ANÁLISIS SNOWFLAKE + ETL – AVANCE 3

Proyecto: FleetLogix – Sistema de Gestión de Transporte y Logística
Autor: Cristian García
Repositorio: https://github.com/cfgarciac/Module-2
Versión del documento: v1.0
Fecha: 13/11/2025

1. INTRODUCCIÓN

La empresa FleetLogix ha operado tradicionalmente con una base de datos transaccional en PostgreSQL, capaz de responder preguntas operativas de corto plazo mediante consultas SQL optimizadas. Sin embargo, la dirección requiere ahora una visión analítica de largo plazo, que permita:

- Evaluar el desempeño histórico del servicio de entregas
- Identificar tendencias por conductor, vehículo, región o cliente
- Analizar eficiencia y costos
- Comparar métricas entre períodos y zonas
- Crear reportes estratégicos y dashboards ejecutivos

Para habilitar este salto hacia el análisis estratégico, se construyó un Data Warehouse en Snowflake bajo un modelo dimensional en estrella, junto con un pipeline ETL automático en Python, capaz de extraer, transformar y cargar los datos diariamente.

Este documento describe:

1. El diseño del Data Warehouse
2. Su implementación en Snowflake
3. El pipeline ETL automático
4. Pruebas realizadas
5. Consultas analíticas
6. Conclusiones

2. DISEÑO DEL DATA WAREHOUSE (MODELO EN ESTRELLA)

2.1. Fundamentación del modelo

El modelo en estrella es ideal para análisis porque:

- Separa mediciones numéricas (fact table) de contexto descriptivo (dimensiones)

- Optimiza las consultas de BI y Machine Learning
- Facilita cálculos agregados (por fecha, ruta, conductor, ciudad)
- Permite historizar cambios (SCD – Slowly Changing Dimensions)

En un negocio como FleetLogix, la unidad de análisis fundamental es una entrega completada, por lo que la tabla central es:

✓ fact_deliveries (tabla de hechos principal)

Contiene métricas clave como:

- tiempo de entrega
- retraso
- distancia
- combustible
- entregas/hora
- ingresos por entrega
- eficiencia del viaje
- firma del cliente
- cumplimiento del SLA (on-time delivery)

Cada fila es una entrega realizada → lo que permite análisis histórico detallado.

2.2. Dimensiones diseñadas

dim_date

- Permite análisis por día, mes, trimestre, año, día hábil, semana fiscal.
- Útil para detectar estacionalidad o picos de demanda.

dim_time

- Horas del día agrupadas en turnos (mañana / tarde / noche).
- Permite medir eficiencia por turno.

dim_vehicle

- Describe características del vehículo (tipo, capacidad, antigüedad).
- Permite medir impacto en rendimiento o consumo de combustible.

dim_driver

- Información del conductor (SCD Type 2).
- Permite medir desempeño individual, cumplimiento de horarios y eficiencia.

dim_route

- Distancia, ciudad destino, costo de peajes.

- Permite comparar rutas y optimizar operaciones.

dim_customer

- Nombre, ciudad, categoría.
- Permite identificar clientes clave y su comportamiento en entregas.

2.3. Ventajas del modelo

Tipo de análisis	Beneficio
Tendencias históricas	Se analizan miles de entregas en años
Comparación por segmentos	Por ciudad, conductor, vehículo
KPI ejecutivos	On-time %, retrasos, revenue real
ML / predicciones	Base perfecta para forecasting

3. IMPLEMENTACIÓN EN SNOWFLAKE

Snowflake fue elegido porque:

Ventajas técnicas clave

- Escalado automático: permite procesar +200k registros por carga sin impacto.
- Time Travel: historial de datos hasta 30 días (útil para auditoría y reversiones).
- Almacenamiento columnar: ideal para grandes volúmenes analíticos.
- Vistas seguras: permite controlar qué área ve qué datos.
- Integración nativa con Python (snowflake-connector + write_pandas).

3.1. Creación del Data Warehouse

Base de datos: FLEETLOGIX_DW
Esquema analítico: ANALYTICS
Tablas del modelo estrella:

- fact_deliveries
- dim_customer
- dim_driver
- dim_vehicle
- dim_route
- dim_date
- dim_time

3.2. Verificación de estructura

DESC TABLE fact_deliveries;

Permite validar:

- tipos correctos
- orden esperado
- columnas necesarias para el análisis

3.3. Verificación de carga exitosa

SELECT COUNT(*) FROM fact_deliveries;

Resultado:
223,051 registros cargados correctamente.

Esto confirma que el ETL funcionó sin problemas y que Snowflake recibió la carga completa.

4. PIPELINE AUTOMÁTICO ETL (PYTHON)

El script 05_etl_pipeline.py implementa un proceso completo:

Extracción
Conexión a PostgreSQL
Lectura de 400,000 registros transaccionales
Filtro: solo entregas completadas
Resultado final extraído: 223,051 registros válidos

Transformación
Transformaciones realizadas:
Cálculo de métricas de negocio
Eliminación de tiempos negativos
Validación de range de pesos
Eficiencia de combustible
Número de entregas por viaje
Revenue por entrega
Estructura SCD-type para dimensiones
Formateo de claves de fecha y tiempo

Carga en Snowflake
Carga en dim_customer con MERGE
Carga de hechos en fact_deliveries
Carga exitosa de 223,051 filas

Automatización
El pipeline quedó programado para ejecutarse cada noche a las 02:00 AM:

schedule.every().day.at("02:00").do(job)

5. PRUEBAS REALIZADAS

Durante la ejecución se verificaron:

Conexión a PostgreSQL
Conexión a Snowflake
Transformación sin errores
Carga por lotes
Inserción en dimensiones
Inserción en hechos +223k filas
Consulta en Snowflake
Métricas finales

Resultado final en Snowflake:
SELECT COUNT(*) FROM fact_deliveries;

-- 223051

6. CONSULTAS ANALÍTICAS (BUSINESS + TÉCNICA)

Estas consultas fueron ejecutadas después de cargar el Data Warehouse para validar que los datos permiten análisis reales de negocio.

6.1. KPI de desempeño general

```
SELECT
    COUNT(*) AS total_deliveries,
    AVG(delay_minutes) AS avg_delay,
    SUM(CASE WHEN is_on_time THEN 1 ELSE 0 END) * 100.0 / COUNT(*)
AS on_time_pct
FROM fact_deliveries;
```

¿Por qué?
Es el dashboard mínimo que un gerente de logística necesita.

¿Para qué?

- Identificar desempeño general
- Evaluar cumplimiento del SLA
- Determinar si la operación está estable

Resultado

Consulta exitosa: confirmó que el dataset soporta KPI ejecutivos sin problemas.

6.2. Ranking de conductores

```
SELECT driver_key,
    COUNT(*) AS deliveries,
    AVG(delay_minutes) AS avg_delay
FROM fact_deliveries
GROUP BY driver_key
ORDER BY deliveries DESC
LIMIT 10;
```

Sentido de negocio

- Detectar conductores más productivos
- Revisar eficiencia por persona
- Identificar necesidad de capacitación

Resultado
Consulta rápida gracias al modelo en estrella (+ almacenamiento columnar).

6.3. Eficiencia por tipo de vehículo

```
SELECT vehicle_key,
    AVG(fuel_efficiency_km_per_liter) AS avg_efficiency
FROM fact_deliveries
GROUP BY vehicle_key;
```

¿Por qué?
La gerencia desea reducir costos de combustible.

¿Para qué?

- Comparar modelos de vehículos
- Tomar decisiones sobre renovación de flota

Resultado
Se obtuvieron promedios correctos: el ETL calculó eficiencias adecuadamente.

6.4. Tendencias históricas (Time Series)

```
SELECT date_key,
    COUNT(*) AS deliveries
```

```
FROM fact_deliveries
GROUP BY date_key
ORDER BY date_key;
```

Uso de negocio

- Identificación de picos de demanda
- Estacionalidad
- Planeación de personal

Resultado
La tabla dim_date permite extender este análisis a niveles: mes, trimestre, año.

6.5. Análisis por ciudad final

```
SELECT r.destination_city,
      COUNT(*) AS deliveries,
      AVG(delay_minutes) AS avg_delay
FROM fact_deliveries f
JOIN dim_route r ON f.route_key = r.route_key
GROUP BY r.destination_city;
```

¿Por qué?
La empresa necesita identificar ciudades problemáticas o congestionadas.

Resultado
Se visualizaron diferencias claras por ciudad: perfecto para dashboards en Power BI.

6.6. Revenue total y por conductor

```
SELECT driver_key,
      SUM(revenue_per_delivery) AS total_revenue
FROM fact_deliveries
GROUP BY driver_key
ORDER BY total_revenue DESC;
```

Uso estratégico

- Medir impacto económico por conductor
- Incentivos, bonos y performance reviews

Resultado
Ejecución perfecta: revenue calculado desde el ETL funciona correctamente.

7. CONCLUSIONES

La migración de un sistema transaccional a un Data Warehouse en Snowflake (OLAP) se logró exitosamente.

El modelo en estrella permite consultas analíticas que antes eran imposibles o muy costosas.
El pipeline ETL cargó correctamente 223,051 registros transformados.
Se verificó la estructura dimensional (dimensiones + hechos) sin inconvenientes.
Las consultas analíticas confirman que el modelo soporta:

- KPI estratégicos
- análisis por conductor
- estudios de eficiencia de vehículos
- análisis por ruta, fecha, cliente y ciudad
- revenue histórico

Snowflake ofrece un entorno escalable, seguro y perfecto para analítica avanzada.