Proyecto final diseño de una arquitectura de Data Lake

Erika Alvarado, Sebastián Coronado, Dustin Marcatoma, Anthony Minga, Alan Ríos, Escuela Politécnica Nacional (EPN), Quito – Ecuador

erika.alvarado, sebastian.coronado, dustin.marcatoma, anthony.minga, alan.rios, (@epn.edu.ec)

***Resumen*— Este proyecto final detalla el diseño de una arquitectura de Data Lake que integra datos de 12 fuentes diversas, como redes sociales y archivos estáticos. Se utilizo al menos tres bases de datos NoSQL y tres relacionales, conectando el concentrador de datos SQL Server a PowerBI para la creación de dashboards. Se propone un caso de estudio con cinco temáticas específicas, cada una con su propio dashboard y al menos dos visualizaciones con geolocalización. Se abordará la extracción de datos, análisis, y se presentaran resultados, concluyendo con recomendaciones y desafíos encontrados en el siguiente informe.**

Palabras clave: SQL Server, Data Lake, juegos, películas, eventos, conciertos, ciencia, Ecuador.

I. Introducción

En este proyecto destacaremos la concepción de una robusta arquitectura de Data Lake para la integración de datos provenientes de diversas fuentes, abarcando redes sociales, archivos estáticos y otros recursos, totalizando al menos 12 fuentes heterogéneas. La implementación se basa en el uso estratégico de bases de datos NoSQL y relacionales, conectadas a un concentrador central SQL Server para la posterior generación de dashboards mediante PowerBI.

Este informe detallara la estructura de la arquitectura, los métodos de extracción de datos, el análisis de información y la configuración de PowerBI, ofreciendo una visión integral del proceso. Los resultados obtenidos en cada caso de estudio se presentarán, seguidos de conclusiones, recomendaciones y desafíos encontrados durante la implementación, garantizando la adhesión a las directrices del formato IEEE.

II. Objetivo general y especifico

Objetivo General

- Abordar cinco casos de estudio, cada uno enfocado en temáticas específicas, cada caso de estudio contara con su propio dashboard y al menos dos visualizaciones con geolocalización para una comprensión más contextualizada.

Objetivos específicos

- Seleccionar y justificar la inclusión de 12 fuentes de datos heterogéneas.
- Integrar las fuentes en la arquitectura de Data Lake, estableciendo flujos de datos y conexiones entre las bases para un almacenamiento coherente.
- Configuración de al menos tres bases de datos NoSQL y tres bases de datos relacionales para el almacenamiento de datos de manera eficiente.

III. Desarrollo del proyecto

A partir de esta sección, se desarrollan los contenidos del proyecto.

A. *Organización y trabajo en equipo*

Las funciones fueron asignadas a cada integrante del equipo de forma aleatoria, los roles de cada integrante se dividen en el líder del equipo, desarrolladores y analistas.

- ✓ Líder: Anthony Minga.
- ✓ Desarrolladores: Alan Ríos, Sebastián Coronado.
- ✓ Analistas: Dustin Marcatoma, Erika Alvarado

B. *Cronograma de actividades*

En la tabla 1 describiremos nuestro cronograma de actividades establecido para el presente proyecto.

Tabla 1
Cronograma de actividades

Tiempo	**Arquitectura Data Lake**		
	Actividad	**Fecha**	**Encargado**
1 Día	Definición del alcance y Requisitos	23-02-2024	Todo el Equipo de trabajo
3 Días	Diseño de la arquitectura del Data Lake: • Seleccionar las bases de datos NoSql y relacionales. • Planificar la estructura del Data Lake para la integración de las fuentes de datos. • Búsqueda de las fuentes para los temas seleccionados.	24-02-2024	Todo el Equipo de trabajo
3 Días	Configuración y conexión de fuentes de datos: • Configuración la conexión de las fuentes de datos. • Web scraping • Comprobación de que los datos se almacenen adecuadamente en el	27-02-2024	Erika Alvarado Alan Ríos

	Data Lake.		
1 Día	Power BI y concentrador de datos: • Conexión de Power BI al concentrador de datos MySQL Server.	01-03-2024	Alan Ríos Dustin Marcatoma
2 Días	Creación de los dashboards para las temáticas seleccionadas.	02-03-2024	Dustin Marcatoma Sebastian Coronado
1 Día	Documentación final: • Pruebas de la conectividad de datos y dashboards. • Informe. • Grabación de video con la participación de los integrantes del equipo.	04-03-2024	Erika Alvarado Anthony Minga

La elaboración de un cronograma ayuda a tener mejor organización en tiempos y desarrollo del proyecto.

C. *Recursos y herramientas utilizadas*

Los recursos empleados en el proyecto se describirán a continuación:

Data Lake es un repositorio centralizado y escalable que permite almacenar grandes volúmenes de datos en su formato bruto, estructurado o no estructurado, con el objetivo de facilitar el análisis y la extracción de información valiosa para las organizaciones.

Web scraping es el proceso de extraer automáticamente datos de sitios web, transformando la información disponible en HTML en un formato estructurado que puede ser utilizado para análisis, investigación u otros fines.

Las bases de datos NoSQL son sistemas de gestión de datos diseñados para manejar grandes volúmenes de información no estructurada o semiestructurada, proporcionando flexibilidad y escalabilidad, sin depender del modelo relacional tradicional.

Una base de datos relacional es un sistema organizado que utiliza tablas para almacenar datos, donde cada tabla se relaciona mediante claves y permite realizar consultas para acceder, gestionar y relacionar la información de manera eficiente.

SQL Server es un sistema de gestión de bases de datos relacionales desarrollado por Microsoft. Proporciona un entorno seguro y eficiente para almacenar, gestionar y recuperar datos, utilizando el lenguaje SQL para realizar consultas y operaciones en la base de datos.

Power BI es una herramienta de análisis empresarial de Microsoft que permite visualizar y compartir datos de manera interactiva. Facilita la creación de informes y paneles dinámicos mediante la conexión con diversas fuentes de datos para tomar decisiones informadas.

GitHub es una plataforma en línea basada en Git que facilita la gestión y colaboración en proyectos de desarrollo de software. Permite almacenar, versionar y compartir el código fuente, así como realizar un seguimiento de cambios y facilitar la colaboración entre desarrolladores.

Visual Studio Code es un editor de código fuente ligero y altamente personalizable desarrollado por Microsoft. Proporciona funciones avanzadas para la edición, depuración y gestión de código en diversos lenguajes de programación. Su diseño modular permite la integración de extensiones para adaptarse a diferentes entornos de desarrollo.

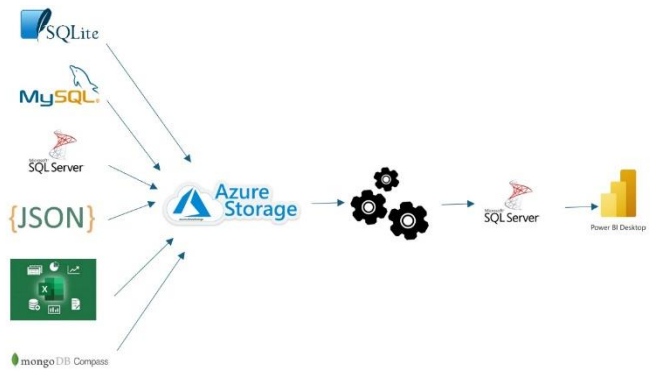

Fig. 1 Arquitectura de la solución.

D. *Extracción de datos*

En esta etapa realizamos la recopilación de datos de diversas fuentes. Con los datos recopilados procedemos a la limpieza de datos de cada tema elegido con el código que se muestra en las siguientes figuras:

```
import pandas as pd
```

```
# Cargar el archivo CSV
df = pd.read_csv('RankingVideoGames.csv')

# Eliminar filas duplicadas
df = df.drop_duplicates()

# Rellenar los valores nulos en columnas de texto con 'sin información'
df.loc[:, df.dtypes == object] = df.loc[:, df.dtypes == object].fillna('sin información')

# Rellenar los valores nulos en columnas numéricas con 0
df.loc[:, df.dtypes != object] = df.loc[:, df.dtypes != object].fillna(0)

# Guardar los datos limpios en un nuevo archivo CSV
df.to_csv('JuegosL1.csv', index=False)
```

Fig. 2 Limpieza de datos data set juegos.

En la figura 2 podemos visualizar el código para la limpieza de datos de los data sets obtenidos para eliminar filas duplicadas, rellenar valores nulos con el texto “sin información”, las columnas numéricas rellenarlas con el valor 0 y finalmente guardar los datos limpios en un nuevo archivo csv.

Este proceso se realizará con todos los data sets de los 5 temas seleccionados.

```
import pandas as pd
```

```
df = pd.read_csv('IMDb movies.csv', encoding='latin-1')

df = df.drop_duplicates()

# Rellenar los valores nulos en columnas de texto con 'sin información'
df.loc[:, df.dtypes == object] = df.loc[:, df.dtypes == object].fillna('sin información')

# Rellenar los valores nulos en columnas numéricas con 0
df.loc[:, df.dtypes != object] = df.loc[:, df.dtypes != object].fillna(0)

df.to_csv('PeliculasL1.csv', index=False)
```

Fig. 3 Limpieza de datos data set películas.

```python
import pandas as pd
```

```python
df = pd.read_csv('articles3.csv')
df = df.drop_duplicates()

# Rellenar los valores nulos en columnas de texto con 'sin información'
df.loc[:, df.dtypes == object] = df.loc[:, df.dtypes == object].fillna('sin información')

# Rellenar los valores nulos en columnas numéricas con 0
df.loc[:, df.dtypes != object] = df.loc[:, df.dtypes != object].fillna(0)

df.to_csv('NoticiasL1.csv', index=False)
```

Fig. 4 Limpieza de datos data set noticias.

```python
import pandas as pd
```

```python
df = pd.read_csv('concerts.csv')

df = df.drop_duplicates()

# Rellenar los valores nulos en columnas de texto con 'sin información'
df.loc[:, df.dtypes == object] = df.loc[:, df.dtypes == object].fillna('sin información')

# Rellenar los valores nulos en columnas numéricas con 0
df.loc[:, df.dtypes != object] = df.loc[:, df.dtypes != object].fillna(0)

df.to_csv('ConciertosL1.csv', index=False)
```

Fig. 5 Limpieza de datos data set conciertos.

```python
import pandas as pd
```

```python
df = pd.read_csv('IMDb movies.csv', encoding='latin-1')

df = df.drop_duplicates()

# Rellenar los valores nulos en columnas de texto con 'sin información'
df.loc[:, df.dtypes == object] = df.loc[:, df.dtypes == object].fillna('sin información')

# Rellenar los valores nulos en columnas numéricas con 0
df.loc[:, df.dtypes != object] = df.loc[:, df.dtypes != object].fillna(0)

df.to_csv('PeliculasL1.csv', index=False)
```

Fig. 6 Limpieza de datos data set ciencia.

E. Análisis de información

En esta etapa se realizó un análisis de los datos previamente almacenados en nuevos data sets, realizando también la conexión al gestor de MySQL Workbench junto con la creación de la base de datos.

```python
[39]: import mysql.connector as mysql #Conector necesario para MySQL
      import pandas as pd #Libreria para leer los datasets
      from datetime import datetime #Libreria para cambiar el formato de fechas de los datasets en caso de ser necesario
      from sqlalchemy import create_engine #Libreria para enviar data frames a MySQL

[2]: x1=mysql.connect(host="localhost",user="root", passwd="Tsuki1904") #Credenciales de conexion
     x1

[2]: <mysql.connector.connection.MySQLConnection at 0x1c419bf3e30>

[3]: x2=x1.cursor() #Variable encargada de ejecutar instrucciones en MySQL
     x2.execute("create database DLVideojuegos1")

[5]: x2.execute("use DLVideojuegos1")

[7]: df=pd.read_csv('JuegosL1.csv') # Leemos el datasets

[8]: df
```

Fig. 7 conexión a worbench y creación de base de datos.

En la figura 7 también vamos a visualizar la lectura de un data set de un caso de estudio seleccionado ‘juegos’.

[8]:

	name	platform	release_date	summary	meta_score	user_review
0	The Legend of Zelda: Ocarina of Time	Nintendo 64	November 23, 1998	As a young boy, Link is tricked by Ganondorf, ...	99	9.1
1	Tony Hawk's Pro Skater 2	PlayStation	September 20, 2000	As most major publishers' development efforts ...	98	7.4
2	Grand Theft Auto IV	PlayStation 3	April 29, 2008	[Metacritic's 2008 PS3 Game of the Year; Also ...	98	7.7
3	SoulCalibur	Dreamcast	September 8, 1999	This is a tale of souls and swords; transcendi...	98	8.4
4	Grand Theft Auto IV	Xbox 360	April 29, 2008	[Metacritic's 2008 Xbox 360 Game of the Year; ...	98	7.9
...	...	...	...	...	...	...
18795	Fast & Furious: Showdown	Xbox 360	May 21, 2013	Fast & Furious: Showdown takes some of the fra...	22	1.3
18796	Drake of the 99 Dragons	Xbox	November 3, 2003	Drake is out for revenge in a supernatural Hon...	22	1.7
18797	Afro Samurai 2: Revenge of Kuma Volume One	PlayStation 4	September 22, 2015	Head out on a journey of redemption, driven by...	21	2.9
18798	Infestation: Survivor Stories (The War Z)	PC	October 15, 2012	(Formerly known as "The War Z") It has been 5 ...	20	1.7
18799	Leisure Suit Larry: Box Office Bust	PC	March 31, 2009	The Leisure Suit Larry: Box Office Bust video ...	20	2

18800 rows × 6 columns

Fig. 8 visualización data set juegos.

A continuación, se realizó la modificación de la columna fecha a un formato especifico para evitar un error en la inserción de los datos a MySQL.

```python
[14]: fechas=df['release_date'] # Guardamos en una variable la columna de las fechas
      fechas

[14]: 0        November 23, 1998
      1       September 20, 2000
      2           April 29, 2008
      3        September 8, 1999
      4           April 29, 2008
                       ...
      18795         May 21, 2013
      18796     November 3, 2003
      18797   September 22, 2015
      18798      October 15, 2012
      18799        March 31, 2009
      Name: release_date, Length: 18800, dtype: object

[18]: fechas=pd.to_datetime(fechas) #Convertimos las fechas en un formato de tipo "Date"

[19]: fechas=fechas.dt.strftime('%Y-%m-%d') # Cambiamos la fecha al formato establecido

[20]: df['release_date']=fechas #Aplicamos el nuevo formato a la columna de fecha de nuestro data frame

[21]: df
```

Fig. 9 modificación de la columna fecha.

En la figura 9 guardamos en una variable la columna de fecha y la convertimos a un formato de tipo date y la cambiamos a un formato establecido y aplicamos el nuevo formato a nuestro data frame.

[21]:

	name	platform	release_date	summary	meta_score	user_review
0	The Legend of Zelda: Ocarina of Time	Nintendo 64	1998-11-23	As a young boy, Link is tricked by Ganondorf, ...	99	9.1
1	Tony Hawk's Pro Skater 2	PlayStation	2000-09-20	As most major publishers' development efforts ...	98	7.4
2	Grand Theft Auto IV	PlayStation 3	2008-04-29	[Metacritic's 2008 PS3 Game of the Year; Also ...	98	7.7
3	SoulCalibur	Dreamcast	1999-09-08	This is a tale of souls and swords; transcendi...	98	8.4
4	Grand Theft Auto IV	Xbox 360	2008-04-29	[Metacritic's 2008 Xbox 360 Game of the Year; ...	98	7.9
...	...	...	...	...	...	...
18795	Fast & Furious: Showdown	Xbox 360	2013-05-21	Fast & Furious: Showdown takes some of the fra...	22	1.3
18796	Drake of the 99 Dragons	Xbox	2003-11-03	Drake is out for revenge in a supernatural Hon...	22	1.7
18797	Afro Samurai 2: Revenge of Kuma Volume One	PlayStation 4	2015-09-22	Head out on a journey of redemption, driven by...	21	2.9
18798	Infestation: Survivor Stories (The War Z)	PC	2012-10-15	(Formerly known as "The War Z") It has been 5 ...	20	1.7
18799	Leisure Suit Larry: Box Office Bust	PC	2009-03-31	The Leisure Suit Larry: Box Office Bust video ...	20	2

18800 rows × 6 columns

Fig. 10 resultados cambio de formato fecha.

Continuaremos con la inserción de los data frame en MySQL.

```python
[22]: df.to_csv('JuegoL1.csv') # Transformamos el data frame a un archivo CSV con los nuevos cambios

[43]: enviar_datos=create_engine('mysql+pymysql://root:Tsuki1904@localhost/DLVideojuegos1')
      df.to_sql('datosxd',con=enviar_datos, if_exists='replace',index=False) # Enviamos el data frame a MySQL

[43]: 18800

[ ]:
```

Fig. 11 inserción en MySQL.

En la figura 11 transformaremos nuestro data frame en un archivo csv y a continuación enviamos ese data frame a MySQL.

La siguiente inserción de los data set será a MySQL Lite.

```python
#DATASET4 A SQLITE
# Carga tu dataset en un DataFrame
data = pd.read_csv('JuegosL3.csv')

# Crea una conexión a la base de datos SQLite
conn = sqlite3.connect('JuegosL3.db')

# Escribe el DataFrame a una tabla en SQLite
data.to_sql('Datos', conn, if_exists='replace', index=False)

# No olvides cerrar la conexión
conn.close()
```

Fig. 12 inserción en MySQL Lite.

También realizamos la inserción de los data sets en SQL server.

```python
import pandas as pd
import sqlalchemy
import urllib
import pyodbc

df = pd.read_csv('ConciertosL1.csv')

# Conectar a SQL Server con autenticación de Windows
params = urllib.parse.quote_plus("DRIVER={ODBC Driver 17 for SQL Server};SERVER=DESKTOP-9KJETOV;DATABASE=ANALISIS;Trusted_Connection=yes;")
engine = sqlalchemy.create_engine("mssql+pyodbc:///?odbc_connect=%s" % params)

# Enviar el DataFrame a SQL Server
df.to_sql('CONCIERTOS', con=engine, if_exists='append', index=False)
```

Fig. 13 inserción en SQL server.

En el siguiente vamos paso vamos a realizar un web scraping con los temas seleccionados para la obtención de datos estructurados para su posterior visualización.

```
const cheerio = require("cheerio");
const axios = require("axios");
const j2cp = require("json2csv").Parser;
const fs = require("fs");

let baseUrl = "https://www.metacritic.com/browse/movie/?releaseYearMin=1910&releaseYearMax=2024&page=";

let movies = [];
let num_pags = 2;

async function getMovies(url) {
  try {
    const response = await axios.get(url);
    const $ = cheerio.load(response.data);
    const movie = $(".c-finderProductCard_info");

    movie.each(function () {
      const spans = $(this).find(".c-finderProductCard_titleHeading");
      const id = $(spans[0]).text();
      const title = $(spans[1]).text();
      const date = $(this).find(".u-text-uppercase").text().trim();
      const description = $(this).find(".c-finderProductCard_description span").text();
      const metascore = $(this).find(".c-siteReviewScore span").text();

      movies.push({ id, title, date, description, metascore });
    });

    if (num_pags < 669) {
      const next_page = baseUrl + num_pags;
      getMovies(next_page);
      num_pags += 1;

      console.log(next_page);
    } else {
      const parser = new j2cp();
      const csv = parser.parse(movies);

      fs.writeFileSync("./moviesxd.csv", csv, "utf-8");
    }
  } catch (error) {
    console.error(error);
  }
}


getMovies("https://www.metacritic.com/browse/movie/?releaseYearMin=1910&releaseYearMax=2024&page=1");
```

Fig. 14 web scraping películas.

```
const cheerio = require("cheerio");
const axios = require("axios");
const j2cp = require("json2csv").Parser;
const fs = require("fs");

let baseUrl = "https://www.metacritic.com/browse/game/?releaseYearMin=1958&releaseYearMax=2024&page=";

let games = [];
let num_pags = 2;

async function getGames(url) {
  try {
    const response = await axios.get(url);
    const $ = cheerio.load(response.data);
    const game = $(".c-finderProductCard_info");

    game.each(function () {
      const spans = $(this).find(".c-finderProductCard_titleHeading span");
      const id = $(spans[0]).text();
      const title = $(spans[1]).text();
      const date = $(this).find(".u-text-uppercase").text().trim();
      const description = $(this).find(".c-finderProductCard_description span").text();
      const metascore = $(this).find(".c-siteReviewScore span").text();

      games.push({ id, title, date, description, metascore });
    });

    if (num_pags < 549) {
      const next_page = baseUrl + num_pags;
      getGames(next_page);
      num_pags += 1;

      console.log(next_page);
    } else {
      const parser = new j2cp();
      const csv = parser.parse(games);

      fs.writeFileSync("./equisde.csv", csv, "utf-8");
    }
  } catch (error) {
    console.error(error);
  }
}

getGames("https://www.metacritic.com/browse/game/?releaseYearMin=1958&releaseYearMax=2024&page=1");
```

Fig. 15 web scraping juegos.

Continuamos con la visualización y carga de nuestros archivos a nuestro respectivo Data lake

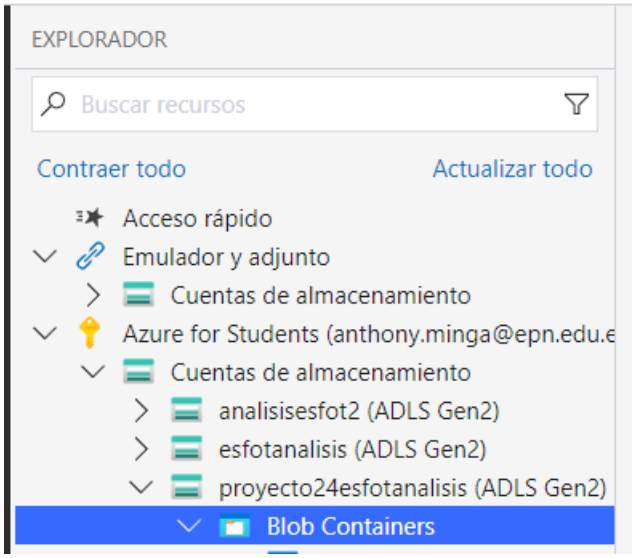

Fig. 16 Interfaz Data Lake.

Fig. 17 ejemplo de carga de archivos al Data Lake.

Como siguiente paso ingresaremos a SQL server para el envío de los data frame.

```
df = pd.read_csv('Conciertos1.csv')

# Conectar a SQL Server con autenticación de Windows
params = urllib.parse.quote_plus("DRIVER={ODBC Driver 17 for SQL Server};SERVER=DESKTOP-9CJETOV;DATABASE=CONCIERTOS;Trusted_Connection=yes;")
engine = sqlalchemy.create_engine("mssql+pyodbc:///?odbc_connect=%s" % params)

# Enviar el DataFrame a SQL Server
df.to_sql('CONCIERTOS1', con=engine, if_exists='append', index=False)
```

Fig. 18 código de conexión a SQL y envió de dataframe.

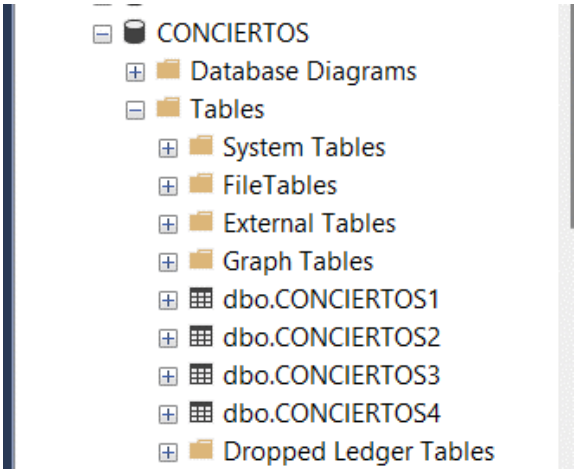

Fig. 19 ejemplo final de base de datos en SQL.

Para la conexión de Power BI con SQL server obtuvimos los siguientes pasos que se muestra en la figura 20.

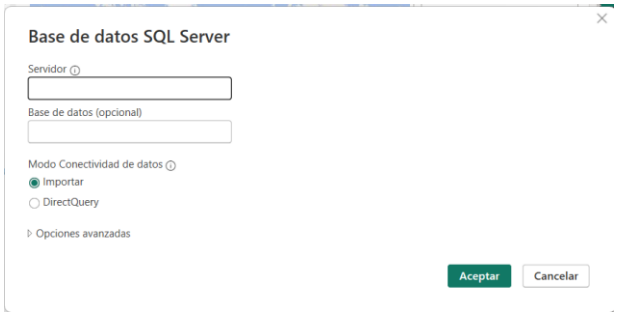

Fig. 20 conexión a Power BI.

En la figura 21 observaremos como con la conexión a nuestra base de datos ya visualizamos todos los dataframe que ya cargamos anteriormente y ahora solo seleccionaremos cada uno de ellos para realizar nuestras visualizaciones y dashboards correspondientes.

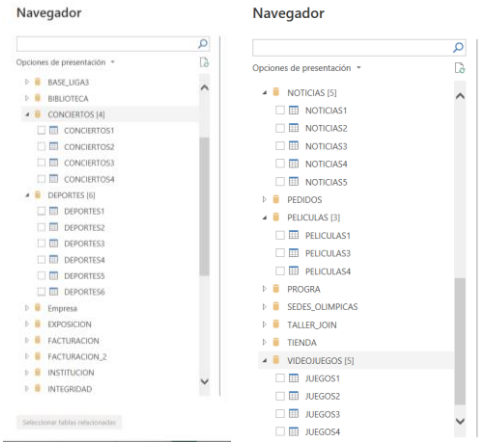

Fig. 21 carga de archivos en Power BI.

F. *Visualización de información*

En esta fase utilizamos la herramienta de Power BI para crear los dashboards interactivos y visualizaciones para representar de manera efectiva los resultados del análisis de cada uno de los casos de estudio para que tengan su propio dashboard.

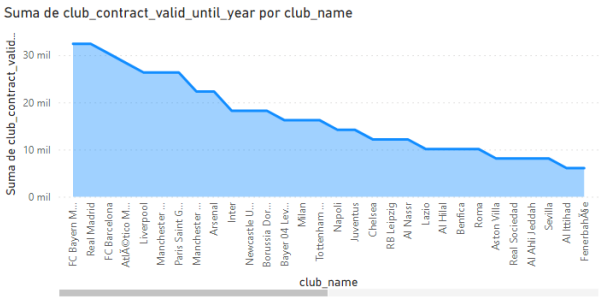

Fig. 22 ejemplo de dashboard en Power BI.

IV. RESULTADOS

A continuación, se presentará los resultados obtenidos a lo largo del proyecto.

Como resultado presentamos las carpetas con todos los archivos que se realizó la limpieza de datashet ordenados en sus respectivas carpetas de cada tema elegido.

Fig. 23 carpetas de archivos limpios.

Resultados de la carga de los archivos en nuestro Data Lake.

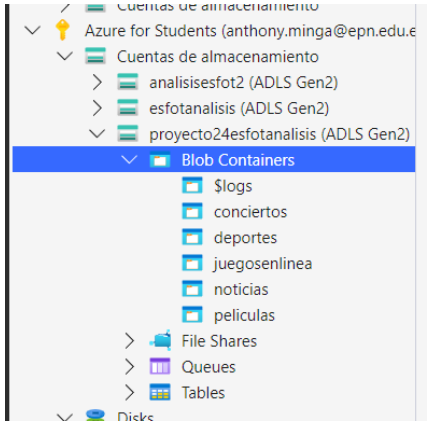

Fig. 24 carga de datos en Data Lake.

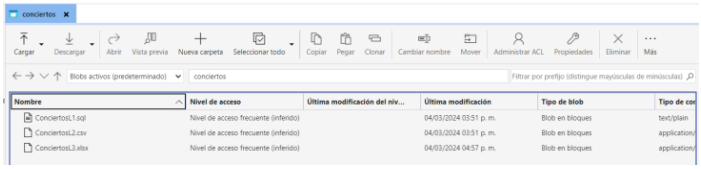

Fig. 25 carga de archivos conciertos en Data Lake.

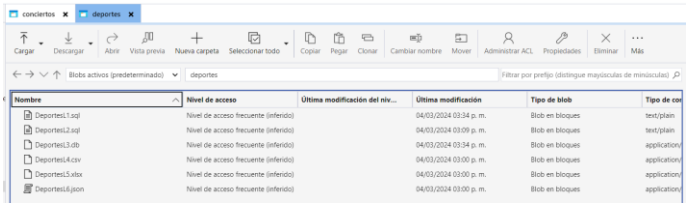

Fig. 26 carga de archivos deportes en Data Lake.

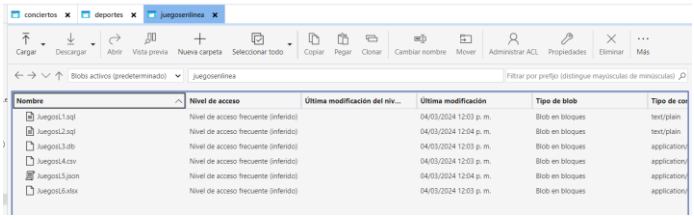

Fig. 27 carga de archivos juegos en Data Lake.

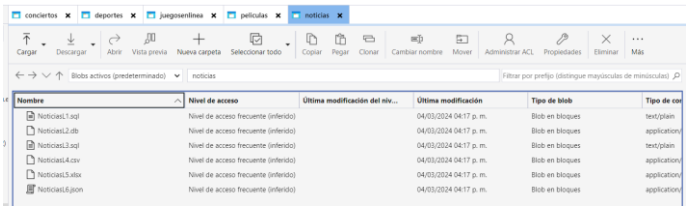

Fig. 28 carga de archivos noticias en Data Lake.

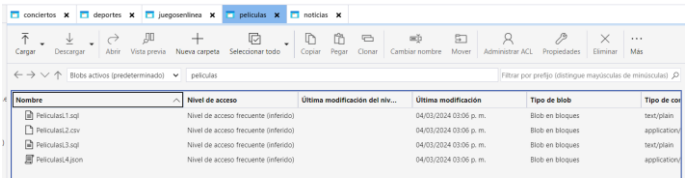

Fig. 29 carga de archivos películas en Data Lake.

Resultados de carga de dataframe en nuestra base de datos SQL server.

Fig. 30 carga de dataframe conciertos a SQL server.

Fig. 31 carga de dataframe conciertos a SQL server.

Fig. 32 carga de dataframe conciertos a SQL server.

Fig. 33 carga de dataframe conciertos a SQL server.

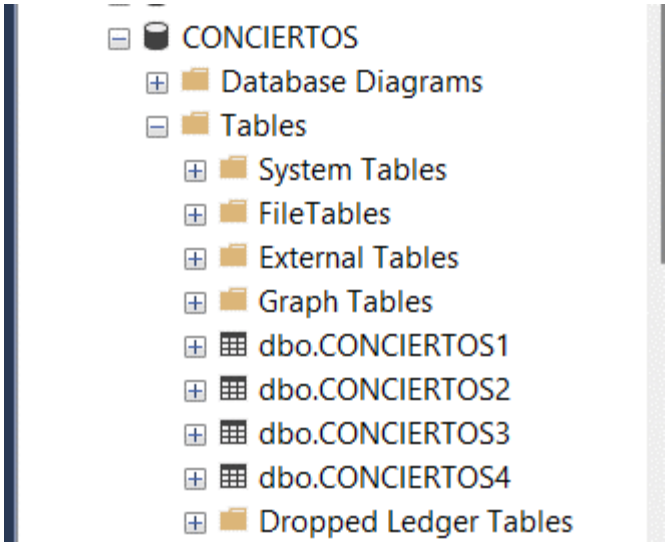

Fig. 34 Base de datos conciertos.

Fig. 35 carga de dataframe deportes a SQL server.

Fig. 36 carga de dataframe deportes a SQL server.

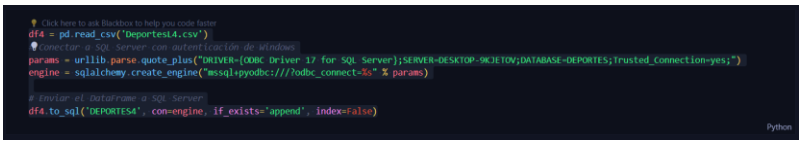

Fig. 37 carga de dataframe deportes a SQL server.

Fig. 38 carga de dataframe deportes a SQL server.

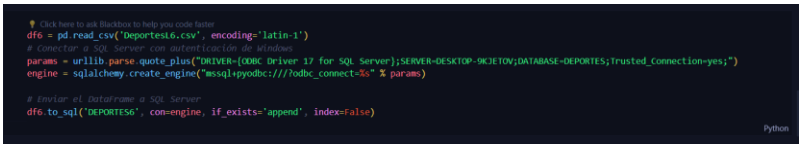

Fig. 39 carga de dataframe deportes a SQL server.

Fig. 40 carga de dataframe deportes a SQL server.

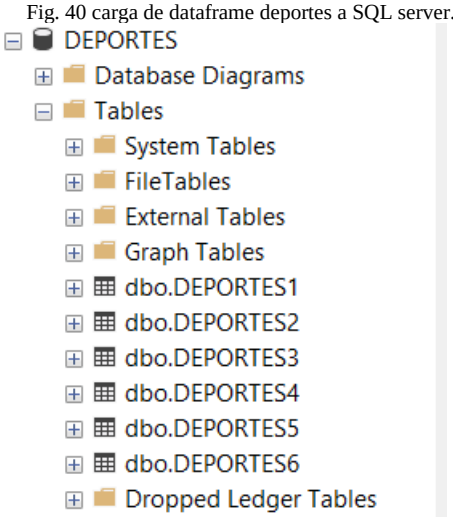

Fig. 50 Base de datos deportes.

Fig. 51 carga de dataframe noticias a SQL server.

Fig. 52 carga de dataframe noticias a SQL server.

Fig. 53 carga de dataframe noticias a SQL server.

```python
df4 = pd.read_csv('NoticiasL4.csv')
# Conectar a SQL Server con autenticación de Windows
params = urllib.parse.quote_plus("DRIVER={ODBC Driver 17 for SQL Server};SERVER=DESKTOP-9KJETOV;DATABASE=NOTICIAS;Trusted_Connection=yes;")
engine = sqlalchemy.create_engine("mssql+pyodbc:///?odbc_connect=%s" % params)

# Enviar el DataFrame a SQL Server
df4.to_sql('NOTICIAS4', con=engine, if_exists='append', index=False)
```

Fig. 54 carga de dataframe noticias a SQL server.

```python
df5 = pd.read_csv('NoticiasL5.csv')
# Conectar a SQL Server con autenticación de Windows
params = urllib.parse.quote_plus("DRIVER={ODBC Driver 17 for SQL Server};SERVER=DESKTOP-9KJETOV;DATABASE=NOTICIAS;Trusted_Connection=yes;")
engine = sqlalchemy.create_engine("mssql+pyodbc:///?odbc_connect=%s" % params)

# Enviar el DataFrame a SQL Server
df5.to_sql('NOTICIAS5', con=engine, if_exists='append', index=False)
```

Fig. 55 carga de dataframe noticias a SQL server.

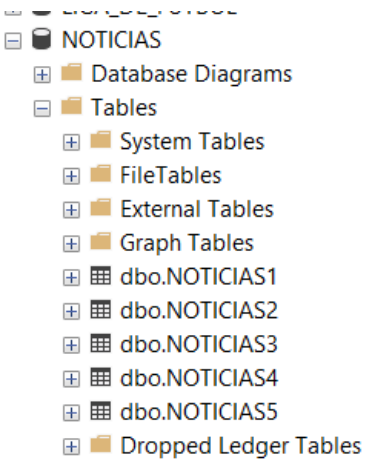

Fig. 56 Base de datos noticias.

```python
df1 = pd.read_csv('JuegosL1.csv')
# Conectar a SQL Server con autenticación de Windows
params = urllib.parse.quote_plus("DRIVER={ODBC Driver 17 for SQL Server};SERVER=DESKTOP-9KJETOV;DATABASE=VIDEOJUEGOS;Trusted_Connectio
engine = sqlalchemy.create_engine("mssql+pyodbc:///?odbc_connect=%s" % params)

# Enviar el DataFrame a SQL Server
df1.to_sql('JUEGOS1', con=engine, if_exists='append', index=False)
```

Fig. 57 carga de dataframe juegos a SQL server.

```python
df2 = pd.read_csv('JuegosL2.csv')
# Conectar a SQL Server con autenticación de Windows
params = urllib.parse.quote_plus("DRIVER={ODBC Driver 17 for SQL Server};SERVER=DESKTOP-9KJETOV;DATABASE=VIDEOJUEGOS;Trusted_Connect
engine = sqlalchemy.create_engine("mssql+pyodbc:///?odbc_connect=%s" % params)

# Enviar el DataFrame a SQL Server
df2.to_sql('JUEGOS2', con=engine, if_exists='append', index=False)
```

Fig. 58 carga de dataframe juegos a SQL server.

```python
df3 = pd.read_csv('JuegosL3.csv')
# Conectar a SQL Server con autenticación de Windows
params = urllib.parse.quote_plus("DRIVER={ODBC Driver 17 for SQL Server};SERVER=DESKTOP-9KJETOV;DATABASE=VIDEOJUEGOS;Trusted_Conne
engine = sqlalchemy.create_engine("mssql+pyodbc:///?odbc_connect=%s" % params)

# Enviar el DataFrame a SQL Server
df3.to_sql('JUEGOS3', con=engine, if_exists='append', index=False)
```

Fig. 59 carga de dataframe juegos a SQL server.

```python
df4 = pd.read_csv('JuegosL4.csv')
# Conectar a SQL Server con autenticación de Windows
params = urllib.parse.quote_plus("DRIVER={ODBC Driver 17 for SQL Server};SERVER=DESKTOP-9KJETOV;DATABASE=VIDEOJUEGOS;Trusted_Connection=yes;")
engine = sqlalchemy.create_engine("mssql+pyodbc:///?odbc_connect=%s" % params)

# Enviar el DataFrame a SQL Server
df4.to_sql('JUEGOS4', con=engine, if_exists='append', index=False)
```

Fig. 60 carga de dataframe juegos a SQL server.

```python
df6 = pd.read_csv('JuegosL6.csv')
# Conectar a SQL Server con autenticación de Windows
params = urllib.parse.quote_plus("DRIVER={ODBC Driver 17 for SQL Server};SERVER=DESKTOP-9KJETOV;DATABASE=VIDEOJUEGOS;Trusted_Connect
engine = sqlalchemy.create_engine("mssql+pyodbc:///?odbc_connect=%s" % params)

# Enviar el DataFrame a SQL Server
df6.to_sql('JUEGOS6', con=engine, if_exists='append', index=False)
```

Fig. 61 carga de dataframe juegos a SQL server.

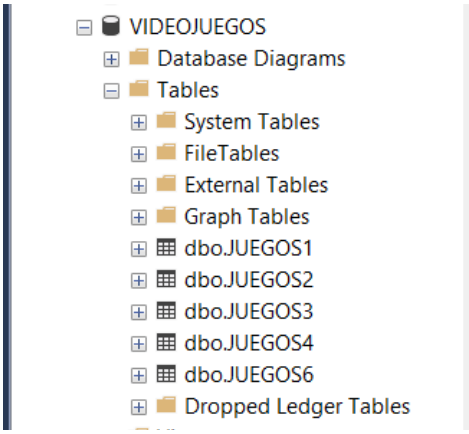

Fig. 62 Base de datos juegos.

```python
df = pd.read_csv('PeliculasL1.csv')
# Conectar a SQL Server con autenticación de Windows
params = urllib.parse.quote_plus("DRIVER={ODBC Driver 17 for SQL Server};SERVER=DESKTOP-9KJETOV;DATABASE=PELICULAS;Trusted_Connection=yes;")
engine = sqlalchemy.create_engine("mssql+pyodbc:///?odbc_connect=%s" % params)

# Enviar el DataFrame a SQL Server
df.to_sql('PELICULAS1', con=engine, if_exists='append', index=False)
```

Fig. 63 carga de dataframe películas a SQL server.

```python
df2 = pd.read_csv('PeliculasL3.csv')
# Conectar a SQL Server con autenticación de Windows
params = urllib.parse.quote_plus("DRIVER={ODBC Driver 17 for SQL Server};SERVER=DESKTOP-9KJETOV;DATABASE=PELICULAS;Trusted_Connection=yes;")
engine = sqlalchemy.create_engine("mssql+pyodbc:///?odbc_connect=%s" % params)

# Enviar el DataFrame a SQL Server
df2.to_sql('PELICULAS3', con=engine, if_exists='append', index=False)
```

Fig. 64 carga de dataframe películas a SQL server.

```python
df3 = pd.read_csv('PeliculasL4.csv')
# Conectar a SQL Server con autenticación de Windows
params = urllib.parse.quote_plus("DRIVER={ODBC Driver 17 for SQL Server};SERVER=DESKTOP-9KJETOV;DATABASE=PELICULAS;Trusted_Connection=yes;")
engine = sqlalchemy.create_engine("mssql+pyodbc:///?odbc_connect=%s" % params)

# Enviar el DataFrame a SQL Server
df3.to_sql('PELICULAS4', con=engine, if_exists='append', index=False)
```

Fig. 65 carga de dataframe películas a SQL server.

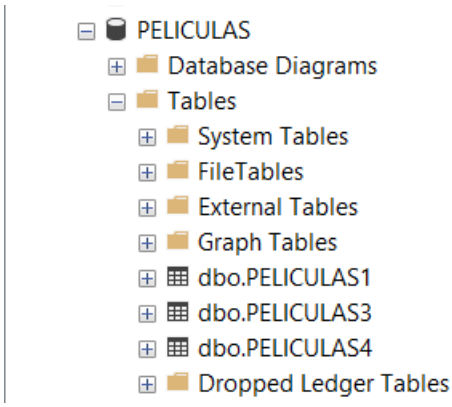

Fig. 62 Base de datos películas.

Resultados de las visualizaciones de cada tema seleccionado en la herramienta Power BI.

CONCIERTOS

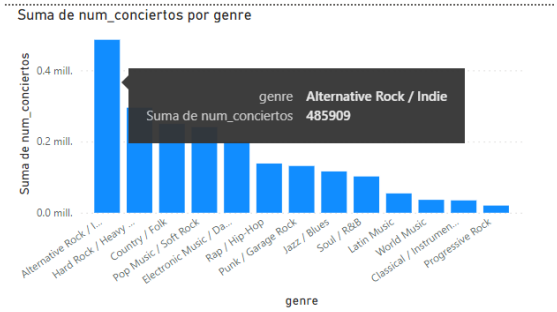

Fig. 63 visualización de conciertos en Power BI

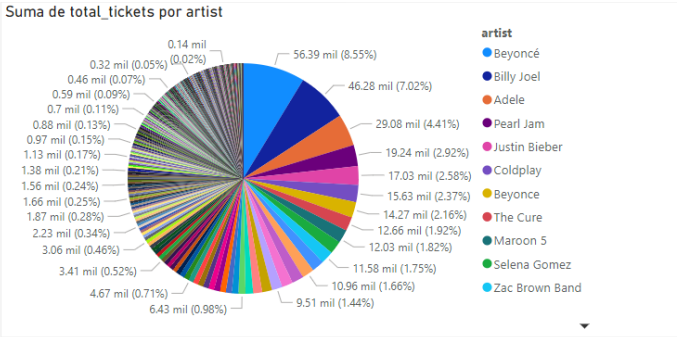

Fig. 64 visualización de conciertos en Power BI

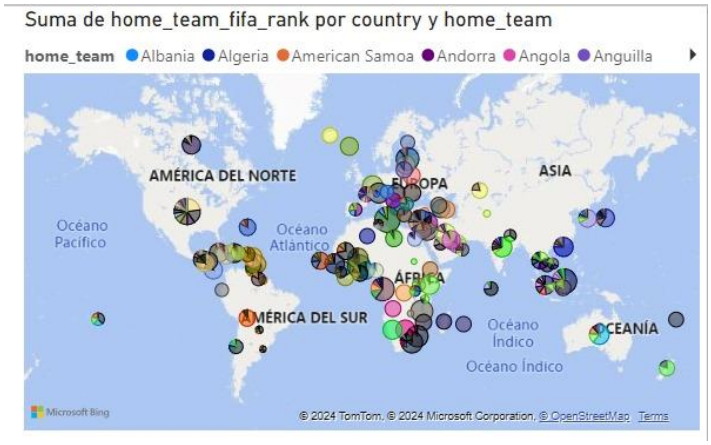

Fig. 65 visualización de conciertos en Power BI

DEPORTES

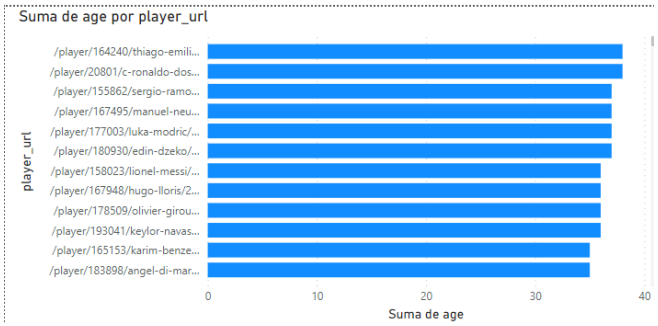

Fig. 66 visualización de deportes en Power BI

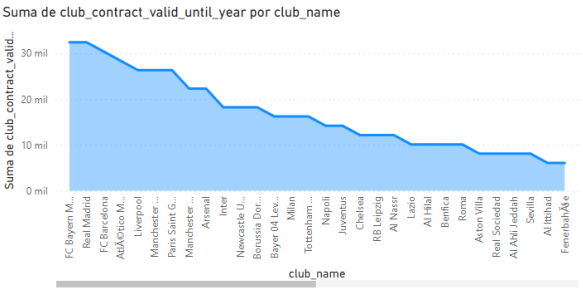

Fig. 67 visualización de deportes en Power BI

JUEGOS

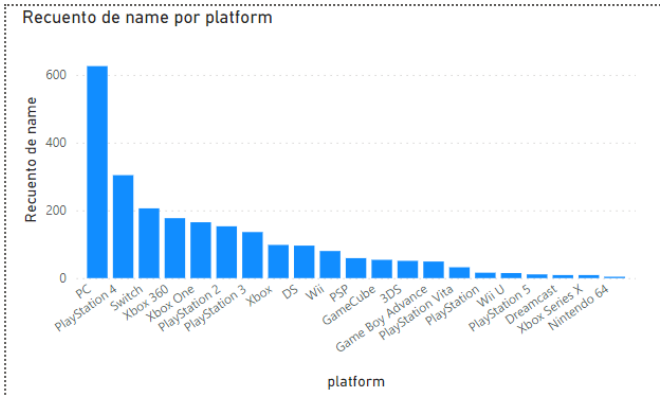

Fig.68 visualización de juegos en Power BI

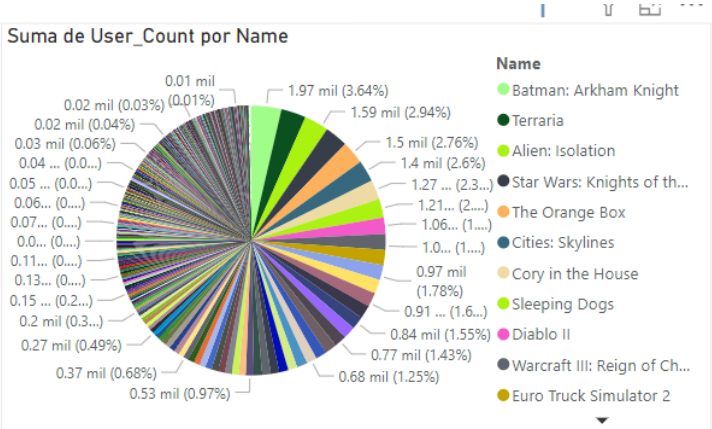

Fig. 69 visualización de juegos en Power BI

Fig. 70 visualización de juegos en Power BI

NOTICIAS

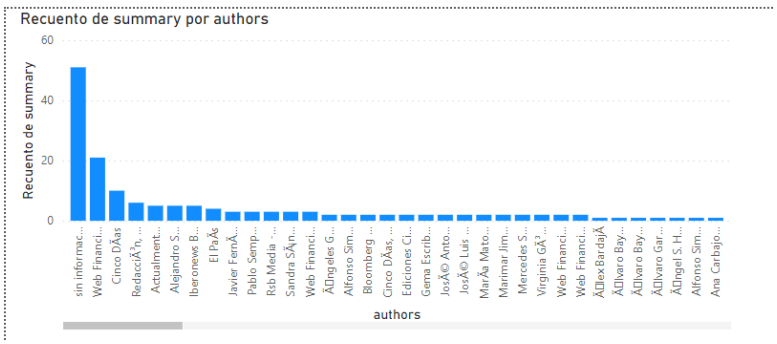

Fig. 71 visualización de noticias en Power BI

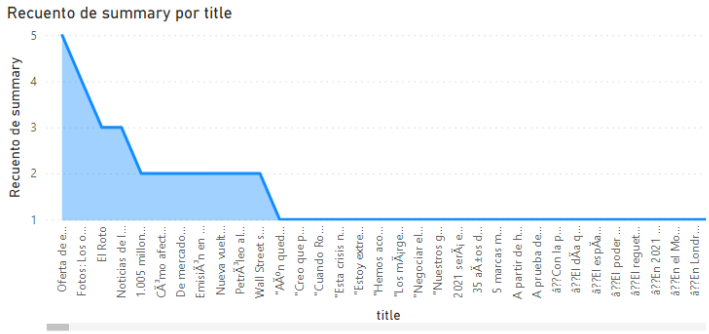

Fig. 72 visualización de noticias en Power BI

PELICULAS

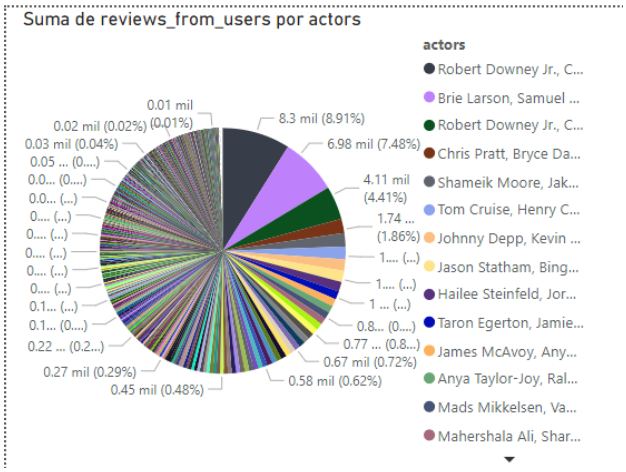

Fig. 73 visualización de conciertos en Power BI

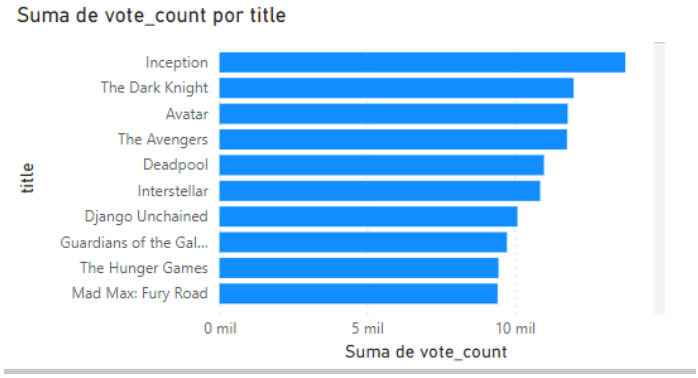

Fig. 74 visualización de conciertos en Power BI

V. CONCLUSIONES Y RECOMENDACIONES

Conclusiones

- La arquitectura y diseño de Data Lake demostró ser eficaz en la integración de datos provenientes de diversas fuentes, la diversidad para la extracción de datos permitió una recopilación completa y actualizada.
- Power BI como herramienta de visualización de datos proporciono dashboards interactivos y comprensibles para cada caso de estudio, también la gráfica facilito la identificación de patrones mejorando así la interpretación de los resultados.
- Durante el desarrollo del proyecto se identificaron áreas de mejora en la documentación de procesos. Esto resalta la importancia de mantener una documentación detallada para facilitar la comprensión del sistema.

Recomendaciones

- Se recomienda una planificación detallada y la gestión efectiva del tiempo son cruciales para cumplir con los plazos establecidos. Delegar tareas específicas ayudará a evitar retrasos y garantizará un progreso constante a lo largo del proyecto.
- La documentación completa de cada etapa del proyecto es esencial ya que facilita la evaluación del trabajo, permite a otros entender el proceso y respalda las decisiones tomadas.

VI. DESAFIOS Y PROBLEMAS ENCONTRADOS

- Manejar la diversidad de formatos y estructuras de datos provenientes de las diferentes fuentes presentó un desafío significativo. La heterogeneidad requería una adaptación cuidadosa durante la fase de extracción.
- Configurar PowerBI para generar dashboards específicos para cada caso de estudio y asegurar una conexión estable con el concentrador de datos presentó un desafío técnico. La optimización de las visualizaciones y la presentación de datos geolocalizados requirió un tiempo significativo de ajuste.
- El proyecto enfrentó problemas relacionados con la eficiencia en la manipulación de grandes volúmenes de datos. La optimización de los procesos y la gestión de recursos se volvieron críticos para mantener un rendimiento aceptable en la integración y análisis de datos.
- La gestión efectiva del tiempo se convirtió en un problema, especialmente al enfrentar la complejidad del proyecto y la necesidad de cumplir con plazos establecidos. La distribución adecuada del tiempo entre las distintas fases del proyecto se volvió un aspecto clave.

VII. LINKS

Visualizaciones Power BI

- ✓ https://app.powerbi.com/view?r=eyJrIjoiNzM4YzQxMzEtY2Q0ZC00MjRhLWI4MzQtYmJjMmI4MjlkZDZlIiwidCI6IjY4MmE0ZTZhLWE3N2YtNDk1OC1hM2FjLTllMjY2ZDE4YWEzNyIsImMiOjR9
- ✓ https://app.powerbi.com/view?r=eyJrIjoiNzM4YzQxMzEtY2Q0ZC00MjRhLWI4MzQtYmJjMmI4MjlkZDZlIiwidCI6IjY4MmE0ZTZhLWE3N2YtNDk1OC1hM2FjLTllMjY2ZDE4YWEzNyIsImMiOjR9

GitHub

- ✓ https://github.com/Dustin1904/Proyecto_Final_Analisis_de_Datos.git

YouTube

- ✓ https://youtu.be/I9tHsws90HM

REFERENCIAS

[1] "Sports Reference | Sports Stats, fast, easy, and up-to-date | Sports-Reference.com". Sports-Reference.com. Accedido el 5 de marzo de 2024. [En línea]. Disponible: https://www.sports-reference.com/

[2] "ESPN - Serving Sports Fans. Anytime. Anywhere." ESPN.com. Accedido el 5 de marzo de 2024. [En línea]. Disponible: https://www.espn.com/analytics/

[3] "Data.gov Home - Data.gov". Data.gov. Accedido el 5 de marzo de 2024. [En línea]. Disponible: https://data.gov/

[4] "Steam Community :: Steam Web API Documentation". Steam Community. Accedido el 5 de marzo de 2024. [En línea]. Disponible: https://steamcommunity.com/dev

[5] "Steamworks Web API Reference (Steamworks Documentation)". Steamworks. Accedido el 5 de marzo de 2024. [En línea]. Disponible: https://partner.steamgames.com/doc/webapi

[6] "Statista - El portal de estadísticas". Statista. Accedido el 5 de marzo de 2024. [En línea]. Disponible: https://es.statista.com/

[7] "Dataset Search". Dataset Search. Accedido el 5 de marzo de 2024. [En línea]. Disponible: https://datasetsearch.research.google.com/

[8] "GitHub: Let's build from here". GitHub. Accedido el 5 de marzo de 2024. [En línea]. Disponible: https://github.com/

[9] "Power BI: visualización de datos | Microsoft Power Platform". Microsoft Corporation. Accedido el 5 de marzo de 2024. [En línea]. Disponible: https://www.microsoft.com/es-es/power-platform/products/power-bi

[10] "EA Sports FC 24 complete player dataset". Kaggle: Your Machine Learning and Data Science Community. Accedido el 5 de marzo de 2024. [En línea]. Disponible: https://www.kaggle.com/datasets/stefanoleone992/ea-sports-fc-24-complete-player-dataset

[11] "Olympic Swimming History (1912 to 2020)". Kaggle: Your Machine Learning and Data Science Community. Accedido el 5 de marzo de 2024. [En línea]. Disponible: https://www.kaggle.com/datasets/datasciencedonut/olympic-swimming-1912-to-2020

[12] "FIFA World Cup 2022 ⚽🏆". Kaggle: Your Machine Learning and Data Science Community. Accedido el 5 de marzo de 2024. [En línea]. Disponible: https://www.kaggle.com/datasets/brenda89/fifa-world-cup-2022

[13] "FIFA World Cup 2022 Catar". Kaggle: Your Machine Learning and Data Science Community. Accedido el 5 de marzo de 2024. [En línea]. Disponible: https://www.kaggle.com/datasets/mcarujo/fifa-world-cup-2022-catar

[14] "UCI Machine Learning Repository". UCI Machine Learning Repository. Accedido el 5 de marzo de 2024. [En línea]. Disponible: https://archive.ics.uci.edu/

[15] "Open Data Network". Open Data Network. Accedido el 5 de marzo de 2024. [En línea]. Disponible: https://www.opendatanetwork.com/

[16] "Data.gov.sg". Data.gov.sg. Accedido el 5 de marzo de 2024. [En línea]. Disponible: https://beta.data.gov.sg/datasets

[17] "Top Artists on Tour in 2024. Bands and Singers on Tour - Concertful". Concertful. Accedido el 5 de marzo de 2024. [En línea]. Disponible: https://concertful.com/top?page=

[18] "What Movies to Watch Right Now - Metacritic". Movie Reviews, TV Reviews, Game Reviews, and Music Reviews - Metacritic. Accedido el 5 de marzo de 2024. [En línea]. Disponible: https://www.metacritic.com/browse/movie/?releaseYearMin=1910&releaseYearMax=2024&page=

[19] "What Games to Play Right Now - Metacritic". Movie Reviews, TV Reviews, Game Reviews, and Music Reviews - Metacritic. Accedido el 5 de marzo de 2024. [En línea]. Disponible: https://www.metacritic.com/browse/game/?releaseYearMin=1958&releaseYearMax=2024&page=

[20] "Our Data". FiveThirtyEight. Accedido el 5 de marzo de 2024. [En línea]. Disponible: https://data.fivethirtyeight.com/