+" "O-" "O+"
##
## $peso
## [1] 95.2 56.4 67.0 73.1
```

```r
# Creando el data frame
df_pacientes2 <- as.data.frame(lista_pacientes)
df_pacientes2
```

```
##          paciente edad tiposangre peso
## 1 Alfredo Aguirre   50         O+ 95.2
## 2 Guadalupe Colin   25         A+ 56.4
## 3    Miguel Garza   32         O- 67.0
## 4   Alicia Zapata   76         O+ 73.1
```

3.2. Importando datos a un *data frame* desde un archivo

Para importar datos desde un archivo a un *data frame*, utilizamos la función **`read.table()`**, que nos permite leer un archivo de texto plano. Por ejemplo:

```
# Creando un data frame desde un archivo y mostrando el resultado
df_ratones <- read.table("Ratones.txt", header =T, sep = "\t", quote = "")
df_ratones
```

```
##        Id Peso Semana
## 1 Raton_1   20      1
## 2 Raton_2   32      2
## 3 Raton_3   18      1
## 4 Raton_4   45      3
```

> La función **str()**, proporciona una visualización compacta de la estructura interna de cualquier objeto R.

Veamos cuál es la estructura de los *data frames* df_ratones y df_pacientes:

```
# Visualizando la estructura de los data frames df_ratones y df_pacientes
str(df_ratones)
```

```
## 'data.frame':    4 obs. of  3 variables:
##  $ Id    : Factor w/ 4 levels "Raton_1","Raton_2",..: 1 2 3 4
##  $ Peso  : int  20 32 18 45
##  $ Semana: int  1 2 1 3
```

```
str(df_pacientes)
```

```
## 'data.frame':    4 obs. of  4 variables:
##  $ paciente  : Factor w/ 4 levels "Alfredo Aguirre",..: 1 3 4 2
##  $ tiposangre: Factor w/ 3 levels "A+","O-","O+": 3 1 2 3
##  $ peso      : num  95.2 56.4 67 73.1
##  $ edad      : num  50 25 32 76
```

Como podemos observar cuando hacemos uso de las funciones **data.frame()** o **read.table()**, las variables de tipo caracter son importadas como variables categóricas, que en R son representadas como factor. Si no quisieramos que esto sea así, podemos especificarlo al momento de crear el *data frame*, agregando el argumento stringsAsFactors = FALSE. Por ejemplo:

```
# Creando el data frame, con la función data.frame con el argumento
# stringsAsFactors = FALSE y consultando su estructura
df_pacientes <- data.frame(paciente,tiposangre,peso,edad,stringsAsFactors = FALSE)
str(df_pacientes)
```

```
## 'data.frame':    4 obs. of  4 variables:
##  $ paciente  : chr  "Alfredo Aguirre" "Guadalupe Colin" "Miguel Garza" "Alicia Zapata"
##  $ tiposangre: chr  "O+" "A+" "O-" "O+"
##  $ peso      : num  95.2 56.4 67 73.1
##  $ edad      : num  50 25 32 76
```

```
# Creando el data frame, con la función read.table y el argumento
# stringsAsFactors = FALSE y consultando su estructura
df_ratones <- read.table("Ratones.txt", header =T, sep = "\t", quote = "",
                         stringsAsFactors = FALSE)
str(df_ratones)
```

```
## 'data.frame':    4 obs. of  3 variables:
##  $ Id    : chr  "Raton_1" "Raton_2" "Raton_3" "Raton_4"
##  $ Peso  : int  20 32 18 45
##  $ Semana: int  1 2 1 3
```

4. Número de filas y columnas

Para visualizar la dimensión de un *data frame*, en términos del número de filas y columnas, utilizamos la función **dim()**. Por ejemplo:

```
# Consultando el número de filas y columnas del data frame
dim(df_ratones)
```

```
## [1] 4 3
```

```
dim(df_pacientes)
```

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

Con la funció **nrow()**, podemos visualizar solo el número de filas y con la función ncol, solo el número de columnas.

```
# Consultando el número de filas del data frame
nrow(df_ratones)
```

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

```
# Consultando el número de columnas del data frame
ncol(df_ratones)
```

```
## [1] 3
```

5. Nombres de filas y columnas

Cada columna en el *data frame* tiene un nombre. Incluso si no los especificamos directamente, R colocará los nombres de las columnas. En nuestro *data frame* df_pacientes, no especificamos los nombres de las columnas y R colocó de manera automática los nombres de cada una de las variables (vectores), con las que formamos el *data frame*.

Para visualizar o modificar los nombres de columnas utilizamos las funciones **colnames()** o **names()**.

```
# Visualizando los nombres de columnas de df_ratones
names(df_ratones)
```

```
## [1] "Id"     "Peso"   "Semana"
```

```
# Visualizando los nombres de columnas de df_pacientes
colnames(df_pacientes)
```

```
## [1] "paciente"   "tiposangre" "peso"       "edad"
```

Estas mismas funciones sirven para cambiar los nombres de las columnas.

```
# Visualizando los nombres de columnas de df_ratones
names(df_ratones)<-c("Name","weigth","week")
df_ratones
```

```
##      Name weigth week
## 1 Raton_1     20    1
## 2 Raton_2     32    2
## 3 Raton_3     18    1
## 4 Raton_4     45    3
```

```r
# Visualizando los nombres de columnas de df_pacientes
colnames(df_pacientes) <-c("Nombre","factorRh","kg","edad")
df_pacientes
```

```
##            Nombre factorRh   kg edad
## 1 Alfredo Aguirre       O+ 95.2   50
## 2 Guadalupe Colin       A+ 56.4   25
## 3    Miguel Garza       O- 67.0   32
## 4   Alicia Zapata       O+ 73.1   76
```

En el caso de las filas, también todas tienen un nombre, incluso sino lo especificamos. Por default los nombres de las filas son un números consecutivos, iniciando en "1". Para consultar o modificar los nombres de filas utilizamos la función **row.names()**.

```r
# Visualizando los nombres de columnas de df_pacientes
rownames(df_pacientes)
```

```
## [1] "1" "2" "3" "4"
```

Ahora cambiaremos los nombres de los renglones, poniendo, por ejemplo, enúmero de expediente:

```r
# Agregando nombres de renglones al data frame
rownames(df_pacientes)<-c("IMSS4010","IMSS5000","IMSS2345","IMSS0123")
df_pacientes
```

```
##                   Nombre factorRh   kg edad
## IMSS4010 Alfredo Aguirre       O+ 95.2   50
## IMSS5000 Guadalupe Colin       A+ 56.4   25
## IMSS2345    Miguel Garza       O- 67.0   32
## IMSS0123   Alicia Zapata       O+ 73.1   76
```

> Es importante mencionar que los nombres de columnas y filas deben ser valores únicos, es decir, no pueden repetirse.

Finalmente, la función **attributes()**, nos muestra los atributos generales del *data frame*.

```r
# Visualizando los atributos del data frame
attributes(df_pacientes)
```

```
## $names
## [1] "Nombre"   "factorRh" "kg"       "edad"
##
## $class
## [1] "data.frame"
##
## $row.names
## [1] "IMSS4010" "IMSS5000" "IMSS2345" "IMSS0123"
```

6. Seleccionando elementos

La sintaxis para acceder a los elementos de un *data frame* es: **nombre_dataframe[filas,columnas]**, donde filas y columnas corresponden a los índices de las filas y columnas de los elementos que queremos consultar.

Para la selección de elementos de un *data frame*, trabajaremos con los datos "iris", que se encuentra disponible en el conjunto de datos de R. Echemos un vistazo general a iris.

```r
# Consultando la estructura de iris
str(iris)
```

```
## 'data.frame':    150 obs. of  5 variables:
##  $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
##  $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
##  $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
##  $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...
```

```r
# Consultando los primeros renglones de iris
head(iris)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

Seeccionemos algunos elementos de iris.

```r
# Obteniendo el elemento de la fila 1, columna 1
iris[1,1]
```

```
## [1] 5.1
```

```r
# Obteniendo el elemento de la fila 1, columna 5
iris[1,5]
```

```
## [1] setosa
## Levels: setosa versicolor virginica
```

```r
# Obteniendo la columna 3
iris[ ,3]
```

```
##   [1] 1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 1.5 1.6 1.4 1.1 1.2 1.5 1.3
##  [18] 1.4 1.7 1.5 1.7 1.5 1.0 1.7 1.9 1.6 1.6 1.5 1.4 1.6 1.6 1.5 1.5 1.4
##  [35] 1.5 1.2 1.3 1.4 1.3 1.5 1.3 1.3 1.3 1.6 1.9 1.4 1.6 1.4 1.5 1.4 4.7
##  [52] 4.5 4.9 4.0 4.6 4.5 4.7 3.3 4.6 3.9 3.5 4.2 4.0 4.7 3.6 4.4 4.5 4.1
##  [69] 4.5 3.9 4.8 4.0 4.9 4.7 4.3 4.4 4.8 5.0 4.5 3.5 3.8 3.7 3.9 5.1 4.5
##  [86] 4.5 4.7 4.4 4.1 4.0 4.4 4.6 4.0 3.3 4.2 4.2 4.2 4.3 3.0 4.1 6.0 5.1
## [103] 5.9 5.6 5.8 6.6 4.5 6.3 5.8 6.1 5.1 5.3 5.5 5.0 5.1 5.3 5.5 6.7 6.9
## [120] 5.0 5.7 4.9 6.7 4.9 5.7 6.0 4.8 4.9 5.6 5.8 6.1 6.4 5.6 5.1 5.6 6.1
## [137] 5.6 5.5 4.8 5.4 5.6 5.1 5.1 5.9 5.7 5.2 5.0 5.2 5.4 5.1
```

```r
# Obteniendo la fila 15
iris[15, ]
```

```
##    Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 15          5.8           4          1.2         0.2  setosa
```

```r
# Obteniendo filas consecutivas todas las columnas
iris[12:16, ]
```

```
##    Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 12          4.8         3.4          1.6         0.2  setosa
## 13          4.8         3.0          1.4         0.1  setosa
## 14          4.3         3.0          1.1         0.1  setosa
```

```
## 15          5.8         4.0          1.2         0.2  setosa
## 16          5.7         4.4          1.5         0.4  setosa
```

```
# Omitiendo una columna
head(iris[ ,-5])
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width
## 1          5.1         3.5          1.4         0.2
## 2          4.9         3.0          1.4         0.2
## 3          4.7         3.2          1.3         0.2
## 4          4.6         3.1          1.5         0.2
## 5          5.0         3.6          1.4         0.2
## 6          5.4         3.9          1.7         0.4
```

```
# seleccionando por nombre
iris["25","Sepal.Width"]
```

```
## [1] 3.4
```

Los *data frames* poseen las características de las listas, por lo tanto podemos acceder a sus columnas por medio del símbolo **$**.

```
# Consultando la columna Sepal.Length, usando el simbolo $
iris$Sepal.Length
```

```
##   [1] 5.1 4.9 4.7 4.6 5.0 5.4 4.6 5.0 4.4 4.9 5.4 4.8 4.8 4.3 5.8 5.7 5.4
##  [18] 5.1 5.7 5.1 5.4 5.1 4.6 5.1 4.8 5.0 5.0 5.2 5.2 4.7 4.8 5.4 5.2 5.5
##  [35] 4.9 5.0 5.5 4.9 4.4 5.1 5.0 4.5 4.4 5.0 5.1 4.8 5.1 4.6 5.3 5.0 7.0
##  [52] 6.4 6.9 5.5 6.5 5.7 6.3 4.9 6.6 5.2 5.0 5.9 6.0 6.1 5.6 6.7 5.6 5.8
##  [69] 6.2 5.6 5.9 6.1 6.3 6.1 6.4 6.6 6.8 6.7 6.0 5.7 5.5 5.5 5.8 6.0 5.4
##  [86] 6.0 6.7 6.3 5.6 5.5 5.5 6.1 5.8 5.0 5.6 5.7 5.7 6.2 5.1 5.7 6.3 5.8
## [103] 7.1 6.3 6.5 7.6 4.9 7.3 6.7 7.2 6.5 6.4 6.8 5.7 5.8 6.4 6.5 7.7 7.7
## [120] 6.0 6.9 5.6 7.7 6.3 6.7 7.2 6.2 6.1 6.4 7.2 7.4 7.9 6.4 6.3 6.1 7.7
## [137] 6.3 6.4 6.0 6.9 6.7 6.9 5.8 6.8 6.7 6.7 6.3 6.5 6.2 5.9
```

También podemos utilizar el formato de **[ ]**, colocando un índice o con el nombre de la columna.

```
# Consultando la columna 1 por medio de corchetes
head(iris[1])
```

```
##   Sepal.Length
## 1          5.1
## 2          4.9
## 3          4.7
## 4          4.6
## 5          5.0
## 6          5.4
```

```
head(iris["Sepal.Length"])
```

```
##   Sepal.Length
## 1          5.1
## 2          4.9
## 3          4.7
## 4          4.6
## 5          5.0
## 6          5.4
```

7. Algunas funciones

Algunas funciones que se pueden aplicar a los *data frames* son:

```
# El máximo de los elementos
max(iris[,1:4])
```

```
## [1] 7.9
```

```
# suma de los elementos por renglón
rowSums(iris[,1:4])
```

```
##   [1] 10.2  9.5  9.4  9.4 10.2 11.4  9.7 10.1  8.9  9.6 10.8 10.0  9.3  8.5
##  [15] 11.2 12.0 11.0 10.3 11.5 10.7 10.7 10.7  9.4 10.6 10.3  9.8 10.4 10.4
##  [29] 10.2  9.7  9.7 10.7 10.9 11.3  9.7  9.6 10.5 10.0  8.9 10.2 10.1  8.4
##  [43]  9.1 10.7 11.2  9.5 10.7  9.4 10.7  9.9 16.3 15.6 16.4 13.1 15.4 14.3
##  [57] 15.9 11.6 15.4 13.2 11.5 14.6 13.2 15.1 13.4 15.6 14.6 13.6 14.4 13.1
##  [71] 15.7 14.2 15.2 14.8 14.9 15.4 15.8 16.4 14.9 12.8 12.8 12.6 13.6 15.4
##  [85] 14.4 15.5 16.0 14.3 14.0 13.3 13.7 15.1 13.6 11.6 13.8 14.1 14.1 14.7
##  [99] 11.7 13.9 18.1 15.5 18.1 16.6 17.5 19.3 13.6 18.3 16.8 19.4 16.8 16.3
## [113] 17.4 15.2 16.1 17.2 16.8 20.4 19.5 14.7 18.1 15.3 19.2 15.7 17.8 18.2
## [127] 15.6 15.8 16.9 17.6 18.2 20.1 17.0 15.7 15.7 19.1 17.7 16.8 15.6 17.5
## [141] 17.8 17.4 15.5 18.2 18.2 17.2 15.7 16.7 17.3 15.8
```

```
# Suma de los elementos por columnas
colSums(iris[ ,-5])
```

```
## Sepal.Length  Sepal.Width Petal.Length  Petal.Width
##        876.5        458.6        563.7        179.9
```

```
# Media de los elementos por columna
colMeans(iris[ ,-5])
```

```
## Sepal.Length  Sepal.Width Petal.Length  Petal.Width
##     5.843333     3.057333     3.758000     1.199333
```

```
# tabla de frecuencias
table(iris$Petal.Width)
```

```
##
## 0.1 0.2 0.3 0.4 0.5 0.6   1 1.1 1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9   2 2.1
##   5  29   7   7   1   1   7   3   5  13   8  12   4   2  12   5   6   6
## 2.2 2.3 2.4 2.5
##   3   8   3   3
```

```
# Resumen estadístico por columnas
summary(iris)
```

```
##   Sepal.Length    Sepal.Width     Petal.Length    Petal.Width
##  Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100
##  1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300
##  Median :5.800   Median :3.000   Median :4.350   Median :1.300
##  Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
##  3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
##  Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
##        Species
##  setosa    :50
##  versicolor:50
##  virginica :50
```

```
##
##
##
```

8. Seleccionando datos por condición

Existen diversas formas de seleccionar datos que cumplan una determinada condición, una de ellas es por medio de la función **subset()**.

```
# Seleccionando las filas donde Sepal.Length sea mayor a 7
subset(iris,iris$Sepal.Length > 7)
```

```
##     Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
## 103          7.1         3.0          5.9         2.1 virginica
## 106          7.6         3.0          6.6         2.1 virginica
## 108          7.3         2.9          6.3         1.8 virginica
## 110          7.2         3.6          6.1         2.5 virginica
## 118          7.7         3.8          6.7         2.2 virginica
## 119          7.7         2.6          6.9         2.3 virginica
## 123          7.7         2.8          6.7         2.0 virginica
## 126          7.2         3.2          6.0         1.8 virginica
## 130          7.2         3.0          5.8         1.6 virginica
## 131          7.4         2.8          6.1         1.9 virginica
## 132          7.9         3.8          6.4         2.0 virginica
## 136          7.7         3.0          6.1         2.3 virginica
```

```
# Seleccionando las filas donde Sepal.Length sea mayor a 7 y mostrando solo las columnas Specie y Sepal
subset(iris,iris$Sepal.Length > 7,select=c(Species,Sepal.Length))
```

```
##       Species Sepal.Length
## 103 virginica          7.1
## 106 virginica          7.6
## 108 virginica          7.3
## 110 virginica          7.2
## 118 virginica          7.7
## 119 virginica          7.7
## 123 virginica          7.7
## 126 virginica          7.2
## 130 virginica          7.2
## 131 virginica          7.4
## 132 virginica          7.9
## 136 virginica          7.7
```

9. Agregando Filas y/o Columnas

Las funciones **rbind()** y **cbind()** se utilizan para añadir renglones y columnas, respectivamente, al *data frame*.

Por ejemplo, podemos agregar una columna llamada sexo al *data frame* df_pacientes.

```
# Generando el vector sexo
sexo<-c("M","F","M","F")
# Agregando la columna sexo al data frame
```

```
df_pacientes<-cbind(df_pacientes,sexo)
df_pacientes
```

```
##                    Nombre factorRh   kg edad sexo
## IMSS4010 Alfredo Aguirre       O+ 95.2   50    M
## IMSS5000 Guadalupe Colin       A+ 56.4   25    F
## IMSS2345    Miguel Garza       O- 67.0   32    M
## IMSS0123   Alicia Zapata       O+ 73.1   76    F
```

Ahora agregaremos un renglón, con la información de un nuevo paciente.

```
# Geneando la lista con los datos de un paciente
nuevo_paciente<-list("Antonio Valencia","O+",38,10,"M")
# Agregando la lista al data frame
df_pacientes<-rbind(df_pacientes,IMSS3967=nuevo_paciente)
df_pacientes
```

```
##                     Nombre factorRh   kg edad sexo
## IMSS4010  Alfredo Aguirre       O+ 95.2   50    M
## IMSS5000  Guadalupe Colin       A+ 56.4   25    F
## IMSS2345     Miguel Garza       O- 67.0   32    M
## IMSS0123    Alicia Zapata       O+ 73.1   76    F
## IMSS3967 Antonio Valencia       O+ 38.0   10    M
```

Del mismo modo que utilizamos los operadores **[,]** y **$** para acceder y cambiar un valor, podemos añadir columnas en un *data frame* del siguiente modo:

```
# Agregando la columna temperatura
df_pacientes$temperatura<-c(37,36.5,38,36.1,39)

# Agregando la columna estatura
df_pacientes["estatura"]<-c(1.75,1.69,1.83,1.5,1.4)
df_pacientes
```

```
##                     Nombre factorRh   kg edad sexo temperatura estatura
## IMSS4010  Alfredo Aguirre       O+ 95.2   50    M        37.0     1.75
## IMSS5000  Guadalupe Colin       A+ 56.4   25    F        36.5     1.69
## IMSS2345     Miguel Garza       O- 67.0   32    M        38.0     1.83
## IMSS0123    Alicia Zapata       O+ 73.1   76    F        36.1     1.50
## IMSS3967 Antonio Valencia       O+ 38.0   10    M        39.0     1.40
```

10. Resumen de comandos

Usar **data.frame()** para crear un *data frame*

Usar **read.table()** para importar datos de un archivo a un *data frame*

Usar **str()** para visualizar la estructura del *data frame*

Usar **dim()** para visualizar el número de filas y columnas de un *data frame*

Usar **nrow()** para consultar el número de renglones de un *data frame*

Usar **ncol()** para consultar el número de columnas de un *data frame*

Usar **names()** o **colnames()** para consultar o modificar los nombres de las columnas de un *data frame*

Usar **rownames()** para consultar o modificar los nombres de renglones de un *data frame*

Usar **`attributes()`** para visualizar los atributos del *data frame*

Usar **`[ , ]`** para acceder a los elementos de un *data frame*

Usar **`head()`** para consultar las primeras filas de un *data frame*

Usar **`rowSums()`** para obtener la suma de elementos por renglón

Usar **`colSums()`** para obtener la suma de elementos por columna

Usar **`colMeans()`** para obtener el promedio de elementos por columna

Usar **`summary()`** para obtener un resumen estadístico del *data frame*

Usar **`subset()`** para seleccionar datos de un *data frame*

Usar **`cbind()`** para agregar una nueva columna a un *data frame*

Usar **`rbind()`** para agregar una nueva fila a un *data frame*