Informe de presentación de resultados

1. *Explicación de la arquitectura de datos y arquetipo de la aplicación*

La arquitectura del pipeline ETL se divide en las siguientes fases:

- Extracción:
  - Origen de datos: se cargan los datos desde un archivo Excel (Films_2 (4).xlsx)
  - Tecnología utilizada: El formato com.crealytics.spark.excel se utiliza con PySpark para leer las hojas del archivo Excel como DataFrames.
  - Propósito: Convertir datos estructurados de Excel en objetos procesables en memoria, capturando cada sheet del archivo en una tabla.
- Transformación:
  - Limpieza de nombres de columnas: se eliminan espacios en blanco de los nombres de columnas para evitar errores de consulta posteriores.
  - Estandarización de datos:
    - ✓ Conversión de tipos de datos: Columnas como film_id, rental_rate, last_update se convierten al tipo correcto.
    - ✓ Limpieza de valores no válidos: Columnas numéricas se filtran para dejar únicamente números y puntos para los decimales
    - ✓ A todas las columnas tipo string y derivados se eliminan espacios inicio-fin
    - ✓ Gestión de valores nulos: Los valores nulos se manejan según el tipo de dato: los números se rellenan con 0, las cadenas con "", y las fechas/timestamps con None
  - Eliminación de duplicados para cada tabla
- Carga:
  - Destino de los datos: Los DataFrames transformados se cargan en una base de datos PostgreSQL.
  - Tecnología utilizada: La integración con PostgreSQL se realiza mediante el método DataFrame.write.jdbc() de PySpark.
  - Orden de carga: Las tablas se cargan en un orden específico (store, film, customer, inventory, rental) para respetar las dependencias de claves foráneas.

El arquetipo es una aplicación ETL modular con estas caracteristicas:

- Modularidad: Cada paso del ETL está encapsulado en métodos específicos que facilita la reutilización y el mantenimiento: extract, transform y load
- Escalabilidad**:** EL uso de la biblioteca PySpark permite procesar grandes volúmenes de datos distribuidos en paralelo. Y el uso de SparkSession: Ofrece un punto centralizado para configurar y ejecutar tareas Spark.
- Gestión de dependencias: El pipeline respeta las relaciones entre tablas (como claves foráneas) al cargar los datos en un orden predeterminado. Esto garantiza que no se violen restricciones al insertar datos en la base de datos.
- Flexibilidad en la configuración: el pipeline tiene parámetros de entrada que permite espeficar dinámicamente las tablas, columnas y tipos de datos para transformación; asi como cambiar la base de datos de destino al modificar los parámetros db_url y db_properties.

2. *Análisis exploratorio de datos*

El origen de datos tiene las siguientes tablas:

- film: Información relacionada con las películas, contiene información que incluye título, descripción, año de lanzamiento, duración del alquiler, tarifas, longitud, clasificación, características especiales, etc.

algunos valores son inconsistentes, como release_year con formato de año "x2006"; también hay algunos registros duplicados

- inventory: inventario relacionado con las películas, relaciona películas con tiendas mediante film_id y store_id, en la columna store_id hay algunos datos inconsistentes como “2*$#"
- rental: datos sobre alquileres, registra alquileres, con fechas de renta y devolución, IDs de cliente y personal, y un identificador de inventario
- customer: información de los clientes, contiene datos nombres, correos electrónicos, segmento de mercado y fechas de registro. Algunos valores son "NULL"
- store: detalles de las tiendas

3. *Preguntas de negocio*

- ¿Cuál es el top 5 de películas más alquiladas?

```
title              |total_rentals|
-------------------+-------------+
BUCKET BROTHERHOOD |           34|
ROCKETEER MOTHER   |           33|
RIDGEMONT SUBMARINE|           32|
SCALAWAG DUCK      |           32|
FORWARD TEMPLE     |           32|
```

Query:

SELECT f.title, COUNT(r.rental_id) AS total_rentals
FROM rental r
JOIN inventory i ON r.inventory_id = i.inventory_id
JOIN film f ON i.film_id = f.film_id
GROUP BY f.title
ORDER BY total_rentals DESC
LIMIT 5;

- ¿Cual es el top 10 de clientes que mas alquilan peliculas?

```
customer_id|first_name|last_name|total_rentals|
-----------+----------+---------+-------------+
        148|ELEANOR   |HUNT     |           46|
        526|KARL      |SEAL     |           45|
        236|MARCIA    |DEAN     |           42|
        144|CLARA     |SHAW     |           42|
         75|TAMMY     |SANDERS  |           41|
        197|SUE       |PETERS   |           40|
        469|WESLEY    |BULL     |           40|
        137|RHONDA    |KENNEDY  |           39|
        178|MARION    |SNYDER   |           39|
        468|TIM       |CARY     |           39|
```

Query:

SELECT c.customer_id, c.first_name, c.last_name, COUNT(r.rental_id) AS total_rentals
FROM rental r
JOIN customer c ON r.customer_id = c.customer_id
GROUP BY c.customer_id, c.first_name, c.last_name
ORDER BY total_rentals DESC
LIMIT 10;

- ¿Qué tiendas generan más ingresos por alquiler?

```
store_id|total_revenue|
--------+-------------+
       2|     47199.68|
       1|        11.88|
```

Query:

SELECT s.store_id, SUM(f.rental_rate) AS total_revenue

```
FROM rental r
JOIN inventory i ON r.inventory_id = i.inventory_id
JOIN film f ON i.film_id = f.film_id
JOIN store s ON i.store_id = s.store_id
GROUP BY s.store_id
ORDER BY total_revenue DESC;
```

- Cual es el top 5 de peliculas con mas votos por los clientes?

```
title                |num_voted_users|
---------------------+---------------+
GOLDFINGER SENSIBILITY|          76900|
BREAKING HOME        |          76850|
SHOW LORD            |          76750|
ACADEMY DINOSAUR     |          76750|
DETAILS PACKER       |          76500|
```

Query:

```
SELECT f.title, f.num_voted_users
FROM film f
WHERE f.num_voted_users IS NOT NULL
ORDER BY f.num_voted_users DESC
LIMIT 5;
```

- ¿Cuánto tiempo tardan en promedio los clientes en devolver las películas alquiladas?

```
avg_hours           |
--------------------+
120.6079282516865267|
```

Query:

```
SELECT AVG(EXTRACT(EPOCH FROM (return_date - rental_date)) / 3600) AS avg_hours
FROM rental
WHERE return_date IS NOT NULL;
```

4. *Conclusiones*

Con la realización de esta prueba enfatizo la importancia de la limpieza de datos, como la eliminación de duplicados, manejo de nulos y transformación de tipos de datos; ya que sin estos ajustes no se podia mapear el MER en la base de datos; es una etapa crítica para garantizar la calidad y consistencia de los datos cargados en el destino.

Tambien recalco la importancia de la carga secuencial, ordenada por las dependencias entre tablas, con esto el pipeline permite manejar correctamente las restricciones de integridad referencial.