2. Procesamiento

La idea de este apartado es entender más a fondo cómo trabajar tablas para poder hacer análisis y visualización. Será necesario ver algunas cuestiones de ejecución básica de `R` y algunas funciones típicas de `Tidyverse` para limpiar y transformar nuestra base de datos. Un pequeño punteo:

- tipos de objetos
- operadores
- funciones de lectura y escritura de archivos
- funciones de resumen y exploración de una base de datos
- funciones de transformación de columnas y valores
- formas típicas de ordenar una base de datos (*wide*, *long*)
- unión y consolidación de distintas tablas

Empecemos.

2.1. Pregunta-problema

Se mantendrá la tradición: definiremos una guía analítica para que las operaciones que hagamos tengan un sentido claro y sea más sencillo entender cuál es la información relevante.

En este caso nos vamos a seguir preguntando por el voto en blanco en las elecciones argentinas. ¿Qué particularidades presenta en las distintas provincias?

2.2. Fuentes de datos

Los datos electorales son una fuente compleja de abordar por el volumen de información que conlleva. El dataset completo está disponible en https://www.argentina.gob.ar/dine/resultados-electorales/elecciones-2023 y vamos a estar utilizando el de las elecciones generales de 2023. No olviden descomprimir el archivo luego de descargarlo. Lo interesante de tomar este tipo de datos es el nivel de desagregación: tenemos resultados por mesa. En los comicios locales, cada mesa pertenece a un circuito, cada circuito pertenence a una sección y cada sección pertenece a un distrito. Por poner un ejemplo: tenemos la mesa **12** en el circuito **12A** pertenenciente a la sección **Comuna 12** del distrito **Capital Federal**. Salvo las mesas, todos los niveles de

agregación tienen un código y un nombre. Por ejemplo, la Provincia de Buenos Aires es el distrito n° 2 y Luján es la sección n° 71.

Vamos primero a cargar la librería de `Tidyverse`. En general, la carga de todas las librerías necesarias se realiza al principio del *script*, por una cuestión de estructura general (sirve para saber si tenemos instaladas todas las necesarias, qué tipo de funciones vamos a utilizar, etc.).

```
library(tidyverse)
```

2.3. Objetos y operadores

En la programación hay distintas lógicas de construcción de lenguajes. `R` se inscribe dentro de la programación orientada a objetos, es decir, cada elemento es tratado como un objeto. No vamos a profundizar, pero nos va a servir saber que una de las características esenciales de los objetos es que pertenencen a una **clase**. ¿Qué implica? La clase del objeto va a determinar un conjunto de propiedades y operaciones que podemos aplicar en dicho objeto. Los tipos o clases de objetos que vamos a estar utilizando son:

Simples

- **character**: texto.
- **numeric**: números.
- **logical** o **bool**: puede ser verdadero (TRUE) o falso (FALSE).

Compuestos

- **vector**: lista de elementos *del mismo tipo*; unidimensional.
- **dataframe**: lo que usualmente llamamos tablas. Tienen filas y columnas.

Hay ciertas palabras y caracteres en los lenguajes de programación que están *reservados* para funciones específicas; en general cambian de color en `Rstudio`. Esas palabras reservadas no se pueden usar para nombrar objetos nuevos, por ejemplo.

```
# veamos las diferencias entre escribir la misma palabra de distintas formas
class(FALSE)
```

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

```
#class(False)
class("False")
```

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

Un repaso rápido por los operadores más comunes nos muestra los siguientes.

```
# operadores aritméticos
10 + 10 # suma
```

```
[1] 20
```

```
10 - 10 # resta
```

```
[1] 0
```

```
10 * 2 # multiplicación
```

```
[1] 20
```

```
10 / 2 # división
```

```
[1] 5
```

```
10**2 # elevación
```

```
[1] 100
```

```
# operadores de relación
10 > 2 # mayor
```

```
[1] TRUE
```

```
"karl" == "marx" # es igual
```

```
[1] FALSE
```

```
"karl" != "marx" # es distinto
```

```
[1] TRUE
```

```
"marx" %in% c("marx", "durkheim", "weber") # está presente en
```

```
[1] TRUE
```

```
# operadores lógicos
TRUE & FALSE # operador lógico "Y"
```

```
[1] FALSE
```

```
TRUE | FALSE # operador lógico "O"
```

```
[1] TRUE
```

```
# de asignación
a = 2 #asignamos lo que está a la derecha a lo que está a la izquierda del operador
a
```

```
[1] 2
```

```
b <- 3 # idem
b
```

```
[1] 3
```