

Acceso a datos en formato RData

```
res = load("dataset.RData")
```

Se cargan como variables `res` y `dataset`.
`Res` es simplemente una referencia y `dataset` es el conjunto de datos.

```
save(dataset, file = "dataset.RData")
```

Importación desde Excel

```
dataset = read.table("dataset.csv", header = TRUE, sep = ";", dec = ",")
```

```
dataset = read.csv2("dataset.csv", header = TRUE, sep = ";", dec = ",")
```

Convertimos el excel a csv y después tenemos dos opciones

Acceso a datos de librerías

```
data(package = .packages(all.available = TRUE))
```

Ver todos los datasets disponibles

```
data(dataset)
```

Cargar uno de los datasets disponibles

Importación desde SPSS

```
library(foreign)
```

```
dataset = read.spss("dataset.sav", to.data.frame = TRUE)
```

```
spss.get()
```

del paquete `Hmisc`

Si hay fechas

Exportación de datos

```
write.table(dataset, file = "dataset.txt")
```

A archivo txt

```
write.csv2(dataset, file = "dataset.csv")
```

A archivo csv

```
save(dataset, file = "dataset.RData")
```

A archivo RData

Manipulación de datos con dplyr

```
data.frame(Etiquetas = attr(empleados, "variable.labels"))
```

Ver información sobre cada columna si está disponible (no es parte de `dplyr`)

```
attr(empleados, "variable.labels") <- NULL
```

Borrar información sobre columnas (no pertenece a `dplyr`)

```
dataset2 = select(dataset1, id, sexo, minoria, tiempemp, salini, salario)
```

Seleccionar únicamente una serie de columnas

```
dataset2 = select(dataset1, id, sexo, noblanca=minoria, tiempemp, salini, salario)
```

Seleccionar únicamente una serie de columnas y cambiar el nombre de una

```
select(empleados, sexo:salario)
```

Seleccionar columnas contenidas entre dos columnas.

```
select(empleados, -(sexo:salario))
```

Seleccionar todas las columnas menos las contenidas en un rango.

```
select(empleados, starts_with("s"))
```

Seleccionar solo las columnas cuya etiqueta empiece por `s`

```
ends_with(), contains(), matches(), one_of()
```

(ver `?select`)

```
mutate(emplea2, incsal = salario - salini, tsal = incsal/tiempemp)
```

Nuevas columnas a partir de otras columnas

```
filter(emplea2, sexo == "Mujer", minoria == "Si")
```

Filtrar datos. Sintaxis parecida a `subset`

```
arrange(emplea2, salario)
```

Organizar datos según una columna. Se cambia el índice de filas.

Manipulación de datos con dplyr (cont)

```
arrange(emplea2, desc(salini), salario)
```

Organizar datos según una columna de forma descendente, y según la siguiente de forma ascendente si dos datos son iguales. Se cambia el índice de filas.

```
summarise(empleados, sal.med = mean(salario), n = n())
```

resumir valores

```
summarise(group_by(empleados, sexo, minoria), sal.med = mean(salario), n = n())
```

agrupar casos

```
empleados %>% filter(catlab == "Directivo") %>% group_by(sexo, minoria) %>% summarise(sal.med = mean(salario), n = n())
```

```
empleados %>% select(sexo, catlab, salario) %>% filter(catlab != "Seguridad") %>% group_by(catlab) %>% mutate(saldif = salario - mean(salario)) %>% ungroup() %>% boxplot(saldif ~ sexo*droplevels(catlab), data = .) abline(h = 0, lty = 2)
```

el operador pipe nos permite canalizar la salida de una función a la entrada de otra función el operador pipe nos permite canalizar la salida de una función a la entrada de otra función

Lectura de archivos de texto

```
dataset = read.table("dataset.txt", header = TRUE, quote="\"")
```

`header`: indica si el fichero tiene cabecera (`header=TRUE`) o no (`header=FALSE`). Por defecto toma el valor `header=FALSE`.

`sep`: carácter separador de columnas que por defecto es un espacio en blanco (`sep=""`). Otras opciones serían: `sep=","` si el separador es un `","`, `sep=" "` si el separador es un `" "`, etc.

`dec`: carácter utilizado en el fichero para los números decimales. Por defecto se establece `dec = "."`. Si los decimales vienen dados por `","` se utiliza `dec = ","`



Lectura de archivos de texto (cont)

```
read.delim(file, header = TRUE, sep = "\t",  
dec = ".")
```

```
read.delim2(file, header = TRUE, sep = "\t",  
dec = ",")
```

Ficheros separados por tabuladores

Operaciones con variables

```
x=cars$speed
```

```
x=cars[,1]
```

Equivalente

```
cars$velocidad = cars$speed / 0.62137
```

Crear una nueva columna a partir de otra

```
cars = cars[, c("velocidad", "distancia")]
```

Eliminar columnas. Funciona también
con índices

```
ii = order(cars$dist)
```

```
cars2 = cars[ii, ]
```

Ordenar filas de un dataframe según
los valores de una columna

```
subset(familia,genero=="mujer"&cabell-  
o=="rubio")
```

```
subset(cars, speed > 10 & speed < 15 & dist  
> 45)
```

elegir una submuestra que cumpla
determinadas condiciones

```
ii = cars$speed > 10 & cars$speed < 15 &  
cars$dist > 45
```

```
cars[ii, ]
```

Filtrado con índices

```
it <- which(ii)
```

```
cars[it, ]
```

Mismo resultado. Which devuelve los
índices de los elementos que son True

```
id <- which(!ii)
```

```
cars[id, ]
```

Resultado opuesto

