Tidyverse

September 25, 2025

1 Introducción a la Programación para Ciencia de Datos

1.1 Lenguaje de programación R

Rocío Romero Zaliz - rocio@decsai.ugr.es

2 Tidyverse

Una colección de paquetes con una gramática, filosofía y estructura similar (https://tidyverse.tidyverse.org)

Wickham H, Averick M, Bryan J, Chang W, McGowan LD, François R, Grolemund G, Hayes A, Henry L, Hester J, Kuhn M, Pedersen TL, Miller E, Bache SM, Müller K, Ooms J, Robinson D, Seidel DP, Spinu V, Takahashi K, Vaughan D, Wilke C, Woo K, Yutani H (2019). "Welcome to the tidyverse." Journal of Open Source Software, 4(43), 1686. doi:10.21105/joss.01686.

[ ]:
```
# Carga del paquete
library(tidyverse)
```

2.1 Cheatsheets

https://posit.co/resources/cheatsheets/

[ ]:
```
class(mtcars)
```

[ ]:
```
mtcars
```

2.2 Pipeline (%>%) - paquete magrittr

Operador que sirve para realizar varias operaciones de forma secuencial sin recurrir a parentesis anidados o a sobrescribir bases de datos.

[ ]:
```
# Sin magrittr
x <- c(1, 4, 6, 8)
y <- round(mean(sqrt(log(x))), 2)
```

```
y
```

[ ]:
```
# Con magrittr
# library(magrittr)
x <- c(1, 4, 6, 8)
y <- x %>% log() %>% sqrt() %>% mean() %>% round(2)
y
```

[ ]:
```
# Con magrittr
x <- c(1, 4, 6, 8)
y <- x %>% log %>% sqrt %>% mean %>% round(2)
y
```

2.3 Paquete dplyr

dplyr es un paquete de R para manipular, limpiar y resumir datos no estructurados. Facilita y agiliza la exploración y manipulación de datos en R.

[ ]:
```
# Vamos a trabajar con un conjunto de datos ya creado
# library(dplyr)
starwars %>% print
```

[ ]:
```
class(starwars)
```

[ ]:
```
mtcars$pepe
```

[ ]:
```
# Tibble: si accedes a una columna que no existe te avisa...
starwars$pepe
```

[ ]:
```
starwars %>% str
```

[ ]:
```
mtcars$pepe <- starwars$films
```

2.3.1 filter()

Selecciona filas en un data frame

[ ]:
```
mtcars[mtcars$hp > 100,]
```

[ ]:
```
# Busco a los andorides...
# Fuera del tidyverso
print(starwars[!is.na(starwars$species) & starwars$species == "Droid",])
```

[ ]:
```
# Dentro del tidyverso
starwars %>% filter(species == "Droid") %>% print
```

```
[ ]: filter(starwars, species == "Droid") %>% print
```

```
[ ]: starwars %>% filter(species == "Droid") %>% filter(homeworld == "Naboo") %>%␣
     ↪print
```

```
[ ]: starwars %>% filter(species == "Droid") %>% filter(height < 100) %>% print
```

```
[ ]: starwars %>% filter(species == "Droid") %>%
         filter(height < 100 | is.na(height)) %>% print
```

```
[ ]: starwars %>% filter(species == "Droid") %>%
         filter(height >= 96 & height < 200) %>% print
```

```
[ ]: starwars %>% filter(species == "Droid", height >= 96 & height < 200) %>% print
```

```
[ ]: starwars %>% filter(species == "Droid") %>%
         filter(height >= 96 & homeworld %in% c("Naboo", "Tatooine")) %>% print
```

```
[ ]: print(starwars[10:15,])
```

```
[ ]: starwars %>% slice(10:15) %>% print
```

```
[ ]: starwars %>% slice_max(n=5, height) %>% print # slice_max
```

```
[ ]: starwars %>% slice_min(n=5, height) %>% print # slice_min, antiguamente top_n
```

2.3.2 select()

Permite seleccionar variables (columnas) del data frame

```
[ ]: print(starwars[,c("name","homeworld")])
```

```
[ ]: starwars %>% select(name, homeworld)
```

```
[ ]: starwars %>% select(-name, -homeworld) %>% print
```

```
[ ]: starwars %>% select(starts_with("s")) %>% head(5) %>% print
```

```
[ ]: starwars %>% select(ends_with("es")) %>% head(5) %>% print
```

```
[ ]: starwars %>% select(contains("a")) %>% head(5) %>% print
```

```
[ ]: starwars %>% select(name)
```

```
[ ]: starwars %>% pull(name)
```

[ ]:
```
class(starwars %>% pull(name))
```

2.3.3 arrange()

Reordena filas en un data frame

[ ]:
```
starwars %>% arrange(height) %>% print
```

[ ]:
```
starwars %>% arrange(desc(height)) %>% print
```

[ ]:
```
starwars %>% arrange(height, desc(birth_year)) %>% print
```

2.3.4 rename()

Renombra columnas en un data frame

[ ]:
```
starwars %>% rename(hair = hair_color) %>% print
```

2.3.5 mutate()

Crea nueva columnas en un data frame o actualiza las ya existentes

[ ]:
```
starwars %>%
    mutate(height_in = height * 0.393701) %>%
    select(starts_with("he")) %>% head(5)
```

[ ]:
```
starwars %>% select(sex) %>% head(5)
```

[ ]:
```
starwars %>%
    mutate(sex = as.factor(sex), height = 1) %>% print
```

También existen funciones `mutate_if` y `mutate_at`.

[ ]:
```
starwars %>% mutate_if(is.character, as.factor) %>% print
```

[ ]:
```
starwars %>% mutate_at(c("name","sex"), as.factor) %>% print
```

2.3.6 summarise()/summarize()

Permite colapsar/resumir filas en un data frame

[ ]:
```
summary(starwars)
```

[ ]:
```
starwars$height
```

[ ]:
```
starwars$height %>% mean(na.rm=TRUE)
```

[ ]:
```
starwars %>% summarise(promedio = mean(height, na.rm=TRUE), NN = n(), desv =␣
↪sd(mass))
# SELECT AVG(height) AS promedio, COUNT(*) AS NN, SD(mass) AS desv FROM starwars
```

[ ]:
```
starwars %>% filter(species == "Droid") %>% summarise(NN = n())
# SELECT COUNT(*) FROM starwars WHERE species='Droid'
```

[ ]:
```
starwars %>% filter(species == "Droid") %>% count
```

[ ]:
```
starwars %>% summarise(desviacion_tipica = sd(height, na.rm=TRUE))
```

[ ]:
```
starwars %>% summarise(max(mass, na.rm=TRUE))
```

[ ]:
```
starwars %>% summarise(mean = mean(height, na.rm=TRUE), sd = sd(height, na.
↪rm=TRUE))
```

[ ]:
```
starwars$species %>% unique %>% length
```

[ ]:
```
starwars %>% summarise(n(), mean(mass, na.rm = TRUE)) # Que feo escribir n() y␣
↪mean() sin alias
```

[ ]:
```
# Atención que se vienen curvas...
starwars %>%
        group_by(species) %>%
        summarise(n = n(), mass = mean(mass, na.rm = TRUE))
```

[ ]:
```
starwars %>%
    group_by(species) %>%
    summarise(n = n(), mass = mean(mass, na.rm = TRUE)) %>%
    filter(n > 2, mass > 50)
```

Más funciones útiles para usar con sumarise(): * Center: mean(), median() * Spread: sd(), IQR(), mad() * Range: min(), max(), quantile() * Position: first(), last(), nth() * Count: n(), n_distinct() * Logical: any(), all()

[ ]:
```
starwars %>% summarise_if(is.numeric, mean, na.rm = TRUE)
```

[ ]:
```
starwars %>% summarise_if(is.numeric, list(prom=mean,sd=sd), na.rm = TRUE)
```

[ ]:
```
starwars %>%
    summarise_at(c("height","mass"), mean, na.rm = TRUE)
```

[ ]:
```
starwars %>%
    summarise_at(vars(height,mass), mean, na.rm = TRUE)
```

[ ]:
```
starwars %>%
    summarise_at(vars(height,mass), list(mean,sd), na.rm = TRUE)
```

[ ]:
```
starwars %>% select(height, mass, birth_year) %>%
    summarise_all(list(minimo = min, maximo = max), na.rm = TRUE)
```

2.3.7 across()

Se utiliza para aplicar una operación a varias columnas de un data frame de manera simultánea. Esta es una versión más moderna y más legible.

[ ]:
```
starwars %>% select(height) %>% unique
```

[ ]:
```
starwars %>%
  group_by(species) %>%
  filter(n() > 1) %>% # ¡Ojo! usad parentesis si no os lía con la función n()
  summarise(across(c(sex, gender, homeworld), n_distinct))
```

[ ]:
```
starwars %>%
  group_by(species) %>%
  filter(n() > 1) %>%
  summarise(across(c(sex, gender, homeworld), list(n_distinct,length))) # WTF?␣
 ↪length vs. n
```

[ ]:
```
starwars %>% n_distinct
```

[ ]:
```
starwars %>% n
```

[ ]:
```
starwars %>% length
```

[ ]:
```
dim(starwars)
```

2.4 Extra

[ ]:
```
df <- data.frame(period=c("Q1_y2019","Q2_y2019", "Q3_y2019","Q4_y2019"),␣
 ↪revenue=c(23,24,27,29))
df
```

[ ]:
```
df %>% separate(period, c("Quarter","Year"), sep="_y") # El contario de␣
 ↪separate es unite
```

[ ]:
```
# Expresiones regulares... más adelante veremos esto
ndf <- df %>% extract(period, c("Quarter","Year"), "Q(.*)_y(.*)") %>%
    mutate_if(is.character, as.numeric)
ndf
```

[ ]:
```r
df <- tibble(
  a = c(1, 5, 3),
  b = c(4, 2, 6),
  c = c(7, 3, 1)
)

# Sin rowwise...
df %>%
    mutate(max_val = max(a, b, c))  # Devuelve el mismo valor para todas las filas
```

[ ]:
```r
# Con rowwise: calcula el máximo por fila
df %>%
  filter(a > 1) %>%
  rowwise() %>%
  mutate(max_val = max(a, b, c))
```

2.4.1 Ejercicios Tidyverse

1) Utiliza el dataset de `mtcars` y el paquete `tidyverse` para:
   - Mostrar las 5 primeras filas del dataset
   - Convertir la variables "cyl", "gear" y "carb" en factores y las variables "vs" y "am" en lógicas, y mostrar la estructura del dataset (usar este dataset transformado de aqui en adelante)
   - Mostrar solo los coches con una potencia ("hp") mayor a 100
   - Seleccionar solo las columnas "mpg", "cyl", "hp" y "qsec" del dataset
   - Calcular la cantidad total de coches para cada valor único en la columna "cyl" (número de cilindros)
   - Encontrar el modelo de coche con la mayor potencia ("hp") y mostrar su información completa
   - Calcular el promedio de potencia de los coches con 8 cilindros
2) Supongamos que tienes un data frame llamado `lego_sets` que contiene información sobre diferentes conjuntos de Lego, incluyendo el nombre del conjunto, el número de piezas, el tema y el año de lanzamiento:
   - Filtra y muestra solo los conjuntos de Lego lanzados en el año 2020
   - Encuentra y muestra el nombre y el número de piezas del conjunto de Lego más grande
   - Calcula la cantidad total de piezas para cada tema y muéstralos en orden descendente por número de piezas
   - Calcula cuántos conjuntos de Lego se lanzaron en cada año y muéstralo ordenado por año de forma ascendente
   - Encuentra los 3 temas más populares (con más conjuntos) y muestra el número de conjuntos y el número total de piezas para cada uno de ellos
3) Dado un data frame llamado `bebidas` que contiene información sobre la edad, género, tipo de bebida y cantidad de copas consumidas por un grupo de personas, filtra y muestra únicamente a las mujeres mayores de 20 años. Para este suconjunto calcula la media, el máximo, la cantidad de copas totales y la desviación estándar de la edad para cada combinación de tipo de bebida. Finalemnte, agrega una nueva columna que indique cuántas personas se encuentran

en cada grupo de tipo de bebida.

4) Dado un data frame llamado `peliculas`, indique los 2 géneros con mayor puntuación media por país pero solo de peliculas para mayores de 13 o superiores, que incluya el número de películas en ese género, su beneficio medio (ganancia-presupuesto), la desviación estándar de la puntuación y la puntuación media. Ordenar los resultados por puntuación media por país de forma descendente.

5) Dado un conjunto de datos llamado notas que contiene información sobre las notas de los estudiantes en varias asignaturas, agrega una columna con la nota promedio de cada estudiante. Previamente reemplaza los valores faltantes con 0 (HINT: usa replace_na).

2.5 Practicando...

[ ]:
```
starwars %>% print
```

[ ]:
```
# Selecciona solo las columnas name, height y mass.
```

[ ]:
```
starwars %>% select(name, height, mass) %>% arrange(mass) %>% head(10)
```

[ ]:
```
starwars %>% select(name, height, mass) %>% slice_min(n = 10, mass)
```

[ ]:
```
# Agrupa los personajes por especie y calcula la masa promedio por grupo.
```

[ ]:
```
starwars %>% group_by(species) %>% summarize(masa_promedio = mean(mass, na.rm =␣
 ↪TRUE)) %>%
    filter(!is.nan(masa_promedio))
```

[ ]:
```
starwars %>% filter(species == "Xexto") %>% print
```

[ ]:
```
starwars %>% filter(species == "Droid") %>% select(name, species, height) %>%␣
 ↪arrange(desc(height))
```

[ ]:
```
# Encuentra el personaje más alto de cada especie.
los_mas_altos <- starwars %>% group_by(species) %>% summarise(height =␣
 ↪max(height, na.rm=TRUE))
print(los_mas_altos)
merge(starwars, los_mas_altos) %>% filter(species == "Droid") %>% print
```

[ ]:
```
starwars %>% group_by(species) %>% summarise(el_mas_alto = max(height, na.
 ↪rm=TRUE))
```

[ ]:
```
# Calcula el IMC (Índice de Masa Corporal) de cada personaje usando la fórmula...
# Luego, ordena los personajes por IMC y filtra los que tienen valores extremos␣
 ↪(IMC > 40 o < 15).
```

[ ]:
```r
starwars %>% select(name, height, mass) %>% mutate(imc = mass / ((height/
↪100)^2)) %>%
   arrange(imc) %>% filter(imc >= 15 & imc <= 40)
```

[ ]:
```r
# Identifica los planetas (homeworld) con más diversidad de especies.
starwars %>%
    filter(!is.na(homeworld)) %>%
    group_by(homeworld) %>%
    summarize(diversidad = n_distinct(species)) %>%
    arrange(desc(diversidad))
```

[ ]:
```r
summary(starwars)
```

[ ]:
```r
?quantile
```

[ ]:
```r
qq0 <- starwars %>% summarize_if(is.numeric, quantile, probs = 0, na.rm = TRUE)
qq1 <- starwars %>% summarize_if(is.numeric, quantile, probs = 0.25, na.rm =␣
↪TRUE)
print(qq0)
print(qq1)
```

[ ]:
```r
starwars %>% summarise_if(is.numeric, list(min, mean, max), na.rm = TRUE)
```

[ ]:
```r
starwars %>% summarise_if(is.numeric, is.na) %>% summarise_all(sum)
```

[ ]:
```r
starwars %>% mutate_if(is.numeric, is.na) %>% summarise_if(is.logical, sum)
```