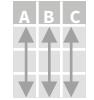


# Transformación de Datos con dplyr :: HOJA DE REFERENCIA



dplyr funciona con conductos y requiere **datos ordenados**.  
En datos ordenados:



Cada **variable** tiene su propia **columna**

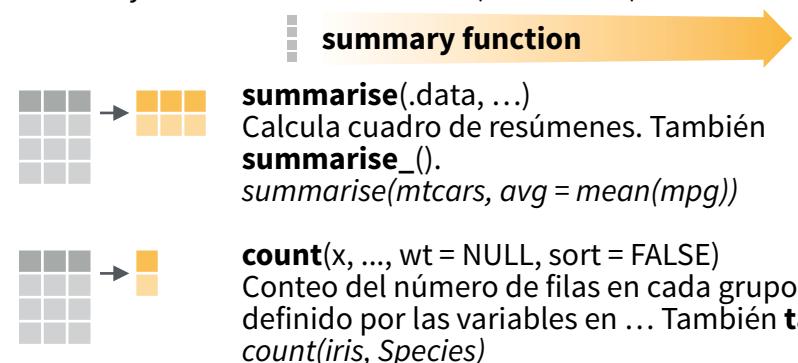


Cada **observación** tiene su propia **fila**



## Resumir Casos

Estos aplican **funciones de resumen** a columnas para crear un nuevo cuadro. Funciones de resumen toman vectores como entrada y devuelven un solo valor (ver reversa).



## VARIACIONES

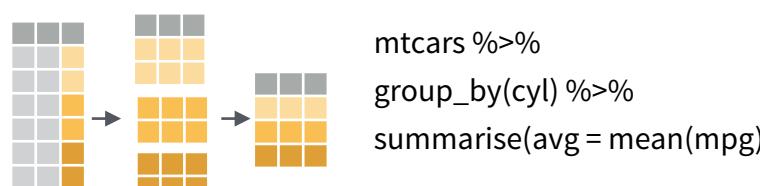
**summarise\_all()** - Aplica funs a cada columna

**summarise\_at()** - Aplica funs a columnas específicas.

**summarise\_if()** - Aplica funs a todas las columnas de un tipo

## Agrupar Casos

Usa **group\_by()** para crear una copia “agrupada” de un cuadro. Funciones dplyr manipularán cada “grupo” por separado para luego combinar los resultados.



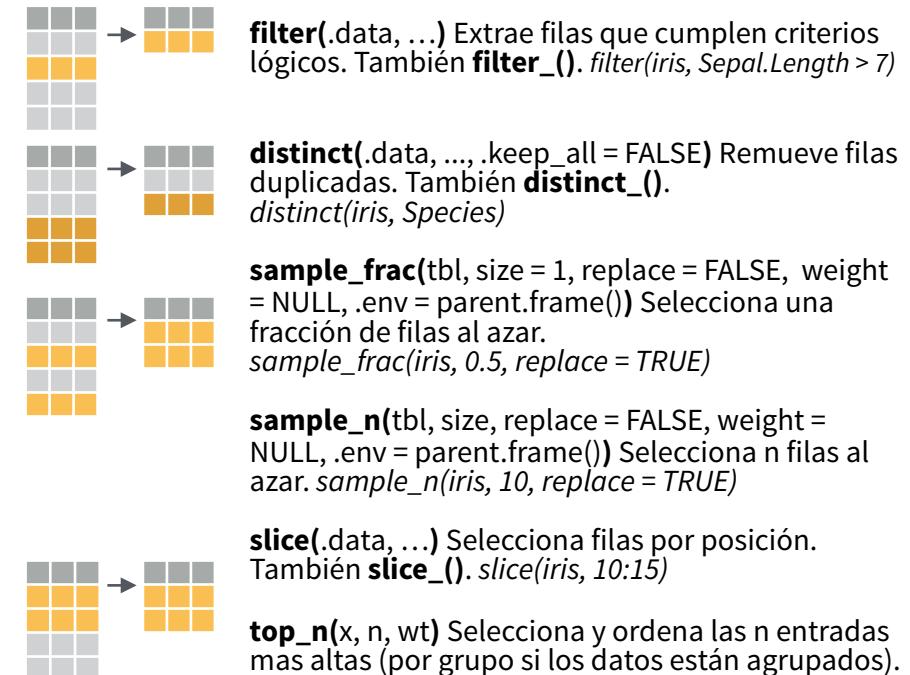
**group\_by(.data, ..., add = FALSE)**  
Devuelve copia del cuadro agrupado por ...  
g\_iris <- group\_by(iris, Species)

**ungroup(x, ...)**  
Devuelve copia no-agrupada del cuadro  
ungroup(g\_iris)

## Manipular Casos

### EXTRAE CASOS

Funciones de Fila devuelven un sub-conjunto de filas como un nuevo cuadro. Usa la variante que termina en \_ para código que funciona con evaluación no-estándar.



### Operadores Lógicos y Booleanos para usar con filter()

<      <=      is.na()      %in%      |      xor()  
>      >=      !is.na()      !      &

Busca **?base::logic** y **?Comparison** para la documentación.

### ORDENA CASOS

**arrange(.data, ...)** Ordena filas por valores de una columna (bajo a alto), usa con **desc()** para ordenar de alto a bajo.  
arrange(mtcars, mpg)  
arrange(mtcars, desc(mpg))

### AÑADE CASOS

**add\_row(.data, ..., .before = NULL, .after = NULL)**  
Añade una o mas filas a un cuadro.  
add\_row(faithful, eruptions = 1, waiting = 1)

## Manipular Variables

### EXTRAER VARIABLES

Funciones de Columnas devuelven un conjunto de columnas como un nuevo cuadro. Usa la variante que termina en \_ para código que funciona con evaluación no-estándar.

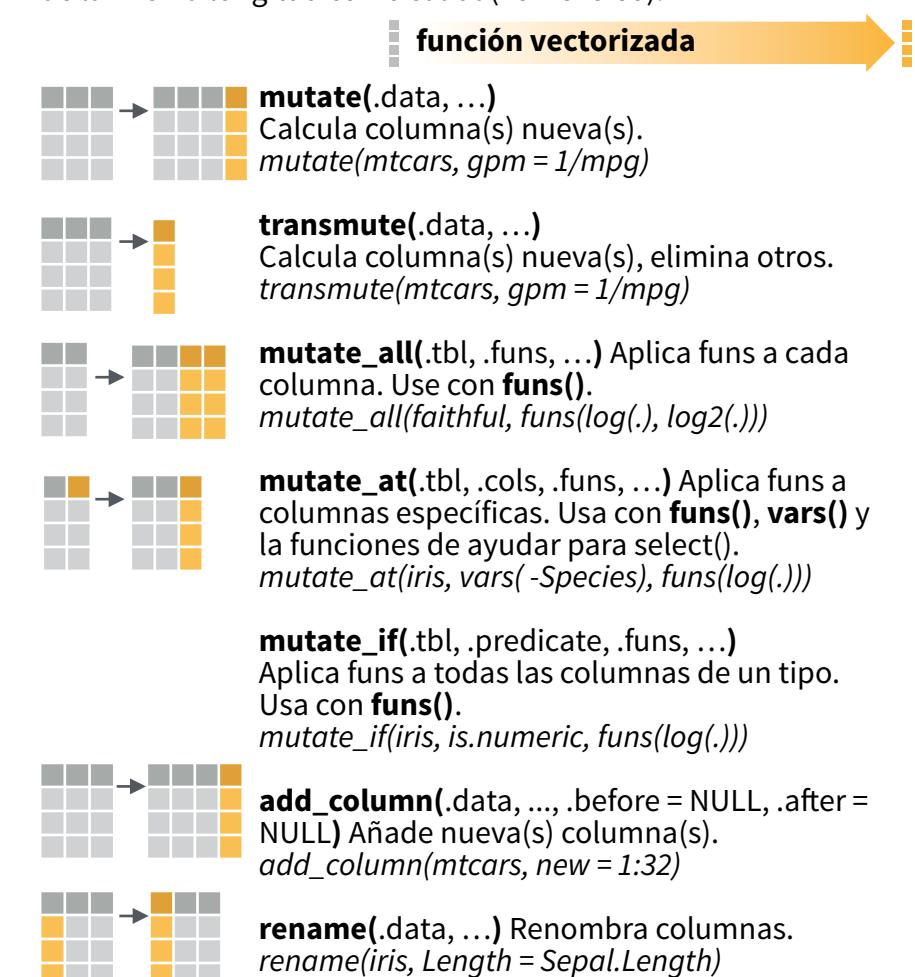
**select(.data, ...)**  
Selecciona columnas por nombre o funciones de ayuda. También **select\_if()**.  
select(iris, Sepal.Length, Species)

Usa estos ayudantes con **select()**, e.g. select(iris, starts\_with("Sepal"))

**contains(match)**      **num\_range(prefix, range)**      :, e.g. mpg:cyl  
**ends\_with(match)**      **one\_of(...)**      -, e.g. -Species  
**matches(match)**      **starts\_with(match)**

### CREA NUEVAS VARIABLES

Estos aplican **funciones vectorizadas** a columnas. Funs vectorizadas toman vectores como entrada y devuelven vectores de la misma longitud como salida (ver reverso).





# Funciones de Vector

## PARA USO CON MUTATE ()

**mutate()** y **transmute()** aplican funciones vectorizadas a columnas para crear nuevas columnas. Funciones vectorizadas toman vectores como entrada y devuelven vectores de la misma longitud.

función vectorizada →

## CONTRARRESTAR

dplyr::lag() - Copia con valores atrasados por 1  
dplyr::lead() - Copia con valores adelantados por 1

## AGREGADOS AAcumulativoS

dplyr::cumall() - all() acumulativo  
dplyr::cumany() - any() acumulativo  
cummax() - max() acumulativo  
dplyr::cummean() - mean() acumulativo  
cummin() - min() acumulativo  
cumprod() - prod() acumulativo  
cumsum() - sum() acumulativo

## RANKINGS

dplyr::cume\_dist() - Proporción de todos los valores <=  
dplyr::dense\_rank() - clasificación con empates = min, sin brechas  
dplyr::min\_rank() - clasificación con empates = min  
dplyr::ntile() - asigna a n intervalos (bins)  
dplyr::percent\_rank() - min\_rank escalado a [0,1]  
dplyr::row\_number() - clasificación con empates = "first" (el primero)

## MATEMATICAS

+, -, \*, /, ^, %/%, %% - ops aritméticas  
log(), log2(), log10() - logs  
<, <=, >, >=, !=, == - comparaciones lógicas

## MISCELÁNEAS

dplyr::between() - x >= izquierda & x <= derecha  
dplyr::case\_when() - casos-multiples if\_else()  
dplyr::coalesce() - primer elemento no-NA por elemento a lo largo de un conjunto de vectores  
dplyr::if\_else() - if() + else() por elemento  
dplyr::na\_if() - reemplaza valores específicos con NA  
pmax() - max() por elemento  
pmin() - min() por elemento  
dplyr::recode() - switch() vectorizado  
dplyr::recode\_factor() - switch() vectorizado para factores

# Funciones de Resumen

## PARA USO CON SUMMARISE ()

**summarise()** aplica funciones de resumen a columnas para crear un nuevo cuadro. Funciones de resumen toman vectores como entrada y devuelven un solo valor.

función de resumen →

## CONTEOS

dplyr::n() - número de valores / filas  
dplyr::n\_distinct() - # de únicos  
sum(!is.na()) - # de no-NA's

## POSICIÓN

mean() - promedio, también **mean(!is.na())**  
median() - mediana

## LÓGICOS

mean() - proporción de TRUE's  
sum() - # de TRUE's

## POSICIÓN/ORDEN

dplyr::first() - primer valor  
dplyr::last() - último valor  
dplyr::nth() - valor en posición n del vector

## RANGO

quantile() - centil n  
min() - valor mínimo  
max() - valor máximo

## PROPAGACIÓN

IQR() - rango inter-centil  
mad() - desviación absoluta media  
sd() - desviación estándar  
var() - varianza

# Nombres Filas

Datos ordenados no usan nombres de filas, que implica un valor fuera de las columnas. Para trabajar con nombres de filas primero muevelos a una columna.

→ **rownames\_to\_column()**  
Mueve nombres de filas a una col.  
a <- rownames\_to\_column(iris, var = "C")

→ **column\_to\_rownames()**  
Mueve columna a nombre de filas.  
column\_to\_rownames(a, var = "C")

También **has\_rownames()**, **remove\_rownames()**

# Combina Cuadros

## COMBINA VARIABLES

X	y	=
A   B   C a   t   1 b   u   2 c   v   3	+   A   B   D a   t   3 b   u   2 d   w   1	=   A   B   C   A   B   D a   t   1   a   t   3 b   u   2   b   u   2 c   v   3   d   w   1

Usa **bind\_cols()** para unir cuadros uno al lado del otro tal como son.

**bind\_cols(...)** Devuelve cuadros posicionados lado a lado como un solo cuadro  
ASEGÚRATE QUE LAS FILAS COINCIDEN.

Usa una "Unión Mutante" para unir un cuadro a columnas de otro cuadro, buscando valores correspondientes en las filas. Cada unión retiene una combinación diferente de los valores de los cuadros.

A   B   C   D	left_join(x, y, by = NULL,
a   t   1   3	copy=FALSE, suffix=c(".x",".y"),...)
b   u   2   2	Une filas coincidentes de y a x.
c   v   3   NA	

A   B   C   D	right_join(x, y, by = NULL, copy =
a   t   1   3	FALSE, suffix=c(".x",".y"),...)
b   u   2   2	Une filas coincidentes de x a y.
d   w   NA   1	

A   B   C   D	inner_join(x, y, by = NULL, copy =
a   t   1   3	FALSE, suffix=c(".x",".y"),...)
b   u   2   2	Une datos. Mantener solo filas en ambos.
c   v   3   NA	

A   B   C   D	full_join(x, y, by = NULL,
a   t   1   3	copy=FALSE, suffix=c(".x",".y"),...)
b   u   2   2	Une datos. Mantener todos los valores, todas las filas.
c   v   3   NA	

Usa **by = c("col1", "col2")** para especificar cuales columnas usar para determinar coincidencias.  
left\_join(x, y, by = "A")

Usa un vector con nombres, **by = c("col1" = "col2")**, para determinar coincidencias en columnas con diferentes nombres en cada conjunto de datos.  
left\_join(x, y, by = c("C" = "D"))

Usa **suffix** para especificar el sufijo para dar a nombres de columnas duplicadas.  
left\_join(x, y, by = c("C" = "D"), suffix = c("1", "2"))

## COMBINA CASOS

X	y	=
A   B   C a   t   1 b   u   2 c   v   3	+   A   B   C C   v   3 d   w   4	=   A   B   C a   t   1 b   u   2 c   v   3 d   w   4

Usa **bind\_rows()** para unir cuadros uno debajo del otro tal como son.

**bind\_rows(..., .id = NULL)**  
Devuelve cuadros uno encima del otro como un solo cuadro. Fija .id a un nombre de columna para añadir una columna con los nombres del cuadro de proveniencia originales (como en la figura)

**intersect(x, y, ...)**  
Filas que aparecen en ambos x y y.

**setdiff(x, y, ...)**  
Filas que aparecen en x pero no en y.

**union(x, y, ...)**  
Filas que aparecen en x o y (removiendo duplicados). union\_all() retiene duplicados.

Usa **setequal()** para probar si dos conjuntos de datos contienen el número exactamente igual de filas (en cualquier orden).

## EXTRAE FILAS

X	y	=
A   B   C a   t   1 b   u   2 c   v   3	+   A   B   D a   t   3 b   u   2 d   w   1	=   A   B   D a   t   3 b   u   2 d   w   1

Usa una "Unión de Filtro" para filtrar un cuadro contra las filas de otro.

**semi\_join(x, y, by = NULL, ...)**  
Devuelve filas de x que coinciden en y.  
ÚTIL PARA VER QUE SE VA A UNIR.

**anti\_join(x, y, by = NULL, ...)**  
Devuelve filas de x que no coinciden en y.  
ÚTIL PARA VER QUE NO SE VA A UNIR.