Detalles Técnicos: El Proceso de Ingesta (ETL)

Para poblar nuestro modelo dimensional, desarrollamos un script en Python que actúa como un pipeline de datos automatizado. Aquí explicamos las herramientas utilizadas:

1. Análisis de Librerías

pyodbc (El Puente)

Es la librería encargada de la conectividad. Permite que Python hable con **SQL Server** utilizando el estándar ODBC (Open Database Connectivity).

- **¿Para qué se usó?** Para enviar comandos SQL (INSERT, DELETE) desde el script hacia nuestra base de datos demo_kimball.

faker (El Generador de Realismo)

En lugar de insertar "Usuario 1", "Usuario 2", usamos **Faker**.

- **¿Para qué se usó?** Para crear nombres de personas, correos electrónicos válidos y fechas de registro aleatorias que parezcan reales. Esto es vital para probar la calidad de los reportes finales.

random (El Motor de Simulación)

Librería estándar de Python para generar números.

- **¿Para qué se usó?** Para simular el comportamiento de compra. Decide aleatoriamente qué cliente compró, cuántas unidades llevó (entre 1 y 5) y a qué precio.

2. Desglose del Código Paso a Paso

La Cadena de Conexión (conn_str)

```
server = r'(localdb)\pruebaserver'
conn_str = (f"Driver={{ODBC Driver 17 for SQL Server}};Server={server};...")
```

- **r (Raw string):** Se usa para que Python no interprete la barra invertida \ como un comando especial, sino como parte del nombre del servidor.
- **ODBC Driver 17:** Es el "traductor" que debe estar instalado en Windows para que la conexión funcione.

Bloque de Control: try, except, finally

Es una buena práctica de ingeniería de software:

1. **try:** Intenta ejecutar la carga de datos.
2. **except:** Si algo falla (ej. el servidor está apagado), atrapa el error y nos avisa en lugar de cerrar el programa bruscamente.
3. **finally: Crucial.** Asegura que la conexión a la base de datos se cierre siempre, liberando memoria del servidor aunque haya ocurrido un error.

Transacciones: commit y rollback

- **conn.commit():** Solo cuando los 1,100 registros están listos, se guardan "de golpe". Esto asegura la integridad de los datos.
- **conn.rollback():** Si algo falla a mitad del camino (por ejemplo, en el registro 500), el script deshace todo lo que hizo para no dejar la base de datos con datos incompletos.

Cómo replicar este paso

1. Asegúrate de tener instalado el Driver de SQL Server.
2. Instala las dependencias: pip install pyodbc faker.
3. Ejecuta el script: python ingesta_datos.py.