Curso Primeros Pasos en R

Clase 3: Manipulación e importación de datos en R

Profesora: Ana María Alvarado

Pontificia Universidad Católica de Chile

Noviembre 2021

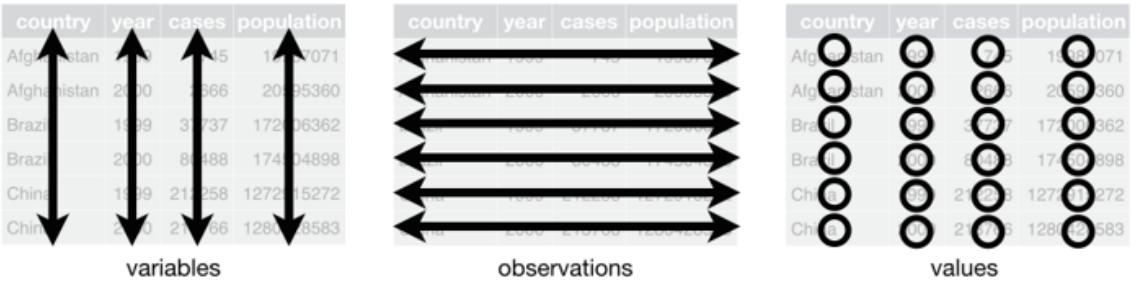
Clase 3: Manipulación e importación de datos

- Exploración de datos
- Importación de datos
- Problemas en la importación de datos
- Taller Práctico 1 y 2

Exploración de datos

Data frames y tibble

En el curso trabajaremos principalmente con bases de datos ordenadas:



Cuando carguemos bases de datos en R, estas pueden ser de dos tipos: `data frame` o `tibble`.

Las funciones `tibble()` y `as_tibble()` del paquete `tibble` nos permiten crear este tipo de objetos y transformar un data frame a este formato.

Exploración de datos

Los primeros comandos para explorar nuestros datos son los siguientes:

`head(df,k)` : Muestra los primeros k registros.

`tail(df,k)` : Muestra los últimos k registros.

`dim(df)` : Filas y columnas de un objeto.

`length(df)` : Número de objetos dentro del objeto df.

`class(df)` : Naturaleza o clase del objeto df.

`names(df)` : Nombres del objeto df.

`str(df)` : Muestra la estructura interna del objeto df.

Actividad práctica

Utilice las operaciones anteriores en el conjunto de datos **paises** de la librería **datos**.

- Se carga el data frame en el objeto **df**.

```
# install.packages("datos")
library(datos)
df <- datos::paises
```

- Se visualizan los primeros 3 valores del data frame.

```
head(df, 3)
```

```
## # A tibble: 3 x 6
##   pais       continente  anio esperanza_de_vida poblacion pib_per_capita
##   <fct>      <fct>      <int>             <dbl>     <int>          <dbl>
## 1 Afganistán Asia        1952              28.8   8425333           779.
## 2 Afganistán Asia        1957              30.3   9240934           821.
## 3 Afganistán Asia        1962              32.0  10267083           853.
```

Actividad práctica

- Se visualizan los últimos 3 valores del data frame.

```
tail(df, 3)
```

```
## # A tibble: 3 x 6
##   pais     continente  anio esperanza_de_vida poblacion pib_per_capita
##   <fct>    <fct>      <int>             <dbl>     <int>          <dbl>
## 1 Zimbabue África      1997              46.8  11404948           792.
## 2 Zimbabue África      2002              40.0  11926563           672.
## 3 Zimbabue África      2007              43.5  12311143           470.
```

- ¿Cuántas observaciones y variables tiene el data frame?

```
dim(df)
```

```
## [1] 1704    6
```

Recuerde que la primera posición representa el número de observaciones y la segunda posición el número de variables.

Actividad práctica

- ¿Cuántos objetos tiene el data frame?

```
length(df)
```

```
## [1] 6
```

- ¿Cuáles son los nombres de las variables del data frame?

```
names(df)
```

```
## [1] "pais"             "continente"       "anio"
## [4] "esperanza_de_vida" "poblacion"        "pib_per_capita"
```

- ¿Cuál es la clase del data frame?

```
class(df)
```

Actividad práctica

- ¿Cuál es la estructura interna del data frame?

```
## tibble [1,704 x 6] (S3: tbl_df/tbl/data.frame)
##  $ pais             : Factor w/ 142 levels "Afganistán","Albania",..: 1 1 1 1 1 1 1 1 1 1
##  $ continente       : Factor w/ 5 levels "África","Américas",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ anio             : int [1:1704] 1952 1957 1962 1967 1972 1977 1982 1987 1992 1997 ...
##  $ esperanza_de_vida: num [1:1704] 28.8 30.3 32 34 36.1 ...
##  $ poblacion        : int [1:1704] 8425333 9240934 10267083 11537966 13079460 14880372 12
##  $ pib_per_capita   : num [1:1704] 779 821 853 836 740 ...

## NULL
```

Análisis descriptivo

A modo exploratorio una de las principales funciones que permite realizar un resumen descriptivo de todas las variables en el data frame es:

```
summary(df)
```

```
##         pais        continente       anio      esperanza_de_vida
##  Afganistán:  12   África  :624   Min.   :1952   Min.   :23.60
##  Albania   :  12   Américas:300   1st Qu.:1966   1st Qu.:48.20
##  Argelia   :  12   Asia    :396   Median :1980   Median :60.71
##  Angola    :  12   Europa  :360   Mean   :1980   Mean   :59.47
##  Argentina :  12   Oceanía : 24   3rd Qu.:1993   3rd Qu.:70.85
##  Australia :  12                  Max.   :2007   Max.   :82.60
##  (Other)   :1632
##    poblacion         pib_per_capita
##  Min.   :6.001e+04   Min.   :   241.2
##  1st Qu.:2.794e+06   1st Qu.:  1202.1
##  Median :7.024e+06   Median :  3531.8
##  Mean   :2.960e+07   Mean   :  7215.3
##  3rd Qu.:1.959e+07   3rd Qu.:  9325.5
##  Max.   :1.319e+09   Max.   :113523.1
##
```

También, la función **skim** de la librería **skimr** puede ser de gran utilidad.

Valores no disponibles (NA)

En R los valores no disponibles (celdas vacías, valores perdidos, missing, etc) se representan con NA, el tener elementos nulos dentro de un objeto de R **no** afecta la clase de este, por ejemplo:

```
vec_ejemplo <- c(1, NA, 3, NA, 5)
class(vec_ejemplo)
```

```
## [1] "numeric"
```

```
library(tibble)
tib_ejemplo <- tibble(
  nombre = c("Claudio","Javiera","Elias", NA,"Camila"),
  valor = c(10, NA, 7, NA, 15)
)
str(tib_ejemplo)
```

```
## tibble [5 x 2] (S3: tbl_df/tbl/data.frame)
##  $ nombre: chr [1:5] "Claudio" "Javiera" "Elias" NA ...
##  $ valor : num [1:5] 10 NA 7 NA 15
```

Valores no disponibles (NA)

Omisión de NAs

A veces necesitamos eliminar las observaciones `NA` de nuestros elementos. El comando `na.omit()` nos limpia nuestros objetos de valores faltantes, por ejemplo:

```
na.omit(tib_ejemplo)
```

```
## # A tibble: 3 x 2
##   nombre  valor
##   <chr>   <dbl>
## 1 Claudio    10
## 2 Elias       7
## 3 Camila     15
```

Nota: Antes de eliminar los NA se recomienda realizar un análisis previo, ya que podrían existir problemas de sesgos.

Valores no disponibles NAs

Omisión de NAs

Algunas funciones contienen argumentos que permiten omitir las observaciones NA de los elementos para realizar sus calculos, por ejemplo en la función `sum()` existe el argumento `na.rm` que elimina los NA del cálculo de la suma:

```
sum(vec_ejemplo)
```

```
## [1] NA
```

```
sum(vec_ejemplo, na.rm=TRUE)
```

```
## [1] 9
```

Importación de datos

Importación de datos

La importación de una base de datos en R dependerá del formato del archivo a importar. Para ello, existen diferentes funciones que permiten llevar a cabo tal procedimiento:

Importar archivos `txt`

```
read.table("<Ruta del archivo>", header = TRUE, ... )
readr::read_csv("<Ruta del archivo>", col_names = TRUE,...)
```

Importar archivos `csv` Dependiendo de cómo esté codificado el archivo csv, hay distintas funciones para importar:

- **`read_csv`** : Si los decimales están con puntos y las variables se separan por comas.
- **`read_csv2`**: Si los decimales están con comas y las variables se separan por punto y coma (;).

```
readr::read_csv("<Ruta del archivo>", col_names = TRUE, ...)
readr::read_csv2("<Ruta del archivo>", col_names = TRUE, ...)
```

Importación de datos

Importar archivos `excel`

La librería `readxl` tiene múltiples funciones para cargar archivos en formato excel. Un argumento importante de estas funciones es **`sheet`**, en donde se puede indicar cuál hoja se importará. Sus principales funciones son:

```
readxl::read_excel("<Ruta del archivo>", col_names = TRUE, ...)
readxl::read_xls()
readxl::read_xlsx()
```

Importar archivos de otros formatos

El paquete `haven` contiene multiples funciones para importar archivos de SPSS, STATA y SAS con funciones, tales como: `read_sas()`, `read_por()`, `read_sav()` y `read_dta()`

El paquete `rio` y su función `import` permiten importar numerosos tipos de archivo de formato, incluyendo Excel, SAS, SPSS, STATA, Minitab, Matlab, JSON, etc. Es recomendable su uso si la base a cargar es limpia y no tiene problemas de importación.

```
rio::import("<Ruta del archivo>")
```

Taller Práctico 1

Taller Práctico 1

El conjunto de datos `viviendasRM.xlsx` contiene avisos de viviendas usadas que se venden en la Región Metropolitana de Chile recolectados de Chile Propiedades en mayo 2020. La base de datos contiene 1139 observaciones de 13 variables (la descripción de las variables se encuentran en la siguiente slide).

1. Importe la base de datos y corrobore su clase (¿es tibble o data frame?).
2. Si su base fue importada como Data Frame, convierta a Tibble.
3. Realice un pequeño análisis exploratorio de la base.
4. Rescate la información contenida en el cuarto registro.
5. Compare los resultados obtenidos al ejecutar las funciones **glimpse** y **str**

Fuente: Kaggle.

Diccionario de datos:

Variable	Descripción
Comuna	Comuna de la región en la cual se encuentra la casa.
Link	Enlace al aviso de la vivienda.
Tipo_Vivienda	Tipo de vivienda.
N_Habitaciones	Número de habitaciones.
N_Banos	Número de baños.
N_Estacionamientos	Número de estacionamientos.
Total_Superficie_M2	Total de superficie en mt2.
Superficie_Construida_M2	Superficie construida en mt2: metros de la construcción habitable.
Valor_UF	Valor en Unidades de fomentos de la propiedad declarado en el portal.
Valor_CLP	Valor en pesos chilenos.
Direccion	Dirección de la casa del aviso.
Quien_Vende	Nombre de la persona que vende la vivienda.
Corredor	Indica nombre de la empresa que vende.

Problemas en la importación de datos

Problemas en la importación de datos

Delimitadores

Algunas veces las observaciones vienen separadas por distintos delimitadores. Esto puede configurarse para realizar correctamente la lectura de los datos en las distintas funciones de importación:

```
read.table(..., sep = "<delimitador>")
readr::read_csv(..., sep = "<delimitador>")
```

Saltos de fila

El argumento `skip` presente en la mayoría de las funciones para cargar datos, permite saltarse un número de filas de observaciones para realizar la lectura de datos. Esto sirve para cargar archivos en donde la base de datos no empieza desde la primera fila.

```
readr::read_csv(...,skip = n)
readxl::read_excel(...,skip = n)
```

Problemas en la importación de datos

Codificación de datos faltantes (NA)

Es común que en las bases de datos, exista una codificación de valores faltantes distinto a una celda vacía (por ejemplo, *, 88 o 99). Dependiendo de cómo se importen los datos, estos puede recodificarse de distintas formas.

Directamente al importar

Algunas funciones de importación tienen argumentos que permiten codificar observaciones como NA dado un vector de referencia:

```
readr::read_csv("archivo", na = vector, ... )
readxl::read_excel("archivo", na = vector, ... )
```

Problemas en la importación de datos

Codificación de datos faltantes (NA)

Después de importar

Hay distintas herramientas para recodificar elementos como NA. A continuación se presentan dos formas equivalentes, usando R base y `dplyr` respectivamente.

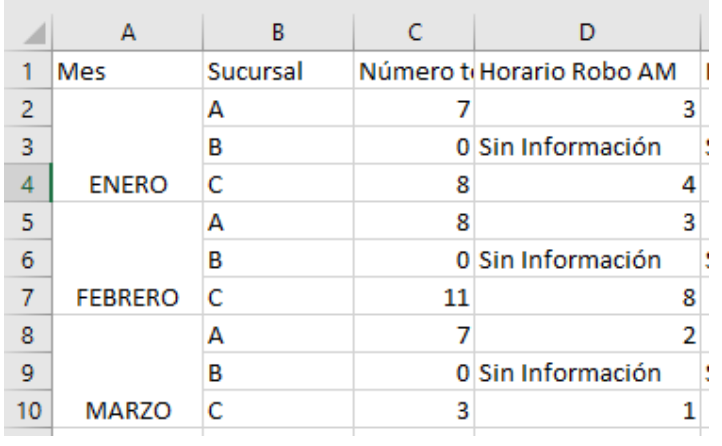
```
nombre_base[ nombre_base %in% vector ] <- NA
nombre_base %>% dplyr::na_if(vector)
```

Problemas en la importación de datos

Observaciones agrupadas

Algunas bases de datos contienen filas agrupadas. Esto es problemático dado que solo se cargará información a la primera celda contenida en esta agrupación. Es decir:

Mes	Sucursal	...
ENERO	A	...
NA	B	...
NA	C	...



	A	B	C	D
1	Mes	Sucursal	Número t	Horario Robo AM
2		A	7	3
3		B	0	Sin Información
4	ENERO	C	8	4
5		A	8	3
6		B	0	Sin Información
7	FEBRERO	C	11	8
8		A	7	2
9		B	0	Sin Información
10	MARZO	C	3	1

Problemas en la importación de datos

Observaciones agrupadas

Si el documento es un Excel, esto se puede solucionar usando el argumento `fillMergedCells = TRUE` al cargar el archivo usando el comando `openxlsx::read.xlsx()`.

Si el archivo ya está cargado, el comando `tidyr::fill` permite llenar bases de datos con variables incompletas de muchas formas, gracias a su argumento `direction`.

Taller Práctico 2

Taller Práctico 2

Cargue los datos de migración disponible en el archivo *migracion.xlsx* que contiene datos de la población nacida fuera del país, por país o continente de nacimiento, según región de residencia habitual actual y periodo de llegada a Chile.

Utilice funciones de R, para solucionar problemas de carga de valores incompletos.

¡Gracias!

Ana María Alvarado Celis
amalvara@uc.cl

Maite Vergara - Esteban Rucán
maite.vergara@uc.cl - errucan@uc.cl