

# Domar Datos con dplyr and tidyr

## Hoja de Referencia



### Sintaxis - Convenciones útiles para la doma

#### dplyr::tbl\_df(iris)

Convierte datos a una *tbl*. Objetos *tbl* son mas fáciles de inspeccionar que data frames. Ro solo muestra los datos que caben en la pantalla:

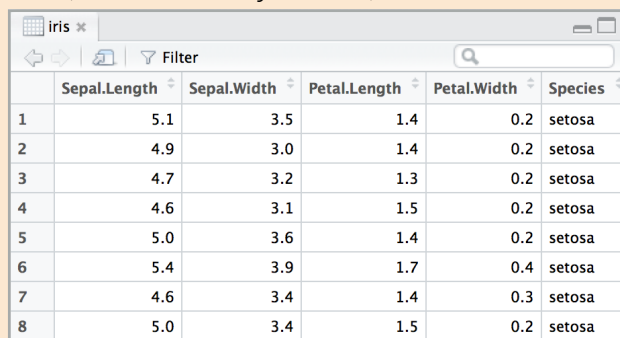
```
Source: local data frame [150 x 5]
  Sepal.Length Sepal.Width Petal.Length
1           5.1          3.5          1.4
2           4.9          3.0          1.4
3           4.7          3.2          1.3
4           4.6          3.1          1.5
5           5.0          3.6          1.4
..          ...          ...          ...
Variables not shown: Petal.Width (dbl),
Species (fctr)
```

#### dplyr::glimpse(iris)

Resumen con mucha información sobre los datos *tbl*.

#### utils::View(iris)

Observa el conjunto de datos en lo que parece una hoja de cálculo (nota la V mayúscula).



	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
7	4.6	3.4	1.4	0.3	setosa
8	5.0	3.4	1.5	0.2	setosa

#### dplyr::%>%

Pasa el objeto a la izquierda al primer argumento (o argumento .) de la función a la derecha creando un tubo.

**x %>% f(y)** es lo mismo que **f(x, y)**

**y %>% f(x, ., z)** es lo mismo que **f(x, y, z)**

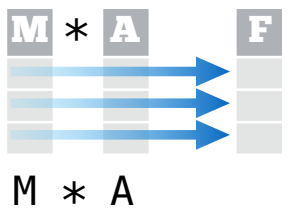
El uso de tubos y tuberías (en Inglés: “Piping”) con %>% resulta en código mas legible. Por ejemplo:

```
iris %>%
  group_by(Species) %>%
  summarise(avg = mean(Sepal.Width)) %>%
  arrange(avg)
```

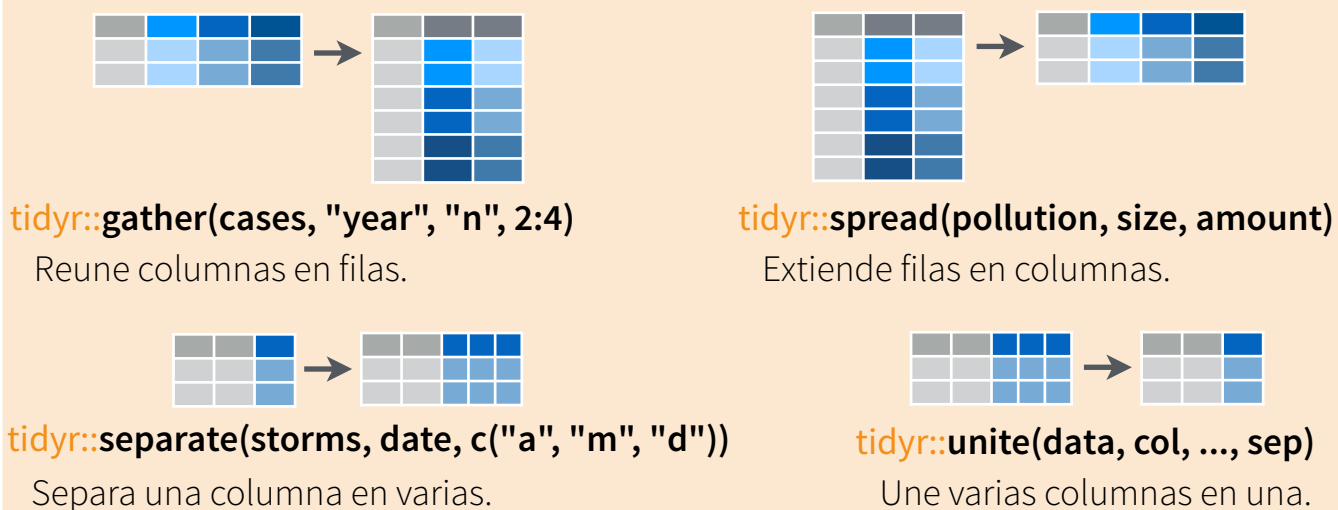
### Datos Ordenados - La base para domar datos en R



Datos ordenados complementan las **operaciones vectorizadas** de R. Automáticamente R preserva observaciones mientras manipulas las variables. No hay otro formato que funciona tan intuitivamente en R.



### Remodelar Datos - Cambia el esquema de los datos



#### dplyr::data\_frame(a = 1:3, b = 4:6)

Combina vectores en un *data frame* (optimizado).

#### dplyr::arrange(mtcars, mpg)

Ordena filas por valores de una columna (bajo a alto).

#### dplyr::arrange(mtcars, desc(mpg))

Ordena filas por valores de una columna (alto a bajo).

#### dplyr::rename(tb, y = year)

Cambia el nombre de columnas de un *data frame*.

### Subconjuntos de Observaciones



#### dplyr::filter(iris, Sepal.Length > 7)

Extrae filas que cumplen criterios lógicos.

#### dplyr::distinct(iris)

Remueve filas duplicadas.

#### dplyr::sample\_frac(iris, 0.5, replace = TRUE)

Selecciona una fracción de filas al azar.

#### dplyr::sample\_n(iris, 10, replace = TRUE)

Selecciona n filas al azar.

#### dplyr::slice(iris, 10:15)

Selecciona filas por posición.

#### dplyr::top\_n(storms, 2, date)

Selecciona y ordena las n entradas mas altas (por grupo si los datos estan agrupados).

### Subconjuntos de Variables



#### dplyr::select(iris, Sepal.Width, Petal.Length, Species)

Selecciona columnas por nombre o funciones de ayuda.

#### Funciones de ayuda para for select - ?select

##### select(iris, contains(" "))

Selecciona columnas cuyos nombres contienen una cadena de caracteres.

##### select(iris, ends\_with("Length"))

Selecciona columnas cuyos nombres terminan con una cadena de caracteres.

##### select(iris, everything())

Selecciona todas las columnas.

##### select(iris, matches(".t."))

Selecciona columnas cuyo nombre cumple con una expresión regular.

##### select(iris, num\_range("x", 1:5))

Selecciona columna con nombres x1, x2, x3, x4, x5.

##### select(iris, one\_of(c("Species", "Genus")))

Selecciona columnas cuyos nombres están en un grupo de nombres.

##### select(iris, starts\_with("Sepal"))

Selecciona columnas cuyos nombres comienzan con una cadena de caracteres.

##### select(iris, Sepal.Length:Petal.Width)

Selecciona todas las columnas entre Sepal.Length and Petal.Width (incluyente).

##### select(iris, -Species)

Selecciona todas las columnas excepto Species.

## Resumir Datos



**dplyr::summarise(iris, avg = mean(Sepal.Length))**

Resume datos a una sola fila de valores.

**dplyr::summarise\_each(iris, funs(mean))**

Aplica la función *summary* a cada columna.

**dplyr::count(iris, Species, wt = Sepal.Length)**

Cuenta el número de valores únicos para cada variable (con o sin ponderación).



*Summarise* usa **funciones de resumen**, funciones que toman un vector de valores y devuelven un solo valor como:

**dplyr::first**

Primer valor de un vector.

**min**

Valor mínimo en un vector.

**dplyr::last**

Último valor de un vector.

**max**

Valor máximo en un vector.

**dplyr::nth**

N-avo valor de un vector.

**mean**

Valor promedio de un vector.

**dplyr::n**

# de valores en u vector.

**median**

Valores mediano en un vector.

**dplyr::n\_distinct**

# valores distintos en un vector

**var**

Varianza de un vector.

**IQR**

IQR de un vector

**sd**

Desviación estándar de un vector.

## Group Data

**dplyr::group\_by(iris, Species)**

Agrupar datos en filas por los valores en Species.

**dplyr::ungroup(iris)**

Remueve la agrupación del data frame.

**iris %>% group\_by(Species) %>% summarise(...)**

Calcula una fila separada con el resumen para cada grupo.



## Crea Nuevas Variables



**dplyr::mutate(iris, sepal = Sepal.Length + Sepal.Width)**

Calcula y añade una o mas columnas nuevas.

**dplyr::mutate\_each(iris, funs(min\_rank))**

Aplica una función de ventana a cada columna.

**dplyr::transmute(iris, sepal = Sepal.Length + Sepal.Width)**

Calcula una o más columnas nuevas, borra columnas originales.



*Mutate* usa **funciones de ventana**, funciones que toman un vector de valores y devuelven otro vector de valores como:

**dplyr::lead**

Copia con valores adelantados por 1.

**dplyr::lag**

Copia con valores atrasados por 1.

**dplyr::dense\_rank**

Rangos sin brechas.

**dplyr::min\_rank**

Rangos. Empates reciben rango min.

**dplyr::percent\_rank**

Rangos con escala del [0, 1].

**dplyr::row\_number**

Rangos. Empates van al primer valor.

**dplyr::ntile**

Separa vector en n baldes.

**dplyr::between**

Los valores están entre a y b?

**dplyr::cume\_dist**

Distribución cumulativa.

**dplyr::cumall**

**all** cumulativo

**dplyr::cumany**

**any** cumulativo

**dplyr::cummean**

**mean** cumulativo

**cumsum**

**sum** cumulativo

**cummax**

**max** cumulativo

**cummin**

**min** cumulativo

**cumprod**

**prod** cumulativo

**pmax**

**max** por elementos

**pmin**

**min** por elementos

## Combina Conjuntos de Datos

a		b		
x1	x2	x1	x3	
A	1	A	T	+
B	2	B	F	
C	3	D	T	

Uniones mutantes

x1	x2	x3
A	1	T
B	2	F
C	3	NA

**dplyr::left\_join(a, b, by = "x1")**

Une filas coincidentes de *b* a *a*.

x1	x3	x2
A	T	1
B	F	2
D	T	NA

**dplyr::right\_join(a, b, by = "x1")**

Une filas coincidentes de *a* a *b*.

x1	x2	x3
A	1	T
B	2	F

**dplyr::inner\_join(a, b, by = "x1")**

Une datos. Mantener solo filas en ambos.

x1	x2	x3
A	1	T
B	2	F
C	3	NA
D	NA	T

**dplyr::full\_join(a, b, by = "x1")**

Une datos. Mantener todos los valores, todas las filas.

Uniones con filtros

x1	x2
A	1
B	2

**dplyr::semi\_join(a, b, by = "x1")**

Todas las filas con coincidencia en *b*.

x1	x2
C	3

**dplyr::anti\_join(a, b, by = "x1")**

Todas las filas sin coincidencia en *b*.

y		z		
x1	x2	x1	x2	
A	1	B	2	+
B	2	C	3	
C	3	D	4	

Operaciones de conjuntos

x1	x2
B	2
C	3

**dplyr::intersect(y, z)**

Filas que aparecen en *y* y *z*.

x1	x2
A	1
B	2
C	3
D	4

**dplyr::union(y, z)**

Filas que aparecen en una o ambas *y* y *z*.

x1	x2
A	1

**dplyr::setdiff(y, z)**

Filas que aparecen en *y* pero no en *z*.

Ligar

x1	x2
A	1
B	2
C	3
B	2
C	3
D	4

**dplyr::bind\_rows(y, z)**

Añade *z* a *y* como nuevas filas

x1	x2	x1	x2
A	1	B	2
B	2	C	3
C	3	D	4

**dplyr::bind\_cols(y, z)**

Añade *z* a *y* como nuevas columnas. Ojo: distribuye filas por posición.