Introducción a R

Reshape Data - change the layout of values in a table

Use **gather()** and **spread()** to reorganize the values of a table into a new layout. Each uses the idea of a key column: value column pair.

gather(data, key, value, ..., na.rm = FALSE, convert = FALSE, factor_key = FALSE)

Gather moves column names into a key column, gathering the column values into a single value column.

table4a

country	1999	2000
A	0.7K	2K
B	37K	80K
C	212K	213K

→

country	year	cases
A	1999	0.7K
B	1999	37K
C	1999	212K
A	2000	2K
B	2000	80K
C	2000	213K

key value

gather(table4a, `1999`, `2000`, key = "year", value = "cases")

spread(data, key, value, fill = NA, convert = FALSE, drop = TRUE, sep = NULL)

Spread moves the unique values of a key column into the column names, spreading the values of a value column across the new columns that result.

table2

country	year	type	count
A	1999	cases	0.7K
A	1999	pop	19M
A	2000	cases	2K
A	2000	pop	20M
B	1999	cases	37K
B	1999	pop	172M
B	2000	cases	80K
B	2000	pop	174M
C	1999	cases	212K
C	1999	pop	1T
C	2000	cases	213K
C	2000	pop	1T

key value

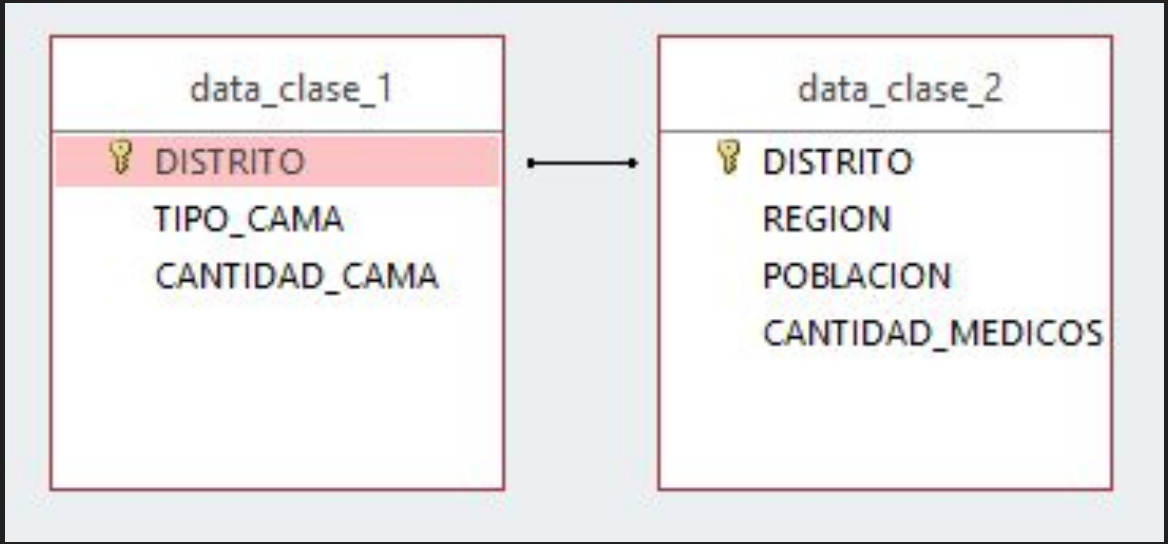
→

country	year	cases	pop
A	1999	0.7K	19M
A	2000	2K	20M
B	1999	37K	172M
B	2000	80K	174M
C	1999	212K	1T
C	2000	213K	1T

spread(table2, type, count)

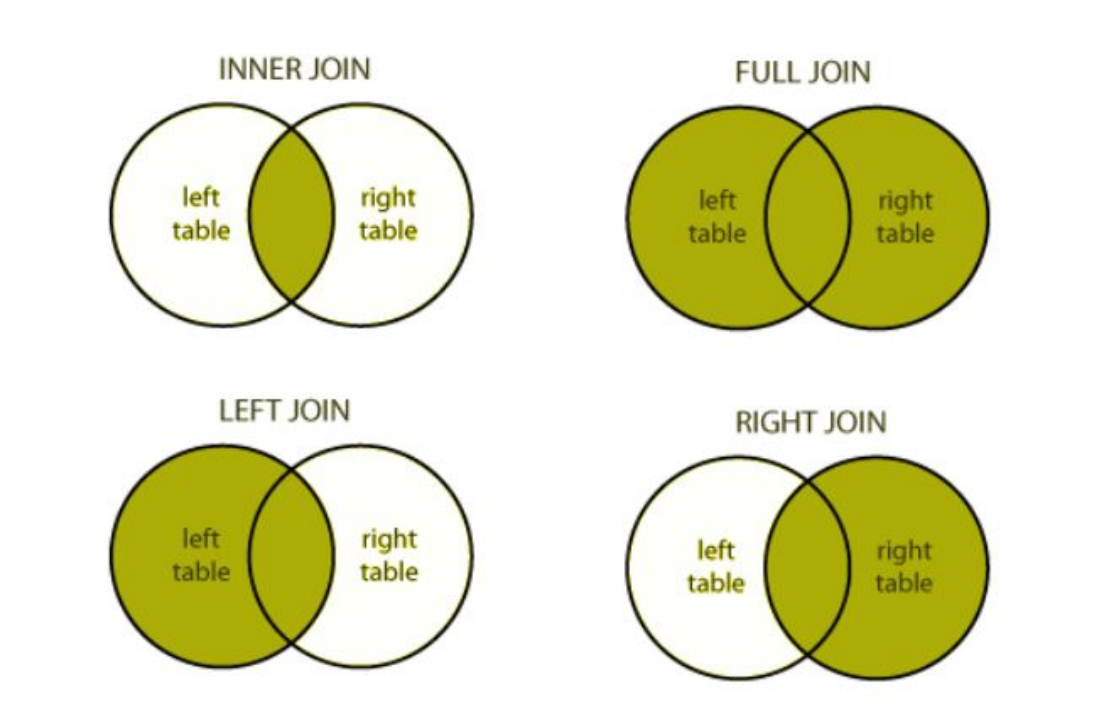
PRIMARY KEY

Permite identificar en un dataset a una columna que identifica de forma única a cada fila de una tabla

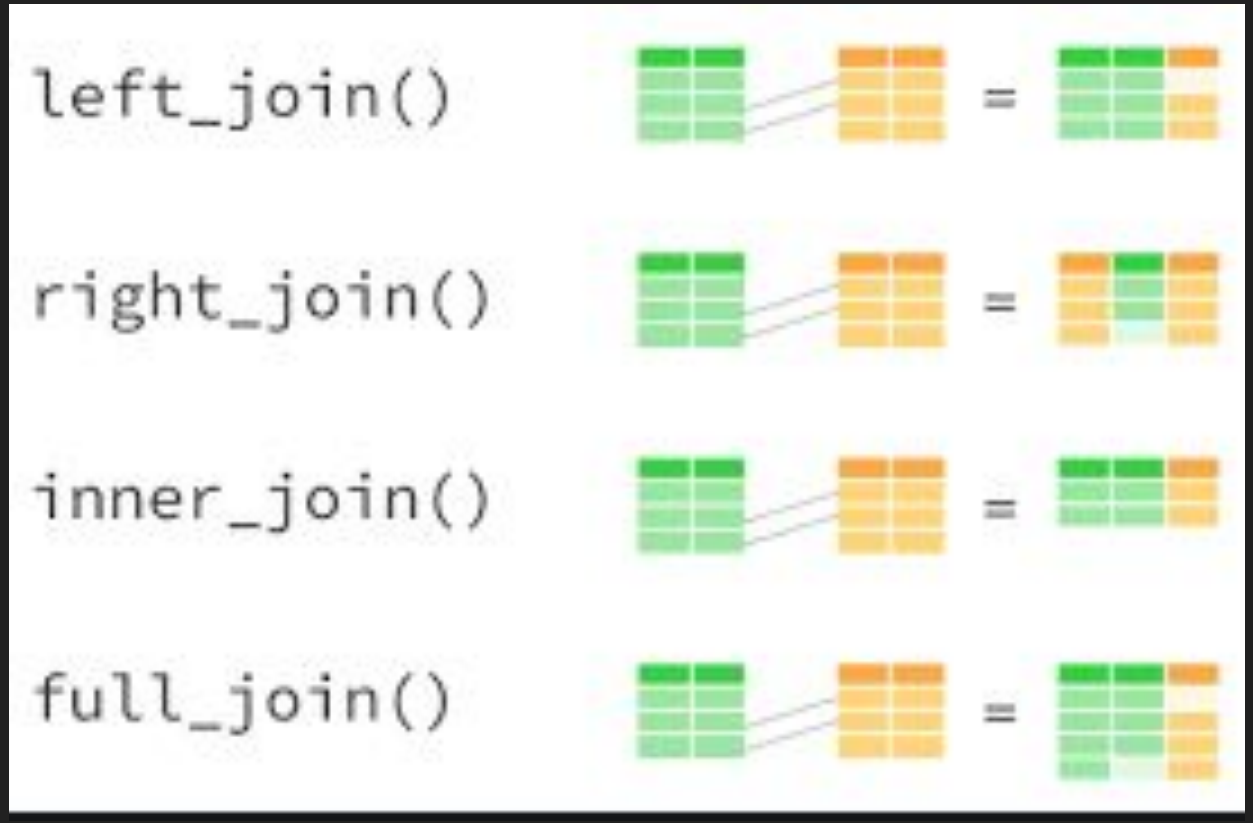


JOINS

Unimos dos o más tablas en una única y nueva tabla

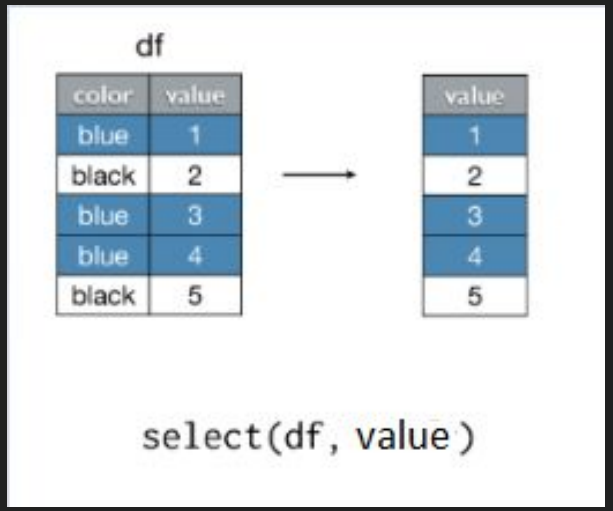


Formas de pedirle a R que haga la unión:



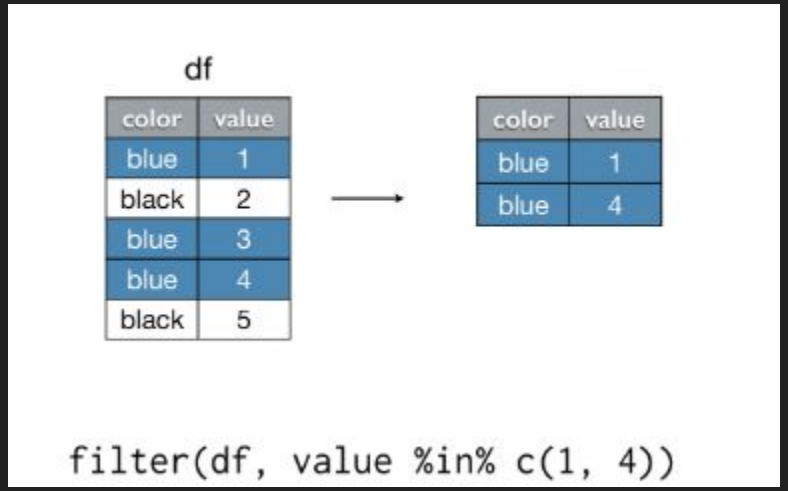
Repaso clase anterior:

```
select()          # para seleccionar columnas
```



http://gauss.inf.um.es:8080/tabular-tidyverse/#section-dplyr

```
filter()          # para filtrar filas
```



`mutate()` # para agregar o modificar valores de columnas no agrupadas

df

color	value
blue	1
black	2
blue	3
blue	4
black	5

→

color	value	double	quadruple
blue	1	2	4
black	2	4	8
blue	3	6	12
blue	4	8	16
black	5	10	20

```
mutate(df, double = 2 * value,
    quadruple = 2 * double)
```

Función	Caracteristicas	Aplicación*
sum()	Devuelve la suma	sum(data$columna)
mean()	Devuelve el promedio	mean(data$columna)
median()	Devuelve la mediana	median(data$columna)
max()	Devuelve el maximo valor	max(data$columna)
min()	Devuelve minimo valor	min(data$columna)
count()	Devuelve la cantidad de observaciones (frecuencia) con respecto a una variable determinada	count(data$columna)
table()	Devuelve la cantidad de observaciones (frecuencia) con respecto a una variable determinada (para groupby)	table(data$columna)
n()	Devuelve la cantidad de observaciones (frecuencia) para una variable no necesariamente determinada (para groupby)	n(data$columna)
quantile()	Devuelve el valor correspondiente al cuantil deseado (0.25, 0.1, n) el que nosotros deseemos	quantile(data$columna, quartildeseado)
sd()	Devuelve el valor correspondiente al desvio estandar	sd(data$columna)

* OJO CON ESTO! SI YA LE DIJERON A R ANTES CUAL ES EL DATAFRAME AL QUE HACEN REFERENCIA NO DEBEN VOLVER A ACLARARLO. Por ejemplo:

ej: mutate(data, Promedio = mean(columna)) ✅

ej: mutate(data, Promedio = mean(data$columna)) ❌

group_by() # para agrupar filas

df

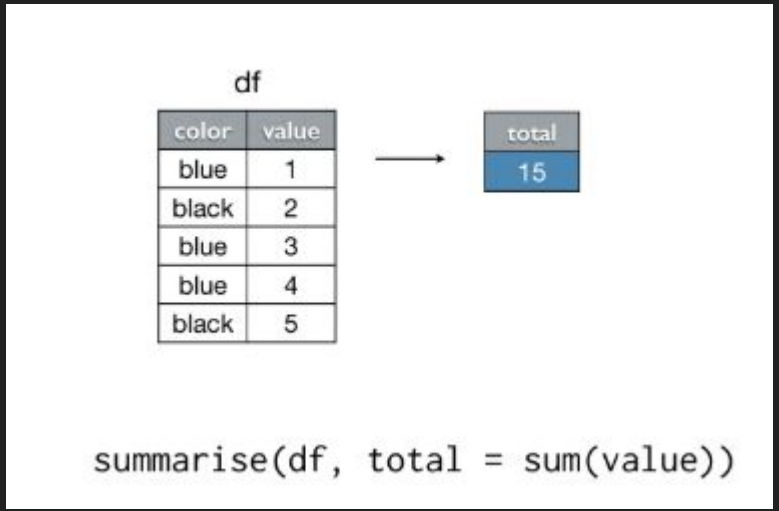
color	value
blue	1
black	2
blue	3
blue	4
black	5

→

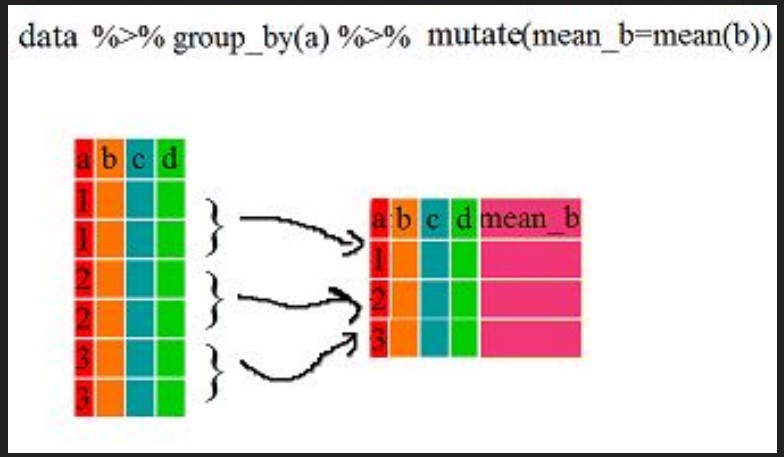
color	total
blue	8
black	7

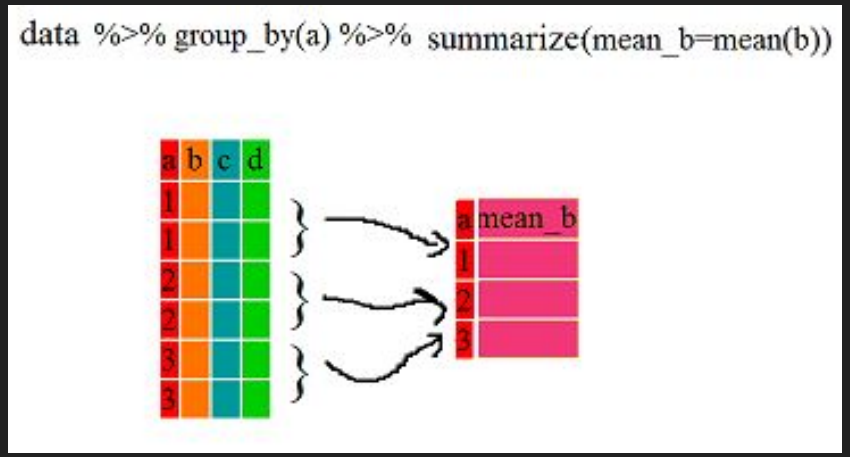
```
by_color <- group_by(df, color)
summarise(by_color, total = sum(value))
```

```
summarise()      # para agregar información de columnnas agrupadas (calcula
                 # 1 sólo valor por grupo)
```



group_by() %>% mutate() VS. group_by() %% summarize()





```
arrange()        # ordena las finlas en base a una o más columnas
```