Proyecto de Análisis de Datos

Nombre:

- Camila Bueno
- Roger Grefa
- Diego Montaluisa

I. DEFINICIÓN DEL CASO DE ESTUDIO

Este proyecto se basa en la realización de distintos casos de estudio, que consisten en analizar diferentes temas para comprender mejor su situación y su importancia. En este trabajo se estudian cuatro temas: Netflix, la política en Ecuador, los Juegos Olímpicos y los accidentes vehiculares, haciendo referencia a la temática del tráfico vehicular. Para ello se utilizan archivos CSV obtenidos de la plataforma Kaggle y se accede a páginas de datos abiertos del Ecuador. Con esta información se busca descubrir tendencias o patrones y encontrar posibles soluciones para cada caso de estudio.

II. OBJETIVO GENERAL Y ESPECÍFICOS

Objetivo general:

Analizar datos de las distintas temáticas, aplicando las técnicas de manejo de datos desde la recolección y limpieza hasta la transformación y visualización aprendidas para descubrir patrones y generar conclusiones.

Objetivos específicos:

- Extracción y limpieza de datos mediante Python y paginas como Kaglee y datos abiertos.
- Integrar los datos en dos bases de datos relacional (MySQL, SQLite, SQL Server) y NoSQL (MongoDB).
- Crear dashboards en Power BI que permitan visualizar tendencias.

III. DESCRIPCIÓN DEL EQUIPO DE TRABAJO Y ACTIVIDADES REALIZADAS POR CADA UNO.

Camila Bueno: Encargada de la búsqueda de datos sobre las temáticas, la exportación a MySQL, el análisis de sentimientos y la elaboración de visualizaciones en Power BI relacionadas con la temática de tránsito.
Roger Grefa: Responsable de la exportación de datos a SQLite sobre la política en Ecuador, la creación de visualizaciones correspondientes a su temática y la elaboración de la presentación en PowerPoint.
Diego Montaluisa: Encargado de la conexión de la temática de Juegos Olímpicos a MongoDB, la integración de toda la data en SQL Server y la creación de visualizaciones en Power BI para sus temáticas.

IV. CRONOGRAMA DE ACTIVIDADES

Actividad	Fecha Inicio	Fecha Fin
Extracción de datos	25/07/2025	26/07/2025
Limpieza de datos	27/07/2025	29/07/2025
Análisis de datos	30/07/2025	1/01/2025
Visualización de datos	02/01/2025	02/08/2025
Redacción del informe	03/02/2025	04/02/2025

V. RECURSO Y HERRAMIENTAS UTILIZADAS

Para el desarrollo del proyecto se utilizaron las siguientes herramientas:

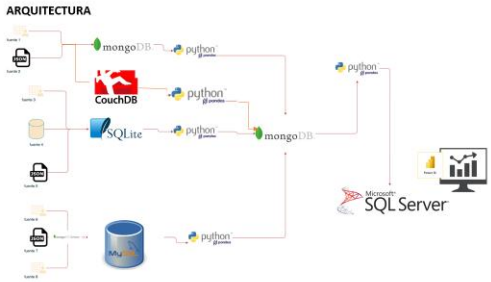
- **Python:** Para la extracción, limpieza y c conexión entre las bases de datos y transformación de archivos.
- **Power BI:** Para la creación de dashboards interactivos.
- **MySQL, SQLite, SQL Server:** Para el almacenamiento de los csv y json , datos relacionales.
- **MongoDB Compass, Mongo DB Atlas :** Para el almacenamiento de csv datos no rlacionales.
- **Librerías de Python**: pandas, pymongo, sqlalchemy.

VI. ARQUITECTURA DE LA SOLUCIÓN.

La arquitectura del proyecto se compone de las siguientes etapas:

1. **Extracción de datos:** Se obtuvieron datos de Kaggle,,y páginas con datos abiertos sobre nuestro país que proporciona el estado como datosabiertos.com o INEC.
2. **Limpieza:** Se utilizó Python y mediante código se realizo la limpieza de los csv y json.

3. **Almacenamiento:** Los datos se almacenaron en SQLite, MySQL, MongoDB y SQL Server.
4. **Análisis:** Se aplicó análisis de sentimientos al dataframe.
5. **Visualización:** Se crearon dashboards en Power BI.

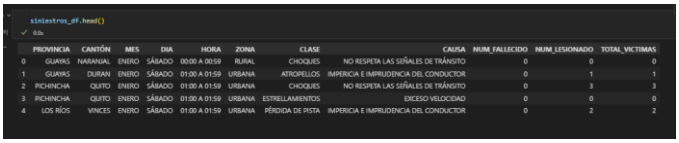


VII. Extracción de datos.

Para la extracción de datos, las plataformas que utilizamos ya ofrecían los archivos en formato CSV, lo que facilitó el proceso. Usamos Python junto con la librería **Pandas** para trabajar con estos archivos. Primero, leímos el CSV como un DataFrame y luego realizamos la limpieza, eliminando duplicados, valores nulos y filas vacías, para dejar la información lista para el análisis.

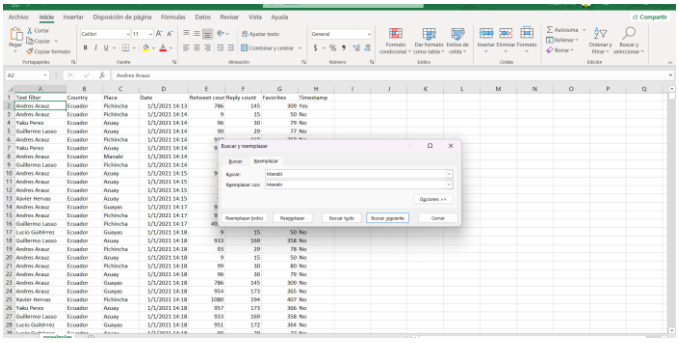
```
import pandas as pd

# Cargar el archivo CSV
siniestros_df = pd.read_csv('2022_SINIESTROS_TRÁNSITO_BDD.csv', sep=';')
# Ver las primeras filas
siniestros_df.head()

# Eliminar filas duplicadas
siniestros_df.drop_duplicates(inplace=True)
# Eliminar filas completamente vacías
siniestros_df.dropna(how="all", inplace=True)
# Guardar el archivo limpio
siniestros_df.to_csv("siniestros_limpios.csv", index=False)
# Mostrar primeras filas
print("Archivo limpio guardado:")
print(siniestros_df.head())
```

Tras cargar los datos en el DataFrame utilizando la función read_csv(), se realizó la lectura de las primeras cinco filas con df.head(). Esto permitió comprobar que los datos se habían cargado correctamente y que el DataFrame no estuviera vacío.

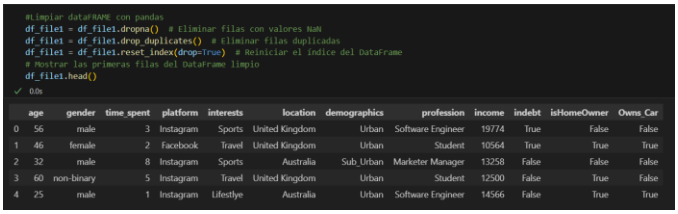


	PROVINCIA	CANTÓN	MES	DIA	HORA	ZONA	CLASE	CAUSA	NUM_FALLECIDO	NUM_LESIONADO	TOTAL_VICTIMAS
0	GUAYAS	NARANJAL	ENERO	SÁBADO	00:00 A 00:59	RURAL	CHOQUES	NO RESPETA LAS SEÑALES DE TRÁNSITO	0	0	0
1	GUAYAS	DURAN	ENERO	SÁBADO	01:00 A 01:59	URBANA	ATROPELLOS	IMPERICIA E IMPRUDENCIA DEL CONDUCTOR	0	1	1
2	PICHINCHA	QUITO	ENERO	SÁBADO	01:00 A 01:59	URBANA	CHOQUES	NO RESPETA LAS SEÑALES DE TRÁNSITO	0	3	3
3	PICHINCHA	QUITO	ENERO	SÁBADO	01:00 A 01:59	URBANA	ESTRELLAMIENTOS	EXCESO VELOCIDAD	0	0	0
4	LOS RÍOS	VINCES	ENERO	SÁBADO	01:00 A 01:59	URBANA	PÉRDIDA DE PISTA	IMPERICIA E IMPRUDENCIA DEL CONDUCTOR	0	2	2

Esa fue una de las formas en que realizamos la limpieza, pero también se puede optar por otro método: limpiar directamente desde Excel, revisando y corrigiendo tildes o caracteres que puedan generar errores en el procesamiento de los datos.



Por otra parte también se resetearon los índices del Dataframe.



	age	gender	time_spent	platform	interests	location	demographics	profession	income	indebt	isHomeOwner	Owns_Car
0	56	male	3	Instagram	Sports	United Kingdom	Urban	Software Engineer	19774	True	False	False
1	46	female	2	Facebook	Travel	United Kingdom	Urban	Student	10564	True	True	True
2	32	male	8	Instagram	Sports	Australia	Sub_Urban	Marketer Manager	13258	False	False	False
3	60	non-binary	5	Instagram	Travel	United Kingdom	Urban	Student	12500	False	True	False
4	25	male	1	Instagram	Lifestlye	Australia	Urban	Software Engineer	14566	False	True	True

Una vez realizada la limpieza, se verificó con las funciones isnull() y sum() si aún existían valores vacíos en los DataFrames. Esto permitió confirmar si la limpieza fue efectiva o si era necesario rectificar y repetir el proceso.

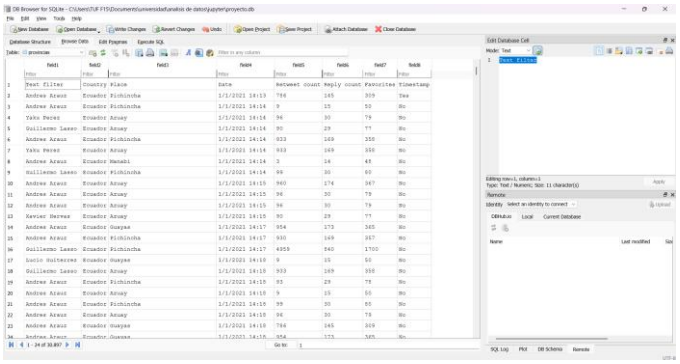
```
siniestros_df.isnull().sum()
```

```
PROVINCIA        0
CANTÓN           0
MES              0
DIA              0
HORA             0
ZONA             0
CLASE            0
CAUSA            0
NUM_FALLECIDO    0
NUM_LESIONADO    0
TOTAL_VICTIMAS   0
dtype: int64
```

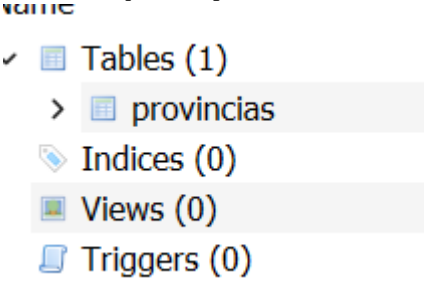
Así comprobamos que no había valores vacíos, y con ello pudimos proceder a la exportación de los archivos a bases de datos relacionales y NoSQL.

SQL LITE

Después de realizar la limpieza del archivo procedemos a exportar el archivo ya limpio a SQL LITE esto lo hacemos directamente desde el programada



Cargamos el archivo ya que no hay ningún error ni caracteres q no sen posibles reconocer desde la interfaz



Y desde la interfaz grafica podemos ver que la tabla ya se encuentra dentro de SQL Lite.
Ademas para comprobar podemos realizar la conexión con esta base desde VS code

Iniciamos con la importación de los recursos necesarios para la conexión

```
import sqlite3
import pandas as pd
```

Procedemos con la conexión

```
# Conexión
conexion = sqlite3.connect("proyecto.db")
```

Verificamos cuantas tablas tenemos en SQLite con el siguiente código e imprimimos las tablas que hayan.

```
consulta = "SELECT name FROM sqlite_master WHERE type='table';"
tablas = pd.read_sql_query(consulta, conexion)
print("Tablas en la base de datos:")
print(tablas)
```

En este caso solo nos muestra una tabla llamada provincia

```
Tablas en la base de datos:
        name
0  provincias
```

Hacemos la conexión con la tabla y asignamos la primera fila como nombres de la columna y borramos el índice anterior.

```
df = pd.read_sql_query("SELECT * FROM provincias", conexion)
df.columns = df.iloc[0]     # Asignar primera fila como nombres de columna
df = df[1:].reset_index(drop=True) #Eliminar la primera fila original y reiniciar índic
```

luego imprimimos los elementos de la tabla y finalizamos la conexión.

```
print("Contenido de la tabla 'provincias':")
print(df.head())

conexion.close()
```

Resultado:

```
Contenido de la tabla 'provincias':
0      Text filter  Country      Place            Date Retweet count  \
0     Andres Arauz  Ecuador  Pichincha  1/1/2021 14:13           786
1     Andres Arauz  Ecuador  Pichincha  1/1/2021 14:14             9
2       Yaku Perez  Ecuador      Azuay  1/1/2021 14:14            96
3  Guillermo Lasso  Ecuador      Azuay  1/1/2021 14:14            90
4     Andres Arauz  Ecuador  Pichincha  1/1/2021 14:14           933

0 Reply count Favorites Timestamp
0         145       309       Yes
1          15        50        No
2          30        79        No
3          29        77        No
4         169       358        No
```

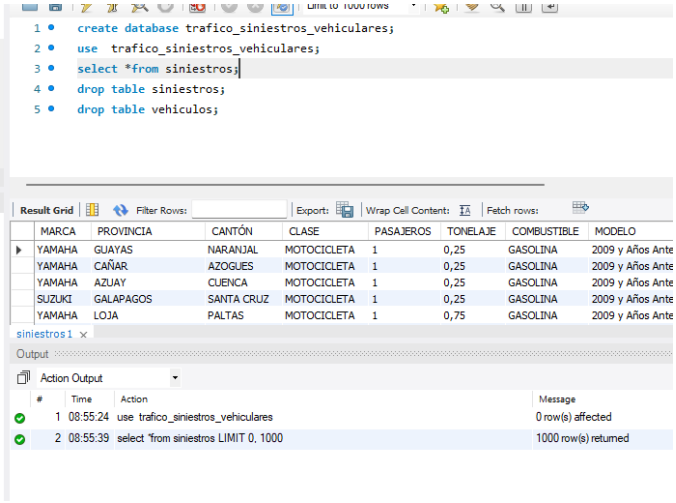
MySQL Wockbench

Para importar los datos limpios a una base de datos relacional en **MySQL Workbench**, se utilizó la librería **SQLAlchemy** en Python, la cual facilita la creación de conexiones con bases de datos. Primero, se cargó el archivo *siniestros_limpio.csv* en un DataFrame usando Pandas. Luego, se definieron los parámetros de conexión (usuario, contraseña, host, puerto y nombre de la base de datos) y se creó el enlace mediante create_engine(). Finalmente, se utilizó el método to_sql() para enviar el contenido del DataFrame a MySQL, creando la tabla *siniestros* de forma automática.

```
import pandas as pd
from sqlalchemy import create_engine
df = pd.read_csv('siniestros_limpio.csv')

# Datos de conexión a MySQL
usuario = 'root'
contrasena = '1234'
host = 'localhost'
puerto = '3306'
basee = 'trafico_siniestros_vehiculares'
# Crear conexión con SQLAlchemy
conexion = create_engine(f'mysql+pymysql://{usuario}:{contrasena}@{host}:{puerto}/{basee}')
# Subir el DataFrame a MySQL
df.to_sql('siniestros', con=conexion, index=False)
print("Datos importados a MySQL.")
```

```
Datos importados a MySQL.
```

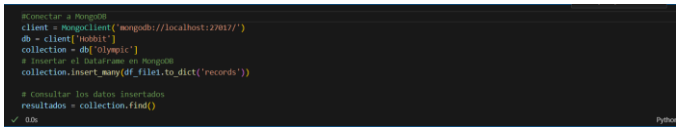
Una vez importado podemos abrir el programa y ejecutar el comando de select *from para comprobar que se exporto de manera correcta el csv.



Con ello comprobamos que si se conecto correctamente y la exportación del csv utilizando Python si se dio.

MongoDB Atlas

Para la temática de los Juegos Olímpicos, se realizó la conexión a **MongoDB** utilizando la librería pymongo. Primero, se creó el cliente con la ruta de conexión mongodb://localhost:27017/ y se seleccionó la base de datos llamada *ProyectoDatos*. Dentro de esta base, se trabajó con la colección *Deportes* El DataFrame con la información previamente procesada se convirtió en una lista de diccionarios mediante el método to_dict('records') y se insertó en la colección con insert_many(). Finalmente, para verificar que los datos fueron cargados correctamente, se utilizó collection.find() para consultar todos los registros almacenados.



Asi desde la interfaz grafica podemos verificar que se cargo de manera correcta.

Procesos de migración entre bases de datos SQL y NoSQL

Migración de MySQL a MongoDB

Para realizar la migración entre estas dos bases se utilizo por código ya los archivos anteriormente limpiados y cargados se los subió a Mongo Atlas De la siguiente manera

De esta forma, primero se estableció la conexión con **MongoClient**, indicando la ruta de conexión al servidor de MongoDB Atlas. Luego, se seleccionó la base de datos predeterminada que había sido creada previamente para el proyecto. A continuación, se subieron los archivos correspondientes a cada temática, en formato CSV o JSON, cargándolos en Python y convirtiéndolos a listas de diccionarios mediante el método to_dict('records'). Finalmente, se insertaron en las colecciones designadas dentro de la base de datos, quedando la información lista para consultas y análisis posteriores.

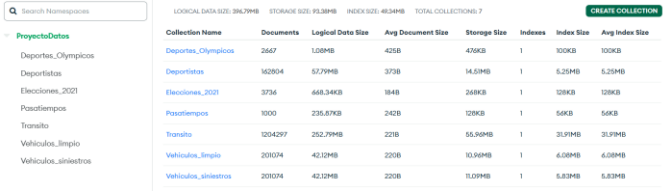
Migracion de SQLite a MongoDB

Para la exportación de esto se realizo de la misma manera por código ya que dado que la data estaba limpia.

Realizamos lo mismo con otros archivos.

Migracion de MongoDB a SQ Server

Así comprobamos que todos los archivos provenientes de las bases de datos relacionales anteriores se encontraban correctamente almacenados en **MongoDB Atlas**, asegurando que la migración y carga de información se realizó de manera exitosa.



Conexion a CouchDB

En este proyecto, se utilizó Python para conectarse a una base de datos NoSQL Couchbase con el objetivo de extraer todos los documentos del bucket asignado. Para ello, se emplearon librerías específicas como couchbase, dns, y time, lo que permitió establecer conexión segura al clúster mediante credenciales configuradas. Luego se seleccionó la colección deseada y se ejecutó una consulta N1QL para

obtener los datos. Finalmente, se incorporó control de errores para asegurar que el proceso se realice de forma correcta y estable. Este procedimiento facilitó el acceso directo a la información, permitiendo futuras tareas de limpieza, análisis y visualización.

```python
import json
from couchbase.cluster import Cluster
from couchbase.options import ClusterOptions
from couchbase.auth import PasswordAuthenticator
from couchbase.exceptions import CouchbaseException

# Conexión
cluster = Cluster("couchbase://localhost", ClusterOptions(
    PasswordAuthenticator("Diego", "Aries1840")))
bucket = cluster.bucket("Noticias")
coleccion = bucket.default_collection()

# Leer lista de objetos
with open("Noticias.json", "r", encoding="utf-8") as f:
    lista = json.load(f)

# Subir cada documento
for doc in lista:
    doc_id = doc.get("id")
    if doc_id:
        coleccion.upsert(doc_id, doc)
        print(f"✅ Documento '{doc_id}' subido.")
    else:
        print("⚠ Documento sin ID, omitido.")
```

Análisis de sentimiento

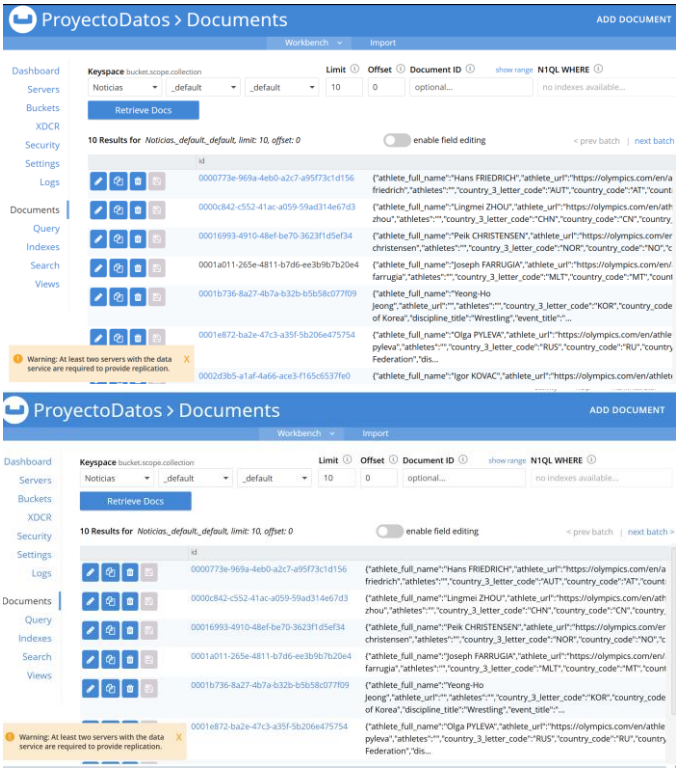
Se usó un archivo con información de películas y series de Netflix. Luego, con Python, se revisaron las descripciones de cada título para ver si tenían palabras positivas, negativas o neutras. Así se pudo saber si el contenido parecía transmitir buenos o malos sentimientos, según las palabras que tenía.

```python
import pandas as pd
df = pd.read_csv('netflix_titles1.csv')

# Palabras clave
positive_words = ['fun', 'happy', 'love', 'great', 'excellent', 'adventure', 'inspiring']
negative_words = ['murder', 'crime', 'death', 'sad', 'abuse', 'tragedy', 'drug']

# Clasificador simple
def sentimient(text):
    text = str(text).lower()
    for word in positive_words:
        if word in text:
            return 'positive'
    for word in negative_words:
        if word in text:
            return 'negative'
    return 'neutral'

df['sentiment'] = df['description'].apply(sentimient)
print(df[['title', 'description', 'sentiment']])
```

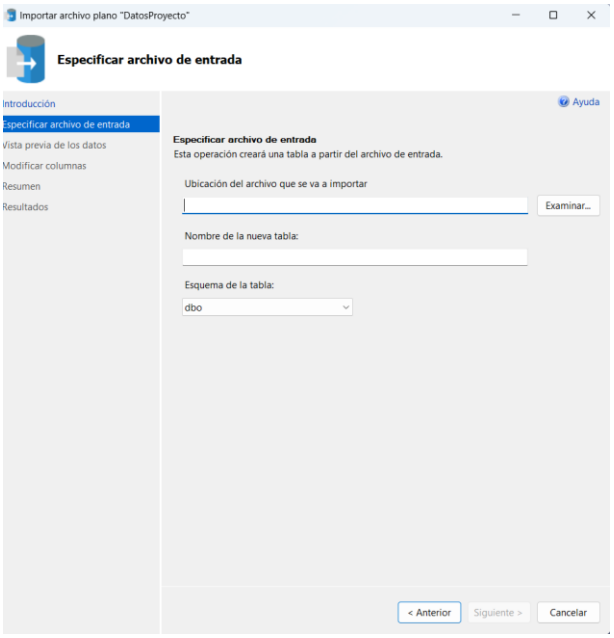
```
                      title  \
0       Dick Johnson Is Dead
1              Blood & Water
2                  Ganglands
3      Jailbirds New Orleans
4               Kota Factory
...                      ...
8802                Zodiac
8803           Zombie Dumb
8804           Zombieland
8805                Zoom
8806               Zubaan
```

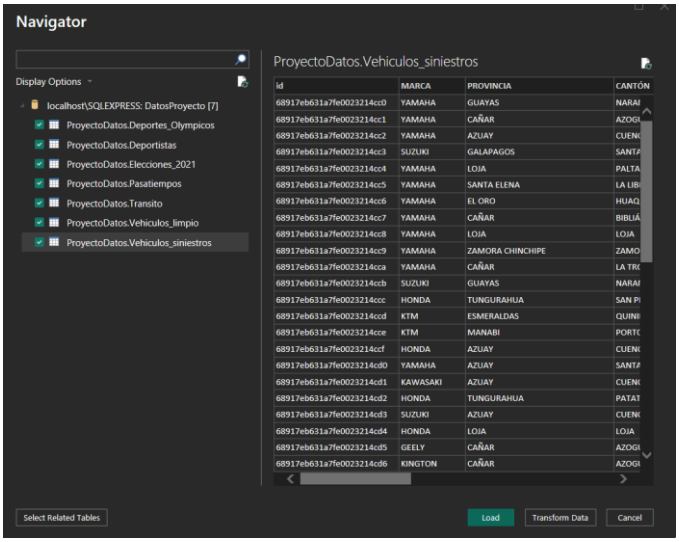
Exportación de SQL SERVER

Se realizo desde la interfaz grafica se exportaron los archivos csv en la cual cada csv representaba la colección previo a esto se exportaron desde Atlas las colecciones a csv ya que la data estaba previamente limpia



Podemos verificar que ya se abrió





IX. VISUALIZACIÓN DE INFORMACIÓN

Grafico 1: Suma de muertos por provincia

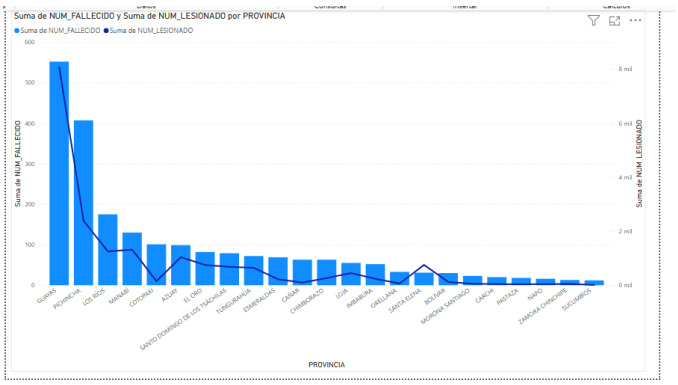


Grafico 2: Siniestros de transito por causa

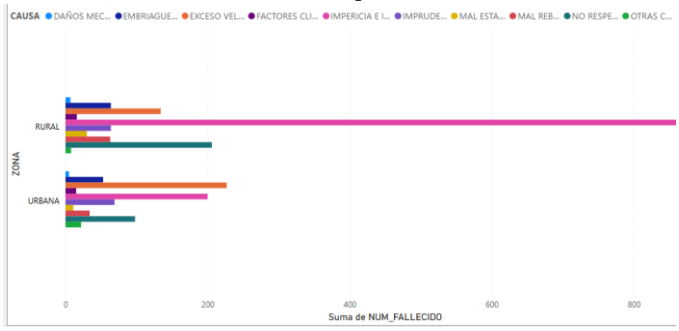


Grafico 3: Numero de fallecidos y lesionados por provincia

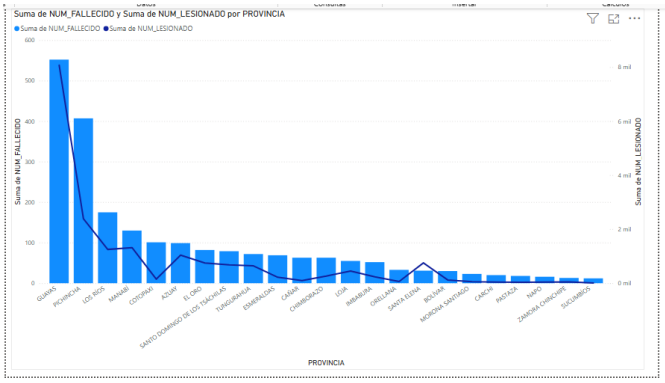


Grafico 4: Distribucion por cada tipo de vehículo

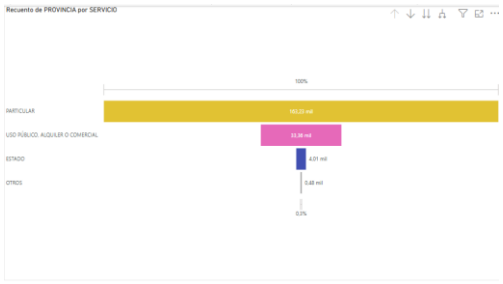


Grafico 5: Dominancia de países por categoría de películas

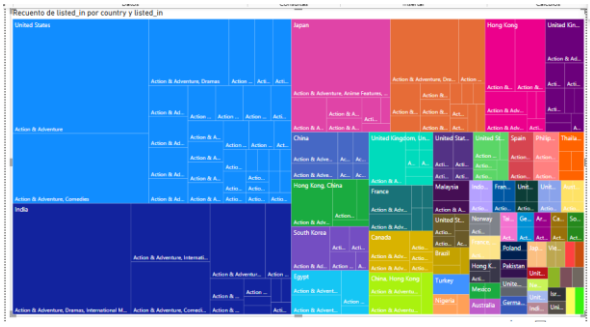
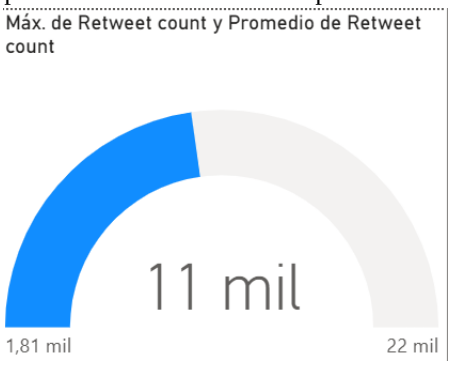


Grafico 6: Podemos ver que esta por debajo el promedio de lo retweet de la población



ecuador, y no hubo interacción de ecuatorianos en otro paices

Grafico 7: En este caso todo las interacciones que hubo entre los candidatos fueron específicamente de

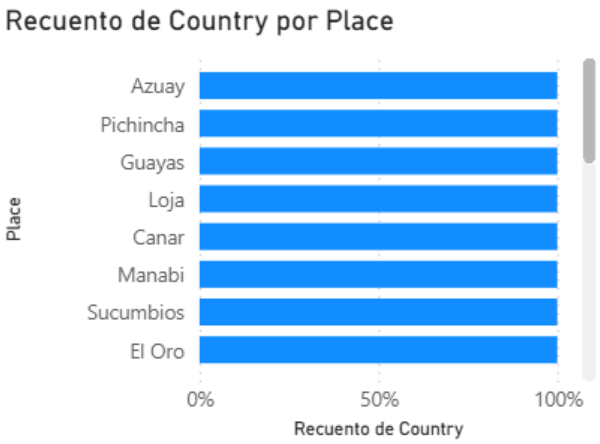


Grafico 8: La fecha con respecto a los retweet y se da la conclusión de que se da en el mes de enero y pocos días de febrero

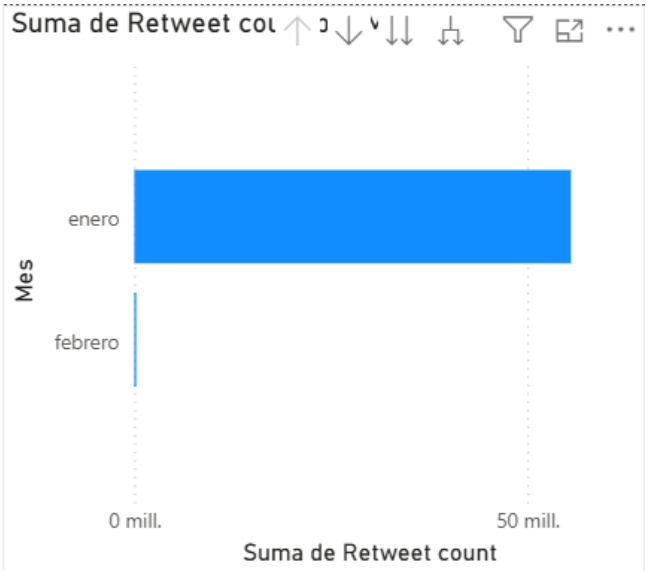


Grafico 9: En este estudio se puede apreciar que en la provincia de pichincha el candidato posee mas apoyo.

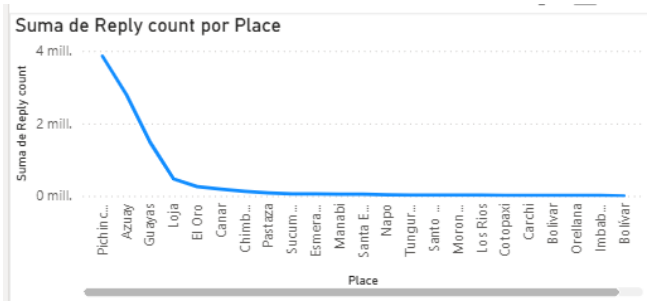


Grafico 9: En este estudio se puede apreciar que en la provincia de pichincha el candidato posee mas apoyo.

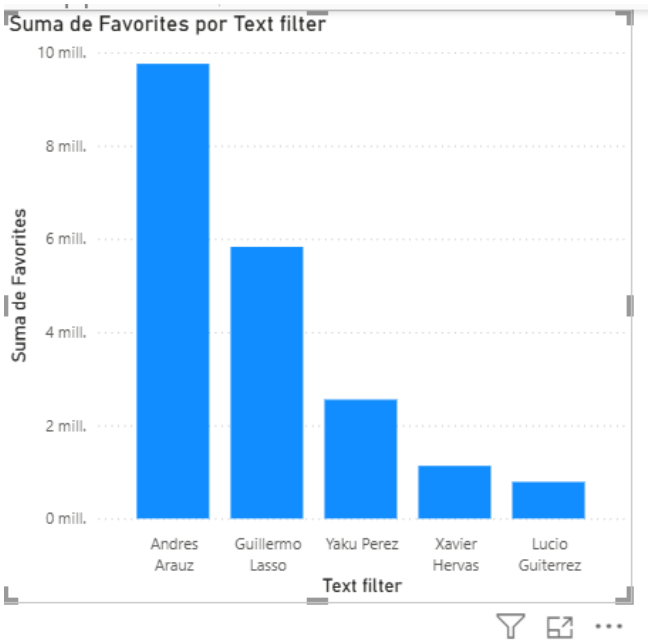


Grafico 10: en esta grafica se puede apreciar que Andres Arauz tiene un dice alto de ser de los favoritos en estas elecciones

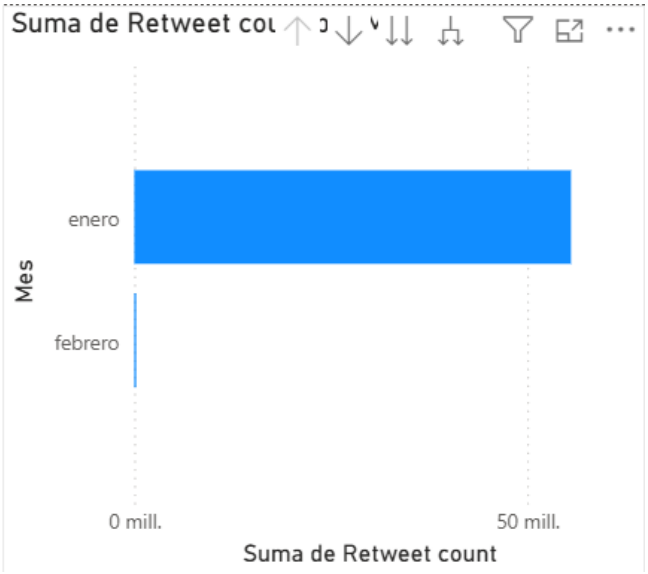


Grafico 6: Podemos ver que esta por debajo el promedio de lo retweet de la población

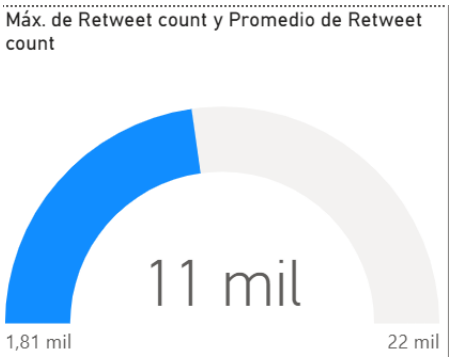


Grafico 11: Número de atletas que posee cada país, nos permitirá visualizar que país posee más atletas

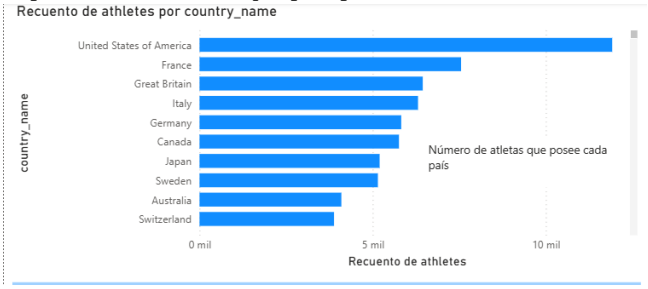


Grafico12:
Visualización que permite ver cuantos atletas hay por cada disciplina

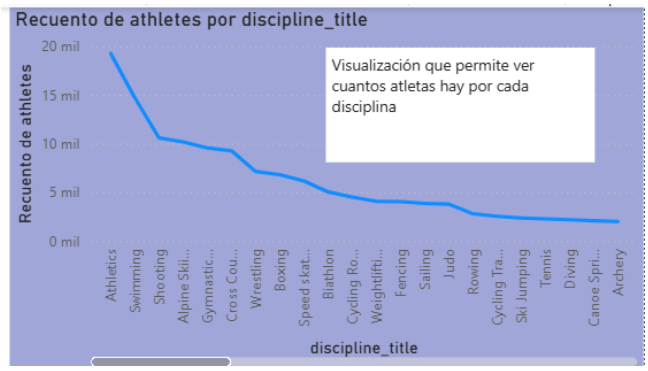


Grafico 13: Número de medallas ganadas por cada país

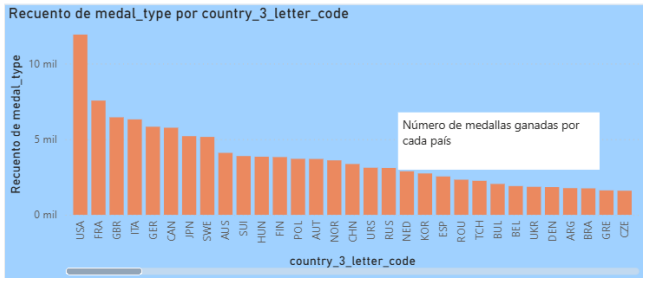


Grafico14:
Cuantas medallas se han otorgado por cada disciplina

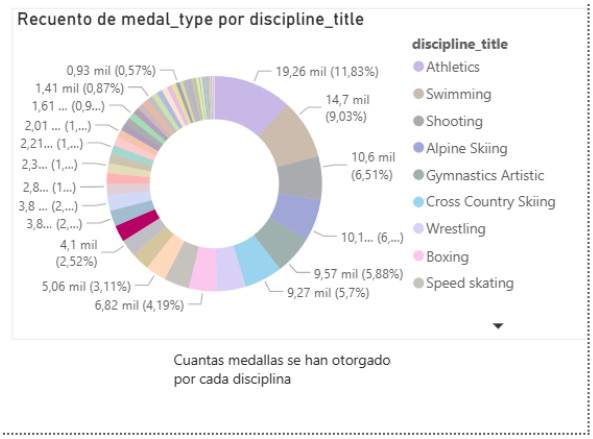


Grafico 6: Podemos ver que esta por debajo el promedio de lo retweet de la población

X. RESULTADOS OBTENIDOS

1. Se podría determinar que la mayoría de muertes se dan en las dos provincias principales del país que son Quito y Guayaquil.
2. La mayoría de muertes se dan por fallas del mecánico o en si choques esto serviría para prevenir o buscar una entidad que se encargue de esto.
3. En Ecuador hay dos pulsos políticos que predominan en Ecuador.
4. En el país quienes predominan son los RC5 y la de Guillermo Lasso.
5. Con respecto a la temática deportiva podemos presencias que Estados Unidos posee el mayor numero de atletas es decir que va a participar en la mayoría de disciplinas.
6. Siguiendo la tendencia anterior comprobamos que Estados Unidos es pionero en los juegos olímpicos ya que cuenta con mas medallas ganadas que otros países.

XI. CONCLUSIONES Y RECOMENDACIONES.

La extracción de datos desde la web representa una estrategia valiosa para fortalecer la calidad y actualidad del análisis de información. Mediante técnicas como web scraping o el uso de APIs abiertas, es posible acceder a datasets relevantes y representativos del contexto ecuatoriano. Esta práctica no solo permite enriquecer los modelos analíticos, sino también facilita la toma de decisiones informadas al integrar múltiples fuentes y temáticas. En definitiva, aprovechar datos dinámicos directamente desde la web impulsa una mejora sustancial en el desarrollo de soluciones tecnológicas orientadas al país

XII. DESAFÍOS Y PROBLEMAS ENCONTRADOS

Se intentó usar APIs para obtener datos, pero no dieron información útil o presentaron errores. También se trató de exportar datos desde el código hacia diferentes bases, pero hubo problemas como fallas del servidor o estructuras incorrectas. Por eso, se usaron otras fuentes como archivos CSV de Kaggle.

XIII. BLIBLIOGRAFIA

[1] **Kaggle Dataset** Kaggle.com, 2025. https://www.kaggle.com/datasets

[2] "Conjunto de datos - Datos Abiertos Ecuador", Datosabiertos.gob.ec , 2021. https://www.datosabiertos.gob.ec/dataset/