

# Tipos de datos

## Sección 2: Tipos de datos

En esta unidad veremos los diferentes tipos de datos de R y como trabajar con ellos.

# Tipos de variables

- Las variables en R pueden ser números, textos, tablas, lista de números
- La manera mas común de almacenar datos en R es con Data Frames.
- Para saber el tipo de una variable podemos utilizar la función `class()`

```
> a<-2  
> class(a)  
[1] "numeric"  
> class(ls)  
[1] "function"
```

# Data Frames

- Conceptualmente los Data Frames son tablas
- Las filas representan observaciones y las diferentes variables son representadas en columnas.
- La función `str()` nos muestra la estructura de un objeto

```
> library(dslabs)
> data(murders)
> str(murders)
'data.frame':  51 obs. of  5 variables:
 $ state      : chr  "Alabama" "Alaska" "Arizona" "Arkansas" ...
 $ abb       : chr  "AL" "AK" "AZ" "AR" ...
 $ region     : Factor w/ 4 levels "Northeast","South",...: 2 4 4 2 4
 4 1 2 2 2 ...
 $ population: num  4779736 710231 6392017 2915918 37253956 ...
 $ total      : num   135 19 232 93 1257 ...
> class(murders)
[1] "data.frame"
> head(murders)
```

	state	abb	region	population	total
1	Alabama	AL	South	4779736	135
2	Alaska	AK	west	710231	19

Diagram illustrating the structure of the `murders` data frame:

- The columns (state, abb, region, population, total) are labeled as **VARIABLES**.
- The rows (1, 2) are labeled as **OBSERVACIONES**.

# Acceso a las variables del DataTable

- Para acceder a los valores de las variables utilizamos el símbolo \$ como \$nombreVariable:

```
> names(murders)
[1] "state"      "abb"        "region"     "population"
[5] "total"
> murders$state
[1] "Alabama"
[3] "Arizona"
[5] "California"
[7] "Connecticut"
[9] "District of Columbia"
[11] "Georgia"
[13] "Hawaii"
[15] "Idaho"
[17] "Illinois"
[19] "Indiana"
[21] "Iowa"
[23] "Kansas"
[25] "Kentucky"
[27] "Louisiana"
[29] "Maine"
[31] "Maryland"
[33] "Massachusetts"
[35] "Michigan"
[37] "Minnesota"
[39] "Mississippi"
[41] "Missouri"
[43] "Montana"
[45] "Nebraska"
[47] "Nevada"
[49] "New Hampshire"
[51] "New Jersey"
[53] "New Mexico"
[55] "New York"
[57] "North Carolina"
[59] "North Dakota"
[61] "Ohio"
[63] "Oklahoma"
[65] "Oregon"
[67] "Pennsylvania"
[69] "Rhode Island"
[71] "South Carolina"
[73] "South Dakota"
[75] "Tennessee"
[77] "Texas"
[79] "Utah"
[81] "Vermont"
[83] "Virginia"
[85] "Washington"
[87] "West Virginia"
[89] "Wisconsin"
[91] "Wyoming"
```

- El orden de las entradas en nuestra variable es el mismo que el orden en el datatable original.

# Vectores numéricos

- La variable `murders$population` no es solo un valor, son 51 valores.
- A este tipo de objetos se le llama vectores.

```
> length(murders$population)
[1] 51
> str(murders$population)
num [1:51] 4779736 710231 6392017 2915918 37253956 ...
```

- Los vectores numéricos son listas de uno o más números.

# Vectores de caracteres

- Podemos almacenar caracteres en R

<pre>&gt; a&lt;-1</pre>	}	Variable numérica a con valor 1
<pre>&gt; a</pre>		
<pre>[1] 1</pre>	}	Caracter de texto a
<pre>&gt; "a"</pre>		
<pre>[1] "a"</pre>		


- Todos los elementos almacenados en un objeto character deben de ser caracteres.

```
> class(murders$state)
[1] "character"
```

# Vectores lógicos

- Son vectores de datos que pueden ser VERDADEROS o FALSOS

Operador relacional



```
> Z <- 3 == 2  
> Z  
[1] FALSE  
> class(Z)  
[1] "logical"
```



# Otro tipo de datos: Factores

- Factores son utilizados para almacenar datos categorizados.

```
> class(murders$region)
[1] "factor"
> levels(murders$region)
[1] "Northeast"      "South"
[3] "North Central"  "West"
```

- Almacenar datos en categorías (niveles) maneja la memoria más eficientemente.
- En R los datos se almacenan en enteros, cada nivel se traduce de fondo en un entero que lo representa.

# Versatilidad de R

- Múltiples maneras de acceder a las variables
- Podemos acceder de la manera

```
murders$population
```

- O mediante paréntesis:

```
murders[["population"]]
```

- Si en cambio queremos crear un nuevo data frame con la variable:

```
murders["population"]
```