

# UNIVERSIDAD TECNOLÓGICA DE CHIHUAHUA

## Tecnologías de la información



### 9N Extracción de Conocimiento en Bases de Datos

#### V.1. Reporte de investigación de técnicas de visualización (50%)

**Docente:**

ING. LUIS ENRIQUE MASCOTE CANO

**Alumno:**

Jorge Alfonso Lopez Bustamante

**IDGS 91N**

# Índice

|  |    |
|--|----|
| Introducción .....   | 3  |
| Fundamentos de Visualización de Datos .....  | 3  |
| Conceptos básicos .....  | 3  |
| Tipos de representación gráfica (definición, cuándo usar, ejemplos visuales) ..... | 4  |
| Proceso de Storytelling con Datos .....  | 5  |
| Definición y componentes .....   | 5  |
| Etapas del proceso .....   | 5  |
| Mejores prácticas .....  | 5  |
| Herramientas de Business Intelligence.....   | 6  |
| Tableau .....  | 6  |
| Power BI .....   | 6  |
| QlikView (vs. Qlik Sense) .....  | 6  |
| Google Looker Studio (antes Data Studio) .....                                     | 7  |
| Bibliotecas de Visualización .....   | 7  |
| Python .....   | 7  |
| JavaScript.....  | 12 |
| Conclusión.....  | 17 |
| 8. Referencias bibliográficas .....  | 18 |

## Introducción

La visualización de datos convierte información cruda en representaciones gráficas que revelan patrones, tendencias y relaciones de forma rápida y comprensible. Es esencial tanto para la exploración (detectar anomalías, correlaciones, conglomerados) como para la comunicación (explicar hallazgos y apoyar decisiones). Autores clásicos como Tufte subrayan la *integridad gráfica* y el *data-ink ratio*; Few y Ware enfatizan claridad, reducción de “chartjunk” y principios perceptuales. Estos fundamentos sustentan prácticas modernas en BI y ciencia de datos.

Objetivo del reporte. Investigar y documentar técnicas, herramientas y bibliotecas de visualización, y el proceso de storytelling con datos, con ejemplos reproducibles y recomendaciones prácticas.

Alcance. Se cubren: conceptos básicos (coordenadas, ejes, color, diseño), tipos gráficos por propósito, storytelling, herramientas BI (Tableau, Power BI, QlikView/Looker Studio), bibliotecas en Python, JavaScript y R, conclusiones y referencias.

## Fundamentos de Visualización de Datos

### Conceptos básicos

#### Sistemas de coordenadas.

- Cartesiano: ejes X–Y (y Z en 3D) para relaciones numéricas y temporales.
- Polar: radio–ángulo para periodicidad (gráficas radiales).
- Geoespacial: proyecciones y escalas geográficas (choropleth, mapas). En D3 existen múltiples proyecciones y geocodificación.

#### Tipos de ejes.

- Lineal, logarítmico, tiempo, categórico; selección según la naturaleza de la variable. Chart.js y Highcharts soportan múltiples escalas y ejes temporales.

#### Esquemas de colores y su importancia psicológica.

- Secuenciales (magnitud), divergentes (desviaciones respecto a un punto crítico), cualitativos (categorías). Los palettes ColorBrewer son perceptualmente equilibrados y aptos para color-blind.

#### Principios de diseño visual.

- *Mostrar los datos* (Tufte), maximizar data-ink, minimizar elementos no informativos, mantener integridad gráfica y contexto; guiar la atención con título, anotaciones y color escaso.

## **Tipos de representación gráfica (definición, cuándo usar, ejemplos visuales)**

**Nota:** Inserta las imágenes generadas por IA donde se indica.

### **A) Visualización de cantidad**

- Barras/Columnas: comparar magnitudes entre categorías; usar eje Y con cero.
- Pictogramas: similares a barras pero con íconos (usar con moderación).  
Ejemplo visual:  
[Imagen] *Barras comparando ventas por categoría (IA)*

### **B) Visualización de distribución**

- Histogramas: forma de una variable; elegir binning adecuado.
- Box plots: resumen robusto (mediana, cuartiles, outliers).
- Violín: distribución completa y simetrías/asimetrías.  
Ejemplo visual:  
[Imagen] *Histograma y box-plot de tiempos de entrega (IA)*

### **C) Visualización de proporción**

- Pastel/Donut: pocas categorías ( $\leq 5$ ), evitar 3D.
- Treemaps: proporción jerárquica y ocupación de espacio.  
Ejemplo visual:  
[Imagen] *Donut de participación de mercado y treemap de portafolio (IA)*

### **D) Relación XY**

- Dispersión (scatter): relación entre dos variables; añadir color/tamaño para tercera dimensión.
- Burbujas: tamaño codifica magnitud adicional (cuidar valores negativos).  
Ejemplo visual:  
[Imagen] *Scatter con tendencia y burbujas para margen (IA)*

### **E) Datos geoespaciales**

- Heatmaps geográficos (densidad), coroplethas (valores por región), cartogramas (área proporcional). D3 y Plotly soportan proyecciones y choropleths; Highcharts Maps también.  
Ejemplo visual:  
[Imagen] *Choropleth de ventas por estado y heatmap urbano (IA)*

### **F) Visualización de incertidumbre**

- Error bars, intervalos de confianza/bandas y técnicas más intuitivas (densidades/violines) para comunicar variabilidad y riesgo. Claus O. Wilke

recomienda confianza/bandas; se sugiere *frequency framing* y paletas que “suprimen valores” cuando la incertidumbre es alta.

Ejemplo visual:

[Imagen] *Línea con banda de confianza y barras de error (IA)*

## Proceso de Storytelling con Datos

### Definición y componentes

#### ¿Qué es el Storytelling con datos?

Es presentar información como una **narrativa** clara que conecta con la audiencia y motiva acción, no solo mostrar gráficos complejos. **Knafllic** propone entender contexto, elegir una visual apropiada, eliminar ruido y dirigir la atención.

**Elementos clave: contexto, mensaje, visualización y llamado a la acción**

### Etapas del proceso

1. **Comprensión de la audiencia** (rol, nivel técnico, decisiones).
  2. **Definición del mensaje clave** (una frase accionable).
  3. **Selección de datos relevantes** (filtrar, agregar, validar).
  4. **Diseño de la narrativa visual** (orden lógico, resaltado por color/etiquetas).
  5. **Presentación efectiva** (títulos “takeaway”, ensayos y feedback).
- Todo este flujo está alineado con el marco de *Storytelling with Data*.

### Mejores prácticas

- **Estructura:** contexto → problema → evidencia → recomendación.
  - **Evitar errores:** pasteles con muchas categorías, ejes truncados en barras, exceso de colores, gráficos 3D sin necesidad. (Tufte/Few)
  - **Casos exitosos:** paneles interactivos y narrativas guiadas en Tableau/Power BI que usan títulos accionables y resaltado mínimo de color.
- [Imagen] *Storyboard de una narrativa de KPIs (IA)*

# Herramientas de Business Intelligence

## Tableau

**Descripción general.** Plataforma de análisis visual con enfoque en dashboards interactivos, conexión a múltiples fuentes y capacidades recientes de AI (Tableau Agent/Pulse).

### Características principales.

- *Show Me* mejorado, *VizQL Data Service* (API), *Pulse* móvil, temas personalizados.

**Ventajas.** Alta interactividad, comunidad amplia, diseño visual potente. **Desventajas.** Costo/licenciamiento y curva de aprendizaje para modelado. (Resumen con base en release notes y blogs técnicos)

**Tipos de gráficas soportadas.** Barras, líneas, mapas, treemaps, burbujas, boxplots, histogramas, etc.

**Casos de uso recomendados.** Dashboards ejecutivos y análisis autoservicio con combinación de datos.

## Power BI

**Descripción general.** Servicio de BI de Microsoft (parte de **Microsoft Fabric**) con creación de reportes, **Copilot** para insights y amplio ecosistema.

### Características principales.

- Copilot en reportes/móvil, administración de datasets, seguridad, integración con M365/Teams. **Novedades Nov 2025:** expansión de matrices, card visual GA, actualización de conectores

**Ventajas.** Integración con Microsoft 365, costo competitivo, colaboración. **Desventajas.** Algunas funciones avanzadas requieren Premium/Fabric; complejidad de gobernanza. (Síntesis de documentación oficial/blog)

**Tipos de gráficas soportadas.** Amplia librería nativa y **visuals** de marketplace (decomposition tree, maps, etc.).

**Casos de uso.** Reportes corporativos, distribución a usuarios M365.

[Imagen] *Interfaz conceptual de Power BI con Copilot (IA)*

## QlikView (vs. Qlik Sense)

**Descripción general.** **QlikView** es plataforma de primera generación enfocada en **guided analytics**; **Qlik Sense** es autoservicio moderno con AI y APIs abiertas.

**Características principales.** Motor asociativo, exploración libre; Qlik Sense agrega visual data prep, NLP y mejor experiencia táctil.

**Ventajas/Desventajas.** QlikView: control y personalización, pero aprendizaje y menor autoservicio; Qlik Sense: facilidad y modernidad, pero migración/estrategia. (Fuentes comparativas técnicas)

**Tipos de gráficas.** Barras, líneas, mapas, treemaps, etc., con buenas capacidades interactivas.

**Casos de uso.** Migraciones de QlikView a entornos self-service; análisis guiado en organizaciones con BI centralizado.

## **Google Looker Studio (antes Data Studio)**

**Descripción general.** Herramienta gratuita (Pro tiene capacidades enterprise) para dashboards web con conectores y colaboración.

**Características principales.** Editor drag-and-drop, filtros de fecha, plantillas, más de 1,000 conectores; documentación ahora en Google Cloud.

**Ventajas/Desventajas.** Ventaja: coste cero y facilidad; desventaja: límites de complejidad/escala comparado con plataformas pagas. (Docs oficiales)

**Tipos de gráficas.** Líneas, barras, pasteles, mapas geo, tablas/pivots, etc.

**Casos de uso.** Marketing y reportes compartibles/embebibles.

[Imagen] *Dashboard conceptual marketing en Looker Studio (IA)*

## **Bibliotecas de Visualización**

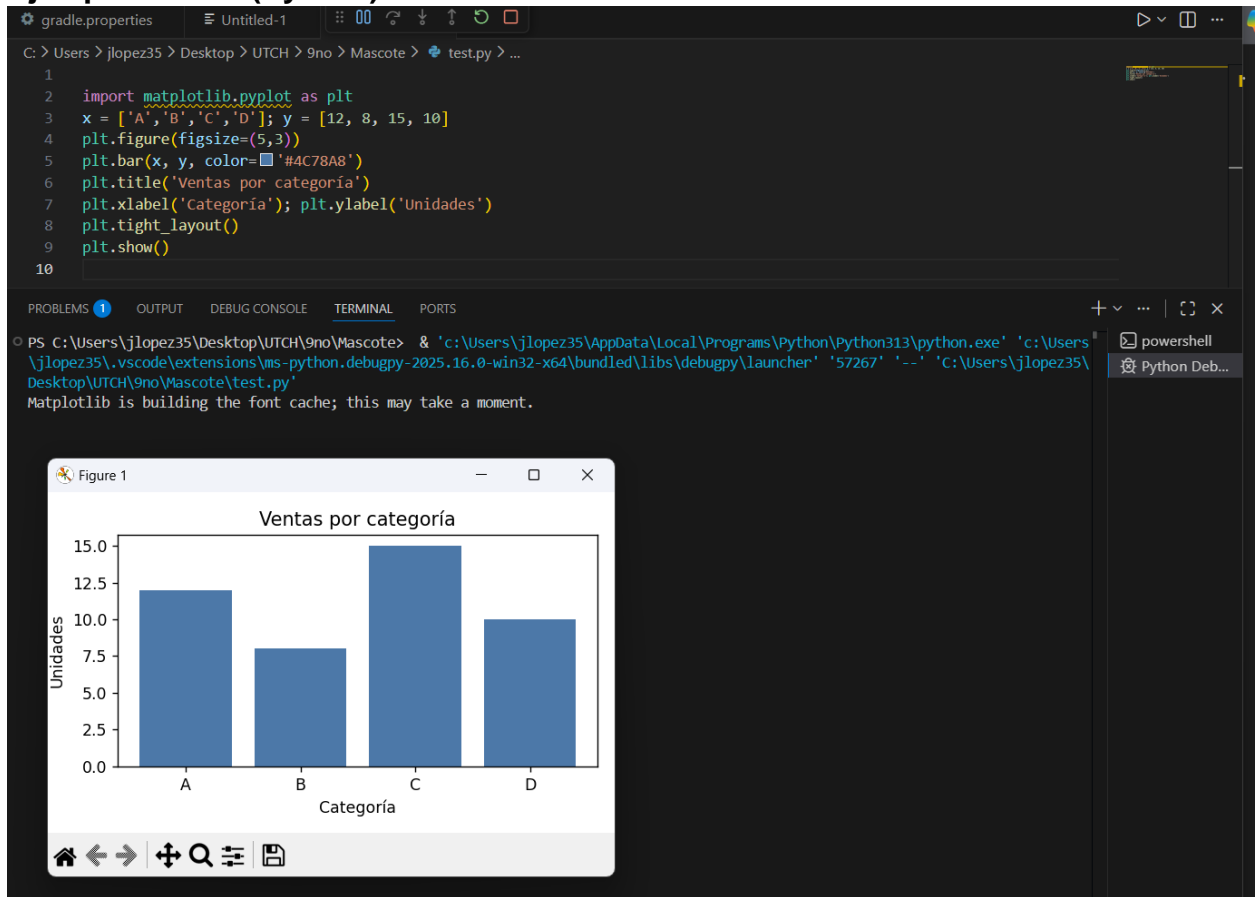
A continuación, al menos cuatro bibliotecas de cada ecosistema.

### **Python**

#### **a) Matplotlib**

- **Lenguaje:** Python.
- **Descripción y filosofía:** Biblioteca base para visualizaciones **estáticas, animadas e interactivas**; “lo difícil es posible”.
- **Características:** Gran personalización, múltiples formatos, OO y pyplot API; extensiones (seaborn, cartopy).
- **Complejidad:** Media–alta (bajo nivel).
- **Gráficas disponibles:** barras, líneas, histogramas, error bars, heatmaps, etc.

- **Ejemplo básico (Python):**



## b) Seaborn

- **Lenguaje:** Python.
- **Descripción:** Capa de alto nivel sobre Matplotlib para **gráficos estadísticos** con estilos y paletas por defecto.
- **Características:** *relplot*, *catplot*, *displot*, *objects API*, faceting.
- **Complejidad:** Baja–media.
- **Gráficas:** distribución, categóricas, regresión, heatmaps.
- **Ejemplo básico (Python):**

```
gradle.properties  Untitled-1  00  ↺  ↻  ↶  ↷  ⏏
C: > Users > jlopez35 > Desktop > UTCH > 9no > Mascote > test.py > ...

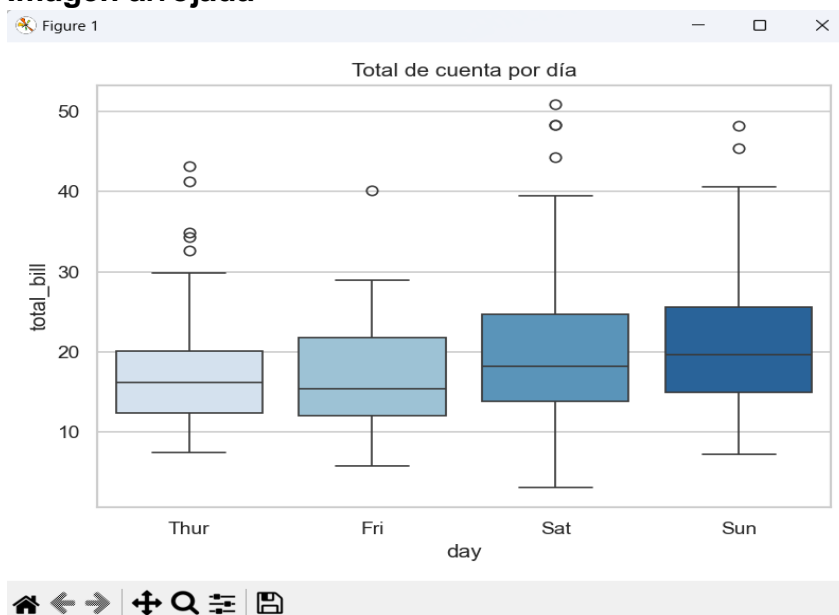
3  import pandas as pd
4  import matplotlib.pyplot as plt # <-- import necesario
5
6  tips = sns.load_dataset("tips")
7  sns.set_theme(style="whitegrid")
8
9  ax = sns.boxplot(data=tips, x="day", y="total_bill", palette="Blues")
10 ax.set_title("Total de cuenta por día")
11
12 plt.tight_layout() # opcional, ajusta márgenes
13 plt.show() # <-- muestra la figura
14

PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\jlopez35\Desktop\UTCH\9no\Mascote> pip install seaborn
PS C:\Users\jlopez35\Desktop\UTCH\9no\Mascote> c:: cd 'c:\Users\jlopez35\Desktop\UTCH\9no\Mascote'; & 'c:\Users\jlopez35\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\jlopez35\.vscode\extensions\ms-python.debugpy-2025.16.0-win32-x64\bundled\libs\debugpy\launcher' '64087' '--' 'C:\Users\jlopez35\Desktop\UTCH\9no\Mascote\test.py'
C:\Users\jlopez35\Desktop\UTCH\9no\Mascote\test.py:5: FutureWarning:
    Passing 'palette' without assigning 'hue' is deprecated and will be removed in v0.14.0. Assign the 'x' variable to 'hue' and set 'legend=False' for the same effect.
    ax = sns.boxplot(data=tips, x="day", y="total_bill", palette="Blues")
PS C:\Users\jlopez35\Desktop\UTCH\9no\Mascote> c:: cd 'c:\Users\jlopez35\Desktop\UTCH\9no\Mascote'; & 'c:\Users\jlopez35\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\jlopez35\.vscode\extensions\ms-python.debugpy-2025.16.0-win32-x64\bundled\libs\debugpy\launcher' '65114' '--' 'C:\Users\jlopez35\Desktop\UTCH\9no\Mascote\test.py'
C:\Users\jlopez35\Desktop\UTCH\9no\Mascote\test.py:9: FutureWarning:
    Passing 'palette' without assigning 'hue' is deprecated and will be removed in v0.14.0. Assign the 'x' variable to 'hue' and set 'legend=False' for the same effect.
    ax = sns.boxplot(data=tips, x="day", y="total_bill", palette="Blues")

```

- **Imagen arrojada**



## c) Plotly (plotly.py)

- **Lenguaje:** Python.

- **Descripción:** Librería **interactiva** basada en plotly.js; más de 30 tipos de gráficos, 3D y mapas. [\[plotly.com\]](https://plotly.com)
- **Características:** Interactividad, exportación (kaleido), integración con Dash. [\[pypi.org\]](https://pypi.org)
- **Complejidad:** Media.
- **Gráficas:** líneas, dispersión, mapas coropleta, 3D, financieros. [\[plotly.com\]](https://plotly.com)
- **Ejemplo básico (Python):**

```

gradle.properties  Untitled-1  test.py  2  X
C: > Users > jlopez35 > Desktop > UTCH > 9no > Mascote > test.py > ...
1
2  import plotly.express as px
3  df = px.data.gapminder().query("year==2007")
4  fig = px.scatter(df, x="gdpPercap", y="lifeExp",
5                  size="pop", color="continent", hover_name="country",
6                  log_x=True, size_max=60, title="Desarrollo vs. Esperanza de vida (2007)")
7  fig.show()
8
PROBLEMS 2  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
PS C:\Users\jlopez35\Desktop\UTCH\9no\Mascote> c:; cd 'c:\Users\jlopez35\Desktop\UTCH\9no\Mascote'; & 'c:\Users\jlopez35\AppData\
Local\Programs\Python\Python313\python.exe' 'c:\Users\jlopez35\.vscode\extensions\ms-python.debugpy-2025.16.0-win32-x64\bundled\li
bs\debugpy\launcher' '65149' '--' 'C:\Users\jlopez35\Desktop\UTCH\9no\Mascote\test.py'
PS C:\Users\jlopez35\Desktop\UTCH\9no\Mascote>

```

## Resultado



## d) Bokeh

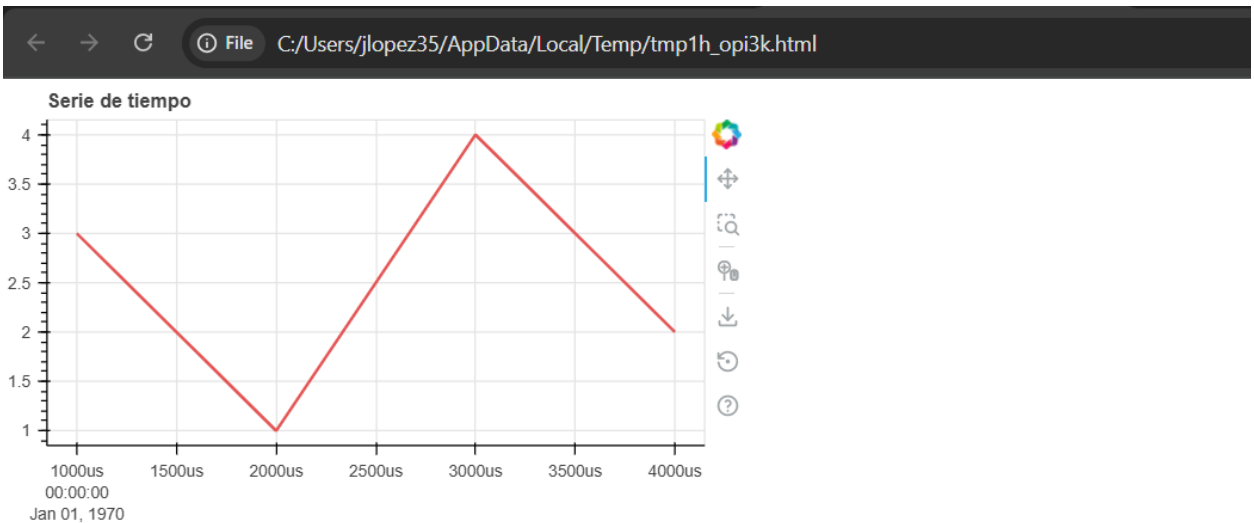
- **Lenguaje:** Python.

- **Descripción:** Visualizaciones interactivas para navegador, dashboards y *streaming*. [\[docs.bokeh.org\]](https://docs.bokeh.org)
- **Características:** Bokeh Server, widgets, callbacks, alta interactividad. [\[docs.bokeh.org\]](https://docs.bokeh.org)
- **Complejidad:** Media.
- **Gráficas:** barras, líneas, mapas, heatmaps, dashboards. [\[docs.bokeh.org\]](https://docs.bokeh.org)
- **Ejemplo básico (Python):**

```
gradle.properties  Untitled-1
```

```
C:\Users\jlopez35 > Desktop > UCH > 9no > Mascote > test.py > ...  
2 from bokeh.plotting import figure, show  
3 p = figure(title="Serie de tiempo", x_axis_type="datetime", width=500, height=300)  
4 p.line([1,2,3,4], [3,1,4,2], color="#E45756", line_width=2)  
5 show(p)  
6
```

## Resultado



### e) Altair (Vega-Altair)

- **Lenguaje:** Python.
- **Descripción:** Librería **declarativa** basada en **Vega-Lite**: se definen mapeos de datos a canales visuales. [\[altair-viz.github.io\]](https://altair-viz.github.io)
- **Características:** Gramática concisa, interactividad, integración con Jupyter. [\[altair-viz.github.io\]](https://altair-viz.github.io)

- **Complejidad:** Baja–media (declarativa).
- **Gráficas:** dispersión, líneas, mapas, facetas, densidades. [\[altair-viz.github.io\]](https://altair-viz.github.io)
- **Ejemplo básico (Python):**

```

1  import altair as alt
2  from vega_datasets import data
3  cars = data.cars()
4  chart = alt.Chart(cars).mark_point().encode(
5      x='Horsepower',
6      y='Miles_per_Gallon',
7      color='Origin'
8  ).interactive()
9  chart
10
11

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```

PS C:\Users\jlopez35\Desktop\UTCH\9no\Mascote> & 'c:\Users\jlopez35\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\jlopez35\.vscode\extensions\ms-python.debugpy-2025.16.0-win32-x64\bundled\libs\debugpy\launcher' '58028' '--' 'C:\Users\jlopez35\Desktop\UTCH\9no\Mascote\test.py'
PS C:\Users\jlopez35\Desktop\UTCH\9no\Mascote>

```

## JavaScript

### a) D3.js

- **Lenguaje:** JavaScript.
- **Descripción:** Biblioteca de bajo nivel para construir **visualizaciones a medida** manipulando DOM/SVG/Canvas; módulos para escalas, ejes, layouts y proyecciones. [\[d3js.org\]](https://d3js.org)
- **Características:** Máxima flexibilidad, layouts (treemap, force, geo), interacciones (zoom/brush). [\[observablehq.com\]](https://observablehq.com)
- **Complejidad:** Alta (arquitectura manual).
- **Tipos:** cualquier gráfico, mapas avanzados (d3-geo). [\[d3js.org\]](https://d3js.org)

- **Ejemplo básico (JS):**

```
C: > Users > jlopez35 > Desktop > UTCH > 9no > Mascote > <> index.html > {} "index.html" > htm
1
2 <!doctype html>
3 <html lang="es">
4 <head>
5   <meta charset="utf-8" />
6   <title>Bar chart con D3</title>
7   <style>
8     body { font-family: system-ui, sans-serif; margin: 24px; }
9     #chart { width: 420px; }
10  </style>
11 </head>
12 <body>
13   <h1>Ejemplo D3 con ES modules</h1>
14   <div id="chart"></div>
15
16   <script type="module">
17     import * as d3 from "https://cdn.jsdelivr.net/npm/d3@7/+e
18
19     const data = [12, 8, 15, 10];
20     const w = 400, h = 240, m = 40;
21
22     const x = d3.scaleBand()
23       .domain(d3.range(data.length))
24       .range([m, w - m])
25       .padding(0.2);
26
27     const y = d3.scaleLinear()
28       .domain([0, d3.max(data)])
29       .range([h - m, m]);
30
31     const svg = d3.select("#chart")
32       .append("svg")
33       .attr("width", w)
34       .attr("height", h);
35
36     svg.append("g")
37       .attr("transform", `translate(0,${h - m})`)
38       .call(d3.axisBottom(x));
39
40     svg.append("g")
41       .attr("transform", `translate(${m},0)`)
42       .call(d3.axisLeft(y));
43
44     svg.selectAll("rect")
45       .data(data)
46       .join("rect")
47       .attr("x", (d, i) => x(i))
48       .attr("y", d => y(d))
49       .attr("width", x.bandwidth())
50       .attr("height", d => y(0) - y(d))
51       .attr("fill", "#4C78A8");
52   </script>
53 </body>
54 </html>
55
```

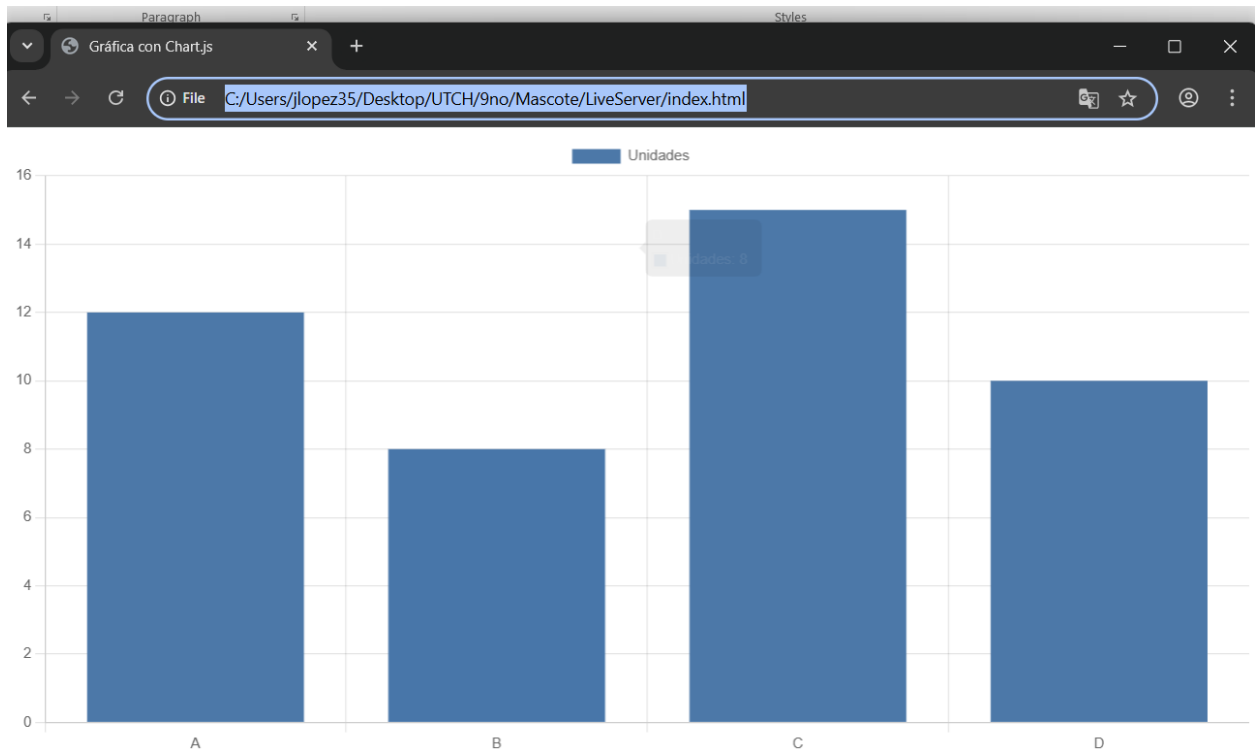
## b) Chart.js

- **Lenguaje:** JavaScript.
- **Descripción:** Librería **simple y flexible** basada en Canvas; gran documentación y comunidad. [\[chartjs.org\]](https://chartjs.org/)
- **Características:** 8+ tipos base, animaciones, plugins (zoom, annotations), tree-shaking. [\[chartjs.org\]](https://chartjs.org/)
- **Complejidad:** Baja.

- **Ejemplo básico (JS):**

```
index.html > html > body > script > scales
1
2 <!DOCTYPE html>
3 <html lang="es">
4 <head>
5   <meta charset="UTF-8">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   <title>Gráfica con Chart.js</title>
8
9   <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
10 </head>
11 <body>
12   <canvas id="myChart"></canvas>
13
14   <script>
15     const ctx = document.getElementById('myChart');
16     new Chart(ctx, {
17       type: 'bar',
18       data: {
19         labels: ['A', 'B', 'C', 'D'],
20         datasets: [{
21           label: 'Unidades',
22           data: [12, 8, 15, 10],
23           backgroundColor: '#4C78A8'
24         }]
25       },
26       scales: {
27         y: { beginAtZero: true }
28       }
29     });
30   </script>
31 </body>
32 </html>
33
34
```

- **Resultado**



### c) Highcharts

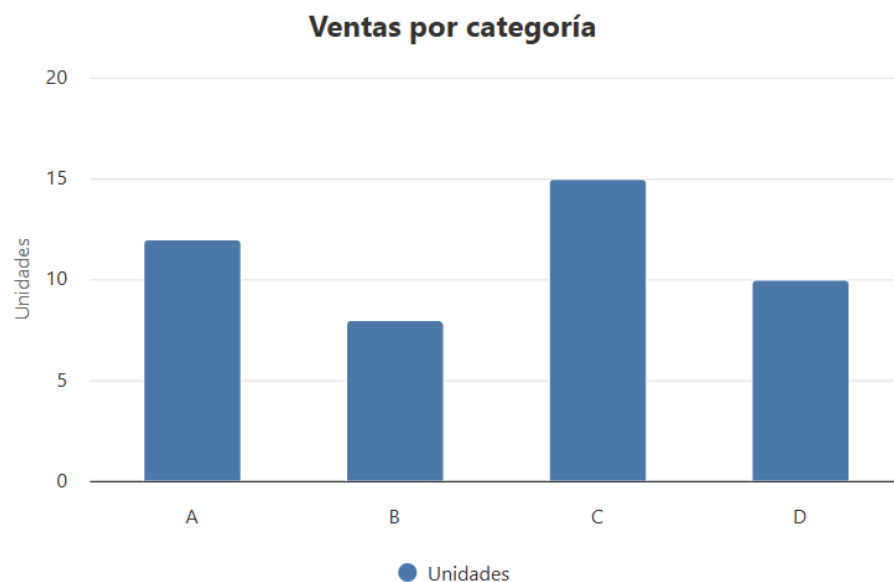
- **Lenguaje:** JavaScript.
- **Descripción:** Librería comercial con gran variedad de tipos (core, maps, stock, gantt) y API extensa.
- **Características:** Accesibilidad, *boost* (WebGL) para grandes volúmenes, exportación, anotaciones.
- **Complejidad:** Media.
- **Ejemplo básico (JS):**

```

index.html ×
<? index.html > <? html > <? body > <? script
1 <!DOCTYPE html>
2 <html lang="es">
3 <head>
4 <meta charset="UTF-8" />
5 <meta name="viewport" content="width=device-width, initial-scale=1.0" />
6 <title>Highcharts demo</title>
7
8 <!-- Carga correcta de Highcharts -->
9 <script src="https://code.highcharts.com/highcharts.js"></script>
10
11 <style>
12 #hc {
13     width: 600px;
14     height: 400px;
15     margin: 24px auto;
16 }
17 </style>
18 </head>
19 <body>
20 <div id="hc"></div>
21
22 <script>
23 Highcharts.chart('hc', {
24     title: { text: 'Ventas por categoria' },
25     xAxis: { categories: ['A', 'B', 'C', 'D'] },
26     yAxis: { title: { text: 'Unidades' } },
27     series: [{
28         type: 'column',
29         name: 'Unidades',
30         data: [12, 8, 15, 10],
31         color: '#AC78AB'
32     }],
33     credits: { enabled: false }
34 });
35 </script>
36 </body>
37 </html>
38

```

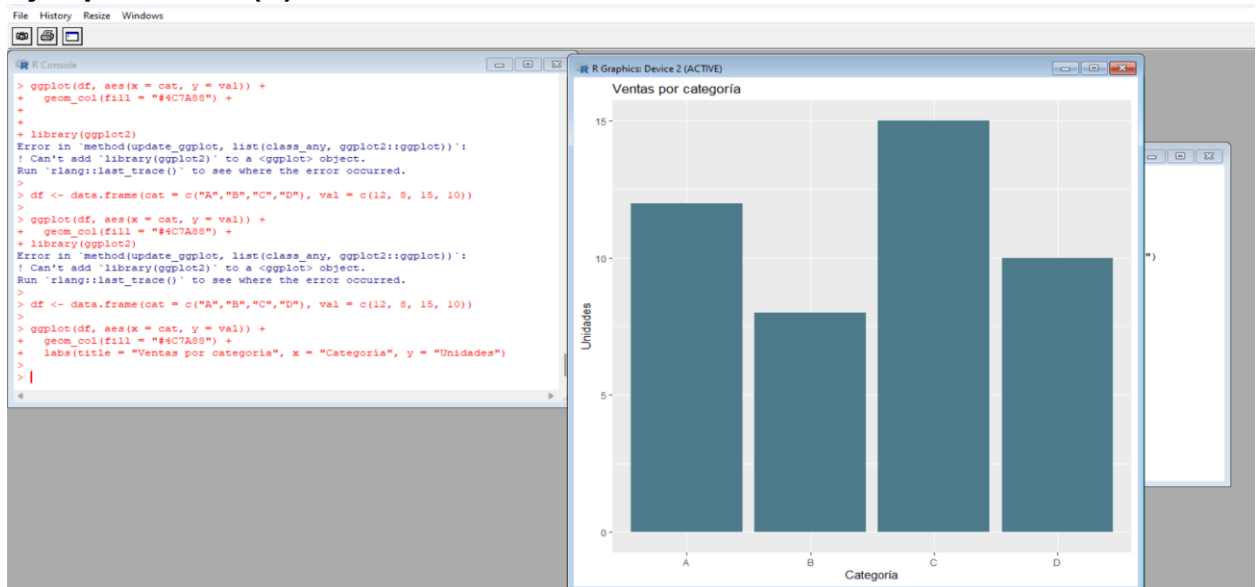
- **Resultado**



## 6.3 R

### a) ggplot2

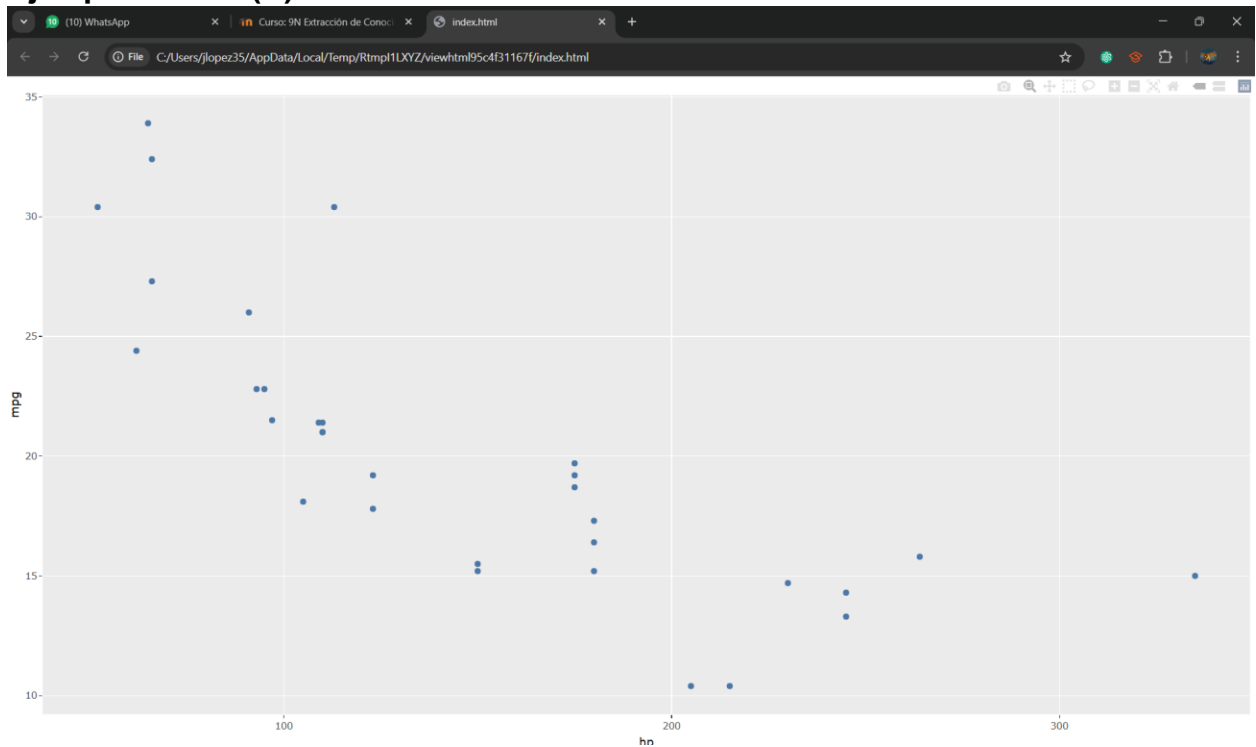
- **Lenguaje:** R.
- **Descripción:** Implementa la **Gramática de Gráficos**; construcción declarativa de capas.
- **Características:** *geoms*, *stats*, *facetas*, escalas (incluye ColorBrewer).
- **Complejidad:** Media.
- **Ejemplo básico (R):**



### b) plotly (R)

- **Lenguaje:** R.
- **Descripción:** Gráficos **interactivos** vía plotly.js; convierte ggplot2 en interactivo (*ggplotly()*).
- **Características:** *subplot*, *layout*, *shiny*; mapas y 3D.
- **Complejidad:** Media.

- **Ejemplo básico (R):**



## Conclusión

- Elegir la visualización adecuada importa. Un gráfico correcto comunica el mensaje clave con claridad y honestidad; uno incorrecto puede inducir a error (ejes truncados, 3D decorativo).
- No es lo mismo utilizar una herramienta que utilizar una biblioteca
  - Herramientas BI (Tableau, Power BI, Qlik, Looker Studio) son ideales para autoservicio, gobernanza y distribución; menor código, mayor productividad.
  - Bibliotecas (Matplotlib/Seaborn/Plotly/Bokeh/D3/ggplot2) brindan control total y reproducibilidad; requieren más conocimiento técnico, pero permiten soluciones a medida.
- Aprendizajes personales. La narrativa efectiva exige: entender la audiencia, definir un takeaway conciso, usar color con intención y representar incertidumbre cuando proceda.

## 8. Referencias bibliográficas

- Knafllic, C. N. (2015/2025). *Storytelling with Data: A Data Visualization Guide for Business Professionals* (10th Anniversary ed.). Wiley.  
[https://books.google.com/books/about/Storytelling\\_with\\_Data.html?id=zcCFEQAAQBAJ](https://books.google.com/books/about/Storytelling_with_Data.html?id=zcCFEQAAQBAJ)  
[\[books.google.com\]](https://books.google.com)
- Tufte, E. R. (2001). *The Visual Display of Quantitative Information* (2nd ed.). Graphics Press. <https://www.edwardtufte.com/book/the-visual-display-of-quantitative-information/>  
[\[edwardtufte.com\]](https://www.edwardtufte.com)
- ColorBrewer. (s.f.). *Color scheme types*.  
[https://colorbrewer2.org/learnmore/schemes\\_full.html](https://colorbrewer2.org/learnmore/schemes_full.html) [\[colorbrewer2.org\]](https://colorbrewer2.org)
- Microsoft Learn. (2025). *Power BI documentation*. <https://learn.microsoft.com/en-us/power-bi/> [\[learn.microsoft.com\]](https://learn.microsoft.com)
- Tableau. (2025). *Tableau 2025.1 New Features*. <https://www.tableau.com/2025-1-features> [\[tableau.com\]](https://www.tableau.com)
- Seaborn Docs. (2024). *seaborn: statistical data visualization*. <https://seaborn.pydata.org/>  
[\[seaborn.pydata.org\]](https://seaborn.pydata.org)
- Plotly. (2025). *Plotly Python Graphing Library*. <https://plotly.com/python/> [\[plotly.com\]](https://plotly.com)
- Bokeh Docs. (2025). *Bokeh documentation*. <https://docs.bokeh.org/en/latest/>  
[\[docs.bokeh.org\]](https://docs.bokeh.org)
- D3 by Observable. (2025). *The JavaScript library for bespoke data visualization*.  
<https://d3js.org/> [\[d3js.org\]](https://d3js.org)
- Wickham, H. (2025). *ggplot2: Create Elegant Data Visualisations Using the Grammar of Graphics*. <https://ggplot2.tidyverse.org/> [\[ggplot2.ti...yverse.org\]](https://ggplot2.tidyverse.org)
- Wilke, C. O. (2019). *Fundamentals of Data Visualization: Visualizing uncertainty*.  
<https://clauswilke.com/dataviz/visualizing-uncertainty.html> [\[clauswilke.com\]](https://clauswilke.com)
- Google Cloud Docs. (2025). *Welcome to Looker Studio*.  
<https://docs.cloud.google.com/looker/docs/studio> [\[docs.cloud...google.com\]](https://docs.cloud.google.com)