

Cargar Los Datos A La Base De Datos De Datamart

Yusep Alexander Ruiz Agudelo

Ingeniería en Desarrollo y Datos

Trabajo realizado para

Bases de Datos II

Profesor

Victor Hugo Mercado

Institución Universitaria Digital de Antioquia

Facultad De Ingenierías

Medellín

2024

Tabla de Contenido

OBJETIVOS	4
OBJETIVO GENERAL	4
OBJETIVOS ESPECÍFICOS.....	4
PLANTEAMIENTO DEL PROBLEMA	5
ANÁLISIS DEL PROBLEMA	6
PROPUESTA DE LA SOLUCIÓN.....	8
DESCRIPCIÓN DEL MODELO ESTRELLA PROPUESTO.	8
<i>Tabla De Hechos</i>	8
<i>Tablas De Dimensiones</i>	9
DISEÑO (IMAGEN) DEL MODELO ESTRELLA	11
LISTA DE DIMENSIONES PROPUESTAS.	12
TABLA DE HECHOS PROPUESTA	14
DESCRIPCIÓN DEL ANÁLISIS REALIZADO A LOS DATOS JARDINERÍA Y CÓMO ESTOS SE TRASLADARON A LA BASE DE DATOS STAGING.	15
IMÁGENES CON LAS SENTENCIAS SQL PARA LA CREACIÓN DE LAS TABLAS STAGING	16
IMÁGENES CON LAS SENTENCIAS SQL PARA LA INSERCIÓN DE LA INFORMACIÓN EN STAGING	19
IMÁGENES CON LAS SENTENCIAS SQL PARA LA TRANSFORMACIÓN DE LA INFORMACIÓN.....	23
IMÁGENES CON LAS SENTENCIAS SQL PARA LA CREACIÓN DE LAS TABLAS DATAMART	25
IMÁGENES CON LAS SENTENCIAS SQL PARA LA INSERCIÓN DE LA INFORMACIÓN EN DATAMART.....	28
IMÁGENES CON LAS SENTENCIAS SQL PARA LA VALIDACIÓN DE DATOS.....	32
CONCLUSIONES	36
BIBLIOGRAFÍA	38

Introducción

“El esquema de estrella es un enfoque de modelado maduro ampliamente adoptado por los almacenes de datos relacionales. Requiere que los modeladores clasifiquen las tablas del modelo como dimensiones o hechos.” (Microsoft, 2023)

Una buena definición que se puede obtener de una tabla de hechos, de una forma clara según (OpenIA, 2024) sería:

una tabla central en un modelo de datos que contiene métricas o medidas cuantitativas relacionadas con los eventos de negocio. Estas tablas registran datos numéricos que son relevantes para el análisis, como ventas, pedidos o transacciones. Cada fila en una tabla de hechos representa un evento individual y contiene claves foráneas que se relacionan con las tablas de dimensiones, lo que permite segmentar y analizar los datos en diferentes contextos. Las tablas de hechos suelen tener un gran volumen de registros, ya que están diseñadas para capturar información a un nivel de detalle específico, conocido como granularidad.

(Microsoft, 2023) dice que las tablas denominadas dimensiones “describen entidades empresariales (las cosas que se modelan)”, a su vez (OpenIA, 2024) informa que “Estas tablas contienen atributos descriptivos que permiten categorizar y segmentar los datos de la tabla de hechos. Por ejemplo, una tabla de dimensiones puede incluir información sobre productos, clientes o fechas”.

Por medio de un análisis detallado de la base de datos de Jardinería, se identificará las relaciones que existen entre cada tabla y la relevancia de los datos que se encuentran en ellas. Posterior al análisis, se definirán los campos necesarios para la construcción de la tabla de

hechos y las dimensionales, proporcionando un diseño claro y estructurado para la construcción del modelo estrella.

Aplicar este modelo brindará la posibilidad de identificar el producto más vendido, la categoría con más productos y el año con más ventas, proporcionando una cantidad de información vital para la toma de decisiones y un mejor entendimiento del comportamiento del mercado.

Posterior a aplicar el modelo estrella a los datos, se aplicará la arquitectura de Staging.
¿Pero qué es Staging?:

Staging es un área de almacenamiento intermedia donde los datos se almacenan y procesan temporalmente antes de ser trasladados a su destino final. Este concepto se usa comúnmente en el contexto del almacenamiento de datos, la integración de datos y los procesos ETL (Extracción, Transformación, Carga).

¿Y para qué se utiliza? El área de preparación de datos se utiliza para facilitar el movimiento fluido y eficiente de los datos desde los sistemas de origen hasta los sistemas de destino, al tiempo que se garantiza la calidad y la integridad de los datos.

También ayuda a garantizar que los datos sean precisos, coherentes y estén listos para el análisis o el uso empresarial. Piense en ello como el paso intermedio crucial que garantiza que los datos estén en óptimas condiciones antes de que ingresen a las bases de datos o herramientas analíticas. (Atlan, 2023)

Objetivos

Objetivo General

Crear una base de datos Staging basada en la base de datos Jardineria, con el fin de almacenar la información temporalmente, de modo que facilite el proceso de ETL (Extract, Transform, Load) posterior al análisis.

Objetivos Específicos

- Analizar la información contenida en la base de datos Jardinería.
- Realizar un análisis de los datos y su estructura.
- Crear una base de datos staging que permita la transformación y preparación de los datos para su análisis.
- Documentar el proceso y los scripts utilizados en la creación de la base de datos.

Planteamiento Del Problema

La base de datos de Jardinería contiene información sobre productos, clientes, pedidos, pagos y empleados, organizada en múltiples tablas interrelacionadas. Sin embargo, la estructura actual se encuentra en un modelo relacional normalizado, generando desafíos en el análisis de los datos. Se está presentando una dificultad a la hora de acceder y analizar la información de una forma eficiente, limitando la capacidad en la compañía para la toma de decisiones.

Para abordar esta problemática, es necesario implementar un modelo estrella que organice los datos con una estructura más accesible y a su vez analizable. Esto brindaría a la compañía la capacidad de simplificar el acceso a la información y a su vez obtener los resultados que desean como la identificación del producto más vendido, la categoría con más productos y el año con más ventas.

Para continuar el proceso anterior, se propone la creación de una base de datos Staging para la transformación, extracción y limpieza de los datos, de modo que permita un mejor análisis y consistencia de la información.

Análisis Del Problema

Teniendo en cuenta la necesidad de obtener las ventas o transacciones de la empresa, el análisis propuesto es el siguiente:

Se analizó la base de datos de Jardinería y se llega a la conclusión de que la tabla más importante es *detalle_pedido*, ya que contiene toda la información de las transacciones que se generaron, almacenando el número del pedido, el producto, unidades vendidas y precio. Esta tabla será considerada la tabla de hechos del ejercicio y se llamará *fac_ventas*.

Ahora bien, para realizar las tablas de dimensiones, se tomará en cuenta las siguientes:

- **dim_fecha:** Esta tabla se conformará de la columna *fecha_pedido* en la tabla *pedido*.
- **dim_geografia:** Permitirá almacenar la información geográfica.
- **dim_producto:** Contiene las características del producto y a su vez la descripción de la categoría.
- **dim_empleado:** Contiene la información básica del empleado junto con su jefe.
- **dim_oficina:** Contiene la descripción de la oficina.
- **dim_cliente:** Contiene la información básica del cliente.

Una vez construido el modelo estrella, se procede a crear la base de datos Staging a través de sentencias SQL. Esta base de datos será un área intermedia entre el origen de los datos y el punto final donde se desea llegar. Los beneficios que traería consigo la implementación de esta arquitectura según (Atlan, 2023) serían los siguientes:

- Consolidación de datos
- Mejora de la calidad de los datos

- Alineación de esquemas
- Eficiencia de rendimiento
- Almacenamiento temporal
- Seguridad de los datos
- Auditoría y seguimiento
- Independencia entre el dato origen y el dato a procesar

Propuesta De La Solución

Descripción Del Modelo Estrella Propuesto.

Tabla De Hechos

fac_ventas: Esta tabla contendrá la información acerca de las ventas realizadas en la empresa.

Campos de la tabla de hechos:

- **id_fac_ventas** (PK)
- **id_dim_geo_empleado** (FK con la tabla dim_geografia)
- **id_dim_geo_cliente** (FK con la tabla dim_geografia)
- **id_dim_geo_oficina** (FK con la tabla dim_geografia)
- **id_dim_producto** (FK con la tabla dim_producto)
- **id_dim_fecha** (FK con la tabla dim_fecha)
- **id_dim_empleado** (FK con la tabla dim_empleado)
- **id_dim_cliente** (FK con la tabla dim_cliente)
- **id_dim_oficina** (FK con la tabla dim_oficina)
- **cantidad** (cantidad vendida)
- **precio_unidad** (precio por unidad)
- **valor_total** (valor total calculado)

Tablas De Dimensiones

Tabla de fecha: Contendrá las fechas en las que se realizaron las ventas.

Campos de la dimensión ***dim_fecha:***

- id_dim_fecha (PK)
- año
- mes
- día
- semestre
- num_semana

Tabla de geografía: Contendrá la información geográfica de los clientes, los empleados y la oficina.

Campos de la dimensión ***dim_geografia:***

- id_dim_geografia(PK)
- ciudad
- país
- región

Tabla de producto: Contendrá la información del nombre del producto, la descripción de la categoría y el precio.

Campos de la dimensión *dim_producto*:

- id_dim_producto (PK)
- nombre
- precio_venta
- desc_categoria

Tabla de empleado: Contendrá información del empleado.

Campos de la dimensión *dim_empleado*:

- id_dim_empleado (PK)
- nombre
- puesto
- nombre_jefe

Tabla de oficina: Contendrá las fechas en las que se realizaron las ventas.

Campos de la dimensión *dim_oficina*:

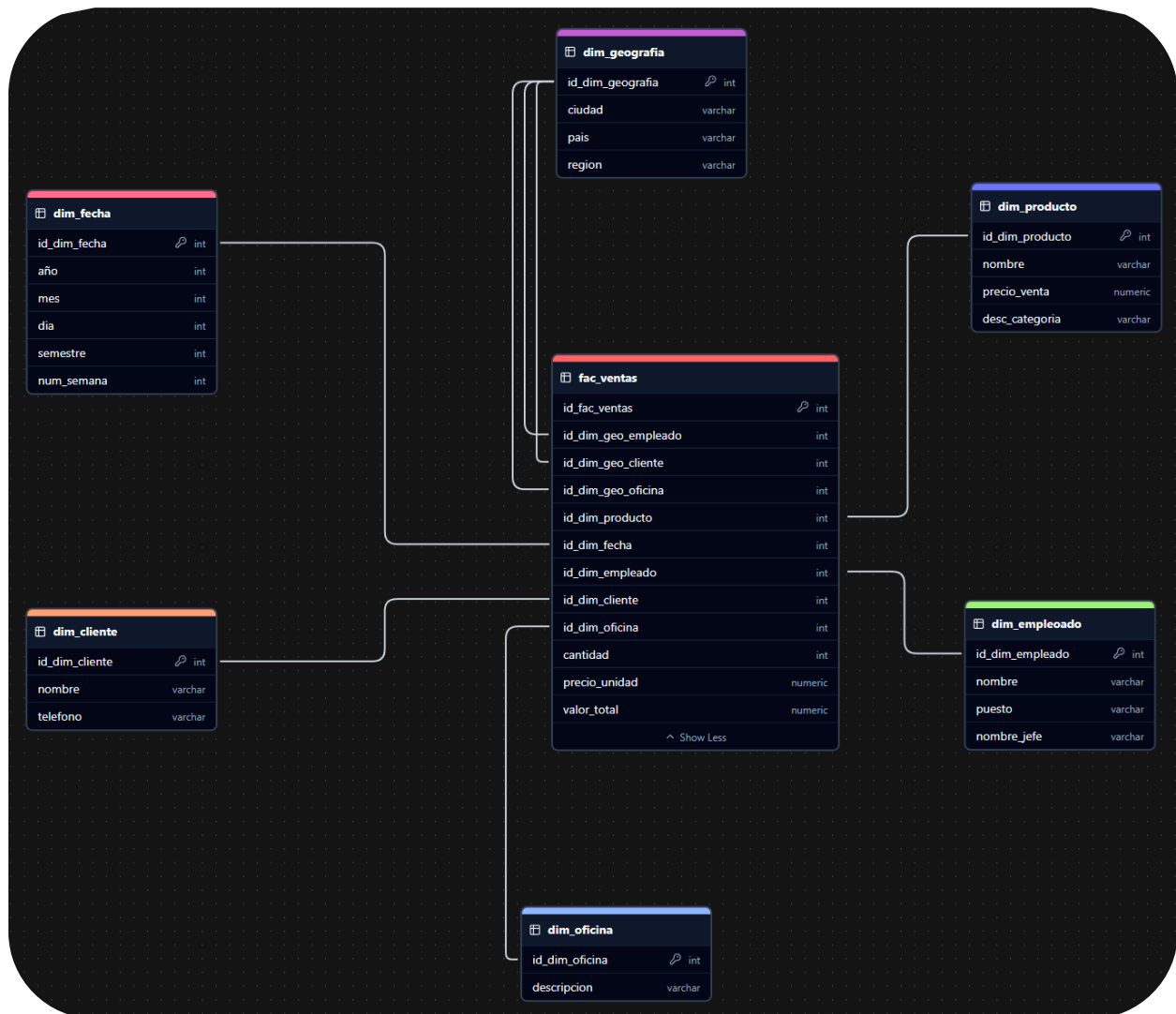
- id_dim_oficina (PK)
- descripción

Tabla de cliente: Contendrá la información del cliente.

Campos de la dimensión *dim_cliente*:

- id_dim_cliente (PK)
- nombre
- telefono

Diseño (Imagen) Del Modelo Estrella



Lista De Dimensiones Propuestas.

Tabla de fecha: Contendrá las fechas en las que se realizaron las ventas.

Campos de la dimensión *dim_fecha*:

- id_dim_fecha INT - PK
- año INT
- mes INT
- día INT
- semestre INT
- num_semana INT

Tabla de geografía: Contendrá la información geográfica de los clientes, los empleados y la oficina.

Campos de la dimensión *dim_geografia*:

- id_dim_geografia INT - PK
- ciudad VARCHAR(255)
- país VARCHAR(255)
- región VARCHAR(255)

Tabla de producto: Contendrá la información del nombre del producto, la descripción de la categoría y el precio.

Campos de la dimensión *dim_producto*:

- id_dim_producto PK
- nombre VARCHAR(255)
- precio_venta NUMERIC(18,2)
- desc_categoria VARCHAR(255)

Tabla de empleado: Contendrá información del empleado.

Campos de la dimensión *dim_empleado*:

- id_dim_empleado PK
- nombre VARCHAR(255)
- puesto VARCHAR(255)
- nombre_jefe VARCHAR(255)

Tabla de oficina: Contendrá las fechas en las que se realizaron las ventas.

Campos de la dimensión *dim_oficina*:

- id_dim_oficina PK
- descripción VARCHAR(255)

Tabla de cliente: Contendrá la información del cliente.

Campos de la dimensión *dim_cliente*:

- id_dim_cliente PK
- nombre VARCHAR(255)
- teléfono VARCHAR(255)

Tabla De Hechos Propuesta

fac_ventas: Esta tabla contendrá la información acerca de las ventas realizadas en la empresa.

Campos de la tabla de hechos:

- id_fac_ventas INT - PK
- id_dim_geo_empleado INT - FK con la tabla dim_geografia
- id_dim_geo_cliente INT - FK con la tabla dim_geografia
- id_dim_geo_oficina INT - FK con la tabla dim_geografia
- id_dim_producto INT - FK con la tabla dim_producto
- id_dim_fecha INT - FK con la tabla dim_fecha
- id_dim_empleado INT - FK con la tabla dim_empleado
- id_dim_cliente INT - FK con la tabla dim_cliente
- id_dim_oficina INT - FK con la tabla dim_oficina
- cantidad INT
- precio_unidad NUMERIC(18,2)
- valor_total NUMERIC(18,2)

Descripción del análisis realizado a los datos Jardinería y cómo estos se trasladaron a la base de datos Staging.

Una vez se tuvo acceso a la información de la base de datos Jardinería, se pudo definir cuáles eran los datos estrictamente necesarios para aplicar la nueva arquitectura Staging y se detectaron valores que llegaban NULL y otras inconsistencias que no permitieron la creación de las claves foráneas.

Los datos geográficos estaban repartidos en varias tablas, por lo tanto, se consolidaron en una sola quitando la duplicidad de la información.

Otro proceso que es similar a este, son las fechas, centralizando la información en una sola tabla y refiriéndose a esta con un id, lo que generaría más optimización de almacenamiento.

Se logra también totalizar el valor de las ventas para facilitar la información desplegada al usuario y evitar posibles errores a la hora de consultar la información.

El proceso para la creación de la base de datos fue el siguiente:

- Se basó en el modelo estrella e identificamos cuál era la tabla que tenía los datos origen.
- Se elaboró el SQL para la creación e inserción de los datos de la tabla.
- Posteriormente se valida que la información destino correspondiera al origen.

Este proceso se repite para cada una de las tablas del modelo estrella.

Imágenes con las sentencias SQL para la creación de las tablas staging


```
1  /* crear tabla de fac_ventas_st */
2
3  DROP TABLE IF EXISTS fac_ventas_st
4
5  CREATE TABLE fac_ventas_st (
6      id_fac_ventas INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
7      id_dim_empleado INT,
8      id_dim_fecha INT,
9      id_dim_oficina INT,
10     id_dim_cliente INT,
11     id_dim_geo_empleado INT,
12     id_dim_geo_cliente INT,
13     id_dim_geo_oficina INT,
14     id_dim_producto INT,
15     cantidad INT,
16     precio_unidad NUMERIC(18, 2),
17     valor_total NUMERIC(18, 2)
18 );
```

```
1  /* crear tabla de dim_fecha_st*/
2
3  DROP TABLE IF EXISTS dim_fecha_st
4
5  CREATE TABLE dim_fecha_st (
6      id_dim_fecha INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
7      fecha date,
8      año INT,
9      mes INT,
10     día INT,
11     semestre INT,
12     num_semana INT
13 );
```


```
1  /* crear tabla de dim_geografia_st*/
2
3  DROP TABLE IF EXISTS dim_geografia_st
4
5  CREATE TABLE dim_geografia_st (
6      id_dim_geografia INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
7      ciudad VARCHAR(255),
8      pais VARCHAR(255),
9      region VARCHAR(255)
10 );
```

```
1  /* crear tabla de dim_producto_st*/
2
3  DROP TABLE IF EXISTS dim_producto_st
4
5  CREATE TABLE dim_producto_st (
6      id_dim_producto INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
7      nombre VARCHAR(255),
8      precio_venta NUMERIC(18, 2),
9      desc_categoria VARCHAR(255)
10 );
```

```
1  /* crear tabla de dim_empleado_st*/
2
3  DROP TABLE IF EXISTS dim_empleado_st
4
5  CREATE TABLE dim_empleado_st (
6      id_dim_empleado INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
7      nombre VARCHAR(255),
8      puesto VARCHAR(255),
9      nombre_jefe VARCHAR(255)
10 );
```



```
1  /* crear tabla de dim_oficina_st*/
2
3  DROP TABLE IF EXISTS dim_oficina_st
4
5  CREATE TABLE dim_oficina_st (
6      id_dim_oficina INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
7      descripcion VARCHAR(255)
8  );
```



```
1  /* crear tabla de dim_cliente_st*/
2
3  DROP TABLE IF EXISTS dim_cliente_st
4
5  CREATE TABLE dim_cliente_st (
6      id_dim_cliente INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
7      nombre VARCHAR(255),
8      telefono VARCHAR(255)
9  );
```

Imágenes con las sentencias SQL para la inserción de la información en staging

```
1  /* Inserción de las fechas a la tabla dim_fecha_st */
2
3  TRUNCATE TABLE dim_fecha_st
4
5  INSERT INTO dim_fecha_st
6      (fecha
7       ,año
8       ,mes
9       ,dia
10      ,semestre
11      ,num_semana)
12
13  SELECT DISTINCT
14      Jardineria.dbo.pedido.fecha_pedido,
15      YEAR(Jardineria.dbo.pedido.fecha_pedido) AS Año,
16      MONTH(Jardineria.dbo.pedido.fecha_pedido) AS Mes,
17      DAY(Jardineria.dbo.pedido.fecha_pedido) AS Dia,
18      DATEPART(WEEK, Jardineria.dbo.pedido.fecha_pedido) AS NumeroDeSemana,
19      DATEPART(QUARTER, fecha_pedido) AS NumeroDeSemestre
20  FROM Jardineria.dbo.pedido
21
22  GO
```

```
1  /* Inserción de las ubicaciones a la tabla dim_geografia_st*/
2
3  TRUNCATE TABLE dim_geografia_st
4
5  INSERT INTO dim_geografia_st
6      (ciudad
7       ,pais
8       ,region)
9
10  SELECT DISTINCT ciudad, pais, region FROM Jardineria.dbo.cliente
11  UNION
12  SELECT DISTINCT ciudad, pais, region FROM Jardineria.dbo.oficina
13
14  GO
15
```

```
1  /* Inserción de los productos a la tabla dim_producto_st */
2
3  TRUNCATE TABLE dim_producto_st
4
5  INSERT INTO dim_producto_st
6      (nombre
7       ,precio_venta
8       ,desc_categoria)
9
10 SELECT Jardineria.dbo.producto.nombre,
11        Jardineria.dbo.producto.precio_venta,
12        Jardineria.dbo.producto.descripcion
13 FROM Jardineria.dbo.producto;
14
15 GO
```

```
1  /* Inserción de los empleados a la tabla dim_empleado_st */
2
3  TRUNCATE TABLE dim_empleado_st
4  SET IDENTITY_INSERT dim_empleado_st ON;
5
6  INSERT INTO dim_empleado_st
7      (id_dim_empleado
8       ,nombre
9       ,puesto
10      ,nombre_jefe)
11
12 SELECT e.ID_empleado
13        ,e.nombre + ' ' + e.apellido1 + ' ' + e.apellido2 AS nombre_empleado
14        ,e.puesto, j.nombre + ' ' + j.apellido1 + ' ' + j.apellido2 AS nombre_jefe
15 FROM Jardineria.dbo.empleado e
16 LEFT JOIN Jardineria.dbo.empleado j
17 ON e.ID_jefe = j.ID_empleado;
18
19 GO
```

```
1  /* Inserción de los empleados a la tabla dim_empleado_st */
2
3  TRUNCATE TABLE dim_empleado_st
4  SET IDENTITY_INSERT dim_empleado_st ON;
5
6  INSERT INTO dim_empleado_st
7      (id_dim_empleado
8       ,nombre
9       ,puesto
10      ,nombre_jefe)
11
12  SELECT e.ID_empleado
13         ,e.nombre + ' ' + e.apellido1 + ' ' + e.apellido2 AS nombre_empleado
14         ,e.puesto, j.nombre + ' ' + j.apellido1 + ' ' + j.apellido2 AS nombre_jefe
15  FROM Jardineria.dbo.empleado e
16  LEFT JOIN Jardineria.dbo.empleado j
17  ON e.ID_jefe = j.ID_empleado;
18
19  GO
```

```
1  /* Inserción de las oficinas a la tabla dim_oficina_st */
2
3  TRUNCATE TABLE dim_oficina_st
4
5  INSERT INTO dim_oficina_st
6      (descripcion)
7
8  SELECT Jardineria.dbo.oficina.Descripcion
9  FROM Jardineria.dbo.oficina
10
11  GO
```

```
1  /* Inserción de los clientes a la tabla dim_cliente_st */
2
3
4  INSERT INTO dim_cliente_st
5      (nombre
6       ,telefono)
7
8  SELECT Jardineria.dbo.cliente.nombre_cliente
9         ,Jardineria.dbo.cliente.telefono
10  FROM Jardineria.dbo.cliente
11
12  GO
```

```

1  /* Inserción de las ventas a la tabla fac_ventas*/
2
3  INSERT INTO fac_ventas_st
4      (id_dim_empleado
5       ,id_dim_fecha
6       ,id_dim_oficina
7       ,id_dim_cliente
8       ,id_dim_geo_empleado
9       ,id_dim_geo_cliente
10      ,id_dim_geo_oficina
11      ,id_dim_producto
12      ,cantidad
13      ,precio_unidad
14      ,valor_total)
15
16  SELECT
17      c.ID_empleado_rep_ventas AS id_empleado,
18      f.id_dim_fecha,
19      e.ID_oficina,
20      p.ID_cliente,
21      gof.id_dim_geografia AS id_geo_empleado,
22      g.id_dim_geografia AS id_geo_cliente,
23      gof.id_dim_geografia AS id_geo_oficina,
24      dp.ID_producto,
25      dp.cantidad,
26      dp.precio_unidad,
27      (dp.cantidad * dp.precio_unidad) AS valor_total
28  FROM
29      Jardineria.dbo.pedido p
30  JOIN
31      Jardineria.dbo.detalle_pedido dp ON p.ID_pedido = dp.ID_pedido
32  JOIN
33      Jardineria.dbo.cliente c ON p.ID_cliente = c.ID_cliente
34  JOIN
35      StagingJardineria.dbo.dim_fecha_st f ON p.fecha_pedido = f.fecha
36  JOIN
37      Jardineria.dbo.empleado e ON c.ID_empleado_rep_ventas = e.ID_empleado
38  LEFT JOIN
39      StagingJardineria.dbo.dim_geografia_st g ON c.pais = g.pais AND c.region = g.region AND c.ciudad = g.ciudad
40  JOIN
41      Jardineria.dbo.oficina o ON o.ID_oficina = e.ID_oficina
42  LEFT JOIN
43      StagingJardineria.dbo.dim_geografia_st gof ON o.pais = gof.pais AND o.region = gof.region AND o.ciudad = gof.ciudad
44  ORDER BY
45      3;
46
47  GO

```

Imágenes con las sentencias SQL para la transformación de la información

```
1  /* Transformación cuando el nombre del jefe es NULL */
2
3  SELECT id_dim_empleado
4         ,nombre
5         ,puesto
6         ,ISNULL(nombre_jefe, 'Sin nombre de jefe') AS nombre_jefe
7  FROM StagingJardineria.dbo.dim_empleado_st
```

```
1  /* Normalización de nombres de ciudades y países */
2
3  UPDATE dbo.dim_geografia_st
4  SET ciudad = 'Londres'
5  WHERE ciudad IN('London')
6  GO
7
8
9  UPDATE dbo.dim_geografia_st
10 SET pais = 'España'
11 WHERE pais IN('Spain')
12 GO
13
14
15 UPDATE dbo.dim_geografia_st
16 SET pais = 'Estados Unidos'
17 WHERE pais IN('EEUU', 'USA')
18 GO
19
20
21 /* Transformación cuando la región es NULL */
22
23 SELECT id_dim_geografia
24        ,ciudad
25        ,pais
26        ,ISNULL(region, 'Sin región') AS region
27 FROM StagingJardineria.dbo.dim_geografia_st
```



```
1  /* Transformación de todos los datos de la columna nombre a minúscula */
2  /* Transformación de las descripciones vacías */
3
4  SELECT id_dim_producto
5         ,LOWER(nombre)
6         ,precio_venta
7         ,(COALESCE(NULLIF(desc_categoria, ''), 'Sin descripción')) AS desc_categoria
8  FROM StagingJardineria.dbo.dim_producto_st
9
```

```
1  /* Transformación cuando el id_dim_geo_cliente es NULL */
2
3  SELECT id_fac_ventas
4         ,id_dim_empleado
5         ,id_dim_fecha
6         ,id_dim_oficina
7         ,id_dim_cliente
8         ,id_dim_geo_empleado
9         ,ISNULL(id_dim_geo_cliente, -999) AS id_dim_geo_cliente
10        ,id_dim_geo_oficina
11        ,id_dim_producto
12        ,cantidad
13        ,precio_unidad
14        ,valor_total
15  FROM StagingJardineria.dbo.fac_ventas_st
```

Imágenes con las sentencias SQL para la creación de las tablas datamart


```
1  /* crear tabla de fac_ventas */
2
3  DROP TABLE IF EXISTS fac_ventas
4
5  CREATE TABLE fac_ventas (
6      id_fac_ventas INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
7      id_dim_empleado INT,
8      id_dim_fecha INT,
9      id_dim_oficina INT,
10     id_dim_cliente INT,
11     id_dim_geo_empleado INT,
12     id_dim_geo_cliente INT,
13     id_dim_geo_oficina INT,
14     id_dim_producto INT,
15     cantidad INT,
16     precio_unidad NUMERIC(18, 2),
17     valor_total NUMERIC(18, 2)
18 );
```

```
1  /* crear tabla de dim_fecha*/
2
3  DROP TABLE IF EXISTS dim_fecha
4
5  CREATE TABLE dim_fecha (
6      id_dim_fecha INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
7      fecha date,
8      año INT,
9      mes INT,
10     dia INT,
11     semestre INT,
12     num_semana INT
13 );
```


```
1  /* crear tabla de dim_geografia*/  
2  
3  DROP TABLE IF EXISTS dim_geografia  
4  
5  CREATE TABLE dim_geografia (  
6      id_dim_geografia INT IDENTITY(1,1) NOT NULL PRIMARY KEY,  
7      ciudad VARCHAR(255),  
8      pais VARCHAR(255),  
9      region VARCHAR(255)  
10 );
```

```
1  /* crear tabla de dim_producto*/  
2  
3  DROP TABLE IF EXISTS dim_producto  
4  
5  CREATE TABLE dim_producto (  
6      id_dim_producto INT IDENTITY(1,1) NOT NULL PRIMARY KEY,  
7      nombre VARCHAR(255),  
8      precio_venta NUMERIC(18, 2),  
9      desc_categoria VARCHAR(MAX)  
10 );
```

```
1  /* crear tabla de dim_empleado*/  
2  
3  DROP TABLE IF EXISTS dim_empleado  
4  
5  CREATE TABLE dim_empleado (  
6      id_dim_empleado INT IDENTITY(1,1) NOT NULL PRIMARY KEY,  
7      nombre VARCHAR(255),  
8      puesto VARCHAR(255),  
9      nombre_jefe VARCHAR(255)  
10 );
```



```
1  /* crear tabla de dim_oficina*/  
2  
3  DROP TABLE IF EXISTS dim_oficina  
4  
5  CREATE TABLE dim_oficina (  
6      id_dim_oficina INT IDENTITY(1,1) NOT NULL PRIMARY KEY,  
7      descripcion VARCHAR(255)  
8  );
```



```
1  /* crear tabla de dim_cliente*/  
2  
3  DROP TABLE IF EXISTS dim_cliente  
4  
5  CREATE TABLE dim_cliente (  
6      id_dim_cliente INT IDENTITY(1,1) NOT NULL PRIMARY KEY,  
7      nombre VARCHAR(255),  
8      telefono VARCHAR(255)  
9  );
```

Imágenes con las sentencias SQL para la inserción de la información en datamart

```
1  /* Inserción de las fechas a la tabla dim_fecha */
2
3  TRUNCATE TABLE dim_fecha
4
5  INSERT INTO dim_fecha
6      (fecha
7       ,año
8       ,mes
9       ,dia
10      ,semestre
11      ,num_semana)
12
13  SELECT
14      StagingJardineria.dbo.dim_fecha_st.fecha,
15      StagingJardineria.dbo.dim_fecha_st.año,
16      StagingJardineria.dbo.dim_fecha_st.mes,
17      StagingJardineria.dbo.dim_fecha_st.dia,
18      StagingJardineria.dbo.dim_fecha_st.semestre,
19      StagingJardineria.dbo.dim_fecha_st.num_semana
20  FROM StagingJardineria.dbo.dim_fecha_st
21
22  GO
```

```
1  /* Inserción de las ubicaciones a la tabla dim_geografia*/
2  TRUNCATE TABLE dim_geografia
3
4  INSERT INTO dim_geografia
5      (ciudad
6       ,pais
7       ,region)
8
9  SELECT ciudad
10     ,pais
11     ,ISNULL(region, 'Sin región')
12  FROM StagingJardineria.dbo.dim_geografia_st
13
14  GO
```

```
1  /* Inserción de los productos a la tabla dim_producto */
2
3  TRUNCATE TABLE dim_producto
4
5  INSERT INTO dim_producto
6      (nombre
7       ,precio_venta
8       ,desc_categoria)
9
10 SELECT LOWER(nombre)
11        ,precio_venta
12        ,(COALESCE(NULLIF(desc_categoria, ''), 'Sin descripción')) AS desc_categoria
13 FROM StagingJardineria.dbo.dim_producto_st
14
15 GO
```

```
1  /* Inserción de los empleados a la tabla dim_empleado */
2
3  TRUNCATE TABLE dim_empleado
4  SET IDENTITY_INSERT dim_empleado ON;
5
6  INSERT INTO dim_empleado
7      (id_dim_empleado
8       ,nombre
9       ,puesto
10      ,nombre_jefe)
11
12 SELECT id_dim_empleado
13        ,nombre
14        ,puesto
15        ,ISNULL(nombre_jefe, 'Sin nombre de jefe') AS nombre_jefe
16 FROM StagingJardineria.dbo.dim_empleado_st
17
18 GO
```



```
1  /* Inserción de las oficinas a la tabla dim_oficina */
2
3  TRUNCATE TABLE dim_oficina
4
5  INSERT INTO dim_oficina
6      (descripcion)
7
8  SELECT descripcion
9  FROM StagingJardineria.dbo.dim_oficina_st
10
11  GO
```



```
1  /* Inserción de los clientes a la tabla dim_cliente */
2
3  TRUNCATE TABLE dim_cliente
4
5  INSERT INTO dim_cliente
6      (nombre
7       ,telefono)
8
9  SELECT nombre, telefono
10 FROM StagingJardineria.dbo.dim_cliente_st
11
12  GO
```

```
1  /* Inserción de las ventas a la tabla fac_ventas*/
2
3  TRUNCATE TABLE fac_ventas
4
5  INSERT INTO fac_ventas
6      (id_dim_empleado
7       ,id_dim_fecha
8       ,id_dim_oficina
9       ,id_dim_cliente
10      ,id_dim_geo_empleado
11      ,id_dim_geo_cliente
12      ,id_dim_geo_oficina
13      ,id_dim_producto
14      ,cantidad
15      ,precio_unidad
16      ,valor_total)
17
18  SELECT id_dim_empleado
19      ,id_dim_fecha
20      ,id_dim_oficina
21      ,id_dim_cliente
22      ,id_dim_geo_empleado
23      ,ISNULL(id_dim_geo_cliente, -999) AS id_dim_geo_cliente
24      ,id_dim_geo_oficina
25      ,id_dim_producto
26      ,cantidad
27      ,precio_unidad
28      ,valor_total
29  FROM
30      StagingJardineria.dbo.fac_ventas_st
31
32  GO
```


Imágenes con las sentencias SQL para la validación de datos

Tabla cliente

```
1 /* Validaciones de la tabla cliente: número de registros */  
2 SELECT COUNT(*) AS cantidad_dm FROM DatamartJardineria.dbo.dim_cliente  
3 SELECT COUNT(*) AS cantidad_dm FROM StagingJardineria.dbo.dim_empleado_st
```

	cantidad_dm
1	72

	cantidad_dm
1	72

Tabla empleado

```
1 /* Validaciones de la tabla empleado: número de registros */  
2 SELECT COUNT(*) AS cantidad_dm FROM DatamartJardineria.dbo.dim_empleado  
3 SELECT COUNT(*) AS cantidad_dm FROM StagingJardineria.dbo.dim_empleado_st
```

	cantidad_dm
1	31

	cantidad_dm
1	31

Tabla fecha

```
1  /* Validaciones de la tabla fecha: número de registros */  
2  SELECT COUNT(*) AS cantidad_dm FROM DatamartJardineria.dbo.dim_fecha  
3  SELECT COUNT(*) AS cantidad_st FROM StagingJardineria.dbo.dim_fecha_st
```

	cantidad_dm
1	78

	cantidad_st
1	78

Tabla geografía

```
1  /* Validaciones de la tabla geografía: número de registros */  
2  SELECT COUNT(*) AS cantidad_st FROM DatamartJardineria.dbo.dim_geografia  
3  SELECT COUNT(*) AS cantidad_st FROM StagingJardineria.dbo.dim_geografia_st  
4
```

	cantidad_st
1	27

	cantidad_dm
1	27

Tabla oficina

```
1  /* Validaciones de la tabla oficina: número de registros */  
2  SELECT COUNT(*) AS cantidad_st FROM DatamartJardineria.dbo.dim_oficina  
3  SELECT COUNT(*) AS cantidad_st FROM StagingJardineria.dbo.dim_oficina_st
```

	cantidad_st
1	9

	cantidad_st
1	9

Tabla producto

```
1  /* Validaciones de la tabla producto: número de registros */  
2  SELECT COUNT(*) AS cantidad_st FROM DatamartJardineria.dbo.dim_producto  
3  SELECT COUNT(*) AS cantidad_st FROM StagingJardineria.dbo.dim_producto_st
```

	cantidad_st
1	276

	cantidad_st
1	276

Tabla ventas

```

1  /* Validaciones de la tabla ventas: número de registros */
2  SELECT COUNT(*) AS cantidad_dm
3      ,SUM(cantidad) AS unidades_vendidas_dm
4      ,SUM(cantidad * precio_unidad) AS valor_total_venta_dm
5  FROM DatamartJardineria.dbo.fac_ventas
6
7  SELECT COUNT(*) AS cantidad_st
8      ,SUM(cantidad) AS unidades_vendidas_st
9      ,SUM(cantidad * precio_unidad) AS valor_total_venta_st
10 FROM StagingJardineria.dbo.fac_ventas_st

```

	cantidad_dm	unidades_vendidas_dm	valor_total_venta_dm
1	636	17080	435476.00

	cantidad_st	unidades_vendidas_st	valor_total_venta_st
1	636	17080	435476.00

Producto más vendido

```

1  /* Productos con mas ventas en la tabla de datamart, staging y jardineria */
2
3  SELECT TOP 1
4      id_dim_producto
5      ,SUM(cantidad) AS cantidad_dm
6  FROM DatamartJardineria.dbo.fac_ventas
7  GROUP BY id_dim_producto
8  ORDER BY cantidad_dm DESC
9
10 SELECT TOP 1
11     id_dim_producto
12     ,SUM(cantidad) AS cantidad_st
13 FROM StagingJardineria.dbo.fac_ventas_st
14 GROUP BY id_dim_producto
15 ORDER BY cantidad_st DESC
16
17 SELECT TOP 1
18     dp.ID_producto,
19     SUM(dp.cantidad) AS cantidad_jard
20 FROM
21     Jardineria.dbo.pedido p
22 JOIN
23     Jardineria.dbo.detalle_pedido dp ON p.ID_pedido = dp.ID_pedido
24 JOIN
25     Jardineria.dbo.cliente c ON p.ID_cliente = c.ID_cliente
26 JOIN
27     StagingJardineria.dbo.dim_fecha_st f ON p.fecha_pedido = f.fecha
28 JOIN
29     Jardineria.dbo.empleado e ON c.ID_empleado_rep_ventas = e.ID_empleado
30 LEFT JOIN
31     StagingJardineria.dbo.dim_geografia_st g ON c.pais = g.pais AND c.region = g.region AND c.ciudad = g.ciudad
32 JOIN
33     Jardineria.dbo.oficina o ON o.ID_oficina = e.ID_oficina
34 LEFT JOIN
35     StagingJardineria.dbo.dim_geografia_st gof ON o.pais = gof.pais AND o.region = gof.region AND o.ciudad = gof.ciudad
36 GROUP BY ID_producto
37 ORDER BY cantidad_jard DESC;
38
39

```

	id_dim_producto	cantidad_dm
1	13	967

	id_dim_producto	cantidad_st
1	13	967

	ID_producto	cantidad_jard
1	13	967

Conclusiones

El modelo estrella permite simplificar las consultas, ya que al tener menos tablas y menos relaciones no es necesario aplicar muchos JOINS al momento de realizar la sentencia sql para obtener la información deseada.

Las dimensiones permiten organizar fácilmente el análisis desde diferentes perspectivas, ya que contiene los datos organizados.

Los modelos de datos en estrella son una manera de organizar la información, utilizada en el análisis de datos, con una estructura simple, clara, conformada por una tabla de hechos y *n* cantidad de tablas denominadas dimensiones que aportan contexto a la tabla principal.

En comparación de los modelos relacionales, donde siempre suelen estar normalizados los datos para evitar redundancias, los modelos en estrella no siguen estas reglas, buscando un mejor rendimiento y comprensión.

Sobre la arquitectura Staging se puede decir que mejora la gestión de los datos debido a que es una capa intermedia entre el procesamiento y la transformación de la información, permitiendo la reducción en las inconsistencias a través del análisis y la limpieza de estos.

Esta arquitectura optimiza el proceso de ETL para el manejo de grandes volúmenes de información, disminuyendo los posibles errores durante la transferencia.

En la transformación de los datos se modificó los valores NULL y se normalizaron nombres de algunos países y ciudades, además de aquellos campos que aparecían vacíos.

Se analizó la coherencia de los datos y los valores albergados.

Se realizó la validación de los siguientes datos de cada tabla:

- Números de registros fuesen igual a la fuente
- Cantidad de unidades vendidas.
- Valor de la venta.
- Producto más vendido.

Bibliografía

Atlan. (18 de Diciembre de 2023). *Data Staging Area Uncovered: From Basics to Best Practices*.

Obtenido de Atlan: <https://atlan.com/what-is/data-staging-area/>

Microsoft. (22 de Marzo de 2023). *Descripción de un esquema de estrella e importancia para*

Power BI. Obtenido de Microsoft: <https://learn.microsoft.com/es-es/power-bi/guidance/star-schema>

OpenIA. (01 de Septiembre de 2024). Obtenido de ChatGPT: <https://chatgpt.com/>