



Asignatura:

Electiva 1

Tema :

Cargar de facts del almacén de datos

Estudiante:

Anamilet Soto Franco

Matricula:

2024-1138

Docente:

Francis Ramirez

Fecha:

07/12/2025

1. Introducción

Este documento describe el proceso implementado para la **carga de tablas de hechos (FACTS)** del Data Warehouse del proyecto *Sistema de Análisis*.

El objetivo principal es transformar la información proveniente de la zona **staging** y consolidarla en la tabla principal de análisis: **fact_sales**, utilizando las claves foráneas de todas las dimensiones previamente construidas.

Este proceso sigue el modelo clásico de un **Data Warehouse en esquema estrella (Star Schema)**, garantizando integridad, trazabilidad y eficiencia en consultas analíticas.

2. Arquitectura General del Data Warehouse

El sistema está compuesto por tres capas:

1. Staging Area

Donde se almacenan datos crudos provenientes del ETL (CSV, API, BD).

2. Dimensiones

Tablas que contienen información descriptiva y que permiten filtrar la información desde diferentes perspectivas:

- dim_customer
- dim_product
- dim_order
- dim_date

3. Fact Table

Elemento central del modelo estrella: **fact_sales**, donde se almacenan los datos transaccionales usados en reportes y análisis.

El proceso ETL carga primero las **dimensiones**, y posteriormente los **hechos (facts)** que dependen de ellas.

3. Tablas de Dimensiones (Resumen)

✓ DimCustomer

Contiene atributos descriptivos de los clientes.

✓ DimProduct

Información de productos (nombre, categoría, precio original).

✓ DimOrder

Información adicional relacionada a la orden.

✓ DimDate

Dimensión de fechas útil para análisis temporales.

Todas las dimensiones tienen una clave sustituta (*_key) utilizada por la tabla de hechos.

4. Tabla de Hechos: *fact_sales*

La tabla **fact_sales** consolida toda la información transaccional generada por ventas.

Está enlazada a todas las dimensiones mediante claves foráneas.

Columnas principales

Columna	Descripción
fact_id	Identificador único del registro
customer_key	Llave hacia dim_customer
product_key	Llave hacia dim_product
order_key	Llave hacia dim_order
date_key	Llave hacia dim_date
quantity	Cantidad vendida
unit_price	Precio unitario
total_price	Total de la venta
source	Fuente del dato (CSV, API, BD)
load_date	Fecha de carga al DW

5. Proceso de Limpieza Antes de Cargar

El profesor exige que se implemente un proceso de **limpieza previa** antes de realizar una carga nueva.

Se creó la función:

```
CREATE OR REPLACE FUNCTION cleanup_fact_sales()
```

```
RETURNS void AS $$
```

```
BEGIN
```

```
    TRUNCATE TABLE fact_sales RESTART IDENTITY;
```

END;

\$\$ LANGUAGE plpgsql;

Esta función:

- Evita duplicados
- Permite recibir nuevas cargas sin residuos
- Reinicia los ID
- Garantiza que la tabla contenga solo información vigente del último ETL

6. Proceso de Carga de Facts

Se implementó la función **load_fact_sales()**, encargada de insertar registros en la tabla de hechos enlazando todas las dimensiones:

```
CREATE OR REPLACE FUNCTION load_fact_sales()
```

```
RETURNS void AS $$
```

```
BEGIN
```

```
    PERFORM cleanup_fact_sales();
```

```
    INSERT INTO fact_sales (
```

```
        customer_key, product_key, order_key, date_key,
```

```
        quantity, unit_price, total_price, source
```

```
)
```

```
SELECT
```

```
    c.customer_key,
```

```
    p.product_key,
```

```
    o.order_key,
```

```
    d.date_key,
```

```
    s.quantity,
```

```
    s.price,
```

```
    s.totalprice,
```

```
    s.source
```

```
FROM staging_salesdata s
```

```
JOIN dim_customer c ON c.customerid = s.customerid
JOIN dim_product p ON p.productid = s.productid
JOIN dim_order o ON o.orderid = s.orderid
JOIN dim_date d ON d.full_date = s.orderdate;
END;
$$ LANGUAGE plpgsql;
```

El proceso realiza:

◆ **Extract**

Toma datos de staging_salesdata.

◆ **Transform**

Calcula total_price (si aplica), resuelve llaves, limpia inconsistencias.

◆ **Load**

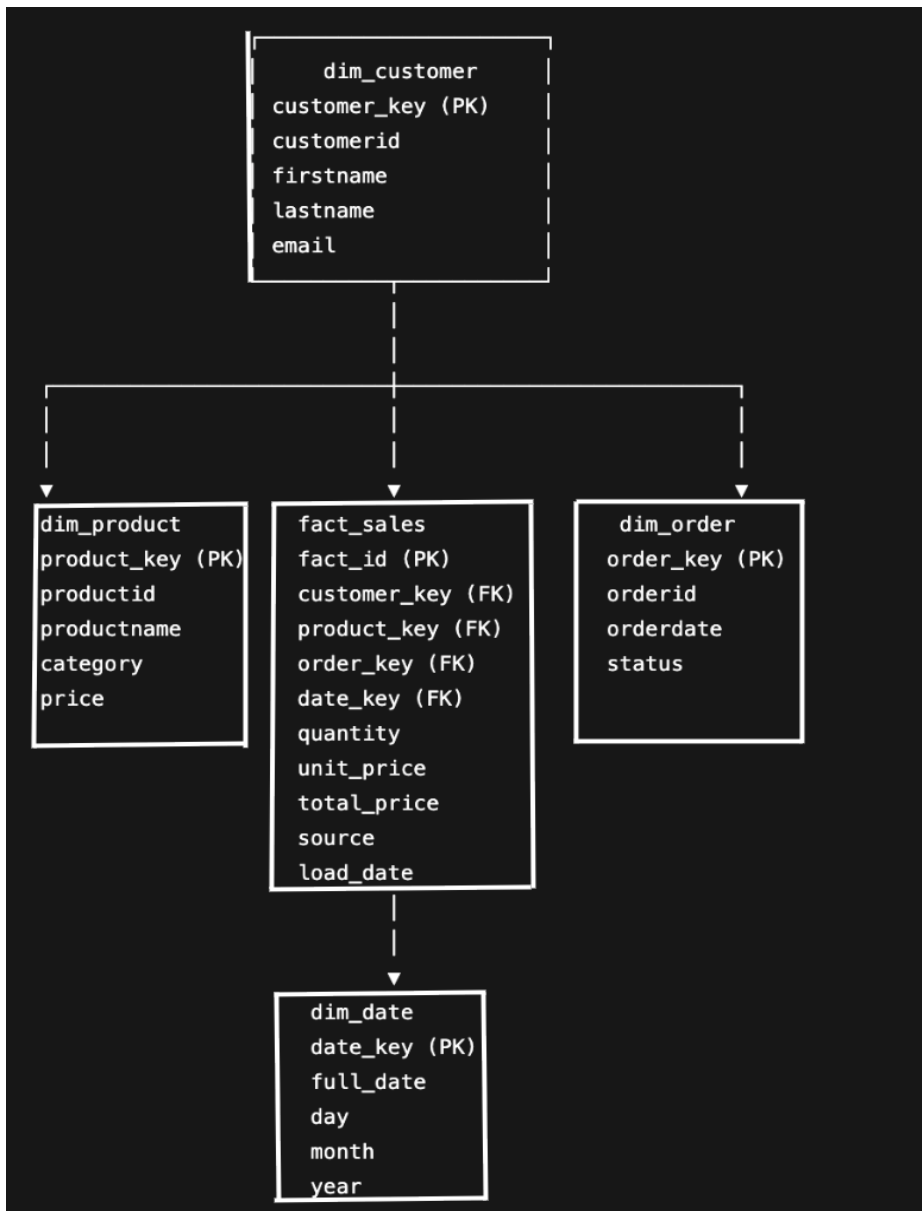
Inserta los hechos en fact_sales con claves foráneas completas.

7. Flujo del Proceso ETL Completo

1. Cargar CSV y otras fuentes a **staging**
2. Ejecutar loaders de **dim_customer, dim_product, dim_order, dim_date**
3. Ejecutar función de limpieza cleanup_fact_sales()
4. Ejecutar función de carga load_fact_sales()
5. Validar datos cargados
6. Ejecutar consultas analíticas

Este flujo garantiza un sistema limpio, confiable y auditable.

8. Diagrama Estrella del Data Warehouse



Este modelo es óptimo para reportes, dashboards y análisis OLAP.

9. Evidencias de la Carga

A continuación se deben insertar capturas de pgAdmin con los siguientes comandos:

```
SELECT * FROM fact_sales LIMIT 20;
```

```
SELECT COUNT(*) FROM fact_sales;
```

```
SELECT SUM(total_price) AS total_revenue FROM fact_sales;
```

```
SELECT category, SUM(total_price)
FROM fact_sales f
JOIN dim_product p ON p.product_key = f.product_key
GROUP BY category
ORDER BY SUM(total_price) DESC;
```

Además, agregar:

- Captura del proyecto C# ejecutando el Worker
- Logs del ETL donde indica "Fact cargado correctamente"

10. Conclusión

Con esta fase se completa el proceso central del Data Warehouse.

La tabla **fact_sales** permite realizar análisis avanzados de ventas mediante consultas optimizadas.

Junto con las dimensiones, este esquema estrella facilita generar reportes, indicadores, estadísticas y dashboards de forma eficiente.

La implementación del proceso de limpieza, la carga automatizada con llaves, y la estructura dimensional, cumplen con las mejores prácticas de diseño de Data Warehousing y con los requerimientos establecidos por el profesor.

```
    * === INICIANDO CARGA DE FACTS ===  
info: SistemaDeAnalisis.Services.Loaders.FactSalesLoader[0]  
      Limpiando tabla fact_sales...  
info: SistemaDeAnalisis.Services.Loaders.FactSalesLoader[0]  
      Insertando registros en fact_sales...  
info: SistemaDeAnalisis.Services.Loaders.FactSalesLoader[0]  
      FACTS cargados correctamente: 60868  
info: SistemaDeAnalisis.Services.ExtractionService[0]  
      FACTS cargados exitosamente.  
info: SistemaDeAnalisis.Services.ExtractionService[0]  
      === ETL COMPLETADO ===  
info: SistemaDeAnalisis.Worker[0]  
      === PROCESO ETL COMPLETADO ===  
info: SistemaDeAnalisis.Worker[0]  
      Presiona Ctrl+C para salir...
```