# Datos Solidarios con R: Procesando datos

Análisis y visualización de datos para el sector solidario en Colombia

### Abriendo una base datos

En el nivel más simple, se puede pensar en una base de datos como una colección de *data frames*. Al igual que un *data frame*, una base de datos es una colección de columnas con nombre, donde cada valor de la columna es del mismo tipo.

R se puede conectar a muchas bases de datos relacionales, como Oracle, MySQL, SQL Server, etc., y obtiene el resultado como un *data frame*. Una vez que el conjunto de resultados se obtiene en un marco de datos, resulta muy fácil visualizarlos y manipularlos.

**Para este tutorial usaremos un banco de datos simulados, es decir con informaciones ficticias, sobre empresas solidarias en Colombia.** Aunque las informaciones son ficticias, las categorías (variables) son muy semejantes a las que usamos en la entidad.

```
#install.packages("readxl")
library("readxl")

datos <- readxl::read_excel("Datos/Datos_Solidarios_2025.xlsx")
#View(datos)
```

Una vez cargamos el banco de datos, lo primero que debemos hacer es revisarlo y ver las informaciones contenidas allí esten correctas para poder realizar el análisis posterior.

```
datos
```

```
## # A tibble: 4,000 x 16
##    Fecha               cod_Entidad nom_dpto        nom_mpio    latitud longitud
##    <dttm>              <chr>       <chr>           <chr>         <dbl>    <dbl>
##  1 2022-12-31 00:00:00 E534285667  Sucre           Sincelejo      9.30    -75.4
##  2 2022-12-31 00:00:00 E668919915  Meta            Acacías        3.98    -73.7
##  3 2022-12-31 00:00:00 E844595490  Bolívar         Cartagena     10.4     -75.5
##  4 2022-12-31 00:00:00 E858744035  Quindío         Armenia        4.55    -75.7
##  5 2022-12-31 00:00:00 E809816108  Cesar           Valledupar    10.5     -73.3
##  6 2022-12-31 00:00:00 E748870905  Valle del Cauca Palmira        3.54    -76.3
##  7 2022-12-31 00:00:00 E363183577  Risaralda       La Virginia    4.90    -75.9
##  8 2022-12-31 00:00:00 E548469222  Quindío         Armenia        4.54    -75.7
##  9 2022-12-31 00:00:00 E572669408  Sucre           Sampués        9.19    -75.4
## 10 2022-12-31 00:00:00 E971959784  Cesar           Valledupar    10.5     -73.3
## # i 3,990 more rows
## # i 10 more variables: nombreentidad <chr>, sigla <chr>, tipo <chr>,
## #   rural_agro <dbl>, firmante_paz <dbl>, zomac <dbl>, Activos <dbl>,
## #   Ingresos <dbl>, Pasivos <dbl>, Patrimonio <dbl>
```

Para este tutorial de *manipulación*, limpieza y visualización de datos usaremos la librería de tidyverse porque:

(i) es más fácil de aprender

(ii) es más sistemático

(iii) es más transparente de leer y comprender que otros enfoques.

Manipular datos en R es similar a preparar una receta. Debemos seleccionar, cortar, combinar y transformar los ingredientes para obtener un buen resultado. De esta manera, siempre comenzamos con nuestros datos brutos (data.frame) y, en la lógica de tidyverse, aplicamos ('verbos') que transforman nuestra base de datos.

Es importante resaltar que, esta lógica de manipulación de datos basada en 'verbos' es solo un enfoque en el flexible mundo de R. Existen diversas formas de llegar al mismo resultado o a resultados parecidos, **esa es la magia de R**.

```
#library(tidyverse)

datos %>% summary()
```

```
##      Fecha             cod_Entidad          nom_dpto           nom_mpio
##  Min.   :2022-12-31   Length:4000        Length:4000        Length:4000
##  1st Qu.:2022-12-31   Class :character   Class :character   Class :character
##  Median :2022-12-31   Mode  :character   Mode  :character   Mode  :character
##  Mean   :2022-12-31
##  3rd Qu.:2022-12-31
##  Max.   :2022-12-31
##     latitud           longitud      nombreentidad         sigla
##  Min.   : 0.4672   Min.   :-78.79   Length:4000        Length:4000
##  1st Qu.: 4.0863   1st Qu.:-75.81   Class :character   Class :character
##  Median : 5.5488   Median :-75.26   Mode  :character   Mode  :character
##  Mean   : 6.1679   Mean   :-74.83
##  3rd Qu.: 9.1847   3rd Qu.:-73.76
##  Max.   :11.7718   Max.   :-70.70
##      tipo             rural_agro      firmante_paz        zomac
##  Length:4000        Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  Class :character   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000
##  Mode  :character   Median :0.0000   Median :0.0000   Median :0.0000
##                     Mean   :0.3385   Mean   :0.1195   Mean   :0.1898
##                     3rd Qu.:1.0000   3rd Qu.:0.0000   3rd Qu.:0.0000
##                     Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##     Activos             Ingresos           Pasivos            Patrimonio
##  Min.   :   315906   Min.   :   18203   Min.   :    92924   Min.   :   206342
##  1st Qu.:  7342384   1st Qu.: 1096881   1st Qu.:  2739153   1st Qu.:  3529001
##  Median : 15154603   Median : 2415718   Median :  6024097   Median :  7674196
##  Mean   : 24972689   Mean   : 4402616   Mean   : 11094745   Mean   : 13877944
##  3rd Qu.: 30418560   3rd Qu.: 5363669   3rd Qu.: 13008680   3rd Qu.: 16490445
##  Max.   :469906958   Max.   :90823673   Max.   :281279256   Max.   :256265520
```

Si tenemos dudas sobre el tipo de variables del banco que estamos usando, podemos usar el siguiente código:

```
class(datos$zomac)
```

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

### Renombrar variables (*rename*)

Muchas veces vamos a trabajar con datos cuyos nombres de las variables no son intuitivos o no están estandarizados. Por lo cual, una de las transformaciones más simples, pero a la vez importantes, es cambiarle el nombre de las variables.

Pero antes necesitamos saber, **¿qué son variables?**

Las variables son contenedores para almacenar valores de datos, las cuales usualmente se encuentran organizados de forma vertical (*Columnas*). Para los fines de este tutorial, **entenderemos las informaciones**

**agrupadas en las columnas como variables**; mientras que, **los datos que se ubican en las filas (horizontalmente) serán observaciones**.

Una función que podemos usar para saber los nombres de las variables de nuestro banco es: `names()`, como se muestra a continuación:

```
names(datos)
```

```
##  [1] "Fecha"         "cod_Entidad"   "nom_dpto"      "nom_mpio"
##  [5] "latitud"       "longitud"      "nombreentidad" "sigla"
##  [9] "tipo"          "rural_agro"    "firmante_paz"  "zomac"
## [13] "Activos"       "Ingresos"      "Pasivos"       "Patrimonio"
```

Para este ejemplo vamos a renombrar las variables `nom_dpto` y `nom_mpio` y atribuir nombres más intuitivos como departamento y municipio, respectivamente.

La estructura básica para usar las funciones de `tidyverse` es: iniciar con el objecto (en este caso la base de datos) luego conectamos la función a ser usada con el simbolo llamado *pipe* (>%>) y seguimos las orientaciones de cada función. Si no sabemos cómo usar una función, siempre podremos consultar su documentación y aprender a usarla.

En el caso de la función *rename*, dentro de los paréntesis () debemos colocar el nombre nuevo primero y luego el nombre antiguo. Como se muestra en el ejemplo siguiente:

```
datos %>% rename(departamento = nom_dpto)
```

```
## # A tibble: 4,000 x 16
##    Fecha               cod_Entidad departamento    nom_mpio    latitud longitud
##    <dttm>              <chr>       <chr>          <chr>         <dbl>    <dbl>
##  1 2022-12-31 00:00:00 E534285667  Sucre          Sincelejo      9.30    -75.4
##  2 2022-12-31 00:00:00 E668919915  Meta           Acacías        3.98    -73.7
##  3 2022-12-31 00:00:00 E844595490  Bolívar        Cartagena     10.4     -75.5
##  4 2022-12-31 00:00:00 E858744035  Quindío        Armenia        4.55    -75.7
##  5 2022-12-31 00:00:00 E809816108  Cesar          Valledupar    10.5     -73.3
##  6 2022-12-31 00:00:00 E748870905  Valle del Cauca Palmira       3.54    -76.3
##  7 2022-12-31 00:00:00 E363183577  Risaralda      La Virginia    4.90    -75.9
##  8 2022-12-31 00:00:00 E548469222  Quindío        Armenia        4.54    -75.7
##  9 2022-12-31 00:00:00 E572669408  Sucre          Sampués        9.19    -75.4
## 10 2022-12-31 00:00:00 E971959784  Cesar          Valledupar    10.5     -73.3
## # i 3,990 more rows
## # i 10 more variables: nombreentidad <chr>, sigla <chr>, tipo <chr>,
## #   rural_agro <dbl>, firmante_paz <dbl>, zomac <dbl>, Activos <dbl>,
## #   Ingresos <dbl>, Pasivos <dbl>, Patrimonio <dbl>
```

### Seleccionar/ Cortar Observaciones (*slice*)

La primera manipulación la realizamos a una variable (columna). No obstante, también podemos manipular observaciones (lineas). Por ejemplo, podemos limitar nuestro análisis a las primeras 100 observaciones (líneas), usando el verbo slice().

```
datos %>% slice(1:100)
```

```
## # A tibble: 100 x 16
##    Fecha               cod_Entidad nom_dpto       nom_mpio    latitud longitud
##    <dttm>              <chr>       <chr>          <chr>         <dbl>    <dbl>
##  1 2022-12-31 00:00:00 E534285667  Sucre          Sincelejo      9.30    -75.4
##  2 2022-12-31 00:00:00 E668919915  Meta           Acacías        3.98    -73.7
##  3 2022-12-31 00:00:00 E844595490  Bolívar        Cartagena     10.4     -75.5
##  4 2022-12-31 00:00:00 E858744035  Quindío        Armenia        4.55    -75.7
```

```
##  5 2022-12-31 00:00:00 E809816108  Cesar           Valledupar    10.5     -73.3
##  6 2022-12-31 00:00:00 E748870905  Valle del Cauca Palmira        3.54    -76.3
##  7 2022-12-31 00:00:00 E363183577  Risaralda       La Virginia    4.90    -75.9
##  8 2022-12-31 00:00:00 E548469222  Quindío         Armenia        4.54    -75.7
##  9 2022-12-31 00:00:00 E572669408  Sucre           Sampués        9.19    -75.4
## 10 2022-12-31 00:00:00 E971959784  Cesar           Valledupar    10.5     -73.3
## # i 90 more rows
## # i 10 more variables: nombreentidad <chr>, sigla <chr>, tipo <chr>,
## #   rural_agro <dbl>, firmante_paz <dbl>, zomac <dbl>, Activos <dbl>,
## #   Ingresos <dbl>, Pasivos <dbl>, Patrimonio <dbl>
```

Sim embargo, los códigos anteriores no modifican la base de datos a menos que así lo establezcamos. Para eso debemos rescribir el objecto que contiene la base de datos, como se muestra a continuación:

```
datos <- datos %>% slice(1:100)
```

### Seleccionar variables (*select*)

Ya aprendimos la estructura básica para usar funciones, y todo lo que aprenderemos a partir de ahora sigue el mismo patrón. Cuando tenemos demasiadas columnas en nuestra base de datos, siempre es útil centrarse en las variables más relevantes, excluyendo las que no se utilizarán. Para seleccionar un subconjunto de las variables (columnas) para mantener en el tibble, usamos el verbo select(). De nuevo, comenzamos con el dataframe, el *pipe* y, finalmente, el verbo select.

En el siguiente ejemplo, vamos a seleccionar 5 de las 15 variables que tenemos en el banco original a seguir: el código de la entidad, nombre de la entidad, Activos, Ingresos y Pasivos.

```
datos %>% select(cod_Entidad, nombreentidad, Activos, Ingresos, Pasivos)
```

```
## # A tibble: 100 x 5
##    cod_Entidad nombreentidad                              Activos Ingresos Pasivos
##    <chr>       <chr>                                        <dbl>    <dbl>   <dbl>
##  1 E534285667  Asociación Mutual Sincelejo Central       8.11e6  2162890  1.14e6
##  2 E668919915  Asociación Mutual Acacías Del Caribe      6.69e6  1367940  1.09e6
##  3 E844595490  Fondo Cartagena Andina                    2.87e7  7423741  1.51e7
##  4 E858744035  Fondo Armenia Andina                      6.05e7  9008196  2.51e7
##  5 E809816108  Coop. Valledupar Solidaria                1.94e7  2402008  5.68e6
##  6 E748870905  Fondo de Empleados Palmira Solidaria      5.95e6  1675551  3.12e6
##  7 E363183577  Asociación Mutual La Virginia Del Caribe  3.63e6   872568  2.18e6
##  8 E548469222  Fondo Armenia Futuro                      5.02e6   425921  2.95e6
##  9 E572669408  Coop. Multiactiva Sampués Solidaria       6.93e7  8976459  5.04e7
## 10 E971959784  F.E. Valledupar Esperanza                 1.86e7  3347344  6.08e6
## # i 90 more rows
```

### Agrupar observaciones (group_by)

Usualmente nuestros datos están organizados en categorías o subcategorías. Por ejemplo, por año, por departamentos, o tipo de entidad. Además, muy a menudo necesitamos generar nuevas informaciones, por medio de tablas o gráficos, basadas en estas categorías.

De este modo, podemos usar la función `group_by ()` para crear múltiples niveles/tipos de agrupaciones. Esta es principalmente una función complementar, que optimiza la eficacia de otras funciones para realizar cálculos como: `summarize()` o `tally ()`.

En el siguiente ejemplo se usa `group_by ()` acompañado de la función `tally ()`, la cual es útil para contar el número de observaciones por grupo.

```
datos %>% group_by(tipo) %>%
  tally()
```

```
## # A tibble: 3 x 2
##   tipo                  n
##   <chr>             <int>
## 1 Asociación Mutual    14
## 2 Cooperativa          59
## 3 Fondo de Empleados   27
```

### Estadísticas Resumidas (summarize)

La función `summarize()` se usa para sintetizar informaciones provenientes de cálculos estadísticos, a partir de su combinación con funciones estadísticas básicas de R como las que se muestran a continuación:

| Estadística | Función en R |
|---|---|
| **Media** | *mean(variable)* |
| **Mediana** | *median(variable)* |
| **Desviación Estándar** | *sd(variable)* |
| **Cuantil (10%)** | *quantile(variable, probs=0.1)* |
| **Máximo** | *max(variable)* |
| **Mínimo** | *min(variable)* |

Así, esta función genera un nuevo tibble pequeño (table/data.frame) para contener las estadísticas de resumen, abandonando nuestro tibble original. La función requiere tres elementos:

(i) el nombre de la nueva variable en el nuevo tibble;

(ii) la función que agregará/resumirá la variable; y

(iii) la variable que se resumirá.

En el ejemplo siguiente, usaremos las funciones `group_by ()` y `summarize()` para calcular la media de Activos de los tres tipos de empresas solidarias contenidas en la base de datos de este tutorial.

```
datos %>% group_by(tipo) %>%
  summarize(media_activos = mean(Activos))
```

```
## # A tibble: 3 x 2
##   tipo               media_activos
##   <chr>                      <dbl>
## 1 Asociación Mutual      11802502.
## 2 Cooperativa            25365734.
## 3 Fondo de Empleados     29847458.
```

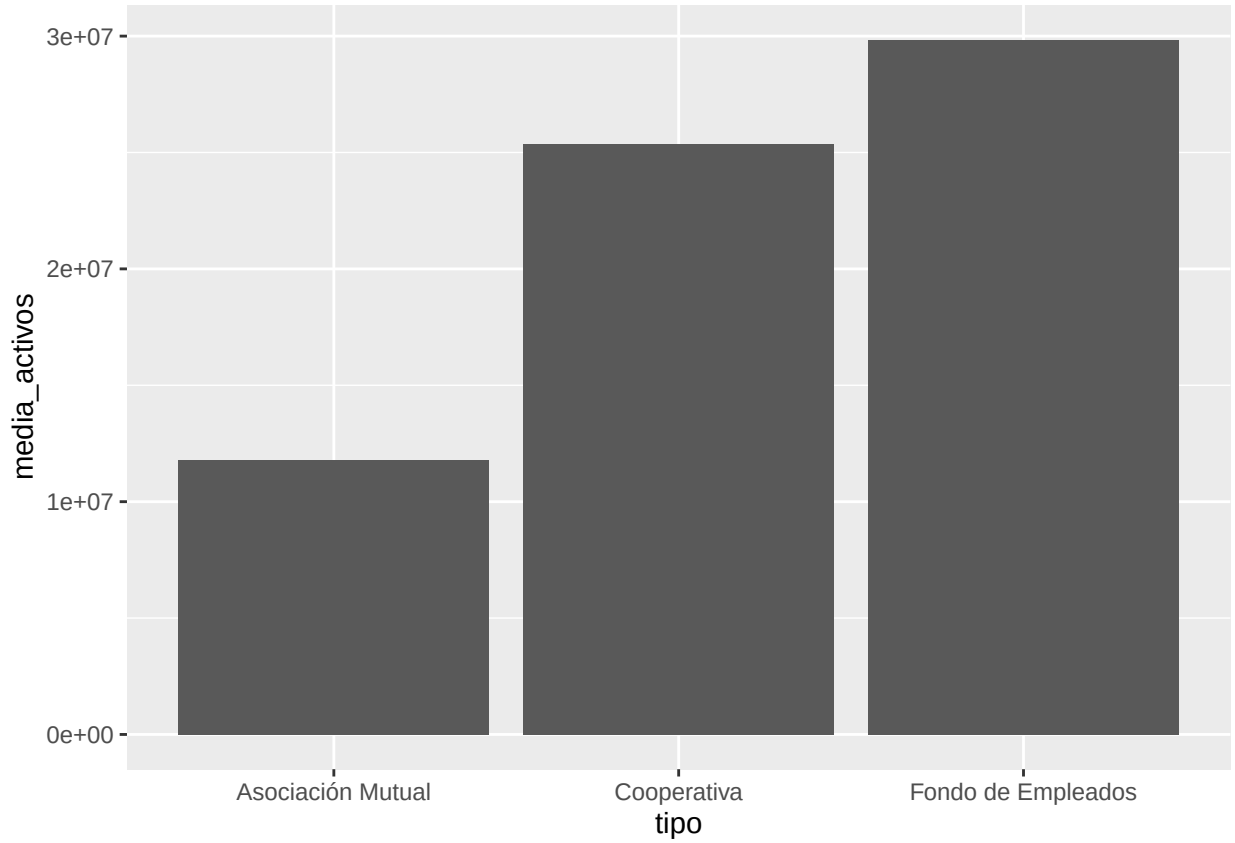
### Gráficos

Terminaremos este tutorial con un pequeño *abre bocas* de cómo podemos generar gráficos en R usando uno de los paquetes de `tidyverse` llamado `ggplot2`.

La estructura (sintaxis) de `ggplot`es nueva y diferente, pero se integra a la perfección en nuestro flujo de trabajo, siguiendo un proceso de preparación de datos tras otra tubería (*pipe* %>%). La conexión con el elemento gráfico se realiza con el signo más (+).

A continuación, realizaremos el *plot* de la tabla generada en la sección anterior transformándola en un gráfico de barras con el siguiente código:

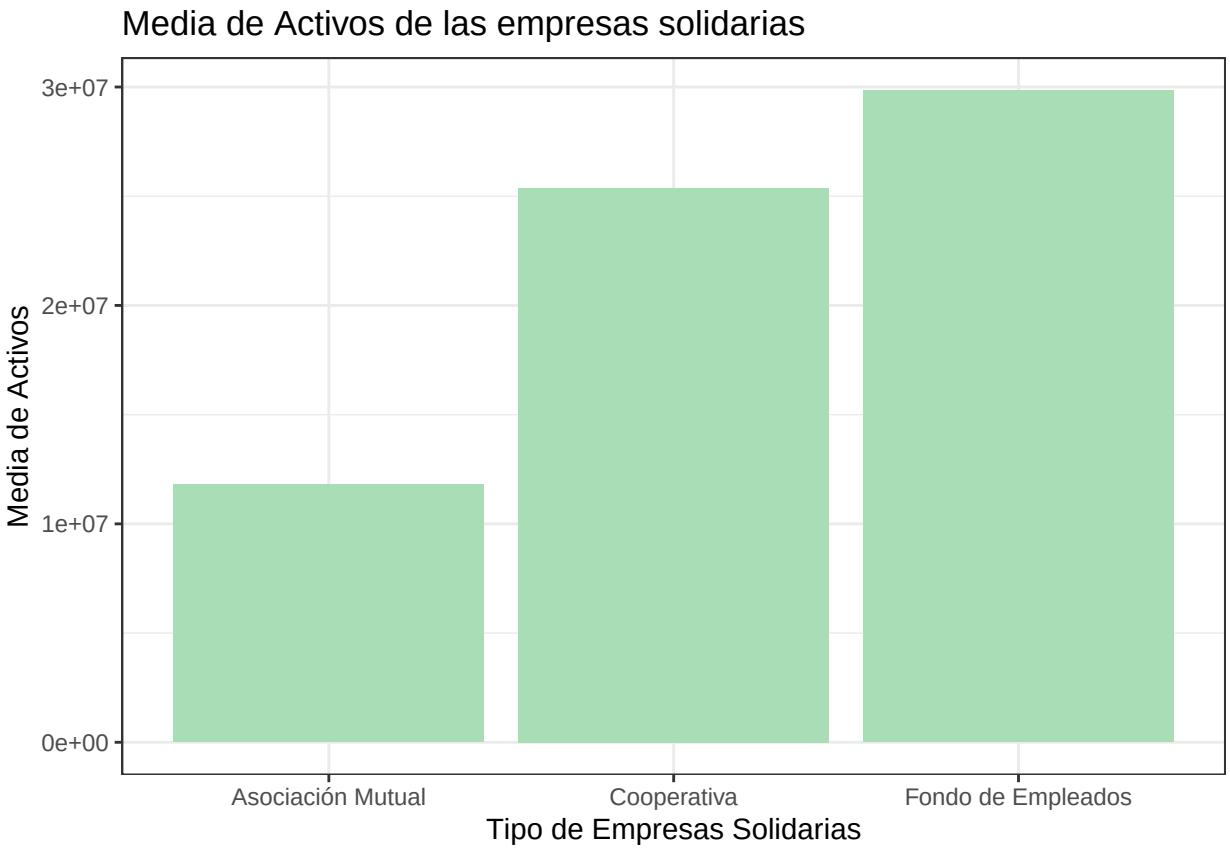
```r
datos %>% group_by(tipo) %>%
  summarize(media_activos = mean(Activos)) %>%
  ggplot() +
  geom_col(aes(x=tipo, y = media_activos))
```




La primera parte del código es exactamente igual a la que habíamos utilizado en la sección anterior, el cambio viene a partir del uso de la función `ggplot2` donde debemos agregar la geometría (*geom*) y la estética (*aes*) **adecuada** y **deseada**.

`ggplot2` nos permite personalizar los gráficos de distintas maneras para obtener resultados más llamativos y cautivantes dependiendo de nuestras necesidades.

```r
datos %>% group_by(tipo) %>%
  summarize(media_activos = mean(Activos)) %>%
  ggplot() +
  geom_col(aes(x=tipo, y = media_activos), fill= "#a8ddb5") +
  ggtitle("Media de Activos de las empresas solidarias") +
  xlab("Tipo de Empresas Solidarias") +
  ylab("Media de Activos") +
  theme_bw()
```




En el código anterior agregamos otras camadas de informaciones a saber:

- **fill** se usa dentro de *aes()* para agregar un color fijo a los gráficos
- **ggtitle** sirve para adicionar títulos a los gráficos
- **xlab** sirve para modificar el nombre de la variable del eje X en el gráfico
- **ylab** sirve para modificar el nombre de la variable del eje Y en el gráfico
- **theme_** se usa para aplicar un tema (layout) al gráfico, en este caso estamos usando un tema llamado "bw"

**Próximos pasos**

Para los participantes que estén interesados en continuar con el aprendizaje de este lenguaje, se recomienda las siguientes opciones:

- Consultar manuales de programación en R disponibles en la web como, por ejemplo: Introducción a R para Ciencias
- Explorar material disponible en páginas como Youtube
- Estar atentos a las redes de Rladies en Colombia, pues se publican cursos online o webinars gratuitos.
- Realizar cursos de análisis de datos en R

---