



Tidyydata y manipulación de datos

Sesión N° 5

05 septiembre 2021

Análisis de datos estadísticos en R

Profesora Valentina Andrade de la Horra
Ayudantes Dafne Jaime y Nicolás Godoy

Contenidos Sesión 5



Estructura de datos

Cálculos agregados

Tidy data

Combinar dara



1: Estructura de datos

1: Estructura de datos



- Hasta ahora hemos asumido una estructura de datos *ad hoc* a los análisis que queremos realizar
- No siempre los datos vendrán "limpios" o con la estructura columna-fila que necesitamos
- No siempre nuestros datos vendrán completos y necesitaremos "unir" distintas fuentes de información

Estructura de los datos



- También ocurrirá que vamos a querer hacer cálculos agregados
- Algunos agregando por cada observación o cada columna

Pensemos algunos ejemplos... 1. 2. 3.



Creación de variables agregadas

rowwise() para agrupar filas



- Construcción de índices para cada observación
- Sumativos y promedio

```
datos %>% #Especificamos que trabajaremos con el dataframe dat  
  rowwise() %>% #Especificamos que agruparemos por filas  
  mutate(ing_tot = sum(ss_t, svar_t, reg_t)) #Creamos una nuev
```

group_by() para agrupar columnas



- Cálculos en base a características de una o varias columnas
- Calcular el promedio de edad según comuna (`group_by()` + `mutate()`)
- Calcular una nueva variable que diga cuántas mujeres tengo en mis datos (`group_by()` + `summarise()`)



```
datos %>%  
  group_by(sexo) %>% #Especificamos que agruparemos por sexo  
  summarise(media = mean(ing_tot)) #Creamos una columna llamada
```



¡Vamos a practicar!

¿Dónde?



Descargar el zip del sesión 5 el sitio del curso

1. Recursos de la práctica



- **Datos:** Encuesta Suplementaria de Ingresos (ESI) en su versión 2020.
- **Libro de códigos.**

Tarea Bonus



- Con CASEN 2020 calcular
 - Cuántas personas en la muestra son de FONASA
 - Cuántas personas en la muestra son Mujeres y de FONASA
 - Replicar los cálculos de ingresos del hogar y contrastar con la variable original obtenida

Entrega: próximo lunes 13 de septiembre



Tidydata



Paquete tidy

tidyr para solucionar los problemas de estructura de datos



- La estructura "**limpia**" considera a las observaciones en las filas y las variables en las columnas
- Esto no siempre podrá ser así.

country	year	cases	population
Afghanistan	1999	7745	19987071
Afghanistan	2000	2666	2059360
Brazil	1999	37737	172006362
Brazil	2000	80488	174504898
China	1999	212258	1272915272
China	2000	216766	128042583

variables

country	year	cases	population
Afghanistan	1999	7745	19987071
Afghanistan	2000	2666	2059360
Brazil	1999	37737	172006362
Brazil	2000	80488	174504898
China	1999	212258	1272915272
China	2000	216766	128042583

observations

country	year	cases	population
Afghanistan	99	7745	19987071
Afghanistan	00	2666	2059360
Brazil	99	37737	172006362
Brazil	00	80488	174504898
China	99	212258	1272915272
China	00	216766	128042583

values



Las tres reglas de un buen dataset



tidyr

1. Cada variable tiene que estar su propia columna
2. Cada observación tiene que estar en su propia fila
3. Cada valor tiene que estar en su propia celda.

¿Cumple con las reglas?



```
#> # A tibble: 3 x 3
#>   country      `1999`  `2000`
#> * <chr>      <int>   <int>
#> 1 Afghanistan    745    2666
#> 2 Brazil        37737  80488
#> 3 China         212258 213766
```

- ¿Qué problema de procesamiento nos podría producir?



```
#> # A tibble: 6 x 3
#>   country      year  cases
#>   <chr>        <chr> <int>
#> 1 Afghanistan 1999     745
#> 2 Afghanistan 2000    2666
#> 3 Brazil       1999   37737
#> 4 Brazil       2000   80488
#> 5 China        1999  212258
#> 6 China        2000  213766
```

`pivot_*`



- Para hacer ese paso de filas a columnas (o viceversa) se hace un procedimiento que se llama *pivote*
- `pivot_longer()` y `pivot_wider()` (son los inversos)

Wide format



- Pensemos en el primer ejemplo donde a1, a2 y a3 son los años

ID	a1	a2	a3
1			
2			
3			

wide format

Long format



ID	key	value
1	a1	
2	a1	
3	a1	
1	a2	
2	a2	
3	a2	
1	a3	
2	a3	
3	a3	

long format

¿Cómo ocupar pivot_longer?



```
table %>%  
  pivot_longer(c(`1999`, `2000`), names_to = "year", values_to = "cases")  
#> # A tibble: 6 x 3  
#>   country      year      cases  
#>   <chr>      <chr>   <int>  
#> 1 Afghanistan 1999       745  
#> 2 Afghanistan 2000      2666  
#> 3 Brazil      1999     37737  
#> 4 Brazil      2000    80488  
#> 5 China       1999   212258  
#> 6 China       2000  213766
```




country	year	cases
Afghanistan	1999	745
Afghanistan	2000	2666
Brazil	1999	37737
Brazil	2000	80488
China	1999	212258
China	2000	213766

country	1999	2000
Afghanistan	745	2666
Brazil	37737	80488
China	212258	213766

table4

Cómo ocupar pivot wider



```
table2
#> # A tibble: 12 x 4
#>   country      year type      count
#>   <chr>      <int> <chr>    <int>
#> 1 Afghanistan 1999 cases      745
#> 2 Afghanistan 1999 population 19987071
#> 3 Afghanistan 2000 cases      2666
#> 4 Afghanistan 2000 population 20595360
#> 5 Brazil      1999 cases      37737
#> 6 Brazil      1999 population 172006362
#> # ... with 6 more rows
```



```
table2 %>%  
  pivot_wider(names_from = type, values_from = count)  
#> # A tibble: 6 x 4  
#>   country      year  cases population  
#>   <chr>      <int> <int>      <int>  
#> 1 Afghanistan 1999     745 19987071  
#> 2 Afghanistan 2000    2666 20595360  
#> 3 Brazil      1999   37737 172006362  
#> 4 Brazil      2000   80488 174504898  
#> 5 China       1999  212258 1272915272  
#> 6 China       2000  213766 1280428583
```



country	year	key	value
Afghanistan	1999	cases	745
Afghanistan	1999	population	19987071
Afghanistan	2000	cases	2666
Afghanistan	2000	population	20595360
Brazil	1999	cases	37737
Brazil	1999	population	172006362
Brazil	2000	cases	80488
Brazil	2000	population	174504898
China	1999	cases	212258
China	1999	population	1272915272
China	2000	cases	213766
China	2000	population	1280428583

table2

country	year	cases	population
Afghanistan	1999	745	19987071
Afghanistan	2000	2666	20595360
Brazil	1999	37737	172006362
Brazil	2000	80488	174504898
China	1999	212258	1272915272
China	2000	213766	1280428583



**¡Vamos a practicar
pivotes en tidyrr!**

¿Qué pasa si la regla Nº3 no se cumple?

`separate()` y `unite`

Columnas con información de más de una variable



```
table3
```

```
#> # A tibble: 6 x 3
#>   country      year rate
#> * <chr>      <int> <chr>
#> 1 Afghanistan 1999 745/19987071
#> 2 Afghanistan 2000 2666/20595360
#> 3 Brazil      1999 37737/172006362
#> 4 Brazil      2000 80488/174504898
#> 5 China       1999 212258/1272915272
#> 6 China       2000 213766/1280428583
```



```
table3 %>%
  separate(rate, into = c("cases", "population"))
#> # A tibble: 6 x 4
#>   country      year cases  population
#>   <chr>      <int> <chr>   <chr>
#> 1 Afghanistan 1999  745    19987071
#> 2 Afghanistan 2000 2666    20595360
#> 3 Brazil      1999 37737   172006362
#> 4 Brazil      2000 80488   174504898
#> 5 China       1999 212258  1272915272
#> 6 China       2000 213766  1280428583
```




country	year	rate
Afghanistan	1999	745 / 19987071
Afghanistan	2000	2666 / 20595360
Brazil	1999	37737 / 172006362
Brazil	2000	80488 / 174504898
China	1999	212258 / 1272915272
China	2000	213766 / 1280428583

table3

country	year	cases	population
Afghanistan	1999	745	19987071
Afghanistan	2000	2666	20595360
Brazil	1999	37737	172006362
Brazil	2000	80488	174504898
China	1999	212258	1272915272
China	2000	213766	1280428583

separate() y algunos argumentos adicionales



- Especificar el separador (sep =)

```
table3 %>%  
  separate(rate, into = c("cases", "population"), sep = "/")
```

- Especificar si queremos que se cambie la clase de la variable

```
table3 %>%  
  separate(rate, into = c("cases", "population"), convert = TRUE)  
#> # A tibble: 6 x 4
```

unite() el proceso inverso



country	year	rate
Afghanistan	1999	745 / 19987071
Afghanistan	2000	2666 / 20595360
Brazil	1999	37737 / 172006362
Brazil	2000	80488 / 174504898
China	1999	212258 / 1272915272
China	2000	213766 / 1280428583

country	century	year	rate
Afghanistan	19	99	745 / 19987071
Afghanistan	20	0	2666 / 20595360
Brazil	19	99	37737 / 172006362
Brazil	20	0	80488 / 174504898
China	19	99	212258 / 1272915272
China	20	0	213766 / 1280428583

table6

unite()



- Será muy útil para construir variables "combinatorias"

```
table5 %>%  
  unite(new, century, year, sep = "'")  
#> # A tibble: 6 x 3  
#>   country      new      rate  
#>   <chr>      <chr> <chr>  
#> 1 Afghanistan 1999 745/19987071  
#> 2 Afghanistan 2000 2666/20595360  
#> 3 Brazil      1999 37737/172006362  
#> 4 Brazil      2000 80488/174504898  
#> 5 China       1999 212258/1272915272  
#> 6 China       2000 213766/1280428583
```

¡Vamos a practicar con ESI!



Unir datos



Unir datos



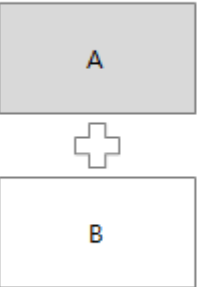

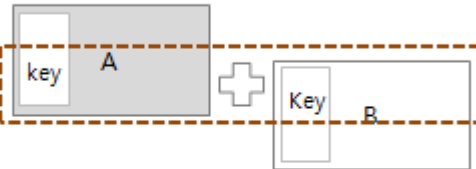
- No es una tarea fácil unir un set de datos x e y

Recomendaciones

1. Conocer bien los datos x e y
2. Verifica las unidades de observación
3. Define tu variable "llave" o **key**
4. Define qué procedimiento quieres hacer

4. Procedimientos de unión

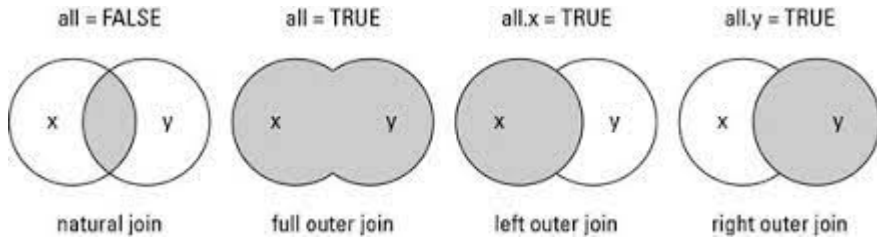


rbind(A, B)	cbind(A, B)	merge(A, B, by='key')
 <p>행 결합</p>	 <p>열 결합</p>	 <p>동일 key 값 기준 결합</p> <p>http://rfriend.tistory.com</p>

merge()



```
merge(a, b, by = "key")
```



bind_cols y bind_rows



- `bind_*` para "pegar" o columnas o filas

En síntesis



Estructura de datos

Cálculos agregados

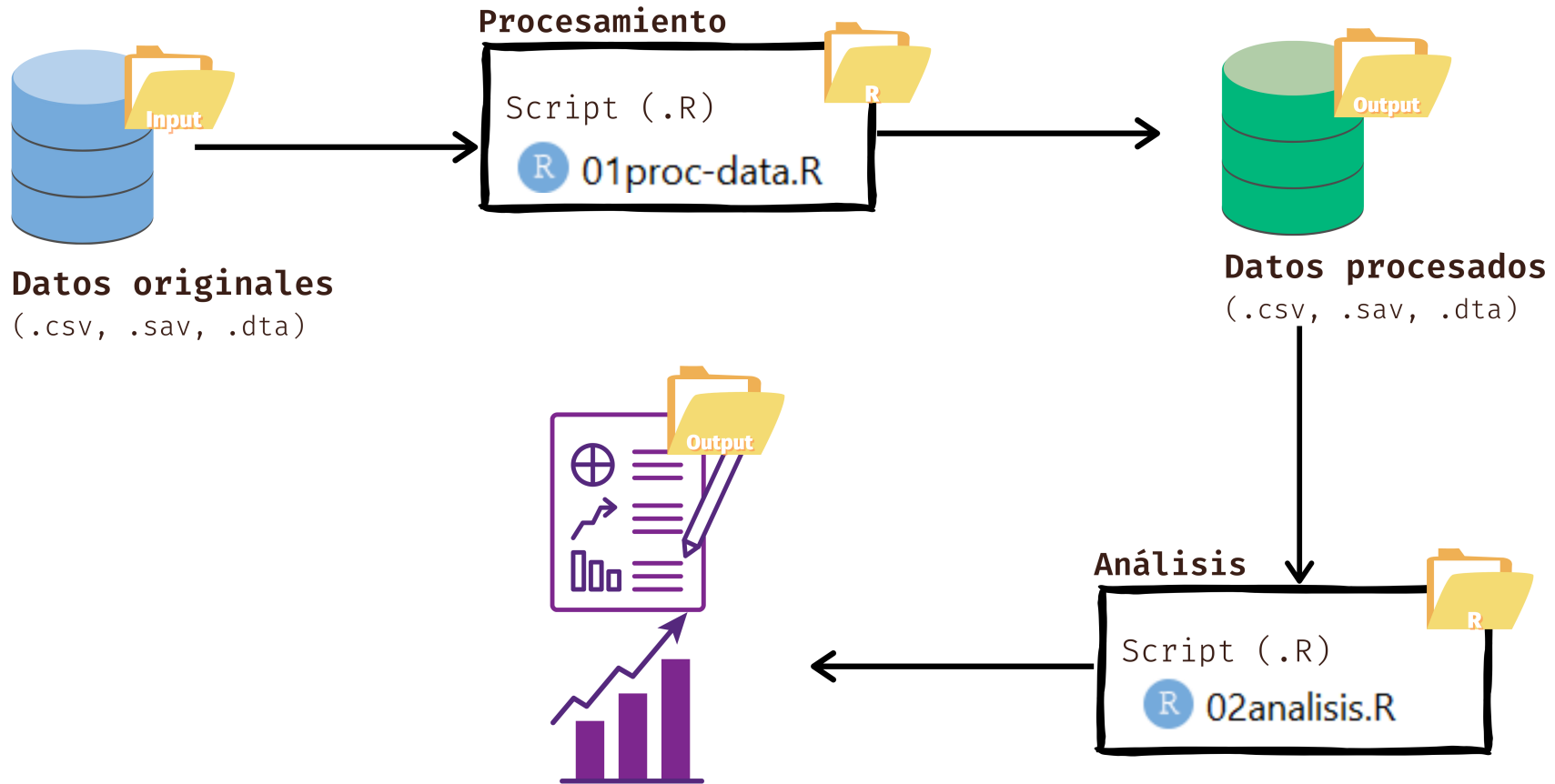
Tidy data

Combinar data

¡Y a no olvidar el flujo para el análisis!



Nos permite hacernos amigas/os más rápido del programa



¿Y eso era?





Tidyydata y manipulación de datos

Sesión N° 5

05 septiembre 2021

Análisis de datos estadísticos en R

Profesora Valentina Andrade de la Horra
Ayudantes Dafne Jaime y Nicolás Godoy