Evidencia de Aprendizaje 3. Proceso ETL

Juan Jose Madriñan Pinzon

Institución Universitaria Digital de Antioquia

Bases de Datos II Gr.095

Profesor: Victor Hugo Mercado

30 De Marzo de 2025

1. **Introducción**
2. **Objetivos**
3. **Planteamiento del Problema**
4. **Análisis del Problema y Propuesta de Solución**
5. **Anexos**
6. **Conclusiones**

Introducción

En la siguiente actividad se realizará la base de datos staging enfocada en extraer los datos necesarios para formar el modelo de estrella planteado mejorando y estructurando mejor los datos conforme a lo desarrollado en la primera actividad.

Objetivos

El objetivo principal será permitir el estudio eficiente y eficaz de los datos, haciendo la extracción de los datos para formar el modelo de estrella propuesto, se agregara ETL que permitirá una mejor calidad y coherencia de los datos del DataMart para el análisis y sus conclusiones.

Planteamiento del Problema

En el camino a resolver se presentan diferentes desafíos, como el estudio y análisis para permitir un paso efectivo y limpio de los datos, evitando destruir la lógica y enfocándose en una mejor visualización de datos. Incluyendo la implementación de scripts con un enfoque ETL (Extract, Transform, Load) de los datos.

Análisis del Problema y Solución propuesta

Al crear la dimensión del cliente, implemente dos cambios el **primero** me permite categorizar mi cliente como europeo, norteamericano o del resto de continentes (Internacional) permitiendo conocer más a profundidad los clientes, el **segundo** añade una columna que captura el tiempo en que ingresan los datos en esa última fila de la columna ***"ultimo_cambio"***:

```
CREATE TABLE cliente (
    ID_cliente INT NOT NULL IDENTITY(1,1) PRIMARY KEY,
    nombre_cliente VARCHAR(50) NOT NULL,
    telefono VARCHAR(15) NOT NULL,
    ciudad VARCHAR(50) NOT NULL,
    region VARCHAR(50) DEFAULT NULL,
    pais VARCHAR(50) DEFAULT NULL,
    codigo_postal VARCHAR(10),
```

```
    /* CAMBIO 1: Añadir categoría de cliente basada en su ubicación */
    categoria_cliente AS CASE
        WHEN pais = 'Spain' THEN 'Europa'
        WHEN pais IN ('USA', 'Canada') THEN 'Norteamérica'
        ELSE 'Internacional'
    END PERSISTED,

    /* CAMBIO 2: Añadir columna para registro de último cambio */
    ultimo_cambio DATETIME DEFAULT GETDATE()
);
```

Agregamos una normalización de prueba muy básica para los países más vistos en la base de datos que permite colocar el indicativo del país al que pertenece el teléfono del cliente.

```
/* CAMBIO 3 Normalizar el formato del teléfono */
UPDATE cliente
SET telefono = CASE
    WHEN telefono LIKE '+%' THEN telefono
    WHEN LEN(telefono) = 9 AND pais = 'Spain' THEN '+34' + telefono
    WHEN LEN(telefono) = 10 AND pais = 'USA' THEN '+1' + telefono
    ELSE telefono
END,
ultimo_cambio = GETDATE();
```

En la dimensión producto **primeramente** se agregó la columna nivel de stock que permite visualizar con una datos cualitativos según la cantidad si está bajo, medio, o alto el nivel de stock del producto, esto permite un mejor estudio de los datos de los productos, secundariamente agregue la columna ***"ultimo_cambio"*** con el mismo objetivo de registrar la fecha y hora exacta de guardado de los datos.

```
CREATE TABLE producto (
    ID_producto INT IDENTITY(1,1) PRIMARY KEY,
    CodigoProducto VARCHAR(15) NOT NULL,
    nombre VARCHAR(70) NOT NULL,
    Categoria INT NOT NULL,
    cantidad_en_stock SMALLINT NOT NULL,
    proveedor VARCHAR(70) NOT NULL,
    precio_venta NUMERIC(15,2) NOT NULL,
    /* CAMBIO 4: Añadir nivel de stock */
```

```
    nivel_stock AS CASE
        WHEN cantidad_en_stock <= 10 THEN 'Bajo'
        WHEN cantidad_en_stock BETWEEN 11 AND 50 THEN 'Medio'
        ELSE 'Alto'
    END PERSISTED,
    /* CAMBIO 5: Añadir columna para registro de último cambio */
    ultimo_cambio DATETIME DEFAULT GETDATE()
);
```

En la dimensión que tiene un mayor enfoque en la ubicación, agregue la columna ***"ultimo_cambio".***

```
CREATE TABLE empleado (
    ID_empleado INT IDENTITY(1,1) PRIMARY KEY,
    nombre VARCHAR(50),
    apellido1 VARCHAR(50),
    extension VARCHAR(10),
    email VARCHAR(100),
    puesto VARCHAR(70),
    pais_oficina VARCHAR(50),
    ciudad_oficina VARCHAR(30),
    /* CAMBIO 6: Añadir columna para registro de último cambio */
    ultimo_cambio DATETIME DEFAULT GETDATE()
);
```

En la dimensión fecha con enfoque en capturar mejor el tiempo, se hizo un mejor tratamiento de los datos permitiendo separar el año, mes, trimestre y de entrega, de igual forma con datos cualitativos verificar que el pedido tenga su estado de entrega. De igual forma se agregó ***"ultimo_cambio"***.

```
CREATE TABLE fecha (
    ID_fecha INT PRIMARY KEY,
    fecha_pedido DATE NOT NULL,
    fecha_esperada DATE NOT NULL,
    fecha_entrega DATE,
    /* CAMBIO 7: Añadir campos para facilitar análisis temporal */
    año_pedido AS YEAR(fecha_pedido) PERSISTED,
    mes_pedido AS MONTH(fecha_pedido) PERSISTED,
    trimestre_pedido AS DATEPART(QUARTER, fecha_pedido) PERSISTED,
    dias_entrega AS CASE
        WHEN fecha_entrega IS NOT NULL THEN DATEDIFF(DAY, fecha_pedido,
```

```
fecha_entrega)
        ELSE NULL
    END PERSISTED,
    estado_entrega AS CASE
        WHEN fecha_entrega IS NULL THEN 'Pendiente'
        WHEN fecha_entrega <= fecha_esperada THEN 'A tiempo'
        ELSE 'Retrasado'
    END PERSISTED,
    /* CAMBIO 8: Añadir columna para registro de último cambio */
    ultimo_cambio DATETIME DEFAULT GETDATE()
);
```

Finalmente se diseñó la tabla de hechos, con las conexiones a las dimensiones del modelo de estrella, añadiendo la información de ***"cantidad, precio_unidad, total_linea"*** que permitirá un mejor análisis de los datos de la venta, junto con la columna de ***"ultimo_cambio"***:

```
CREATE TABLE ventas (
    ID_cliente INT NOT NULL,
    ID_producto INT NOT NULL,
    ID_empleado INT NOT NULL,
    ID_fecha INT NOT NULL,
    /* CAMBIO 9: Añadir información de cantidad y precio para análisis */
    cantidad INT NOT NULL,
    precio_unidad NUMERIC(15,2) NOT NULL,
    total_linea AS (cantidad * precio_unidad) PERSISTED,
    /* CAMBIO 10: Añadir columna para registro de último cambio */
    ultimo_cambio DATETIME DEFAULT GETDATE(),
    FOREIGN KEY (ID_cliente) REFERENCES cliente(ID_cliente),
    FOREIGN KEY (ID_producto) REFERENCES producto(ID_producto),
    FOREIGN KEY (ID_empleado) REFERENCES empleado(ID_empleado),
    FOREIGN KEY (ID_fecha) REFERENCES fecha(ID_fecha)
);
```

Finalmente se creó una **vista** ordenada de todos los datos:

```
CREATE VIEW vista_ventas_ordenada AS
SELECT
    v.ID_cliente,
    c.nombre_cliente,
    c.categoria_cliente,
```

```
    v.ID_producto,
    p.nombre AS nombre_producto,
    p.nivel_stock,
    v.ID_empleado,
    e.nombre + ' ' + e.apellido1 AS nombre_empleado,
    v.ID_fecha,
    f.fecha_pedido,
    f.año_pedido,
    f.mes_pedido,
    f.trimestre_pedido,
    f.fecha_entrega,
    f.estado_entrega,
    f.dias_entrega,
    v.cantidad,
    v.precio_unidad,
    v.total_linea,
    v.ultimo_cambio AS ultimo_cambio_ventas,
    c.ultimo_cambio AS ultimo_cambio_cliente,
    p.ultimo_cambio AS ultimo_cambio_producto,
    e.ultimo_cambio AS ultimo_cambio_empleado,
    f.ultimo_cambio AS ultimo_cambio_fecha
FROM ventas v
INNER JOIN cliente c ON v.ID_cliente = c.ID_cliente
INNER JOIN producto p ON v.ID_producto = p.ID_producto
INNER JOIN empleado e ON v.ID_empleado = e.ID_empleado
INNER JOIN fecha f ON v.ID_fecha = f.ID_fecha;
```

Anexos:

Google Drive - Con los archivos SQL y BK de las bases de datos.

Conclusión

Obtuvimos que el modelo de estrella, junto con la utilización de lógica ETL permite un mejor tratamiento de datos, gracias a la base de datos staging no hacemos daño al Datamart original y guardamos los datos de la forma más segura, con los backups y scripts podemos recuperar y utilizar los datos para su análisis previo de una forma efectiva y mucho más limpia para sacar las conclusiones necesarios de los datos obtenidos.