

Integrantes del grupo:

Hendrick Rolando Calderón Aguirre – Matrícula: 5185535 – 447942

Pedro David Gabriel Wong – Matrícula: 5160143-423682

Oscar Canseco Salas – Matrícula: 5182949 - 445615

**Maestría en Análisis y Visualización de Datos Masivos**

Análisis grupal tendencias D3

hERRAMIENTAS DE VISUALIZACIÓN

Hendrick Rolando Calderon Aguirre

2021

**Índice general**

[Índice de ilustraciones I](#_Toc91602561)

[Introducción II](#_Toc91602562)

[Objetivos III](#_Toc91602563)

[General III](#_Toc91602564)

[Específicos III](#_Toc91602565)

[Hoja de control IV](#_Toc91602566)

[Contenido 1](#_Toc91602567)

[Estructura general 4](#_Toc91602568)

[Código fuente 5](#_Toc91602569)

[1. limpiezaCodigosPais.ipynb 5](#_Toc91602570)

[2. index.html 12](#_Toc91602571)

[3. main.css 14](#_Toc91602572)

[4. main.js 15](#_Toc91602573)

[Conclusiones 23](#_Toc91602574)

[Recomendaciones 24](#_Toc91602575)

[Bibliografía 25](#_Toc91602576)

# Índice de ilustraciones

**Gráficos**

[Gráfico 1. Mapa coroplético mundial de contagios de COVID-19 2](#_Toc91594130)

[Gráfico 2. Contagios y fallecimientos por COVID-19 3](#_Toc91594131)

**Tablas**

[Tabla 1. Descripción de la estructura general de la actividad 4](#_Toc91595128)

**Imágenes**

[Imagen 1. Estructura general de la actividad 4](#_Toc91595156)

# Introducción

La visualización es un fenómeno humano tratado por estudiosos provenientes de múltiples ámbitos científicos, lo que ha contribuido a la conformación desde distintos enfoques de un marco teórico-práctico amplio. En general, se destacan tres perspectivas de análisis asociadas: cognitiva, tecnológica, y comunicativa.

Por un lado la perspectiva cognitiva crea representaciones visuales para apoyar distintas actividades es un acto que se emprende con frecuencia, con la finalidad de comprender las relaciones que se dan en un contexto particular, y ayuda a la comprensión del mundo externo a partir del pensamiento y el razonamiento.

La perspectiva tecnológica conserva diversos estudios sobre las interacciones que ocurren entre los humanos y las computadoras describen cómo las visualizaciones externas pueden amplificar, reforzar o aumentar la cognición

Finalmente la perspectiva comunicativa considera una tarea del proceso comunicativo, por medio del cual se transforman los datos abstractos y los fenómenos complejos de la realidad en mensajes visibles, y que lleva a un proceso de descubrimiento del conocimiento. (Torres, 2009)

En el presente trabajo, se consultaron los principios de Gestalt para poder brindar una serie de visualizaciones con respecto a una serie de data sets recopilados con respecto a la presente situación ocasionada por el virus SARS-COV-2.

El conjunto de datos utilizado fue recuperado de <https://covid19.who.int/>, y cuentan con registros del periodo comprendido entre el 03/01/2020 al 29/11/2021. Se decidió utilizar este data-set debido a que la actual pandemia, es un fenómeno de alto interés público y que puede brindar más información e interpretación para el usuario que desee consultarlo y pueda informarse acerca de las tendencias actuales que esta ha tenido.

# Objetivos

## G**eneral**

Crear una visualización interactiva en D3.js que permita analizar tendencias presentes a nivel mundial debido a la enfermad COVID-19 tales como contagios y defunciones.

## Específicos

1. Investigar un dataset que permita desarrollar una visualización interactiva con la librería D3 de JavaScript.
2. Desarrollar un script de programación en Python que permita limpiar y agrupar los datos para su correcta interpretación.
3. Programar un mapa coroplético que represente por medio de una escala de color rojo la cantidad de contagios de COVID-19 a nivel mundial.
4. Programar un gráfico de barras interactivo que represente la cantidad de contagios y fallecidos por país a causa del COVID-19.

# Hoja de control

|  |  |  |
| --- | --- | --- |
| Integrante | Actividad desarrollada | Fecha |
| Pedro Gabriel y Oscar Canseco | * Investigación y propuesta del dataset **WHO-COVID-19-global-data.csv. Recuperado de:** <https://covid19.who.int/table> **el 29 de noviembre de 2021** | 29/11/2021 |
| Hendrick Calderón | Generación de la estructura inicial del proyecto:   * resources: main.js, main.css * README.txt * **WHO-COVID-19-global-data-alpha3.csv** * **WHO-COVID-19-global-data.csv** * **WHO-COVID-19-summary.csv** * index.html * limpiezaCodigosPais.ipynb * Análisis grupal tendencias D3.docx   Programación inicial de la visualización:   * Map * BarPlot | 05/12/2021 |
| Pedro Gabriel | * Programación de final del archivo CSS. * Actualización de la visualización. * Agregación de leyenda y escala de colores. | 19/12/2021 |
| Oscar Canseco | * Elaboración escrita del informe y testeo de visualizaciones. * Programación final de la visualización. | 27/12/2021 |
| Hendrick  Calderón | * Elaboración audiovisual y demostración de las visualizaciones. | 27/12/2021 |

Referencia del trabajo colaborativo: <https://github.com/heroca60/tendencias-d3.git>

# Contenido

Se realizó un cambio al data-set original (WHO-COVID-19-global-data.csv), haciendo una copia con configuración Alpha-3 debido a que el data-set original cuenta con configuración Alpha-2. Esto es debido a que los polígonos utilizados para dibujar las visualizaciones en el mapa global, cuentan con un enlace externo a un archivo que presenta los países con los países en Alpha-3. Para ello, se utilizó Python.

En este mismo código, al finalizar la transformación ya mencionada, se hizo una agrupación en cuanto a los nuevos casos y casos de defunción por cada país, generando un nuevo archivo llamado *WHO-COVID-19-summary.csv*.

Se realizó un video donde se hace una demostración y explicación del presente trabajo, dicho video puede ser es ubicado por medio del siguiente enlace:

<https://drive.google.com/file/d/1k5ir4DD71msXLznfq6qTcMaDX71lRK4l/view?usp=sharing>

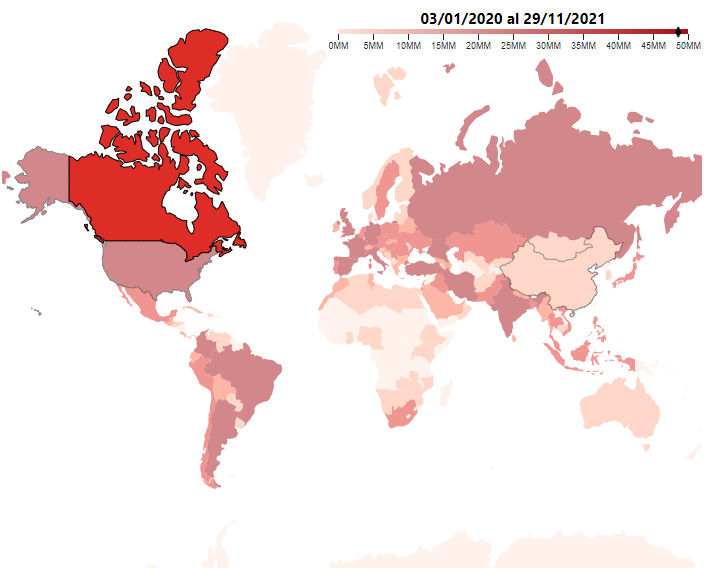
Así mismo, se puede encontrar el repositorio de GitHub con todos los archivos mencionados en el siguiente enlace:

<https://github.com/heroca60/tendencias-d3>

Para el desarrollo del siguiente trabajo se utilizó la versión 6 de D3.

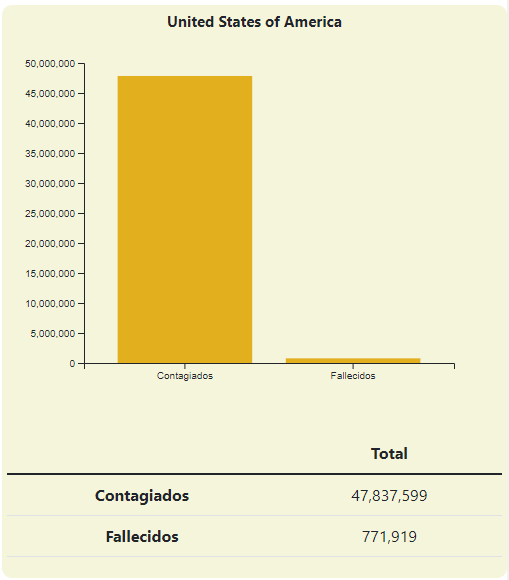
La visualización cuenta con dos gráficos, en primer lugar se puede observar:

Gráfico 1. Mapa coroplético mundial de contagios de COVID-19



En el *Gráfico 1. Mapa coroplético mundial de contagios de COVID-19*, el usuario puede observar a nivel mundial, el comportamiento de contagiados y fallecidos por país. La escala de colores utilizada es directamente proporcional al número de personas contagiadas, es decir, a mayor número de contagios, el color del país será de un tono más obscuro. La visualización cuenta con la característica de que posicionarse sobre un país, se tendrá un enfoque visual sobre la selección y haciendo un desenfoque del resto.

Gráfico 2. Contagios y fallecimientos por COVID-19



Al proporcionar un click sobre un país del gráfico 1, se desplegará del lado derecho, el detalle de dicha selección mostrado en un gráfico de barras el número de contagios y fallecimientos (Gráfica 2). El eje y, número de personas, tiene un rango dinámico que se irá adecuando según las unidades que cada país muestre.

Se utilizó dos gráficos no figurativos (mapa coroplético y gráfico de barras) ya que los gráficos no figurativos tienen la ventaja de presentarse de forma fácil y sencilla de reflejan patrones y tendencias procedentes de un data-set, así como las relaciones que se pueden establecer entre ellos. Es de suma importancia resaltar que el modo de visualizar los datos es abstracto y no guarda algún parecido real con lo que se ha presentado. (UNIR, 2021)

## Estructura general

Imagen 1. Estructura general de la actividad

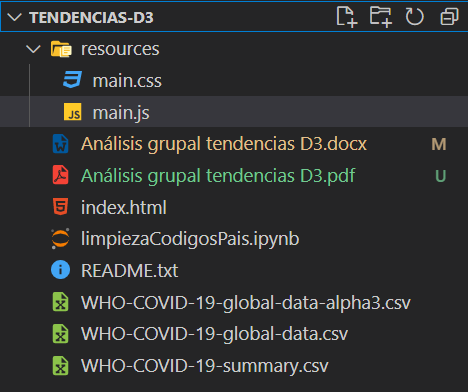


Tabla 1. Descripción de la estructura general de la actividad

|  |  |
| --- | --- |
| **Elemento** | **Descripción** |
| resources | Carpeta que contiene los archivos main.css y main.js |
| main.css | Archivo de hoja de estilos en cascada |
| main.js | Archivo de javascript de contiene la programación utilizando la librería D3.js |
| Análisis grupal tendencias D3.docx y Análisis grupal tendencias D3.pdf | Archivo general de explicación de dicha actividad. |
| index.html | Archivo web que contiene la visualización. |

Continuación Tabla 1.

|  |  |
| --- | --- |
| limpiezaCodigosPais.ipynb | Archivo de Jupyter notebook que contiene programación en Python para la limpieza y agrupación de datos. |
| README.txt | Archivo de explicación básica de la estructura de la actividad. |
| WHO-COVID-19-global-data.csv | Dataset original recuperado de: <https://covid19.who.int/table> **el 29 de noviembre de 2021** |
| WHO-COVID-19-global-data-alpha3.csv | Archivo csv generado con codificación Alpha 3 por medio del script del archivo limpiezaCodigosPais.ipynb |
| WHO-COVID-19-summary.csv | Archivo csv generado con la agrupación de contagios y fallecidos por COVID-19. |

Fuente: elaboración propia

## Código fuente

### limpiezaCodigosPais.ipynb

path = 'WHO-COVID-19-global-data.csv'

import pandas as pd

import numpy as np

# leyendo el archivo csv original

df = pd.read\_csv ('WHO-COVID-19-global-data.csv')

# imprimiendo únicamente los encabezados

df.head()

# Visualizando la composición del dataFrame = 165189 filas y 8 columnas

df.shape

# Sustitutendo de la columna 'Country\_code' los códigos de país alpha-2 por código alpha-3

df['Country\_code'] = df['Country\_code'].replace(['AD'],'AND')

df['Country\_code'] = df['Country\_code'].replace(['AE'],'ARE')

df['Country\_code'] = df['Country\_code'].replace(['AF'],'AFG')

df['Country\_code'] = df['Country\_code'].replace(['AG'],'ATG')

df['Country\_code'] = df['Country\_code'].replace(['AI'],'AIA')

df['Country\_code'] = df['Country\_code'].replace(['AL'],'ALB')

df['Country\_code'] = df['Country\_code'].replace(['AM'],'ARM')

df['Country\_code'] = df['Country\_code'].replace(['AN'],'ANT')

df['Country\_code'] = df['Country\_code'].replace(['AO'],'AGO')

df['Country\_code'] = df['Country\_code'].replace(['AQ'],'ATA')

df['Country\_code'] = df['Country\_code'].replace(['AR'],'ARG')

df['Country\_code'] = df['Country\_code'].replace(['AS'],'ASM')

df['Country\_code'] = df['Country\_code'].replace(['AT'],'AUT')

df['Country\_code'] = df['Country\_code'].replace(['AU'],'AUS')

df['Country\_code'] = df['Country\_code'].replace(['AW'],'ABW')

df['Country\_code'] = df['Country\_code'].replace(['AX'],'ALA')

df['Country\_code'] = df['Country\_code'].replace(['AZ'],'AZE')

# Sustitutendo de la columna 'Country\_code' los códigos de país alpha-2 por código alpha-3

df['Country\_code'] = df['Country\_code'].replace(['BA'],'BIH')

df['Country\_code'] = df['Country\_code'].replace(['BB'],'BRB')

df['Country\_code'] = df['Country\_code'].replace(['BD'],'BGD')

df['Country\_code'] = df['Country\_code'].replace(['BE'],'BEL')

df['Country\_code'] = df['Country\_code'].replace(['BF'],'BFA')

df['Country\_code'] = df['Country\_code'].replace(['BG'],'BGR')

df['Country\_code'] = df['Country\_code'].replace(['BH'],'BHR')

df['Country\_code'] = df['Country\_code'].replace(['BI'],'BDI')

df['Country\_code'] = df['Country\_code'].replace(['BJ'],'BEN')

df['Country\_code'] = df['Country\_code'].replace(['BL'],'BLM')

df['Country\_code'] = df['Country\_code'].replace(['BM'],'BMU')

df['Country\_code'] = df['Country\_code'].replace(['BN'],'BRN')

df['Country\_code'] = df['Country\_code'].replace(['BO'],'BOL')

df['Country\_code'] = df['Country\_code'].replace(['BR'],'BRA')

df['Country\_code'] = df['Country\_code'].replace(['BS'],'BHS')

df['Country\_code'] = df['Country\_code'].replace(['BT'],'BTN')

df['Country\_code'] = df['Country\_code'].replace(['BV'],'BVT')

df['Country\_code'] = df['Country\_code'].replace(['BW'],'BWA')

df['Country\_code'] = df['Country\_code'].replace(['BY'],'BLR')

df['Country\_code'] = df['Country\_code'].replace(['BZ'],'BLZ')

# Sustitutendo de la columna 'Country\_code' los códigos de país alpha-2 por código alpha-3

df['Country\_code'] = df['Country\_code'].replace(['CA'],'CAN')

df['Country\_code'] = df['Country\_code'].replace(['CC'],'CCK')

df['Country\_code'] = df['Country\_code'].replace(['CD'],'COD')

df['Country\_code'] = df['Country\_code'].replace(['CF'],'CAF')

df['Country\_code'] = df['Country\_code'].replace(['CG'],'COG')

df['Country\_code'] = df['Country\_code'].replace(['CH'],'CHE')

df['Country\_code'] = df['Country\_code'].replace(['CI'],'CIV')

df['Country\_code'] = df['Country\_code'].replace(['CK'],'COK')

df['Country\_code'] = df['Country\_code'].replace(['CL'],'CHL')

df['Country\_code'] = df['Country\_code'].replace(['CM'],'CMR')

df['Country\_code'] = df['Country\_code'].replace(['CN'],'CHN')

df['Country\_code'] = df['Country\_code'].replace(['CO'],'COL')

df['Country\_code'] = df['Country\_code'].replace(['CR'],'CRI')

df['Country\_code'] = df['Country\_code'].replace(['CU'],'CUB')

df['Country\_code'] = df['Country\_code'].replace(['CV'],'CPV')

df['Country\_code'] = df['Country\_code'].replace(['CW'],'CUW')

df['Country\_code'] = df['Country\_code'].replace(['CX'],'CXR')

df['Country\_code'] = df['Country\_code'].replace(['CY'],'CYP')

df['Country\_code'] = df['Country\_code'].replace(['CZ'],'CZE')

# Sustitutendo de la columna 'Country\_code' los códigos de país alpha-2 por código alpha-3

df['Country\_code'] = df['Country\_code'].replace(['DE'],'DEU')

df['Country\_code'] = df['Country\_code'].replace(['DJ'],'DJI')

df['Country\_code'] = df['Country\_code'].replace(['DK'],'DNK')

df['Country\_code'] = df['Country\_code'].replace(['DM'],'DMA')

df['Country\_code'] = df['Country\_code'].replace(['DO'],'DOM')

df['Country\_code'] = df['Country\_code'].replace(['DZ'],'DZA')

# Sustitutendo de la columna 'Country\_code' los códigos de país alpha-2 por código alpha-3

df['Country\_code'] = df['Country\_code'].replace(['EC'],'ECU')

df['Country\_code'] = df['Country\_code'].replace(['EE'],'EST')

df['Country\_code'] = df['Country\_code'].replace(['EG'],'EGY')

df['Country\_code'] = df['Country\_code'].replace(['EH'],'ESH')

df['Country\_code'] = df['Country\_code'].replace(['ER'],'ERI')

df['Country\_code'] = df['Country\_code'].replace(['ES'],'ESP')

df['Country\_code'] = df['Country\_code'].replace(['ET'],'ETH')

# Sustitutendo de la columna 'Country\_code' los códigos de país alpha-2 por código alpha-3

df['Country\_code'] = df['Country\_code'].replace(['FI'],'FIN')

df['Country\_code'] = df['Country\_code'].replace(['FJ'],'FJI')

df['Country\_code'] = df['Country\_code'].replace(['FK'],'KLK')

df['Country\_code'] = df['Country\_code'].replace(['FM'],'FSM')

df['Country\_code'] = df['Country\_code'].replace(['FO'],'FRO')

df['Country\_code'] = df['Country\_code'].replace(['FR'],'FRA')

# Sustitutendo de la columna 'Country\_code' los códigos de país alpha-2 por código alpha-3

df['Country\_code'] = df['Country\_code'].replace(['GA'],'GAB')

df['Country\_code'] = df['Country\_code'].replace(['GB'],'GBR')

df['Country\_code'] = df['Country\_code'].replace(['GD'],'GRD')

df['Country\_code'] = df['Country\_code'].replace(['GE'],'GEO')

df['Country\_code'] = df['Country\_code'].replace(['GF'],'GUF')

df['Country\_code'] = df['Country\_code'].replace(['GG'],'GGY')

df['Country\_code'] = df['Country\_code'].replace(['GH'],'GHA')

df['Country\_code'] = df['Country\_code'].replace(['GI'],'GIB')

df['Country\_code'] = df['Country\_code'].replace(['GL'],'GRL')

df['Country\_code'] = df['Country\_code'].replace(['GM'],'GMB')

df['Country\_code'] = df['Country\_code'].replace(['GN'],'GIN')

df['Country\_code'] = df['Country\_code'].replace(['GP'],'GLP')

df['Country\_code'] = df['Country\_code'].replace(['GQ'],'GNQ')

df['Country\_code'] = df['Country\_code'].replace(['GR'],'GRC')

df['Country\_code'] = df['Country\_code'].replace(['GS'],'SGS')

df['Country\_code'] = df['Country\_code'].replace(['GT'],'GTM')

df['Country\_code'] = df['Country\_code'].replace(['GU'],'GUM')

df['Country\_code'] = df['Country\_code'].replace(['GW'],'GNB')

df['Country\_code'] = df['Country\_code'].replace(['GY'],'GUY')

# Sustitutendo de la columna 'Country\_code' los códigos de país alpha-2 por código alpha-3

df['Country\_code'] = df['Country\_code'].replace(['HK'],'HKG')

df['Country\_code'] = df['Country\_code'].replace(['HM'],'HMD')

df['Country\_code'] = df['Country\_code'].replace(['HN'],'HND')

df['Country\_code'] = df['Country\_code'].replace(['HR'],'HRV')

df['Country\_code'] = df['Country\_code'].replace(['HT'],'HTI')

df['Country\_code'] = df['Country\_code'].replace(['HU'],'HUN')

# Sustitutendo de la columna 'Country\_code' los códigos de país alpha-2 por código alpha-3

df['Country\_code'] = df['Country\_code'].replace(['ID'],'IDN')

df['Country\_code'] = df['Country\_code'].replace(['IE'],'IRL')

df['Country\_code'] = df['Country\_code'].replace(['IL'],'ISR')

df['Country\_code'] = df['Country\_code'].replace(['IM'],'IMN')

df['Country\_code'] = df['Country\_code'].replace(['IN'],'IND')

df['Country\_code'] = df['Country\_code'].replace(['IO'],'IOT')

df['Country\_code'] = df['Country\_code'].replace(['IQ'],'IRQ')

df['Country\_code'] = df['Country\_code'].replace(['IR'],'IRN')

df['Country\_code'] = df['Country\_code'].replace(['IS'],'ISL')

df['Country\_code'] = df['Country\_code'].replace(['IT'],'ITA')

# Sustitutendo de la columna 'Country\_code' los códigos de país alpha-2 por código alpha-3

df['Country\_code'] = df['Country\_code'].replace(['JE'],'JEY')

df['Country\_code'] = df['Country\_code'].replace(['JM'],'JAM')

df['Country\_code'] = df['Country\_code'].replace(['JO'],'JOR')

df['Country\_code'] = df['Country\_code'].replace(['JP'],'JPN')

# Sustitutendo de la columna 'Country\_code' los códigos de país alpha-2 por código alpha-3

df['Country\_code'] = df['Country\_code'].replace(['KE'],'KEN')

df['Country\_code'] = df['Country\_code'].replace(['KG'],'KGZ')

df['Country\_code'] = df['Country\_code'].replace(['KH'],'KHM')

df['Country\_code'] = df['Country\_code'].replace(['KI'],'KIR')

df['Country\_code'] = df['Country\_code'].replace(['KM'],'COM')

df['Country\_code'] = df['Country\_code'].replace(['KN'],'KNA')

df['Country\_code'] = df['Country\_code'].replace(['KP'],'PRK')

df['Country\_code'] = df['Country\_code'].replace(['KR'],'KOR')

df['Country\_code'] = df['Country\_code'].replace(['KW'],'KWT')

df['Country\_code'] = df['Country\_code'].replace(['KY'],'CYM')

df['Country\_code'] = df['Country\_code'].replace(['KZ'],'KAZ')

# Sustitutendo de la columna 'Country\_code' los códigos de país alpha-2 por código alpha-3

df['Country\_code'] = df['Country\_code'].replace(['LA'],'LAO')

df['Country\_code'] = df['Country\_code'].replace(['LB'],'LBN')

df['Country\_code'] = df['Country\_code'].replace(['LC'],'LCA')

df['Country\_code'] = df['Country\_code'].replace(['LI'],'LIE')

df['Country\_code'] = df['Country\_code'].replace(['LK'],'LKA')

df['Country\_code'] = df['Country\_code'].replace(['LR'],'LBR')

df['Country\_code'] = df['Country\_code'].replace(['LS'],'LSO')

df['Country\_code'] = df['Country\_code'].replace(['LT'],'LTU')

df['Country\_code'] = df['Country\_code'].replace(['LU'],'LUX')

df['Country\_code'] = df['Country\_code'].replace(['LV'],'LVA')

df['Country\_code'] = df['Country\_code'].replace(['LY'],'LBY')

# Sustitutendo de la columna 'Country\_code' los códigos de país alpha-2 por código alpha-3

df['Country\_code'] = df['Country\_code'].replace(['MA'],'MAR')

df['Country\_code'] = df['Country\_code'].replace(['MC'],'MCO')

df['Country\_code'] = df['Country\_code'].replace(['MD'],'MDA')

df['Country\_code'] = df['Country\_code'].replace(['ME'],'MNE')

df['Country\_code'] = df['Country\_code'].replace(['MF'],'MAF')

df['Country\_code'] = df['Country\_code'].replace(['MG'],'MDG')

df['Country\_code'] = df['Country\_code'].replace(['MH'],'MHL')

df['Country\_code'] = df['Country\_code'].replace(['MK'],'MKD')

df['Country\_code'] = df['Country\_code'].replace(['ML'],'MLI')

df['Country\_code'] = df['Country\_code'].replace(['MM'],'MMR')

df['Country\_code'] = df['Country\_code'].replace(['MN'],'MNG')

df['Country\_code'] = df['Country\_code'].replace(['MO'],'MAC')

df['Country\_code'] = df['Country\_code'].replace(['MP'],'MNP')

df['Country\_code'] = df['Country\_code'].replace(['MQ'],'MTQ')

df['Country\_code'] = df['Country\_code'].replace(['MR'],'MRT')

df['Country\_code'] = df['Country\_code'].replace(['MS'],'MSR')

df['Country\_code'] = df['Country\_code'].replace(['MT'],'MLT')

df['Country\_code'] = df['Country\_code'].replace(['MU'],'MUS')

df['Country\_code'] = df['Country\_code'].replace(['MV'],'MDV')

df['Country\_code'] = df['Country\_code'].replace(['MW'],'MWI')

df['Country\_code'] = df['Country\_code'].replace(['MX'],'MEX')

df['Country\_code'] = df['Country\_code'].replace(['MY'],'MYS')

df['Country\_code'] = df['Country\_code'].replace(['MZ'],'MOZ')

# Sustitutendo de la columna 'Country\_code' los códigos de país alpha-2 por código alpha-3

df['Country\_code'] = df['Country\_code'].replace(['NA'],'NAM')

df['Country\_code'] = df['Country\_code'].replace(['NC'],'NCL')

df['Country\_code'] = df['Country\_code'].replace(['NE'],'NER')

df['Country\_code'] = df['Country\_code'].replace(['NF'],'NFK')

df['Country\_code'] = df['Country\_code'].replace(['NG'],'NGA')

df['Country\_code'] = df['Country\_code'].replace(['NI'],'NIC')

df['Country\_code'] = df['Country\_code'].replace(['NL'],'NLD')

df['Country\_code'] = df['Country\_code'].replace(['NO'],'NOR')

df['Country\_code'] = df['Country\_code'].replace(['NP'],'NPL')

df['Country\_code'] = df['Country\_code'].replace(['NR'],'NRU')

df['Country\_code'] = df['Country\_code'].replace(['NU'],'NIU')

df['Country\_code'] = df['Country\_code'].replace(['NZ'],'NZL')

df['Country\_code'] = df['Country\_code'].replace(['OM'],'OMN')

# Sustitutendo de la columna 'Country\_code' los códigos de país alpha-2 por código alpha-3

df['Country\_code'] = df['Country\_code'].replace(['PA'],'PAN')

df['Country\_code'] = df['Country\_code'].replace(['PE'],'PER')

df['Country\_code'] = df['Country\_code'].replace(['PF'],'PYF')

df['Country\_code'] = df['Country\_code'].replace(['PG'],'PNG')

df['Country\_code'] = df['Country\_code'].replace(['PH'],'PHL')

df['Country\_code'] = df['Country\_code'].replace(['PK'],'PAK')

df['Country\_code'] = df['Country\_code'].replace(['PL'],'POL')

df['Country\_code'] = df['Country\_code'].replace(['PM'],'SPM')

df['Country\_code'] = df['Country\_code'].replace(['PN'],'PCN')

df['Country\_code'] = df['Country\_code'].replace(['PR'],'PRI')

df['Country\_code'] = df['Country\_code'].replace(['PS'],'PSE')

df['Country\_code'] = df['Country\_code'].replace(['PT'],'PRT')

df['Country\_code'] = df['Country\_code'].replace(['PW'],'PLW')

df['Country\_code'] = df['Country\_code'].replace(['PY'],'PRY')

df['Country\_code'] = df['Country\_code'].replace(['QA'],'QAT')

# Sustitutendo de la columna 'Country\_code' los códigos de país alpha-2 por código alpha-3

df['Country\_code'] = df['Country\_code'].replace(['RE'],'REU')

df['Country\_code'] = df['Country\_code'].replace(['RO'],'ROU')

df['Country\_code'] = df['Country\_code'].replace(['RS'],'SRB')

df['Country\_code'] = df['Country\_code'].replace(['RU'],'RUS')

df['Country\_code'] = df['Country\_code'].replace(['RW'],'RWA')

# Sustitutendo de la columna 'Country\_code' los códigos de país alpha-2 por código alpha-3

df['Country\_code'] = df['Country\_code'].replace(['SA'],'SAU')

df['Country\_code'] = df['Country\_code'].replace(['SB'],'SLB')

df['Country\_code'] = df['Country\_code'].replace(['SC'],'SYC')

df['Country\_code'] = df['Country\_code'].replace(['SD'],'SDN')

df['Country\_code'] = df['Country\_code'].replace(['SE'],'SWE')

df['Country\_code'] = df['Country\_code'].replace(['SG'],'SGP')

df['Country\_code'] = df['Country\_code'].replace(['SH'],'SHN')

df['Country\_code'] = df['Country\_code'].replace(['SI'],'SVN')

df['Country\_code'] = df['Country\_code'].replace(['SJ'],'SJM')

df['Country\_code'] = df['Country\_code'].replace(['SK'],'SVK')

df['Country\_code'] = df['Country\_code'].replace(['SL'],'SLE')

df['Country\_code'] = df['Country\_code'].replace(['SM'],'SMR')

df['Country\_code'] = df['Country\_code'].replace(['SN'],'SEN')

df['Country\_code'] = df['Country\_code'].replace(['SO'],'SOM')

df['Country\_code'] = df['Country\_code'].replace(['SR'],'SUR')

df['Country\_code'] = df['Country\_code'].replace(['SS'],'SSD')

df['Country\_code'] = df['Country\_code'].replace(['ST'],'STP')

df['Country\_code'] = df['Country\_code'].replace(['SV'],'SLV')

df['Country\_code'] = df['Country\_code'].replace(['SX'],'SXM')

df['Country\_code'] = df['Country\_code'].replace(['SY'],'SYR')

df['Country\_code'] = df['Country\_code'].replace(['SZ'],'SWZ')

# Sustitutendo de la columna 'Country\_code' los códigos de país alpha-2 por código alpha-3

df['Country\_code'] = df['Country\_code'].replace(['TC'],'TCA')

df['Country\_code'] = df['Country\_code'].replace(['TD'],'TCD')

df['Country\_code'] = df['Country\_code'].replace(['TF'],'ATF')

df['Country\_code'] = df['Country\_code'].replace(['TG'],'TGO')

df['Country\_code'] = df['Country\_code'].replace(['TH'],'THA')

df['Country\_code'] = df['Country\_code'].replace(['TZ'],'TZA')

df['Country\_code'] = df['Country\_code'].replace(['TJ'],'TJK')

df['Country\_code'] = df['Country\_code'].replace(['TK'],'TKL')

df['Country\_code'] = df['Country\_code'].replace(['TL'],'TLS')

df['Country\_code'] = df['Country\_code'].replace(['TM'],'TKM')

df['Country\_code'] = df['Country\_code'].replace(['TN'],'TUN')

df['Country\_code'] = df['Country\_code'].replace(['TO'],'TON')

df['Country\_code'] = df['Country\_code'].replace(['TR'],'TUR')

df['Country\_code'] = df['Country\_code'].replace(['TT'],'TTO')

df['Country\_code'] = df['Country\_code'].replace(['TV'],'TUV')

df['Country\_code'] = df['Country\_code'].replace(['TW'],'TWN')

# Sustitutendo de la columna 'Country\_code' los códigos de país alpha-2 por código alpha-3

df['Country\_code'] = df['Country\_code'].replace(['UA'],'UKR')

df['Country\_code'] = df['Country\_code'].replace(['UG'],'UGA')

df['Country\_code'] = df['Country\_code'].replace(['US'],'USA')

df['Country\_code'] = df['Country\_code'].replace(['UY'],'URY')

df['Country\_code'] = df['Country\_code'].replace(['UZ'],'UZB')

# Sustitutendo de la columna 'Country\_code' los códigos de país alpha-2 por código alpha-3

df['Country\_code'] = df['Country\_code'].replace(['VA'],'VAT')

df['Country\_code'] = df['Country\_code'].replace(['VC'],'VCT')

df['Country\_code'] = df['Country\_code'].replace(['VE'],'VEN')

df['Country\_code'] = df['Country\_code'].replace(['VG'],'VGB')

df['Country\_code'] = df['Country\_code'].replace(['VI'],'VIR')

df['Country\_code'] = df['Country\_code'].replace(['VN'],'VNM')

df['Country\_code'] = df['Country\_code'].replace(['VU'],'VUT')

# Sustitutendo de la columna 'Country\_code' los códigos de país alpha-2 por código alpha-3

df['Country\_code'] = df['Country\_code'].replace(['WF'],'WLF')

df['Country\_code'] = df['Country\_code'].replace(['WS'],'WSM')

df['Country\_code'] = df['Country\_code'].replace(['YE'],'YEM')

df['Country\_code'] = df['Country\_code'].replace(['YT'],'MYT')

df['Country\_code'] = df['Country\_code'].replace(['ZA'],'ZAF')

df['Country\_code'] = df['Country\_code'].replace(['ZM'],'ZMB')

df['Country\_code'] = df['Country\_code'].replace(['ZW'],'ZWE')

# rectificando valores nulos

country\_null = df[df.Country\_code.isnull()]

country\_null.head()

df['Country\_code'] = df['Country\_code'].replace([np.nan],'NAM')

# rectificando valores nulos

country\_null = df[df.Country\_code.isnull()]

country\_null.head()

# exportando dataFrame sin índices a un nuevo archivo csv con códigos de país alpha-3

df.to\_csv("WHO-COVID-19-global-data-alpha3.csv", index = False)

# Agrupando países

dfSumary = df.groupby(['Country\_code'])[['New\_cases', 'New\_deaths']].apply(sum)

dfSumary.head()

# exportando dataFrame con índices a un nuevo archivo csv con códigos de país alpha-3

dfSumary.to\_csv("WHO-COVID-19-summary.csv", index= True)

### index.html

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <!-- Load d3.js -->

    <script src="https://d3js.org/d3.v6.js"></script>

    <script defer src="resources/main.js"></script>

    <!-- CSS only -->

    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet"

        integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3" crossorigin="anonymous">

    <link rel="stylesheet" type="text/css" href="resources/main.css" />

    <title>Tendencias en D3</title>

</head>

<body>

    <div class="collapse" id="navbarToggleExternalContent">

        <div class="bg-dark p-4">

            <h5 class="text-white h4">Collapsed content</h5>

            <span class="text-muted">Toggleable via the navbar brand.</span>

        </div>

    </div>

    <nav class="navbar navbar-dark bg-dark">

        <div class="container-fluid">

            <button class="navbar-toggler" type="button" data-bs-toggle="collapse"

                data-bs-target="#navbarToggleExternalContent" aria-controls="navbarToggleExternalContent"

                aria-expanded="false" aria-label="Toggle navigation">

                <span class="navbar-toggler-icon"></span>

            </button>

        </div>

    </nav>

    <div>

        <h2>

            COVID-19

        </h2>

    </div>

    <br>

    <div class="left">

        <!-- Create an element where the map will take place -->

        <svg id="my\_dataviz" width="700" height="600">

        </svg>

        <div class="datascale" id="datascale">

        </div>

    </div>

    <div class="right" id="barplot">

        <p id="barplotTitle"></p>

        <svg id="my\_barplot">

        </svg>

        <table class="table">

            <thead>

                <tr>

                    <th scope="col"></th>

                    <th scope="col">Total</th>

                </tr>

            </thead>

            <tbody>

                <tr>

                    <th scope="row">Contagiados</th>

                    <td><p id="contagiados"></p></td>

                </tr>

                <tr>

                    <th scope="row">Fallecidos</th>

                    <td><p id="fallecidos"></p></td>

                </tr>

            </tbody>

        </table>

    </div>

</body>

</html>

### main.css

body{

    text-align: center;

}

div{

    border-radius: 10px;

}

p{

    margin: 0px;

}

h2{

    font-weight: bold;

}

.left{

    float: left;

    position: relative;

    width: 60%;

    padding: 5px;

}

.right{

    background-color: beige;

    text-align: center;

    float: left;

    position: relative;

    width: 40%;

    padding: 5px;

}

.legendTitle{

    font-size: 15px;

    font-weight: bold;

}

.legendAxis{

    font-size: 9px;

}

### main.js

/\*

declarando constante COVID que contendrá el resumen general de contagiados

y fallecidos

\*/

COVID = []

// variable que albergará el país seleccionado del mapa

sumary = [

    { group: "Contagiados", value: 1 },

    { group: "Fallecidos", value: 1 }

]

// leyendo archivo CSV y cargando su contenido al array COVID

d3.csv("WHO-COVID-19-summary.csv").then((res) => {

    res.forEach(element => {

        COVID.push(element)

    });

})

// Definiendo parámetros de ancho y alto para del SVG del mapa

const svg = d3.select("svg"),

    width = +svg.attr("width"),

    height = +svg.attr("height");

// Realizando la proyección y el escalado

const path = d3.geoPath();

const projection = d3.geoMercator()

    .scale(115)

    .center([0, 20])

    .translate([width / 2, height / 2]);

// Definiendo la escala de colores para el mapa

// La escala de color responde a la cantidad de contagiados por país

const data = new Map();

const colorScale = d3.scaleThreshold()

    .domain([100000, 500000, 1000000, 5000000, 10000000, 25000000, 50000000])

    .range(d3.schemeReds[5]);

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Begin: Legenda del Mapa \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

const legendWidth = Math.min(width \* 0.50, 400);

// Asignando el svg para la escala de datos

const svg\_legend = svg.append("g")

    .attr("class", "legendWrapper")

    .attr("transform",

        "translate(" + (width \* 0.73) + "," + (16) + ")");

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*End:  Legenda del Mapa \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*BAR PLOT\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// definiendo margenes y ancho y largo del barplot

const margin = {

    top: 30,

    right: 30,

    bottom: 70,

    left: 60

},

    width\_bp = 460 - margin.left - margin.right,

    height\_bp = 400 - margin.top - margin.bottom;

// Asignando el svg para el barplot sobre el div #barplot

const svg\_bp = d3.select("#my\_barplot")

    //    .append("svg")

    .attr("width", width\_bp + margin.left + margin.right)

    .attr("height", height\_bp + margin.top + margin.bottom)

    .append("g")

    .attr("transform",

        `translate(${margin.left},${margin.top})`);

// Inicializando valores para el eje X

const x = d3.scaleBand()

    .range([0, width\_bp])

    .padding(0.2);

const xAxis = svg\_bp.append("g")

    .attr("transform", `translate(0,${height\_bp})`)

// Inicializando valores para el eje Y

const y = d3.scaleLinear()

    .range([height\_bp, 0]);

const yAxis = svg\_bp.append("g")

    .attr("class", "myYaxis")

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*BAR PLOT\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// Cargando información externa

Promise.all([

    //Obteniendo todos las coordenadas del póligo para cada país del mundo

    d3.json("https://raw.githubusercontent.com/holtzy/D3-graph-gallery/master/DATA/world.geojson"),

    //Obteniendo el resumen de reportes y fallecidos por país

    d3.csv("WHO-COVID-19-summary.csv", function (d) {

        data.set(d.Country\_code, +d.New\_cases)

    })]).then(function (loadData) {

        let topo = loadData[0]

        let mouseOver = function (d) {

            d3.selectAll(".Country")

                .transition()

                .duration(200)

                .style("opacity", .5)

            d3.select(this)

                .transition()

                .duration(200)

                .style("opacity", 1)

                .style("stroke", "black")

        }

        let mouseLeave = function (d) {

            d3.selectAll(".Country")

                .transition()

                .duration(200)

                .style("opacity", .8)

            d3.select(this)

                .transition()

                .duration(200)

                .style("stroke", "transparent")

        }

        // Dibujando el mapa

        svg.append("g")

            .selectAll("path")

            .data(topo.features)

            .enter()

            .append("path")

            // Dibujando cada país

            .attr("d", d3.geoPath()

                .projection(projection)

            )

            // Asignando color a cada país

            .attr("fill", function (d) {

                d.total = data.get(d.id) || 0;

                return colorScale(d.total);

            })

            .style("stroke", "transparent")

            .attr("class", function (d) { return "Country" })

            .style("opacity", .8)

            //Asignando evento cuando el puntero del mouse está sobre el área cada polígono del país

            .on("mouseover", mouseOver)

            //Asignando evento cuando el puntero se retira del área del polígono del país

            .on("mouseleave", mouseLeave)

            //Cuando el usuario proporciona click sobre el polígono se redibuja el barplot

            .on("click", function (d, i) {

                drawBarplot(i.id);

                drawLegendPointer(i.id);

            })

        // Dibuja la legenda del Mapa

        drawMapLegend();

    })

function drawBarplot(i) {

    //Se busca la información relacionda con el país selecciondo

    //dentro del array inicial COVID

    let res = COVID.find(pais => pais.Country\_code === i)

    //Limpia el array

    sumary = []

    //Asignando los elementos encontrados en la búsqueda

    //al array sumary

    sumary.push(

        {

            'group': 'Contagiados',

            'value': res.New\_cases

        },

        {

            'group': 'Fallecidos',

            'value': res.New\_deaths

        }

    )

    // Actualizando el eje x en función del país seleccionado

    x.domain(sumary.map(d => d.group))

    xAxis.call(d3.axisBottom(x))

    // Actualizando el eje y en función del país seleccionado

    // reescalando el eje dependiendo de la cantidad de personas contagiadas

    var contagiados = parseInt(sumary[0].value, 10)

    if (contagiados <= 100000) {

        max = 100000

    } else if (contagiados > 100000 && contagiados <= 500000) {

        max = 500000

    } else if (contagiados > 500000 && contagiados <= 1000000) {

        max = 1000000

    } else if (contagiados > 1000000 && contagiados <= 5000000) {

        max = 5000000

    } else if (contagiados > 5000000 && contagiados <= 10000000) {

        max = 10000000

    } else if (contagiados > 10000000 && contagiados <= 25000000) {

        max = 25000000

    } else if (contagiados > 25000000 && contagiados <= 50000000) {

        max = 50000000

    }

    y.domain([0, max]);

    // Asignando animación

    yAxis.transition().duration(1000).call(d3.axisLeft(y));

    var u = svg\_bp.selectAll("rect")

        .data(sumary)

    //Dibujando las barras y asignando animaciones de transición

    u.join("rect")

        .transition()

        .duration(1000)

        .attr("x", d => x(d.group))

        .attr("y", d => y(d.value))

        .attr("width", x.bandwidth())

        .attr("height", d => height\_bp - y(d.value))

        .attr("fill", "rgb(223, 163, 0, 0.863)")

    //Asignando dinámicamente el título del gráfico

    d3.select("#barplotTitle")

        .text(res.Country)

        .attr("class", "legendTitle")

    //Asignando dinámicamente el resumen del país

    d3.select("#contagiados")

        .text(parseInt(res.New\_cases).toLocaleString('es-MX'))

    d3.select("#fallecidos")

        .text(parseInt(res.New\_deaths).toLocaleString('es-MX'))

}

// Función encargada de dibujar la legenda del mapa

function drawMapLegend() {

    /\* -- Definición del gradiente de forma lineal -- \*/

    var def\_lg = svg.append("defs")

        .append("linearGradient")

        .attr("id", "def\_linear\_gradient")

        .attr("x1", "0%").attr("y1", "0%")

        .attr("x2", "100%").attr("y2", "0%");

    // Color de finalización del gradiente

    def\_lg.append("stop")

        .attr("offset", "0%")

        .style("stop-color", d3.max(colorScale.range()))

        .style("stop-opacity", 1)

    // Color de inicio del gradiente

    def\_lg.append("stop")

        .attr("offset", "100%")

        .style("stop-color", d3.min(colorScale.range()))

        .style("stop-opacity", 1)

    /\* -- \*/

    /\* -- Dibujando el rectángulo -- \*/

    svg\_legend.append("rect")

        .attr("class", "legendRect")

        .attr("x", -legendWidth / 2)

        .attr("y", 0)

        .attr("width", legendWidth)

        .attr("height", 5)

        .style("fill", "url(#def\_linear\_gradient)");

    /\* -- \*/

    /\* -- Título de la leyenda -- \*/

    svg\_legend.append("text")

        .attr("class", "legendTitle")

        .attr("x", 0)

        .attr("y", -5)

        .style("text-anchor", "middle")

        .text("03/01/2020 al 29/11/2021");

    /\* -- \*/

    /\* -- Leyenda de la escala de los datos -- \*/

    // Escala para cantidades

    var Leg\_xScale = d3.scaleLinear()

        .domain([0, 50000000])

        .range([-legendWidth / 2, legendWidth / 2])

    // Axis para dar formato a los datos (abreviación de Millones)

    var Leg\_xAxis = d3.axisBottom(Leg\_xScale).tickFormat(function (d) {

        return d / 1000000 + "MM";

    })

    //Configura el eje X

    svg\_legend.append("g")

        .attr("class", "legendAxis")

        .attr("transform", "translate(0," + (5) + ")")

        .call(Leg\_xAxis);

    /\* -- \*/

}

// Función drawLegendPointer: dibuja sobre la escala de color

// un puntero con la cantidad de casos para una rápida identificación.

function drawLegendPointer(i) {

    // Filtra los datos específicos del país seleccionado

    let res = COVID.find(pais => pais.Country\_code === i);

    // El valor a representar en el apuntados, es la suma de los nuevos casos

    var value = parseInt(res.New\_cases) + parseInt(res.New\_deaths);

    /\* -- Leyenda de la escala de los datos -- \*/

    var PtScale = d3.scaleLinear()

        .range([-legendWidth / 2, legendWidth / 2])

        .domain([0, 50000000]);

    // Definición del objeto apuntador (Diamante)

    var symb = d3.symbol().type(d3.symbolDiamond).size(30);

    // Dibuja el apuntador

    svg\_legend.selectAll("path")

        .attr("d", symb)

        .attr("fill", "black")

        .attr("transform", "translate(" + PtScale(value) + ", -2)");

}

function showTitleBarPlot(res) {

}

# Conclusiones

1. Los paquetes de pandas y numpy de Python proveen las herramientas necesarias para la correcta preparación de un dataset.
2. Un mapa coroplético es un mapa temático coloreado bajo la misma gama cromática de un color.
3. Para dibujar un mapa con D3 es necesario obtener todos los puntos de geolocalización del país para trazar el polígono respectivo.
4. La librería D3 cuenta con una amplia gama de ejemplos para el correcto desarrollo de visualizaciones.
5. D3 permite asociar múltiples eventos a una visualización.
6. El evento “mouseover” hace referencia al momento en que el puntero del mouse está sobre un elemento en particular.
7. El evento “mouseleave” hace referencia al momento en que el puntero del mouse se retira del área del elemento sobre el que se posicionó.
8. Los eventos manejados por javascript son los mismos que se pueden asociar directamente con la librería D3.js.
9. Previo a programar la visualización, es necesario tener claro el tipo de gráfica y un boceto inicial, ya que la librería D3.js nos provee de varias herramientas que pueden ser utilizadas de diversas formas. Tiene objetos bastante flexibles por lo es necesario tener claro el propósito del uso.
10. Las funciones de javascript para gestionar escalas que trae D3.js tienen una gran utilidad, nos permites realizar mapeos sobre los datos que tenemos y lo que queremos mostrar. Podemos hacer un mapeo de posiciones, longitudes, textos y color (Cook, 2021).

# Recomendaciones

1. Independientemente de la herramienta de visualización que se utilice no hay que olvidar los principios gestaltistas para el correcto desarrollo de la visualización.
2. Considerar que el correcto proceso para desarrollo de una visualización consta de Investigación, Selección de datos, Creación del Boceto, Elaboración, Percepción y Conocimiento.
3. Para crear un mapa con D3 es necesario utilizar todos los puntos geolocalización de los países, una alternativa para este caso particular es el siguiente enlace: <https://raw.githubusercontent.com/holtzy/D3-graph-gallery/master/DATA/world.geojson>
4. Para profundizar sobre los eventos que se pueden asociar a una visualización es recomendable considerar la documentación correcta, como por ejemplo <https://developer.mozilla.org/es/docs/Web/Events>

# Bibliografía

Cook, P. (2021). *D3 in Depth*. Obtenido de Scale functions: https://www.d3indepth.com/scales/

Torres, D. (2009). Aproximaciones a la visualización como disciplina científica. *ACIMED, 20*(6). Obtenido de http://scielo.sld.cu/scielo.php?script=sci\_arttext&pid=S1024-94352009001200005

UNIR. (2020). *Tema 6: Definición y tipologías de gráficos.*