PROGRAMACIÓ
Grau en Estadística UB-UPC, Abril 2021
Prof Robert Joan-Arinyo

Taules (Data Frames)

---

1 Definició

En el llenguatge de programació `R`, una taula és una estructura específica per a l'emmagatzemament de dades configurada com una llista de vectors tots de la mateixa llargada.

Entre les dades d'una taula acostuma a haver alguna mena de relació lògica. Per exemple, en estadística s'acostuma a suposar que cada columna emmagatzema els valors d'una variable mentre que cada fila correspon a una observació dels valors de les variables. Les taules o *data frames* en llenguatge `R`, són les estructures de dades més utilitzades en estadística per a l'anàlisi de dades.

Hom podrà individualitzar les informacions components d'una taula de manera semblant a com es feia en els vectors, llistes o matrius. Un exemple de taula és

```
        edad pes alçada  esport
Manel    20   65    174  basquet
Gisela   22   70    180  paddle
Roger    19   68    170  futbol
```

2 Creació de taules

Les taules es poden crear de diverses maneres. Una manera és crear una taula buida. Una altra permet crear-les tot definint el nombre i nom de les columnes. Finalment hom pot definir una taula amb un conjunt determinat d'informacions. Veiem-ne alguns casos.

2.1 Taula buida

La creació de qualsevol taula es fa amb l'operador `data.frame`. En el cas de crear una taula buida la sintaxi és

```
nomTaula <- data.frame()
```

Recordeu la similitud d'aquesta construcció amb la creació d'altres tipus de dades buides en llenguatge `R`.

Una altra manera més adient de definir una taula buida implica definir prèviament el conjunt de vectors columna que la compondran. Per exemple

```
nom    <- character()
edat   <- numeric()
dni    <- integer()
genere <- factor()
dataN  <- as.Date(character())

taula  <- data.frame(c(nom, edat, dni, genere, dataN))
```

Notis la presència de l'operador de concatenació `c(...)`.

2.2 Taula amb dades

La manera básica de crear una taula inicialitzada amb dades respon a una senzilla variant del cas que acabem de veure per a les taules buides. Només cal definir primer els vectors que la compondran amb les dades corresponents.

```
nom       <- c("Joan", "Pau", "Maria", "Roger")
cognom    <- c("Franquet", "Ripoll", "Xandri", "Garcia")
genere    <- c("HOME", "HOME", "DONA", "HOME")
dataN     <- c("2015-05-10", "1849-10-07", "1892-03-26","1817-07-18")
telefon   <- c(134761925, 934787625, 934760920, 634661901)
```

i tot seguit definr la taula com

```
grupXat <- data.frame(nom, cognom, genere, dataN, telefon)
```

La taula resultant seria

```
    nom   cognom genere       dataN   telefon
1  Joan Franquet   HOME  2015-05-10 134761925
2   Pau   Ripoll   HOME  1849-10-07 934787625
3 Maria   Xandri   DONA  1892-03-26 934760920
4 Roger   Garcia   HOME  1817-07-18 634661901
```

Recorda que cada vector defineix una columna de la taula.

2.3 Taula llegida de fitxer

Quan els conjunts de dades són grans, no és viable entrar-los pel teclat cada cop que hom executa un tractament. En aquestes circumstàncies, les dades s'emmagatzemen en fitxers. Aleshores, les taules s'inicialitzen per lectura d'un fitxer de text amb la comanda:

```
dat <- read.table("nomDelFitxer.txt", header=X, sringsAsFactors=FALSE)
```

tot i que l'extensió del nom del fitxer pot ser qualsevol, per exemple `nomDelFitxer.dades`.

Quan el fitxer és de tipus `csv` (*comma separated values*) s'usarà la comanda

```
dat <- read.csv("nomDelFitxer.csv", header=X, sringsAsFactors=FALSE)
```

El paràmetre `X` assignat a `header` prendrà valor `TRUE` o `FALSE` en funció de que la taula tingui o no una fila inicial amb els noms de les variables, és a dir, les columnes.

3 Individualització de components

Hi ha dues maneres d'individualitzar les components d'una taula i accedir a la informació que contenen. Una manera utilitza el nom de la component. Per exemple, en la taula de més amunt, les comandes equivalents

```
grupXat$nom          names(grupXat)
```

mostarien

```
Joan  Pau  Maria  Roger
```

La manera més general d'individualitzar components en una taula és fer referència directa a la component mitjançant el valor dels índexs de la posició que ocupa la component amb la notació estàndard `nomTaula[fila, columna]`. Recorda que cada vector de la definició representa una columna de la taula. Per tant, per l'exemple precedent,

```
grupXat[2, 4]
```

contindrà la cadena de caràcters `1849-10-07`. De manera natural, s'apliquen els filtres de files i columnes. Per exemple

```
grupXat[2, ]  és  Pau Ripoll HOME 1849-10-07 934787625
grupXat[, 3]  és  HOME HOME DONA HOME
```

4 Operacions amb taules

En tot el que segueix, suposarem que el nom de la taula de la qual es parla és `grupXat`.

- Escriure una taula per pantalla.

  ```
  print(nomTaula)
  ```

- Afegir una columna és immediat. Per exemple, el codi

  ```
  grupXat$moto <- c("si", "no", "si", "si")
  ```

  afegeix a la taula `grupXat` una nova columna que indica quin membre té motocicleta i quin no en té. La taula resultant és

  ```
     nom   cognom genere       dataN  telefon  moto
  1  Joan Franquet   HOME  2015-05-10 134761925   si
  2   Pau   Ripoll   HOME  1849-10-07 934787625   no
  3 Maria   Xandri   DONA  1892-03-26 934760920   si
  4 Roger   Garcia   HOME  1817-07-18 634661901   si
  ```

- Afegir una fila resulta un xic més complicat i requeriex la construcció prèvia de la fila i la utilització de la funció `rbind()`. Aquesta funció es pot usar per a combinar per files vectors, matrius o data frames. La sintaxi general és

```
rbind(unDataFrame, filaNova)
```

Per exemple, el codi

```
nom      <- "Gisela"
cognom   <- "Sanchis"
genere   <- "DONA"
dataN    <- "1950-01-23"
telefon  <- 777333337
moto     <- no
novaFila <- data.frame(nom, cognom, genere, dataN, telefon, moto)
grupXat  <- rbind(grupXat, novaFila)
```

genera la nova taula `grupXat`

```
     nom   cognom genere       dataN  telefon  moto
1   Joan Franquet   HOME  2015-05-10 134761925   si
2    Pau   Ripoll   HOME  1849-10-07 934787625   no
3  Maria   Xandri   DONA  1892-03-26 934760920   si
4  Roger   Garcia   HOME  1817-07-18 634661901   si
5 Gisela  Sanchis   DONA  1950-01-23 777333337   no
```

Nota que el que realment s'afegeix a un data frame és un altre data frame amb una sola fila.

- Nombre de files i de columnes d'una taula. Per la taula `grupXat` tindrem que

```
nrow(grupXat)   és 4
ncol(grupXat)   és 6
```

- Els recorreguts i les cerques sobrer taules es resolen de la mateixa manera que es feia en les matrius.

5 Operacions avançades

- Selecció d'un subconjunt d'una taula.

  Aquesta operació resulta força útil quan l'objectiu és tractar només parts d'una taula. L'operació es porta a terme usant la funció `subset()` i la sintaxi general és

```
subset(x, condicio)
arguments:
  * x: data frame d'on s'extreu el sunconjunt
  * condicio: defineix la condio que caracteritza el subconjunt
```

Per exemple, si considerem la taula `grupXat`, els subconjunt definit per

```
subset(grupXat, genere == "HOME")
```

resulta ser

```
    nom   cognom genere       dataN  telefon  moto
1  Joan Franquet   HOME  2015-05-10 134761925   si
2   Pau   Ripoll   HOME  1849-10-07 934787625   no
3 Roger   Garcia   HOME  1817-07-18 634661901   si
```

- Fusió de dos data frames.

  En general hom té dades provinents de diverses fonts. Per tal de analitzar-les com si tinguèssin el mateix origen, cal fusionar-les segons una o més variables de referència. Hi ha dos tipus de fusió: amb coincidència total de variables clau o amb coincidència parcial. De moment només presentarem la coincidència total en la qual el resultat de la fusió és un data frame que només contè les informacions comunes. la sintaxi de la qual és

```
merge(x, y, by.x = colx, by.y = coly)
arguments:
 * x:    Data frame base
 * y:    Data frame a fusionar
 * colx: Nom de la columna del data frame x usada per a la fusio.
 * coly: Nom de la columna del data frame y usada per a la fusio.
```

  Suposem que disposem de la taula de nom `grupMotards`

```
      nom    cognom genere    telefon  maquina
1    Joan  Franquet   HOME  134761925   harley
2     Pau    Ripoll   HOME  934787625  triumph
3 Mariona    Llopis   DONA  788833337  montesa
4  Gisela   Sanchis   DONA  775553300  montesa
```

```
  5   Maria    Xandri   DONA  934760920     ossa
  6   Roger    Garcia   HOME  634661901   harley
  7  Ricard  Martinez   HOME  777333222   ducati
```

La fusió

```
xatXM <- merge(grupXat, grupMotards, by.x="nom", by.y="nom")
```

genera el data frame

```
    nom   cognom genere.x      dataN telefon.x  malnom genere.y telefon.y maquina
1  Joan Franquet     HOME 2015-05-10 134761925    Juni     HOME 134761925  harley
2 Maria   Xandri     DONA 1892-03-26 934760920   Maria     DONA 934760920    ossa
3   Pau   Ripoll     HOME 1849-10-07 934787625  Paueti     HOME 934787625 triumph
4 Roger   Garcia     HOME 1817-07-18 634661901 Rinjols     HOME 634661901  harley
```

Mentre que

```
grupXM <- merge(grupXat, grupMotards, by.x = "telefon", by.y="telefon")
```

genera

```
    telefon nom.x   cognom genere.x      dataN nom.y  malnom genere.y maquina
1 134761925  Joan Franquet     HOME 2015-05-10  Joan    Juni     HOME  harley
2 634661901 Roger   Garcia     HOME 1817-07-18 Roger Rinjols     HOME  harley
3 934760920 Maria   Xandri     DONA 1892-03-26 Maria   Maria     DONA    ossa
4 934787625   Pau   Ripoll     HOME 1849-10-07   Pau  Paueti     HOME triumph
```

Notis la inclusió repetida en la taula final de les informacions comuns a les taules fusionades. En general, quam definim taules dins d'un mateix problema serà convenient no repetir informacions en una taula que es pugui obtenir a partir d'una altra.

- Ordenació d'un data frame segons una variable

  En l'anàlisis de dades, l'ordenació de les dades disponibles és una operació cabdal entre d'altres raons perquè facilita i accelera enormement els processos de càlcul posteriors. De fet, una màxima del tractament automàtic de la informació diu

  Primer ordenar, després calcular

  El llenguatge `R` ofereix la funció `order()` que permet ordenar vectors segons valors creixents o decreixents. La sintaxi mes senzilla és

```
sort(v, decreasing = FALSE}
arguments:
 * x:    vector a ordenar.
 * decreasing: control de l'ordre. Per defecte, el paràmetre pren
   valor FALSE.
```

L'aplicació a l'ordenació de dataframes té algunes particularitats. Si considerem el dataframe `grupMotards` definit més amunt, el codi

```
df <- grupMotards[order(grupMotards$cognom), ]
```

generaria

```
      nom    cognom genere    telefon  maquina
1    Joan  Franquet   HOME  134761925   harley
2   Roger    Garcia   HOME  634661901   harley
3 Mariona    Llopis   DONA  788833337  montesa
4  Ricard  Martinez   HOME  777333222   ducati
5     Pau    Ripoll   HOME  934787625  triumph
6  Gisela   Sanchis   DONA  775553300  montesa
7   Maria    Xandri   DONA  934760920     ossa
```

Nota la necessitat d'escriure una coma abans de tancar el claudàtor! La funció `order()` ofereix moltes més possibilitats. Llegeix i investiga-les pel teu compte.