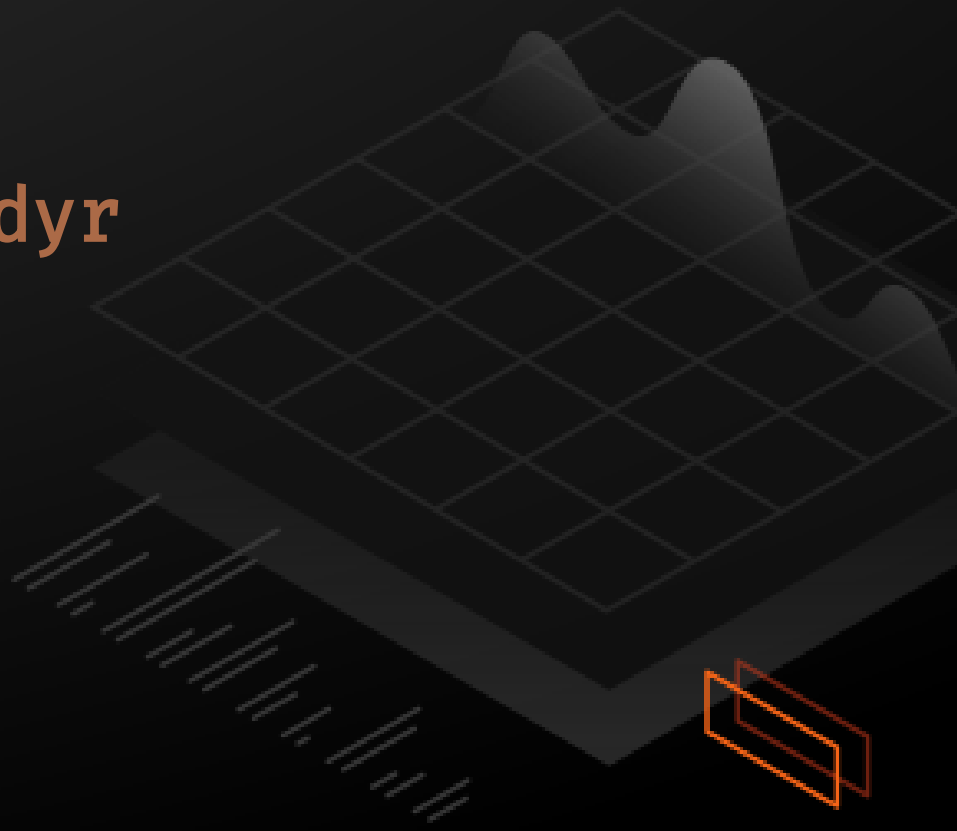
Computación Estadística EPG3308

05 Manipulación de datos `dplyr` `tidyr`



Joshua Kunst
@jbkunst

2023-04-11

Datos ordenados

Datos ordenados

En este capítulo aprenderás una manera consistente para organizar tus datos en R a la que llamaremos tidy data (datos ordenados).

Llevar tus datos a este formato requiere algo de trabajo previo; sin embargo, dicho trabajo tiene retorno positivo en el largo plazo. Una vez que tengas tus datos ordenados y las herramientas para ordenar datos que provee el tidyverse, vas a gastar mucho menos tiempo pasando de una forma de representar datos a otra, lo que te permitirá destinar más tiempo a las preguntas analíticas.

R4DS

Tabla 1

Puedes representar los mismos datos subyacentes de múltiples formas. El ejemplo a continuación muestra los mismos datos organizados de cuatro maneras distintas. Cada dataset muestra los mismos valores de cuatro variables —pais, anio, poblacion y casos (de tuberculosis)—, pero cada uno organiza los valores de forma distinta.

```
library(datos)

tabla1
```

```
## # A tibble: 6 × 4
##   pais        anio  casos  poblacion
##   <chr>      <dbl>  <dbl>      <dbl>
## 1 Afganistán  1999    745   19987071
## 2 Afganistán  2000   2666   20595360
## 3 Brasil      1999  37737  172006362
## 4 Brasil      2000  80488  174504898
## 5 China       1999 212258 1272915272
## 6 China       2000 213766 1280428583
```

Tabla 2

Identifiquemos los valores asociados a pais, anio, poblacion y casos.

```
tabla2
```

```
## # A tibble: 12 × 4
##    pais        anio tipo          cuenta
##    <chr>      <dbl> <chr>          <dbl>
##  1 Afganistán  1999 casos            745
##  2 Afganistán  1999 población    19987071
##  3 Afganistán  2000 casos           2666
##  4 Afganistán  2000 población    20595360
##  5 Brasil      1999 casos           37737
##  6 Brasil      1999 población   172006362
##  7 Brasil      2000 casos           80488
##  8 Brasil      2000 población   174504898
##  9 China       1999 casos          212258
## 10 China       1999 población  1272915272
## 11 China       2000 casos          213766
## 12 China       2000 población  1280428583
```

Tabla 3

Identifiquemos los valores asociados a pais, anio, poblacion y casos.

```
tabla3
```

```
## # A tibble: 6 × 3
##   pais        anio tasa
##   <chr>      <dbl> <chr>
## 1 Afganistán  1999 745/19987071
## 2 Afganistán  2000 2666/20595360
## 3 Brasil      1999 37737/172006362
## 4 Brasil      2000 80488/174504898
## 5 China       1999 212258/1272915272
## 6 China       2000 213766/1280428583
```

Tabla 4a y 4b

Identifiquemos los valores asociados a pais, anio, poblacion y casos.

```
tabla4a
```

```
## # A tibble: 3 × 3
##   pais       `1999` `2000`
##   <chr>       <dbl>  <dbl>
## 1 Afganistán    745   2666
## 2 Brasil      37737  80488
## 3 China      212258 213766
```

```
tabla4b
```

```
## # A tibble: 3 × 3
##   pais           `1999`     `2000`
##   <chr>           <dbl>      <dbl>
## 1 Afganistán   19987071   20595360
## 2 Brasil      172006362  174504898
## 3 China      1272915272 1280428583
```

Varias representaciones/misma información

Las tablas:

- `tabla1`
- `tabla2`
- `tabla3`
- `tabla4a` y `tabla4b`

Son representaciones de los mismos datos subyacentes, pero no todas son igualmente fáciles de usar. Por ejemplo, revisemos nuevamente las tablas y *obtenga la tasa de tuberculosis para cada país/año*.

Un tipo de conjunto de datos, el conjunto de **datos ordenado**, será mucho más fácil de trabajar.

Definición Datos ordenados

Existen tres reglas interrelacionadas que hacen que un conjunto de datos sea ordenado:

- Cada variable debe tener su propia columna.
- Cada observación debe tener su propia fila.
- Cada valor debe tener su propia celda.

Pivotar

Como previamente vimos ciertas tablas poseen los siguientes problemas.

1. Una variable se extiende por varias columnas, por ejemplo `tabla4a`.
2. Una observación está dispersa entre múltiples filas, por ejemplo la `tabla2`.

Para solucionar estos problemas revisaremos dos funciones del paquete **tidyr** (que viene incluído en el paquet **tidyverse**, asi que cargas solamente este último), que son:

- `pivot_longer` que es para pivotar a lo largo.
- `pivot_wider` para pivotar a lo ancho.

Ejemplo `pivot_longer`

```
library(tidyverse)
library(datos)

tabla4a |>
  pivot_longer(
    cols = c(`1999`, `2000`),
    names_to = "anio",
    values_to = "casos"
    )
```

```
## # A tibble: 6 × 3
##   pais        anio   casos
##   <chr>       <chr>  <dbl>
## 1 Afganistán 1999     745
## 2 Afganistán 2000    2666
## 3 Brasil     1999   37737
## 4 Brasil     2000   80488
## 5 China      1999  212258
## 6 China      2000  213766
```

`pivot_longer` Y `pivot_wider` NO SON SIMÉTRICAS

```
# no son perfectanente simétricas
acciones <- tibble(
  anio = c(2015, 2015, 2016, 2016),
  semestre = c(1, 2, 1, 2),
  retorno = c(1.88, 0.59, 0.92, 0.17)
)

acciones %>%
  pivot_wider(
    names_from = anio,
    values_from = retorno
    ) %>%
  pivot_longer(
    cols = `2015`:`2016`,
    names_to = "anio",
    values_to = "retorno"
    )
```

```
## # A tibble: 4 × 3
##   semestre anio  retorno
##      <dbl> <chr>   <dbl>
## 1        1 2015     1.88
## 2        1 2016     0.92
## 3        2 2015     0.59
## 4        2 2016     0.17
```

Ejemplo `separate unite`

```
tabla3 %>%
  separate(
    tasa,
    into = c("casos", "poblacion"),
    convert = TRUE  # convierte al tipo adecuado
  ) |>
  unite(
    nueva_tasa,
    casos, poblacion,
    sep = " dividido "
    )
```

```
## # A tibble: 6 × 3
##   pais         anio nueva_tasa
##   <chr>       <dbl> <chr>
## 1 Afganistán   1999 745 dividido 19987071
## 2 Afganistán   2000 2666 dividido 20595360
## 3 Brasil       1999 37737 dividido 172006362
## 4 Brasil       2000 80488 dividido 174504898
## 5 China        1999 212258 dividido 1272915272
## 6 China        2000 213766 dividido 1280428583
```

Ejercicios `pivot_wider` / `pivot_longer` / `separate`

1. Para cada una de las tablas, realizar las transformaciones necesarias para obtener la tasa de casos de tuberculosis:
    - `tabla1`
    - `tabla2`
    - `tabla3`
    - `tabla4a` y `tabla4b`
2. Para `tabla1` graficar la evolución de la tasa para cada país. ¿Qué observa?

Datos relacionales

Datos relacionales

Es raro que un análisis de datos involucre una única tabla de datos. Lo típico es que tengas muchas tablas que debes combinar para responder a tus preguntas de interés. De manera colectiva, se le llama datos relacionales a esas múltiples tablas de datos, ya que sus relaciones, y no solo los conjuntos de datos individuales, son importantes.

Las relaciones siempre se definen sobre un par de tablas. Todas las otras relaciones se construyen sobre esta idea simple: las relaciones entre tres o más tablas son siempre una propiedad de las relaciones entre cada par. ¡A veces ambos elementos de un par pueden ser la misma tabla! Esto se necesita si, por ejemplo, tienes una tabla de personas y cada persona tiene una referencia a sus padres.

Lo anterior viene de datos relacionales de R4DS.

Tipos de opearciones entre 2 tablas

Existen distintas formas de operar con 2 data frames:

- **Uniones de transformación** (del inglés *mutating joins*), que agregan nuevas variables a un data frame a partir de las observaciones coincidentes en otra tabla.
- **Uniones de filtro** (del inglés *filtering joins*), que filtran observaciones en un data frame con base en si coinciden o no con una observación de otra tabla.
- **Operaciones de conjuntos** (del inglés *set operations*), que tratan las observaciones como elementos de un conjunto.

Los *mutating joins* (en adelante *joins*) serán los más usuales y los que estudiaremos con más detalle.

De forma visual

Un poco más ejemplificado

ID	X1
1	a1
2	a2

ID	X2
2	b1
3	b2

inner_join

ID	X1	X2
2	a2	b1

left_join

ID	X1	X2
1	a1	NA
2	a2	b1

right_join

ID	X1	X2
2	a2	b1
3	NA	b2

full_join

ID	X1	X2
1	a1	NA
2	a2	b1
3	NA	b2

semi_join

ID	X1
2	a2

anti_join

ID	X1
1	a1

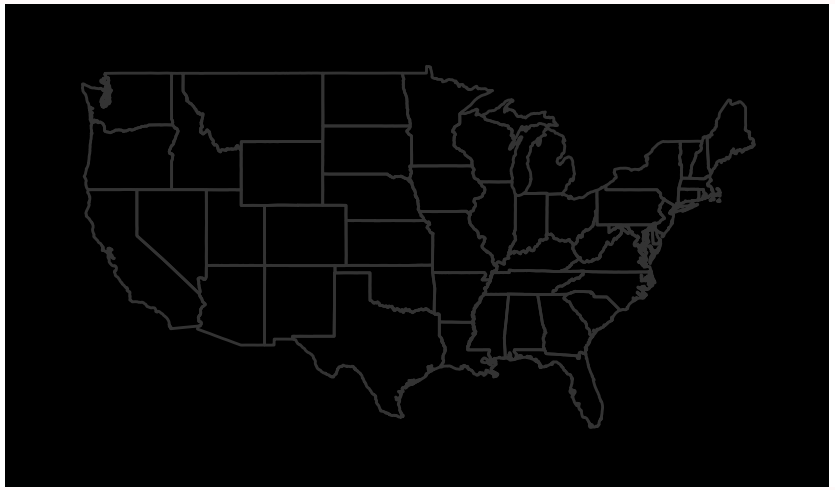
Ejemplo `full_join`

```
library(tidyverse)

band_members

band_instruments

full_join(
  band_members,
  band_instruments,
  by = "name"
  )
```

```
## # A tibble: 3 × 2
##   name  band
##   <chr> <chr>
## 1 Mick  Stones
## 2 John  Beatles
## 3 Paul  Beatles

## # A tibble: 3 × 2
##   name  plays
##   <chr> <chr>
## 1 John  guitar
## 2 Paul  bass
## 3 Keith guitar

## # A tibble: 4 × 3
##   name  band    plays
##   <chr> <chr>   <chr>
## 1 Mick  Stones  <NA>
## 2 John  Beatles guitar
## 3 Paul  Beatles bass
## 4 Keith <NA>    guitar
```

Hint

```
library(tidyverse)
library(ggraph)

states <- map_data("state")

ggplot() +
  geom_polygon(
    data = states,
    aes(long, lat, group = group),
    col = "gray20",
    linewidth = 1,
    fill = NA
    ) +
  coord_map() +
  ggraph::theme_graph(background = "black")
```



Más información sobre transformación de datos

- Datos ordenados en R4DS.
- Buenos artículos de `tidyr` en la página de documentación.