



JIS Go Live
2020

< iVirtual
experience! >

R para Ciencia de Datos

Jornadas de Informática en Salud

Equipo

Julieta Terzano

Ernesto Surijon

Fabricio Sobral

Alejandro Renato

Hernán Berinsky

Fernando Binder

Contenidos

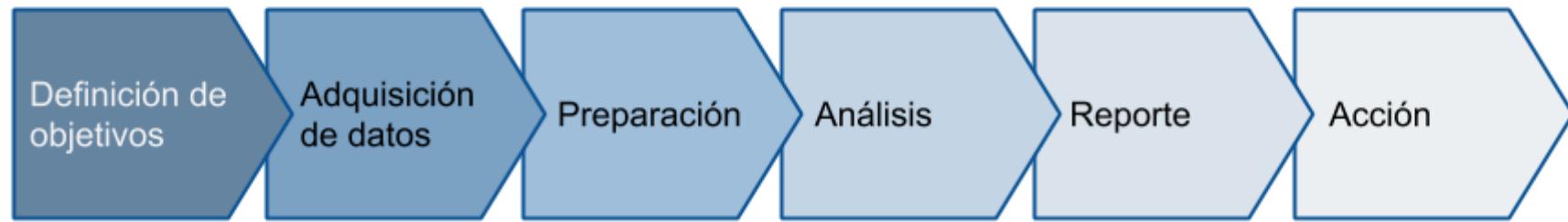
- Pasos de un proyecto de ciencia de datos
- Software preenvasado y lenguajes de programación
- R y RStudio IDE:
 - Paneles de RStudio
 - Objetos y funciones en R
 - Carga de datos
 - Tablas
 - Tidyverse: librerías para ciencia de datos
 - Gráficos con ggplot2
 - Tests de hipótesis



Proyectos de Ciencia de Datos

Pasos propuestos

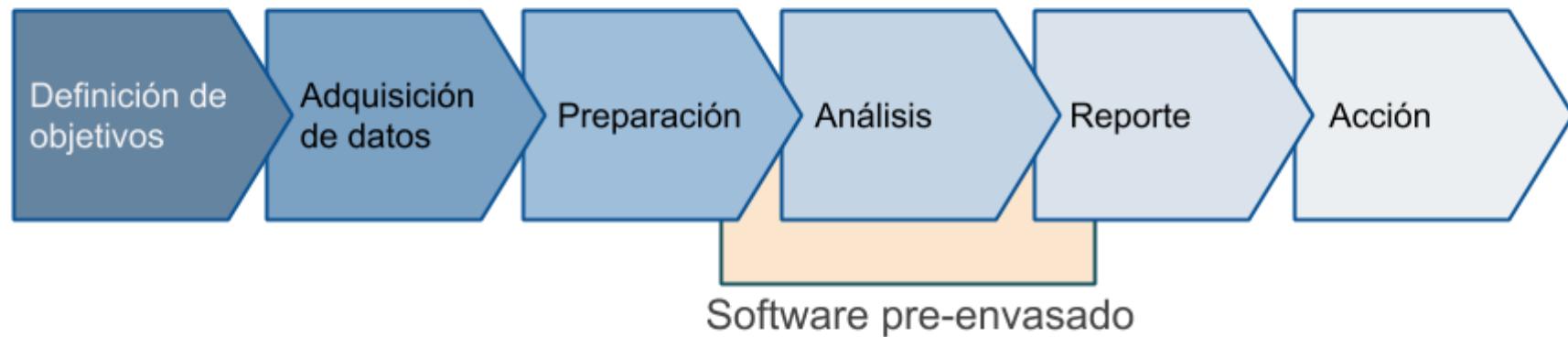
Pasos en un proyecto de ciencia de datos



Las herramientas de software

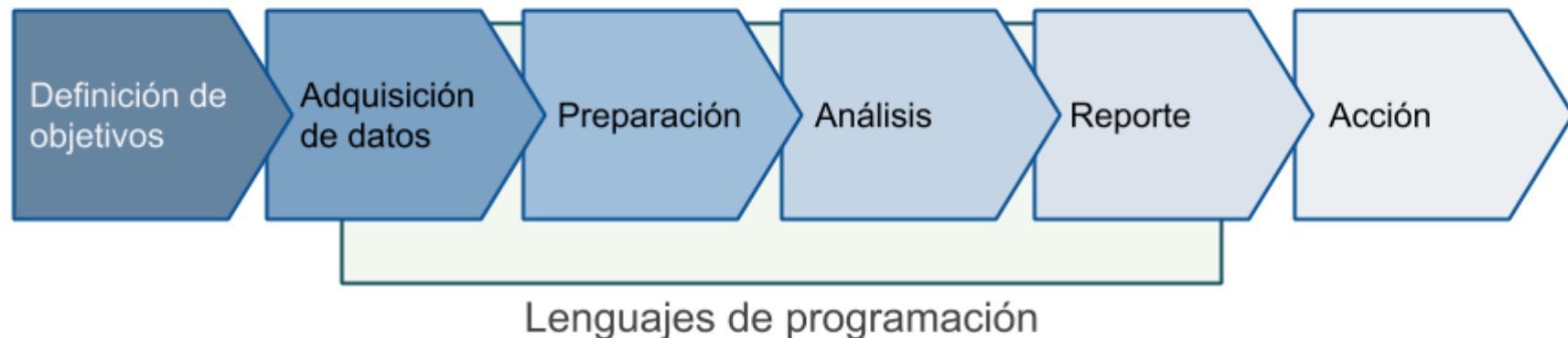
Software pre-envasado y lenguajes de programación

Pasos en un proyecto de ciencia de datos: herramientas



“Usar software pre-envasado es como comprar comida para llevar cada vez que tenés hambre.” - *J.P. Onnela*

Pasos en un proyecto de ciencia de datos: herramientas





R para Análisis de Datos

- Lenguaje de programación y entorno de trabajo para análisis estadístico y visualización de gráficos
- Código abierto
- Flexible y extensible
- Comunidad activa



Principal entorno de programación cuantitativa en **estadística académica** y creciente popularidad en **análisis de datos en salud**

R para Análisis de Datos

- Lenguaje de programación y entorno de trabajo para análisis estadístico y visualización de gráficos
- Código abierto
- Flexible y extensible
- Comunidad activa

Instalación de R y R Studio

Instalación de R y R Studio

Paso 1

Instalar R



Paso 2

Instalar **RStudio**

(Entorno de Desarrollo Integrado -
IDE)



Alternativa: RStudio Cloud

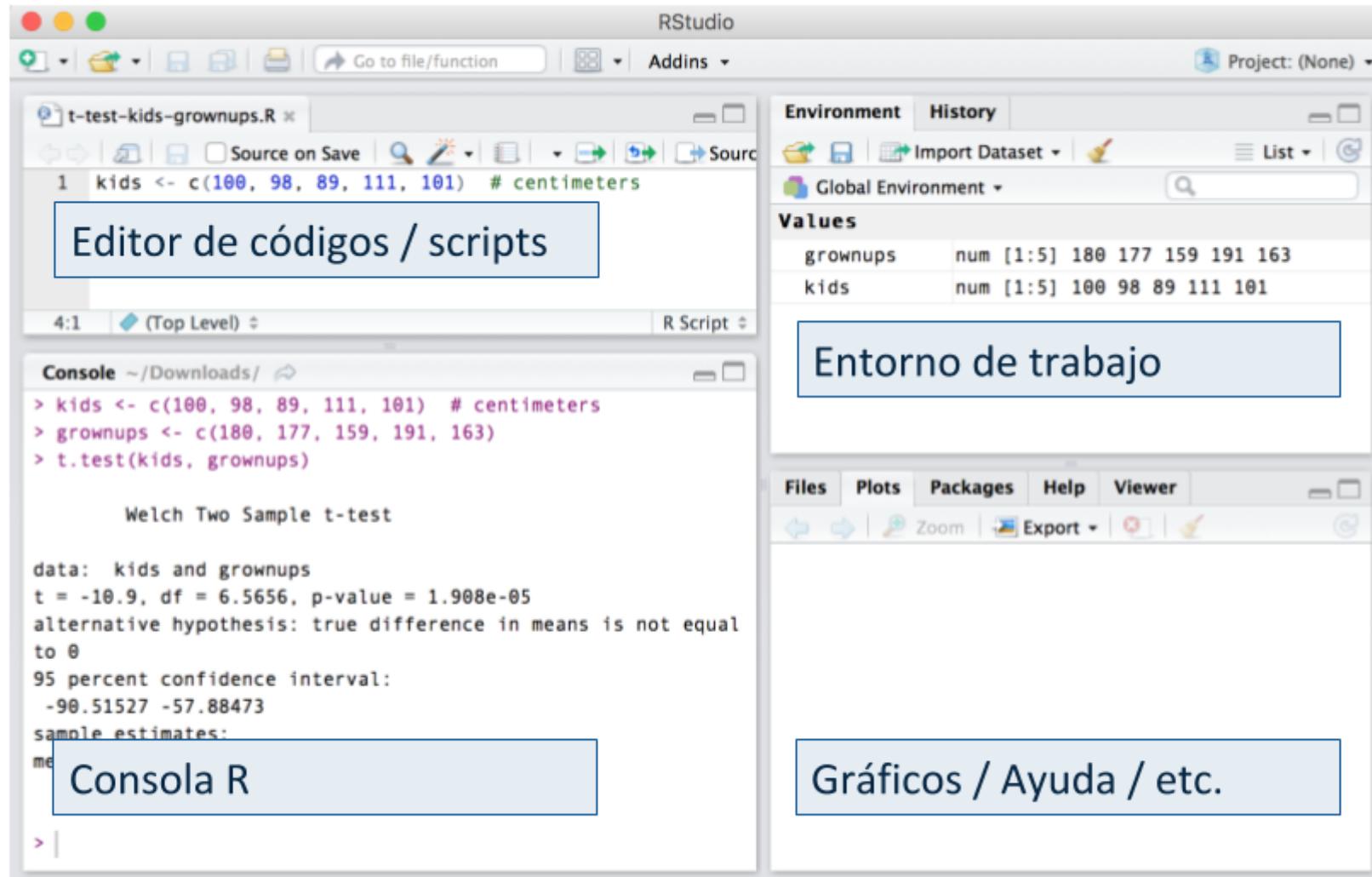
rstudio.cloud



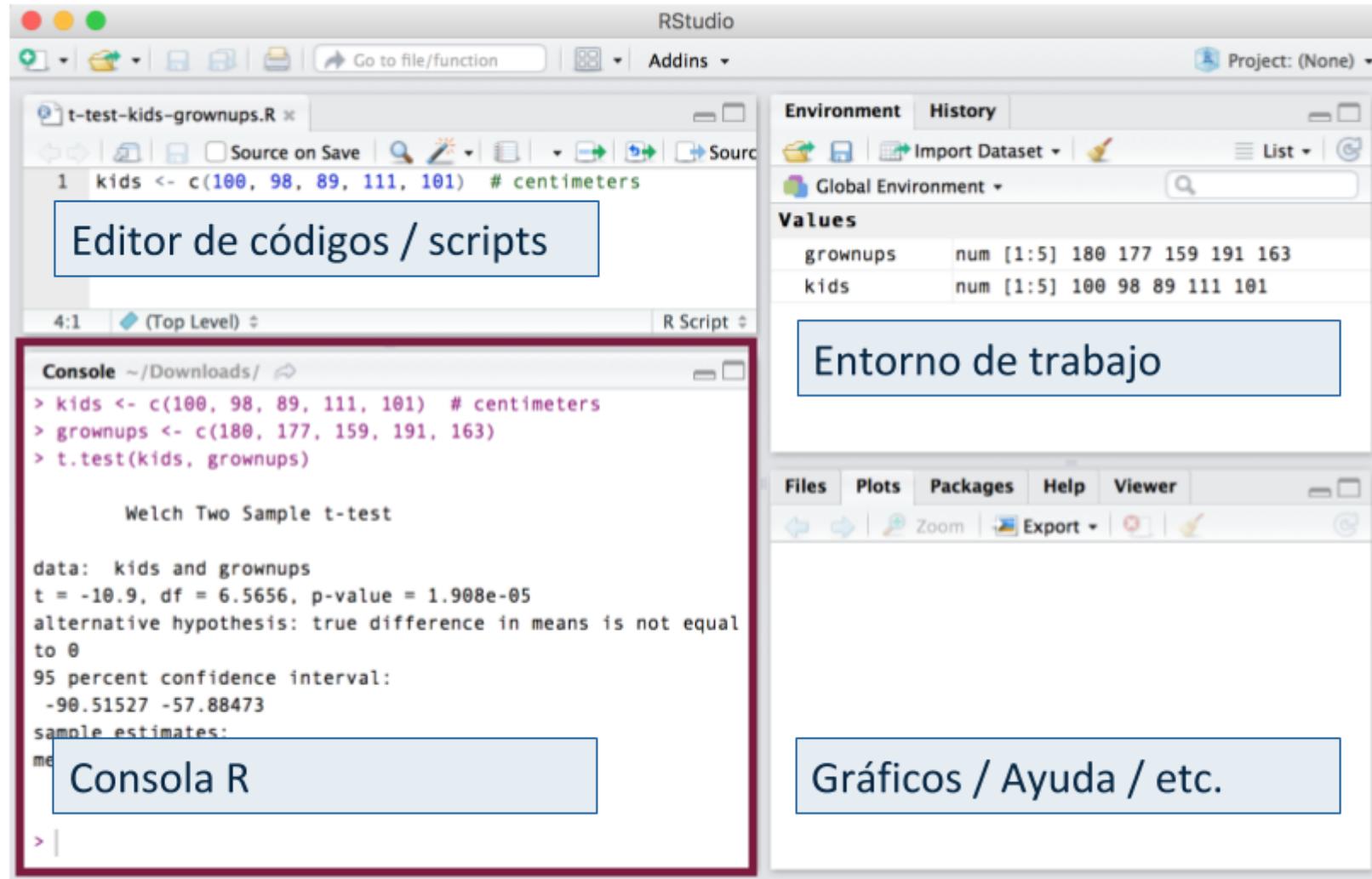
Paneles de RStudio



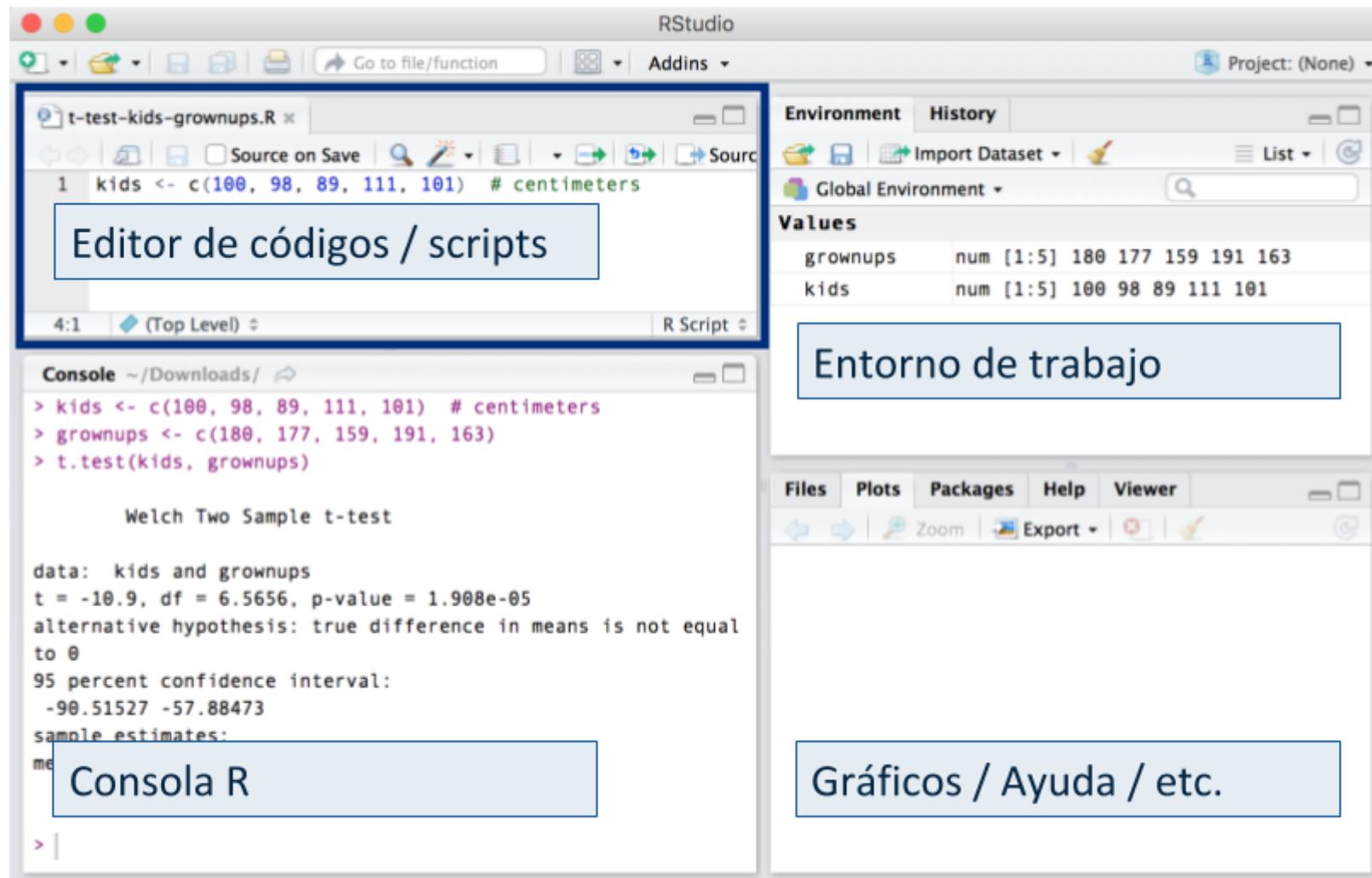
Paneles de RStudio



Paneles de RStudio

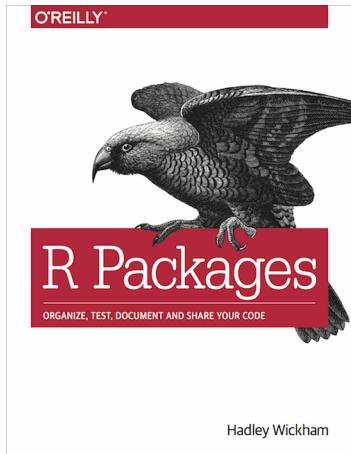


Paneles de RStudio



Librerías / paquetes

Flexibilidad y extensibilidad de R



Conjuntos de:

- Funciones
- Documentación
- Sets de datos de ejemplo

<https://r-pkgs.org/>

https://cran.r-project.org/web/packages/available_packages_by_name.html

Tidyverse visualization manipulation basics, *G Grolemund 2017*

Librerías / paquetes

Flexibilidad y extensibilidad de R

Instalación

```
install.packages("dplyr")
install.packages("ggplot2")
```

"1 vez por computadora"

Carga de paquetes

```
library(dplyr)
library(ggplot2)
```

"1 vez por sesión"

Tidyverse: Librerías para Ciencia de Datos



```
install.packages("tidyverse")
```

```
library(tidyverse)
```

— Attaching packages —

✓ tibble 3.0.1	✓ purrr 0.3.4
✓ tidyverse 1.1.2	✓ stringr 1.4.0
✓ readr 1.3.1	✓forcats 0.5.0

— Conflicts — tidyverse_conflicts() —

x dplyr::filter()	masks stats::filter()
x dplyr::lag()	masks stats::lag()

Actividad



Actividad



Instalar y cargar el paquete Tidyverse





Objetos en R: Variables, vectores y data.frames

Objetos en R

Crear un objeto

Operador 'asignar': <-

```
a <- 4
```

Imprimir el contenido del objeto

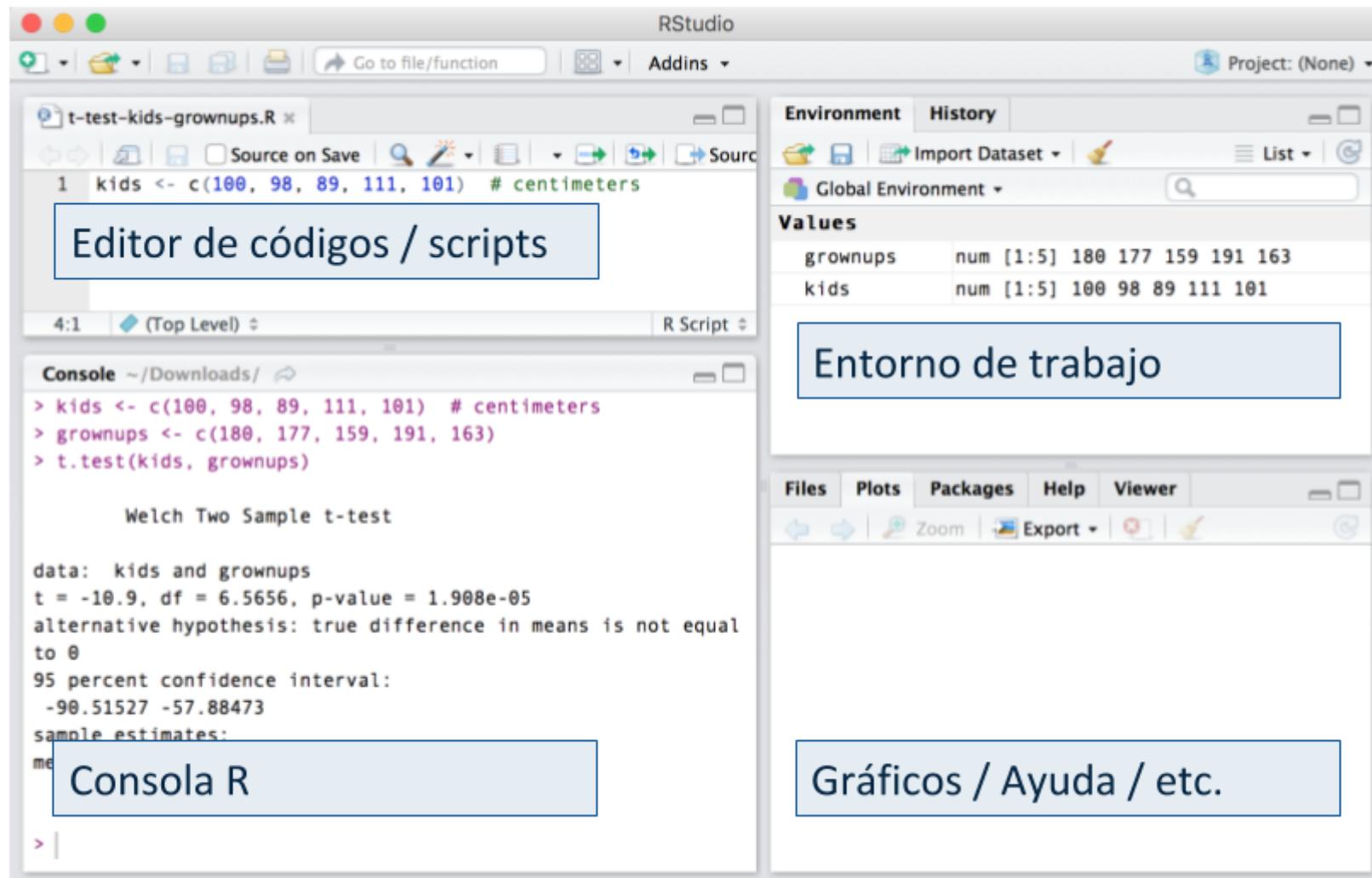
```
print(a)
```

```
[1] 4
```

```
a
```

```
[1] 4
```

Paneles de RStudio



Objetos en R

Crear un objeto

Operador 'asignar': <-

```
a <- 4
```

Imprimir el contenido del objeto

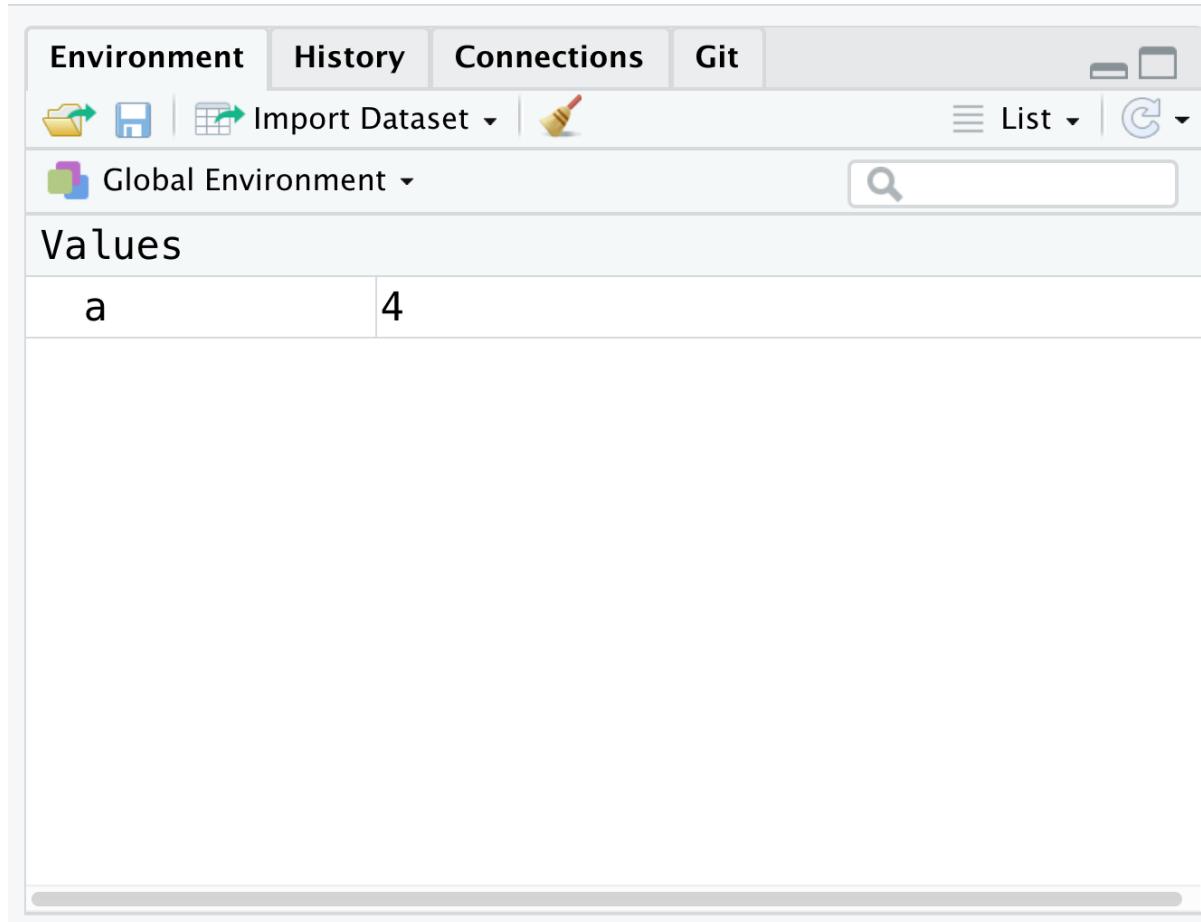
```
print(a)
```

```
[1] 4
```

```
a
```

```
[1] 4
```

Objetos en el Entorno de Trabajo



The screenshot shows the RStudio interface with the 'Environment' tab selected. The global environment contains one object named 'a' with the value '4'. The interface includes a toolbar with icons for file operations, a search bar, and a list view.

Values	
a	4

Tipos de objetos

función `typeof()`

```
a <- 4  
typeof(a)
```

```
[1] "double"
```

```
b <- "texto"  
typeof(b)
```

```
[1] "character"
```

```
c <- TRUE  
typeof(c)
```

```
[1] "logical"
```

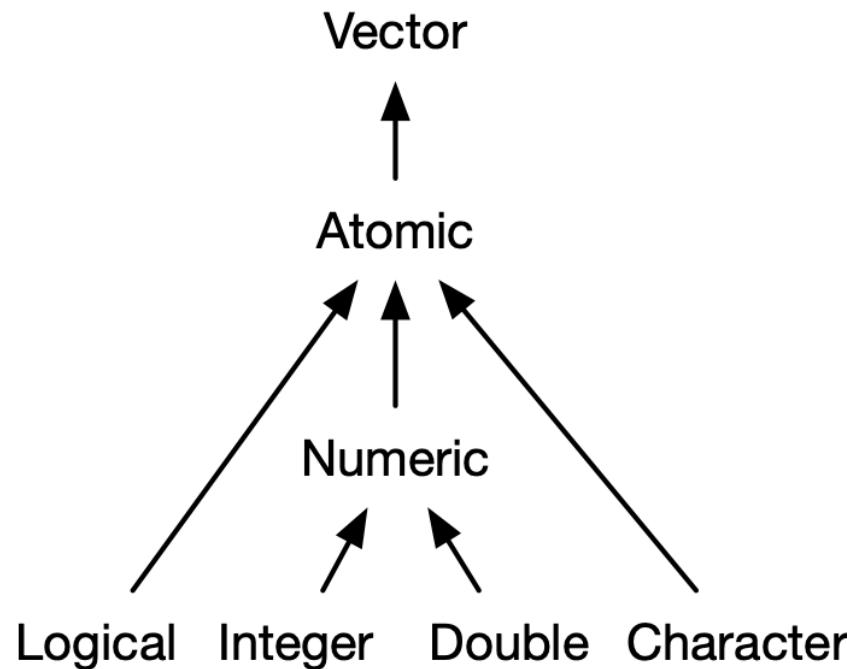
```
d <- 3.1416  
typeof(d)
```

```
[1] "double"
```

Clases de Objetos

Clases principales			
numeric		character	logical
integer	double		
1 25	34.6 3.1415	"texto"	TRUE FALSE

Clases de Objetos



Advanced R. H Wickham. Chapman & Hall/CRC, 2014

Vectores

Objetos lineales. Función c()



```
vect <- c(1,1,2,3)
```

```
vect
```

```
[1] 1 1 2 3
```

Vectores

Vectores de texto

"Lucia"

"Rocamadour"

"Ronald"

"Babs"

```
vect <- c("Lucia", "Rocamadour", "Ronald", "Babs")
```

```
vect
```

```
[1] "Lucia"      "Rocamadour" "Ronald"      "Babs"
```

Crear Vectores

Opeardores c(), : y función seq()

```
vect <- c(1,3,5,7)  
vect
```

```
[1] 1 3 5 7
```

seq(from, to, by)

```
vect <- seq(1, 15, by = 2)  
vect
```

```
[1] 1 3 5 7 9 11 13 15
```

```
vect <- 1:5  
vect
```

```
[1] 1 2 3 4 5
```

Crear Vectores

Función seq()

funcion(objeto)

funcion(argumento1, argumento2)

Crear Vectores

Función seq()

```
?seq()
```

The screenshot shows the R Help Viewer interface. The title bar reads "R: Sequence Generation". The main content area displays the documentation for the "seq {base}" function. The title is "Sequence Generation". The "Description" section states: "Generate regular sequences. seq is a standard generic with a default method. seq.int is a primitive which can be much faster but has a few restrictions. seq_along and seq_len are very fast primitives for two common cases." The "Usage" section shows the function signature: "seq(...)" followed by the "#> Default S3 method:" and the source code: "seq(from = 1, to = 1, by = ((to - from)/(length.out - 1)), length.out = NULL, along.with = NULL, ...)".

Crear Vectores

Función seq()

```
seq(from = 3, to = 21, by = 3)
```

```
[1] 3 6 9 12 15 18 21
```

```
seq(from = 0, to = 2, length.out = 9)
```

```
[1] 0.00 0.25 0.50 0.75 1.00 1.25 1.50 1.75 2.00
```

Actividad



Crear objetos y vectores atómicos

Usar las funciones:

`c()`

`:`

`seq()`

Crear vectores numéricos:

- los números pares del 2 al 10
- números del 0 al 1 en saltos de 0.01 unidades
- 0, 50, 100, 150, 200... hasta 1000.

Crear vectores numéricos

Los números pares del 2 al 10

```
seq(2, 10, by = 2)
```

```
[1] 2 4 6 8 10
```

```
(1:5) * 2
```

```
[1] 2 4 6 8 10
```

Crear vectores numéricos

Números del 0 al 1 en saltos de 0.01 unidades

```
seq(0, 1, by = 0.01)
```

```
[1] 0.00 0.01 0.02 0.03 0.04 0.05 0.06 0.07 0.08 0.09 0.10 0.11 0.12 0.13  
[16] 0.15 0.16 0.17 0.18 0.19 0.20 0.21 0.22 0.23 0.24 0.25 0.26 0.27 0.28  
[31] 0.30 0.31 0.32 0.33 0.34 0.35 0.36 0.37 0.38 0.39 0.40 0.41 0.42 0.43  
[46] 0.45 0.46 0.47 0.48 0.49 0.50 0.51 0.52 0.53 0.54 0.55 0.56 0.57 0.58  
[61] 0.60 0.61 0.62 0.63 0.64 0.65 0.66 0.67 0.68 0.69 0.70 0.71 0.72 0.73  
[76] 0.75 0.76 0.77 0.78 0.79 0.80 0.81 0.82 0.83 0.84 0.85 0.86 0.87 0.88  
[91] 0.90 0.91 0.92 0.93 0.94 0.95 0.96 0.97 0.98 0.99 1.00
```

```
(0:100) / 100
```

```
[1] 0.00 0.01 0.02 0.03 0.04 0.05 0.06 0.07 0.08 0.09 0.10 0.11 0.12 0.13  
[16] 0.15 0.16 0.17 0.18 0.19 0.20 0.21 0.22 0.23 0.24 0.25 0.26 0.27 0.28  
[31] 0.30 0.31 0.32 0.33 0.34 0.35 0.36 0.37 0.38 0.39 0.40 0.41 0.42 0.43  
[46] 0.45 0.46 0.47 0.48 0.49 0.50 0.51 0.52 0.53 0.54 0.55 0.56 0.57 0.58  
[61] 0.60 0.61 0.62 0.63 0.64 0.65 0.66 0.67 0.68 0.69 0.70 0.71 0.72 0.73
```

Crear vectores numéricos

0, 50, 100, 150, 200... hasta 1000

```
seq(0, 1000, by = 50)
```

```
[1]    0    50   100   150   200   250   300   350   400   450   500   550   600   650  
[16] 750 800 850 900 950 1000
```

Acceder a elementos de un vector

Operador []



Acceder a elementos de un vector

Operador []



```
vect <- c(3, 3, 4, 5, 9)  
vect
```

```
[1] 3 3 4 5 9
```

```
vect[1]
```

```
[1] 3
```

```
vect[3]
```

```
[1] 4
```

Acceder a elementos de un vector

Operador []



```
vect[1:3]
```

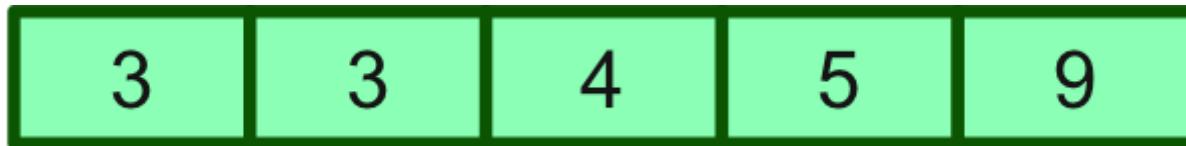
```
[1] 3 3 4
```

```
vect[-4]
```

```
[1] 3 3 4 9
```

Acceder a elementos de un vector

Operador []



Acceso a elementos usando un vector tipo lógico (TRUE | FALSE)

Trae únicamente valores que coinciden con TRUE

```
vect[c(FALSE, FALSE, TRUE, TRUE, FALSE)]
```

```
[1] 4 5
```

Acceder a elementos de un vector

Operador []



Acceso a elementos usando un vector tipo lógico (TRUE | FALSE)

```
vect > 3
```

```
[1] FALSE FALSE TRUE TRUE TRUE
```

```
vect[vect > 3]
```

```
[1] 4 5 9
```

Acceder a elementos de un vector

Operador []



Acceso a elementos usando un vector tipo lógico (TRUE | FALSE)

```
vect[vect > 3 & vect < 6]
```

```
[1] 4 5
```

```
vect[vect < 4 | vect > 6]
```

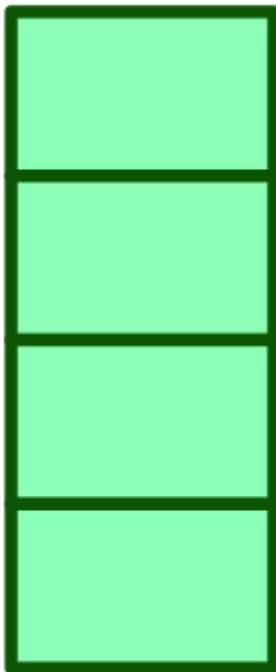
```
[1] 3 3 9
```

Tablas

Data.frames y tibbles



Tablas como vectores columnares



Tablas como vectores columnares

1
2
3
4

"A"
"B"
"C"
"D"

2019
2020
2021
2022

Data.frame

Código para crear objeto

`data.frame()`

`data.frame(col1, col2, col3)`

Data.frame

Código para crear objeto

```
data.frame(columna1 = c(1, 2, 3, 4),  
           columna2 = c("A", "B", "C", "D"),  
           columna3 = 2019:2022)
```

Data.frame

Código para crear objeto

```
data.frame(columna1 = c(1, 2, 3, 4),  
           columna2 = c("A", "B", "C", "D"),  
           columna3 = 2019:2022)
```

	columna1	columna2	columna3
1	1	A	2019
2	2	B	2020
3	3	C	2021
4	4	D	2022

Data.frame

Código para crear objeto: asignar <–

```
base <- data.frame(columna1 = c(1, 2, 3, 4),  
                    columna2 = c("A", "B", "C", "D"),  
                    columna3 = 2019:2022)
```

```
base
```

	columna1	columna2	columna3
1	1	A	2019
2	2	B	2020
3	3	C	2021
4	4	D	2022

Actividad



Crear data.frame

```
base <- data.frame(columna1 = c(1, 2, 3, 4),  
                    columna2 = c("A", "B", "C", "D"),  
                    columna3 = 2019:2022)
```

Exploración de datos



Test con datos: gapminder

```
install.packages("dslabs")
```

DSLabs package - Rafael A. Irizarry, Amy Gill

Exploración de datos



Crear base 'mundo'

```
mundo <- dslabs::gapminder
```

DSLabs package - Rafael A. Irizarry, Amy Gill

Exploración inicial mínima

Exploración inicial mínima

```
dim()
```

```
names()
```

```
str()
```

```
head()
```

Exploración inicial

Dimensiones Nombres Estructura Cabecera

Dimensiones

```
dim(mundo)
```

```
[1] 10545      9
```

Nombres

```
names(mundo)
```

```
[1] "country"          "year"           "infant_mortality" "life_expectancy"
[5] "fertility"         "population"       "gdp"              "continent"
[9] "region"
```

Exploración inicial

Dimensiones Nombres **Estructura** Cabecera

Estructura

```
str(mundo)
```

```
'data.frame': 10545 obs. of 9 variables:  
 $ country       : Factor w/ 185 levels "Albania","Algeria",...: 1 2 3 4 5 ...  
 $ year          : int  1960 1960 1960 1960 1960 1960 1960 1960 1960 1960 ...  
 $ infant_mortality: num  115.4 148.2 208 NA 59.9 ...  
 $ life_expectancy: num  62.9 47.5 36 63 65.4 ...  
 $ fertility      : num  6.19 7.65 7.32 4.43 3.11 4.55 4.82 3.45 2.7 5.57 ...  
 $ population     : num  1636054 11124892 5270844 54681 20619075 ...  
 $ gdp            : num  NA 1.38e+10 NA NA 1.08e+11 ...  
 $ continent      : Factor w/ 5 levels "Africa","Americas",...: 4 1 1 2 2 1 1 1 1 1 ...  
 $ region         : Factor w/ 22 levels "Australia and New Zealand",...: 19 19 19 19 19 19 19 19 19 19 ...
```

Exploración inicial

Dimensiones Nombres **Estructura** Cabecera

Estructura (2): tibble::glimpse

```
glimpse(mundo)
```

Rows: 10,545

Columns: 9

```
$ country           <fct> Albania, Algeria, Angola, Antigua and Barbuda, Ar  
$ year              <int> 1960, 1960, 1960, 1960, 1960, 1960, 1960, 1960, 1960, 1960  
$ infant_mortality <dbl> 115.40, 148.20, 208.00, NA, 59.87, NA, NA, 20.30, 110.00, 110.00  
$ life_expectancy   <dbl> 62.87, 47.50, 35.98, 62.97, 65.39, 66.86, 65.66, 55.70, 55.70, 55.70  
$ fertility          <dbl> 6.19, 7.65, 7.32, 4.43, 3.11, 4.55, 4.82, 3.45, 2.45, 2.45  
$ population         <dbl> 1636054, 11124892, 5270844, 54681, 20619075, 18673000, 14500000, 13828152297, 13000000, 12000000  
$ gdp                <dbl> NA, 13828152297, NA, NA, 108322326649, NA, NA, 96000000, 90000000, 80000000  
$ continent          <fct> Europe, Africa, Africa, Americas, Americas, Asia, Asia, Asia, Asia, Asia  
$ region             <fct> Southern Europe, Northern Africa, Middle Africa
```

Exploración inicial

Dimensiones Nombres Estructura **Cabecera**

Cabecera

```
head(mundo)
```

	country	year	infant_mortality	life_expectancy	fertility
1	Albania	1960	115.40	62.87	6.19
2	Algeria	1960	148.20	47.50	7.65
3	Angola	1960	208.00	35.98	7.32
4	Antigua and Barbuda	1960	NA	62.97	4.43
5	Argentina	1960	59.87	65.39	3.11
6	Armenia	1960	NA	66.86	4.55
	population	gdp	continent	region	
1	1636054	NA	Europe	Southern Europe	
2	11124892	13828152297	Africa	Northern Africa	
3	5270844	NA	Africa	Middle Africa	
4	54681	NA	Americas	Caribbean	
5	20619075	108322326649	Americas	South America	

Exploración inicial

Dimensiones Nombres Estructura Cabecera

View()

```
View(mundo)
```

r_intro_2020_presentacion.Rmd* mundo Untitled2* xaringan_theme_jis2020.css

Filter

	country	year	infant_mortality	life_expectancy	fertility	population	gd
1	Albania	1960	115.40	62.87	6.19	1636054	
2	Algeria	1960	148.20	47.50	7.65	11124892	
3	Angola	1960	208.00	35.98	7.32	5270844	
4	Antigua and Barbuda	1960	NA	62.97	4.43	54681	
5	Argentina	1960	59.87	65.39	3.11	20619075	
6	Armenia	1960	NA	66.86	4.55	1867396	
7	Aruba	1960	NA	65.66	4.82	54208	
8	Australia	1960	20.30	70.87	3.45	10292328	
9	Austria	1960	37.30	68.75	2.70	7065525	
10	Azerbaijan	1960	NA	61.33	5.57	3897889	
11	Bahamas	1960	51.00	62.00	4.50	109526	

Showing 1 to 12 of 10,545 entries, 9 total columns

Actividad



Importar set de datos gapminder

Obtener sus dimensiones, nombres y tipos de variables

Exploración de set de datos gapminder

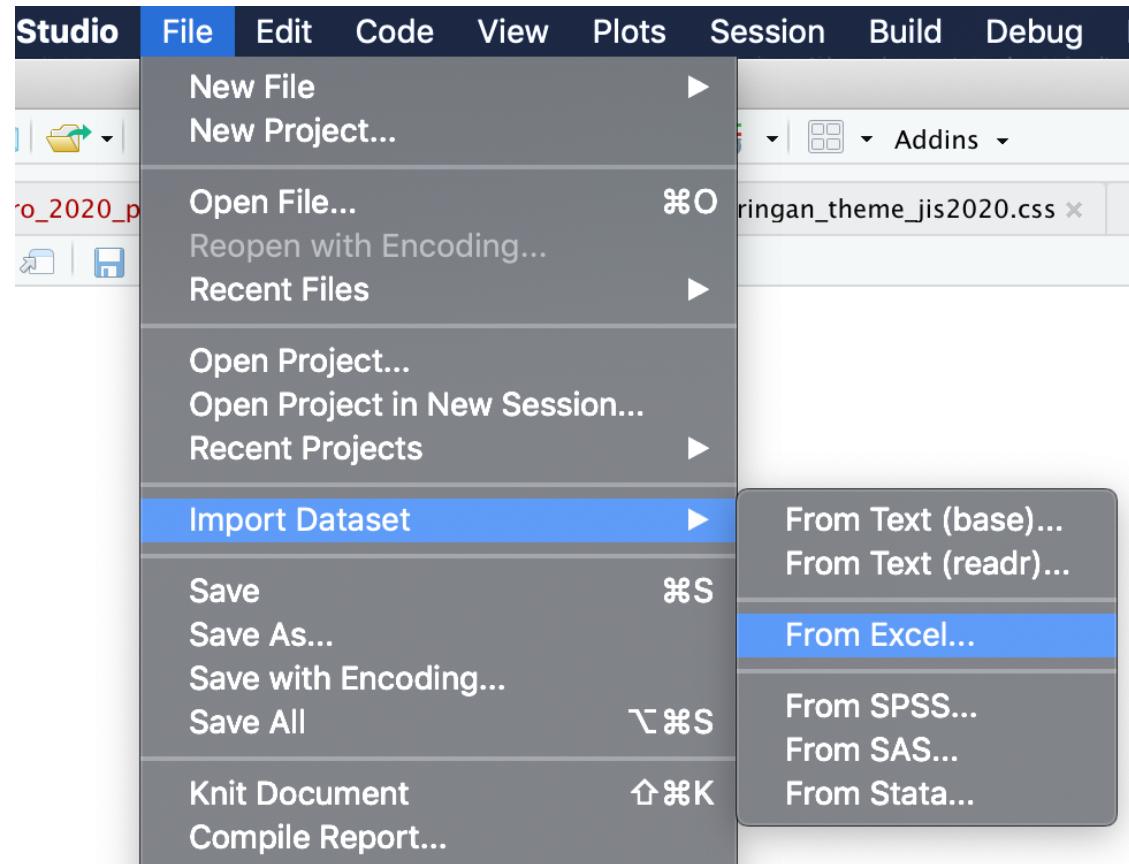
```
mando <- dslabs::gapminder  
  
dim(mundo)  
names(mundo)  
str(mundo)  
head(mundo)
```

Carga de datos



Carga de datos

Desde el menú de RStudio



Carga de datos

Con código

```
nombre_objeto <- funcion.lectura("camino/a/archivo")
```

Carga de base de datos

```
consultas <- read_csv("datos/consultas.csv")
```

Carga de datos

Con código

```
nombre_objeto <- funcion.lectura("camino/a/archivo")
```

```
consultas <- read_csv("consultas.csv") # requiere tidyverse  
consultas <- read_csv("~/proyectos/taller_r_jis/consultas.csv")
```

Carga de datos

Con código

```
nombre_objeto <- funcion.lectura("camino/a/archivo")
```

```
consultas <- read_csv("consultas.csv") # requiere tidyverse  
consultas <-  
  read_csv("https://raw.githubusercontent.com/f-binder/datos_intro
```

Carga de datos

Directorio de trabajo

```
getwd()
```

```
[1] "/Usuario/proyectos/intro_r_jis2020"
```

Actividad



Carga de datos:

consultas.csv



Carga de datos: Consultas

```
consultas <- read_csv("datos/consultas.csv") # read_csv requiere t
```

Consultas: exploración inicial

```
dim(consultas)
```

```
[1] 670 14
```

Consultas: exploración inicial

```
names(consultas)
```

```
id  
edad  
edad35  
embarazos  
glucemia  
presion_art  
bmi  
turno  
dia_semana  
hora_presente  
hora_atencion  
mins_espera  
uso_portal  
urgencias
```


Consultas: exploración inicial

```
head(consultas)
```

```
# A tibble: 6 x 14
  id     edad  edad35  embarazos  glucemia presion_art    bmi turno dia_semana
  <dbl>   <dbl>   <dbl>      <dbl>      <dbl>        <dbl> <dbl> <chr>   <chr>
1     1      50       1          3        148         72  33.6 am     Vie
2     2      31       0          0        85          66  26.6 am     Lun
3     3      32       0          4        183         64  23.3 pm    Lun
4     4      21       0          0        89          66  28.1 am    Vie
5     5      33       0          0        137         40  43.1 am    Mar
6     6      30       0          2        116         74  25.6 pm    Mar
# ... with 5 more variables: hora_presente <dttm>, hora_atencion <dttm>,
#   mins_espera <dbl>, uso_portal <dbl>, urgencias <dbl>
```

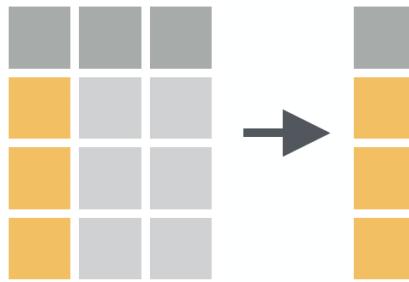
Tidyverse: Librerías para Ciencia de Datos



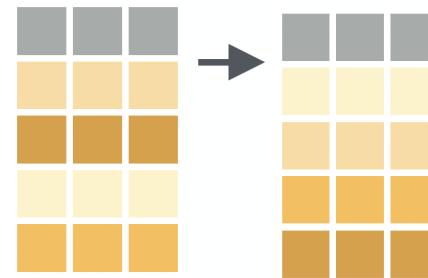
Conjunto de paquetes para ciencia de datos

Funciones de dplyr

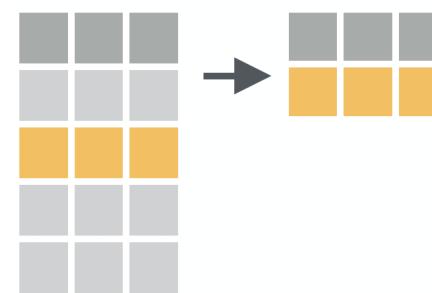
`select()`



`arrange()`

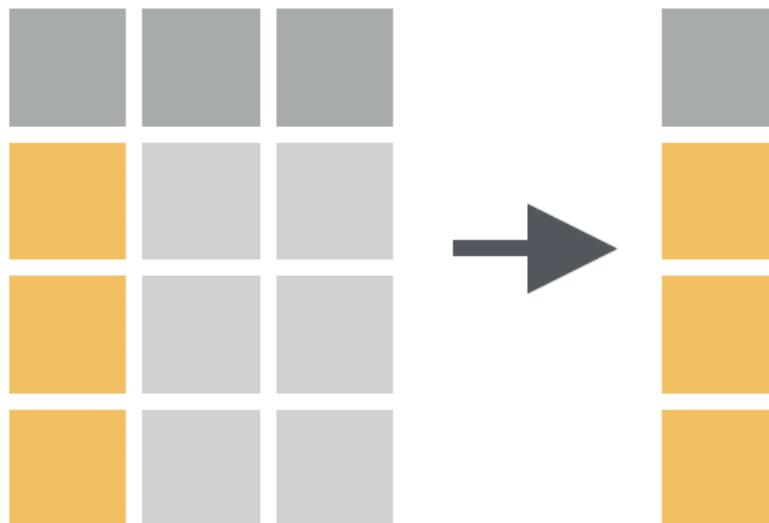


`filter()`



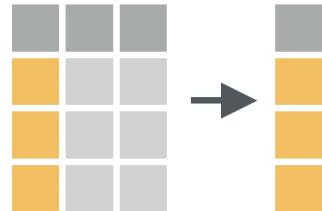
Dplyr Cheat Sheet - <https://github.com/rstudio/cheatsheets/blob/master/data-transformation.pdf>

select()



Dplyr Cheat Sheet - <https://github.com/rstudio/cheatsheets/blob/master/data-transformation.pdf>

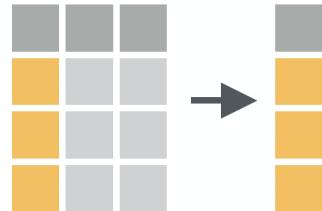
select()



```
select(consultas, id, edad, mins_espera)
```

```
# A tibble: 670 x 3
  id   edad  mins_espera
  <dbl> <dbl>      <dbl>
1     1     50          12
2     2     31          39
3     3     32          40
4     4     21          16
5     5     33           8
6     6     30          39
7     7     26          25
8     8     53          12
9     9     30          25
10    10    59          31
```

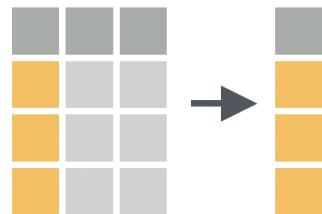
select()



```
consultas %>% select(id, edad, mins_espera)
```

```
# A tibble: 670 x 3
  id   edad  mins_espera
  <dbl> <dbl>      <dbl>
1     1     50          12
2     2     31          39
3     3     32          40
4     4     21          16
5     5     33           8
6     6     30          39
7     7     26          25
8     8     53          12
9     9     30          25
10    10    59          31
```

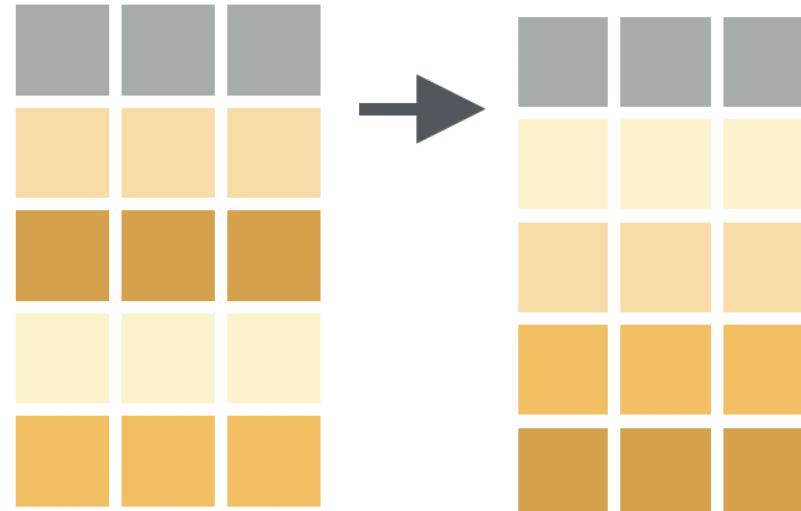
select()



```
consultas %>%  
  select(edad, edad35, everything())
```

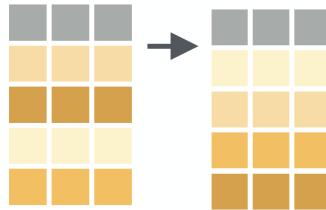
	# A tibble: 670 × 14	edad	edad35	id	embarazos	glucemia	presion_art	bmi	turno	dia_semana			
		<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<chr>	<chr>	<chr>	<chr>	<chr>
1	50	1	1	3	148	72	33.6	am	Vie				
2	31	0	2	0	85	66	26.6	am	Lun				
3	32	0	3	4	183	64	23.3	pm	Lun				
4	21	0	4	0	89	66	28.1	am	Vie				
5	33	0	5	0	137	40	43.1	am	Mar				
6	30	0	6	2	116	74	25.6	pm	Mar				
7	26	0	7	2	78	50	31	pm	Vie				
8	53	1	8	1	197	70	30.5	am	Lun				
9	30	0	9	2	110	92	37.6	am	Mie				

arrange()



Dplyr Cheat Sheet - <https://github.com/rstudio/cheatsheets/blob/master/data-transformation.pdf>

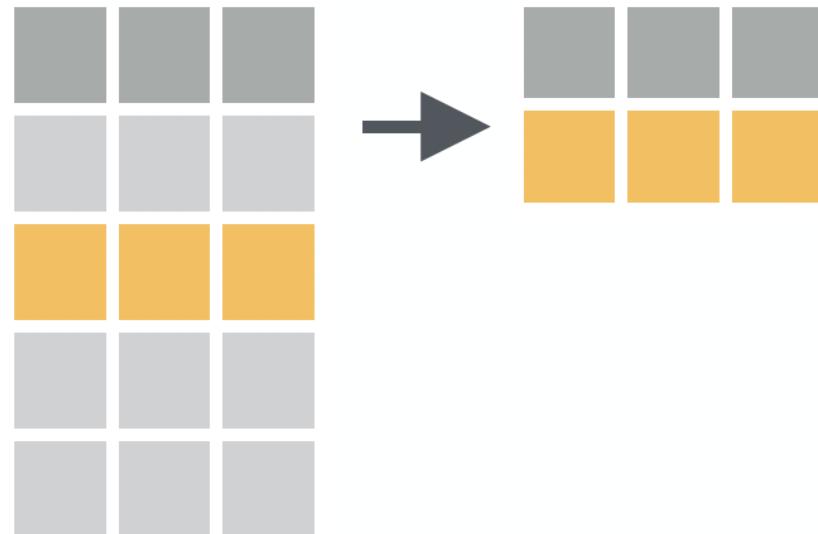
arrange()



```
arrange(consultas, mins_espera)
```

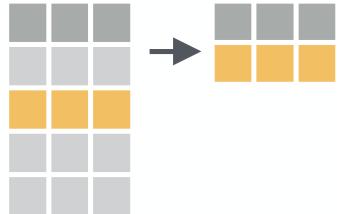
```
# A tibble: 670 x 14
  id mins_espera edad edad35 embarazos glucemia presion_art bmi tur...
  <dbl>      <dbl> <dbl>    <dbl>     <dbl>     <dbl>       <dbl> <dbl> <ch...
1 125          1     22      0        1       108         52   32.5 am
2 212          1     23      0        0       179         90   44.1 pm
3 240          1     40      1        2       122         78   23   pm
4 353          1     35      1        2       136         84   35   am
5 408          1     21      0        0       124         56   21.8 pm
6 447          1     64      1        4       120         78   25   am
7 590          1     53      1        4       156         86   24.8 am
8 52           2     24      0        1       141         58   25.4 am
9 142          2     27      0        0       114         80   44.2 am
10 102          4     21      0        2       99          76   23.2 am
# ... with 660 more rows, and 5 more variables: dia_semana <chr>,
```

filter()



Dplyr Cheat Sheet - <https://github.com/rstudio/cheatsheets/blob/master/data-transformation.pdf>

filter()



```
filter(consultas, mins_espera < 10)
```

```
# A tibble: 74 x 14
  id   edad mins_espera edad35 embarazos glucemia presion_art bmi tur...
  <dbl> <dbl>       <dbl>    <dbl>      <dbl>     <dbl>        <dbl> <dbl> <ch...
1    5     33          8      0         0      137         40  43.1 am
2   22     38          9      1         2      117         92  34.1 pm
3   31     26          9      0         2      180         64  34   am
4   46     41          8      1         4      187         68  37.7 pm
5   52     24          2      0         1      141         58  25.4 am
6   56     54          9      1         1      109         92  42.7 am
7   75     42          5      1         4       81         78  46.7 pm
8   84     22          8      0         0      151         60  26.1 am
9  102     21          4      0         2       99         76  23.2 am
10  110    23          5      0         0      118         58  33.3 am
# ... with 64 more rows, and 5 more variables: dia_semana <chr>,
```

Actividad



Seleccionar columnas y filtrar registros

Funciones de dplyr

Operador 'pipe': %>%

fx2(fx1(objeto))

fx1(objeto) %>% fx2()

Ejemplo de uso de pipes

Seleccionar columnas y filtrar observaciones

```
consultas %>%
  select(edad, bmi, mins_espera, glucemia) %>%
  filter(bmi < 20 & edad > 50)
```

```
# A tibble: 1 x 4
  edad    bmi  mins_espera  glucemia
  <dbl>   <dbl>      <dbl>     <dbl>
1     60    19.6        11       129
```

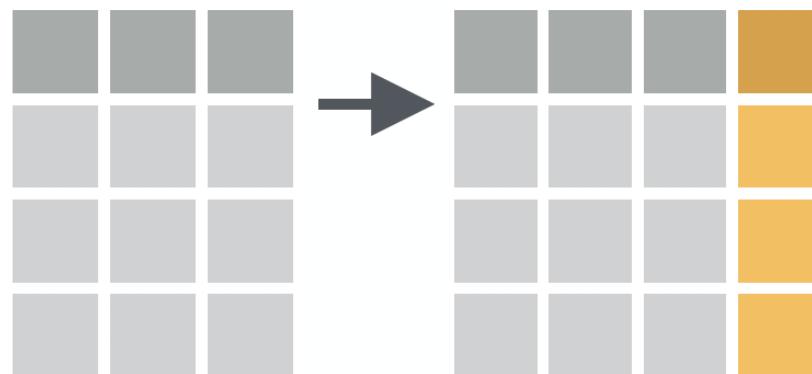
Ejemplo de uso de pipes

Seleccionar columnas y filtrar observaciones

```
consultas %>%
  select(edad, glucemia, dia_semana) %>%
  filter(dia_semana %in% c("Mie", "Jue")) %>%
  arrange(desc(glucemia))
```

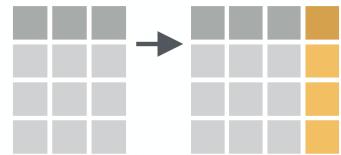
```
# A tibble: 267 x 3
  edad   glucemia dia_semana
  <dbl>     <dbl>    <chr>
1 28       198     Mie
2 39       197     Mie
3 62       197     Mie
4 41       196     Jue
5 57       196     Mie
6 29       196     Mie
7 55       195     Mie
8 31       195     Jue
9 41       194     Mie
```

mutate()



Dplyr Cheat Sheet - <https://github.com/rstudio/cheatsheets/blob/master/data-transformation.pdf>

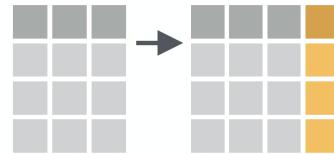
mutate()



```
mutate(consultas, horas_espera = mins_espera/60)
```

```
# A tibble: 670 x 15
  id    edad  edad35 embarazos glucemia presion_art     bmi turno dia_semana
  <dbl> <dbl>   <dbl>      <dbl>     <dbl>       <dbl> <dbl> <chr> <chr>
1     1     50       1          3      148        72  33.6 am    Vie
2     2     31       0          0       85        66  26.6 am    Lun
3     3     32       0          4      183        64  23.3 pm   Lun
4     4     21       0          0       89        66  28.1 am   Vie
5     5     33       0          0      137        40  43.1 am   Mar
6     6     30       0          2      116        74  25.6 pm   Mar
7     7     26       0          2       78        50  31.0 pm  Vie
8     8     53       1          1      197        70  30.5 am  Lun
9     9     30       0          2      110        92  37.6 am  Mie
10    10    59       1          0      189        60  30.1 pm  Vie
# ... with 660 more rows, and 6 more variables: hora_presente <dttm>,
#   hora_atencion <dttm>, mins_espera <dbl>, uso_portal <dbl>, urgencias <dbl>
```

mutate()



```
mutate(consultas, horas_espera = mins_espera/60) %>%
  select(mins_espera, horas_espera, everything())
```

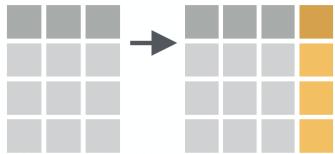
```
# A tibble: 670 x 15
  mins_espera horas_espera     id   edad edad35 embarazos glucemia presion_...
    <dbl>        <dbl> <dbl> <dbl>   <dbl>      <dbl>      <dbl>      <dbl>      <dbl>
1       12        0.2     1     50      1        3      148
2       39        0.65    2     31      0        0      85
3       40        0.667   3     32      0        4      183
4       16        0.267   4     21      0        0      89
5        8        0.133   5     33      0        0      137
6       39        0.65    6     30      0        2      116
7       25        0.417   7     26      0        2      78
8       12        0.2     8     53      1        1      197
9       25        0.417   9     30      0        2      110
10      31        0.517  10     59      1        0      189
# ... with 660 more rows, and 7 more variables: bmi <dbl>, turno <chr>, ...
```

Función `ifelse()`

3 argumentos

```
ifelse(condición, si_verdadero,  
si_falso)
```

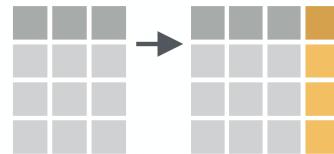
mutate()



```
consultas %>%
  mutate(partos_dicot = ifelse(embarazos > 0,
                               "atc_embarazos",
                               "nuligesta")) %>%
  select(embarazos, partos_dicot, everything())
```

```
# A tibble: 670 x 15
  embarazos partos_dicot     id   edad edad35 glucemia presion_art    bmi  tu
  <dbl> <chr>        <dbl> <dbl>   <dbl>      <dbl>       <dbl> <dbl> <dbl> <cl
1       3 atc_embaraz...     1     50      1      148        72  33.6 am
2       0 nuligesta       2     31      0       85        66  26.6 am
3       4 atc_embaraz...     3     32      0      183        64  23.3 pm
4       0 nuligesta       4     21      0       89        66  28.1 am
5       0 nuligesta       5     33      0      137        40  43.1 am
6       2 atc_embaraz...     6     30      0      116        74  25.6 pm
7       2 atc_embaraz...     7     26      0       78        50  31.0 pm
8       1 atc_embaraz...     8     53      1      197        70 100.0 am
```

mutate()



```
consultas %>%
  mutate(partos_dicot = ifelse(embarazos > 0,
                               "atc_embarazos",
                               "nuligesta")) %>%
  distinct(embarazos, partos_dicot) %>% arrange(embarazos)
```

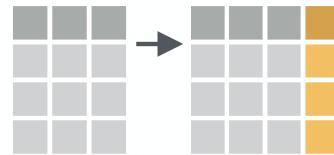
```
# A tibble: 5 x 2
  embarazos partos_dicot
  <dbl>     <chr>
1       0 nuligesta
2       1 atc_embarazos
3       2 atc_embarazos
4       3 atc_embarazos
5       4 atc_embarazos
```

Función

case_when()

```
case_when(condicion1 ~ resultado1,  
         condicion2 ~ resultado2,  
         condición3 ~ resultado3,  
         ...)
```

mutate()



```
consultas %>%  
  mutate(seguimiento =  
    case_when(glucemia > 126 ~ "consulta endocr.",  
              glucemia >= 110 ~ "estrecho",  
              glucemia >= 100 | edad >= 65 ~ "intermedio",  
              TRUE ~ "bajo"))
```

```
# A tibble: 8 x 3  
  edad glucemia seguimiento  
  <dbl>   <dbl> <chr>  
1     31      85  bajo  
2     21      89  bajo  
3     50     148  consulta endocr.  
4     32     183  consulta endocr.  
5     30     116  estrecho  
6     30     110  estrecho  
7     31     107  intermedio
```

Crear variable con case_when()

```
# A tibble: 8 x 3
  edad glucemia seguimiento
  <dbl>   <dbl>   <chr>
1     31      85  bajo
2     21      89  bajo
3     50     148 consulta endocr.
4     32     183 consulta endocr.
5     30     116 estrecho
6     30     110 estrecho
7     31     107 intermedio
8     33     103 intermedio
```

Tidyverse: función summarise()

Medidas de resumen

Medidas de resumen generales

Sintaxis de **summarise()**

```
datos %>%
  summarise(nombre_resultado1 = formula1,
            nombre_resultado2 = formula2,
            ...)
```

Ejemplo de summarise()

Medidas de resumen global

```
consultas %>%  
  summarise(promedio_espera = mean(mins_espera))
```

```
# A tibble: 1 x 1  
  promedio_espera  
    <dbl>  
1             24.6
```

Ejemplo de summarise()

Medidas de resumen global

```
consultas %>%
  summarise(n = n(),
            promedio_espera = mean(mins_espera),
            maxima_espera = max(mins_espera))
```

```
# A tibble: 1 x 3
  n     promedio_espera maxima_espera
  <int>          <dbl>        <dbl>
1    670           24.6         58
```

Ejemplo de summarise()

Medidas de resumen por grupos:

```
consultas %>%
  group_by(dia_semana) %>%
  summarise(n = n(),
            espera_media = mean(mins_espera),
            espera_max = max(mins_espera))
```

Ejemplo de summarise()

Medidas de resumen por grupos:

```
consultas %>%  
  group_by(dia_semana) %>%  
  summarise(n = n(),  
            espera_media = mean(mins_espera),  
            espera_max = max(mins_espera))
```

```
# A tibble: 5 x 4  
  dia_semana     n  espera_media  espera_max  
  <chr>      <int>      <dbl>        <dbl>  
1 Jue          128       24.1         56  
2 Lun          151       25.8         58  
3 Mar          116       24.5         54  
4 Mie          139       20.6         50  
5 Vie          136       27.6         56
```

Combinaciones!

```
consultas %>%
  mutate(gestas = ifelse(embarazos > 0,
                        "atc_gestas",
                        "nuligesta")) %>%
  group_by(gestas) %>%
  summarise(n = n(),
            gluc_media = mean(glucemia),
            gluc_min = min(glucemia))
```

Combinaciones!

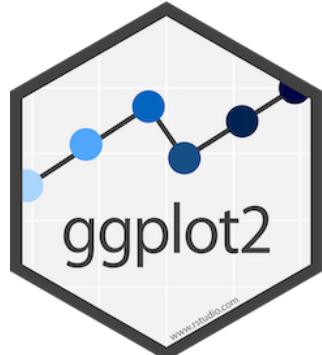
```
consultas %>%
  mutate(gestas = ifelse(embarazos > 0,
                        "atc_gestas",
                        "nuligesta")) %>%
  group_by(gestas) %>%
  summarise(n = n(),
            gluc_media = mean(glucemia),
            gluc_min = min(glucemia))

# A tibble: 2 x 4
gestas      n  gluc_media  gluc_min
<chr>     <int>    <dbl>     <dbl>
1 atc_gestas  440     124.       44
2 nuligesta   230     118.       57
```

Gráficos con Ggplot2



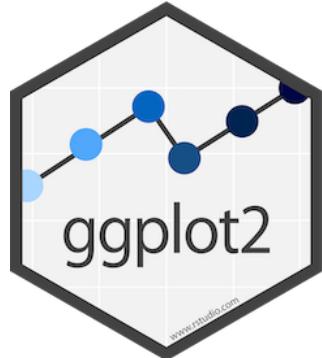
Ggplot2: gramática de gráficos



Componentes de un gráfico:

- **Datos**
- **'Geoms' o 'capas'**
- **'Mappings'** vínculo entre datos y geoms

Ggplot2: gramática de gráficos



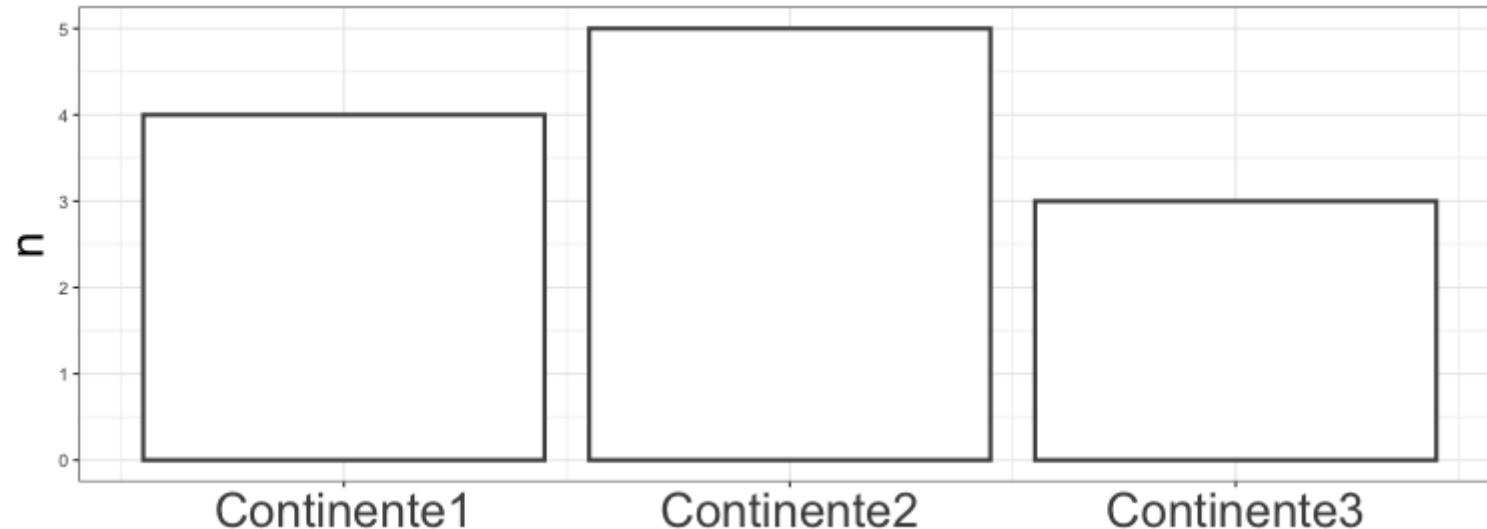
Crear base 'mundo'

```
mundo <- dslabs::gapminder
```

DSLabs package - *Rafael A. Irizarry, Amy Gill*

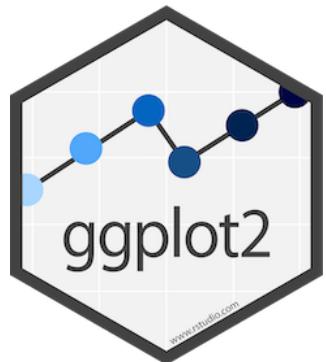
Ggplot2: sintaxis

Gráfico de barras



Ggplot2: sintaxis

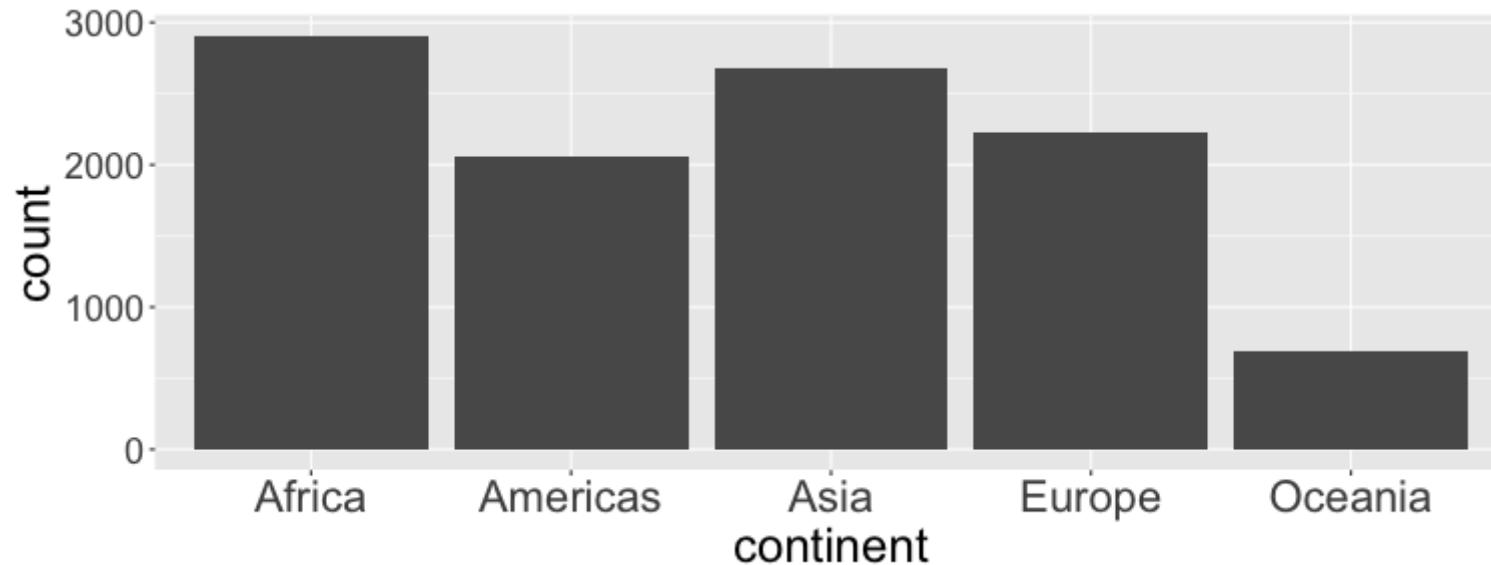
Gráfico de barras



Parámetro	valor
Data	mundo
Geom	bar
Mapping (aes)	x = continent

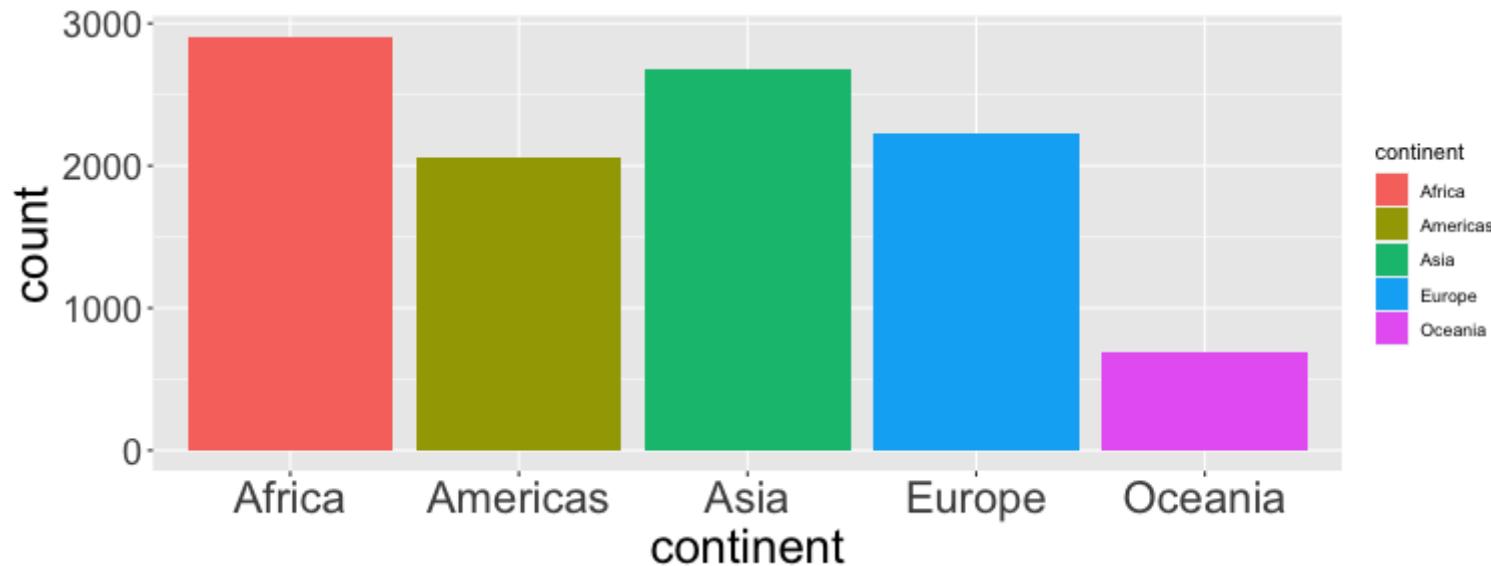
Ggplot2: gramática de gráficos

```
ggplot(data = mundo, aes(x = continent)) +  
  geom_bar()
```



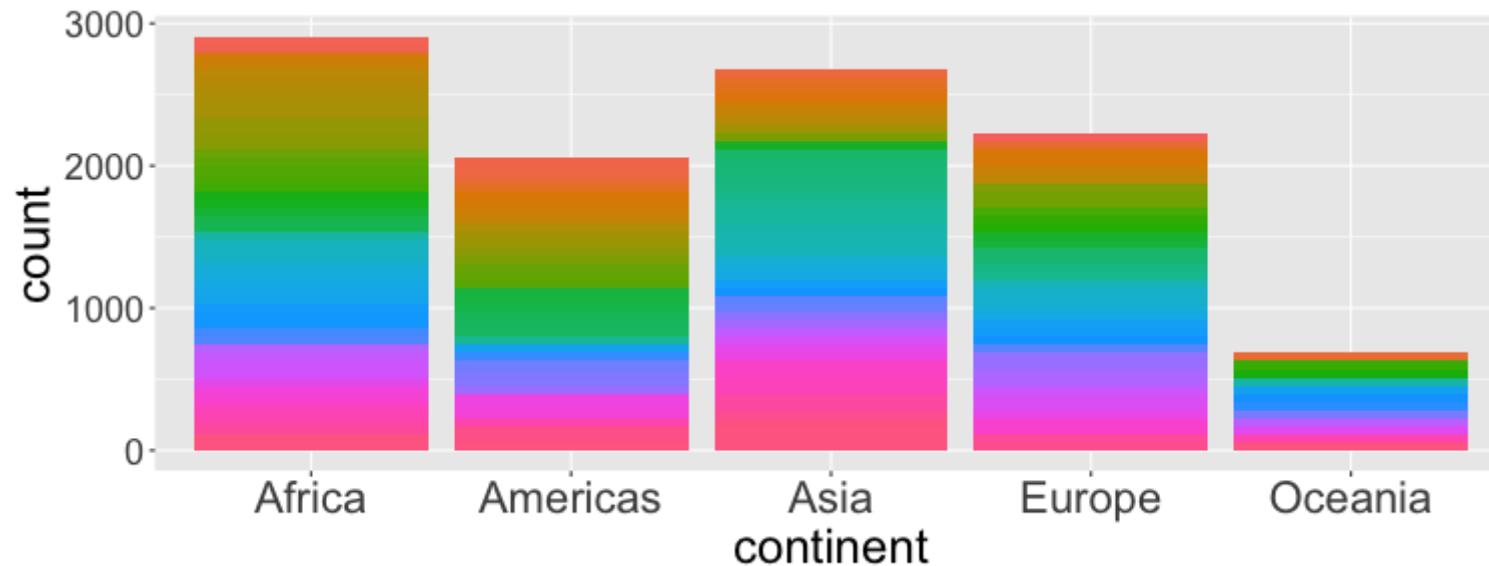
Ggplot2: gramática de gráficos

```
ggplot(data = mundo, aes(x = continent)) +  
  geom_bar(aes(fill = continent))
```



Ggplot2: gramática de gráficos

```
ggplot(data = mundo, aes(x = continent)) +  
  geom_bar(aes(fill = country))
```



Ggplot y pipes

%>%

¿Seleccionar variables y filtrar observaciones antes de graficar?

Ggplot y pipes

```
ggplot(data, aes(x, y))
```

```
data %>% ggplot(., aes(x, y))
```

```
data %>% ggplot(aes(x, y))
```

Ggplot y pipes

```
mando %>% dim()
```

```
[1] 10545      9
```

```
mando %>%
  filter(continent == "Oceania") %>%
  dim()
```

```
[1] 684      9
```

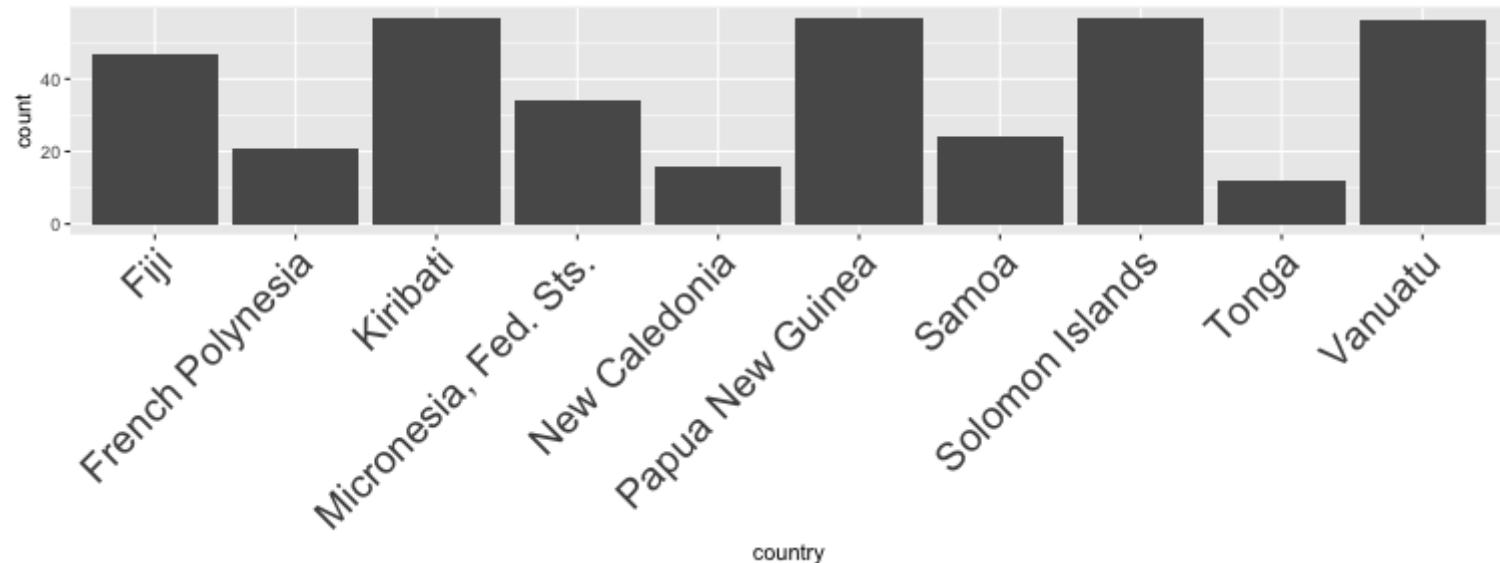
Ggplot y pipes

```
mundo %>%
  filter(continent == "Oceania" & life_expectancy < 65) %>%
  dim()
```

```
[1] 381    9
```

Ggplot y pipes

```
mundo %>%
  filter(continent == "Oceania" & life_expectancy < 65) %>%
  ggplot(aes(x = country)) +
  geom_bar()
```

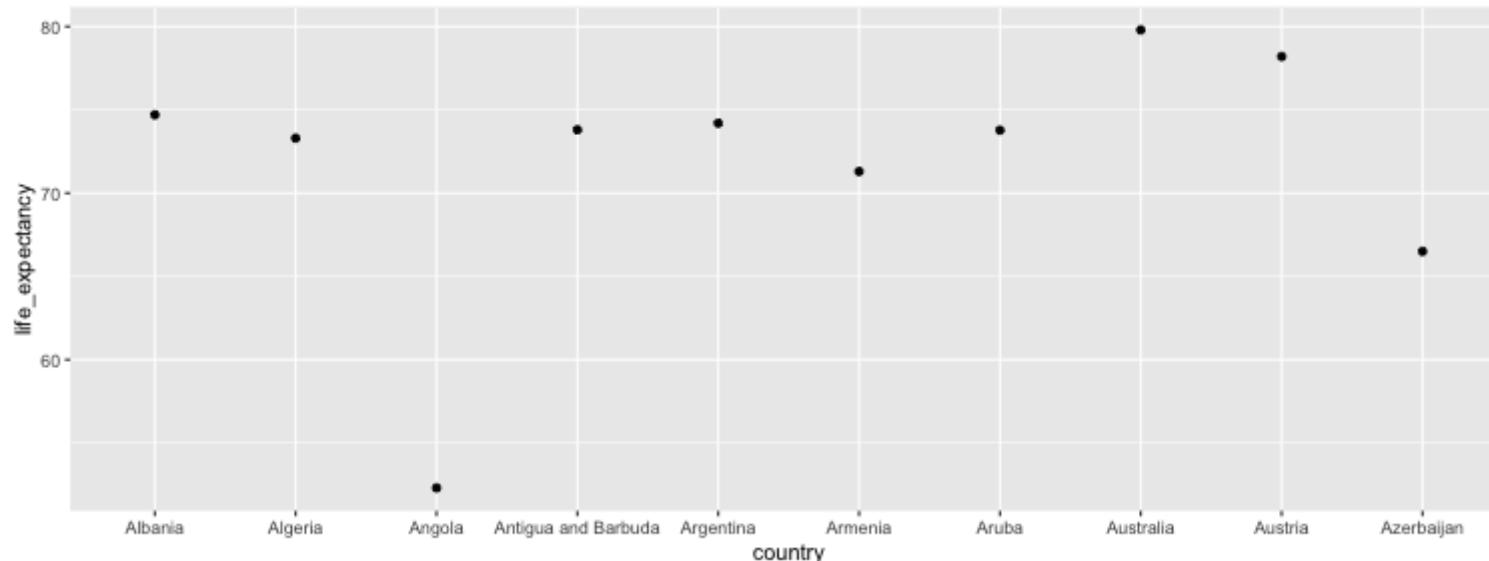


Ggplot y pipes

```
mundo %>%
  filter(str_detect(country, "^[A"] &
    year == 2000) %>%
  ggplot(aes(x = country,
             y = life_expectancy)) +
  geom_point()
```

Ggplot y pipes

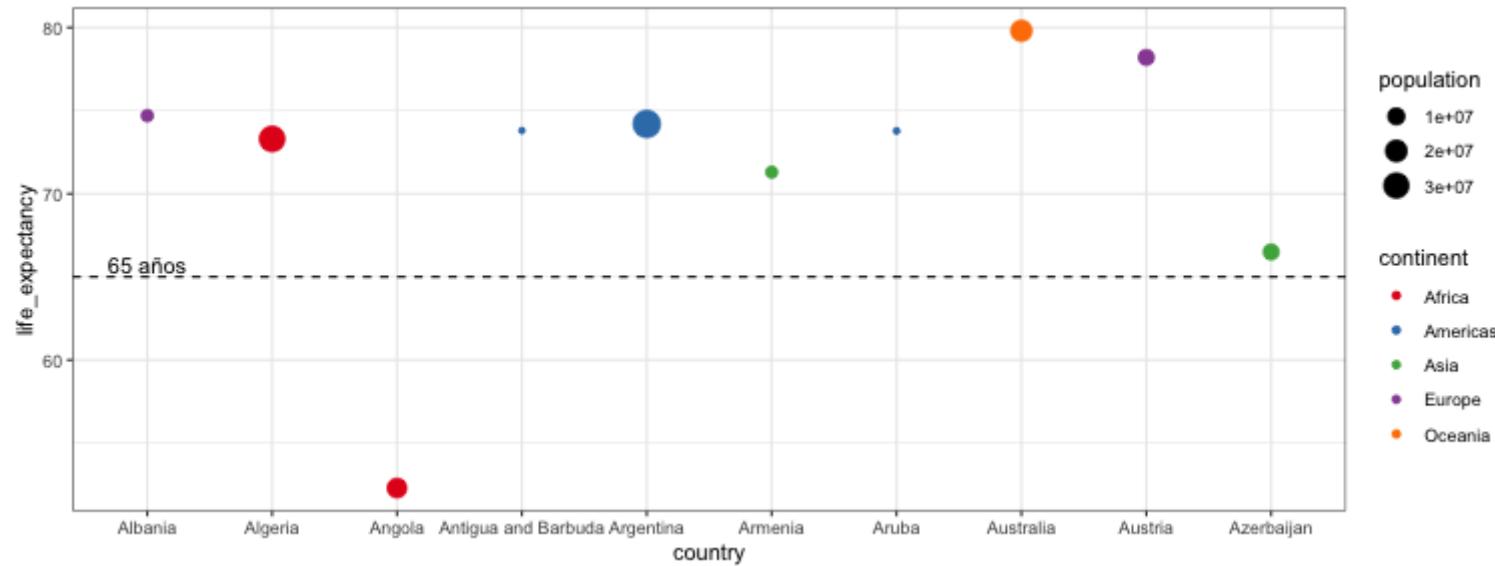
```
mundo %>%
  filter(str_detect(country, "^[A") &
    year == 2000) %>%
  ggplot(aes(x = country,
             y = life_expectancy)) +
  geom_point()
```



Ggplot y pipes

```
mundo %>%
  filter(str_detect(country, "^[A"] &
    year == 2000) %>%
  ggplot(aes(x = country,
             y = life_expectancy)) +
  geom_point(aes(size = population, color = continent)) +
  geom_hline(yintercept = 65, lty = "dashed") +
  annotate("text", y = 65, x = 1, vjust = -0.3,
          label = "65 años") +
  theme_bw()
```

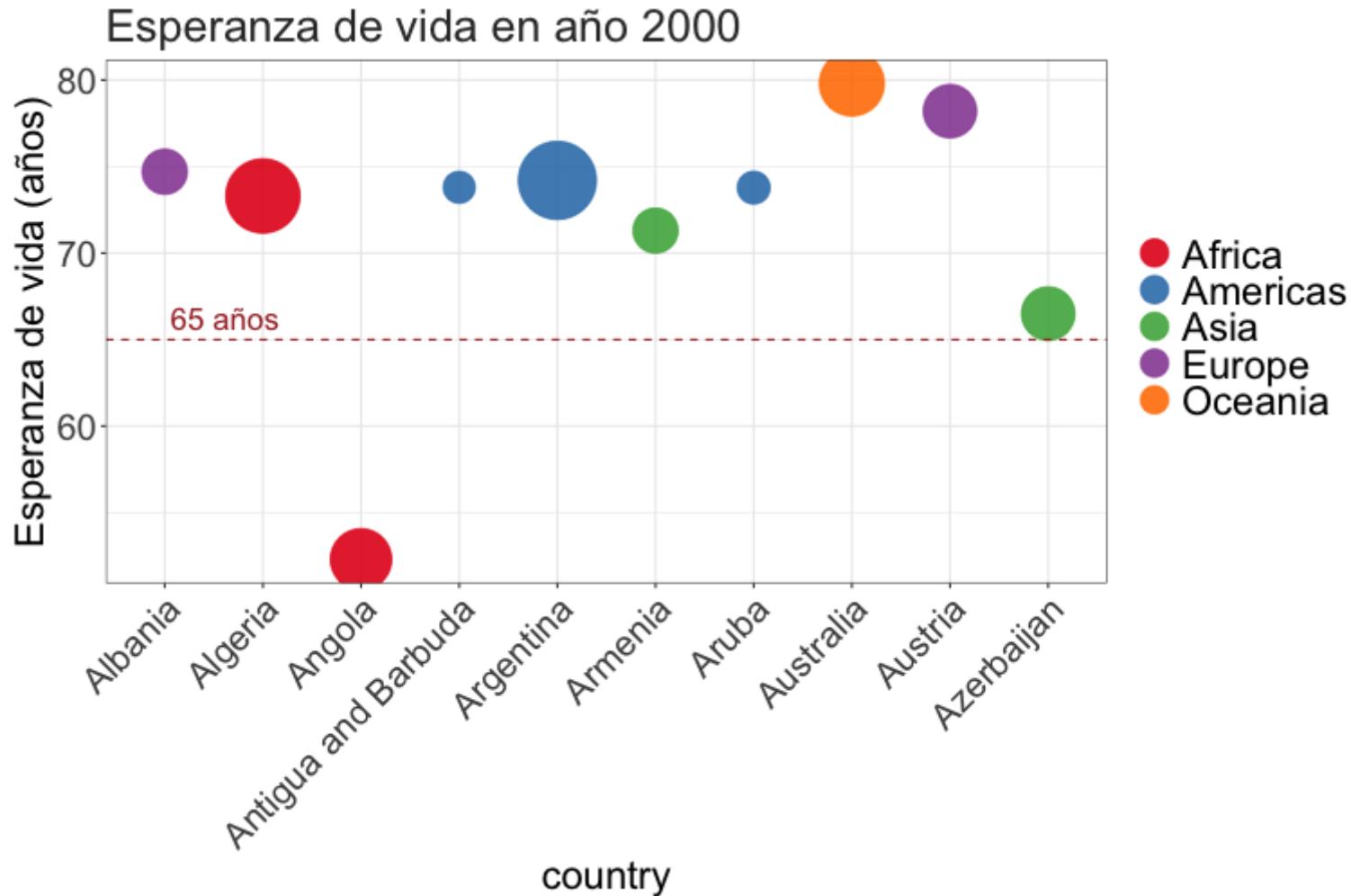
Ggplot y pipes



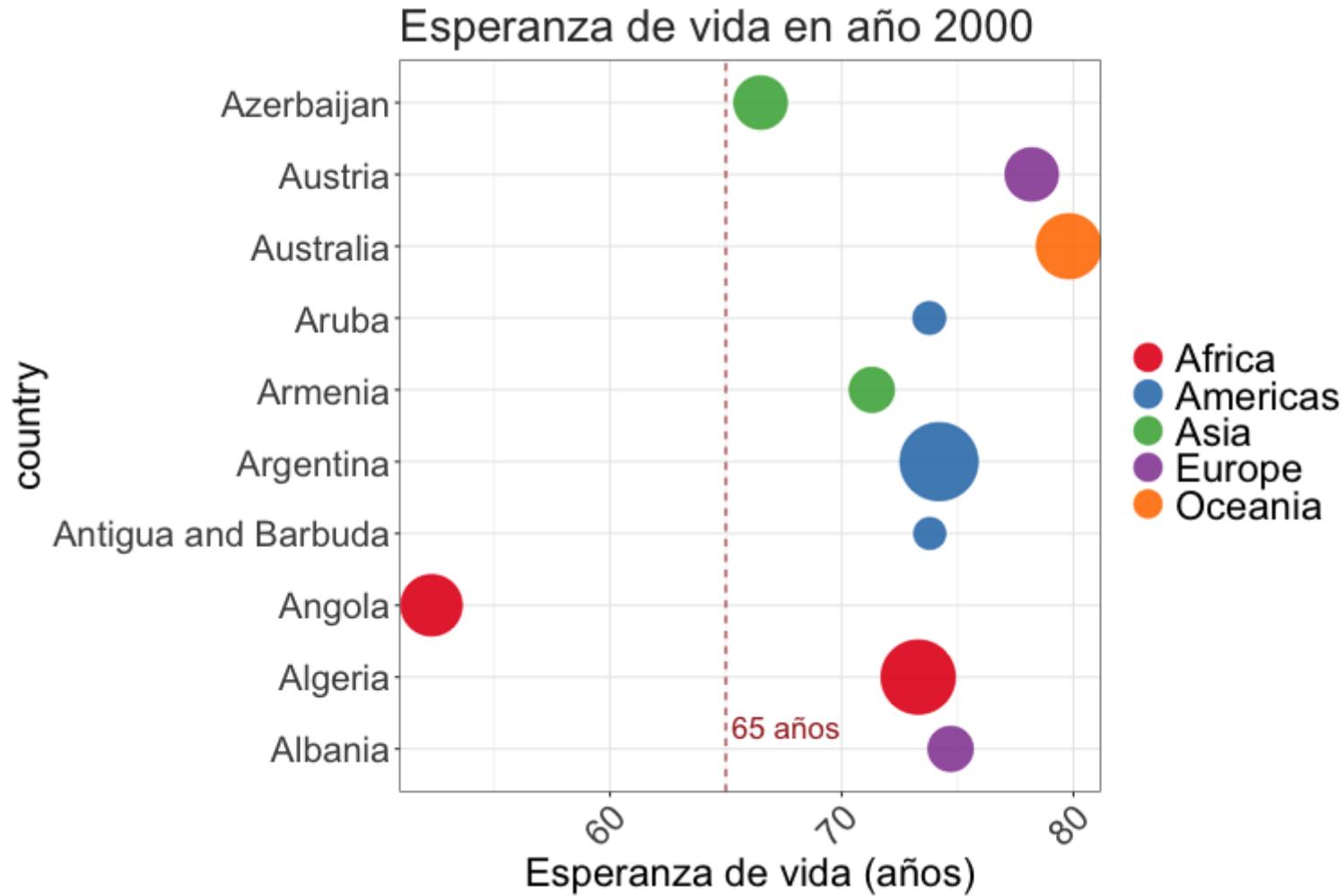
Ggplot y pipes

```
mundo %>%
  filter(str_detect(country, "^[A") &
    year == 2000) %>%
  ggplot(aes(x = country,
             y = life_expectancy)) +
  geom_point(aes(size = population, color = continent)) +
  geom_hline(yintercept = 65, lty = "dashed") +
  annotate("text", y = 65, x = 1, vjust = -0.3,
           label = "Esperanza de vida = 65 años") +
  labs(title = "Esperanza de vida en año 2000") +
  theme_bw() +
  theme(axis.text.x =
        element_text(angle = 45, hjust = 1, size = 14),
        axis.title = element_text(size = 15),
        legend.position = "bottom")
```

Ggplot y pipes



Ggplot y pipes

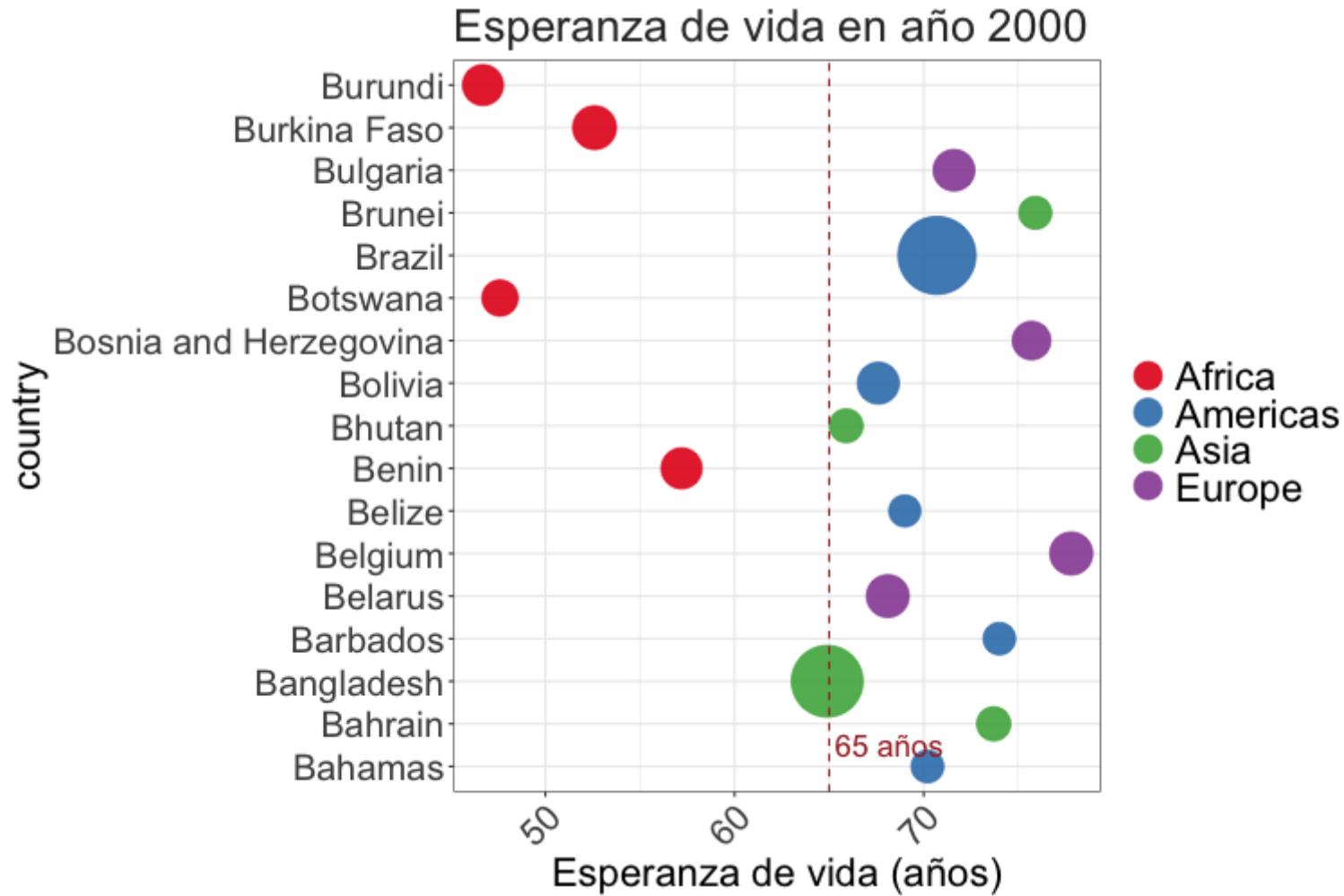


Filtrar B

```
mundo %>%
  filter(str_detect(country, "^[A") &
         year == 2000) %>%
ggplot(aes(x = country,
            y = life_expectancy)) + (...)
```

```
mundo %>%
  filter(str_detect(country, "^[B") &
         year == 2000) %>%
ggplot(aes(x = country,
            y = life_expectancy)) + (...)
```

Filtrar B



Tests de hipótesis

Tests de hipótesis

Pregunta

¿Es diferente el tiempo de espera los lunes y viernes vs mitad de semana?

Tests de hipótesis: Scripts para Investigación



```
## Cargar librerías ----  
  
## Cargar datos -----  
datos <- readxl::read_xlsx()  
  
# Exploración inicial ----  
  
# Crear variables -----  
datos %>% mutate()  
  
# Análisis descriptivo ---  
datos %>% count(var)  
  
# Comparación - test 1 ---  
t.test(var1 ~ grupo)
```

Tests de hipótesis: Scripts para Investigación



```
## Cargar librerías ----  
  
## Cargar datos -----  
datos <- readxl::read_xlsx()  
  
# Exploración inicial ----  
  
# Crear variables -----  
datos %>% mutate()  
  
# Análisis descriptivo ---  
datos %>% count(var)  
  
# Comparación - test 1 ---  
t.test(var1 ~ grupo)
```

Test de hipótesis: Exploración de datos

Repasar base de datos

```
dim(consultas)
```

```
[1] 670 14
```

Test de hipótesis: Exploración de datos

Repasar base de datos

```
consultas %>%
  select(mins_espera, dia_semana, hora_presente, hora_atencion) %>
  head(10)
```

```
# A tibble: 10 x 4
  mins_espera dia_semana hora_presente      hora_atencion
  <dbl>     <chr>        <dttm>          <dttm>
1       12  Vie  2019-11-08 11:03:00 2019-11-08 11:13:00
2       39  Lun  2019-10-07 09:35:00 2019-10-07 10:11:00
3       40  Lun  2019-11-04 15:01:00 2019-11-04 15:42:00
4       16  Vie  2019-10-25 10:29:00 2019-10-25 10:42:00
5        8  Mar  2019-11-05 10:09:00 2019-11-05 10:20:00
6       39  Mar  2019-10-15 14:38:00 2019-10-15 15:18:00
7       25  Vie  2019-10-18 14:37:00 2019-10-18 14:58:00
8       12  Lun  2019-10-14 09:46:00 2019-10-14 09:54:00
9       25  Mie  2019-10-16 09:23:00 2019-10-16 09:51:00
```

Test de hipótesis: Exploración de datos

¿Es diferente el tiempo de espera los lunes y viernes vs mitad de semana?

Test de hipótesis: Exploración de datos

¿Es diferente el tiempo de espera los lunes y viernes vs mitad de semana?

Variables de interés: dia_semana y mins_espera

Días Semana

Tiempo de Espera

Test de hipótesis: Exploración de datos

¿Es diferente el tiempo de espera los lunes y viernes vs mitad de semana?

Variables de interés: dia_semana y mins_espera

Días Semana

Tiempo de Espera

Test de hipótesis: Exploración de datos

Verificar días de la semana disponibles y valores vacíos de ambos

```
consultas %>%
  distinct(dia_semana)
```

```
# A tibble: 5 x 1
  dia_semana
  <chr>
1 Vie
2 Lun
3 Mar
4 Mie
5 Jue
```

Test de hipótesis: Exploración de datos

Verificar días de la semana disponibles y valores vacíos de ambos

```
consultas %>%
  summarise(n_vacios_dias = sum(is.na(dia_semana)),
            n_vacios_espera = sum(is.na(mins_espera)))
```

```
# A tibble: 1 x 2
  n_vacios_dias n_vacios_espera
  <int>           <int>
1          0              0
```

```
sum(is.na(consultas$mins_espera))
```

```
[1] 0
```

Test de hipótesis: Exploración de datos

```
consultas %>%  
  count(dia_semana)
```

```
# A tibble: 5 x 2  
  dia_semana     n  
  <chr>       <int>  
1 Jue            128  
2 Lun            151  
3 Mar            116  
4 Mie            139  
5 Vie            136
```

Test de hipótesis: Exploración de datos

```
consultas %>%
  count(dia_semana) %>%
  mutate(perc = 100*n/sum(n))
```

```
# A tibble: 5 x 3
  dia_semana     n   perc
  <chr>       <int> <dbl>
1 Jue            128  19.1
2 Lun            151  22.5
3 Mar            116  17.3
4 Mie            139  20.7
5 Vie            136  20.3
```

Test de hipótesis: Exploración de datos

¿Es diferente el tiempo de espera los lunes y viernes vs mitad de semana?

Variables de interés: dia_semana y mins_espera

Días Semana

Tiempo de Espera

Test de hipótesis: Exploración de datos

summary()

```
summary(consultas$mins_espera)
```

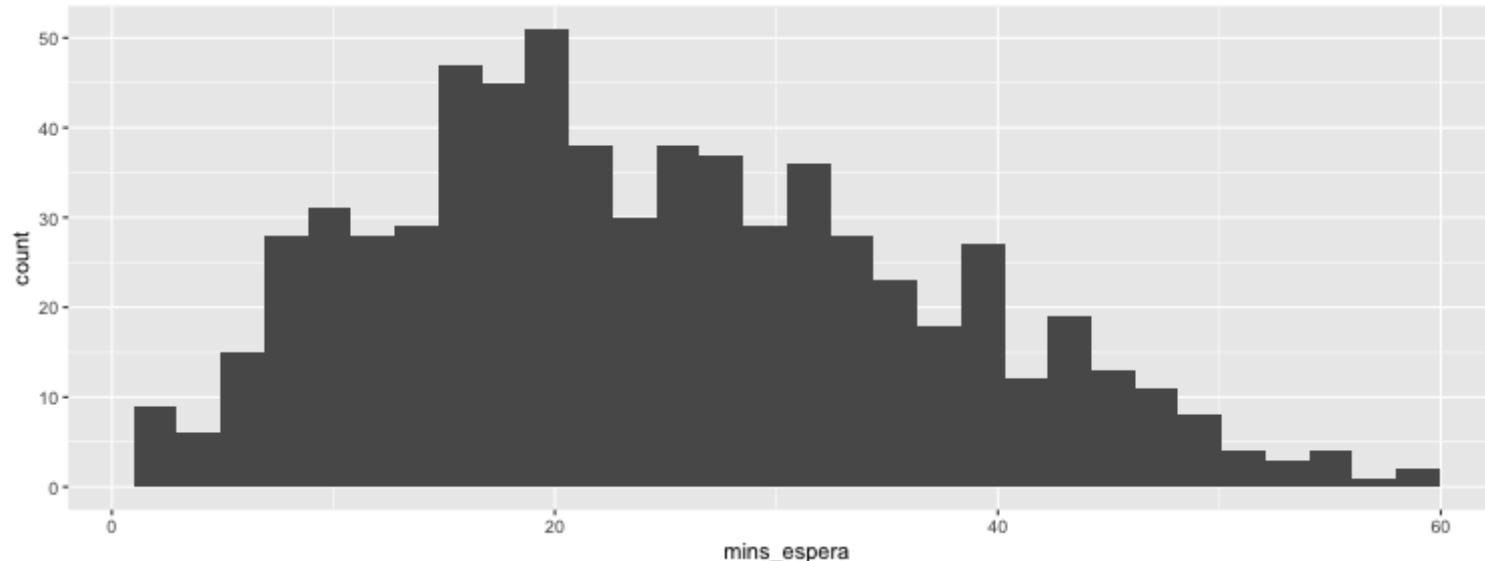
Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
1.00	16.00	23.00	24.55	33.00	58.00

Test de hipótesis: Exploración de datos

```
consultas %>%
  ggplot(aes(x = mins_espera)) +
  geom_histogram()
```

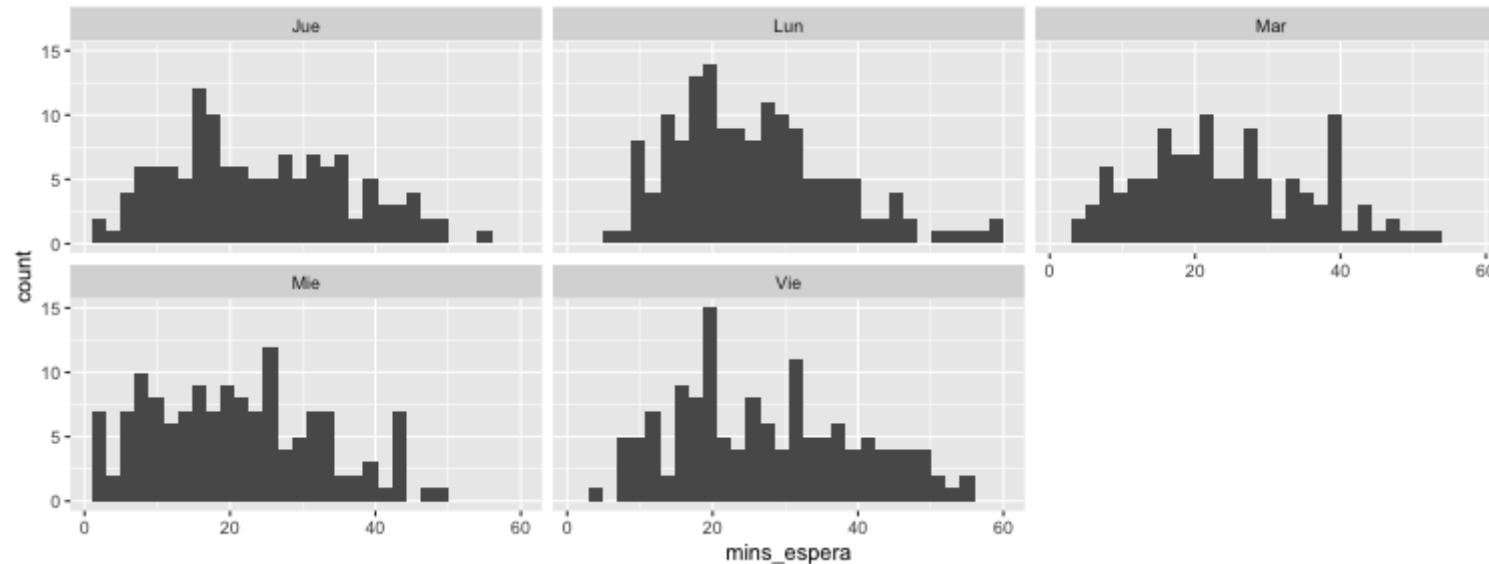
Test de hipótesis: Exploración de datos

```
consultas %>%
  ggplot(aes(x = mins_espera)) +
  geom_histogram()
```



Test de hipótesis: Exploración de datos

```
consultas %>%
  ggplot(aes(x = mins_espera)) +
  geom_histogram() +
  facet_wrap(~dia_semana)
```



Tests de hipótesis: Scripts para Investigación



```
## Cargar librerías ----  
  
## Cargar datos -----  
datos <- readxl::read_xlsx()  
  
# Exploración inicial ----  
  
# Crear variables -----  
datos %>% mutate()  
  
# Análisis descriptivo ---  
datos %>% count(var)  
  
# Comparación - test 1 ---  
t.test(var1 ~ grupo)
```

Test de hipótesis: Crear grupos

```
consultas %>%
  mutate(extr_semana =
    ifelse(dia_semana %in% c("Lun", "Vie"),
           "extremo",
           "centro"))
```

```
# A tibble: 5 x 2
  dia_semana extr_semana
  <chr>       <chr>
1 Vie         extremo
2 Lun         extremo
3 Mar         centro
4 Mie         centro
5 Jue         centro
```

Test de hipótesis: Crear grupos

```
consultas <- consultas %>%
  mutate(extr_semana =
    ifelse(dia_semana %in% c("Lun", "Vie"),
           "extremo",
           "centro"))
```

```
dim(consultas)
```

```
[1] 670 15
```

Tests de hipótesis: Scripts para Investigación



```
## Cargar librerías ----  
  
## Cargar datos -----  
datos <- readxl::read_xlsx()  
  
# Exploración inicial ----  
  
# Crear variables -----  
datos %>% mutate()  
  
# Análisis descriptivo ---  
datos %>% count(var)  
  
# Comparación - test 1 ---  
t.test(var1 ~ grupo)
```

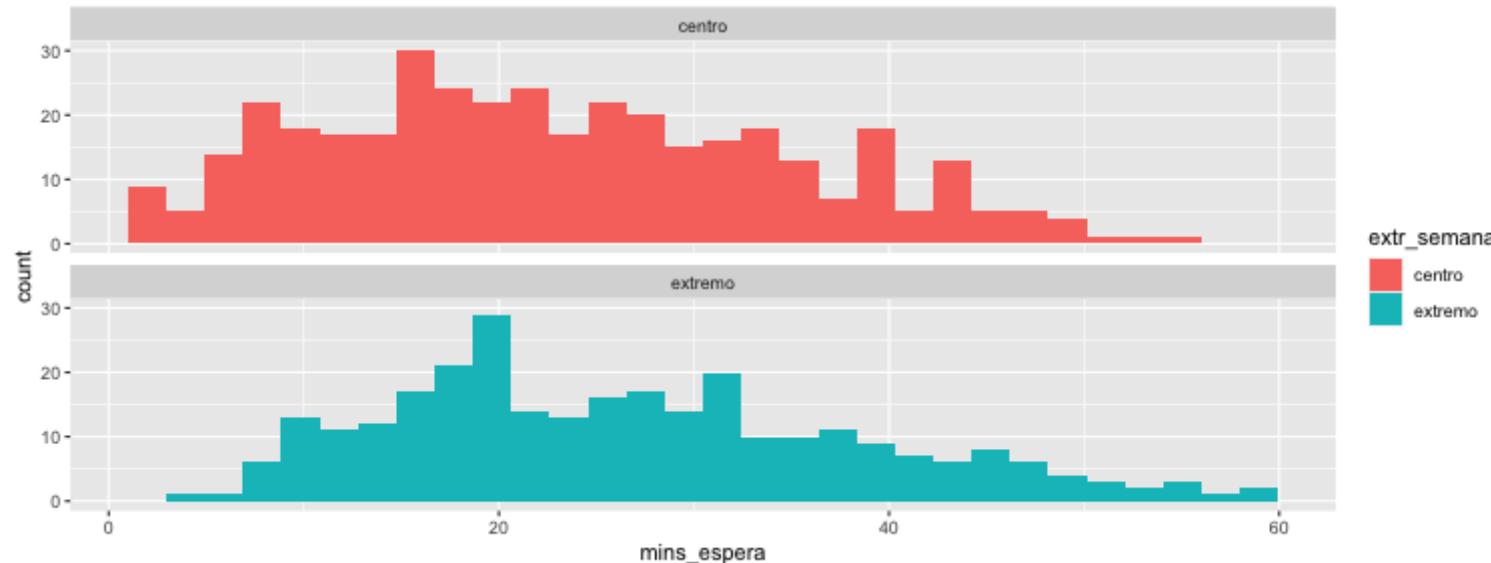
Test de hipótesis: Análisis descriptivo

```
consultas %>%
  group_by(extr_semana) %>%
  summarise(promedio_espera = mean(mins_espera),
            maxima_espera = max(mins_espera))

# A tibble: 2 x 3
  extr_semana promedio_espera maxima_espera
  <chr>           <dbl>          <dbl>
1 centro            23.0            56
2 extremo           26.7            58
```

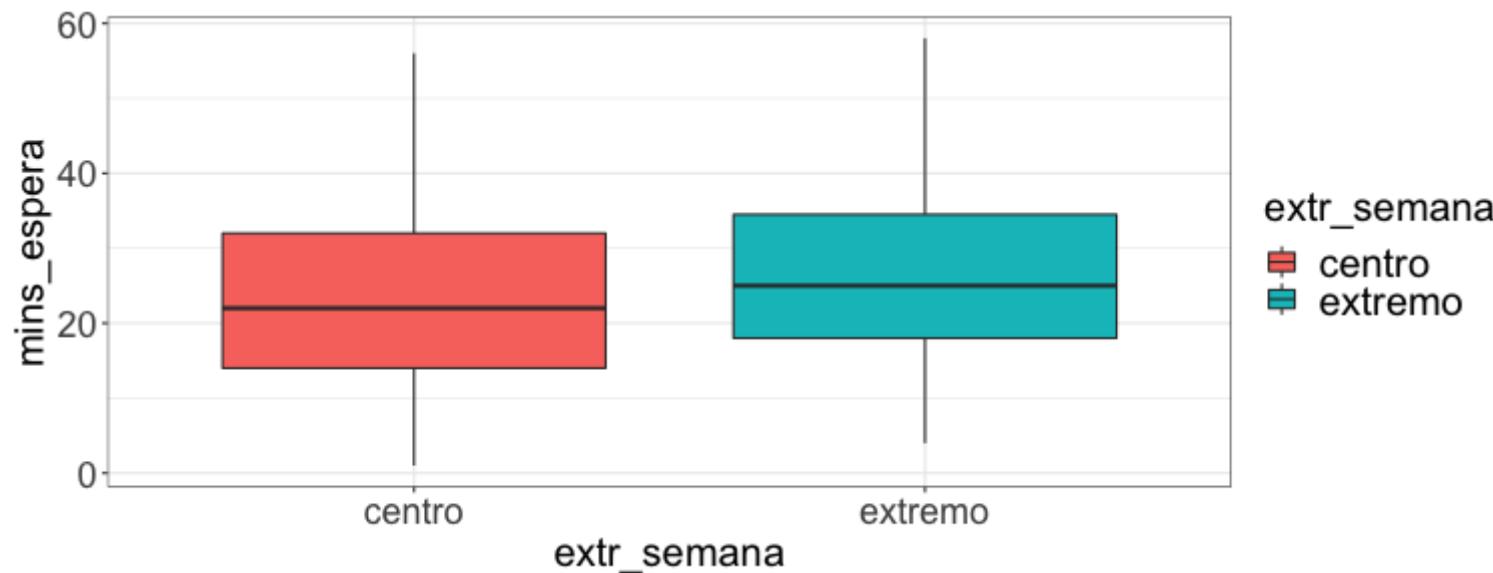
Test de hipótesis: Análisis descriptivo

```
consultas %>%
  ggplot(aes(x = mins_espera)) +
  geom_histogram(aes(fill = extr_semana)) +
  facet_wrap(~extr_semana, ncol = 1)
```



Test de hipótesis: Análisis descriptivo

```
consultas %>%
  ggplot(aes(x = extr_semana,
             y = mins_espera)) +
  geom_boxplot(aes(fill = extr_semana))
```



Tests de hipótesis: Scripts para Investigación



```
## Cargar librerías ----  
  
## Cargar datos -----  
datos <- readxl::read_xlsx()  
  
# Exploración inicial ----  
  
# Crear variables -----  
datos %>% mutate()  
  
# Análisis descriptivo ---  
datos %>% count(var)  
  
# Comparación - test 1 ---  
t.test(var1 ~ grupo)
```

Test de hipótesis: Test

```
t.test(consultas$mins_espera ~ consultas$extr_semana)
```

Welch Two Sample t-test

```
data: consultas$mins_espera by consultas$extr_semana
t = -3.9455, df = 619.32, p-value = 8.875e-05
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
-5.596872 -1.876914
sample estimates:
mean in group centro mean in group extremo
22.9530                26.6899
```

Test de hipótesis: Resultados

```
t.test(consultas$mins_espera ~  
       consultas$extr_semana) ["p.value"]
```

```
$p.value  
[1] 8.875334e-05
```

```
resultado <-  
  t.test(consultas$mins_espera ~ consultas$extr_semana)  
  
sprintf("%1.5f",  
       as.numeric(resultado$p.value))
```

```
[1] "0.00009"
```

Extras

RMarkdown



R y Big Data



Extras 1: RMarkdown



- Interfaz de Notebooks
- Tejido de código y **Texto**
- Múltiples lenguajes (R, SQL, Python...)
- Output estático o dinámico: HTML, .pdf, Word, Shiny

<https://rmarkdown.rstudio.com/>

~/Documents/rmarkdown - gh-pages - RStudio

rmarkdown

1-example.Rmd x ABC Knit Run

```
1 ---  
2 title: "Viridis Demo"  
3 output: html_document  
4 ---  
5  
6 ```{r include = FALSE}  
7 library(viridis)  
8 ````  
9  
10 The code below demonstrates two color palettes in the  
11 [viridis](https://github.com/sjmgarnier/viridis) package. Each  
12 plot displays a contour map of the Maunga Whau volcano in  
13 Auckland, New Zealand.  
14 ## Viridis colors  
15 image(volcano, col = viridis(200))  
16 ````  
17  
18 ## Magma colors  
19  
20 ```{r}  
21 image(volcano, col = viridis(200, option = "A"))  
22 ````  
23
```

Environment History Build Git

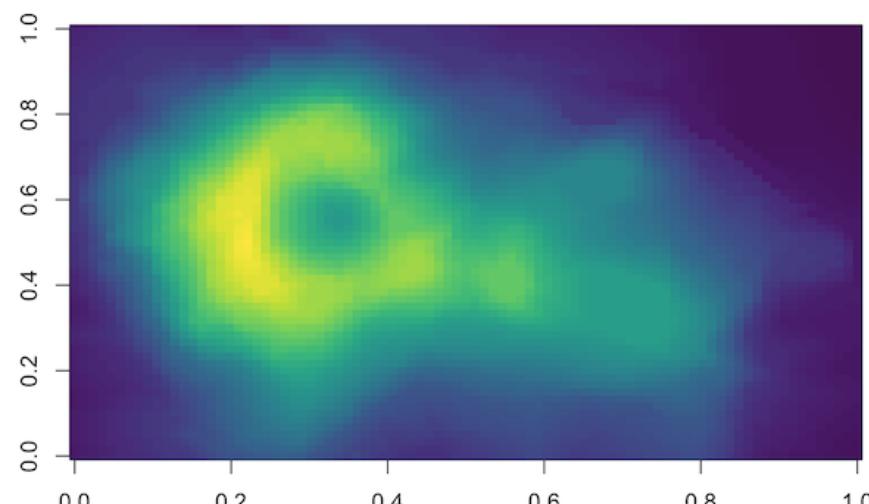
Files Plots Packages Help Viewer

Viridis Demo

The code below demonstrates two color palettes in the `viridis` package. Each plot displays a contour map of the Maunga Whau volcano in Auckland, New Zealand.

Viridis colors

```
image(volcano, col = viridis(200))
```



A contour plot showing the elevation of the Maunga Whau volcano in Auckland, New Zealand. The plot uses the Viridis color palette, transitioning from dark purple at the base to bright yellow-green at the peak. The x-axis and y-axis both range from 0.0 to 1.0, with major ticks at 0.0, 0.2, 0.4, 0.6, 0.8, and 1.0. The volcano's peak is located approximately at coordinates (0.3, 0.7).

Magma colors



A contour plot showing the elevation of the Maunga Whau volcano in Auckland, New Zealand. The plot uses the Magma color palette, transitioning from dark purple at the base to bright yellow-green at the peak. The plot is visually similar to the one above but with a different color scheme. The x-axis and y-axis both range from 0.0 to 1.0, with major ticks at 0.0, 0.2, 0.4, 0.6, 0.8, and 1.0. The volcano's peak is located approximately at coordinates (0.3, 0.7).

1:1 Viridis Demo R Markdown

Console

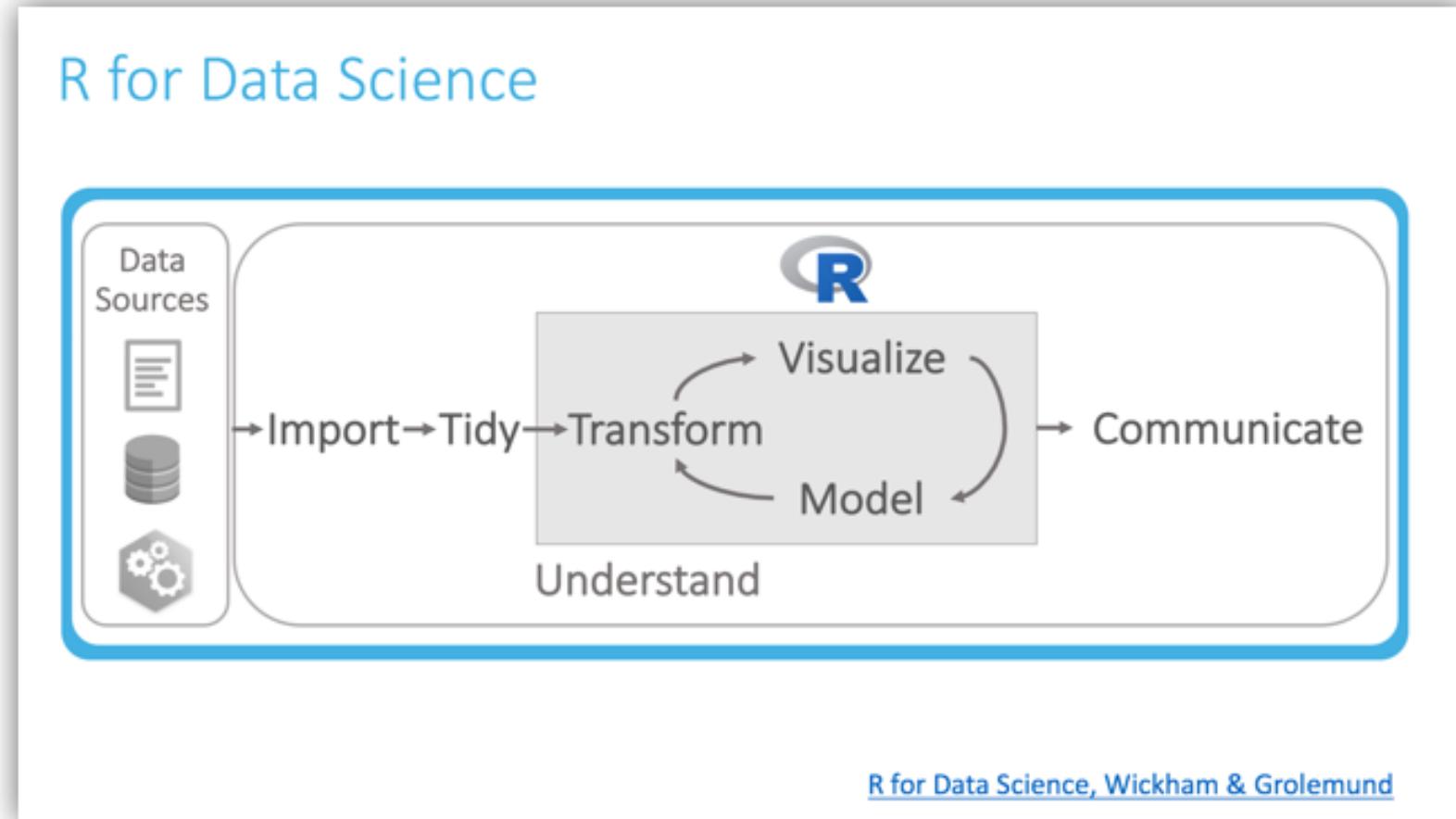
165 / 183

Extras 2: R y Big Data

¿Qué tamaño es Big Data?

Manejo de memoria en R.

R y Big Data: manejo local



R y Big Data

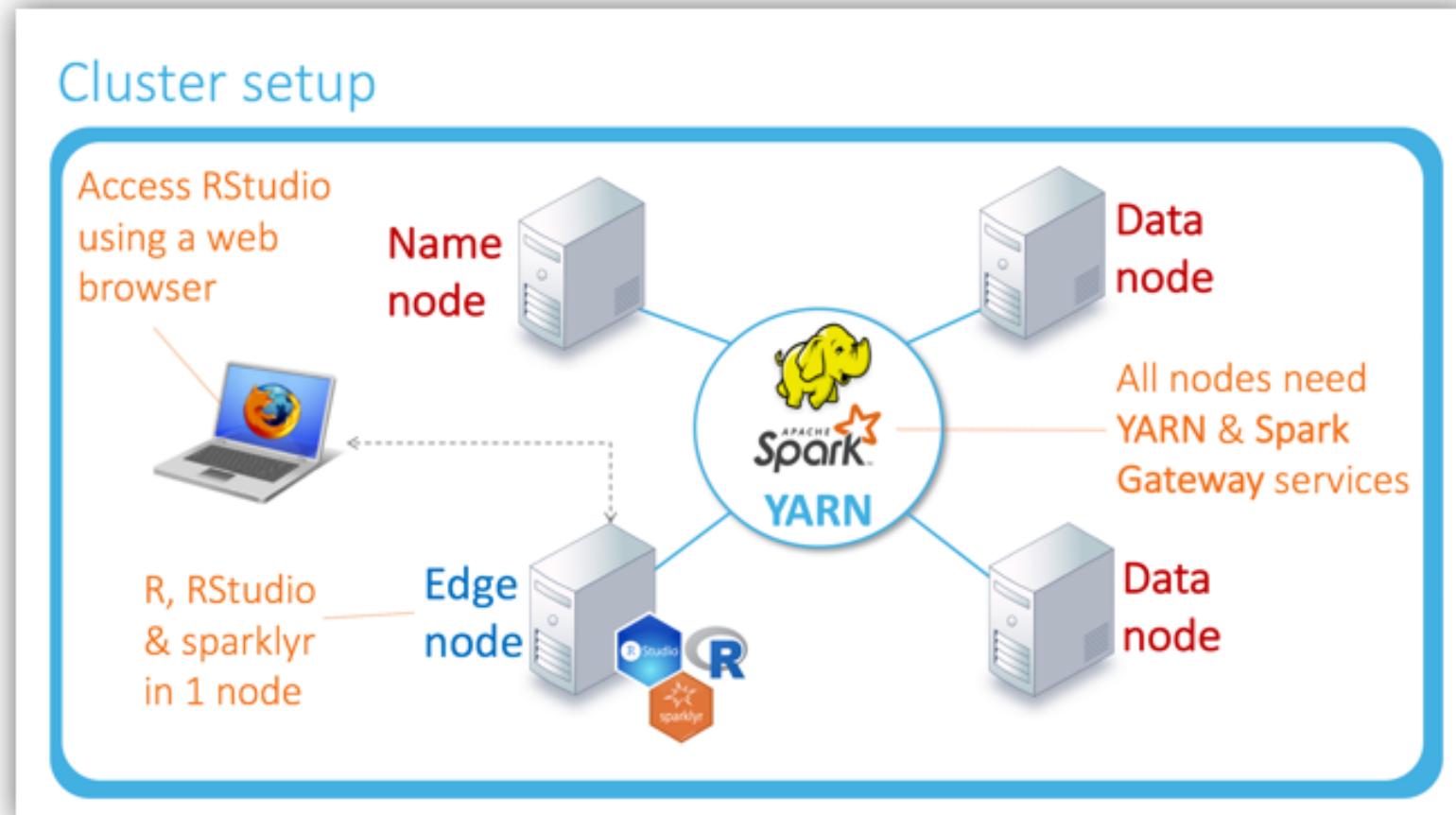
Spark as an Analysis Engine

Solution: Use sparklyr to access & analyze the data inside Spark.
Only bring results into R.



<https://spark.rstudio.com/guides/data-lakes/>

R y Big Data



<https://spark.rstudio.com/guides/data-lakes/>

Recursos Educativos



Recursos online

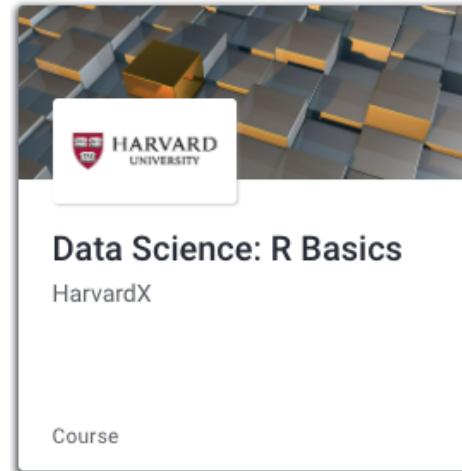
- Cursos online
- R Cheat Sheets
- Libros/e-books
- Comunidades de usuarios

Recursos online

- Cursos online
- R Cheat Sheets
- Libros/e-books
- Comunidades de usuarios

Cursos online

EdX, Coursera, Udemy...



A thumbnail for a HarvardX course. It features the Harvard University logo at the top left. The main title is "Data Science: R Basics" in bold black text, followed by "HarvardX" in smaller text. At the bottom, it says "Course". The background of the thumbnail shows a 3D perspective view of grey and gold cubes.

Recursos online

- Cursos online
- R Cheat Sheets
- Libros/e-books
- Comunidades de usuarios

Cheat Sheets

<https://rstudio.com/resources/cheatsheets/>

Dates and times with lubridate :: CHEAT SHEET

Date-times

2017-11-28 12:00:00 2017-11-28 12:00:00:000 12:00:00
A date-time is a point on the timeline, stored as the number of days since 1970-01-01 00:00:00 UTC.
`dt <- as_datetime("1511879400")` `d <- as_date("17/6")` `t <- hms(as.hms("05"))`
`dt <- as_datetime("2017-11-28 12:00 UTC")` `ds <- as_date("2017-11-28")` `tt <- hms(as.hms("05"))`
`ds <- as_date("2017-11-28")` `dt <- as_datetime(ds)` `tt <- hms(as.hms("05"))`
`ds <- as_date("2017-11-28")` `dt <- as_datetime(ds)` `tt <- hms(as.hms("05"))`

PARSE DATE/TIMES (Convert strings or numbers to date-times)
1. Identify the part (y, m, d, h, m, s, ms) and separate them by colons if your date has more than one colon.
2. Use the function below where name replicates the order. Each accepts a wide variety of input formats.

2017-11-28T12:00:00 ymd_hms(), ymd_hm(), ymd_h(), ymd(), hms()
2017-22-12 10:00:00 ymd(), ymd_hm(), ymd_h(), ymd()
11/22/2017 10:00:00 ymd_hms(), ymd_hm(), ymd_h(), ymd(), hms()
1 Jan 2017 12:00:00 ymd_hms(), ymd_hm(), ymd_h(), ymd(), hms()
20170101 ymd(), ymd_hm(), ymd_h(), ymd()
July 4th, 2000 ymd(), ymd_hm(), ymd_h(), ymd()
4th of July '99 ymd(), ymd_hm(), ymd_h(), ymd()
2001 03 ymd_hm(), ymd_hm(), ymd_h(), ymd()
2:01 ymd_hm(), ymd_hm(), ymd_h(), ymd()

2017.5 date_decimal(decimal, tz = "UTC")
tz <- "EST" current_time_in_tz(tz)
today() <- "Current date in a tz (default is system tz); today()
fast_time() <- "Fast time in a tz (default is system tz); fast_time()
force_tz(datetime, "US/Eastern", "US/Pacific")
parse_date_time() <- easier_stptime()
parse_date_time("%d/%m/%y")

2018-01-31 11:59:59 date(x) Date component, date(dt)
year(x) Year (ISO 8601 year, ISO 8601 week, ISO 8601 week-year)
month(x) Month (ISO 8601 month, month of year)
month(dt, label) Month, month(dt)
day(x) Day of month (day(dt))
week(x) Day of week (week(dt))
hour(x) Hour, hour(dt)
minute(x) Minutes, minute(dt)
second(x) Seconds, second(dt)
week(dt, year) Week of the year, week(dt)
is_leap(dt) Is it a leap year? (is_leap(dt))
is_leap(dt) Is it a leap year? (is_leap(dt))
quarter(dt, year = FALSE) Quarter, quarter(dt)
quarter(dt) Quarter (quarter(dt))
semester(dt, year = FALSE) Semester (semester(dt))
am(dt) Is it in the am? (am(dt))
pm(dt) Is it in the pm? (pm(dt))
dst(dt) Is it daylight savings? (dst(dt))
leap_year(dt) Is it a leap year? (leap_year(dt))
update(object, ..., simple = FALSE)
update(dt, mdy = 2, hour = 12)

Round Date-times

`floor_date(x, unit = "second")`
Round down to nearest unit.
`floor_date(x, unit = "month")`
`round_date(x, unit = "second")`
Round to nearest unit.
`round_date(x, unit = "month")`
`ceiling_date(x, unit = "second")`
change on boundary (NA/NULL)
Round up to nearest unit.
`ceiling_date(x, unit = "month")`
`rollback(date, n, to, first = FALSE, period, units = "TRIM")`
Roll back to the last day of the previous month (rollback(dt))

Stamp Date-times

`stamp(date) and stamp_time()`
Stamp a date/timestamp from an example string and return a new function that will apply the template to date-times. Also
stamp(date) and stamp_time().
1. Define a template, e.g. a function
 `stamp("Croatian Sunday, Jan 17, 1999 3:34")`
2. Apply the template to dates
 `stamp("2010-04-05")`
 `#> [1] "Croatian Monday, Apr 05, 2010 00:00"`

Time Zones

R recognizes ~600 time zones. Each encodes the time zone, Daylight Saving Time, and historical calendar variations for an area. R assigns one time zone per vector.
Use the UTC time zone to avoid Daylight Savings.
`OlsonName()` Returns a list of valid time zone names. OlsonNames()

5:00	6:00
Mountain	Central
4:00	7:00
MT	CT
7:00	7:00
Mountain	Central
7:00	7:00

with_tz(datetime, tz = "") Get the same date/time in a new time zone (a new clock time), with_tz(dt, "US/Pacific")
force_tz(datetime, tz = "") Get the same clock time in a new time zone (a new clock time), force_tz("US/Pacific")

RStudio® is a trademark of RStudio, Inc. • CC BY SA RStudio • info@rstudio.com • +44 448 1212 • rstudio.com • Learn more at lubridate.tidyverse.org • Lubridate 1.6.0 • Updated 2021-12



Recursos online

- Cursos online
- R Cheat Sheets
- Libros/e-books
- Comunidades de usuarios

Cheat Sheets

<https://rstudio.com/resources/cheatsheets/>

Data Transformation with dplyr :: CHEAT SHEET

dplyr

Manipulate Cases

EXTRACT CASES

Row functions return a subset of rows as a new table.

filter(data, ...) Extract rows that meet logical criteria. filter(iris, Sepal.Length > 7)

distinct(data, ..., keep = all = FALSE) Remove rows with duplicate values.

sample_n(data, n) Randomly sample n rows.

sample_frac(tbl, size = 1, replace = FALSE, weight = NULL, prob = parent.frame()\$prob) Randomly sample size fraction of rows.

sample_size(tbl, size, replace = FALSE, weight = NULL, prob = parent.frame()\$prob) Randomly sample size rows. sample_n() and sample_size() detect size rows. sample_n(n, 10, replace = TRUE)

slice(data, ...) Select rows by position. slice(iris, 10:25)

top_n(data, n, wt) Select and order top n entries (by group if grouped data). top_n(iris, 5, Sepal.Length)

Manipulate Variables

EXTRACT VARIABLES

Column functions return a set of columns as a new vector or table.

pull(data, var = -1) Extract column values as a vector. Choose by name or index. pull(iris, Sepal.Length)

select(data, ...) Extract column as a table. Also `select_if()`, `select_vars()`, `select_if()`.

ends_with(match) Starts with match.

starts_with(match) Starts with match.

Use these helpers with `match` (e.g. `select_if(matches("mpg"))`)

contains(match) num_range(prefs, range) e.g. `mpg cyl`

ends_with(match) one_of(.) e.g. `Species`

matches(match) starts_with(match)

MAKE NEW VARIABLES

These apply `vectorized` functions to columns. Vectorized funs take vectors as input and return vectors of the same length as output (see back).

vectorized function

Logical and boolean operators to use with filter()

< <= == != | & xor()

See `base::logic` and `?Comparison` for help.

Group Cases

Use `group_by()` to create a "grouped" copy of a table. dplyr functions will manipulate each "group" separately and then combine the results.

mtcars %>%
 group_by(cyl) %>%
 summarise(mpg = mean(mpg))

group_by(data, ..., add = FALSE) Returns grouped copy of table grouped by ...

ungroup(....) Returns ungrouped copy of table ungroup(g, iris)

ARRANGE CASES

arrange(data, ...) Order rows by values of a column or columns (low to high), use with `desc` to flip order. arrange(mtcars, mpg, desc(mpg))

mutate_all(tbl, funs, ...) Apply funs to every column of a table. mutate_all(mtcars, log10(.))

transmute(data, ...) Compute new column(s), drop others. transmute(mtcars, gpm = 1/mpg)

mutate_all(tbl, funs, ..., add = TRUE) Apply funs to every column of a table and add them to the end. mutate_all(mtcars, log10(.))

mutate_at(tbl, cols, funs, ..., add = TRUE) Apply funs to specific columns of a table. mutate_at(mtcars, 1, log10(.))

arrange(mtcars, mpg, desc(mpg))

ADD CASES

add_rownames(data, ..., before = NULL, after = NULL) Add one or more rows to a table. add_rownames(iris, 1, 2, existing = 1:132)

add_row(data, ..., n) Add one or more rows to a table. add_row(iris, 1, n = 2, existing = 1:132)

rename(data, ...) Rename columns. rename(iris, length = Sepal.Length)

R Studio

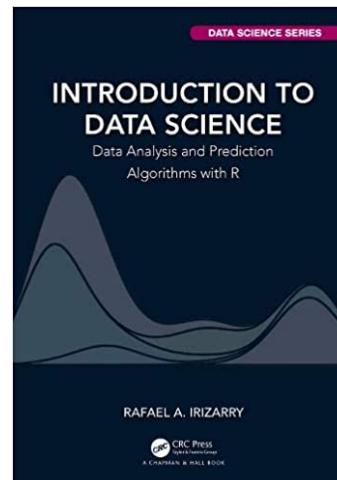
RStudio® is a trademark of RStudio, Inc. • CC BY SA RStudio • info@rstudio.com • 844-448-1212 • RStudio.com • Learn more with browser/greeter/package<-c("dplyr","tidyverse") | dplyr 0.7.0 + tidyverse 1.2.0 + Update 2017-03

Recursos online

- Cursos online
- R Cheat Sheets
- Libros/e-books
- Comunidades de usuarios

e-books(1)

**Introduction to Data Science:
Data Analysis and Prediction
Algorithms with R - Rafael Irizarry**



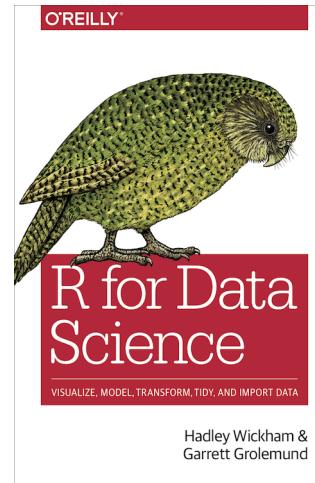
<https://rafalab.github.io/dsbook/>

Recursos online

- Cursos online
- R Cheat Sheets
- Libros/e-books
- Comunidades de usuarios

e-books(2)

R for Data Science - Hadley Wickham & Garrett Grolemund



<https://r4ds.hadley.nz/>

Recursos online

- Cursos online
- R Cheat Sheets
- Libros/e-books
- Comunidades de usuarios

e-books(2)

R para Ciencia de Datos - Hadley Wickham



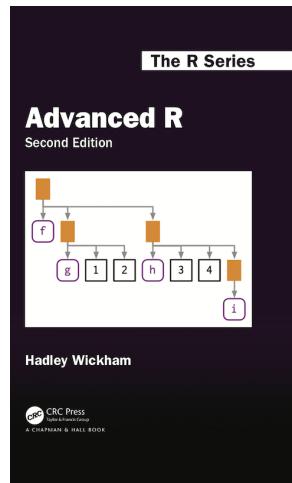
<https://es.r4ds.hadley.nz/>

Recursos online

- Cursos online
- R Cheat Sheets
- Libros/e-books
- Comunidades de usuarios

e-books(3)

Advanced R - Hadley Wickham



<https://adv-r.hadley.nz>

Recursos online

- Cursos online
- R Cheat Sheets
- Libros/e-books
- Comunidades de usuarios

Comunidades de usuarios

- <https://stackoverflow.com/>
- community.rstudio.com

Ejemplos reproducibles: [Reprex](#)

Comunidades de usuarios

Comunidades de usuarios

About 889,000 results (0.61 seconds)

[stackoverflow.com](#) › [questions](#) › [how-to-change-line-w...](#) ▾

[How to change line width in ggplot? - Stack Overflow](#)

Feb 10, 2013 — Line width in `ggplot2` can be changed with argument `size=` in `geom_`.

5 answers

[Increase the `size` of `line` in `geom_line` - Stack Overflow](#) 1 answer Nov 7, 2017

[ggplot specific thick `line` - Stack Overflow](#) 1 answer Apr 12, 2017

[Changing `width` of `line` using `ggplot` and `geom_line` ...](#) 1 answer Apr 4, 2017

[Change `line width` in `ggplot`, not `size` - Stack Overflow](#) 1 answer Oct 19, 2018

[More results from stackoverflow.com](#)

Comunidades de usuarios

¡Muchas Gracias!

Diapositivas creadas con el paquete de R `xaringan`. <https://github.com/yihui/xaringan>