

Proyecto de Profundización Estructura de Datos

Este proyecto tiene por objetivo diseñar e implementar un sistema ETL eficiente que permita a Superstore a gestionar grandes volúmenes de datos dispersos, optimizando el almacenamiento y la estructura de los datos para facilitar su análisis. A través de la creación de un sistema jerárquico con tablas de hechos y dimensiones.



by **Jaqueline Mera**



ETL

1

Extracción

Consiste en recolectar datos de diferentes fuentes, como bases de datos, archivos o APIs.

2

Transformación

Implica convertir, limpiar y estructurar los datos según las reglas y necesidades del negocio.

3

Carga

Inserción de los datos transformados en un sistema de almacenamiento, como un data warehouse.

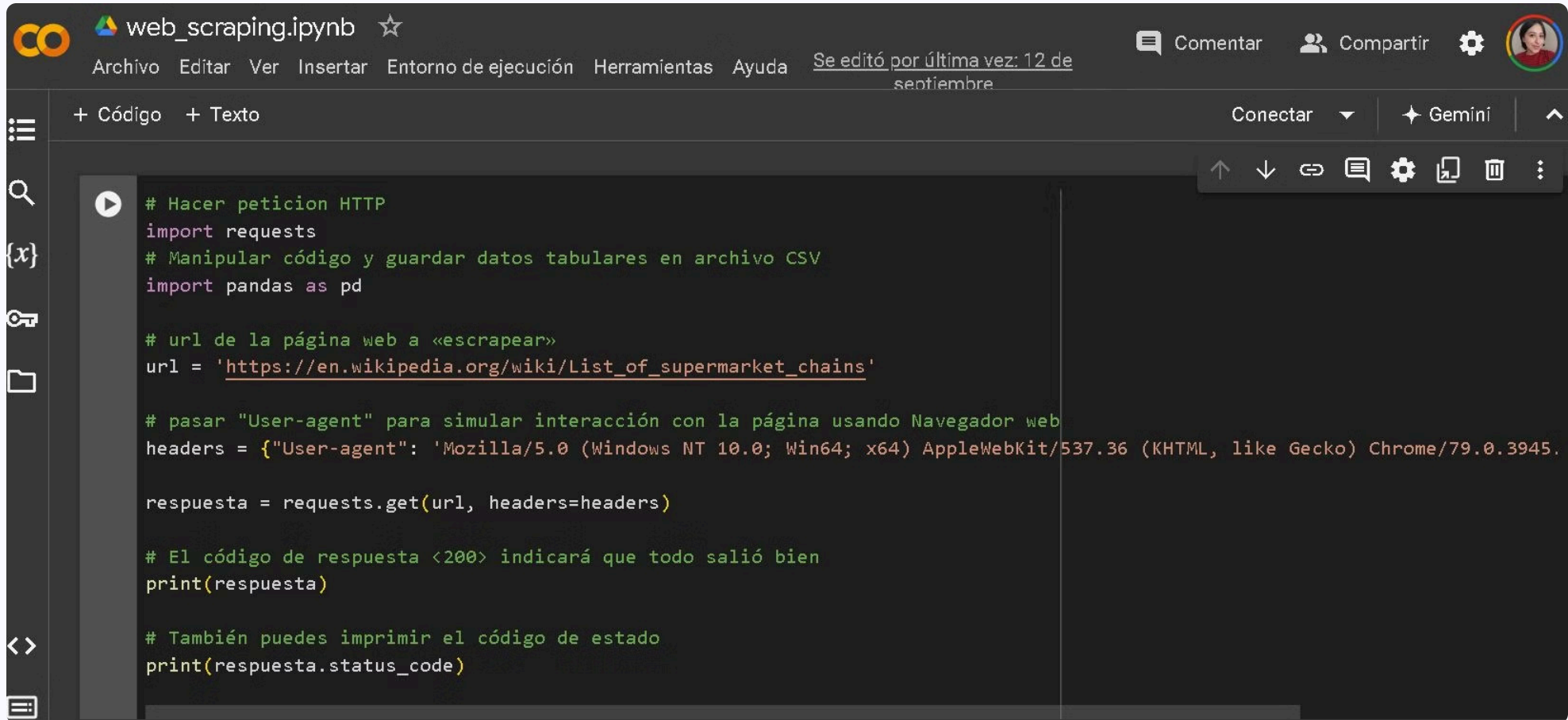
4

Análisis

Utilizando herramientas como dashboards para visualizar patrones, tendencias y métricas clave que facilitan la toma de decisiones estratégicas.

Extracción: Flat table Superstore y Web scraping

La información se extrajo de un archivo csv (Flat table de Superstore), además se realizó Web scraping para obtener información de sitios web de la competencia. Se utilizó el paquete Beautiful Soup de Python en Google Colab. Se obtuvo una tabla de las cadenas de supermercados multinacionales de Wikipedia.



```
# Hacer petición HTTP
import requests
# Manipular código y guardar datos tabulares en archivo CSV
import pandas as pd

# url de la página web a «escraper»
url = 'https://en.wikipedia.org/wiki/List_of_supermarket_chains'

# pasar "User-agent" para simular interacción con la página usando Navegador web
headers = {"User-agent": 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/79.0.3945.'}

respuesta = requests.get(url, headers=headers)

# El código de respuesta <200> indicará que todo salió bien
print(respuesta)

# También puedes imprimir el código de estado
print(respuesta.status_code)
```

Transformación: Procesamiento y limpieza

El primer paso consistió en importar los datos de Superstore a una tabla dentro del ambiente de BigQuery. Se identificaron y manejaron valores nulos y duplicados, utilizando comandos SQL, así como la creación de variables.

1 Identificar Valores Nulos

Se utilizaron comandos SQL, COUNTIF y IS NULL, para identificar valores nulos en la tabla de Superstore. No se encontraron valores nulos.

2 Identificar Valores Duplicados

Se buscaron valores duplicados utilizando comandos SQL, COUNT, GROUP BY, y HAVING. No se encontraron duplicados en cada transacción, pero se identificó que el order_id se repite en diferentes clientes (customer_id).

3 Comprobar Tipo de Dato

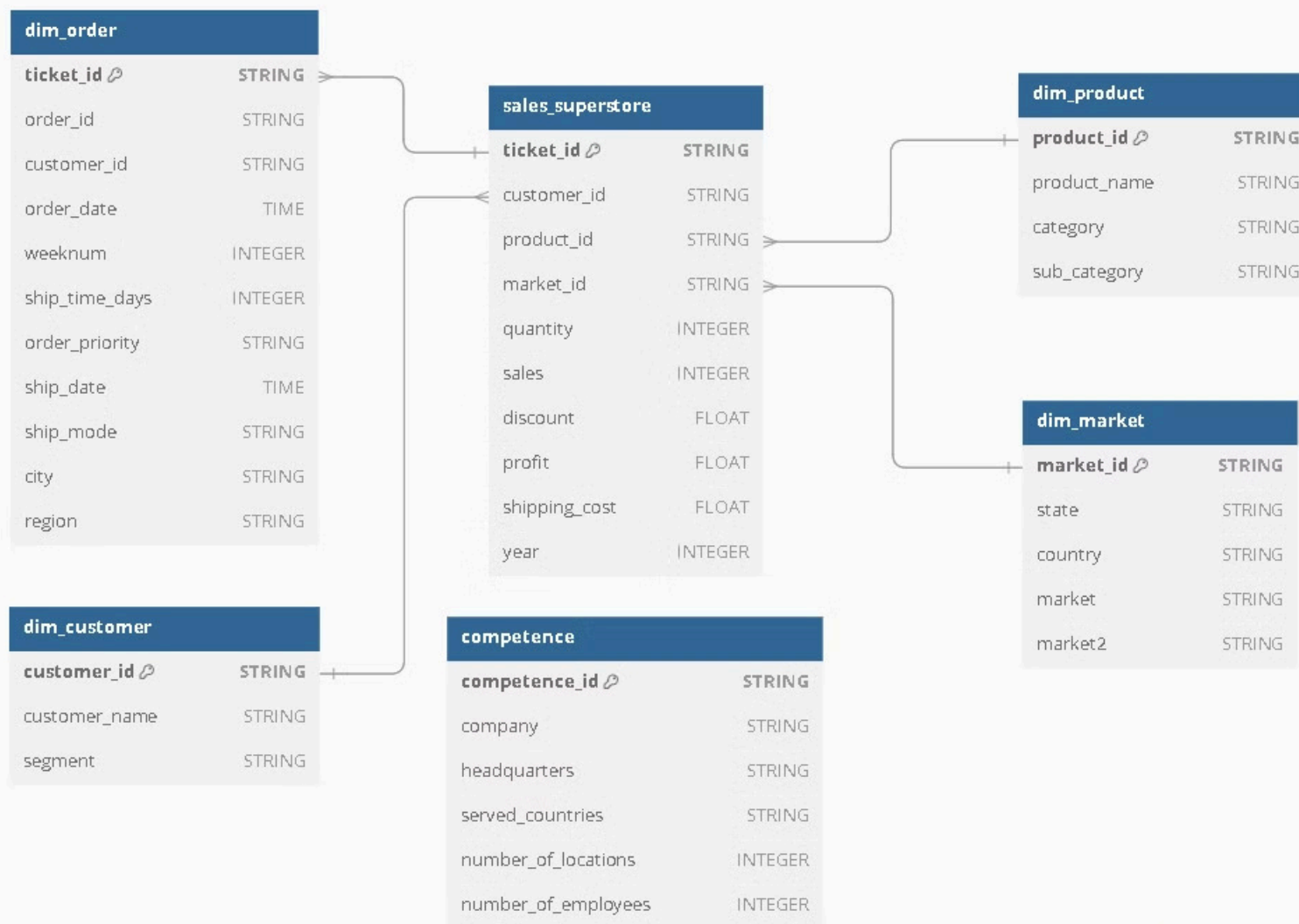
Se cambió el tipo de dato de las columnas order_date y ship_date de formato TIMESTAMP a formato DATE.

4 Creación de Nuevas Variables

Creación de nuevos Id's, ticket_id, concatenación de order_id y customer_id, para generar un id único. market_id concatenación de estado y país, para generar un Id local para cada mercado. competence_id, clave única para las compañías que compiten con Superstore.

Diseño de Base de Datos: Esquema Estrella

Se diseñó la estructura de la base de datos a partir del diccionario de datos, identificando entidades, atributos y relaciones. El diagrama se construyó en db diagram.io. El esquema estrella para Superstore se diseñó con una tabla de hechos (sales_superstore) y varias tablas de dimensiones: dim_order, dim_customer, dim_product y dim_market.



Carga: Creación de la Base de Datos en un Data Warehouse

Se generaron las tablas de hechos y dimensiones en BigQuery, utilizando comandos SQL como CREATE OR REPLACE TABLE.

Tabla	Descripción	Número de Registros
dim_order	Información detallada sobre los pedidos	25,754
dim_customer	Descripción de los clientes	4,873
dim_product	Detalles de los productos	10,768
dim_market	Información geográfica de los mercados	1,126
sales_superstore	Datos transaccionales clave para el análisis de ventas	51,290
competencia	Lista de compañías multinacionales	374

Estrategia de Actualización de Datos

Se implementó una estrategia de actualización de datos, priorizando la actualización de las tablas de dimensiones antes que las de hechos. Se utilizaron cargas incrementales para actualizar solo los registros nuevos o modificados. Se propone implementar técnicas de CDC (Change Data Capture) o marcas de tiempo para detectar cambios y actualizar únicamente los registros modificados.

1

Actualización de Dimensiones

Se actualizan las tablas de dimensiones (dim_order, dim_customer, dim_product, dim_market) primero, ya que las tablas de hechos dependen de las dimensiones para sus relaciones.

2

Actualización de Hechos

Se actualizan las tablas de hechos (sales_superstore) después de las dimensiones, asegurando que cualquier nueva información o cambio en los datos de referencia esté disponible.

3

Carga Incremental

Se implementan cargas incrementales tanto para las dimensiones como para los hechos, actualizando solo los registros nuevos o modificados.

4

Detección de Cambios

Se propone utilizar técnicas de CDC (Change Data Capture) o marcas de tiempo para detectar cambios y actualizar únicamente los registros modificados.

Pipelines de Actualización

Se propusieron dos pipelines para la actualización de datos: un pipeline en GCP (Google Cloud Platform) y un pipeline Open Source. Ambos pipelines incluyen etapas de ingesta (extracción), procesamiento (transformación), modelado de datos (carga), visualización (análisis) y orquestación/automatización.

Pipeline en GCP

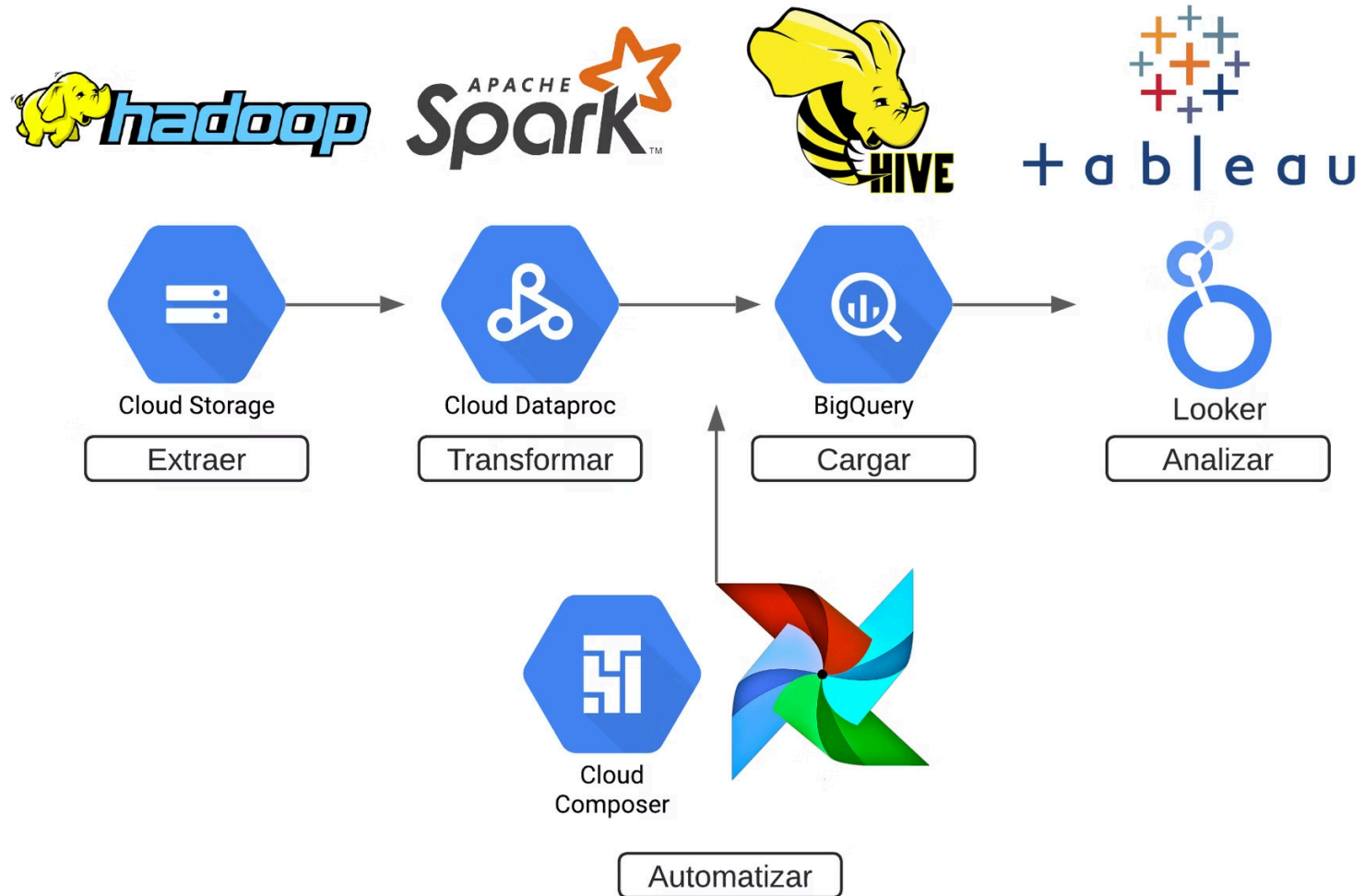
1. Cloud Storage (Ingesta de datos crudos)
2. Dataproc (Procesamiento de datos)
3. BigQuery (Modelado de hechos y dimensiones)
4. Looker Studio (Visualización)
5. Cloud Composer (Orquestación/Automatización)

Pipeline Open Source

1. Hadoop HDFS (Ingesta de datos crudos)
2. Apache Spark (Procesamiento)
3. Hive (Modelado de hechos y dimensiones)
4. Tableau (Visualización)
5. Airflow (Orquestación/Automatización)

Pipelines de Actualización

Pipeline GCP / Open Source



Análisis Exploratorio

Se creó una tabla para el análisis exploratorio a partir de las tablas de hechos y dimensiones, utilizando comandos SQL como LEFT JOIN. El análisis exploratorio se llevó a cabo en Looker Studio, cargando el consolidado desde BigQuery. Se agruparon datos según variables categóricas, se visualizaron las variables categóricas, se aplicaron medidas de tendencia central y dispersión, se visualizó la distribución y se visualizó el comportamiento de los datos a lo largo del tiempo.



Ventas Globales

Las ventas globales alcanzaron los \$12.6 millones con una ganancia de \$1.5 millones, representando un margen de ganancia aproximado del 11.9%. Se registraron 51,290 transacciones con un ticket promedio de \$245.88 y un promedio de 7 unidades vendidas por ticket.



Segmentación

El segmento Consumer representa el mayor segmento, con el 51.7% de las ventas (\$6.5 millones) y ganancias de \$749.2 mil. Corporate contribuye con el 30.1% de las ventas (\$3.8 millones) y \$441.2 mil de ganancias. Home Office tiene una menor participación (18.2%) con \$2.3 millones de ventas y \$227 mil de ganancias.



Evolución Temporal

Las ventas aumentaron año tras año desde 2011 (\$2.3M) hasta 2014 (\$4.3M). Las ganancias también crecieron proporcionalmente, mostrando un buen control sobre los costos. Los costos de envío han crecido un 88% entre 2011 (\$244.3K) y 2014 (\$460.5K) lo que podría estar impactando las ganancias si no se gestionan adecuadamente.



Conclusiones y Recomendaciones: Superstore

Las tablas de hechos y dimensiones permiten un análisis profundo de las ventas, abarcando múltiples perspectivas: clientes, productos, pedidos y mercados.

1

Conclusiones Superstore

1. **Segmento Consumer como motor principal:** Este segmento genera más de la mitad de las ventas y ganancias, pero la proporción de ganancias (11.5%) es algo menor que en los otros segmentos, lo que sugiere oportunidades de mejora en eficiencia o precios.
2. **Crecimiento sólido pero con aumento de costos:** El crecimiento de las ventas ha sido consistente, pero los costos de envío han aumentado significativamente. Se debe investigar cómo optimizar estos costos para mantener márgenes saludables.
3. **Disparidades regionales:** APAC y la UE son los mercados más rentables, mientras que LATAM muestra un margen más bajo. Podría ser útil investigar factores locales que afectan las ganancias en LATAM.
4. **Categorías tecnológicas dominantes:** Los productos de tecnología no solo son los más vendidos, sino que también tienen un margen alto. Sin embargo, otras categorías como muebles, aunque generan muchas ventas, tienen márgenes más bajos.

2

Recomendaciones

1. **Optimización de costos de envío:** Considerar negociaciones con proveedores de logística o la implementación de estrategias de distribución más eficientes para reducir los costos de envío que han crecido considerablemente.
2. **Revisar precios en LATAM:** Dado que las ventas son fuertes, pero las ganancias son relativamente bajas, podría ser recomendable ajustar los precios o reducir costos en LATAM para mejorar el margen.
3. **Enfoque en subcategorías rentables:** Fortalecer las estrategias de ventas en subcategorías con altos márgenes, como **Copiers** y **Phones**, ya que ofrecen los mayores beneficios. Al mismo tiempo, se debe evaluar cómo mejorar el margen en otras subcategorías como **Chairs**.
4. **Estrategias de fidelización en el segmento Consumer:** Dado que este segmento es el más grande, se puede considerar la implementación de programas de fidelización o personalización de ofertas para mantener el crecimiento de ventas y aumentar los márgenes de ganancia.

Conclusiones y Recomendaciones: ETL

El proyecto de ETL para Superstore ha implementado un pipeline sólido que integra la ingesta, transformación y almacenamiento de datos con el uso eficiente de BigQuery para el manejo de tablas de hechos y dimensiones. Las propuestas de pipeline permitirán una actualización eficiente, utilizando cargas incrementales y procesos automatizados que minimizan la carga computacional.

Recomendaciones

1. **Automatización Completa del Pipeline:** Considera implementar **Cloud Composer (Airflow)** para la orquestación automática de todas las etapas del pipeline, programando actualizaciones diarias, semanales o mensuales según sea necesario.
2. **Mejorar el Rendimiento con Particionamiento y Clustering:** Aprovecha el **particionamiento** en BigQuery, particionando la tabla de hechos por año o fechas de transacción para mejorar el rendimiento de consultas sobre grandes volúmenes de datos.
3. **Monitoreo y Alertas:** Implementa un sistema de **monitoreo** y **alertas** para asegurar que las cargas y transformaciones de datos se ejecuten correctamente. Herramientas como **Google Cloud Monitoring** pueden detectar fallos en el pipeline y enviarte notificaciones.
4. **Incrementar la Capacidad de Visualización:** Considera expandir el uso de **Looker Studio** o herramientas adicionales como **Tableau** si los análisis requieren visualizaciones más complejas.
5. **Escalabilidad del Pipeline:** A medida que los volúmenes de datos crezcan, puedes explorar el uso de **Dataflow** para el procesamiento en streaming, lo que te permitirá manejar datos en tiempo real, adaptando el pipeline para análisis en vivo.
6. **Optimización del Almacenamiento:** Revisa periódicamente las tablas de **hechos** y **dimensiones** para eliminar registros innecesarios o consolidar datos históricos, evitando que los costos de almacenamiento en BigQuery crezcan desproporcionadamente.

¡Gracias!

Conoce más en: <https://github.com/JaquelineMera/etl-superstore>

