Importar Datos: : **GUÍA RÁPIDA**

El **tidyverse** de R se basa en **datos ordenados** (tidy data) almacenados en **tibbles**, que son data frames mejorados.

El frente de esta hoja muestra cómo leer archivos de texto en R con **readr.**

El reverso muestra cómo crear tibbles con **tibble** y diseñar datos ordenados con **tidyr.**

OTROS TIPOS DE DATOS

Prueba uno de los siguientes paquetes para importar otros tipos de archivos

- **haven** – archivos SPSS, Stata y SAS
- **readxl** – archivos excel (.xls y .xlsx)
- **DBI** – base de datos
- **jsonlite** - json
- **xml2** - XML
- **httr** - Web APIs
- **rvest** - HTML (Web Scraping)

Guardar datos

Guardar **x**, un objeto de R, a **path**, una ruta de acceso a un archivo, como:

Archivo separado por comas
write_csv(xpath, na = "NA", append = FALSE, col_names = !append)

Archivo con separador arbitrario
write_delim(x path, delim = " ", na = "NA", append = FALSE, col_names = !append)

CSV para excel
write_excel_csv(x path, na = "NA", append = FALSE, col_names = !append)

Cadena (string) a archivo
write_file(x path, append = FALSE)

Vector de cadena a archivo, un elemento por línea
write_lines(x,path, na = "NA", append = FALSE)

Objecto a archivo RDS
write_rds(x, path, compress = c("none", "gz", "bz2", "xz"), ...)

Archivo separado por tabulaciones
write_tsv(x, path, na = "NA", append = FALSE, col_names = !append)

Leer datos tabulares - Estas funciones comparten estos argumentos:

read_*(file, col_names = TRUE, col_types = NULL, locale = default_locale(), na = c("", "NA"), quoted_na = TRUE, comment = "", trim_ws = TRUE, skip = 0, n_max = Inf, guess_max = min(1000, n_max), progress = interactive()**)**

a,b,c
1,2,3
4,5,NA

A	B	C
1	2	3
4	5	NA

Archivo separado por comas
read_csv("archivo.csv"**)**
Para generar archivo .csv ejecuta:
write_file(x = "a,b,c\n1,2,3\n4,5,NA", path = "archivo.csv")

a;b;c
1;2;3
4;5;NA

A	B	C
1	2	3
4	5	NA

Archivo separado por punto y coma
read_csv2("archivo2.csv"**)**
write_file(x = "a;b;c\n1;2;3\n4;5;NA", path = " archivo 2.csv")

a|b|c
1|2|3
4|5|NA

A	B	C
1	2	3
4	5	NA

Archivo con cualquier separador
read_delim("archivo.txt", delim = "|"**)**
write_file(x = "a|b|c\n1|2|3\n4|5|NA", path = "archivo.txt")

a b c
1 2 3
4 5 NA

A	B	C
1	2	3
4	5	NA

Archivos de ancho fijo
read_fwf("archivo.fwf", col_positions = c(1, 3, 5)**)**
write_file(x = "a b c\n1 2 3\n4 5 NA", path = "archivo.fwf")

Archivo separado por tabulaciones
read_tsv("archivo.tsv"**)** también con **read_table().**
write_file(x = "a\tb\tc\n1\t2\t3\n4\t5\tNA", path = "archivo.tsv")

ARGUMENTOS ÚTILES

a,b,c
1,2,3
4,5,NA

Archivo de ejemplo
write_file("a,b,c\n1,2,3\n4,5,NA", "archivo.csv")
f <- "archivo.csv"

A	B	C
1	2	3
4	5	NA

Sin encabezado
read_csv(f, **col_names = FALSE**)

x	y	z
A	B	C
1	2	3
4	5	NA

Proporcionar el encabezado
read_csv(f, **col_names = c("x", "y", "z")**)

1	2	3
4	5	NA

Saltar líneas
read_csv(f, **skip = 1**)

A	B	C
1	2	3

Leer un subconjunto
read_csv(f, **n_max = 1**)

A	B	C
NA	2	3
4	5	NA

Valores faltantes
read_csv(f, **na = c("1", ".")**)

Leer datos no tabulares

Leer un archivo en una sola cadena
read_file(file, locale = default_locale()**)**

Lee cada línea en una cadena
read_lines(file, skip = 0, n_max = -1L, na = character(), locale = default_locale(), progress = interactive()**)**

Lee ficheros de log estilo Apache
read_log(file, col_names = FALSE, col_types = NULL, skip = 0, n_max = -1, progress = interactive()**)**

Leer un archivo en un vector
read_file_raw(file**)**

Lee cada línea en un vector
read_lines_raw(file, skip = 0, n_max = -1L, progress = interactive()**)**

Tipos de datos

Las funciones de readr interpretan los tipos de cada columna y convierten los tipos cuando corresponde (pero NO convertirá cadenas en factores automáticamente).

Un mensaje muestra el tipo de cada columna en el resultado.

```
## Con especificación de columnas:
## cols(
##   age = col_integer(),
##   sex = col_character(),
##   earn = col_double()
## )
```

age es un entero
sex es un caracter
earn es un decimal (numerico)

1. Usa **problems()** para diagnosticar problemas.

x <- read_csv("archivo.csv"); problems(x)

2. Usa una col_ function para guiar el parseado.
   - **col_guess()** - el valor por defecto
   - **col_character()**
   - **col_double()**, **col_euro_double()**
   - **col_datetime(**format = ""**)** También **col_date(**format = ""**)**, **col_time(**format = ""**)**
   - **col_factor(**levels, ordered = FALSE**)**
   - **col_integer()**
   - **col_logical()**
   - **col_number()**, **col_numeric()**
   - **col_skip()**

```
x <- read_csv("archivo.csv", col_types = cols(
  A = col_double(),
  B = col_logical(),
  C = col_factor()))
```

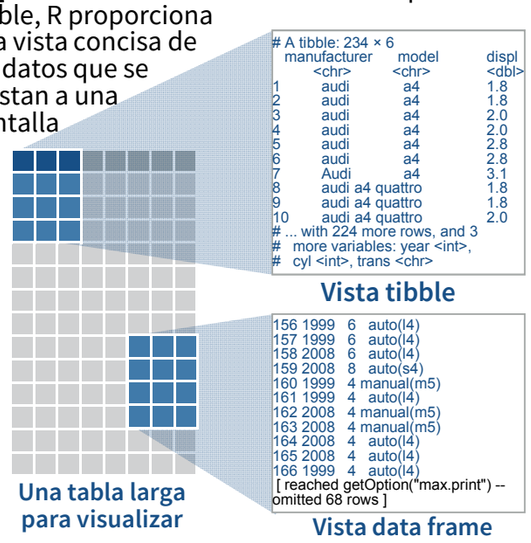
3. Sino, leé como vectores de caracteres y luego parsea con una parse_ function.
   - **parse_guess()**
   - **parse_character()**
   - **parse_datetime()** También **parse_date()** y **parse_time()**
   - **parse_double()**
   - **parse_factor()**
   - **parse_integer()**
   - **parse_logical()**
   - **parse_number()**

x$A <- parse_number(x$A)

Tibbles - un data frame mejorado

El paquete **tibble** proporciona una nueva clase S3 para almacenar datos tabulares, el tibble. Tibbles hereda la clase data frame, pero mejora tres comportamientos:

- **Creación de subconjuntos** - [siempre retorna un nuevo tibble, [[y $ siempre retornan un vector.
- **Sin coincidencias parciales**- se deben utilizar los nombres completos de las columnas al crear subconjuntos
- **Impresión en la consola** – Cuando imprimes un tibble, R proporciona una vista concisa de los datos que se ajustan a una pantalla



- Controla la apariencia por defecto con las opciones:

  options(tibble.print_max = n, tibble.print_min = m, tibble.width = Inf**)**

- Ver el conjunto de datos completo con **View()** o **glimpse()**
- Convierte a data frame con **as.data.frame()**

CONSTRUYE UN TIBBLE DE DOS MANERAS

tibble(…)
Construir por columnas.
tibble(x = 1:3, y = c("a", "b", "c"))

tribble(…)
Construir por filas.
tribble(~x, ~y,
1, "a",
2, "b",
3, "c")

Ambos crean este tibble

A tibble: 3 × 2

x	y
<int>	<chr>
1	a
2	b
3	c

as_tibble(x, …) Convierte un data frame a un tibble.

enframe(x, name = "name", value = "value") Convierte un vector con nombre en un tibble

is_tibble(x) Comprueba si x es un tibble.

Tidy Data con tidyr

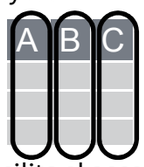
Tidy data es una forma de organizar datos tabulares. Proporciona una estructura de datos consistente entre paquetes

Una tabla es tidy si:

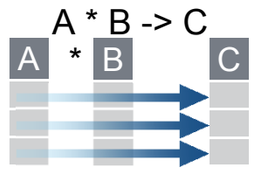
Cada **variable** es una **columna**

Cada **observación** o **caso** es una **fila**

Tidy data:



Facilita el acceso a las variables como vectores.



Preserva los casos durante las operaciones vectorizadas.

Remoldear Datos – cambiar la forma de los valores en una tabla

Usa **gather()** y **spread()** para reorganizar los valores de una tabla en una nueva forma.

gather(data, key, value, ..., na.rm = FALSE, convert = FALSE, factor_key = FALSE**)**

gather() mueve los nombres de las columnas a una columna **key** , reuniendo los valores de la columna en una sola columna **value**.

table4a

pais	1999	2000
A	0.7K	2K
B	37K	80K
C	212K	213K

→

pais	anio	casos
A	1999	0.7K
B	1999	37K
C	1999	212K
A	2000	2K
B	2000	80K
C	2000	213K

key value

gather(table4a, `1999`, `2000`, key = "anio", value = "casos")

spread(data, key, value, fill = NA, convert = FALSE, drop = TRUE, sep = NULL**)**

spread() mueve los valores únicos de una columna **key** como nombres de columnas, esparciendo los valores de una columna **value** a través de las nuevas columnas.

table2

pais	anio	tipo	cuenta
A	1999	cases	0.7K
A	1999	pop	19M
A	2000	cases	2K
A	2000	pop	20M
B	1999	cases	37K
B	1999	pop	172M
B	2000	cases	80K
B	2000	pop	174M
C	1999	cases	212K
C	1999	pop	1T
C	2000	cases	213K
C	2000	pop	1T

→

pais	anio	casos	pob
A	1999	0.7K	19M
A	2000	2K	20M
B	1999	37K	172M
B	2000	80K	174M
C	1999	212K	1T
C	2000	213K	1T

key value

spread(table2, tipo, cuenta)

Manejar Valores Perdidos

drop_na(data, ...**)**
Elimina las filas que contienen NA's en las columnas especificadas (…)

x

x1	x2
A	1
B	NA
C	NA
D	3
E	NA

→

x1	x2
A	1
D	3

drop_na(x, x2)

fill(data, ..., .direction = c("down", "up")**)**
Completa los NA's en las columnas especificadas (…) con el valor no-NA más cercano.

x

x1	x2
A	1
B	NA
C	NA
D	3
E	NA

→

x1	x2
A	1
B	1
C	1
D	3
E	3

fill(x, x2)

replace_na(data, replace = list(), ...**)**
Reemplaza NA's por columna.

x

x1	x2
A	1
B	NA
C	NA
D	3
E	NA

→

x1	x2
A	1
B	2
C	2
D	3
E	2

replace_na(x, list(x2 = 2))

Expandir tablas – crea rápidamente tablas con combinaciones de valores

complete(data, ..., fill = list()**)**
Completa los datos con combinaciones faltantes de las variables listadas en … usando los valores de fill o NA's.

complete(mtcars, cyl, gear, carb)

expand(data, ...**)**
Crea un nuevo tibble con todas las combinaciones posibles de los valores de las variables listadas en …

expand(mtcars, cyl, gear, carb)

Separar Celdas

Usa estas funciones para dividir o combinar celdas en valores individuales aislados.

separate(data, col, into, sep = "[^[:alnum:]]+", remove = TRUE, convert = FALSE, extra = "warn", fill = "warn", ...**)**

Separa cada celda en una columna para hacer varias columnas.

table3

pais	anio	tasa
A	1999	0.7K/19M
A	2000	2K/20M
B	1999	37K/172M
B	2000	80K/174M
C	1999	212K/1T
C	2000	213K/1T

→

pais	anio	casos	pob
A	1999	0.7K	19M
A	2000	2K	20M
B	1999	37K	172M
B	2000	80K	174M
C	1999	212K	1T
C	2000	213K	1T

separate(table3, tasa, sep = "/", into = c("casos", "pob"))

separate_rows(data, ..., sep = "[^[:alnum:].]+", convert = FALSE**)**

Separa cada celda en una columna para formar varias filas.

table3

pais	anio	tasa
A	1999	0.7K/19M
A	2000	2K/20M
B	1999	37K/172M
B	2000	80K/174M
C	1999	212K/1T
C	2000	213K/1T

→

pais	anio	tasa
A	1999	0.7K
A	1999	19M
A	2000	2K
A	2000	20M
B	1999	37K
B	1999	172M
B	2000	80K
B	2000	174M
C	1999	212K
C	1999	1T
C	2000	213K
C	2000	1T

separate_rows(table3, tasa, sep = "/")

unite(data, col, ..., sep = "_", remove = TRUE**)**

Une múltiples columnas en una única columna

table5

pais	siglo	anio
Afghan	19	99
Afghan	20	00
Brazil	19	99
Brazil	20	00
China	19	99
China	20	00

→

pais	anio
Afghan	1999
Afghan	2000
Brazil	1999
Brazil	2000
China	1999
China	2000

unite(table5, siglo, anio, col = "anio", sep = "")