CDDEIA-ELNO-5-2

Ciencias de Datos e inteligencia Artificial

Grupo E

Roberto Alvarez
Jesús Mendoza

Tema:
Proyecto DW con PDI

Materia:
Almacenes y minería de datos

Docente:
Msc. Oscar Dario Leon Granizo

Año:
2025

Informe

PROYECTO DW CON PDI: DATA WAREHOUSE ADVENTUREWORKS

1. Introducción y Objetivo

El presente proyecto tiene como objetivo diseñar e implementar una solución integral de Business Intelligence (BI) para la empresa ficticia AdventureWorks. La meta principal es transformar los datos operativos dispersos en una base de datos transaccional (OLTP) hacia un Data Warehouse corporativo centralizado y optimizado para el análisis estratégico de ventas.

Para lograr esto, se desarrolló un proceso de Extracción, Transformación y Carga (ETL**)** robusto utilizando la herramienta Pentaho Data Integration (PDI). Este flujo automatizado se encarga de extraer la información desde PostgreSQL, aplicar reglas de limpieza y estandarización, gestionar la integridad referencial mediante llaves subrogadas (SK) y finalmente poblar un modelo dimensional en esquema estrella, habilitando así la generación de reportes analíticos (OLAP) confiables para la toma de decisiones.

2. Proceso de Negocio a Modelar

El Data Warehouse deberá analizar el comportamiento de **ventas corporativas**, considerando:

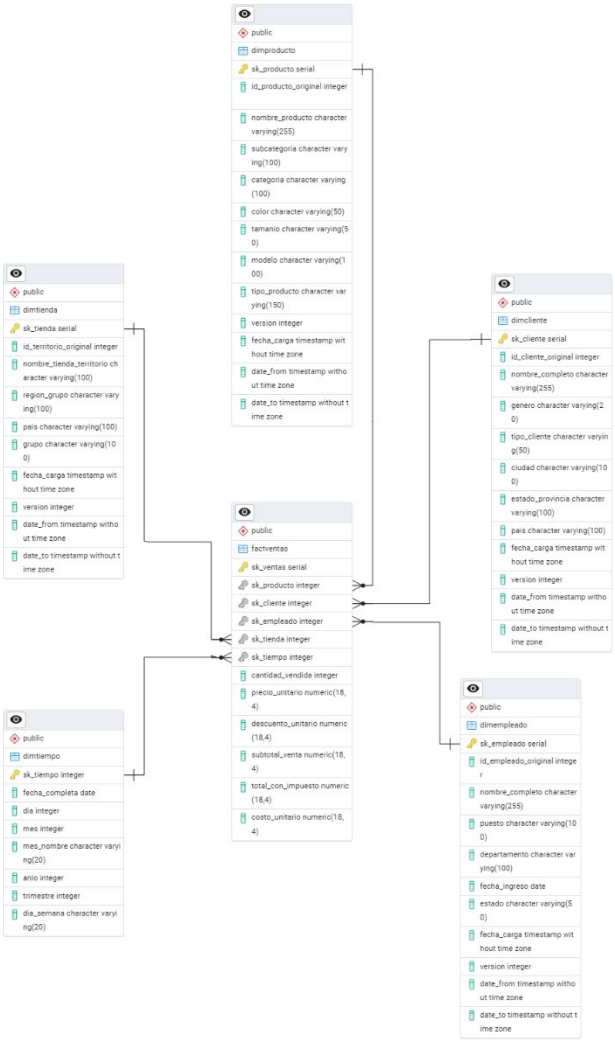
- Productos vendidos
- Clientes
- Empleados que gestionan la venta
- Tiendas/Territorios
- Línea temporal de las ventas

Este proceso permitirá realizar análisis como:

- Ventas por categoría de producto
- Ventas por periodo (mes/año)
- Comparaciones entre territorios
- Clientes más rentables
- Ventas por empleado

3. Modelo Dimensional (Esquema Estrella)
(Diagrama_esquema_estrella.pgerd)

A continuación, se presenta el diagrama Entidad-Relación del Data Warehouse implementado en PostgreSQL (AdventureWorksDW).



Descripción del Modelo:

- **Tabla de Hechos:** FactVentas (Centro del modelo, contiene métricas transaccionales).
- **Dimensiones:** DimTiempo, DimProducto, DimCliente, DimEmpleado, DimTienda.
- **Integridad:** Se implementaron Llaves Subrogadas (SK) secuenciales como identificadores principales en todas las dimensiones. Estas mismas SK se utilizaron en la Tabla de Hechos para establecer las relaciones, garantizando la integridad referencial del modelo."

4. Script SQL para crear el Data Warehouse (01_EstructuraDW.sql)

```sql
-- 1. DIMENSIÓN TIEMPO
CREATE TABLE DimTiempo (
    sk_tiempo INT PRIMARY KEY,
    fecha_completa DATE,
    dia INT,
    mes INT,
    mes_nombre VARCHAR(20),
    anio INT,
    trimestre INT,
    dia_semana VARCHAR(20)
);

-- 2. DIMENSIÓN PRODUCTO
CREATE TABLE DimProducto (
    sk_producto SERIAL PRIMARY KEY,
    id_producto_original INT,
    nombre_producto VARCHAR(255),
    subcategoria VARCHAR(100),
    categoria VARCHAR(100),
    color VARCHAR(50),
    tamanio VARCHAR(50),
    modelo VARCHAR(100),
    tipo_producto VARCHAR(150),
    version INT DEFAULT 1,
    fecha_carga TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    date_from TIMESTAMP,
    date_to TIMESTAMP
);
-- Fila Cero (Protección contra errores de Pentaho)
INSERT INTO DimProducto (sk_producto, id_producto_original,
VALUES (0, 0, 'Desconocido', 'N/D', 'N/D', 'N/D', 'N/D', 'N
ALTER SEQUENCE dimproducto_sk_producto_seq RESTART WITH 1;
```

```sql
-- 3. DIMENSIÓN CLIENTE
CREATE TABLE DimCliente (
    sk_cliente SERIAL PRIMARY KEY,
    id_cliente_original INT,
    nombre_completo VARCHAR(255),
    genero VARCHAR(20),
    tipo_cliente VARCHAR(50),
    ciudad VARCHAR(100),
    estado_provincia VARCHAR(100),
    pais VARCHAR(100),
    fecha_carga TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    version INT DEFAULT 1,
    date_from TIMESTAMP,
    date_to TIMESTAMP
);
-- Fila Cero
INSERT INTO DimCliente (sk_cliente, id_cliente_original,
VALUES (0, 0, 'Desconocido', 'N/D', 'N/D', 'N/D', 'N/D',
ALTER SEQUENCE dimcliente_sk_cliente_seq RESTART WITH 1;

-- 4. DIMENSIÓN EMPLEADO
CREATE TABLE DimEmpleado (
    sk_empleado SERIAL PRIMARY KEY,
    id_empleado_original INT,
    nombre_completo VARCHAR(255),
    puesto VARCHAR(100),
    departamento VARCHAR(100),
    fecha_ingreso DATE,
    estado VARCHAR(50),
    fecha_carga TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    version INT DEFAULT 1,
    date_from TIMESTAMP,
    date_to TIMESTAMP
);
-- Fila Cero
INSERT INTO DimEmpleado (sk_empleado, id_empleado_origir
VALUES (0, 0, 'Desconocido', 'N/D', 'N/D', '1900-01-01',
ALTER SEQUENCE dimempleado_sk_empleado_seq RESTART WITH
```

```sql
-- 5. DIMENSIÓN TIENDA
CREATE TABLE DimTienda (
    sk_tienda SERIAL PRIMARY KEY,
    id_territorio_original INT,
    nombre_tienda_territorio VARCHAR(100),
    region_grupo VARCHAR(100),
    pais VARCHAR(100),
    grupo VARCHAR(100),
    fecha_carga TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    version INT DEFAULT 1,
    date_from TIMESTAMP,
    date_to TIMESTAMP
);
-- Fila Cero
INSERT INTO DimTienda (sk_tienda, id_territorio_original,
VALUES (0, 0, 'Desconocido', 'N/D', 'N/D', 'N/D', 0, '196
ALTER SEQUENCE dimtienda_sk_tienda_seq RESTART WITH 1;

-- 6. TABLA DE HECHOS
CREATE TABLE FactVentas (
    sk_ventas SERIAL PRIMARY KEY,
    sk_producto INT REFERENCES DimProducto(sk_producto),
    sk_cliente INT REFERENCES DimCliente(sk_cliente),
    sk_empleado INT REFERENCES DimEmpleado(sk_empleado),
    sk_tienda INT REFERENCES DimTienda(sk_tienda),
    sk_tiempo INT REFERENCES DimTiempo(sk_tiempo),
    cantidad_vendida INT,
    precio_unitario NUMERIC(18,4),
    descuento_unitario NUMERIC(18,4),
    subtotal_venta NUMERIC(18,4),
    total_con_impuesto NUMERIC(18,4),
    costo_unitario NUMERIC(18,4)
);
```

5. Consultas Analíticas y Resultados
(02_Consultas_Reporte.sql)

Se ejecutaron 5 consultas clave para validar la utilidad del Data Warehouse.
(Consultas ejecutadas antes de hacer la validación de carga incremental)

5.1. Ventas mensuales por categoría de producto
Análisis: Esta consulta permite visualizar la estacionalidad de las ventas agrupadas por la categoría principal del producto.

Código SQL:

```sql
-- CONSULTA 1: Ventas mensuales por categoría
-- Análisis: Tendencia temporal de ventas por familias de productos.
SELECT
    t.anio AS "Año",
    t.mes_nombre AS "Mes",
    p.categoria AS "Categoría",
    SUM(f.subtotal_venta) AS "Ventas Totales ($)"
FROM FactVentas f
JOIN DimTiempo t ON f.sk_tiempo = t.sk_tiempo
JOIN DimProducto p ON f.sk_producto = p.sk_producto
GROUP BY t.anio, t.mes, t.mes_nombre, p.categoria
ORDER BY t.anio DESC, t.mes DESC, "Ventas Totales ($)" DESC;
```

Resultado:

Data Output | Mensajes | Notificaciones

	Año integer	Mes character varying (20)	Categoría character varying (50)	Ventas Totales ($) numeric
1	2014	Junio	Accessories	32728.7200
2	2014	Junio	Clothing	16277.1200
3	2014	Mayo	Bikes	4565082.5645
4	2014	Mayo	Components	577138.5547
5	2014	Mayo	Clothing	119667.8358
6	2014	Mayo	Accessories	104786.0150
7	2014	Abril	Bikes	1696566.2400
8	2014	Abril	Accessories	69061.5480
9	2014	Abril	Clothing	30832.3390
10	2014	Abril	Components	713.7960
11	2014	Marzo	Bikes	6044367.6469
12	2014	Marzo	Components	855916.6785

5.2. Ventas anuales por territorio

Análisis: Comparativa del rendimiento financiero desglosado por región geográfica y año.

Código SQL:

```sql
-- CONSULTA 2: Ventas anuales por territorio
-- Análisis: Rendimiento geográfico.
SELECT
    t.anio AS "Año",
    ti.pais AS "País",
    ti.nombre_tienda_territorio AS "Territorio",
    SUM(f.subtotal_venta) AS "Ventas Anuales ($)"
FROM FactVentas f
JOIN DimTiempo t ON f.sk_tiempo = t.sk_tiempo
JOIN DimTienda ti ON f.sk_tienda = ti.sk_tienda
GROUP BY t.anio, ti.pais, ti.nombre_tienda_territorio
ORDER BY t.anio DESC, "Ventas Anuales ($)" DESC;
```

Resultado:

Data Output | Mensajes | Notificaciones

	Año integer	País character varying (50)	Territorio character varying (50)	Ventas Anuales ($) numeric
1	2014	United States	Southwest	3971898.0185
2	2014	United States	Northwest	2997750.7117
3	2014	Australia	Australia	2767732.1467
4	2014	Canada	Canada	2390813.0033
5	2014	United Kingdom	United Kingdom	2092554.1321
6	2014	France	France	1674078.8043
7	2014	Germany	Germany	1553255.2560
8	2014	United States	Central	955864.7342
9	2014	United States	Southeast	875604.2141
10	2014	United States	Northeast	778377.7924
11	2013	United States	Southwest	9116540.3170
12	2013	Canada	Canada	6229517.5655

Total rows: 40 of 40 Query complete 00:00:00.167 Ln 64, Col 1

5.3. Top 10 Clientes con mayor facturación

Análisis: Identificación de los clientes VIP (Pareto). Se implementó lógica para imputar la ubicación de la tienda en casos donde el cliente no tenía dirección registrada.

Código SQL:

```sql
-- CONSULTA 3: Top 10 Clientes VIP
-- Análisis: Identificación de clientes clave (Pareto).
SELECT
    c.nombre_completo AS "Cliente",
    c.tipo_cliente AS "Tipo",
    MAX(CASE
        WHEN c.pais IS NULL OR c.pais IN ('N/D', 'N/A') THEN ti.pais
        ELSE c.pais
    END) AS "País",

    COUNT(f.sk_ventas) as "Nro. Transacciones",
    SUM(f.subtotal_venta) AS "Facturación Total ($)"
FROM FactVentas f
JOIN DimCliente c ON f.sk_cliente = c.sk_cliente
JOIN DimTienda ti ON f.sk_tienda = ti.sk_tienda
GROUP BY c.nombre_completo, c.tipo_cliente
ORDER BY "Facturación Total ($)" DESC
LIMIT 10;
```

Resultado:

Data Output | Mensajes | Notificaciones

	Cliente text	Tipo text	País text	Nro. Transacciones bigint	Facturación Total ($) numeric
1	Roger Harui	Empresa	United States	298	877107.1923
2	Andrew Dixon	Empresa	United States	366	853849.1795
3	Reuben D'sa	Empresa	Canada	530	841908.7708
4	Robert Vessa	Empresa	United States	436	816755.5763
5	Ryan Calafato	Empresa	Canada	451	799277.8953
6	Joseph Castellucio	Empresa	Canada	429	787773.0436
7	Kirk DeGrasse	Empresa	United States	332	746317.5293
8	Lindsey Camacho	Empresa	United States	358	740985.8338
9	Robin McGuigan	Empresa	Canada	352	730798.7139
10	Stacey Cereghino	Empresa	United States	336	727272.6494

5.4. Comparación de ventas por empleado

Análisis: Ranking de productividad de la fuerza de ventas interna.

Código SQL:

```sql
-- CONSULTA 4: Comparacion de ventas por empleado (Ranking empleados)
-- Análisis: Productividad del equipo comercial.
SELECT
    e.nombre_completo AS "Vendedor",
    e.puesto AS "Cargo",
    e.departamento AS "Departamento",
    SUM(f.subtotal_venta) AS "Ventas Generadas ($)"
FROM FactVentas f
JOIN DimEmpleado e ON f.sk_empleado = e.sk_empleado
WHERE e.nombre_completo <> 'Cliente Desconocido' -- Filtramos ventas online
GROUP BY e.nombre_completo, e.puesto, e.departamento
ORDER BY "Ventas Generadas ($)" DESC;
```

Resultado:

Data Output | Mensajes | Notificaciones

	Vendedor text	Cargo character varying (50)	Departamento character varying (50)	Ventas Generadas ($) numeric
1	Linda Mitchell	Sales Representative	Sales	10367007.4286
2	Jillian Carson	Sales Representative	Sales	10065803.5429
3	Michael Blythe	Sales Representative	Sales	9293903.0055
4	Jae Pak	Sales Representative	Sales	8503338.6472
5	Tsvi Reiter	Sales Representative	Sales	7171012.7514
6	Shu Ito	Sales Representative	Sales	6427005.5556
7	José Saraiva	Sales Representative	Sales	5926418.3574
8	Ranjit Varkey Chudukatil	Sales Representative	Sales	4509888.9330
9	David Campbell	Sales Representative	Sales	3729945.3501
10	Garrett Vargas	Sales Representative	Sales	3609447.2163
11	Pamela Ansman-Wolfe	Sales Representative	Sales	3325102.5952
12	Tete Mensa-Annan	Sales Representative	Sales	2312545.6905
13	Rachel Valdez	Sales Representative	Sales	1827066.7133
14	Lynn Tsoflias	Sales Representative	Sales	1421810.9252
15	Stephen Jiang	North American Sales Manager	Sales	1092123.8562
16	Amy Alberts	European Sales Manager	Sales	732759.1844
17	Syed Abbas	Pacific Sales Manager	Sales	172524.4515

5.5. Margen estimado por producto

Análisis: Cálculo de rentabilidad real restando el costo estándar al precio de venta unitario.

Código SQL:

```
-- CONSULTA 5: Margen de Ganancia estimado por Producto
-- Análisis: Rentabilidad real por SKU y Modelo.
SELECT
    p.modelo AS "Modelo (Genérico)",           -- Agregado para claridad
    p.nombre_producto AS "Producto (SKU)",     -- El nombre específico
    p.subcategoria AS "Subcategoría",
    SUM(f.cantidad_vendida) as "Unidades",
    -- Cálculo del Margen: (Precio Venta Real - Costo Estándar) * Cantidad
    SUM((f.precio_unitario - f.costo_unitario) * f.cantidad_vendida) AS "Margen Total ($)"
FROM FactVentas f
JOIN DimProducto p ON f.sk_producto = p.sk_producto
GROUP BY p.modelo, p.nombre_producto, p.subcategoria
ORDER BY "Margen Total ($)" DESC
LIMIT 15;
```

Resultado:

Data Output | Mensajes | Notificaciones

	Modelo (Genérico) character varying (50)	Producto (SKU) character varying (50)	Subcategoría character varying (50)	Unidades bigint	Margen Total ($) numeric
1	Mountain-200	Mountain-200 Black, 38	Mountain Bikes	2977	679002.9361
2	Mountain-200	Mountain-200 Black, 42	Mountain Bikes	2664	678789.6167
3	Mountain-200	Mountain-200 Black, 46	Mountain Bikes	2111	668165.9142
4	Mountain-200	Mountain-200 Silver, 38	Mountain Bikes	2394	666593.3896
5	Mountain-200	Mountain-200 Silver, 46	Mountain Bikes	2216	631477.9826
6	Mountain-200	Mountain-200 Silver, 42	Mountain Bikes	2234	613898.5813
7	Road-150	Road-150 Red, 48	Road Bikes	493	470355.0214
8	Road-150	Road-150 Red, 62	Road Bikes	600	466320.1680
9	Road-150	Road-150 Red, 52	Road Bikes	458	421110.8684
10	Road-150	Road-150 Red, 56	Road Bikes	664	406079.2792
11	Road-150	Road-150 Red, 44	Road Bikes	437	391564.3766
12	Touring-1000	Touring-1000 Blue, 54	Touring Bikes	413	131312.6733
13	Road-250	Road-250 Red, 58	Road Bikes	946	116027.4691
14	Road-250	Road-250 Red, 48	Road Bikes	812	115816.7122
15	Road-350-W	Road-350-W Yellow, 44	Road Bikes	536	113778.5600

6. Validación de Carga Incremental y Manejo de Historia (03_Script_Prueba_Incremental.sql)

Para garantizar la robustez del Data Warehouse, se diseñó un escenario de prueba en la base de datos origen (OLTP) con el objetivo de verificar dos funcionalidades críticas del proceso ETL:

1. **Detección de Cambios (SCD Tipo 2):** Verificar que el sistema cierra la versión anterior de un producto y crea una nueva al detectar cambios en atributos clave.

2. **Integridad Transaccional:** Asegurar que las nuevas ventas se vinculen correctamente a la versión vigente del producto.

Para certificar la fiabilidad del Data Warehouse, se diseñó y ejecutó un escenario de prueba de estrés en el sistema origen (OLTP). El objetivo fue verificar que el proceso ETL detecta correctamente los cambios en los datos maestros (SCD Tipo 2) y vincula las nuevas transacciones a la versión vigente correspondiente.

Procedimiento de la Prueba

El escenario consistió en la simulación manual de un ciclo de negocio completo mediante los siguientes pasos:

6.1. Modificación de Datos Maestros (Simulación de Cambio)

Se ejecutó una actualización directa (UPDATE) sobre el Producto 707 en la base de datos origen.

Version original: (adventure)

	productid [PK] integer	name character varying (50)	color character varying (15)	listprice numeric	modifieddate timestamp without time zone
1	707	Sport-100 Helmet, Red	Red	34.99	2014-02-08 10:01:36.827

Cambio Realizado: El color se modificó a 'Metallic Red' y el precio de lista se incrementó a 1,500.00.

	productid [PK] integer	name character varying (50)	color character varying (15)	listprice numeric	modifieddate timestamp without time zone
1	707	Sport-100 Helmet, Red	Metallic Red	1500.00	2025-11-26 12:11:45.235055

Marcador de Tiempo: Se actualizó el campo ModifiedDate a la fecha actual (NOW()) para activar el disparador de cambio en el ETL.

6.2. Simulación de Nueva Transacción (Venta Futura)

Para probar la integridad referencial temporal, se insertó manualmente un nuevo pedido (SalesOrderHeader y SalesOrderDetail) en el sistema origen.

- **Estrategia de Tiempo:** Se registró la venta con un sello de tiempo de 1 hora en el futuro (NOW() + INTERVAL '1 hour') respecto a la modificación del producto. Esto garantiza que, lógicamente, la venta ocurra después del cambio de precio, obligando al Data Warehouse a vincularla con la nueva versión del producto (Versión 2).

6.3. Ejecución del ETL y Auditoría

Tras ejecutar el Job de carga incremental en Pentaho, se realizaron consultas de verificación en el Data Warehouse (AdventureWorksDW) que confirmaron:

- La creación de una nueva versión (Versión 2) en DimProducto con los nuevos atributos.
- El cierre correcto de la vigencia de la Versión 1.
- Los cambios en las fechas date_from y date_to al haber sido creada una nueva versión

Antes: (DimProducto)

Data Output | Mensajes | Notificaciones

	sk_producto [PK] integer	id_producto_original integer	nombre_producto character varying (255)	color character varying (50)	version integer	date_from timestamp without time zone	date_to timestamp without time zone
1	212	707	Sport-100 Helmet, Red	Red	1	1900-01-01 00:00:00	2199-12-31 23:59:59.999

Después: (DimProducto)

Data Output | Mensajes | Notificaciones

	sk_producto [PK] integer	id_producto_original integer	nombre_producto character varying (255)	color character varying (50)	version integer	date_from timestamp without time zone	date_to timestamp without time zone
1	505	707	Sport-100 Helmet, Red	Metallic Red	2	2025-11-26 12:11:45.235055	2199-12-31 23:59:59.999
2	212	707	Sport-100 Helmet, Red	Red	1	1900-01-01 00:00:00	2025-11-26 12:11:45.235055

Consultas más detalladas en la tabla de hechos (FactVentas):

Ordenado por versión:

	sk_ventas integer	nombre_producto character varying (255)	color character varying (50)	version integer	precio_unitario numeric (18,4)	date_from timestamp without time zone	date_to timestamp without time zone	cantidad_vendida integer
1	242635	Sport-100 Helmet, Red	Metallic Red	2	1500.0000	2025-11-26 12:11:45.235055	2199-12-31 23:59:59.999	2
2	121404	Sport-100 Helmet, Red	Red	1	20.1865	1900-01-01 00:00:00	2025-11-26 12:11:45.235055	2
3	121456	Sport-100 Helmet, Red	Red	1	20.1865	1900-01-01 00:00:00	2025-11-26 12:11:45.235055	4
4	121481	Sport-100 Helmet, Red	Red	1	20.1865	1900-01-01 00:00:00	2025-11-26 12:11:45.235055	1
5	121502	Sport-100 Helmet, Red	Red	1	20.1865	1900-01-01 00:00:00	2025-11-26 12:11:45.235055	1
6	121509	Sport-100 Helmet, Red	Red	1	20.1865	1900-01-01 00:00:00	2025-11-26 12:11:45.235055	4

Ordenado por fecha de ventas:

	sk_ventas integer	Fecha de Venta date	nombre_producto character varying (255)	Color del Producto character varying (50)	Precio Cobrado numeric (18,4)	Versión Usada integer	Vigencia Desde timestamp without time zone	Vigencia Hasta timestamp without time zone
1	242635	2025-11-26	Sport-100 Helmet, Red	Metallic Red	1500.0000	2	2025-11-26 12:11:45.235055	2199-12-31 23:59:59.999
2	242588	2014-06-30	Sport-100 Helmet, Red	Red	34.9900	1	1900-01-01 00:00:00	2025-11-26 12:11:45.235055
3	242543	2014-06-30	Sport-100 Helmet, Red	Red	34.9900	1	1900-01-01 00:00:00	2025-11-26 12:11:45.235055
4	242629	2014-06-30	Sport-100 Helmet, Red	Red	34.9900	1	1900-01-01 00:00:00	2025-11-26 12:11:45.235055
5	242416	2014-06-28	Sport-100 Helmet, Red	Red	34.9900	1	1900-01-01 00:00:00	2025-11-26 12:11:45.235055
6	242338	2014-06-27	Sport-100 Helmet, Red	Red	34.9900	1	1900-01-01 00:00:00	2025-11-26 12:11:45.235055

7. Conclusiones

El proyecto culminó con la implementación exitosa de una solución de Data Warehousing completa utilizando Pentaho Data Integration (PDI) y PostgreSQL. Se logró transformar un entorno transaccional (OLTP) en un modelo dimensional optimizado (OLAP), cumpliendo con los estándares de calidad, integridad y automatización requeridos para los almacenes de datos e inteligencia de negocios.

Los hitos principales alcanzados son:

1. **Automatización y Orquestación del Flujo:** Mediante el diseño de un Job maestro, se consolidó un proceso ETL re-ejecutable (idempotente) que orquesta la carga secuencial de dimensiones y hechos. Esto asegura la consistencia de los datos en cada ejecución sin duplicidad ni intervención manual.

2. **Gestión Avanzada de Carga Incremental (SCD Tipo 2):** Se implementó una lógica de detección de cambios que permite la historización de los datos maestros. El sistema es capaz de identificar modificaciones en el origen (ej. cambios de precio o características de producto), cerrar la vigencia del registro anterior y generar una nueva versión, garantizando que las transacciones históricas mantengan sus valores originales mientras las nuevas se asocian a la información actualizada.

3. **Calidad y Estandarización de Datos:** Se aplicaron técnicas efectivas de limpieza (tratamiento de valores nulos) y homogeneización de textos durante la transformación, elevando la confiabilidad de la información respecto a la fuente original y asegurando la integridad referencial del modelo.

4. **Visión Estratégica del Modelo Estrella:** La arquitectura dimensional diseñada facilita la ejecución eficiente de consultas analíticas complejas, permitiendo el cálculo preciso de métricas clave como el margen real por producto y el análisis de rendimiento geográfico, que eran costosos de obtener en el sistema transaccional.

8. Anexos

LINK AL REPOSITORIO EN GITHUB: https://github.com/Robe1o/adventureworks-data-warehouse-etl