

# Análisis de datos

Integrantes:

Nayely Ayol

Alejandro Guanoluisa

Carlos Simbaña

Oscar Vasquez



## Objetivo General

Aplicar técnicas del análisis de datos a registros extraídos de diferentes fuentes.

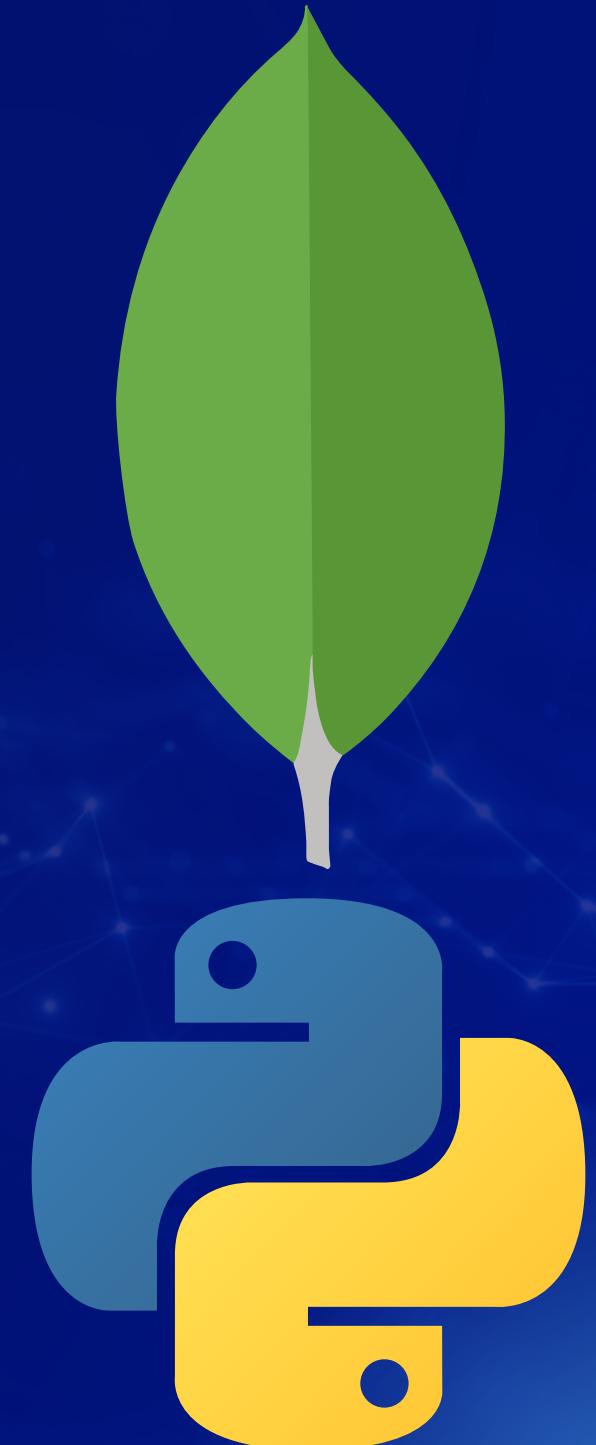
## Objetivos Específicos

- Usar herramientas de gestión de datos relaciones y no relacionales.
- Realizar una limpieza de datos para asegurar resultados sin errores.
- Diseñar y elaborar visualización y dashboards de los datos

# Metodología



HERRAMIENTAS	
Bases relacionales	Sql server, My SQL
Bases no relacionales	Mongo DB, CouchDB
Extracción, transformación y limpieza de datos	Python – Librería Pandas
Análisis de sentimientos	Python o Jupyter Notebook – Librería TextBold
Visualizaciones	Power BI
Datos en la nube	MongoDB Atlas
Transformación de datos	Python, csv o json
Cronograma de actividades	GanttPRO



# Cronograma

Nombre de tarea	Asignado	Fecha de inicio	Fecha final	Estado
1 Tarea de resumen		07/21/2025	08/04/2025	
1.1 Extracción de datos desde ...	AG Alejandro Guanoluisa	07/21/2025	07/21/2025	Terminado
1.2 Limpieza y transformación ...	AG Alejandro Guanoluisa	07/23/2025	07/23/2025	Terminado
1.3 Almacenamiento en bases ...	AG OV	07/25/2025	07/28/2025	Terminado
1.4 Sincronización entre siste...	AG OV	07/30/2025	07/30/2025	Terminado
1.5 Análisis de sentimientos	CS Carlos Simbaña	07/31/2025	07/31/2025	Terminado
1.6 Datos para la visualización	CS Carlos Simbaña	07/31/2025	08/01/2025	Terminado
1.7 Visualización en Dashborads	N Nayely Ayol	08/01/2025	08/01/2025	Terminado
1.8 Definición de casos de est...	CS Carlos Simbaña	08/01/2025	08/01/2025	Terminado
1.9 Estructura de las presentac...	N Nayely Ayol	08/01/2025	08/01/2025	Terminado
1.10 Edición de videos	N Nayely Ayol	08/04/2025	08/04/2025	Terminado
1.11 Creación del informe	N Nayely Ayol	08/04/2025	08/04/2025	Terminado
1.12 Diapositivas	N AG +2	08/04/2025	08/04/2025	Terminado

Asignado      Fecha de inicio

Personas

- AG Alejandro Guan... 4 h
- OV Oscar Vasquez 4 h

07/25/2025

# Diagrama de Gantt

**Cronograma\_AD** ★ Sin estado

Calendario | Tablero | Lista | **Calendario** | Carga de trabajo | Personas | Panel de control

< Julio 2025 > Hoy ¿Falta algo?

DO	LU	MA
29 Jun	30 Jun	1 Jul
6	7	8
13	14	15
20	21	22
27	28	29

**Extracción de datos desde diferentes fuentes en la web**

Creador: Nayely Ayol, 08/04/2025

100% TERMINADO ALTA Tiempo registrado: 0

AG Alejandro Guanolusa Adjuntar archivos Agregar nueva dependencia Registrar tiempo

Fecha de inicio: 07/21/2025 09:00AM - Fecha final: 07/21/2025 06:00PM Estimación: 8h Duración: 8h Fecha límite:

Agregar descripción de la tarea

Dejar comentario

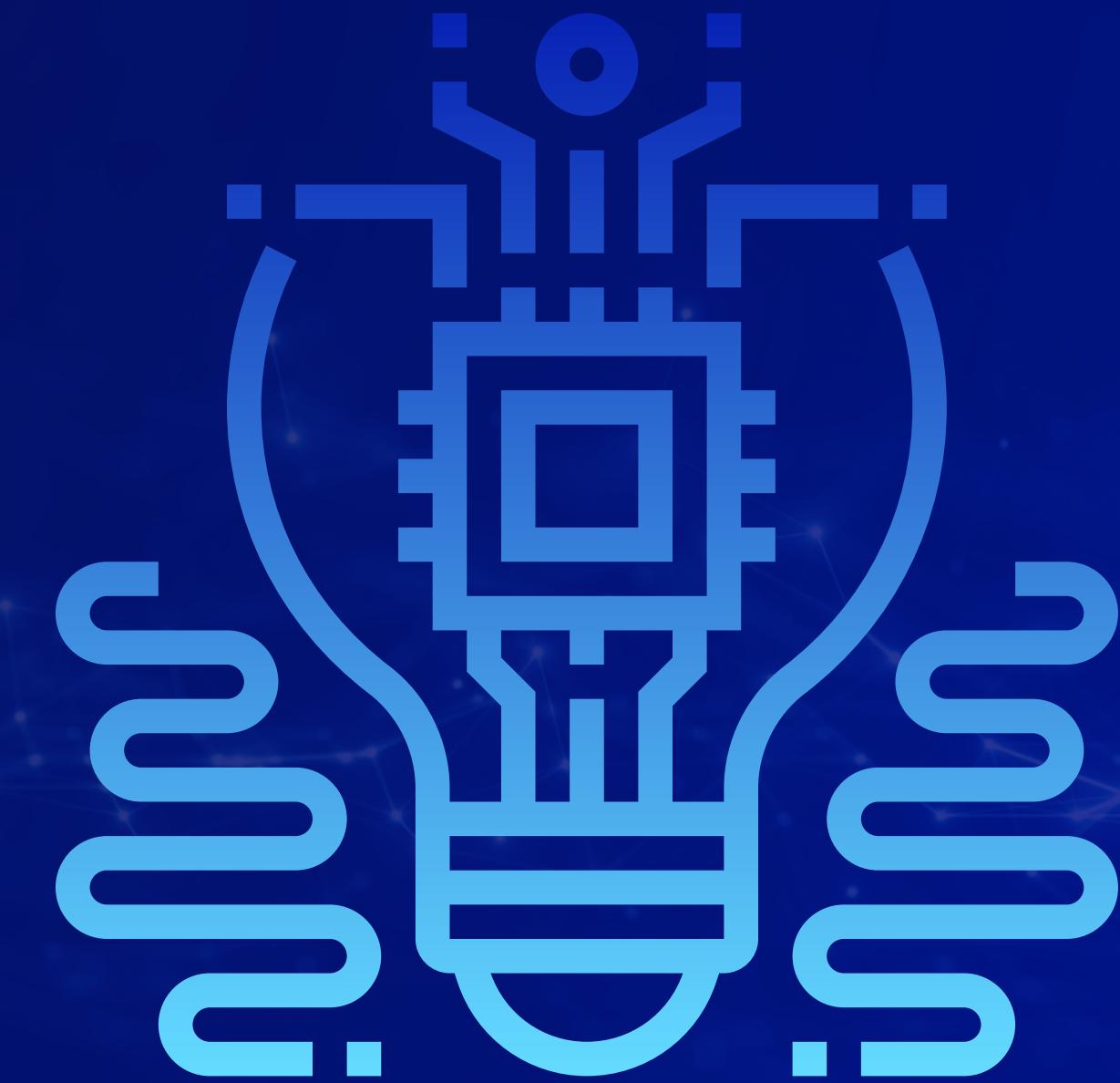
Activar Windows  
Ve a Configuración para activar Windows.



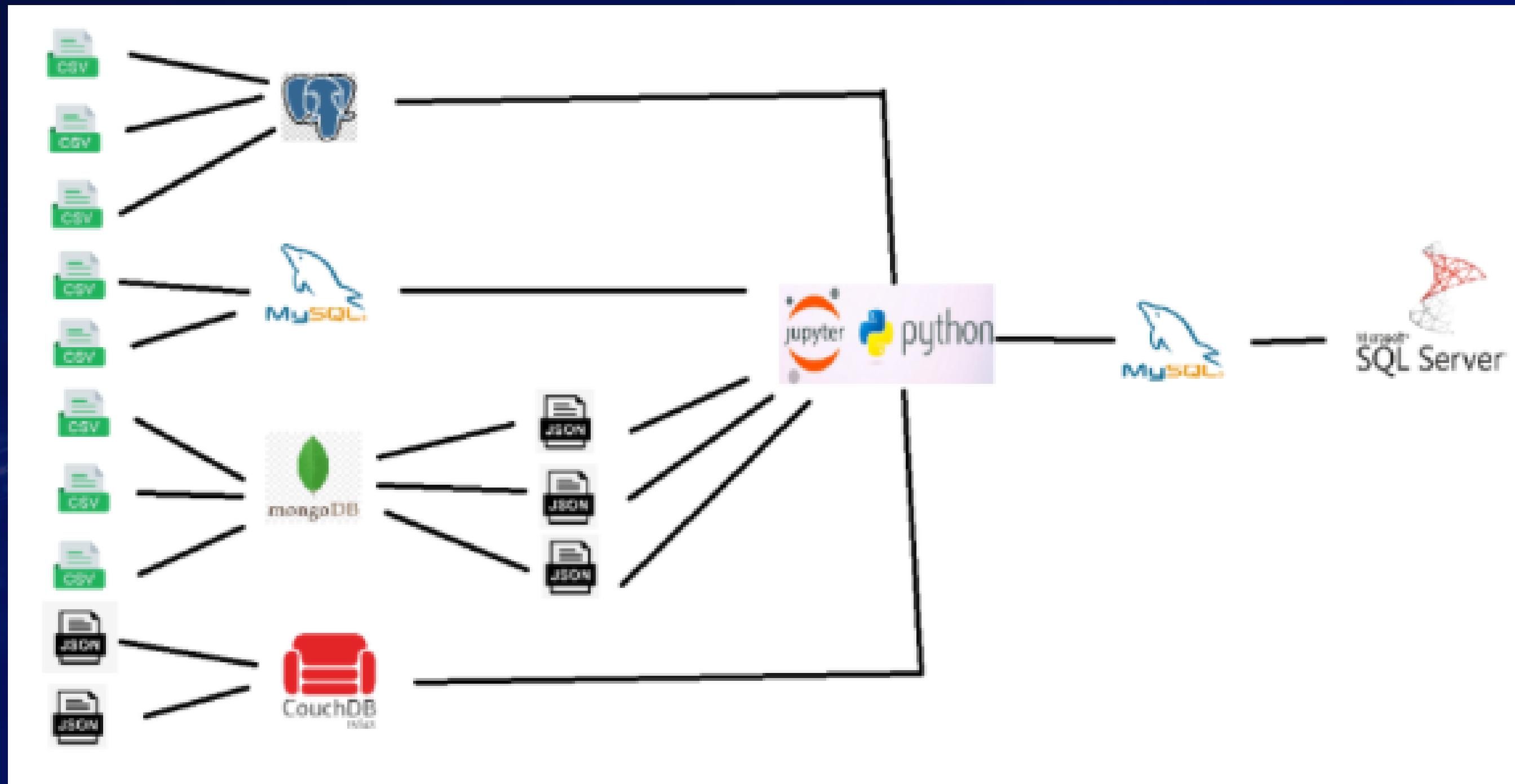
Permite organizar y planificar de mejor manera un proyecto para que este se culmine sin mayores contratiempos.

# Fuentes de datos

Fuente	Base de datos extraída
INEC	<ul style="list-style-type: none"><li>- Siniestros de tránsito en el Ecuador</li><li>- Vehículos matriculados</li></ul>
Kaggle	<ul style="list-style-type: none"><li>- Rutina matutina</li><li>- Productos falsificados</li><li>- Datos de Empleados</li><li>- Datos de Netflix</li><li>- Categoría de noticias</li></ul>
Worldometer	<ul style="list-style-type: none"><li>- Personas desnutridas en el mundo</li><li>- Consumo de agua en el mundo</li><li>- Reservas de petróleo</li></ul>



# Arquitectura



# pgAdmin

```
CREATE TABLE vehiculos_matriculados (
    id SERIAL PRIMARY KEY,
    provincia VARCHAR(50),
    mes VARCHAR(20),
    pasajeros INT,
    marca VARCHAR(100),
    clase VARCHAR(50),
    servicio INT,
    tonelaje DECIMAL(5,2),
    modelo INT,
    estrapasajero INT
);
```

```
C:\Users\Flia Guanoluisa>psql -U postgres -d vehiculos_matriculados
Contraseña para usuario postgres:
psql (16.9)
ADVERTENCIA: El código de página de la consola (850) difiere del código
de página de Windows (1252).
Los caracteres de 8 bits pueden funcionar incorrectamente.
Vea la página de referencia de psql «Notes for Windows users»
para obtener más detalles.
Digite «help» para obtener ayuda.

vehiculos_matriculados=# \copy vehiculos_matriculados(provincia,mes,pasajeros,marca,clase,servicio,tonelaje,modelo,estrapasajero)
   FROM 'C:/Users/Flia Guanoluisa/OneDrive/Documentos/Proyecto_analisis/vehiculos_matriculados_2023_limpio.csv' DELIMITER ',' CSV HEADER;
COPY 3865967

vehiculos_matriculados=# SELECT * FROM vehiculos_matriculados LIMIT 10;
 id | provincia | mes | pasajeros | marca | clase | servicio | tonelaje | modelo | estrapasajero
---+-----+-----+-----+-----+-----+-----+-----+-----+-----+
  1 | guayas   | mayo |        2 | SHINERAY | motocicleta |      3 | 0.25 | 2023 |      1
  2 | guayas   | mayo |        2 | SHINERAY | motocicleta |      3 | 0.25 | 2023 |      1
  3 | guayas   | mayo |        2 | SHINERAY | motocicleta |      3 | 0.25 | 2023 |      1
  4 | guayas   | mayo |        2 | SHINERAY | motocicleta |      3 | 0.25 | 2023 |      1
  5 | guayas   | mayo |        2 | SHINERAY | motocicleta |      3 | 0.25 | 2023 |      1
  6 | guayas   | mayo |        2 | SHINERAY | motocicleta |      3 | 0.25 | 2023 |      1
  7 | guayas   | mayo |        2 | SHINERAY | motocicleta |      3 | 0.25 | 2023 |      1
  8 | guayas   | mayo |        2 | SHINERAY | motocicleta |      3 | 0.25 | 2023 |      1
  9 | guayas   | mayo |        2 | SHINERAY | motocicleta |      3 | 0.25 | 2023 |      1
 10 | guayas   | mayo |        2 | SHINERAY | motocicleta |      3 | 0.25 | 2023 |      1
(10 filas)
```

# pgAdmin

```
CREATE TABLE siniestros_transito (
    id SERIAL PRIMARY KEY,
    provincia VARCHAR(50),
    mes VARCHAR(20),
    dia VARCHAR(20),
    hora INT,
    clase VARCHAR(50),
    causa TEXT,
    num_fallecidos INT,
    num_lesionados INT,
    total_victimas INT
);
```

```
C:\Users\Flia Guanoluisa>psql -U postgres -d siniestros_transito
Contraseña para usuario postgres:
psql (16.9)
ADVERTENCIA: El código de página de la consola (850) difiere del código
de página de Windows (1252).
Los caracteres de 8 bits pueden funcionar incorrectamente.
Vea la página de referencia de psql «Notes for Windows users»
para obtener más detalles.

Digite «help» para obtener ayuda.

siniestros_transito=# \copy siniestros_transito(provincia, mes, dia, hora, clase, causa, num_fallecidos, num_lesionados, total_victimas) FROM 'C:/Users/Flia Guanoluisa/OneDrive/Documentos/Proyecto_analisis/siniestros_transito.csv' DELIMITER
';' CSV HEADER;
COPY 20994
siniestros_transito=# SELECT * FROM siniestros_transito LIMIT 5;
 id | provincia | mes | dia | hora | clase | causa | num_fallecidos | num_lesionados | total_victimas
----+-----+-----+-----+-----+-----+-----+-----+-----+
 1 | guayas | enero | domingo |   0 | atropellos | impericia e imprudencia del conductor |   0 |
 1 |           |       |          |   1 |
 2 | guayas | enero | domingo |   5 | atropellos | impericia e imprudencia del conductor |   0 |
 1 |           |       |          |   1 |
 3 | guayas | enero | domingo |  11 | atropellos | impericia e imprudencia del conductor |   0 |
 1 |           |       |          |   1 |
 4 | guayas | enero | domingo |   9 | atropellos | impericia e imprudencia del conductor |   0 |
 1 |           |       |          |   1 |
 5 | guayas | enero | domingo |   9 | atropellos | impericia e imprudencia del conductor |   0 |
 1 |           |       |          |   1 |
(5 filas)
```

# pgAdmin

```
CREATE TABLE reservas_petroleo (
    id SERIAL PRIMARY KEY,
    pais VARCHAR(100),
    reservas_2016 BIGINT,
    porcentaje_mundial DECIMAL(5,2)
);
```

```
C:\Users\Flia Guanoluisa>psql -U postgres -d reservas_petroleo
Contraseña para usuario postgres:
psql (16.9)
ADVERTENCIA: El código de página de la consola (850) difiere del código
de página de Windows (1252).
Los caracteres de 8 bits pueden funcionar incorrectamente.
Vea la página de referencia de psql «Notes for Windows users»
para obtener más detalles.

Digite «help» para obtener ayuda.
```

```
reservas_petroleo=# \copy reservas_petroleo(pais, reservas_2016, porcentaje_mundial) FROM 'C:\Users\Flia Guanoluisa\OneDrive\Documentos\Proyecto_analisis\reservas_petroleo_limpio.csv' DELIMITER ',' CSV HEADER;
COPY 98
```

```
reservas_petroleo=# select * from reservas_petroleo limit 5;
 id |      pais      | reservas_2016 | porcentaje_mundial
----+-----+-----+-----
  1 | Venezuela     | 299953000000 |        18.17
  2 | Saudi Arabia  | 266578000000 |        16.15
  3 | Canada        | 170863000000 |        10.35
  4 | Iran           | 157530000000 |         9.54
  5 | Iraq           | 143069000000 |         8.67
(5 filas)
```

# MySQL Workbench

The screenshot shows the MySQL Workbench interface. In the top-left pane, there is a code editor containing the following SQL script:

```
4 *+ CREATE TABLE rutina_matutina (
5     id INT AUTO_INCREMENT PRIMARY KEY,
6     fecha DATE,
7     horas_sueno FLOAT,
8     minutos_meditacion INT,
9     minutos_ejercicio INT,
10    tipo_desayuno VARCHAR(50),
11    journaling BOOLEAN,
12    puntaje_productividad INT,
13    estado_animo VARCHAR(20),
14    notas TEXT
15 );
16 *+ SELECT * FROM rutina_matutina;
17
```

In the bottom-right pane, there is a result grid displaying the data from the 'rutina\_matutina' table:

	<b>id</b>	<b>fecha</b>	<b>horas_sueno</b>	<b>minutos_meditacion</b>	<b>minutos_ejercicio</b>	<b>tipo_desayuno</b>	<b>journaling</b>	<b>puntaje_productividad</b>
1	1	2023-02-09	7	5	10	Heavy	1	5
2	2	2023-04-06	7	5	60	Light	1	5
3	3	2023-04-09	6	30	20	Heavy	1	5
4	4	2023-04-13	7	5	40	Light	0	5
5	5	2023-04-22	7	20	0	Carb-rich	1	5
6	6	2023-05-03	7	0	20	Heavy	0	5
7	7	2023-06-06	6	20	50	Protein-rich	0	5

A vertical toolbar on the right side of the result grid provides options for 'Result Grid', 'Form Editor', and other data manipulation tools.

# MySQL Workbench

# Fuentes de Datos Utilizadas



CSV (valores separados por comas)



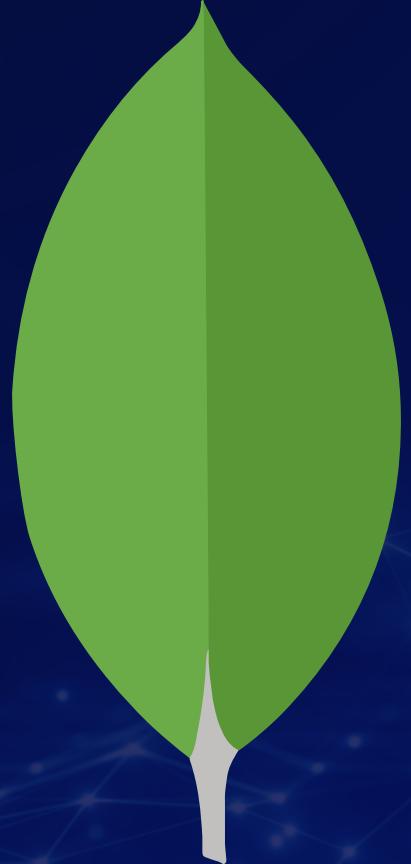
JSON (datos estructurados en formato clave-valor)



SQL (sentencias estructuradas para bases de datos relacionales)



# Estructura en MongoDB



MongoDB fue utilizado como base de datos NoSQL tipo documental, donde los datos provenientes de archivos CSV fueron transformados en objetos JSON utilizando la librería Pandas de Python.

```
import pandas as pd
```

# MongoDB Compass

The screenshot shows the MongoDB Compass interface for the 'empleados' collection in the 'admin' database of the 'ProyectoFinal' project. The 'Documents' tab is selected, displaying 10.0K documents. A search bar at the top allows querying by field and value or generating a query. Below the search bar are buttons for 'ADD DATA', 'EXPORT DATA', 'UPDATE', and 'DELETE'. The document list shows two entries:

```
_id: ObjectId('688ffa199c97ecfc239a084')
Name : "William Atherton"
Age : 43
Gender : "Male"
Department : "Marketing"
Job_Title : "Manager"
Experience_Years : 20
Education_Level : "Master"
Location : "New York"
Salary : 135000
```

```
_id: ObjectId('688ffa199c97ecfc239a085')
Name : "Maria Isch"
Age : 41
Gender : "Female"
Department : "Product"
Job_Title : "Manager"
Experience_Years : 18
Education_Level : "Master"
Location : "Chicago"
Salary : 125000
```

Each document has a set of edit and delete icons to its right.

# MongoDB Compass

The screenshot shows the MongoDB Compass interface for the 'consumo\_agua' collection in the 'admin' database of the 'ProyectoFinal' project. The 'Documents' tab is selected, displaying 181 documents. The interface includes a query builder, search, and filtering tools at the top, and a toolbar with 'ADD DATA', 'EXPORT DATA', 'UPDATE', and 'DELETE' buttons. The main area lists three documents with their details:

- Document 1:**

```
_id: ObjectId("688fffae39c97ecfcd239c796")
pais : "Afghanistan"
uso diario agua per capita(litros) : "1,76"
consumo anual de agua(m3) : "20,288,000,000"
poblacion : "10,130,327"
año : 2000
```
- Document 2:**

```
_id: ObjectId("688fffae39c97ecfcd239c796")
pais : "Albania"
uso diario agua per capita(litros) : "1,177"
consumo anual de agua(m3) : "1,311,000,000"
poblacion : "3,650,889"
año : 2000
```
- Document 3:**

```
_id: ObjectId("688fffae39c97ecfcd239c797")
pais : "Algeria"
uso diario agua per capita(litros) : 665
consumo anual de agua(m3) : "9,970,000,000"
poblacion : "40,850,721"
año : 2010
```

# MongoDB Compass

The screenshot shows the MongoDB Compass interface for a database named 'admin' and a collection named 'netflix'. The 'Documents' tab is selected, displaying two movie documents. The top document is:

```
_id: ObjectId('688ffffef9c97ecfcfd239c84b')
Category : "Movie"
Title : "Adam Devine: Best. Time. of. Our. Lives."
Director : "Jay Karas"
Cast : "Adam Devine"
Country : "United States"
Release_Date : "2019-06-18 00:00:00,000"
Rating : "TV-MA"
Duration : "59 min"
Type : "Stand-Up Comedy"
Description : "Frenetic comic Adam Devine talks teen awkwardness, celebrity encounter..."
```

The bottom document is:

```
_id: ObjectId('688ffffef9c97ecfcfd239c84c')
Category : "Movie"
Title : "ADAM SANDLER 100% FRESH"
Director : "Steve Brill"
Cast : "Adam Sandler"
Country : "United States"
Release_Date : "2018-10-23 00:00:00,000"
Rating : "TV-MA"
Duration : "74 min"
Type : "Stand-Up Comedy"
```

The interface includes a search bar, filter, and sorting options at the top, and a toolbar with 'ADD DATA', 'EXPORT DATA', 'UPDATE', and 'DELETE' buttons at the bottom.



# Estructura CouchDB



En lugar de Redis, se usó CouchDB como segundo modelo NoSQL, donde se adjuntaron directamente archivos JSON.

# CouchDB

```
1
2   "_id": "40738cf5ce91b47bed85a28be400a0f2",
3   "_rev": "3-25eec6e3e4086768c589129bbdc25fad",
4   "_attachments": {
5     "desnutricion_mundo.json": {
6       "content_type": "application/json",
7       "revpos": 2,
8       "digest": "md5-Dm797AHmGof9DbVaUNWsjw==",
9       "length": 22535,
10      "stub": true
11    }
12  }
```

```
← → ⌂ ① 127.0.0.1:5984/proyect/40738cf5ce91b47bed85a28be400a0f2/desnutricion_mundo.json
Impresión con formato estilístico □
{
  "_id": {
    "$oid": "68901a877960956274df0bc4"
  },
  "pais": "India",
  "personas desnutridas": 194400000,
  "porcentaje de la poblacion": "1,41",
  "poblacion(2018)": 1374659064
,
  "_id": {
    "$oid": "68901a877960956274df0bc5"
  },
  "pais": "China",
  "personas desnutridas": 121400000,
  "porcentaje de la poblacion": "0,86",
  "poblacion(2018)": 1419008956
,
  "_id": {
    "$oid": "68901a877960956274df0bc6"
  },
  "pais": "Pakistan",
  "personas desnutridas": 40000000,
  "porcentaje de la poblacion": "1,76",
  "poblacion(2018)": 226928892
,
  "_id": {
    "$oid": "68901a877960956274df0bc7"
  },
  "pais": "Nigeria",
  "personas desnutridas": 25600000,
  "porcentaje de la poblacion": "1,25",
  "poblacion(2018)": 204938755
,
  "_id": {
    "$oid": "68901a877960956274df0bc8"
  }
```

# CouchDB

```
1
2   "_id": "40738cf5ce91b47bed85a28be4007734",
3   "_rev": "3-34dc4d208abf27d0223dfd37e5533c5a",
4   "_attachments": {
5     "noticias_mundo.json": {
6       "content_type": "application/json",
7       "revpos": 2,
8       "digest": "md5-6kg+Ux++1zRzS9hbETwYIQ==",
9       "length": 80560616,
10      "stub": true
11    }
12  }
13 }
```

```
← → C ① 127.0.0.1:5984/proyect/40738cf5ce91b47bed85a28be4007734/noticias_mundo.json
Impresión con formato estilístico □

[{
  "_id": {
    "$oid": "68901aff7960956274df0c3e"
  },
  "headline": "How Ted Cruz And Marco Rubio Are Battling For The Future Of GOP Foreign Policy",
  "category": "POLITICS",
  "date": "2015-12-28 00:00:00,000"
},
{
  "_id": {
    "$oid": "68901aff7960956274df0c3f"
  },
  "headline": "Major Issues Remain To Be Resolved In Iran Talks: U.S. Official",
  "category": "POLITICS",
  "date": "2015-07-12 00:00:00,000"
},
{
  "_id": {
    "$oid": "68901aff7960956274df0c40"
  },
  "headline": "U.S. Military Drills Stoke Politics Of Suspicion In Texas",
  "category": "POLITICS",
  "date": "2015-07-12 00:00:00,000"
},
{
  "_id": {
    "$oid": "68901aff7960956274df0c41"
  },
  "headline": "Mitch McConnell Thinks Congress Will Block Obama's Efforts To Engage With Cuba",
  "category": "POLITICS",
  "date": "2015-07-12 00:00:00,000"
},
{
  "_id": {
    "$oid": "68901aff7960956274df0c42"
  },
  "headline": "Rep. Todd Young Enters Competitive Indiana Senate Race",
  "category": "POLITICS",
  "date": "2015-07-12 00:00:00,000"
}]
```

# Importar desde CSV a MySQL

## NoSQL → SQL

```
import mysql.connector
# Conectar a MySQL
conn = mysql.connector.connect(
    host='localhost',
    user='root',
    password='1234',
    database='repositorio_final2'
)
cursor = conn.cursor()

# Crear tabla si no existe
cursor.execute('''
CREATE TABLE IF NOT EXISTS datos_completos (
    id INT, fecha VARCHAR(50), horas_sueno DOUBLE, minutos_meditacion DOUBLE,
    minutos_ejercicio DOUBLE, tipo_desayuno VARCHAR(100), journaling VARCHAR(10),
    puntaje_productividad DOUBLE, estado_animo VARCHAR(100), notas VARCHAR(255),
    fecha_transaccion VARCHAR(50), edad_cliente DOUBLE, ubicacion_cliente VARCHAR(100),
    cantidad DOUBLE, precio_unitario DOUBLE
);
''')

# Insertar primeros 10 registros
for _, fila in df_clean.head(10).iterrows():
    cursor.execute('''
        INSERT INTO datos_completos VALUES (%s, %s, %s)
    ''', tuple(fila[col] if pd.notna(fila[col]) else None for col in [
        'id', 'fecha', 'horas_sueno', 'minutos_meditacion', 'minutos_ejercicio',
        'tipo_desayuno', 'journaling', 'puntaje_productividad', 'estado_animo',
        'notas', 'fecha_transaccion', 'edad_cliente', 'ubicacion_cliente',
        'cantidad', 'precio_unitario'
    ]))

# Confirmar y cerrar
conn.commit()
conn.close()

print("Se crearon la tabla y se insertaron 10 registros correctamente.")
```



83 • select \* from datos\_completos;

Result Grid					
	id	fecha	horas_sueno	minutos_meditacion	minutos_ejercicio
10	otros	6.9431818181818e15	1.3295454545454544e16	3.102272727272727e	
20	otros	6.9431818181818e15	1.3295454545454544e16	3.102272727272727e	
30	otros	6.9431818181818e15	1.3295454545454544e16	3.102272727272727e	
40	otros	6.9431818181818e15	1.3295454545454544e16	3.102272727272727e	
50	otros	6.9431818181818e15	1.3295454545454544e16	3.102272727272727e	
60	otros	6.9431818181818e15	1.3295454545454544e16	3.102272727272727e	
70	otros	6.9431818181818e15	1.3295454545454544e16	3.102272727272727e	
80	otros	6.9431818181818e15	1.3295454545454544e16	3.102272727272727e	
90	otros	6.9431818181818e15	1.3295454545454544e16	3.102272727272727e	

# Exportar desde MySQL a MongoDB

## SQL → NoSQL

```
import mysql.connector
import pandas as pd
from pymongo import MongoClient

# Conectar a MySQL
conn = mysql.connector.connect(
    host='localhost',
    user='root',
    password='1234',
    database='repositorio_final2'
)

df_clean = pd.read_sql("SELECT * FROM datos_completos LIMIT 10", conn)

# Conectar a MongoDB
cliente_mongo = MongoClient("mongodb://localhost:27017/")
db = cliente_mongo["proyecto_datos"]
colección = db["datos_integrados"]

# Exportar los datos
colección.insert_many(df_clean.to_dict(orient="records"))

print("Exportado de MySQL a MongoDB (10 registros)")
```

The screenshot shows the MongoDB Compass interface. The left sidebar displays the connection tree under 'local': 'admin', 'config', 'local' (selected), 'startup.log', and 'proyecto\_datos' (selected). The main area shows the 'datos\_integrados' collection with 10 documents. A document is selected, showing its details:

```
_id: ObjectId('6891525ecdbc8ff89383503f')
id : 10
fecha : "otros"
horas_sueno : 6943181818181818
minutos_meditacion : 13295454545454545
minutos_ejercicio : 31022727272727273
tipo_desayuno : "otros"
journaling : 0
puntaje_productividad : 5738636363636363
estado_animo : "Desconocido"
notas : "otros"
fecha_transaccion : "otros"
edad_cliente : 48665666666666667
ubicacion_cliente : "otros"
cantidad : 437233333333334
precio_unitario : 12305791333333336
monto_total : 39000102333333336
metodo_pago : "otros"
velocidad_envio : "otros"
historial_pedidos : 19355666666666668
descuento_aplicado : 10
porcentaje_descuento : 12021433333333333
costo_envio : 1349424
dias_entrega : 15046333333333333
pais : "otros"
```

Below the document details, there is a link 'Show 14 more fields'. At the bottom of the Compass interface, there is a footer bar with buttons for 'ADD DATA', 'EXPORT DATA', 'UPDATE', and 'DELETE'.

# Importación a SQL Server



```
base_final.sql - DES...ia Guanoluisa (56)  → X
/*!48101 SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='NO_AUTO_VALUE_ON_ZERO';
/*!48111 SET @OLD_SQL_NOTES=@@SQL_NOTES, SQL_NOTES=0 */;

-- Table structure for table `datos_completos`

DROP TABLE IF EXISTS `datos_completos`;
/*!48101 SET @saved_cs_client      = @@character_set_client */;
/*!50539 SET character_set_client = utf8mb4 */;
CREATE TABLE `datos_completos` (
  `id` int DEFAULT NULL,
  `fecha` varchar(50) DEFAULT NULL,
  `horas_sueno` double DEFAULT NULL,
  `minutos_meditacion` double DEFAULT NULL,
  `minutos_ejercicio` double DEFAULT NULL,
  `tipo_desayuno` varchar(100) DEFAULT NULL,
  `journaling` varchar(10) DEFAULT NULL,
  `notas` text
);

Completion time: 2026-08-04T13:26:24.4134458-06:00
```

# Archivos

 categoria_noticias.csv		2/8/2025 12:24	Archivo de valores...	51.337 KB
 consumo_agua_mundo.csv		1/8/2025 23:53	Archivo de valores...	9 KB
 datos_empleados.csv		2/8/2025 12:09	Archivo de valores...	760 KB
 datos_netflix.csv		2/8/2025 11:47	Archivo de valores...	1.916 KB
 Parte1.ipynb		2/8/2025 11:27	Archivo de origen ...	20 KB
 personas_desnutridas_mundo.csv		1/8/2025 23:02	Archivo de valores...	4 KB
 productos_falsificados_comercio_eletronico.csv		2/8/2025 12:16	Archivo de valores...	258 KB
 reservas_petroleo.csv		1/8/2025 23:23	Archivo de valores...	4 KB
 rutina_matutina.csv		2/8/2025 12:00	Archivo de valores...	129 KB
 siniestros_transito.csv		1/8/2025 22:20	Archivo de valores...	1.730 KB
 vehiculos_matriculados_2023.csv		1/8/2025 22:28	Archivo de valores...	174.808 KB



# Union y limpieza del Dataframe global

```
import psycopg2
import pandas as pd

# Datos de conexión
conn = psycopg2.connect(
    host="localhost",          # o IP del servidor PostgreSQL
    database="reservas_petroleo",
    user="postgres",
    password="1234",
    port="5432"                # puerto por defecto PostgreSQL
)

# Ejecutar consulta y cargar datos en DataFrame
query = "SELECT * FROM reservas_petroleo;"
df3 = pd.read_sql(query, conn)

print(df3)

conn.close()
```

## Librerias

```
!pip install mysql-connector-python
!pip install sqlalchemy
!pip install pymysql
!pip install pandas
```

```
import pandas as pd
import mysql.connector

# Configura tu conexión a MySQL (ajusta host, usuario, contraseña y base de datos)
conn = mysql.connector.connect(
    host='localhost',          # o IP de servidor MySQL
    user='root',
    password='1234',
    database='habitos_diarios'
)

# Consulta una tabla y carga a pandas DataFrame
query = "SELECT * FROM rutina_matutina;"
df1 = pd.read_sql(query, conn)

print(df1)

# Cierra la conexión
conn.close()
```

```
import pandas as pd

json_file = r'C:\Users\Flia Guanoluisa\OneDrive\Documentos\Proyecto_analisis\admin.consumo_agua.json'

df6 = pd.read_json(json_file)

print(df6)
```



# Union y limpieza del Dataframe global

```
import pandas as pd
# Unir todos en uno solo
df_global = pd.concat([df1, df2, df3, df4, df5, df6, df7, df8, df9, df10], ignore_index=True)
# Ver las primeras filas
print(df_global)
```

```
# Paso 3: Eliminar columnas completamente vacías
df_clean = df_global.copy()
df_clean = df_clean.dropna(axis=1, how='all')

# Paso 4: Eliminar filas completamente vacías
df_clean = df_clean.dropna(axis=0, how='all')

# Paso 5: Eliminar duplicados (sólo si no hay columnas tipo dict)
df_clean = df_clean[~df_clean.applymap(type).eq(dict).any(axis=1)]
df_clean = df_clean.drop_duplicates()

# Paso 6: Rellenar valores nulos solo si la columna existe
if 'journaling' in df_clean.columns:
    df_clean['journaling'] = df_clean['journaling'].fillna(0)

if 'estado_animo' in df_clean.columns:
    df_clean['estado_animo'] = df_clean['estado_animo'].fillna('Desconocido')

# Paso 7: Convertir fechas si hay columna 'fecha'
if 'fecha' in df_clean.columns:
    df_clean['fecha'] = pd.to_datetime(df_clean['fecha'], errors='coerce')
```

```
# Paso 8: Convertir columnas clave a tipo adecuado (si aplican)
# Ejemplo: fechas
if 'fecha' in df_clean.columns:
    df_clean['fecha'] = pd.to_datetime(df_clean['fecha'], errors='coerce')

if 'release_date' in df_clean.columns:
    df_clean['release_date'] = pd.to_datetime(df_clean['release_date'], errors='coerce')

# Paso 9: Analizar columnas con muchos nulos
nulls_percent = df_clean.isnull().mean() * 100
cols_to_drop = nulls_percent[nulls_percent > 90].index.tolist()

# Opcional: Rellena columnas con más del 90% de valores nulos
for col in df_clean.columns:
    if df_clean[col].isnull().mean() > 0.90: # solo columnas con >90% nulos
        if pd.api.types.is_numeric_dtype(df_clean[col]):
            # Si es numérica: llenar con el promedio (ignorando NaN)
            promedio = df_clean[col].mean()
            df_clean[col] = df_clean[col].fillna(promedio)
        else:
            # Si es tipo string/objeto: llenar con 'otros'
            df_clean[col] = df_clean[col].fillna('otros')

print("Shape final:", df_clean.shape)
print(df_clean.info())
```

# Visualización de datos y análisis de sentimientos

```
import pandas as pd
import matplotlib.pyplot as plt
import json
from textblob import TextBlob
```

```
# 1. Cargar el archivo JSON
file_path = "admin.consumo_agua.json"
with open(file_path, "r", encoding="utf-8") as f:
    data = json.load(f)

# 2. Crear DataFrame y limpiar columnas
df = pd.DataFrame(data)
df["uso diario agua per capita(litros)"] = pd.to_numeric(
    df["uso diario agua per capita(litros)"], errors="coerce")
df["consumo anuel de agua(m3)"] = (
    df["consumo anuel de agua(m3)"]
    .str.replace(",","")
    .astype(float))
df["poblacion"] = df["poblacion"].str.replace(",","").astype(float)
```

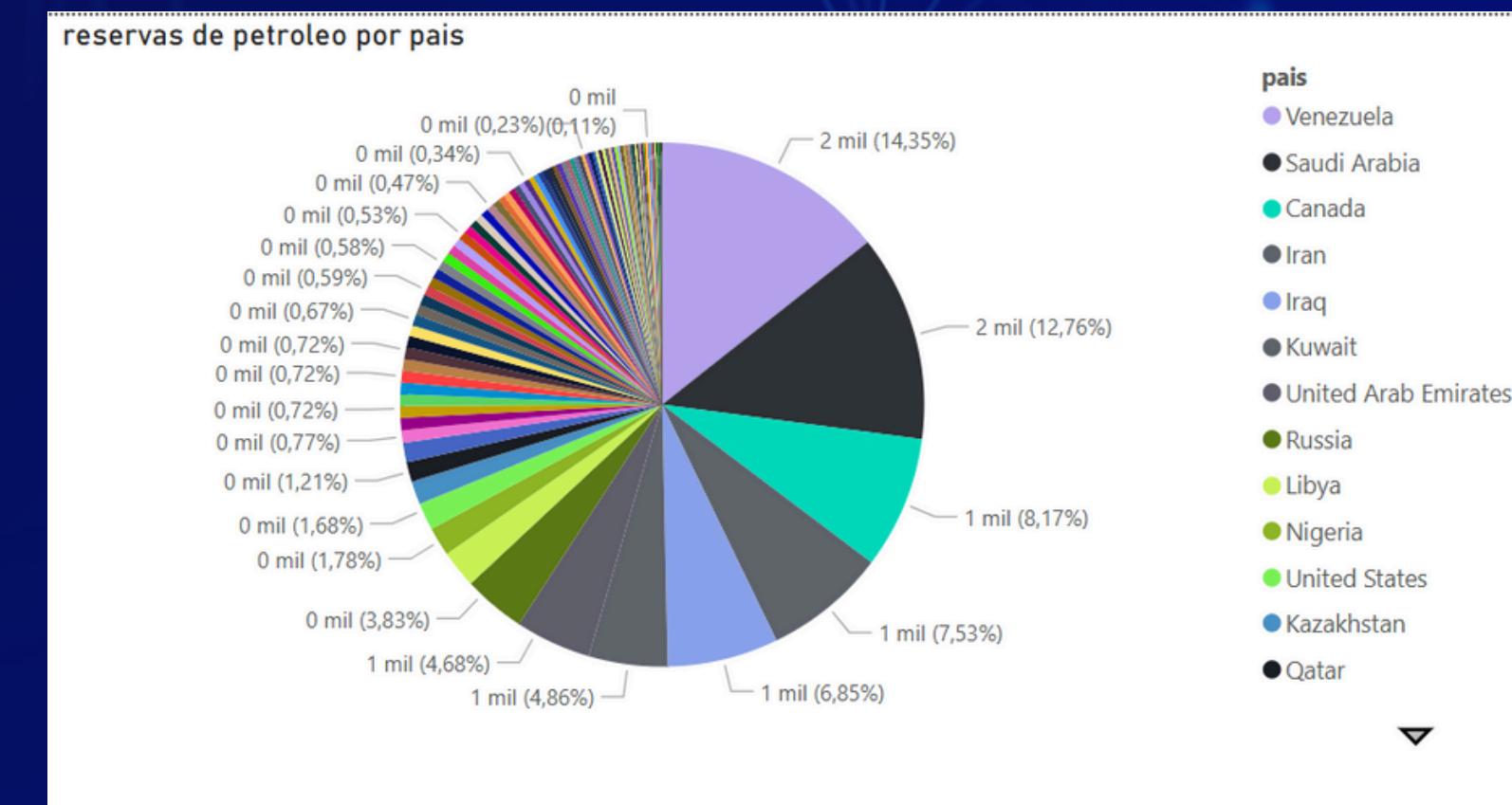
**La librería TextBlob es una biblioteca de procesamiento de lenguaje natural (NLP) en Python que sirve principalmente para analizar y manipular texto, de forma simple e intuitiva. Es muy usada para tareas básicas de análisis de sentimientos, corrección gramatical y traducción.**

```
# 4. Análisis de sentimiento de las opiniones
df["sentimiento"] = df["opinion"].apply(lambda text: TextBlob(text).sentiment.polarity)

# 5. Mostrar resumen de sentimientos
from IPython.display import display
display(df[["pais", "uso diario agua per capita(litros)", "opinion", "sentimiento"]])

# 6. Visualización: uso vs sentimiento
plt.figure(figsize=(10, 6))
sizes = (df["poblacion"] / df["poblacion"].max()) * 200 # tamaño relativo por población
plt.scatter(
    df["uso diario agua per capita(litros)"],
    df["sentimiento"],
    s=sizes,
    alpha=0.6
)
plt.title("Uso diario de agua per cápita vs Polaridad de Sentimiento")
plt.xlabel("Uso diario per cápita (litros)")
plt.ylabel("Polaridad del Sentimiento")
plt.grid(True)
plt.tight_layout()
plt.show()
```

# Visualización de datos con POWERBI



Confidencialidad Compartir

Visualizaciones >

Compilar visual

Valores

Aregar campos de datos a...

Obtener detalles

Entre varios informes

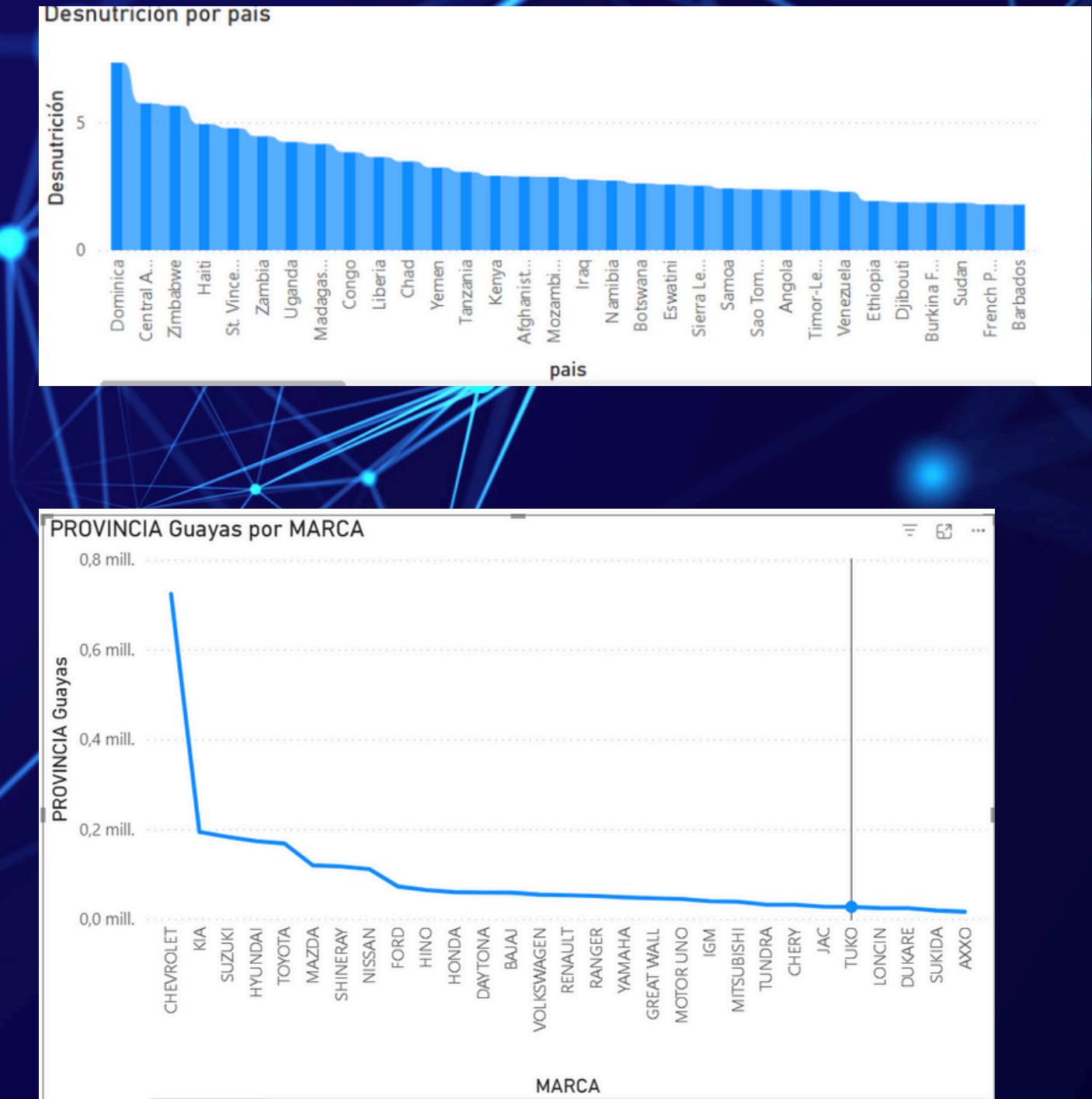
Mantener todos los filtros

Agregue los campos de ob...



# CASOS DE ESTUDIO GENERADOS

- Dominica vs. Eswatini:  
Contraste de desnutrición  
extrema
- Tendencias históricas de  
reservas de petróleo de  
Venezuela
- Preferencias de marca según  
región
- Impacto de la hora del día  
en la clase de siniestros





# Conclusiones del análisis



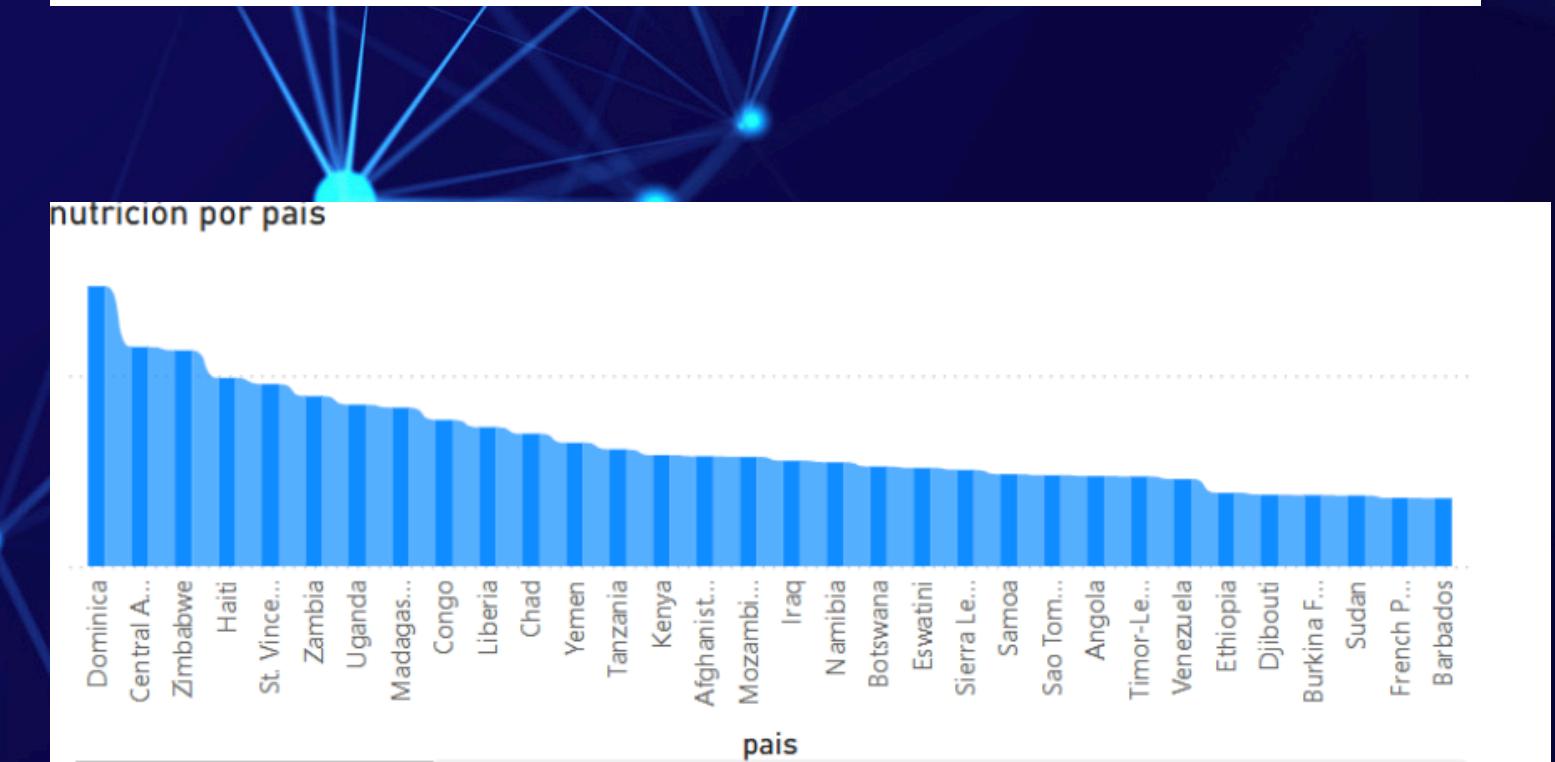
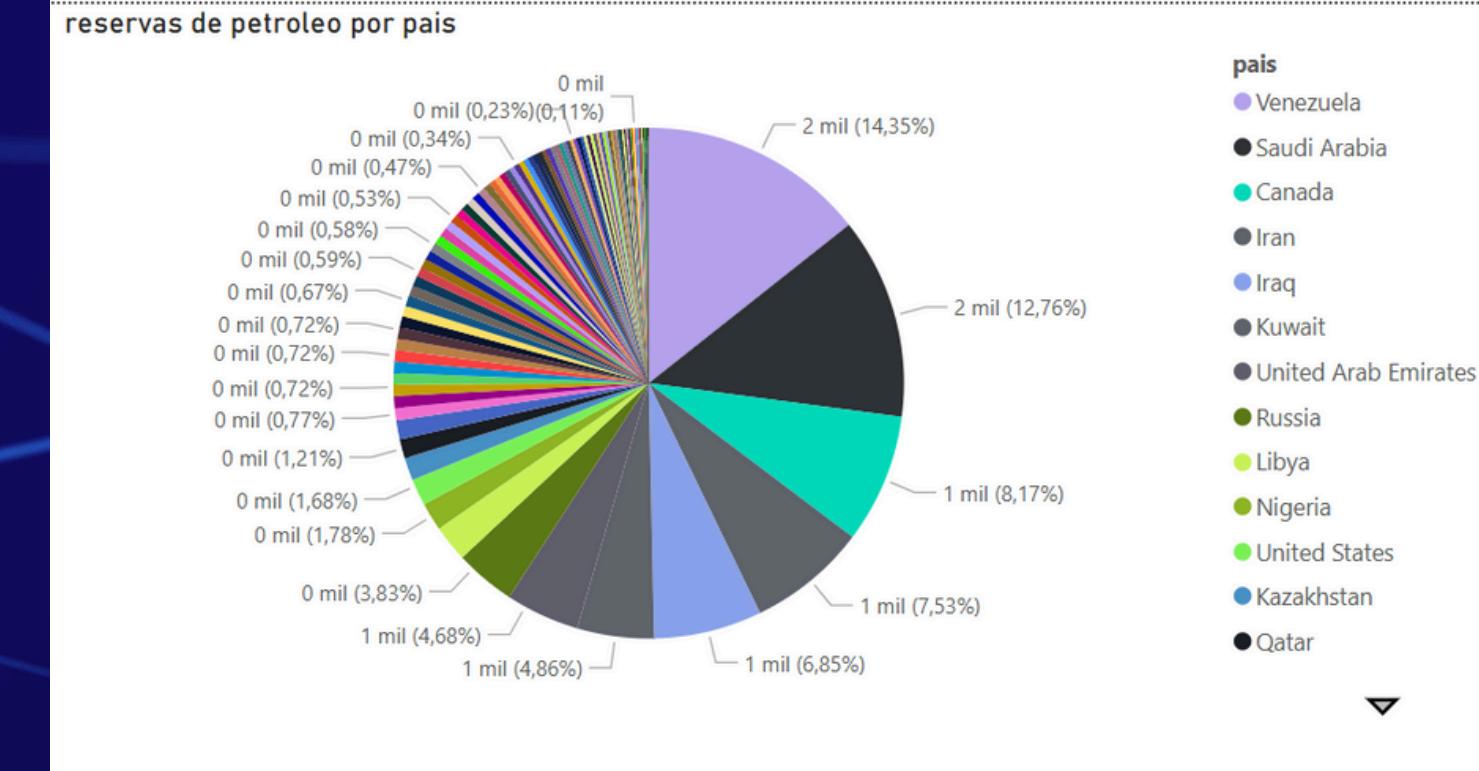
## Desnutrición en el mundo

Existe una concentración de los índices más elevados en varios países de África y el Caribe, con Dominica encabezando la lista.

Las diferencias regionales sugieren que factores como inestabilidad política, pobreza extrema y falta de acceso a servicios sanitarios son determinantes clave.

## Reservas de petróleo (2016)

Venezuela y Arabia Saudita concentran casi el 35 % de las reservas mundiales, lo que refuerza su peso geopolítico y su influencia en los mercados energéticos.





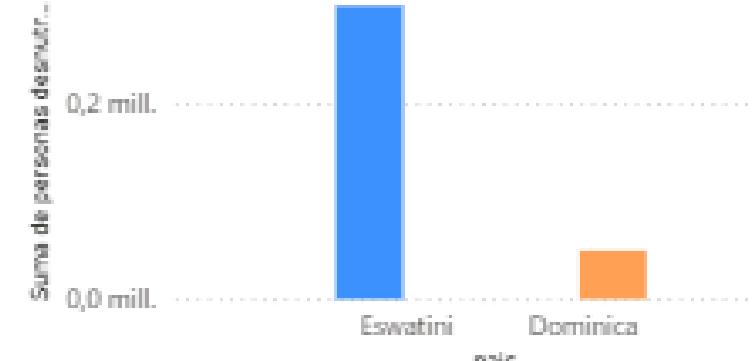
**Thynk Unlimited**

# Dashboards

## Análisis de Datos - Casos definidos

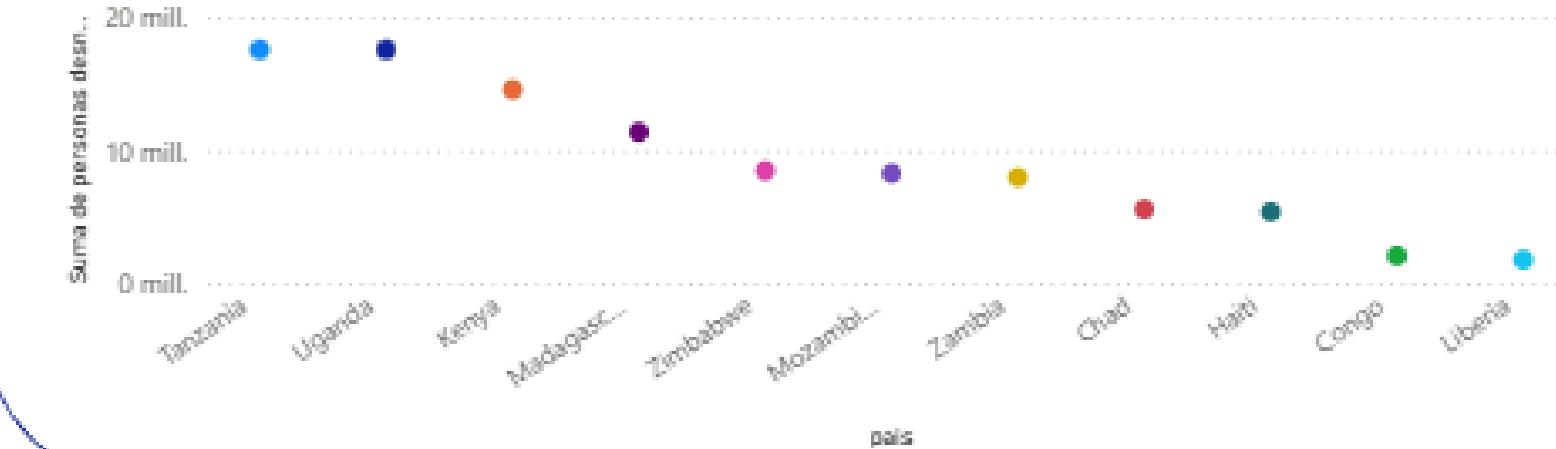
**Dominica vs. Eswatini: Contraste de desnutrición extrema**

país: ● Eswatini ● Dominica

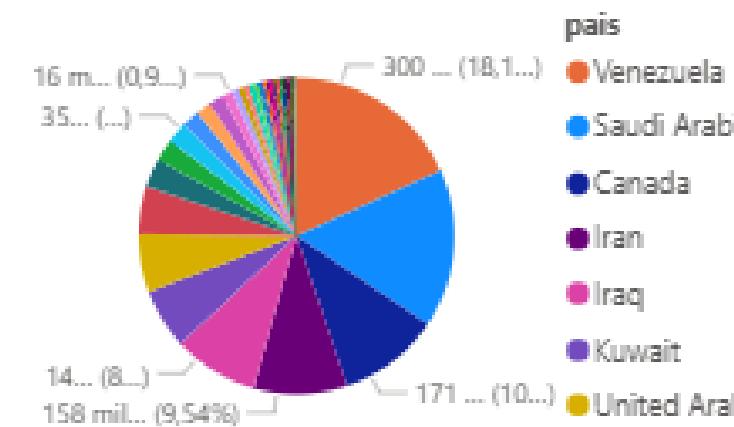


**África Subsahariana: Foco regional en desnutrición**

país: ● Tanzania ● Uganda ● Kenya ● Madagascar ● Zimbabwe ● Mozambique ● Zambia ● Chad ● Haiti ● Congo ● Liberia

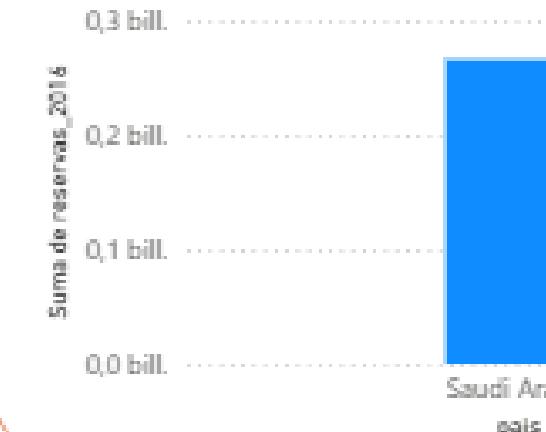


**Distribución de reservas mundiales de petróleo en Venezuela**



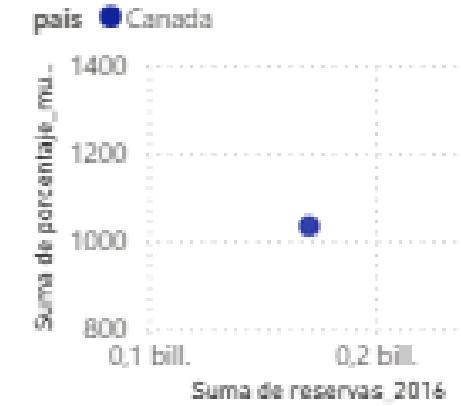
**Dependencia mundial del petróleo Saudita**

país: ● Saudi Arabia



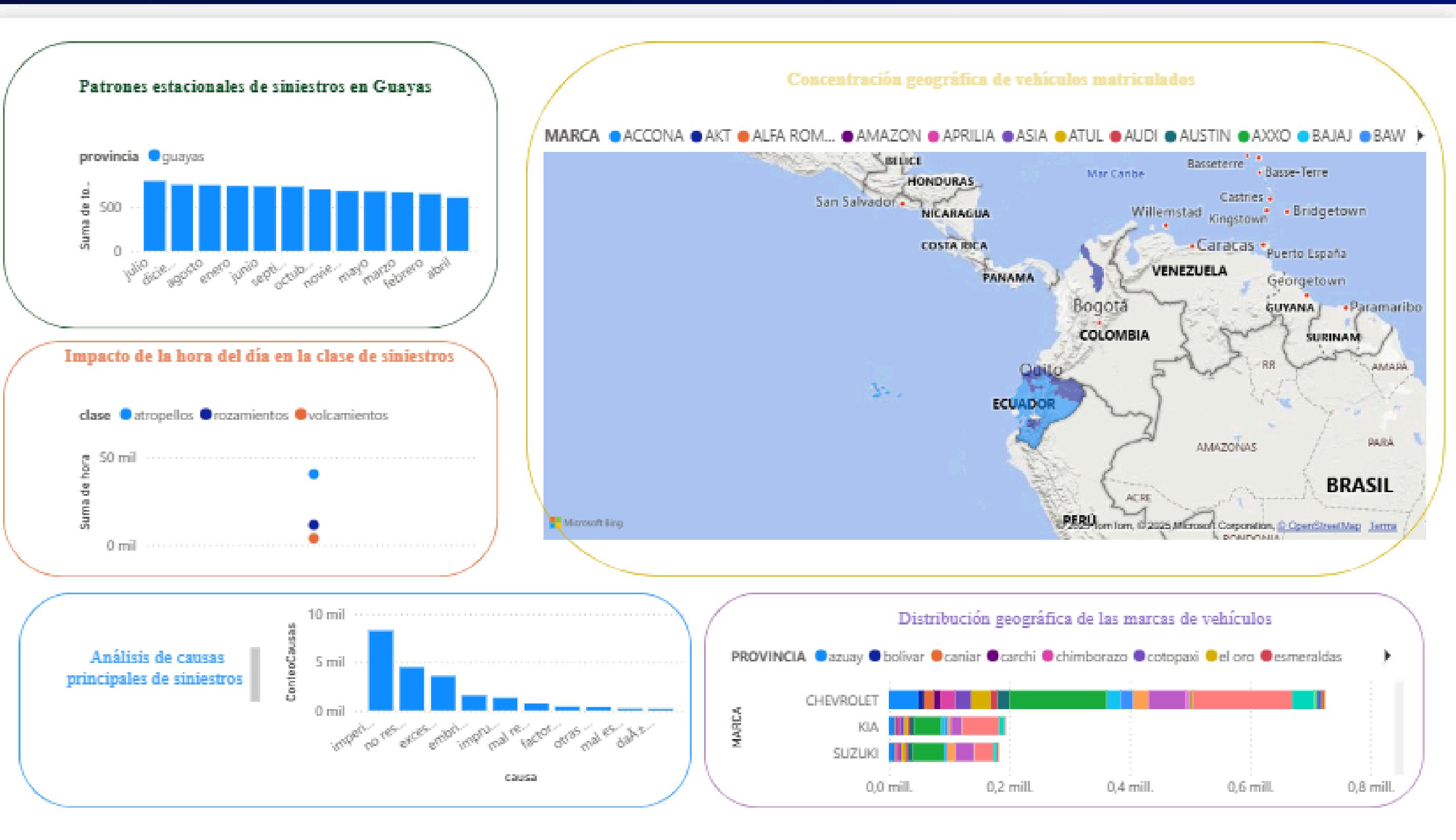
**Comparativa de reservas absolutas vs. porcentaje mundial**

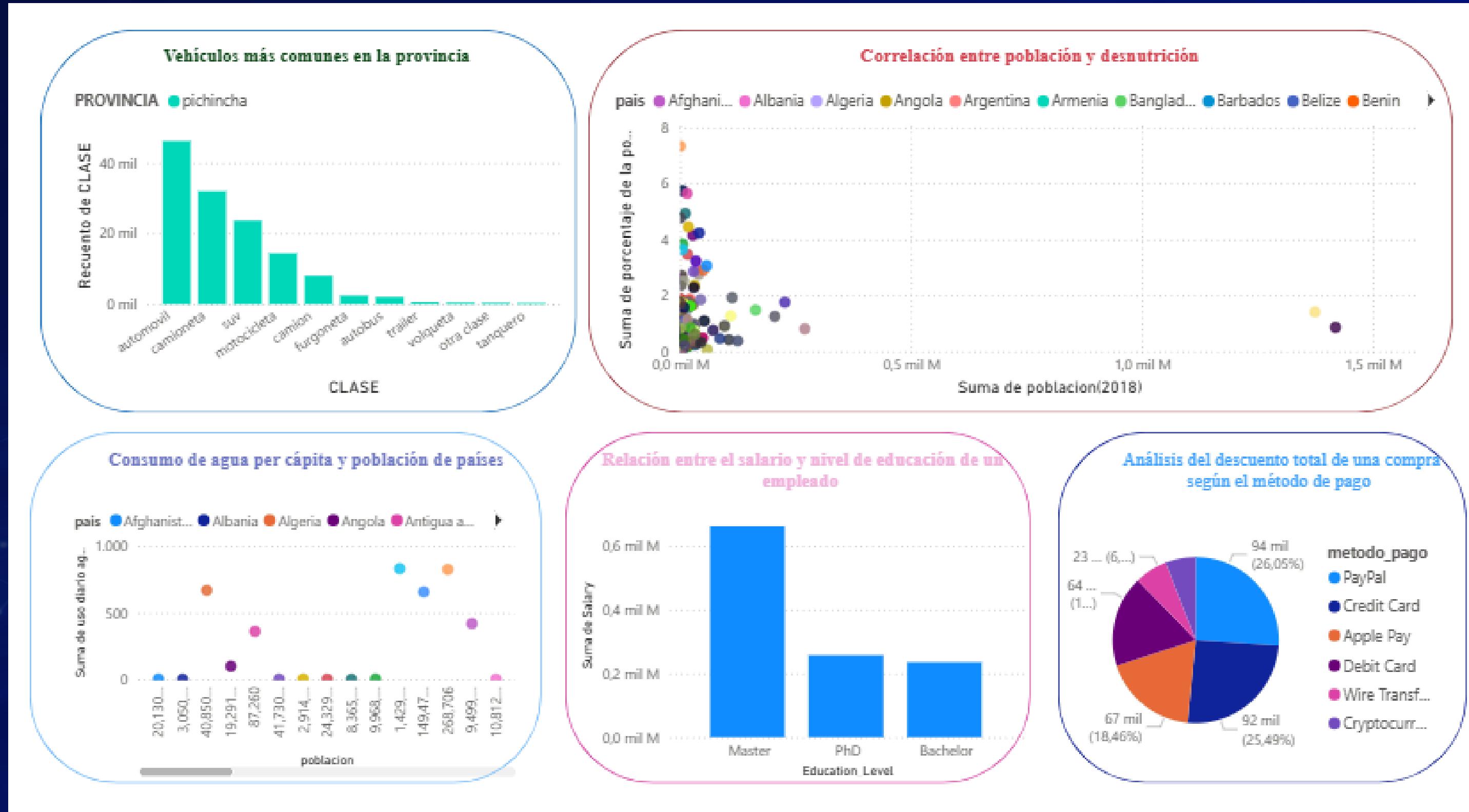
país: ● Canada





# Thynk Unlimited







**Thynk Unlimited**



**Muchas Gracias**