ESCUELA POLITÉCNICA NACIONAL

ESCUELA DE FORMACIÓN DE TECNÓLOGOS

Análisis de datos
(TDSD232)

ASIGNATURA:	Arquitectura de Computadores
PROFESOR:	Ing. Lorena Chulde MSc.
PERÍODO ACADÉMICO:	2023-B

Proyecto final

ESTUDIANTES:	Ashqui Ariel
	Parra Joel

Contenido

1.1 Tabla de ilustraciones

Informe final de Análisis y Visualización de Datos

1.2 Definición del caso de estudio

El presente proyecto se centra en la aplicación de técnicas avanzadas de análisis de datos para abordar y resolver problemáticas complejas a través de la integración y visualización de grandes volúmenes de información. Utilizando datos provenientes de múltiples fuentes, tanto estructuradas como no estructuradas, se ha diseñado y desarrollado una arquitectura de datos robusta que permite la recolección, limpieza, transformación y análisis de la información. El objetivo principal es extraer conocimientos significativos que puedan apoyar la toma de decisiones informadas en distintos contextos. A lo largo del proyecto, se han empleado diversas herramientas y tecnologías, combinando bases de datos relacionales y no relacionales, para maximizar la eficiencia y efectividad del proceso analítico. Los resultados obtenidos se han visualizado a través de dashboards interactivos, permitiendo a los usuarios explorar y comprender los datos de manera intuitiva y dinámica.

1.3 Introducción

1.3.1 Objetivo General

Aplicar técnicas avanzadas de análisis de datos para integrar y visualizar información proveniente de múltiples fuentes, con el fin de generar conocimientos que apoyen la toma de decisiones informadas en diversos contextos.

1.3.2 Objetivos Específicos

Desarrollar una arquitectura de datos eficiente que integre y procese información de diversas fuentes, utilizando tanto bases de datos relacionales como no relacionales.

Implementar procesos de extracción, limpieza, y transformación de datos para convertir la información cruda en formatos estandarizados, facilitando su análisis y visualización posterior.

Crear dashboards interactivos que permitan a los usuarios explorar los datos de manera intuitiva, identificando patrones y tendencias relevantes que contribuyan a resolver problemáticas específicas.

1.4 Descripción del Equipo de Trabajo y Actividades Realizadas por Cada Miembro

El equipo de trabajo estuvo conformado por dos miembros principales, quienes colaboraron estrechamente para cumplir con los requisitos del proyecto. A continuación, se detalla la contribución

de cada miembro del equipo y las actividades clave realizadas:

1.4.1 Joel Parra

Actividad Principal: Implementación y configuración de las bases de datos tanto relacionales como NoSQL.

Responsabilidades:

Configuró las bases de datos relacionales utilizando SQL Server y MySQL, diseñando el esquema y optimizando el rendimiento para las consultas requeridas.

Implementó las bases de datos NoSQL, utilizando MongoDB y Redis, asegurando que los datos no estructurados fueran almacenados y recuperados de manera eficiente.

Desarrolló los procesos necesarios para la migración de datos entre las bases de datos SQL y NoSQL, demostrando el flujo bidireccional de datos entre estas tecnologías.

Participó en la conversión de diferentes tipos de datos, incluyendo la transformación de JSON a CSV y viceversa.

1.4.2 Ariel Ashqui

Actividad Principal: Integración de datos y desarrollo de dashboards interactivos para la visualización de la información.

Responsabilidades:

Diseñó y ejecutó procesos de extracción, transformación y carga (ETL) para asegurar la correcta integración de datos de diversas fuentes en la arquitectura del sistema.

Implementó scripts para la conversión de datos entre formatos, como JSON a CSV, garantizando la coherencia y calidad de los datos durante la transformación.

Desarrolló dashboards en Power BI que permiten la visualización de los datos transformados, facilitando la identificación de patrones y tendencias clave para la toma de decisiones.

Colaboró en la gestión del repositorio final en SQL Server, asegurando que todos los scripts y configuraciones utilizados estuvieran correctamente documentados y almacenados.

1.5 Diagrama de Gantt

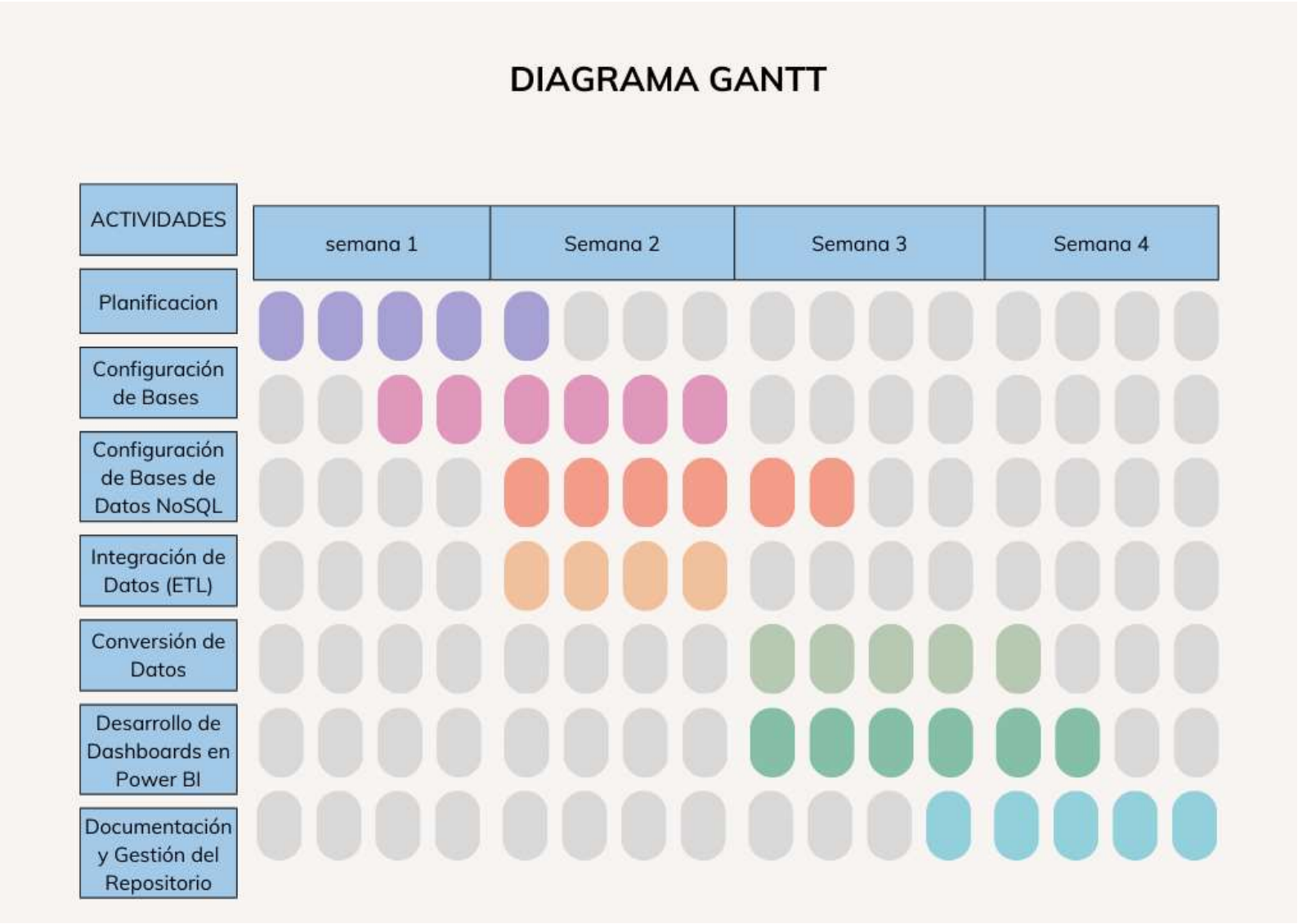


Ilustración 1. Diagrama de Gantt

1.6 Recurso y herramientas utilizadas

1.6.1 Recursos Tecnológicos

- **Computadoras Personales:** Cada miembro del equipo utilizó una computadora con especificaciones adecuadas para ejecutar entornos de desarrollo, bases de datos y herramientas de análisis.
- **Conexión a Internet:** Se requirió una conexión a Internet estable para acceder a bases de datos en la nube, descargar herramientas, y colaborar en tiempo real.

1.6.2 Bases de Datos

- **SQL Server:** Utilizado como base de datos relacional para almacenar y gestionar datos estructurados, permitiendo la realización de consultas complejas y la integración con otras bases de datos.
- **MySQL:** Otro sistema de gestión de bases de datos relacional utilizado para complementar la arquitectura SQL y manejar diferentes tipos de datos.
- **MongoDB:** Base de datos NoSQL de documentos utilizada para almacenar datos no estructurados en formato JSON, facilitando su escalabilidad y flexibilidad.

1.6.3 Herramientas de Desarrollo y ETL

- **Python:** Lenguaje de programación utilizado para desarrollar scripts de extracción, transformación y carga (ETL), así como para la conversión de formatos de datos (JSON a CSV, CSV a JSON).
- **Pandas:** Biblioteca de Python utilizada para la manipulación y análisis de datos, especialmente en el procesamiento de archivos CSV y JSON.
- **MongoDB Compass:** Herramienta gráfica utilizada para la gestión y visualización de la base de datos MongoDB.

1.6.4 Herramientas de Visualización

- **Power BI:** Herramienta de inteligencia empresarial utilizada para crear dashboards interactivos que permiten la visualización de los datos transformados, facilitando la interpretación y toma de decisiones.
- **Excel:** Utilizado para el manejo inicial de datos en formato CSV y para la creación de diagramas de Gantt que representan la planificación y ejecución del proyecto.

1.6.5 Gestión de Repositorios y Documentación

- **GitHub:** Plataforma utilizada para alojar el código fuente, scripts de migración y documentos relacionados con el

proyecto. Se utilizó Git para el control de versiones, asegurando que todos los cambios fueran registrados y gestionados de manera colaborativa.

- **Microsoft Word:** Utilizado para la redacción del informe final en formato IEEE, facilitando la creación de un documento estructurado y bien formateado.

1.7 Arquitectura de la solución.

La arquitectura de la solución desarrollada en este proyecto se diseñó para integrar y procesar eficientemente grandes volúmenes de datos provenientes de diversas fuentes, utilizando tanto bases de datos relacionales como no relacionales. A continuación, se describe la estructura y componentes principales de la arquitectura.

1.7.1 Origen de los Datos:

Los datos utilizados en este proyecto provinieron de múltiples fuentes, incluyendo archivos CSV, JSON, y bases de datos existentes. Estas fuentes fueron seleccionadas por su relevancia en el contexto del análisis y por su capacidad de aportar información valiosa para el estudio.

1.7.2 Bases de Datos Relacionales:

Se implementaron bases de datos relacionales utilizando SQL Server y MySQL. Estas bases de datos fueron responsables de almacenar datos estructurados, como tablas de usuarios, transacciones, y otras entidades con relaciones definidas entre sí. La elección de SQL Server como repositorio final permitió gestionar grandes volúmenes de datos de manera eficiente, ofreciendo soporte para consultas complejas y garantizando la integridad referencial [1].

1.7.3 Bases de Datos NoSQL:

Para manejar datos no estructurados y semiestructurados, se implementaron bases de datos NoSQL como MongoDB y Redis. MongoDB se utilizó para almacenar documentos en formato JSON, lo que permitió una mayor flexibilidad en la gestión de datos heterogéneos. Redis, por su parte, se empleó para el almacenamiento en caché y la gestión de datos de acceso rápido, mejorando el rendimiento en la recuperación de información [2].

1.7.4 Integración de Datos y Procesos ETL:

La arquitectura incluyó un pipeline de procesos ETL (Extracción, Transformación y Carga) desarrollado en Python, que facilitó la integración de datos entre las bases de datos SQL y NoSQL. Este pipeline automatizó la conversión de datos entre diferentes formatos, como CSV a JSON y viceversa, y aseguró que los datos fueran limpiados y transformados antes de ser almacenados en las bases de datos correspondientes [3]. Además, se implementaron procesos para migrar datos de SQL a NoSQL y de NoSQL a

SQL, demostrando la interoperabilidad entre ambos tipos de bases de datos.

1.7.5 Visualización de Datos:

Una vez que los datos fueron integrados y procesados, la información fue visualizada a través de dashboards interactivos en Power BI. Estos dashboards permitieron a los usuarios finales explorar los datos de manera intuitiva, facilitando la identificación de patrones y tendencias relevantes. La arquitectura de la solución soportó la actualización dinámica de los dashboards, lo que permitió la visualización en tiempo real de los datos más recientes [4].

1.7.6 Repositorio de Código y Documentación:

El código fuente, scripts de ETL, y configuraciones de bases de datos fueron gestionados en un repositorio GitHub, asegurando un control de versiones efectivo y permitiendo la colaboración en tiempo real entre los miembros del equipo. SQL Server fue utilizado como el repositorio final para todos los datos procesados, consolidando la información en un solo lugar y facilitando el acceso para análisis posteriores [5].

1.8 Conclusiones

La implementación de bases de datos relacionales y NoSQL en conjunto brindó flexibilidad y escalabilidad a la arquitectura, permitiendo una gestión eficaz de diferentes tipos de datos.

La automatización de procesos ETL garantizó la integridad y la calidad de los datos a lo largo de su ciclo de vida, desde la extracción hasta la visualización.

Los dashboards desarrollados proporcionaron una herramienta poderosa para explorar y comprender los datos en tiempo real, lo que resultó en una mayor capacidad de respuesta ante situaciones cambiantes.

1.9 Recomendaciones

Se recomienda la adopción continua de tecnologías híbridas que combinen bases de datos SQL y NoSQL en proyectos futuros, para aprovechar las ventajas de ambos enfoques en la gestión de datos.

Es aconsejable mejorar la automatización de los procesos ETL con el uso de herramientas más avanzadas, como Apache NiFi o Talend, para manejar flujos de datos más complejos y grandes volúmenes de información.

Para maximizar el valor de los dashboards, se sugiere capacitar a los usuarios finales en el uso de herramientas de visualización como Power BI, asegurando que puedan extraer el máximo provecho de las visualizaciones disponibles.

1.10 Desafíos y Problemas Encontrados

Durante el desarrollo del proyecto, el equipo enfrentó varios desafíos técnicos y metodológicos que requirieron soluciones creativas para garantizar el éxito de la implementación. A continuación, se describen los principales problemas encontrados y cómo fueron abordados:

1.10.1 Integración de Datos entre SQL y NoSQL

Uno de los desafíos más significativos fue la integración de datos entre bases de datos SQL y NoSQL. Debido a las diferencias en los modelos de datos y las estructuras de almacenamiento, la transferencia de información entre estos sistemas presentó problemas de consistencia y compatibilidad. Este desafío fue abordado mediante el desarrollo de procesos ETL personalizados que aseguraron la correcta conversión de datos entre formatos y la preservación de su integridad durante la migración.

1.10.2 Manejo de Volúmenes de Datos

La arquitectura debía manejar grandes volúmenes de datos de diversas fuentes, lo que puso a prueba las capacidades de almacenamiento y procesamiento de las bases de datos utilizadas. Para mitigar este problema, se optimizaron las consultas y se empleó Redis para el almacenamiento en caché, lo que mejoró el rendimiento en la recuperación de datos y redujo la carga en las bases de datos principales.

1.10.3 Visualización en Tiempo Real

La actualización dinámica de los dashboards en Power BI presentó dificultades, especialmente en la sincronización de datos provenientes de múltiples fuentes en tiempo real. Este problema fue resuelto mediante la implementación de actualizaciones programadas y la configuración de flujos de datos en Power BI, lo que permitió una visualización casi en tiempo real con datos actualizados.

1.10.4 Compatibilidad de Formatos de Datos

La necesidad de convertir datos entre diferentes formatos, como JSON a CSV y viceversa, resultó en problemas de compatibilidad y pérdida de datos en algunas instancias. Este desafío se abordó mediante la creación de scripts de transformación específicos que validaban y corregían los errores en los datos antes de su procesamiento final.

1.11 Extracción de datos

El proceso comenzó con la extracción de datos desde archivos CSV y su conversión a una base de datos SQL. Utilizamos Python, específicamente la biblioteca pandas, para leer los archivos CSV y cargarlos en estructuras de datos tabulares conocidas como DataFrames. Estas estructuras de datos se asemejan a tablas de una base de datos, lo que facilita su manipulación y análisis.

Una vez que los datos estaban en DataFrames, procedimos a crear una base de datos SQL. Para ello, utilizamos SQLAlchemy, una biblioteca que permite interactuar con bases de datos SQL desde Python. Establecimos una conexión a una base de datos SQLite, un

sistema de gestión de bases de datos ligero que guarda toda la información en un único archivo. Los datos de los DataFrames se almacenaron en la base de datos como tablas, asegurándonos de que cada tabla se creara correctamente y contuviera los datos esperados.

Posteriormente, realizamos la transformación de la base de datos SQL a una base de datos NoSQL. En este caso, utilizamos MongoDB, un sistema de bases de datos NoSQL que es ideal para almacenar datos semiestructurados o no estructurados. Para llevar a cabo esta transformación, primero conectamos Python a una instancia de MongoDB mediante la biblioteca pymongo. Luego, migramos los datos desde las tablas de la base de datos SQL a colecciones en MongoDB. Convertimos los DataFrames de pandas en listas de diccionarios, un formato que MongoDB puede manejar fácilmente, e insertamos estos datos en las colecciones correspondientes.

Finalmente, realizamos la extracción de datos desde la base de datos NoSQL y los convertimos a archivos JSON. Los datos almacenados en las colecciones de MongoDB se leyeron y volvieron a convertir en DataFrames de pandas. Desde estos DataFrames, exportamos los datos a archivos JSON, un formato ampliamente utilizado en aplicaciones web y APIs debido a su simplicidad y compatibilidad con diversos lenguajes de programación. Cada fila del DataFrame se convirtió en un objeto JSON independiente, y el archivo resultante se estructuró de manera que cada objeto JSON ocupaba una línea del archivo, facilitando su lectura y procesamiento posterior.

En resumen, el proceso abarcó desde la integración de datos de archivos CSV en una base de datos SQL, la transformación de esos datos a un entorno NoSQL, hasta la extracción final de los datos en un formato JSON adecuado para su distribución y uso en múltiples plataformas. Este flujo de trabajo demostró la versatilidad y eficacia de manejar datos a través de diferentes sistemas de almacenamiento, permitiendo adaptarse a diversos contextos y necesidades de análisis.

1.12 Análisis de información y Visualización de información

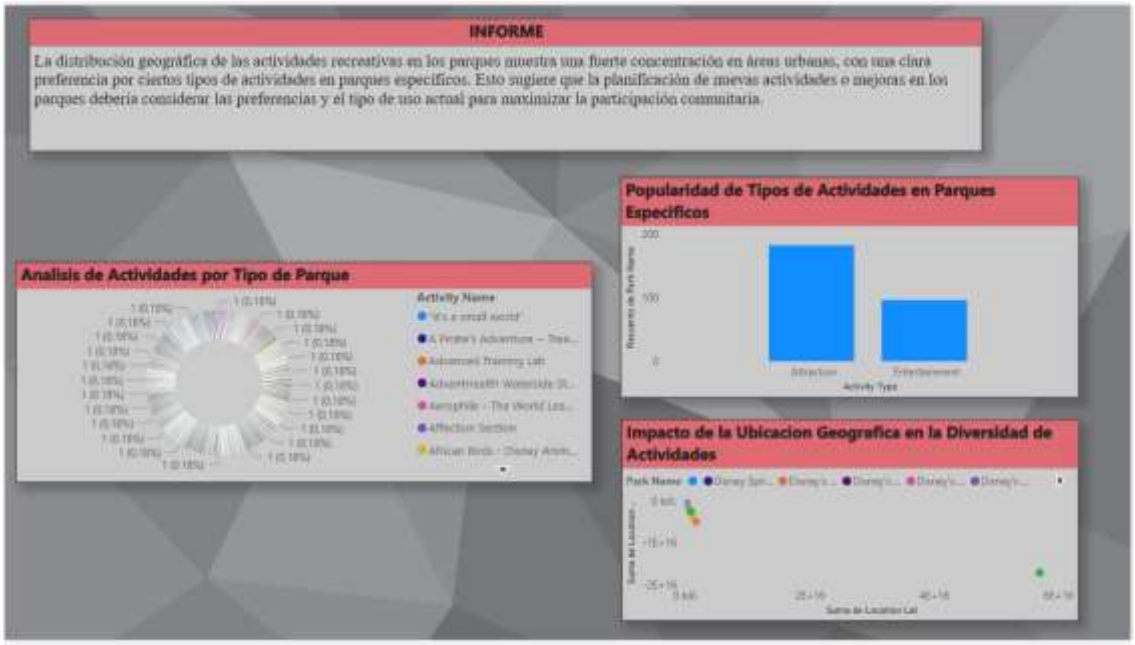


Ilustración 2. Análisis 1 (3 casos de estudio)

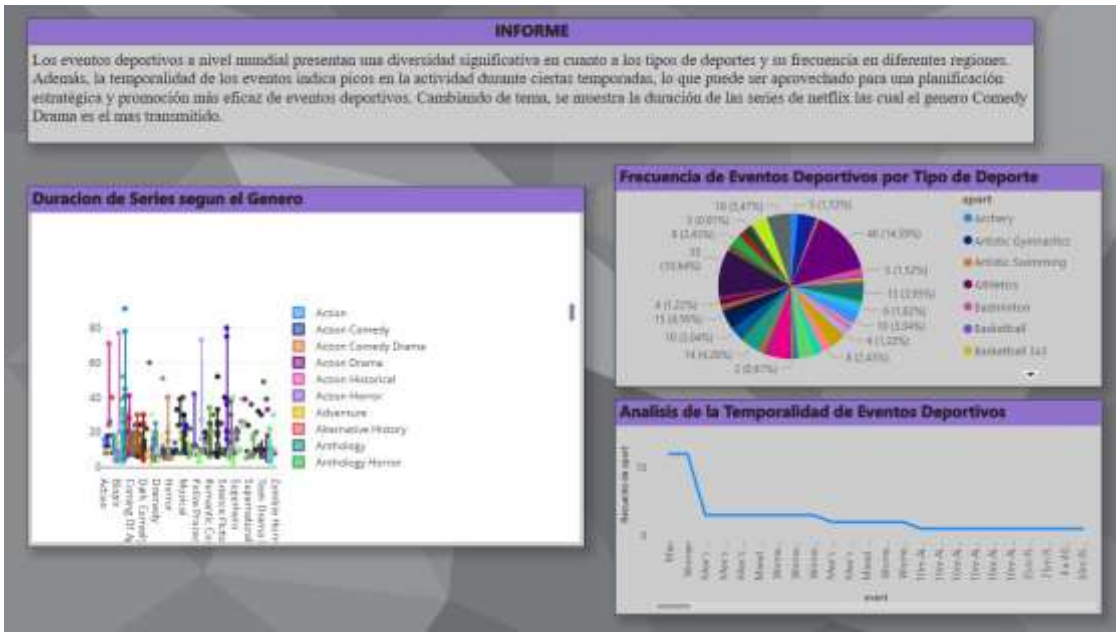


Ilustración 3. Análisis 2 (3 casos de estudio)

Ilustración 4. Análisis 3 (3 casos de estudio)

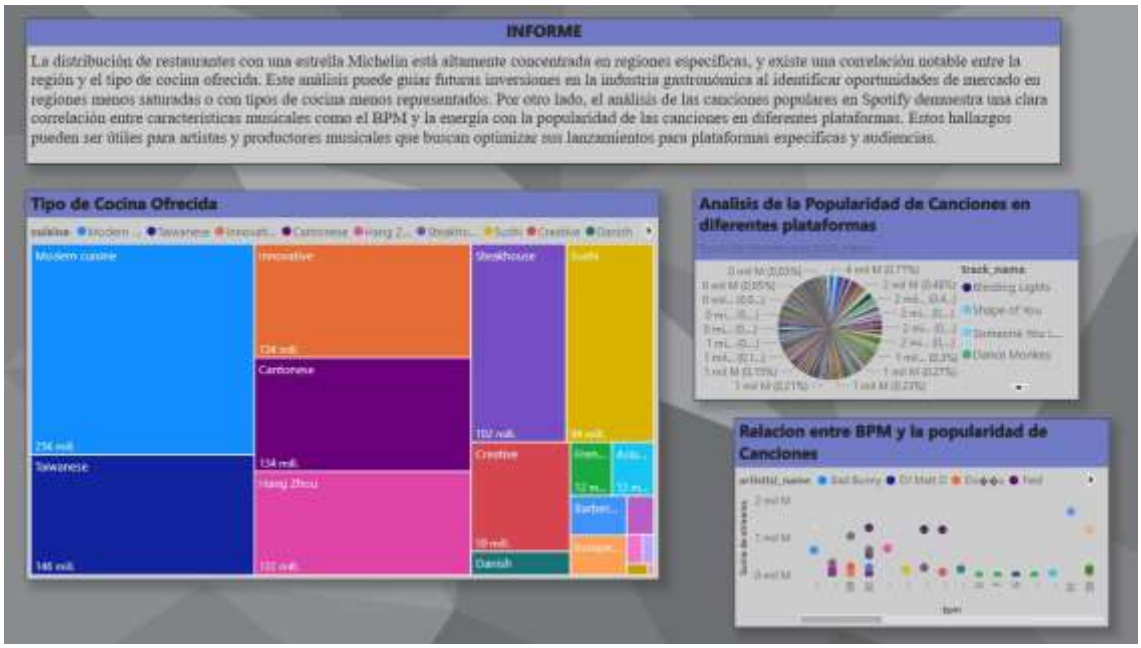


Ilustración 5. Análisis 4 (3 casos de estudio)

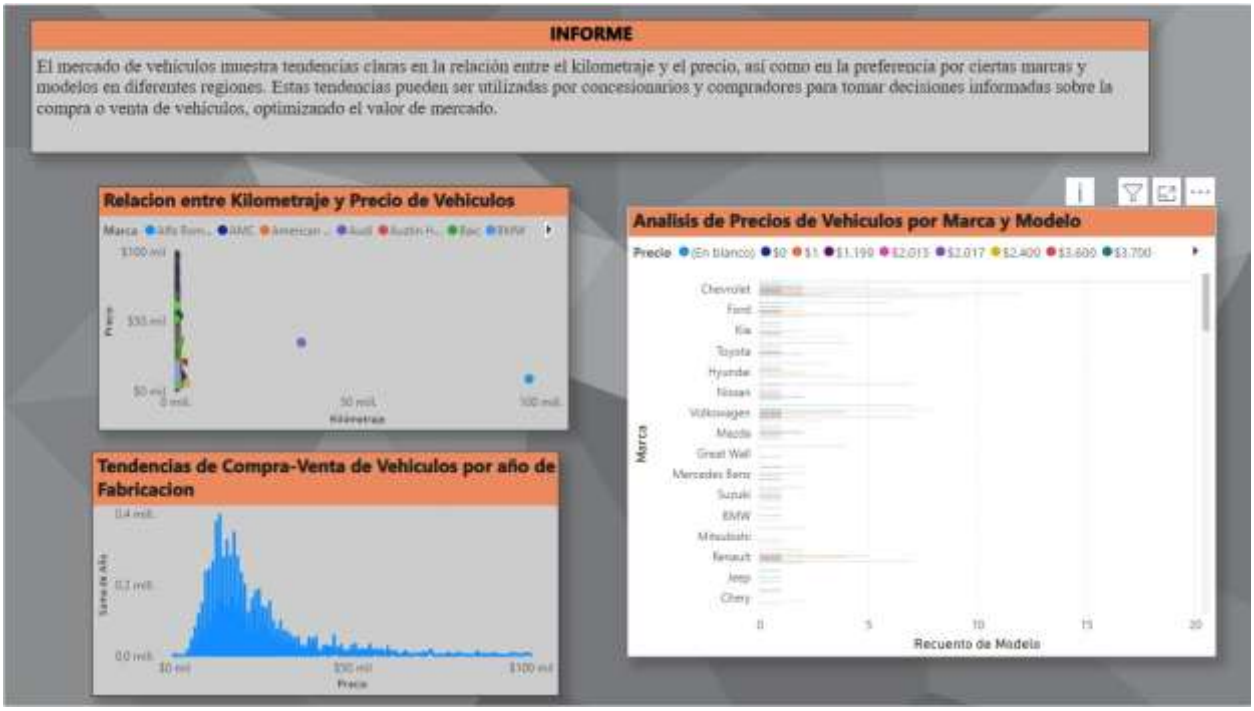


Ilustración 6. Análisis 5 (3 casos de estudio)

1.13 Anexos

1.13.1 Link GitHub

Link GitHub

1.13.2 Link del video sobre el proceso completo de recopilación de datos

Video sobre el proceso completo de recopilación de datos

1.13.3 Link del video sobre las conclusiones de los dashboards

Video sobre la conclusión de los dashboards

1.13.4 Link presentación

Presentación Prezi

1.13.5 Imágenes de conversión y extracción de datos

```
# Extraccion de datos y transformacion de archivos csv a base de datos

import pandas as pd
from sqlalchemy import create_engine

# Lista de nombres de archivos CSV
csv_files = [
    'activities (1).csv',
    'events.csv',
    'medals.csv',
    'one-star-michelin-restaurants.csv',
    'patiotuerca2023-02-101.csv',
    'patiotuerca2023-02-102.csv',
    'Spotify Songs 2023.csv',
    'tv_shows_data.csv'
]

# Cargar cada archivo CSV en un DataFrame con manejo de codificación
dataframes = {}
for file in csv_files:
    df_name = file.split('.')[0]  # Nombre del DataFrame basado en el nombre del archivo
    try:
        dataframes[df_name] = pd.read_csv(file, encoding='utf-8')  # Intento cargar con utf-8
    except UnicodeDecodeError:
        dataframes[df_name] = pd.read_csv(file, encoding='latin1')  # Si falla, usa latin1

# Crear una conexión a una base de datos SQLite
engine = create_engine('sqlite:///mi_base_de_datos.db')

# Guardar cada DataFrame como una tabla en la base de datos SQL
for name, df in dataframes.items():
    df.to_sql(name, con=engine, index=False, if_exists='replace')
    print(f"Tabla '{name}' guardada en la base de datos.")

from sqlalchemy import text

# Verifica las tablas creadas
with engine.connect() as con:
    result = con.execute(text("SELECT name FROM sqlite_master WHERE type='table';"))
    tables = result.fetchall()
    print("Tablas en la base de datos:")
    for table in tables:
        print(table[0])
```

Ilustración 7. Extraccion de datos y transformacion de archivos csv a base de datos

```
Tabla 'activities (1)' guardada en la base de datos.
Tabla 'events' guardada en la base de datos.
Tabla 'medals' guardada en la base de datos.
Tabla 'one-star-michelin-restaurants' guardada en la base de datos.
Tabla 'patiotuerca2023-02-101' guardada en la base de datos.
Tabla 'patiotuerca2023-02-102' guardada en la base de datos.
Tabla 'Spotify Songs 2023' guardada en la base de datos.
Tabla 'tv_shows_data' guardada en la base de datos.
Tablas en la base de datos:
activities (1)
events
medals
one-star-michelin-restaurants
patiotuerca2023-02-101
patiotuerca2023-02-102
Spotify Songs 2023
tv_shows_data
```

Ilustración 8. Extraccion de datos y transformacion de archivos

```
# Conversion de base de datos sql a no sql
```

```
from pymongo import MongoClient

# Conectar a MongoDB (cambia 'your_username', 'your_password' y 'your_cluster_url' por los valores reales)
client = MongoClient('mongodb+srv://jp7mmcort:1234@final.01zdk.mongodb.net/')

# Crear una base de datos en MongoDB
db = client['mi_base_de_datos_nosql']
```

```
# Verificar la conexión listando las bases de datos en MongoDB
print(client.list_database_names())
```

```
['AutoPartsXpress', 'sample_mflix', 'admin', 'local']
```

```
# Leer las tablas que existen en la base de datos SQLite
table_names = ['medals', 'events']  # Ajusta la lista con las tablas que existen

dataframes = {}
for name in table_names:
    dataframes[name] = pd.read_sql(f"SELECT * FROM {name}", con=engine)

# Verifica que las tablas se han cargado correctamente
for name, df in dataframes.items():
    print(f"Primeras filas de la tabla '{name}':")
    print(df.head(), "\n")
```

```
Primeras filas de la table 'medals':
     medal_type  medal_code  medal_date            name country_code gender  \
0    Gold Medal           1  2024-07-27  Remco EVENEPOEL          BEL      M
1  Silver Medal           2  2024-07-27    Filippo GANNA          ITA      M
2  Bronze Medal           3  2024-07-27    Wout van AERT          BEL      M
3    Gold Medal           1  2024-07-27      Grace BROWN          AUS      W
4  Silver Medal           2  2024-07-27   Anna HENDERSON          GBR      W

     discipline                          event event_type  \
0  Cycling Road    Men's Individual Time Trial        ATH
1  Cycling Road    Men's Individual Time Trial        ATH
2  Cycling Road    Men's Individual Time Trial        ATH
3  Cycling Road  Women's Individual Time Trial        ATH
```

Ilustración 9. Extraccion de datos y transformacion de archivos

```
[16]: # Insertar cada DataFrame como una colección en MongoDB
for name, df in dataframes.items():
    collection = db[name]  # Crear una colección con el nombre de la tabla
    data_dict = df.to_dict('records')  # Convertir el DataFrame a una lista de diccionarios
    collection.insert_many(data_dict)  # Insertar los datos en MongoDB
    print(f"Datos de la tabla '{name}' insertados en la colección MongoDB '{name}'")
```

```
Datos de la tabla 'medals' insertados en la colección MongoDB 'medals'
Datos de la tabla 'events' insertados en la colección MongoDB 'events'
```

```
[17]: # Listar las colecciones en la base de datos MongoDB
collections = db.list_collection_names()
print("Colecciones en la base de datos MongoDB:")
for collection in collections:
    print(collection)
```

```
Colecciones en la base de datos MongoDB:
medals
events
```

Ilustración 10. Extraccion de datos y transformacion de archivos

```
[ ]: # Conversion de base de datos mongo a sql

[11]: from pymongo import MongoClient
import pandas as pd
from sqlalchemy import create_engine

# Conectar a MongoDB
client = MongoClient('mongodb+srv://jp7macort:12345@final.01ldd.mongodb.net/')

# Seleccionar la base de datos y la colección que deseas convertir
db = client['ml_base_de_datos_nosql']
collection = db['medals']  # Ejemplo: colección 'medals'

# Leer los datos de la colección y convertirlos en un DataFrame de pandas
data = list(collection.find())  # Convertir la colección en una lista de diccionarios
df = pd.DataFrame(data)

# Mostrar las primeras filas para verificar que se han cargado correctamente
print(df.head())
```

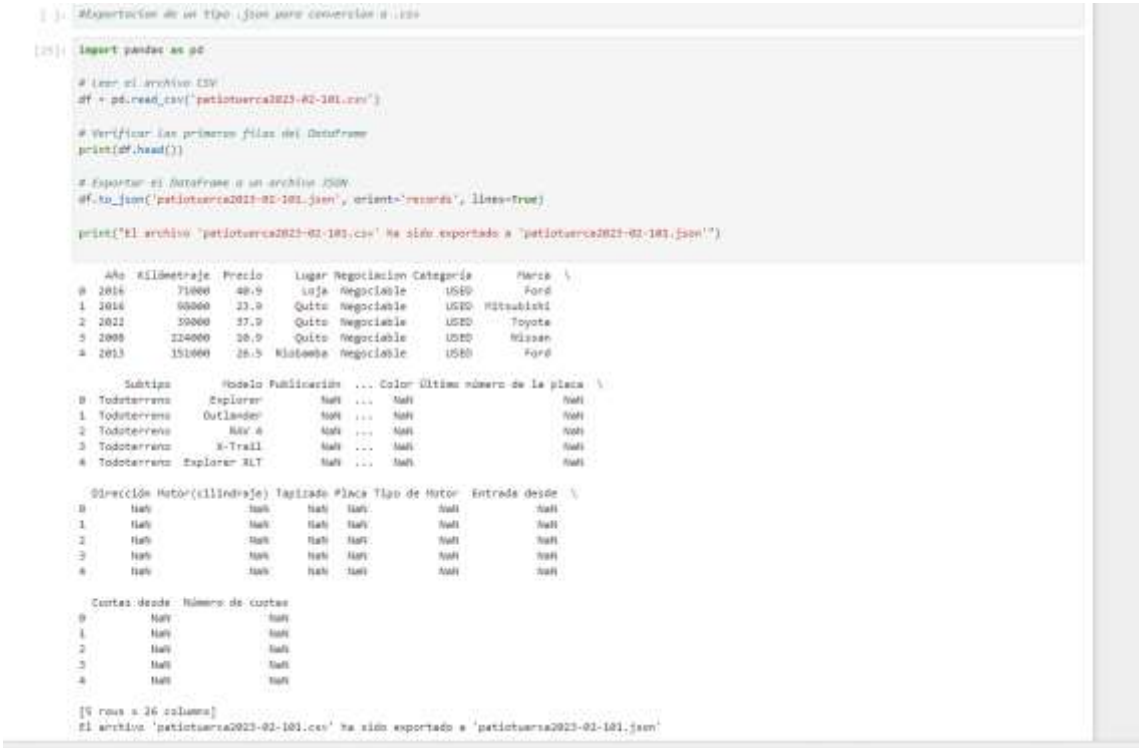
```
                        _id    medal_type  medal_code  medal_date  \
0  66baf13bd30a77da8d94c8cb    Gold Medal           1  2024-07-27
1  66baf13bd30a77da8d94c8cc  Silver Medal           2  2024-07-27
2  66baf13bd30a77da8d94c8cd  Bronze Medal           3  2024-07-27
3  66baf13bd30a77da8d94c8ce    Gold Medal           1  2024-07-27
4  66baf13bd30a77da8d94c8cf  Silver Medal           2  2024-07-27

             name country_code gender    discipline  \
0  Remco EVENEPOEL          BEL      M  Cycling Road
1    Filippo GANNA          ITA      M  Cycling Road
2    Wout van AERT          BEL      M  Cycling Road
3      Grace BROWN          AUS      W  Cycling Road
4   Anna HENDERSON          GBR      W  Cycling Road

                        event event_type  \
0    Men's Individual Time Trial        ATH
1    Men's Individual Time Trial        ATH
2    Men's Individual Time Trial        ATH
3  Women's Individual Time Trial        ATH
4  Women's Individual Time Trial        ATH
```

Ilustración 11. Extraccion de datos y transformacion de archivos

```
[11]: # Convertir ObjectId a str en la columna '_id'
df['_id'] = df['_id'].astype(str)

# Ahora, guarda el Dataframe en la base de datos SQL
df.to_sql('medals_sql', con=engine, index=False, if_exists='replace')

from sqlalchemy import text

# Verificar que la tabla se ha creado correctamente
with engine.connect() as con:
    result = con.execute(text("SELECT * FROM medals_sql LIMIT 5;"))
    for row in result:
        print(row)
```

```
('66baf13bd30a77da8d94c8cb', 'Gold Medal', 1, '2024-07-27', 'Remco EVENEPOEL', 'BEL', 'M', 'Cycling Road', "Men's Individual Time Trial", 'ATH', '/en/par
is-2024/results/cycling-road/men-s-individual-time-trial/fnl-000100--', '1503130')
('66baf13bd30a77da8d94c8cc', 'Silver Medal', 2, '2024-07-27', 'Filippo GANNA', 'ITA', 'M', 'Cycling Road', "Men's Individual Time Trial", 'ATH', '/en/par
is-2024/results/cycling-road/men-s-individual-time-trial/fnl-000100--', '1023510')
('66baf13bd30a77da8d94c8cd', 'Bronze Medal', 3, '2024-07-27', 'Wout van AERT', 'BEL', 'M', 'Cycling Road', "Men's Individual Time Trial", 'ATH', '/en/par
is-2024/results/cycling-road/men-s-individual-time-trial/fnl-000100--', '1003147')
('66baf13bd30a77da8d94c8ce', 'Gold Medal', 1, '2024-07-27', 'Grace BROWN', 'AUS', 'W', 'Cycling Road', "Women's Individual Time Trial", 'ATH', '/en/paris
-2024/results/cycling-road/women-s-individual-time-trial/fnl-000100--', '1340173')
('66baf13bd30a77da8d94c8cf', 'Silver Medal', 2, '2024-07-27', 'Anna HENDERSON', 'GBR', 'W', 'Cycling Road', "Women's Individual Time Trial", 'ATH', '/en/
paris-2024/results/cycling-road/women-s-individual-time-trial/fnl-000100--', '1012525')
```

Ilustración 12. Extraccion de datos y transformacion de archivos



```
#Exportacion de un tipo .json para conversion a .csv

import pandas as pd

# Leer el archivo CSV
df = pd.read_csv('patiotuerca2023-02-101.csv')

# Verificar las primeras filas del DataFrame
print(df.head())

# Exportar el DataFrame a un archivo JSON
df.to_json('patiotuerca2023-02-101.json', orient='records', lines=True)

print("El archivo 'patiotuerca2023-02-101.csv' ha sido exportado a 'patiotuerca2023-02-101.json'")
```

```
    Año  Kilometraje  Precio     Lugar Negociacion Categoría       Marca  \
0  2016        71000    40.9      Loja  Negociable      USED        Ford
1  2016        60000    23.9     Quito  Negociable      USED  Mitsubishi
2  2022        30000    37.9     Quito  Negociable      USED      Toyota
3  2008       224000    10.9     Quito  Negociable      USED      Nissan
4  2013       151000    26.5  Riobamba  Negociable      USED        Ford

       Subtipo       Modelo Publicación  ... Color Último número de la placa  \
0  Todoterreno     Explorer         NaN  ...   NaN                       NaN
1  Todoterreno    Outlander         NaN  ...   NaN                       NaN
2  Todoterreno        RAV 4         NaN  ...   NaN                       NaN
3  Todoterreno       X-Trail         NaN  ...   NaN                       NaN
4  Todoterreno  Explorer XLT         NaN  ...   NaN                       NaN

  Dirección Motor(cilindraje) Tapizado Placa Tipo de Motor  Entrada desde  \
0       NaN               NaN      NaN   NaN           NaN            NaN
1       NaN               NaN      NaN   NaN           NaN            NaN
2       NaN               NaN      NaN   NaN           NaN            NaN
3       NaN               NaN      NaN   NaN           NaN            NaN
4       NaN               NaN      NaN   NaN           NaN            NaN

  Cuotas desde  Número de cuotas
0          NaN               NaN
1          NaN               NaN
2          NaN               NaN
3          NaN               NaN
4          NaN               NaN

[5 rows x 26 columns]
El archivo 'patiotuerca2023-02-101.csv' ha sido exportado a 'patiotuerca2023-02-101.json'
```

Ilustración 13. Extraccion de datos y transformacion de archivos

1.14 Referencias

[1] E. F. Codd, "A relational model of data for large shared data banks," *Commun. ACM*, vol. 13, no. 6, pp. 377-387, 1970.

[2] J. Han, E. Haihong, G. Le, and J. Du, "Survey on NoSQL database," in *2011 6th International Conference on Pervasive Computing and Applications*, 2011, pp. 363-366.

[3] R. Kimball and J. Caserta, *The Data Warehouse ETL Toolkit: Practical Techniques for Extracting, Cleaning, Conforming, and Delivering Data*, 1st ed., Wiley, 2011.

[4] C. Few, *Show Me the Numbers: Designing Tables and Graphs to Enlighten*, 2nd ed., Analytics Press, 2012.

[5] D. Chappell, "Introducing SQL Server 2016," *Microsoft Corporation*, Redmond, WA, USA, 2015.