OPSO79-1-UCSH2021

Operador pipe, profundización dplyr y buena codificación. Bloque práctico (7b)

01/10/2021

Pipe

comando para concatenar funciones

Operador pipe

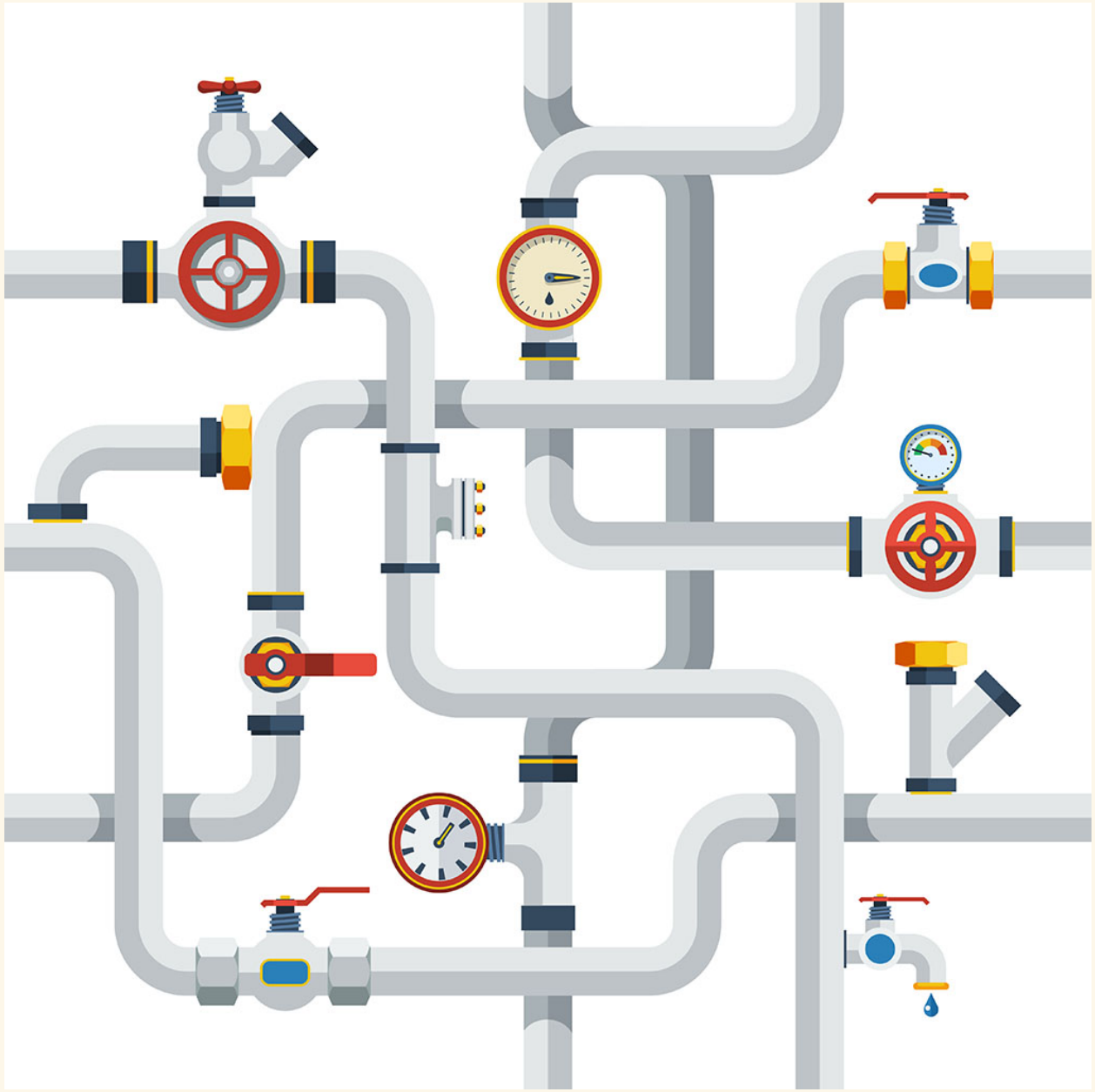
Significa "tubo", "tubería" o "cañería".

Es un operador de `magrittr` que se combina con los verbos de `dplyr`

(si cargamos `dplyr` ya podemos usarlo).

Se escribe %>% (ctrl + shift + m)

Nos permite concatenar funciones, haciendo más sencilla la lectura del código.



Operador pipe

¿Como saber los nombres de las variables de la base de datos `guaguas`?

```
names(guaguas)   ## la manera "tradicional" o R base
```

```
## [1] "anio"       "nombre"     "sexo"       "n"          "proporcion"
```

Con pipes sería así:

```
guaguas %>% names()  ## el objeto primero, luego la función
```

```
## [1] "anio"       "nombre"     "sexo"       "n"          "proporcion"
```

Operador pipe

¿Cuál es la importancia relativa del nombre María en 1920, 1950, 1980 y 2020?

```
filter(guaguas,
       (anio==1920|anio==1950|anio==1980|anio==2020) & nombre=="María")
```

```
## # A tibble: 4 x 5
##    anio nombre sexo      n proporcion
##   <dbl> <chr>  <chr> <dbl>      <dbl>
## 1  1920 María  F      2130    0.104
## 2  1950 María  F     15023    0.0859
## 3  1980 María  F     10382    0.0365
## 4  2020 María  F      1345    0.00690
```

Con pipes sería así:

```
guaguas %>%
  filter( (anio==1920|anio==1950|anio==1980|anio==2020) &
            nombre=="María")
```

Operador pipe

¿Y cuál es la gracia de la tubería?

Con pipes podemos concatenar funciones.

¿Cuáles son los años en los que el nombre María es más importante?, ¿Que porcentajes representa el nombre María?

Sin pipes tendríamos que hacer algo más o menos así (con el riesgo de confundir paréntesis)

```
select(mutate(head(arrange(
  filter(guaguas,nombre=="María"),
  -proporcion),4),porcentaje=round(proporcion*100,1)),
  anio,nombre,porcentaje)
```

Se dificulta bastante la lectura (**desde el centro hacia afuera**)

Operador pipe

Con pipes es como leer, de izquierda a derecha:

(y de arriba hacia abajo opcional)

```
guaguas %>%
  filter(nombre=="María") %>%
  arrange(-proporcion) %>%
  head(4) %>%
  mutate(porcentaje=round(proporcion*100,1)) %>%
  select(anio,nombre,porcentaje)
```

```
## # A tibble: 4 x 3
##    anio nombre porcentaje
##   <dbl> <chr>       <dbl>
## 1  1920 María        10.4
## 2  1921 María         9.2
## 3  1922 María         9
## 4  1948 María         8.8
```

Todo es una sola línea de código. No tuve que crear ningún objeto extra.

Continuación encuestas en Chile

aplicando operador *pipe*

Encuesta Laboral (Encla)

La encuesta busca diagnosticar el estado y evolución de las **condiciones de empleo y trabajo**, de las **relaciones laborales** y de la **igualdad de género** en las empresas regidas por el Código del Trabajo en Chile.

Su población objetivo corresponde a las empresas formales vigentes con cinco o más trabajadores contratados directamente.

Cada empresa cuenta con tres unidades de observación (informante). Informantes en microdatos diferentes:

- Empleadores (gerente RRHH)
- Autoaplicado (gerente RRHH)
- Trabajadores (trabajador aleatorio)
- Sindicatos (dirigente sindical)
- El año 2019 se aplicó la última versión: 3.670 empresas, 4 microdato.

Encuesta Laboral (Encla)

Solo descarguemos la data de sindicatos (esta vez en formato `stata`).

La lógica será la misma que en spss:

- guardar en carpeta donde esté el R Project con el que estamos trabajando.
- Cargar paquete y llamar función `read_dta()`

```
library(haven); library(dplyr)
encla <- read_dta("data/bbdd-sindicatos-bp.dta")
```

Veamos las primeras 4 columnas y 2 filas

```
encla %>% select(1:4) %>% slice(1:2)
```

```
## # A tibble: 2 x 4
##     id_bp        a1        a2        a3
##     <dbl> <dbl+lbl> <dbl+lbl> <dbl+lbl>
## 1 4554856    2 [No]   NA             NA
## 2 7550575    1 [Sí]    2 [No]        NA
```

Encuesta Laboral (Encla)

Para explorar la data usemos `sjmisc` y `sjlabelled`

(ambos paquetes de Daniel Lüdecke)

```
library(sjmisc)
#install.packages("sjlabelled")
library(sjlabelled)
```

Para ver etiqueta de **todas** las variables:

```
get_label(encla)
```

Encuesta Laboral (Encla)

Para buscar palabras o conceptos en data:

```
find_var(encla,"conflicto")
```

```
##                                                                                v
## 1  h2_1. Durante el año 2018, ¿Qué motivó el(los) conflicto(s) en esta empr
## 2  h2_2. Durante el año 2018, ¿Qué motivó el(los) conflicto(s) en esta empr
## 3  h2_3. Durante el año 2018, ¿Qué motivó el(los) conflicto(s) en esta empr
## 4  h2_4. Durante el año 2018, ¿Qué motivó el(los) conflicto(s) en esta empr
## 5  h2_5. Durante el año 2018, ¿Qué motivó el(los) conflicto(s) en esta empr
## 6  h2_6. Durante el año 2018, ¿Qué motivó el(los) conflicto(s) en esta empr
## 7  h2_7. Durante el año 2018, ¿Qué motivó el(los) conflicto(s) en esta empr
## 8  h2_8. Durante el año 2018, ¿Qué motivó el(los) conflicto(s) en esta empr
## 9  h2_9. Durante el año 2018, ¿Qué motivó el(los) conflicto(s) en esta empr
## 10 h2_77. Durante el año 2018, ¿Qué motivó el(los) conflicto(s) en esta emp
## 11 h2_77_otra_c. Durante el año 2018, ¿Qué motivó el(los) conflicto(s) en e
## 12 h4. El o los conflictos mencionados ocurridos durante el año 2018, tuvie
```

Encuesta Laboral (Encla)

ATENCIÓN: *Las preguntas H2, H3 y H4 solo deben responderse en caso de que el informante haya señalado la ocurrencia de uno o más conflictos en la empresa durante el año 2018.*

H2 **Durante el año 2018, ¿qué motivó el(los) conflicto(s) en esta empresa?**

Motivos		Sí	No	NS	NR
01	Problemas de trato por parte de los superiores	1	2	88	99
02	Causas salariales	1	2	88	99
03	La organización del trabajo (jornadas, horas extras, cambio de funciones, etc.)	1	2	88	99
04	Cambios asociados a procesos de automatización de la empresa	1	2	88	99
05	Despidos o reducciones de personal	1	2	88	99
06	Incumplimiento de la normativa laboral	1	2	88	99
07	Malas relaciones de la empresa con el sindicato	1	2	88	99
08	Prácticas desleales o antisindicales durante la negociación colectiva	1	2	88	99
09	No se llegó a acuerdo durante la negociación colectiva	1	2	88	99
77	Otra. Especifique:	1	2	88	99

Encuesta Laboral (Encla)

Etiqueta de variable:

```
get_label(encla$h2_1)  # Pregunta o enunciado
```

```
get_labels(encla$h2_1) # Categorías de respuesta
```

Distribución de la variable con `frq(encla$h2_1)`

```
##
## h2_1. Durante el año 2018, ¿Qué motivó el(los) conflicto(s) en esta empresa
## # total N=1172  valid N=517  mean=3.97  sd=15.01
##
## Value |         Label |   N | Raw % | Valid % | Cum. %
## ------------------------------------------------------
##     1 |            Sí | 263 | 22.44 |   50.87 |  50.87
##     2 |            No | 240 | 20.48 |   46.42 |  97.29
##    88 |       No Sabe |   4 |  0.34 |    0.77 |  98.07
##    96 | Valor perdido |  10 |  0.85 |    1.93 | 100.00
##    99 |   No Responde |   0 |  0.00 |    0.00 | 100.00
##  <NA> |          <NA> | 655 | 55.89 |    <NA> |   <NA>
```

Encuesta Laboral (Encla)

Ahora con una variable cuantitativa

```
get_label(encla$g2_3)
```

```
## [1] "g2_3. ¿Cuántos hombres y mujeres se encuentran afiliados a su sindicat
```

Descripción con `R base`

```
summary(encla$g2_3)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     1.0    50.0   110.5   278.4   241.2 20000.0
```

Descripción con `sjmisc`

```
descr(encla$g2_3)
```

Encuesta Laboral (Encla)

¿Cuántos socios/as hay en promedio por sector económico?

```
table(encla$agrupacion_actividad)
```

```
##
##   1   2   3   4   5   6   7   8   9  10  11  12  13
##  33  29 162  50  45 164 131  45  66 139 169  96  43
```

```
get_labels(encla$agrupacion_actividad)
```

```
##  [1] "Agricultura, ganadería, silvicultura y pesca"
##  [2] "Explotación de minas y canteras"
##  [3] "Industrias manufactureras"
##  [4] "Suministro de electricidad, gas, vapor y aire acondicionado; y Sumini
##  [5] "Construcción"
##  [6] "Comercio al por mayor y al por menor; Reparación de vehículos automot
##  [7] "Transporte y almacenamiento; información y comunicaciones"
##  [8] "Actividades de alojamiento y de servicio de comidas"
##  [9] "Actividades financieras y de seguros; Actividades inmobiliarias"
## [10] "Actividades profesionales, científicas y técnicas; Actividades de ser
## [11] "Enseñanza"
```

Encuesta Laboral (Encla)

A esta altura ya somos conscientes de que en R hay muchas formas de hacer las cosas.

La forma manual sería la más intuitiva

Que sería como hacer esto para cada valor de `agrupacion_actividad`

```
encla %>% filter(agrupacion_actividad==1) %>%
  select(g2_3) %>% summary()
```

```
##       g2_3
##  Min.   : 10.00
##  1st Qu.: 25.00
##  Median : 49.00
##  Mean   : 94.06
##  3rd Qu.:120.00
##  Max.   :350.00
```

Encuesta Laboral (Encla)

¿El problema de esta forma?

- Poca economía en el código (muchas líneas)
- posibilidad de cometer errores
- dificultad para combinar cada `summary()`
- Desgastante para variables con muchas categorías (e.g. comuna)

Por suerte, el paquete dplyr nos trae una solución con group_by()

La variable de sector económico la definimos como la que establece los grupos.

Para cada grupo queremos un estadístico particular (media y mediana, por ejemplo)

Encuesta Laboral (Encla)

```
encla %>%
  group_by(agrupacion_actividad) %>%
  summarise(media=mean(g2_3),mediana=median(g2_3))
```

```
##    agrupacion_actividad     media mediana
## 1                     1  94.06061    49.0
## 2                     2 317.55172   208.0
## 3                     3 241.94444   100.0
## 4                     4 168.38000    99.0
## 5                     5 345.82222    79.0
## 6                     6 410.87195   109.5
## 7                     7 196.11450   110.0
## 8                     8 322.64444    95.0
## 9                     9 569.18182   282.5
## 10                   10 184.64748   124.0
## 11                   11 256.01183   107.0
## 12                   12 323.78125   168.5
## 13                   13 132.06977   104.0
```

Comprendamos la función...

Profundización *dplyr*:

`group_by()`, `if_else()` y `case_when()`

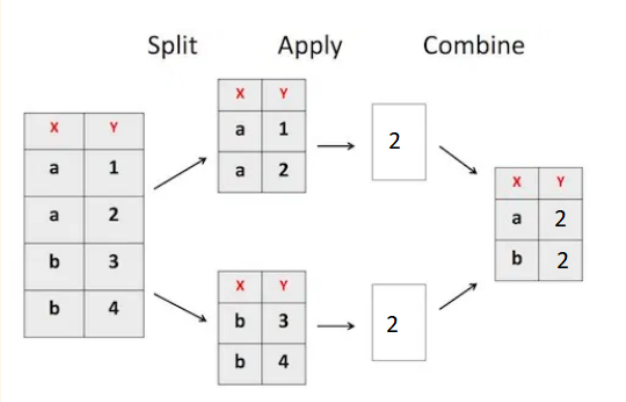
group_by()

Estrategia split-apply-combine.

Esta estrategia sucede tras bambalinas (no la vemos). Solo observamos el resultados.

Con group_by()

- se divide el data frame en x grupos (x es el número de categorías de la variable).
- Luego se opera una función en cada grupo
- Finalmente se combinan los resultados alcanzados en cada grupo.



group_by()

Vimos la función `group_by()` para medias:

```
encla %>%
  group_by(agrupacion_actividad) %>%
  summarise(media=mean(g2_3))
```

```
##    agrupacion_actividad     media
## 1                     1  94.06061
## 2                     2 317.55172
## 3                     3 241.94444
## 4                     4 168.38000
## 5                     5 345.82222
## 6                     6 410.87195
## 7                     7 196.11450
## 8                     8 322.64444
## 9                     9 569.18182
## 10                   10 184.64748
## 11                   11 256.01183
## 12                   12 323.78125
## 13                   13 132.06977
```

group_by()

Pero perfectamente podríamos haber sumado el número de casos (empresas)

```
encla %>%
  group_by(agrupacion_actividad) %>%
  summarise(empresas=n())
```

```
##    agrupacion_actividad empresas
## 1                     1       33
## 2                     2       29
## 3                     3      162
## 4                     4       50
## 5                     5       45
## 6                     6      164
## 7                     7      131
## 8                     8       45
## 9                     9       66
## 10                   10      139
## 11                   11      169
## 12                   12       96
## 13                   13       43
```

group_by()

El resultado suma 1172 (número de empresas en la data)

```
encla %>% group_by(agrupacion_actividad) %>%
  summarise(empresas=n()) %>% select(empresas) %>% sum()
```

```
## [1] 1172
```

Como vimos, también podemos agrupar sacando medianas y otras medidas

```
encla %>%
  group_by(agrupacion_actividad) %>%
  summarise(median(g2_3),min(g2_3),max(g2_3)) %>%
  slice(1:3)
```

```
##   agrupacion_actividad median.g2_3. min.g2_3. max.g2_3.
## 1                    1           49        10       350
## 2                    2          208        34      1480
## 3                    3          100         6     12000
```

group_by()

Incluso podemos ordenar la data en base a un criterio y seleccionar al primer caso que lo cumple.

¿Cuáles fueron los nombres más populares en el año 2020 según sexo?

```
guaguas::guaguas %>%
  filter(anio==2020) %>%
  arrange(-n) %>%
  group_by(sexo) %>%
  slice(1)
```

```
## # A tibble: 2 x 5
## # Groups:   sexo [2]
##    anio nombre sexo      n proporcion
##   <dbl> <chr>  <chr> <dbl>      <dbl>
## 1  2020 Sofía  F      2465     0.0126
## 2  2020 Mateo  M      3867     0.0198
```

group_by()

Incluso podemos mezclar `group_by()` con `sjmisc`

```
encla %>%
group_by(agrupacion_actividad) %>%
descr(g2_3)
```

```
##
## ## Basic descriptive statistics
##
##
## Grouped by: Agricultura, ganadería, silvicultura y pesca
##
##   var    type
##  g2_3 numeric
##
##  g2_3. ¿Cuántos hombres y mujeres se encuentran afiliados a su sindicato el
##   n NA.prc  mean    sd   se md trimmed        range iqr skew
##  33      0 94.06 95.35 16.6 49   94.06 340 (10-350)  95 1.34
##
##
```

if_else()

La hemos ocupado para crear dos categorías:

```
esi2020 <- esi2020 %>%                          # Data
           mutate(filtro_mas_de_1M =            # Nueva variable
                    if_else(ing_t_d>1000000,    # Condición
                                         1,     # Verdadero
                                        0))     # Falso
```

```
table(esi2020$filtro_mas_de_1M)
```

```
##
##     0     1
## 69136  2799
```

¿Podríamos usar if_else() para crear variables con más de dos categorías?

if_else()

Pensemos en 4 tramos de ingresos...

Podemos usar varios if_else() consecutivamente

```
esi2020 <- esi2020 %>%

 mutate(filtro_4 =
  if_else(ing_t_d==0, "sin ingresos", ""),

 filtro_4 =
  if_else(ing_t_d>0 & ing_t_d<300000, "primer tramo", filtro_4),

 filtro_4 =
  if_else(ing_t_d<700000 & ing_t_d>=300000, "segundo tramo", filtro_4),

 filtro_4 =
  if_else(ing_t_d<1000000 & ing_t_d>=700000, "tercer tramo", filtro_4),

 filtro_4 =
  if_else(ing_t_d>=1000000, "cuarto tramo", filtro_4)
                  )
```

if_else()

```
table(esi2020$filtro_4)
```

```
##
##  cuarto tramo  primer tramo segundo tramo  sin ingresos  tercer tramo
##          2961          3122          9855         53653          2344
```

Funciona, pero es bastante código.

Hay otras formas de hacerlo más sencillo.

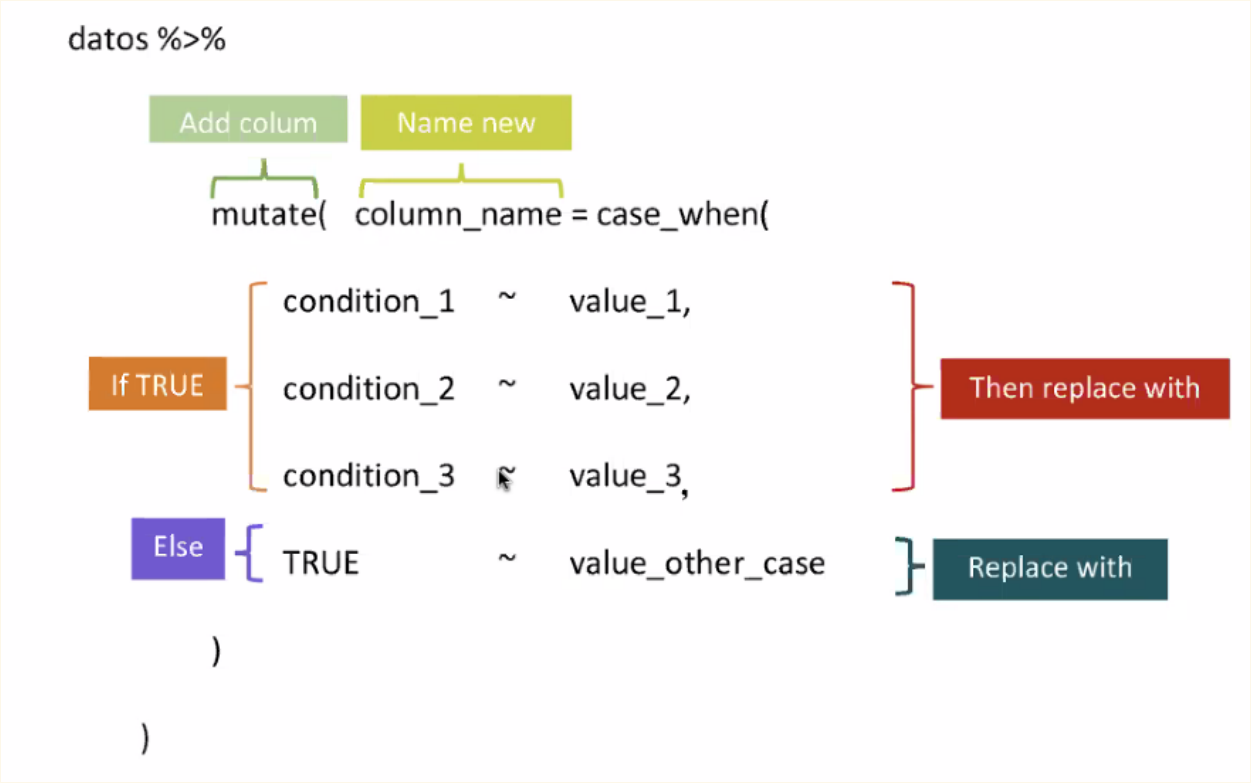
Una de esas es con case_when()

Esta es la lógica (cada línea independiente)

```
dataframe %>%
  mutate(nueva = case_when(condicion_1 ~ valor_1,
                           condicion_2 ~ valor_2,
                           condicion_3 ~ valor_3))
```

case_when()

Y así su forma general:



case_when()

Recodificar ingresos con case_when()

```
esi2020 <- esi2020 %>%
    mutate(filtro_4 = case_when(
              ing_t_d>0 & ing_t_d<300000 ~ "primer tramo",
              ing_t_d<700000 & ing_t_d>=300000 ~ "segundo tramo",
              ing_t_d<1000000 & ing_t_d>=700000 ~ "tercer tramo",
              ing_t_d>=1000000 ~ "cuarto tramo",
              TRUE ~ "sin ingresos",
                  ))
```

```
table(esi2020$filtro_4)
```

```
##
##  cuarto tramo  primer tramo segundo tramo  sin ingresos  tercer tramo
##          2961          3122          9855         53653          2344
```

case_when

La virgulilla (~) tiene que se utilizada entre la condición y el valor a asignar.

`Alt + Control + + = ~`

Los valores a asignar deben ser de la misma clase.

No funciona con variables factores, tienen que ser character o numeric

La nueva variable debe ser numérica o carácter, no puede ser una combinación.

RECOMENDACIONES:

- Siempre probar la variable creada con `table( )` (que tenga sentido)
- Llamar con nuevo nombre a la nueva variable (no sobre escribir)
- Usar espacios para ordenar y facilitar la lectura

Paciencia con la función, al principio saltan varios errores

Prácticas para una buena codificación

Por Lindsay Carr

Por Lindsay Carr

- Carga de librerías al inicio del código (usando `library()`)
- Usa RStudio projects para organizar scripts, data y salidas
- Modulariza el código (todavía no)
- No guardar *workplace image*
- No usar funciones que cambian el computador de otro (`install.package()` o `setwd()`)
- Comenta el código, pero sin pasarte (no incluir interpretaciones o resultados).
- El principal destinatario de tus comentarios eres tú en 3 meses más.
- Si quieres interpretar el código y mostrar los resultados usa RMarkdown

Por Lindsay Carr

- Aprovecha el autollenado de RStudio (evita errores de tipeo)
- Copia y pega código utilizado anteriormente o por otros.
- Utiliza loops o **funciones** cuando te veas copiando y pegando código reemplazando valores
- Las tareas mecánicas en R pueden automatizarse
- Evita códigos anchos (sobre todo con pipes)

```
data %>%
  funcion1() %>%
  funcion2() %>%
  funcion3()
```

Ejercicio práctico

Utilizando Encuesta Suplementaria de Ingresos (ESI)

Ejercicio práctico ESI

- Carga el microdato de la ESI personas 2020 (la de la prueba 1)
- Crea 3 tablas con el promedio de ingresos (cualquier variable de ingresos) según sexo, región y sector económico
- ¿Cuántas personas tienen entre 0 y 17 años, entre 18 y 29 años, entre 30 y 64 años, y 65 o más años? (cuatro grupos)
- Crea una nueva data donde cada unidad (fila) sean los hogares y que solo tenga 3 variables: identificador del hogar, ingresos del hogar y región en la que se ubica el hogar.
- ¿Cuántos hogares son?, ¿Cómo se distribuyen regionalmente estos hogares?
- Envía por correo a equipo docente (antes del 08 de octubre)

Recursos web utilizados

Xaringan: Presentation Ninja, de Yihui Xie. Para generar esta presentación.

Ilustraciones de Allison Horst

Para reforzar y seguir aprendiendo

Explicaciones simples y rápidas de función group_by() y operador pipe

Un poco más del operador pipe (por Wickham)

Bibliografía utilizada

Wickham, H. (2021). *R Para Ciencia de Datos*.