

**¡Hola 🙌! Espera mientras comienza la sesión.**

**Antes que todo, ¿cómo están?**

# Visualización de Información

**IIC2026 2020-2**

# Selecciones y *join* de datos en D3.js

Visualización de Información

IIC2026 2020-2

**Repaso**

# Repaso

1. **Selecciones en D3.js I**
2. **Selecciones en D3.js II**
3. ***Join* de datos en D3.js I**
4. ***Join* de datos en D3.js II**

# Selecciones

Objeto de D3.js que se comporta como una colección de elementos HTML.

```
d3.select()
```

```
d3.selectAll()
```

```
seleccion.select()
```

```
seleccion.selectAll()
```

# Selecciones

```
d3.selectAll("rect")  
  .attr("y", 50)  
  .style("fill", "red")  
  .attr("x", (d, i, all) => 100 * i);
```

```
1 <svg>  
2   <rect y="50" style="fill: red" x="0"></rect>  
3   <rect y="50" style="fill: red" x="100"></rect>  
4   <rect y="50" style="fill: red" x="200"></rect>  
5 </svg>
```



Antes:

```
1 <body>
2   <ul></ul>
3   <ul></ul>
4   <ul></ul>
5 </body>
```

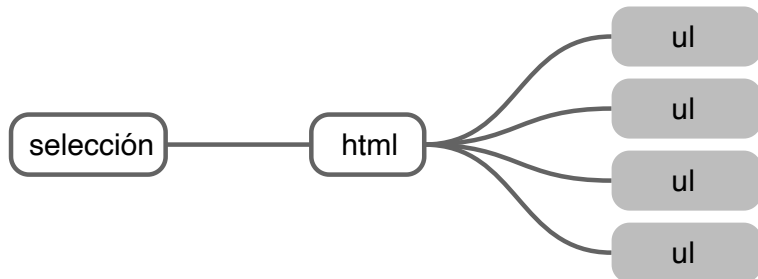
```
d3.selectAll("ul")
  .append("li");
```

Después:

```
1 <body>
2   <ul>
3     <li></li>
4   </ul>
5   <ul>
6     <li></li>
7   </ul>
8   <ul>
9     <li></li>
10  </ul>
11 </body>
```

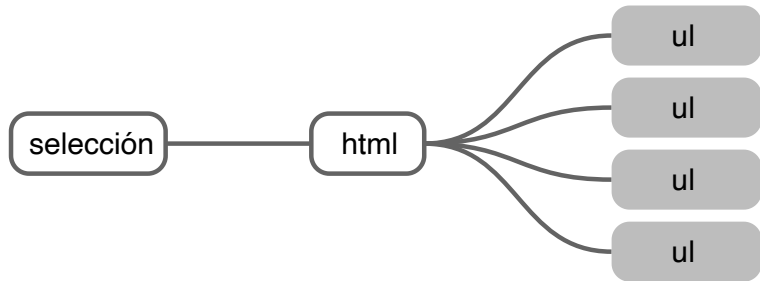
# Múltiples grupos

```
d3.selectAll("ul");
```



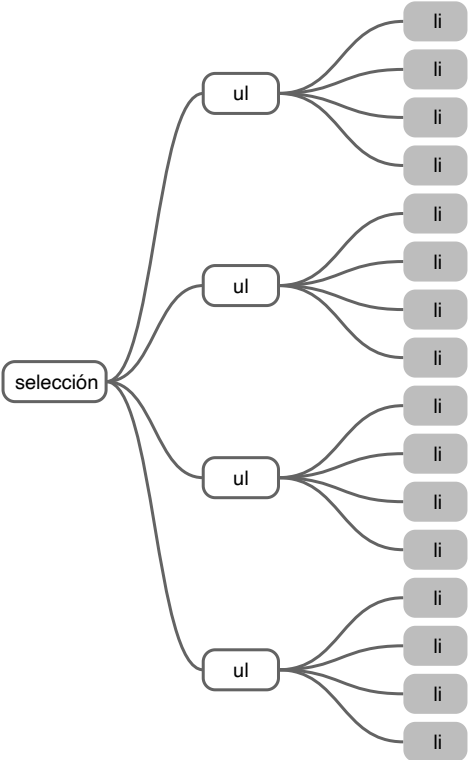
# Múltiples grupos

```
d3.selectAll("ul")  
  .selectAll("li");
```



# Múltiples grupos

```
d3.selectAll("ul")  
  .selectAll("li");
```

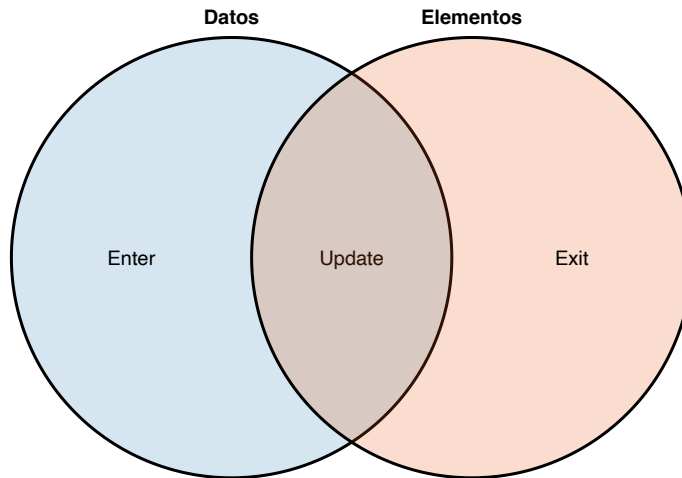


## Duda publicada

- Si yo hago un `selectAll( "li" )` por ejemplo en verdad no va a seleccionar todos todos todos, sino que seleccionará todos los que se encuentren en el primer grupo? Por lo que debo de cierta forma ir avanzando de a poco en los grupos para llegar a un elemento que puede estar muy al fondo?

# seleccion.data

- Hay datos que no se le asocian elementos ➡ *enter*
- Hay elementos y datos que se asocian entre ellos ➡ *update*
- Hay elementos que no se le asocian datos ➡ *exit*



## Update

```
1 const datos = [23, 45, 120, 64];  
2  
3 d3.select("#svg")  
4   .selectAll("rect")  
5   .data(datos);
```

```
<svg id="svg" width="400" height="250">  
  <rect></rect> <!-- 23 -->  
  <rect></rect> <!-- 45 -->  
  <rect></rect> <!-- 120 -->  
  <rect></rect> <!-- 64 -->  
</svg>
```

## Update

```
1 const datos = [23, 45, 120, 64];  
2  
3 const update = d3.select("#svg")  
4   .selectAll("rect")  
5   .data(datos);
```

```
<svg id="svg" width="400" height="250">  
  <rect></rect> <!-- 23 -->  
  <rect></rect> <!-- 45 -->  
  <rect></rect> <!-- 120 -->  
  <rect></rect> <!-- 64 -->  
</svg>
```



## Update

```
1 const datos = [23, 45, 120, 64];
2
3 const update = d3.select("#svg")
4   .selectAll("rect")
5   .data(datos);
6
7 update.attr("width", 50)
8   .attr("y", 0)
9   .attr("x", (d, i, all) => i * 100);
```

```
<svg id="svg" width="400" height="250">
  <rect></rect> <!-- 23 -->
  <rect></rect> <!-- 45 -->
  <rect></rect> <!-- 120 -->
  <rect></rect> <!-- 64 -->
</svg>
```

## Update

```
1 const datos = [23, 45, 120, 64];
2
3 const update = d3.select("#svg")
4   .selectAll("rect")
5   .data(datos);
6
7 update.attr("width", 50)
8   .attr("y", 0)
9   .attr("x", (d, i, all) => i * 100);
```

```
<svg id="svg" width="400" height="250">
  <rect width="50" y="0" x="0"></rect> <!-- 23 -->
  <rect width="50" y="0" x="100"></rect> <!-- 45 -->
  <rect width="50" y="0" x="200"></rect> <!-- 120 -->
  <rect width="50" y="0" x="300"></rect> <!-- 64 -->
</svg>
```

## Update

```
1  const datos = [23, 45, 120, 64];
2
3  const update = d3.select("#svg")
4    .selectAll("rect")
5    .data(datos);
6
7  update.attr("width", 50)
8    .attr("y", 0)
9    .attr("x", (d, i, all) => i * 100)
10   .attr("height", (d, i, all) => 2 * d);
```

```
<svg id="svg" width="400" height="250">
  <rect width="50" y="0" x="0"></rect> <!-- 23 -->
  <rect width="50" y="0" x="100"></rect> <!-- 45 -->
  <rect width="50" y="0" x="200"></rect> <!-- 120 -->
  <rect width="50" y="0" x="300"></rect> <!-- 64 -->
</svg>
```

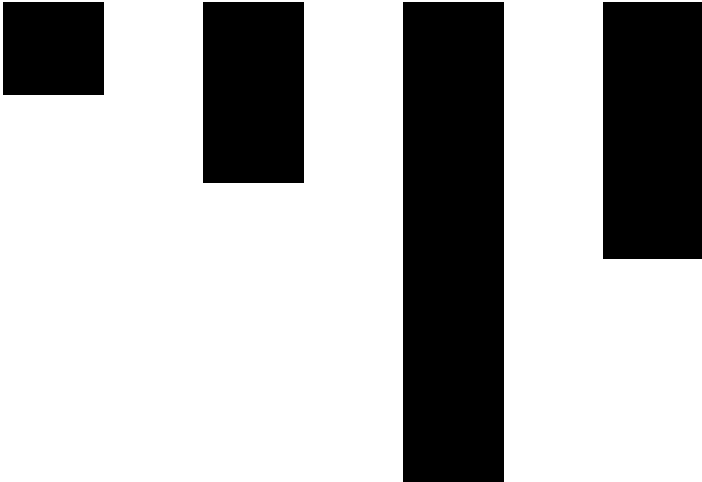
## Update

```
1 const datos = [23, 45, 120, 64];
2
3 const update = d3.select("#svg")
4   .selectAll("rect")
5   .data(datos);
6
7 update.attr("width", 50)
8   .attr("y", 0)
9   .attr("x", (d, i, all) => i * 100)
10  .attr("height", (d, i, all) => 2 * d);
```

```
<svg id="svg" width="400" height="250">
  <rect width="50" y="0" x="0" height="46"></rect> <!-- 23 -->
  <rect width="50" y="0" x="100" height="90"></rect> <!-- 45 -->
  <rect width="50" y="0" x="200" height="240"></rect> <!-- 120 -->
  <rect width="50" y="0" x="300" height="128"></rect> <!-- 64 -->
</svg>
```

# Update

```
1 const datos = [23, 45, 120, 64];  
2  
3 const update = d3.select("#svg")  
4   .selectAll("rect")  
5   .data(datos);  
6  
7 update.attr("width", 50)  
8   .attr("y", 0)  
9   .attr("x", (d, i, all) => i * 100)  
10  .attr("height", (d, i, all) => 2 * d);
```



## Duda publicada

- Respecto al atributo "d" en (d, i, all); el parametro lo reconoce como dato por ser "d" o por la posicion dentro de la funcion?

## Exit

```
<svg id="svg" width="400" height="250">  
  <rect></rect>  
  <rect></rect>  
  <rect></rect>  
  <rect></rect>  
</svg>
```

```
1 const datos = [23, 45];  
2  
3 d3.select("#svg")  
4   .selectAll("rect")  
5   .data(datos);
```

```
<svg id="svg" width="400" height="250">  
  <rect></rect> <!-- 23 -->  
  <rect></rect> <!-- 45 -->  
  <rect></rect> <!-- ? -->  
  <rect></rect> <!-- ? -->  
</svg>
```

## Exit

```
<svg id="svg" width="400" height="250">  
  <rect></rect>  
  <rect></rect>  
  <rect></rect>  
  <rect></rect>  
</svg>
```

```
1 const datos = [23, 45];  
2  
3 const update = d3.select("#svg")  
4   .selectAll("rect")  
5   .data(datos);  
6  
7 update.exit().remove();
```

```
<svg id="svg" width="400" height="250">  
  <rect></rect> <!-- 23 -->  
  <rect></rect> <!-- 45 -->  
  <rect></rect> <!-- ? -->  
  <rect></rect> <!-- ? -->  
</svg>
```



## Exit

```
<svg id="svg" width="400" height="250">
  <rect></rect>
  <rect></rect>
  <rect></rect>
  <rect></rect>
</svg>
```

```
1 const datos = [23, 45];
2
3 const update = d3.select("#svg")
4   .selectAll("rect")
5   .data(datos);
6
7 update.exit().remove();
```

```
<svg id="svg" width="400" height="250">
  <rect></rect> <!-- 23 -->
  <rect></rect> <!-- 45 -->
</svg>
```

*Enter*

```
<svg id="svg" width="400" height="250">  
</svg>
```

```
1 const datos = [23, 45, 120, 64];  
2  
3 d3.select("#svg")  
4   .selectAll("rect")  
5   .data(datos);
```

```
<svg id="svg" width="400" height="250">  
  <!-- ? -->  
</svg>
```

*Enter*

```
<svg id="svg" width="400" height="250">  
</svg>
```

```
1 const datos = [23, 45, 120, 64];  
2  
3 d3.select("#svg")  
4   .selectAll("rect")  
5   .data(datos)  
6   .enter();
```

```
<svg id="svg" width="400" height="250">  
  <!-- ? -->  
</svg>
```

## Enter

```
<svg id="svg" width="400" height="250">  
</svg>
```

```
1 const datos = [23, 45, 120, 64];  
2  
3 d3.select("#svg")  
4   .selectAll("rect")  
5   .data(datos)  
6   .enter()  
7   .append("rect");
```

```
<svg id="svg" width="400" height="250">  
  <!-- ? -->  
</svg>
```

## Enter

```
<svg id="svg" width="400" height="250">  
</svg>
```

```
1 const datos = [23, 45, 120, 64];  
2  
3 d3.select("#svg")  
4   .selectAll("rect")  
5   .data(datos)  
6   .enter()  
7   .append("rect");
```

```
<svg id="svg" width="400" height="250">  
  <rect></rect> <!-- 23 -->  
  <rect></rect> <!-- 45 -->  
  <rect></rect> <!-- 120 -->  
  <rect></rect> <!-- 64 -->  
</svg>
```

## Enter

```
1  const datos = [23, 45, 120, 64];
2
3  d3.select("#svg")
4    .selectAll("rect")
5    .data(datos)
6    .enter()
7    .append("rect")
8    .attr("width", 50)
9    .attr("y", 0)
10   .attr("x", (d, i, all) => i * 100)
11   .attr("height", (d, i, all) => 2 * d);
```

```
<svg id="svg" width="400" height="250">
  <rect></rect> <!-- 23 -->
  <rect></rect> <!-- 45 -->
  <rect></rect> <!-- 120 -->
  <rect></rect> <!-- 64 -->
</svg>
```

## Enter

```
1  const datos = [23, 45, 120, 64];
2
3  d3.select("#svg")
4    .selectAll("rect")
5    .data(datos)
6    .enter()
7    .append("rect")
8    .attr("width", 50)
9    .attr("y", 0)
10   .attr("x", (d, i, all) => i * 100)
11   .attr("height", (d, i, all) => 2 * d);
```

```
<svg id="svg" width="400" height="250">
  <rect width="50" y="0" x="0" height="46"></rect> <!-- 23 -->
  <rect width="50" y="0" x="100" height="90"></rect> <!-- 45 -->
  <rect width="50" y="0" x="200" height="240"></rect> <!-- 120 -->
  <rect width="50" y="0" x="300" height="128"></rect> <!-- 64 -->
</svg>
```

```
1 const svg = d3.select("body").append("svg");
2
3 const datos = [150, 256, 130, 0, 23, 422, 235];
4
5 svg.attr("width", 50 + datos.length * 100).attr("height", 500);
6
7 svg
8   .selectAll("rect")
9   .data(datos)
10  .enter()
11  .append("rect")
12  .attr("width", 50)
13  .attr("fill", "magenta")
14  .attr("height", (d) => d)
15  .attr("x", (_, i) => 50 + i * 100);
```



```
1 const svg = d3.select("body").append("svg");
2
3 const datos = [150, 256, 130, 0, 23, 422, 235];
4
5 svg.attr("width", 50 + datos.length * 100).attr("height", 500);
6
7 svg
8   .selectAll("rect")
9   .data(datos)
10  .enter()
11  .append("rect")
12  .attr("width", 50)
13  .attr("fill", "magenta")
14  .attr("height", (d) => d)
15  .attr("x", (_, i) => 50 + i * 100);
```

```
1 const svg = d3.select("body").append("svg");
2
3 const datos = [150, 256, 130, 0, 23, 422, 235];
4
5 svg.attr("width", 50 + datos.length * 100).attr("height", 500);
6
7 svg
8   .selectAll("rect")
9   .data(datos)
10  .enter()
11  .append("rect")
12  .attr("width", 50)
13  .attr("fill", "magenta")
14  .attr("height", (d) => d)
15  .attr("x", (_, i) => 50 + i * 100);
```

```
1 const svg = d3.select("body").append("svg");
2
3 const datos = [150, 256, 130, 0, 23, 422, 235];
4
5 svg.attr("width", 50 + datos.length * 100).attr("height", 500);
6
7 svg
8   .selectAll("rect")
9   .data(datos)
10  .enter()
11  .append("rect")
12  .attr("width", 50)
13  .attr("fill", "magenta")
14  .attr("height", (d) => d)
15  .attr("x", (_, i) => 50 + i * 100);
```

```
1 const svg = d3.select("body").append("svg");
2
3 const datos = [150, 256, 130, 0, 23, 422, 235];
4
5 svg.attr("width", 50 + datos.length * 100).attr("height", 500);
6
7 svg
8   .selectAll("rect")
9   .data(datos)
10  .enter()
11  .append("rect")
12  .attr("width", 50)
13  .attr("fill", "magenta")
14  .attr("height", (d) => d)
15  .attr("x", (_, i) => 50 + i * 100);
```

## Método `join` para flujo usual de *data join*

```
1 svg
2   .selectAll("rect")
3   .data(datos)
4   .join("rect");
```

```
1 svg
2   .selectAll("rect")
3   .data(datos)
4   .join(
5     enter => enter.append("rect"),
6     update => update,
7     exit => exit.remove()
8   );
```

## Algunas aclaraciones

## Algunas aclaraciones

- La vinculación entre elementos y datos mediante `data` queda en los elementos, no en la selección misma.

## Algunas aclaraciones

- La vinculación entre elementos y datos mediante `data` queda en los elementos, no en la selección misma.
- La vinculación de un elemento a un dato potencialmente se sobre escribe con más llamadas a `data`.

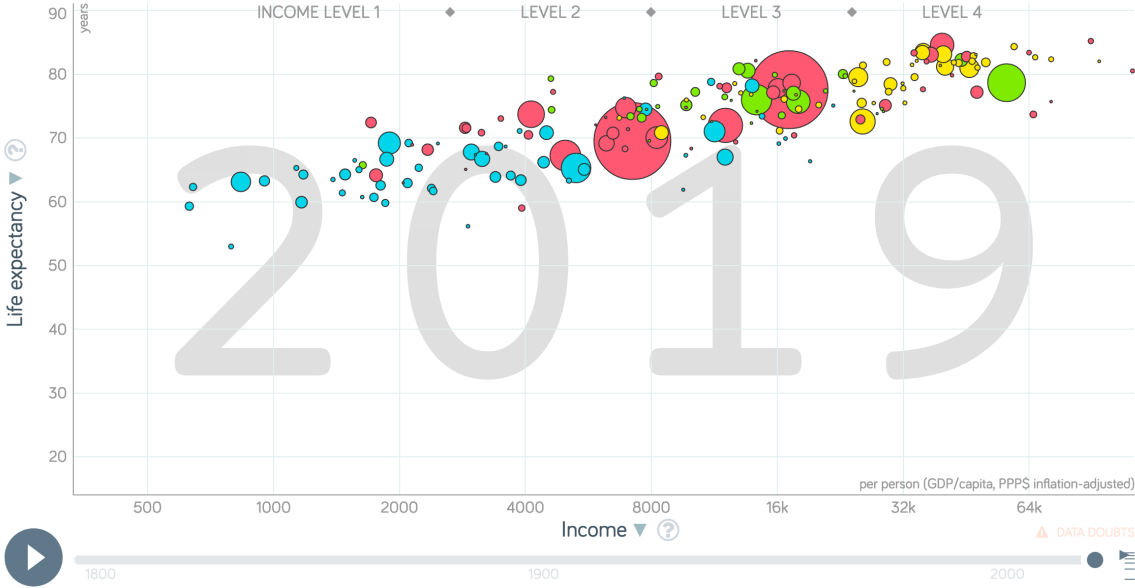


## Algunas aclaraciones

- La vinculación entre elementos y datos mediante `data` queda en los elementos, no en la selección misma.
- La vinculación de un elemento a un dato potencialmente se sobre escribe con más llamadas a `data`.
- La vinculación entre arreglo de datos se hace por grupo, en vez de a nivel de selección.

**¡Visualización del día!**

# ¡Visualización del día!

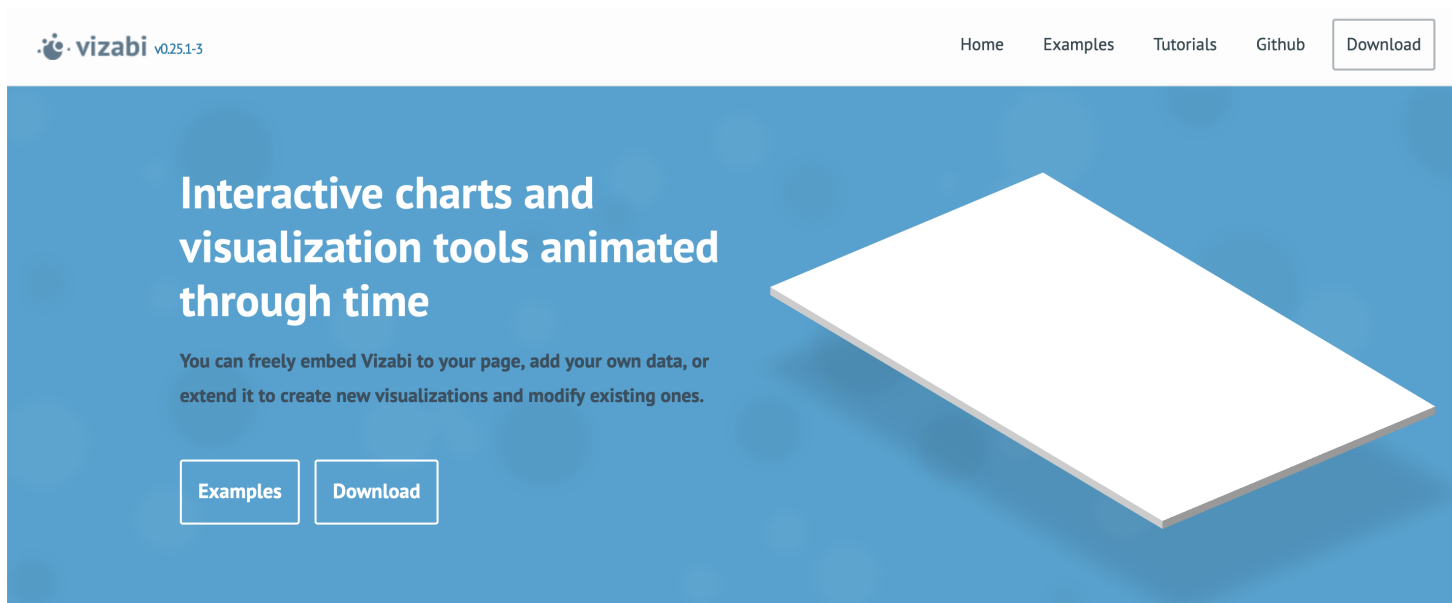


Propuesto por estudiante Fabián Sepúlveda Rivas.

(Fuente: [Gapminder Tools](#))



# ¡Visualización del día!



The screenshot shows the homepage of the Vizabi website. At the top left is the logo 'vizabi v0.25.1-3'. The navigation menu includes 'Home', 'Examples', 'Tutorials', 'Github', and a 'Download' button. The main content area has a blue background with the text 'Interactive charts and visualization tools animated through time'. Below this is a sub-headline: 'You can freely embed Vizabi to your page, add your own data, or extend it to create new visualizations and modify existing ones.' There are two buttons: 'Examples' and 'Download'. On the right side of the main content area is a 3D white rectangular block.

¡Fundación Gapminder desarrolla herramienta de visualización que contruye sobre D3!

(Fuente: [Vizabi](#))

**¿Más dudas?**

## Próximos eventos:

Ayudantía 4 (7 de septiembre) comenzarán jugando con D3.js. ¡Recomendada!

## Próximos eventos:

Ayudantía 4 (7 de septiembre) comenzarán jugando con D3.js. ¡Recomendada!  
Viernes 11 de septiembre (20:00) termina plazo de **Hito 1**.



# Selecciones y *join* de datos en D3.js

Visualización de Información

IIC2026 2020-2

¡Deja tus preguntas en los comentarios!