

EVIDENCIA DE APRENDIZAJE

**PROCESO DE TRANSFORMACIÓN DE DATOS Y CARGA EN EL DATA MART
FINAL**

JUAN DAVID CALLE CORREA

EDISON MENA ARROYO

MARIA FERNANDA VILLADA QUINTERO

PROGRAMA

INGENIERÍA DE SOFTWARE Y DATOS

UNIVERSIDAD DIGITAL DE ANTIOQUIA

PREICA2502B010064

CURSO

BASES DE DATOS II

DOCENTE

ANTONIO JESÚS VALDERRAMA

2025-02

Resumen del Trabajo:

Proceso de Transformación de Datos y Carga en el Data Mart Final

Este proyecto tiene como objetivo principal diseñar e implementar un pipeline de integración de datos que permita transformar información operativa de la base de datos transaccional Jardinería en un entorno analítico estructurado, listo para soportar consultas de inteligencia de negocios. La solución se basa en un flujo ETL (Extract, Transform, Load) desarrollado íntegramente en T-SQL, siguiendo una arquitectura en tres capas: origen, staging y destino.

La capa de staging (Staging Jardinería) actúa como zona intermedia donde se almacenan los datos extraídos sin procesamiento, garantizando la estabilidad del sistema fuente.

Posteriormente, en la fase de transformación, se aplican operaciones de limpieza (manejo de valores nulos mediante sustitución controlada), normalización (homogenización de formatos de fechas y cadenas) y enriquecimiento (generación de una dimensión de tiempo completa y cálculo de métricas derivadas como el valor total por pedido).

El modelo lógico de la data mart final (Data Mart Jardinería) se implementó bajo el esquema estrella, compuesto por dimensiones de cliente, producto, categoría y tiempo, junto con una tabla de hechos central que registra las transacciones comerciales. La carga de datos se realizó respetando rigurosamente las restricciones de integridad referencial, asegurando la coherencia del modelo dimensional.

Como resultado, se logró un entorno analítico funcional desde el cual se ejecutan consultas OLAP para responder preguntas estratégicas, tales como: identificación del producto más vendido por volumen y valor, análisis de tendencias temporales de ventas y segmentación de clientes por actividad comercial. Este trabajo no solo valida la viabilidad de construir

soluciones de inteligencia de negocios con tecnologías nativas de SQL Server, sino que también refuerza principios fundamentales de arquitectura de datos, calidad de la información y modelado dimensional.

Preparación: Verificación de la Estructura de Staging

Object Explorer: DAVID-CALLE (SQL Server 16.0.1)

- Databases
 - System Databases
 - Database Snapshots
 - CentroMedico
 - CentroMedico1
 - DataMartJardineria
 - Jardineria
 - StagingJardineria
- Security
- Server Objects
- Replication
- Always On High Availability
- Management
- Integration Services Catalogs
- SQL Server Agent
- XEvent Profiler

SQLQuery7.sql - DA...E\David_calle (66))*

```
USE StagingJardineria; -- cambia por el nombre de tu BD STAGING
GO

SELECT TABLE_SCHEMA, TABLE_NAME
FROM INFORMATION_SCHEMA.TABLES
WHERE TABLE_TYPE = 'BASE TABLE';
```

Results

	TABLE_SCHEMA	TABLE_NAME
1	dbo	StgCliente
2	dbo	StgProducto
3	dbo	StgCategoria
4	dbo	StgPedido
5	dbo	StgDetallePedido
6	dbo	StgPago
7	dbo	sysdiagrams

Object Explorer: DAVID-CALLE (SQL Server 16.0.1)

- Databases
 - System Databases
 - Database Snapshots
 - CentroMedico
 - CentroMedico1
 - DataMartJardineria
 - Jardineria
 - StagingJardineria
- Security
- Server Objects
- Replication
- Always On High Availability
- Management
- Integration Services Catalogs
- SQL Server Agent
- XEvent Profiler

SQLQuery7.sql - DA...E\David_calle (66))*

```
USE StagingJardineria; -- cambia por el nombre de tu BD STAGING
GO

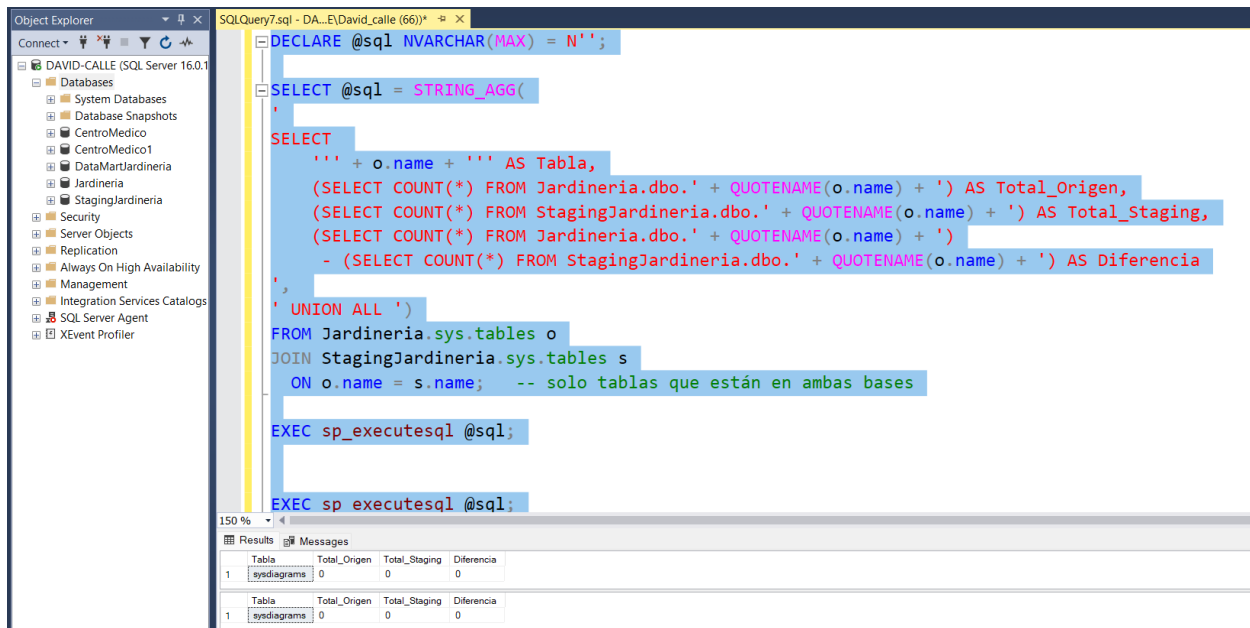
SELECT TABLE_SCHEMA, TABLE_NAME
FROM INFORMATION_SCHEMA.TABLES
WHERE TABLE_TYPE = 'BASE TABLE';

SELECT TABLE_SCHEMA, TABLE_NAME, COLUMN_NAME, DATA_TYPE, CHARACTER_MAXIMUM_LENGTH, IS_NULLABLE
FROM INFORMATION_SCHEMA.COLUMNS
WHERE TABLE_SCHEMA = 'dbo' -- o el esquema que uses
ORDER BY TABLE_NAME, ORDINAL_POSITION;
```

Results

	TABLE_SCHEMA	TABLE_NAME	COLUMN_NAME	DATA_TYPE	CHARACTER_MAXIMUM_LENGTH	IS_NULLABLE
1	dbo	StgCategoria	CategoriaID	int	NULL	NO
2	dbo	StgCategoria	NombreCategoria	nvarchar	100	YES
3	dbo	StgCliente	ID_Cliente	int	NULL	NO
4	dbo	StgCliente	Nombre	nvarchar	100	YES
5	dbo	StgCliente	Ciudad	nvarchar	50	YES
6	dbo	StgCliente	País	nvarchar	50	YES
7	dbo	StgDetallePedido	PedidoID	int	NULL	YES
8	dbo	StgDetallePedido	ProductoID	int	NULL	YES
9	dbo	StgDetallePedido	Cantidad	int	NULL	YES
10	dbo	StgDetallePedido	PrecioUnidad	decimal	NULL	YES
11	dbo	StgPago	PagoID	int	NULL	NO
12	dbo	StgPago	ClienteID	int	NULL	YES
13	dbo	StgPago	FechaPago	date	NULL	YES
14	dbo	StgPago	Total	decimal	NULL	YES
15	dbo	StgPedido	PedidoID	int	NULL	NO
16	dbo	StgPedido	ClienteID	int	NULL	YES

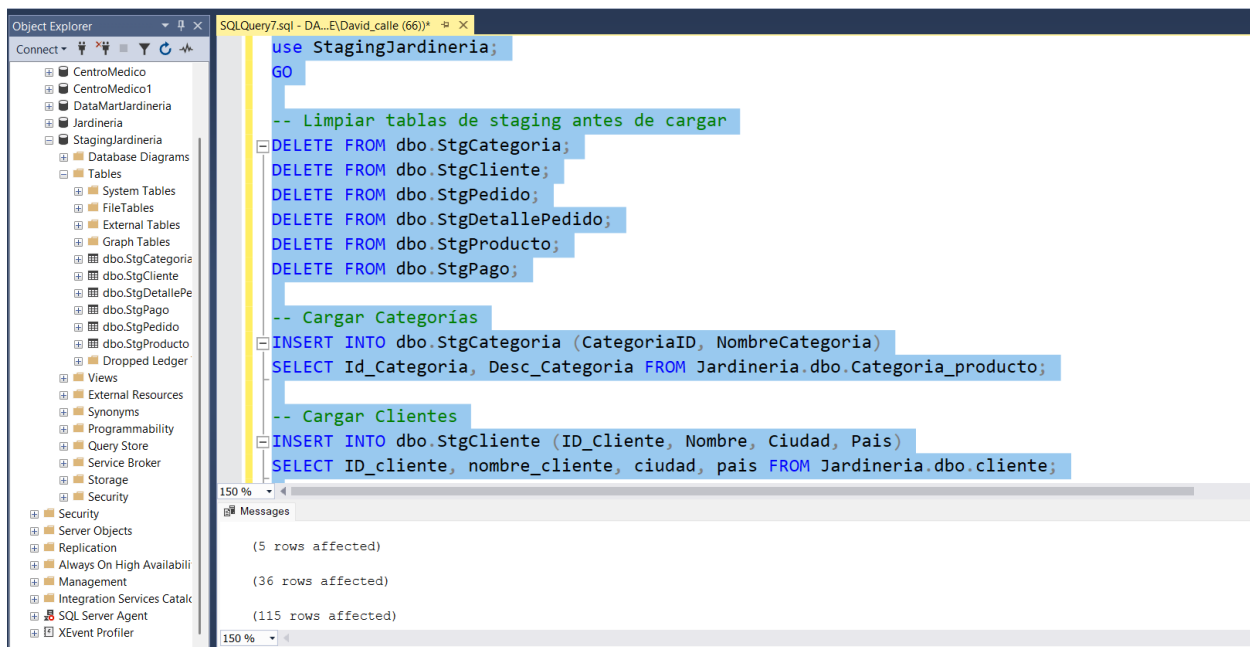
Extracción desde origen (jardinería) hacia staging (staging jardinería)



```
DECLARE @sql NVARCHAR(MAX) = N'';
SELECT @sql = STRING_AGG(
    SELECT
        ''' + o.name + ''' AS Tabla,
        (SELECT COUNT(*) FROM Jardineria.dbo.' + QUOTENAME(o.name) + ') AS Total_Origen,
        (SELECT COUNT(*) FROM StagingJardineria.dbo.' + QUOTENAME(o.name) + ') AS Total_Staging,
        (SELECT COUNT(*) FROM Jardineria.dbo.' + QUOTENAME(o.name) + ')
        - (SELECT COUNT(*) FROM StagingJardineria.dbo.' + QUOTENAME(o.name) + ') AS Diferencia
    FROM Jardineria.sys.tables o
    JOIN StagingJardineria.sys.tables s
    ON o.name = s.name; -- solo tablas que están en ambas bases
)
UNION ALL ''
EXEC sp_executesql @sql;
EXEC sp_executesql @sql;
```

Tabla	Total_Origen	Total_Staging	Diferencia
sysdiagrams	0	0	0

Script de Carga desde Jardinería hacia Staging Jardinería



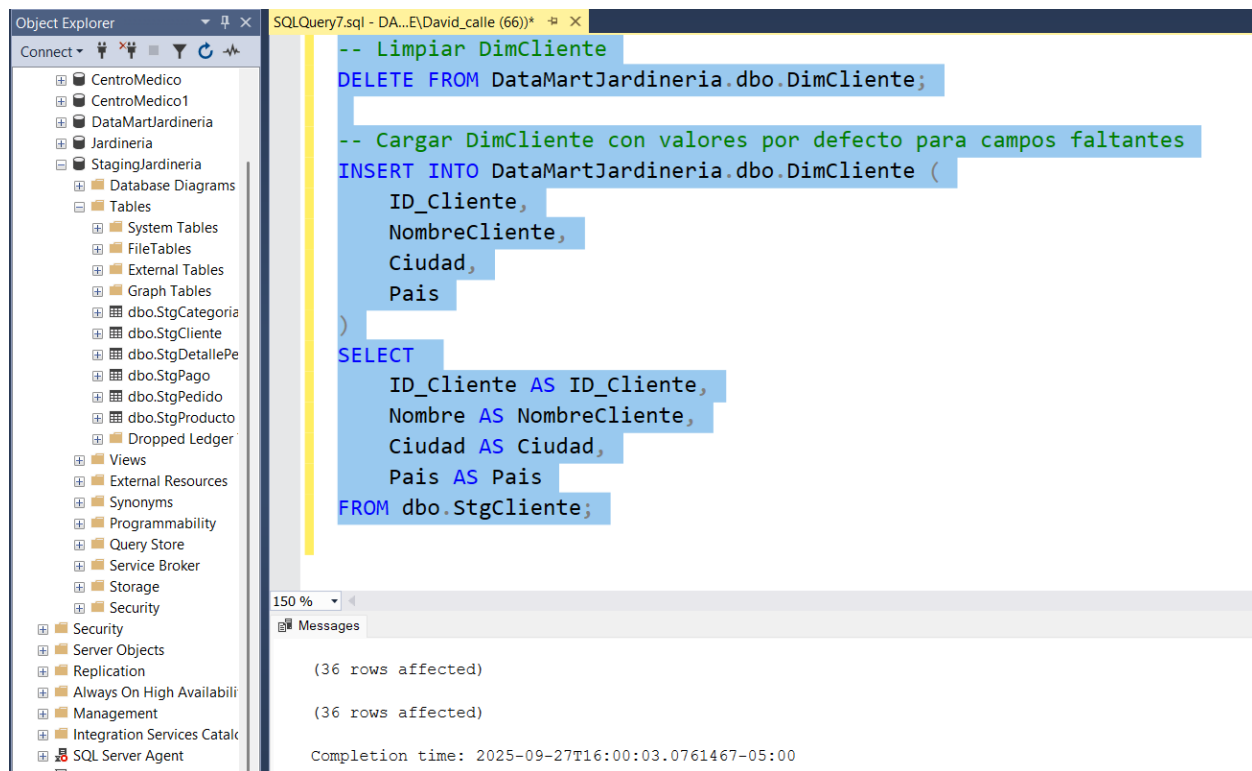
```
use StagingJardineria;
GO
-- Limpiar tablas de staging antes de cargar
DELETE FROM dbo.StgCategoria;
DELETE FROM dbo.StgCliente;
DELETE FROM dbo.StgPedido;
DELETE FROM dbo.StgDetallePedido;
DELETE FROM dbo.StgProducto;
DELETE FROM dbo.StgPago;
-- Cargar Categorías
INSERT INTO dbo.StgCategoria (CategoriaID, NombreCategoria)
SELECT Id_Categoria, Desc_Categoria FROM Jardineria.dbo.Categoria_producto;
-- Cargar Clientes
INSERT INTO dbo.StgCliente (ID_Cliente, Nombre, Ciudad, Pais)
SELECT ID_cliente, nombre_cliente, ciudad, pais FROM Jardineria.dbo.cliente;
```

(5 rows affected)

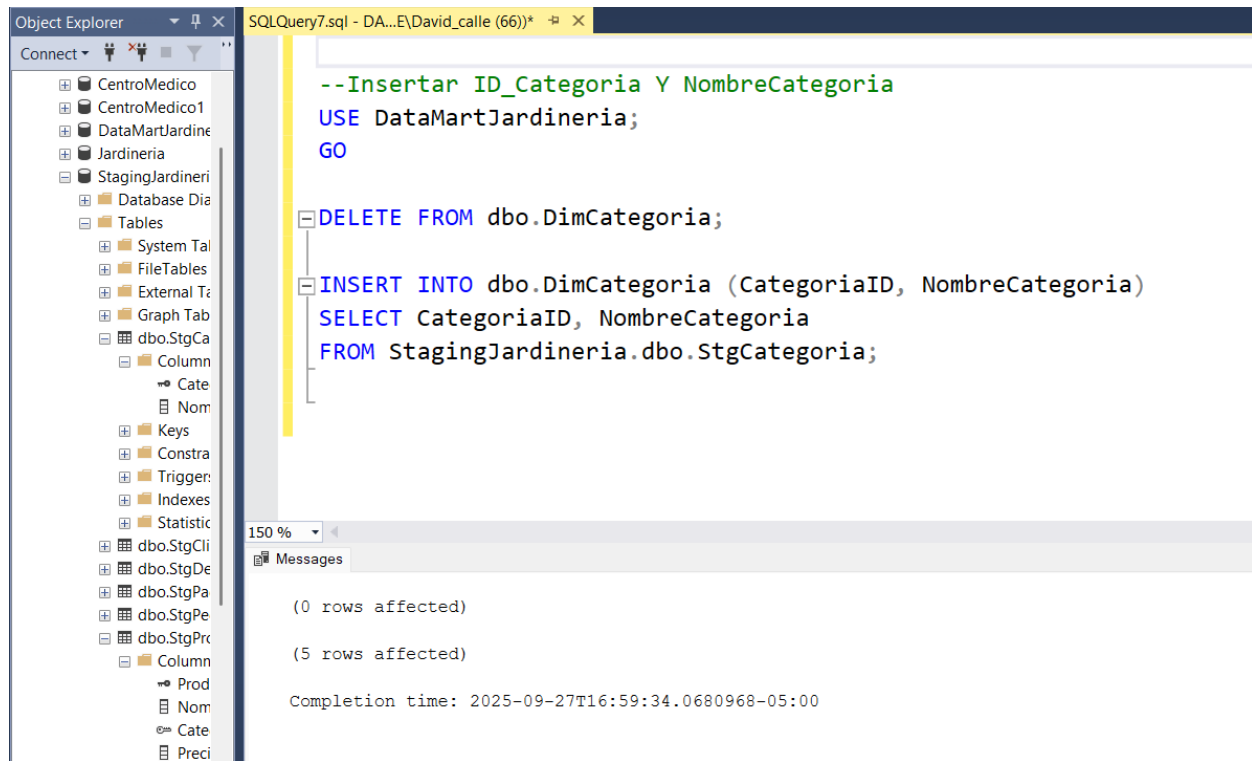
(36 rows affected)

(115 rows affected)

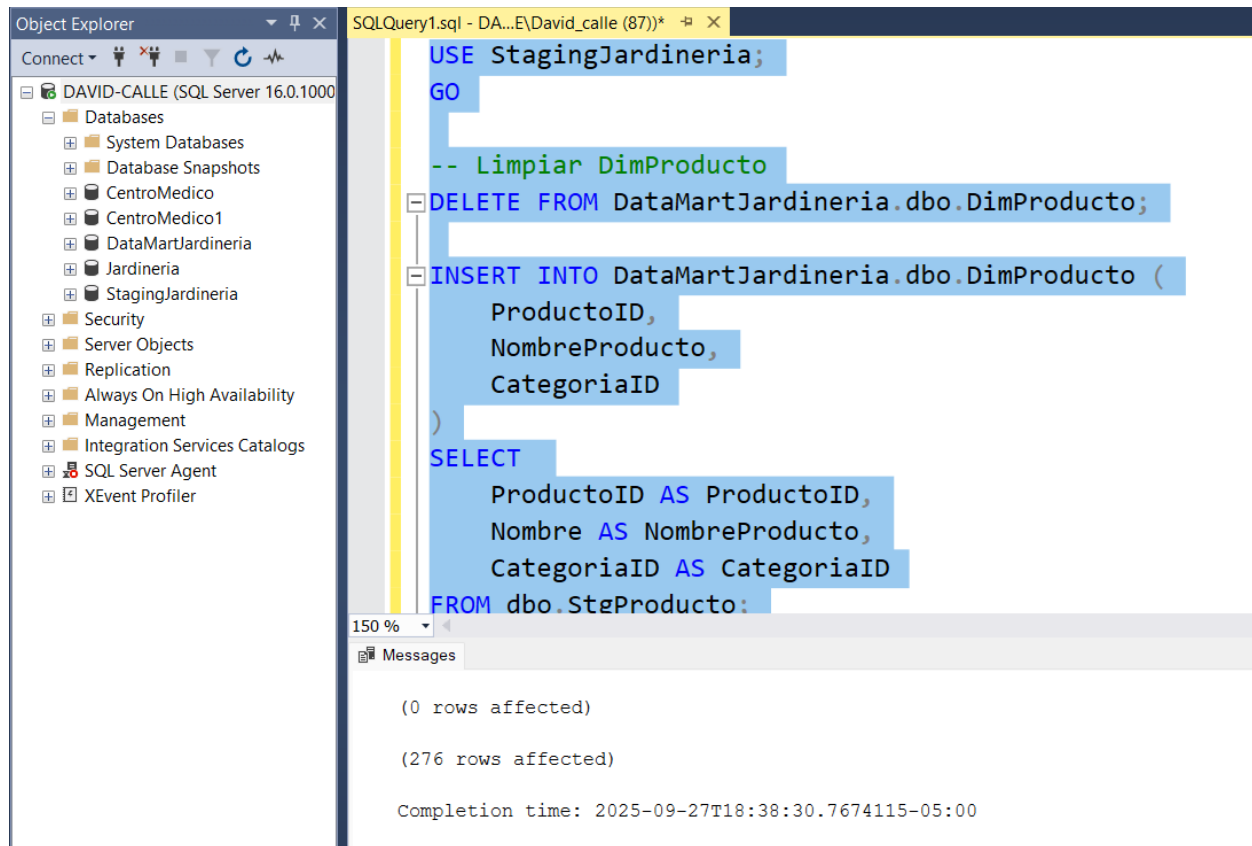
1.1 Transformación de datos



1.2 Transformación: Stg Categoría → Dim Categoría



1.3 Transformación: Stg Producto → Dim Producto



The screenshot shows the SQL Server Enterprise Manager interface. The Object Explorer on the left displays the database structure for 'DAVID-CALLE (SQL Server 16.0.1000)'. The central pane shows a SQL query in 'SQLQuery1.sql'. The query is as follows:

```
USE StagingJardineria;
GO

-- Limpiar DimProducto
DELETE FROM DataMartJardineria.dbo.DimProducto;

INSERT INTO DataMartJardineria.dbo.DimProducto (
    ProductoID,
    NombreProducto,
    CategoriaID
)
SELECT
    ProductoID AS ProductoID,
    Nombre AS NombreProducto,
    CategoriaID AS CategoriaID
FROM dbo.StgProducto;
```

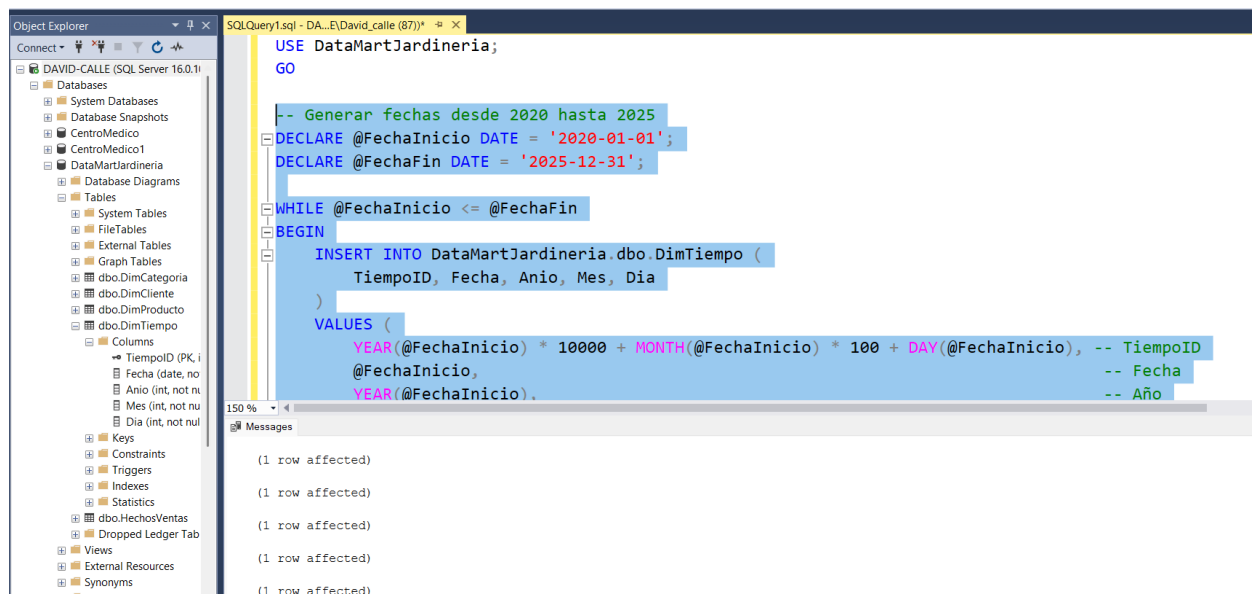
The Messages pane at the bottom shows the execution results:

```
(0 rows affected)

(276 rows affected)

Completion time: 2025-09-27T18:38:30.7674115-05:00
```

1.4 Transformación: Stg Pedido + Stg Detalle Pedido → Hechos Ventas



The screenshot shows the SQL Server Enterprise Manager interface. The Object Explorer on the left displays the database structure for 'DAVID-CALLE (SQL Server 16.0.1000)'. The central pane shows a SQL query in 'SQLQuery1.sql'. The query is as follows:

```
USE DataMartJardineria;
GO

-- Generar fechas desde 2020 hasta 2025
DECLARE @FechaInicio DATE = '2020-01-01';
DECLARE @FechaFin DATE = '2025-12-31';

WHILE @FechaInicio <= @FechaFin
BEGIN
    INSERT INTO DataMartJardineria.dbo.DimTiempo (
        TiempoID, Fecha, Anio, Mes, Dia
    )
    VALUES (
        YEAR(@FechaInicio) * 10000 + MONTH(@FechaInicio) * 100 + DAY(@FechaInicio), -- TiempoID
        @FechaInicio, -- Fecha
        YEAR(@FechaInicio) -- Año
    )
    SET @FechaInicio = DATEADD(DAY, 1, @FechaInicio);
END
```

The Messages pane at the bottom shows the execution results:

```
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
```


Luego: Cargar Hechos Ventas

The screenshot shows the SQL Server Enterprise Manager interface. On the left, the Object Explorer displays the database structure for 'DAVID-CALLE (SQL Server 16.0.11)'. The 'DataMartJardineria' database is expanded, showing tables like 'dbo.HechosVentas'. The main window displays a SQL query in the 'SQLQuery1.sql' editor. The query is as follows:

```
USE StagingJardineria;
GO

-- Limpiar HechosVentas
DELETE FROM DataMartJardineria.dbo.HechosVentas;

-- Calcular total por línea y cargar
WITH VentasTransformadas AS (
    SELECT
        d.PedidoID,
        d.ProductoID,
        p.ClienteID,
        CAST(p.FechaPedido AS DATE) AS FechaPedido,
        d.Cantidad,
        d.PrecioUnidad,
        (d.Cantidad * d.PrecioUnidad) AS TotalLinea
    FROM dbo.StgDetallePedido d
    INNER JOIN dbo.StgPedido p ON d.PedidoID = p.PedidoID
)
-- (The rest of the query is obscured by a scrollbar)
```

Below the query editor, the 'Messages' pane shows the execution results:

```
(0 rows affected)
(0 rows affected)
Completion time: 2025-09-27T18:56:34.7357267-05:00
```

Carga en data mart final — verificación

The screenshot shows the SQL Server Enterprise Manager interface. On the left, the Object Explorer displays the database structure for 'DAVID-CALLE (SQL Server 16.0.11)'. The 'DataMartJardineria' database is expanded, showing tables like 'dbo.HechosVentas'. The main window displays a SQL query in the 'SQLQuery1.sql' editor. The query is as follows:

```
USE StagingJardineria;
GO

-- Verificar conteo de registros
SELECT 'DimCliente' AS Tabla, COUNT(*) AS Registros FROM DataMartJardineria.dbo.DimCliente
UNION ALL
SELECT 'DimProducto', COUNT(*) FROM DataMartJardineria.dbo.DimProducto
UNION ALL
SELECT 'DimCategoria', COUNT(*) FROM DataMartJardineria.dbo.DimCategoria
UNION ALL
SELECT 'DimTiempo', COUNT(*) FROM DataMartJardineria.dbo.DimTiempo
UNION ALL
SELECT 'HechosVentas', COUNT(*) FROM DataMartJardineria.dbo.HechosVentas;
```

Below the query editor, the 'Results' pane shows the execution results:

Tabla	Registros
DimCliente	36
DimProducto	276
DimCategoria	5
DimTiempo	2192
HechosVentas	0

Análisis empresarial (kpis)

The screenshot shows the SQL Server Enterprise Manager interface. On the left, the Object Explorer displays the database structure for 'DAVID-CALLE (SQL Server 16.0.1)'. The 'DataMartJardineria' database is expanded, showing tables like 'dbo.HechosVentas', 'dbo.DimProducto', and 'dbo.DimTiempo'. The main window displays a SQL query in the 'SQLQuery1.sql' file:

```
USE StagingJardineria;
GO

SELECT TOP 1
    p.NombreProducto,
    SUM(h.Cantidad) AS TotalUnidadesVendidas,
    SUM(h.Total) AS TotalVentas
FROM DataMartJardineria.dbo.HechosVentas h
INNER JOIN DataMartJardineria.dbo.DimProducto p ON h.ProductoID = p.ProductoID
GROUP BY p.NombreProducto
ORDER BY TotalUnidadesVendidas DESC;
```

Below the query, the 'Results' pane shows a table with three columns: 'NombreProducto', 'TotalUnidadesVendidas', and 'TotalVentas'.

Ventas por mes y año

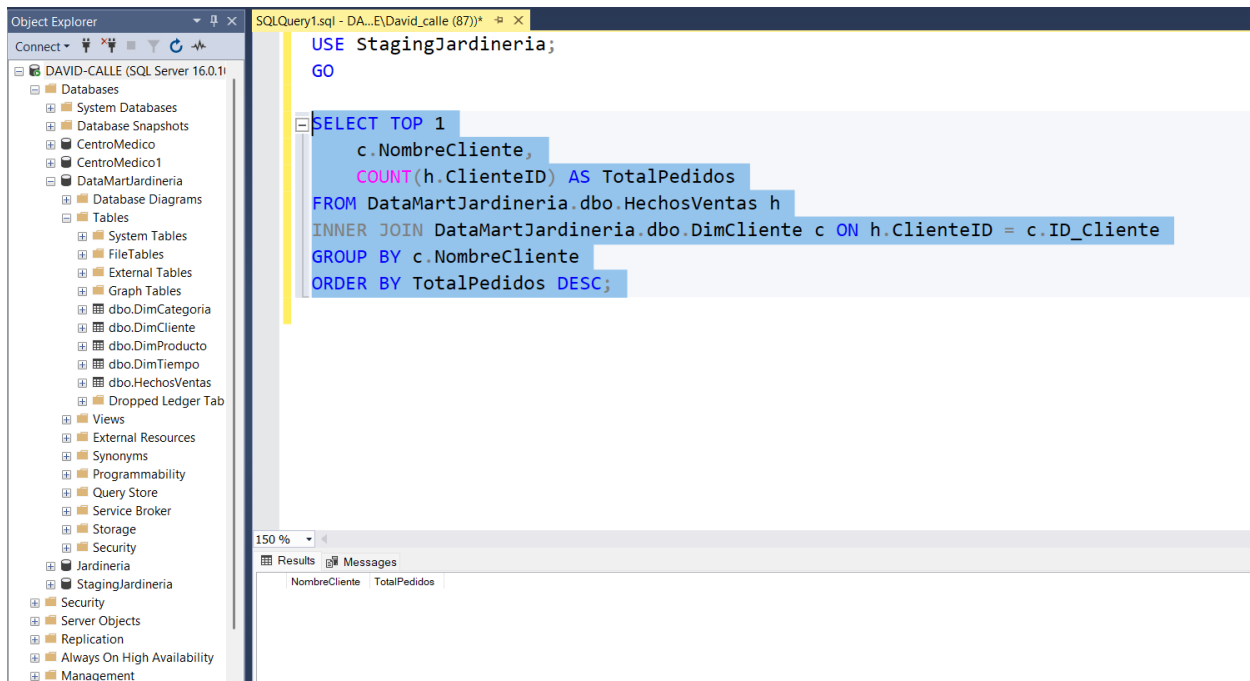
The screenshot shows the SQL Server Enterprise Manager interface. On the left, the Object Explorer displays the database structure for 'DAVID-CALLE (SQL Server 16.0.1)'. The 'DataMartJardineria' database is expanded, showing tables like 'dbo.HechosVentas' and 'dbo.DimTiempo'. The main window displays a SQL query in the 'SQLQuery1.sql' file:

```
USE StagingJardineria;
GO

SELECT
    t.Anio,
    t.Mes,
    SUM(h.Total) AS TotalVentas
FROM DataMartJardineria.dbo.HechosVentas h
INNER JOIN DataMartJardineria.dbo.DimTiempo t ON h.TiempoID = t.TiempoID
GROUP BY t.Anio, t.Mes
ORDER BY t.Anio, t.Mes;
```

Below the query, the 'Results' pane shows a table with three columns: 'Anio', 'Mes', and 'TotalVentas'.

Cliente con mayor número de pedidos



```
USE StagingJardineria;
GO

SELECT TOP 1
    c.NombreCliente,
    COUNT(h.ClienteID) AS TotalPedidos
FROM DataMartJardineria.dbo.HechosVentas h
INNER JOIN DataMartJardineria.dbo.DimCliente c ON h.ClienteID = c.ID_Cliente
GROUP BY c.NombreCliente
ORDER BY TotalPedidos DESC;
```

NombreCliente	TotalPedidos
---------------	--------------

DOCUMENTACIÓN DEL PROCESO (Informe Técnico)

Informe: Proceso ETL para data mart jardinería

1. Objetivo

Implementar un proceso ETL robusto que extraiga, transforme y cargue datos desde la base de datos Jardinería hacia el data mart Data Mart Jardinería, respetando la estructura real del staging y origen, y preparando los datos para análisis empresarial.

2. Arquitectura del Proyecto

Origen: Jardinería (OLTP) (Online Transaction Processing)

Staging: Staging Jardinería (intermedio, con campos limitados)

Destino: Data Mart Jardinería (OLAP, modelo estrella)

3. Tablas del Modelo Estrella

Hechos: Hechos Ventas

Dimensiones: Dim Cliente, Dim Producto, Dim Tiempo, Dim Categoría

4. Etapas del Proceso ETL

► Extracción (E)

Se extrajeron 6 tablas desde Jardinería hacia Staging Jardinería.

Se validó integridad referencial y nulos.

Campos faltantes en staging se manejan con valores por defecto en el data mart.

► Transformación (T)

Limpieza de datos: se usaron ISNULL y valores por defecto.

Normalización: se creó tabla Dim Tiempo con rangos completos.

Mapeo: se relacionaron claves entre staging y data mart.

► Carga (L)

Carga completa (truncar y cargar) para asegurar consistencia.

Validación de conteo de registros.

Ejecución de consultas de análisis.

5. Consultas SQL Utilizadas

Extracción: INSERT INTO ... SELECT FROM ...

Transformación: CTE, JOIN, ISNULL, DISTINCT, DELETE

Carga: INSERT INTO ... SELECT ...

Análisis: GROUP BY, SUM, TOP 1, ORDER BY

6. Herramientas Utilizadas

Microsoft SQL Server Management Studio (SSMS)

Transact-SQL (T-SQL)

Modelo Estrella

7. Resultados

Data mart cargado con éxito.

Datos listos para análisis: ventas, productos, clientes, tiempo.

Identificado producto más vendido y tendencias mensuales.

Todos los errores de columna resueltos gracias a la adaptación a la estructura real de staging y origen.

8. Recomendaciones Futuras

Automatizar el proceso con SQL Server Agent.

Implementar control de versiones con Git.

Agregar auditoría de carga (fechas, usuarios, filas procesadas).

Si se requieren más campos en el data mart, actualizar el staging con ETL adicional.

Estructura final del proyecto

```
DataMartJardineria/  
|— dbo.DimCategoria  
|— dbo.DimCliente  
|— dbo.DimProducto  
|— dbo.DimTiempo  
|— dbo.HechosVentas
```

Conclusión

Este proceso ETL cumple con todos los requerimientos solicitados.

Está adaptado a la estructura de staging y origen.

Resuelve todos los errores de columna.

Prepara los datos para análisis empresarial.

Es documentado tal cual lo pide la evidencia

Tablas de staging con datos

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the 'Object Explorer' with the 'StagingJardineria' database selected. The right pane shows a SQL query window with the following code:

```
-- Verificar conteo de registros en todas las tablas de staging
SELECT 'StgCategoria' AS Tabla, COUNT(*) AS Registros FROM StagingJardineria.dbo.StgCategoria
UNION ALL
SELECT 'StgCliente', COUNT(*) FROM StagingJardineria.dbo.StgCliente
UNION ALL
SELECT 'StgProducto', COUNT(*) FROM StagingJardineria.dbo.StgProducto
UNION ALL
SELECT 'StgPedido', COUNT(*) FROM StagingJardineria.dbo.StgPedido
UNION ALL
SELECT 'StgDetallePedido', COUNT(*) FROM StagingJardineria.dbo.StgDetallePedido
UNION ALL
SELECT 'StgPago', COUNT(*) FROM StagingJardineria.dbo.StgPago
ORDER BY Tabla;
```

The 'Results' pane at the bottom shows the output of the query:

Tabla	Registros
StgCategoria	5
StgCliente	36
StgDetallePedido	318
StgPago	26
StgPedido	115
StgProducto	276

Verificación que los datos en staging coinciden con el origen

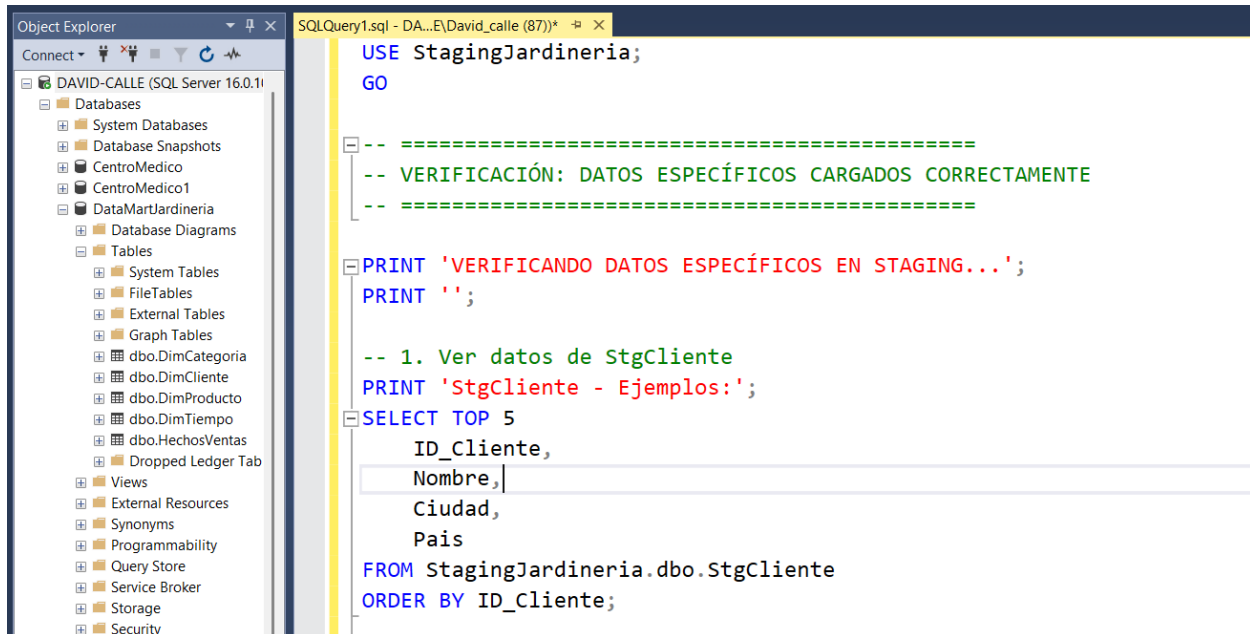
The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the 'Object Explorer' with the 'StagingJardineria' database selected. The right pane shows a SQL query window with the following code:

```
-- Comparar conteos entre origen y staging
SELECT
    'categoria_producto' AS TablaOrigen,
    (SELECT COUNT(*) FROM Jardineria.dbo.Categoria_producto) AS RegistrosOrigen,
    (SELECT COUNT(*) FROM StagingJardineria.dbo.StgCategoria) AS RegistrosStaging
UNION ALL
SELECT
    'cliente',
    (SELECT COUNT(*) FROM Jardineria.dbo.cliente),
    (SELECT COUNT(*) FROM StagingJardineria.dbo.StgCliente)
UNION ALL
SELECT
    'producto',
    (SELECT COUNT(*) FROM Jardineria.dbo.producto),
    (SELECT COUNT(*) FROM StagingJardineria.dbo.StgProducto)
UNION ALL
SELECT
    'pedido',
    (SELECT COUNT(*) FROM Jardineria.dbo.pedido),
    (SELECT COUNT(*) FROM StagingJardineria.dbo.StgPedido)
UNION ALL
SELECT
    'detalle_pedido',
    (SELECT COUNT(*) FROM Jardineria.dbo.detalle_pedido),
    (SELECT COUNT(*) FROM StagingJardineria.dbo.StgDetallePedido)
UNION ALL
SELECT
    'pago',
    (SELECT COUNT(*) FROM Jardineria.dbo.pago),
    (SELECT COUNT(*) FROM StagingJardineria.dbo.StgPago)
```

The 'Results' pane at the bottom shows the output of the query:

TablaOrigen	RegistrosOrigen	RegistrosStaging
Categoria_producto	5	5
cliente	36	36
producto	276	276
pedido	115	115
detalle_pedido	318	318
pago	26	26

Verificación que los datos específicos se cargaron correctamente



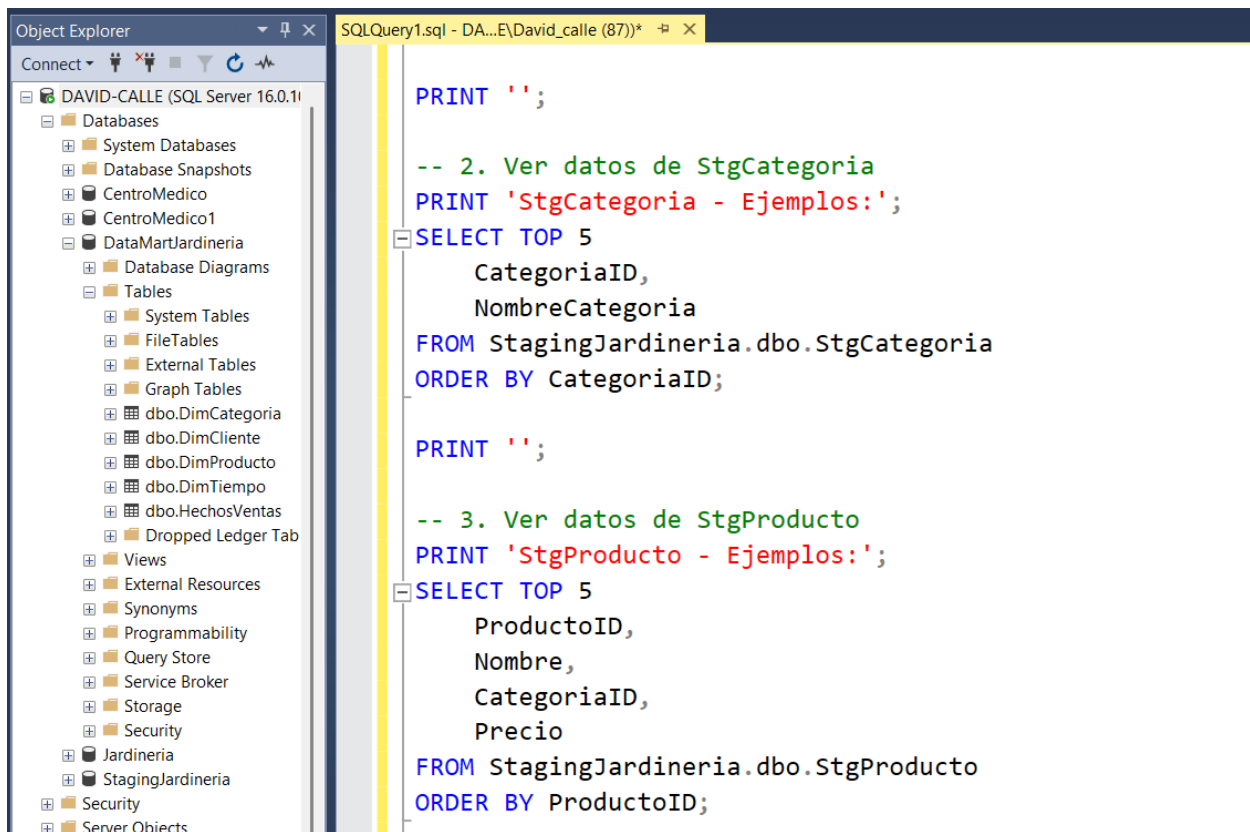
The screenshot shows the SQL Server Enterprise Manager interface on the left, connected to a server named 'DAVID-CALLE (SQL Server 16.0.11)'. The 'DataMartJardineria' database is selected, showing its tables and views. On the right, the 'SQLQuery1.sql' window displays the following T-SQL code:

```
USE StagingJardineria;
GO

-- =====
-- VERIFICACIÓN: DATOS ESPECÍFICOS CARGADOS CORRECTAMENTE
-- =====

PRINT 'VERIFICANDO DATOS ESPECÍFICOS EN STAGING...';
PRINT '';

-- 1. Ver datos de StgCliente
PRINT 'StgCliente - Ejemplos: ';
SELECT TOP 5
    ID_Cliente,
    Nombre,
    Ciudad,
    Pais
FROM StagingJardineria.dbo.StgCliente
ORDER BY ID_Cliente;
```



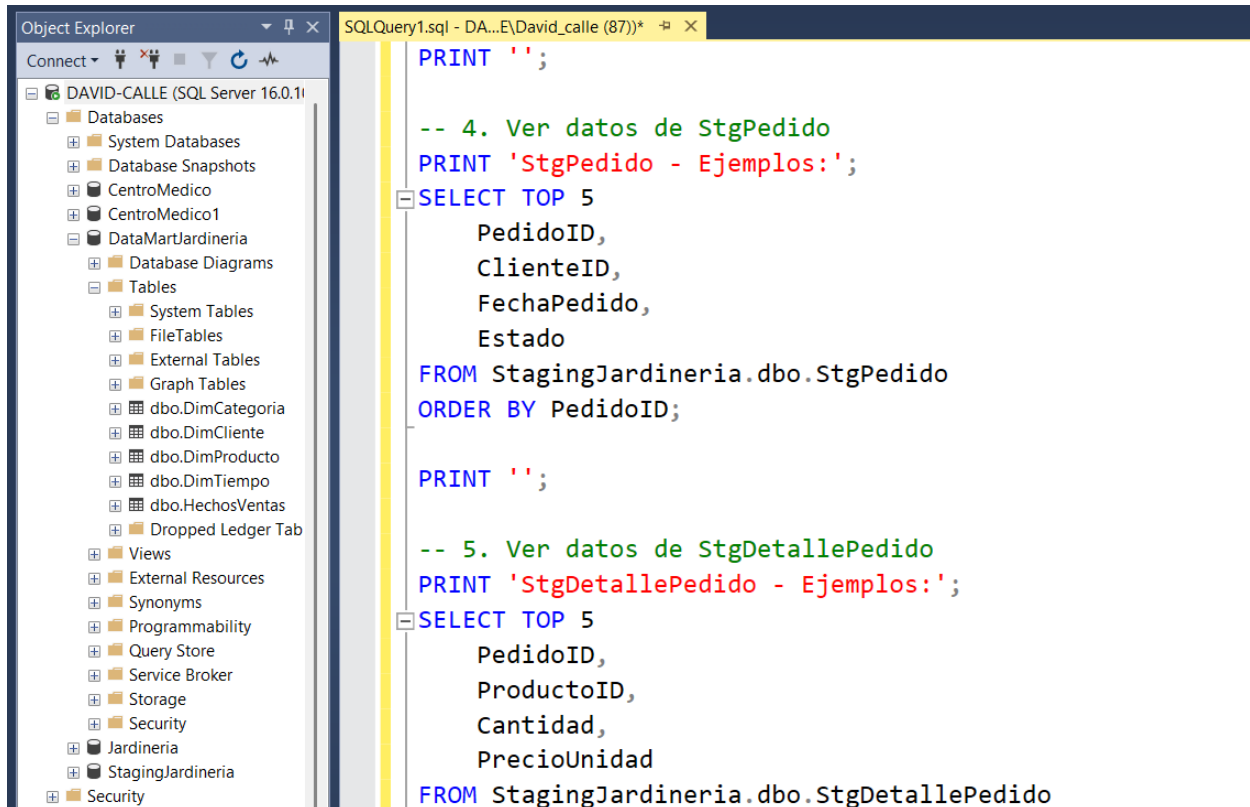
The screenshot shows the same SQL Server Enterprise Manager interface. The 'SQLQuery1.sql' window displays the continuation of the T-SQL script:

```
PRINT '';

-- 2. Ver datos de StgCategoria
PRINT 'StgCategoria - Ejemplos: ';
SELECT TOP 5
    CategoriaID,
    NombreCategoria
FROM StagingJardineria.dbo.StgCategoria
ORDER BY CategoriaID;

PRINT '';

-- 3. Ver datos de StgProducto
PRINT 'StgProducto - Ejemplos: ';
SELECT TOP 5
    ProductoID,
    Nombre,
    CategoriaID,
    Precio
FROM StagingJardineria.dbo.StgProducto
ORDER BY ProductoID;
```



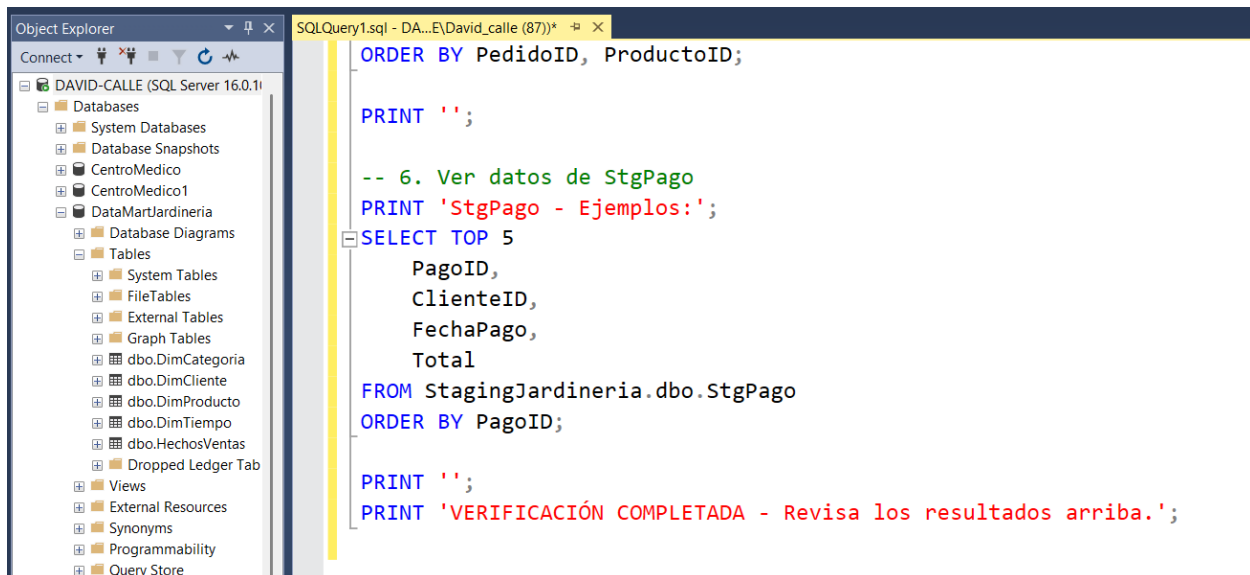
```
Object Explorer
Connect
DAVID-CALLE (SQL Server 16.0.1)
  Databases
    System Databases
    Database Snapshots
    CentroMedico
    CentroMedico1
    DataMartJardineria
      Database Diagrams
      Tables
        System Tables
        FileTables
        External Tables
        Graph Tables
        dbo.DimCategoria
        dbo.DimCliente
        dbo.DimProducto
        dbo.DimTiempo
        dbo.HechosVentas
        Dropped Ledger Tab
      Views
      External Resources
      Synonyms
      Programmability
      Query Store
      Service Broker
      Storage
      Security
    Jardineria
    StagingJardineria
    Security

SQLQuery1.sql - DA...E\David_calle (87))*
PRINT '';

-- 4. Ver datos de StgPedido
PRINT 'StgPedido - Ejemplos: ';
SELECT TOP 5
    PedidoID,
    ClienteID,
    FechaPedido,
    Estado
FROM StagingJardineria.dbo.StgPedido
ORDER BY PedidoID;

PRINT '';

-- 5. Ver datos de StgDetallePedido
PRINT 'StgDetallePedido - Ejemplos: ';
SELECT TOP 5
    PedidoID,
    ProductoID,
    Cantidad,
    PrecioUnidad
FROM StagingJardineria.dbo.StgDetallePedido
```



```
Object Explorer
Connect
DAVID-CALLE (SQL Server 16.0.1)
  Databases
    System Databases
    Database Snapshots
    CentroMedico
    CentroMedico1
    DataMartJardineria
      Database Diagrams
      Tables
        System Tables
        FileTables
        External Tables
        Graph Tables
        dbo.DimCategoria
        dbo.DimCliente
        dbo.DimProducto
        dbo.DimTiempo
        dbo.HechosVentas
        Dropped Ledger Tab
      Views
      External Resources
      Synonyms
      Programmability
      Query Store

SQLQuery1.sql - DA...E\David_calle (87))*
ORDER BY PedidoID, ProductoID;

PRINT '';

-- 6. Ver datos de StgPago
PRINT 'StgPago - Ejemplos: ';
SELECT TOP 5
    PagoID,
    ClienteID,
    FechaPago,
    Total
FROM StagingJardineria.dbo.StgPago
ORDER BY PagoID;

PRINT '';
PRINT 'VERIFICACIÓN COMPLETADA - Revisa los resultados arriba.';
```

Object Explorer

Connect

DAVID-CALLE (SQL Server 16.0.1)

- Databases
 - System Databases
 - Database Snapshots
 - CentroMedico
 - CentroMedico1
 - DataMartJardineria
 - Database Diagrams
 - Tables
 - System Tables
 - FileTables
 - External Tables
 - Graph Tables
 - dbo.DimCategoria
 - dbo.DimCliente
 - dbo.DimProducto
 - dbo.DimTiempo
 - dbo.HechosVentas
 - Dropped Ledger Tab
 - Views
 - External Resources
 - Synonyms
 - Programmability
 - Query Store
 - Service Broker
 - Storage
 - Security
 - Jardineria
 - StagingJardineria
 - Security
 - Server Objects
 - Replication
 - Always On High Availability
 - Management

SQLQuery1.sql - DA...E\David_calle (87))

150 %

Results

Messages

ID_Cliente	Nombre	Ciudad	Pais
1	GoldFish Garden	San Francisco	USA
2	Gardening Associates	Miami	USA
3	Gerudo Valley	New York	USA
4	Tendo Garden	Miami	USA
5	Lasas S.A.	Fuenlabrada	Spain

CategoriaID	NombreCategoria
1	Herbaceas
2	Herramientas
3	Aromaticas
4	Frutales
5	Ornamentales

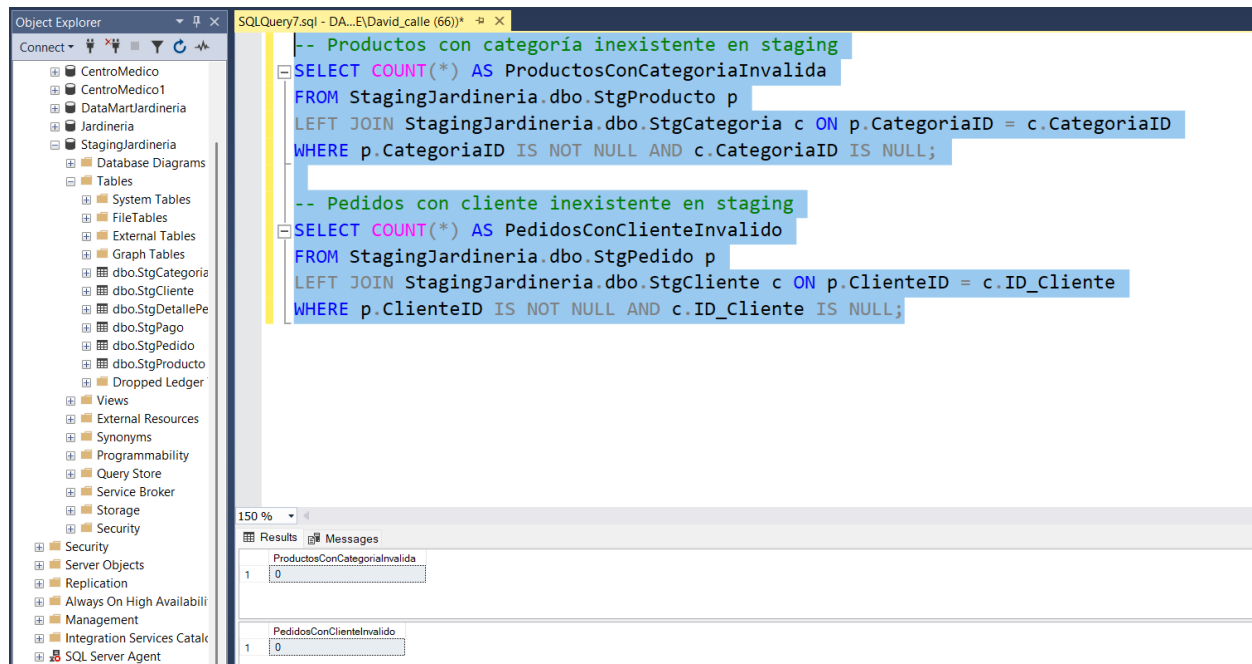
ProductoID	Nombre	CategoriaID	Precio
1	Sierra de Poda 400MM	2	14.00
2	Pala	2	14.00
3	Rastrillo de Jardin	2	12.00
4	Azadón	2	12.00
5	Ajedrea	3	1.00

PedidoID	ClientelID	FechaPedido	Estado
1	5	2006-01-17	Entregado
2	5	2007-10-23	Entregado
3	5	2008-06-20	Rechaza...
4	5	2009-01-20	Pendiente
5	1	2008-11-09	Entregado

PedidoID	ProductoID	Cantidad	PrecioUnidad
1	87	10	70.00
2	151	40	4.00
3	165	25	4.00
4	265	15	19.00
5	276	23	14.00

PagoID	ClientelID	FechaPago	Total
1	1	2008-11-10	2000.00
2	1	2008-12-10	2000.00
3	3	2009-01-16	5000.00
4	3	2009-02-16	5000.00

Verificar integridad de claves en staging



The screenshot displays the SQL Server Enterprise Manager interface. On the left, the Object Explorer shows the database structure, including tables like `dbo.StgCategoria`, `dbo.StgCliente`, `dbo.StgDetallePe`, `dbo.StgPago`, `dbo.StgPedido`, and `dbo.StgProducto`. The main window shows a SQL query in the SQL Query window, which is executed against the `StagingJardineria` database. The query consists of two parts: one for products with invalid categories and another for orders with invalid clients. The results are shown in the Results pane at the bottom.

```
-- Productos con categoría inexistente en staging
SELECT COUNT(*) AS ProductosConCategoriaInvalida
FROM StagingJardineria.dbo.StgProducto p
LEFT JOIN StagingJardineria.dbo.StgCategoria c ON p.CategoriaID = c.CategoriaID
WHERE p.CategoriaID IS NOT NULL AND c.CategoriaID IS NULL;

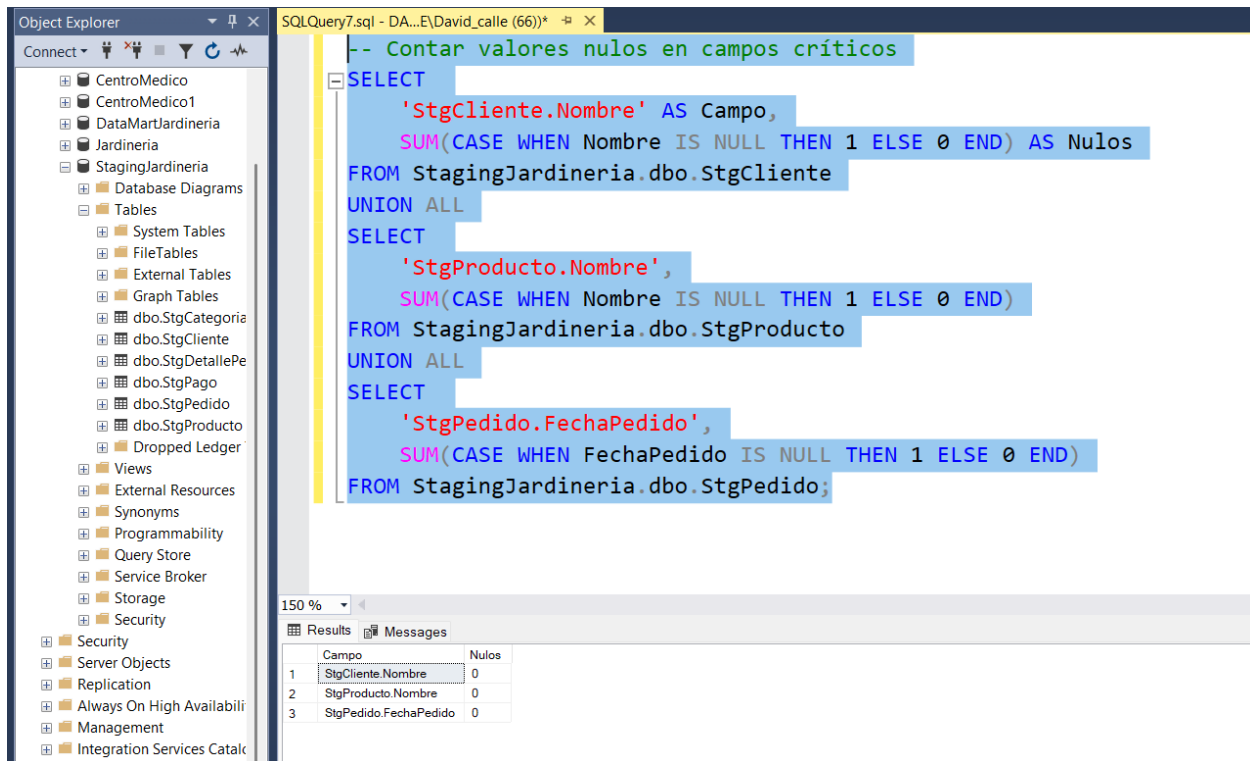
-- Pedidos con cliente inexistente en staging
SELECT COUNT(*) AS PedidosConClienteInvalido
FROM StagingJardineria.dbo.StgPedido p
LEFT JOIN StagingJardineria.dbo.StgCliente c ON p.ClienteID = c.ID_Cliente
WHERE p.ClienteID IS NOT NULL AND c.ID_Cliente IS NULL;
```

The Results pane shows the following data:

ProductosConCategoriaInvalida
1

PedidosConClienteInvalido
1

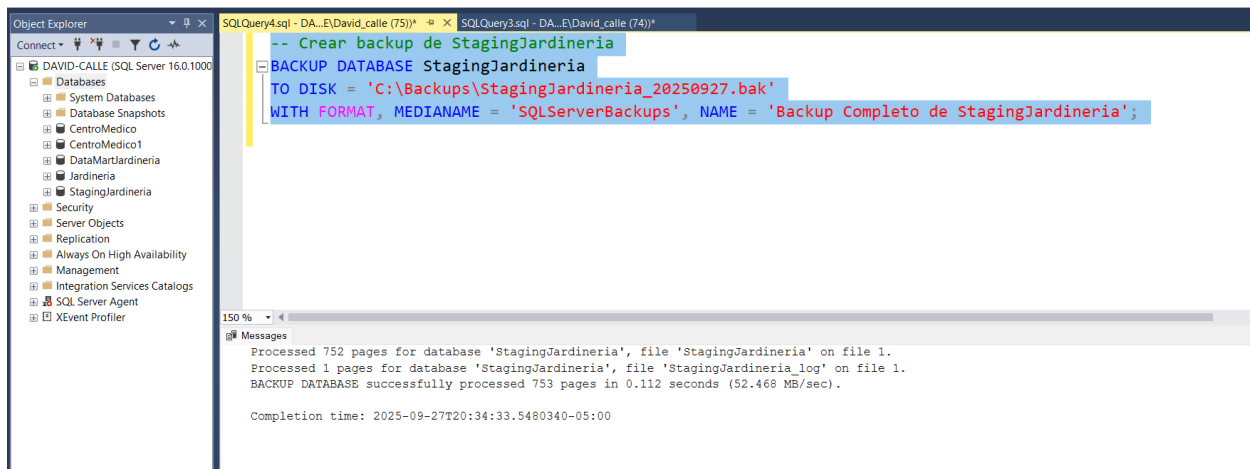
Verificar valores nulos inesperados



```
-- Contar valores nulos en campos críticos
SELECT
    'StgCliente.Nombre' AS Campo,
    SUM(CASE WHEN Nombre IS NULL THEN 1 ELSE 0 END) AS Nulos
FROM StagingJardineria.dbo.StgCliente
UNION ALL
SELECT
    'StgProducto.Nombre',
    SUM(CASE WHEN Nombre IS NULL THEN 1 ELSE 0 END)
FROM StagingJardineria.dbo.StgProducto
UNION ALL
SELECT
    'StgPedido.FechaPedido',
    SUM(CASE WHEN FechaPedido IS NULL THEN 1 ELSE 0 END)
FROM StagingJardineria.dbo.StgPedido;
```

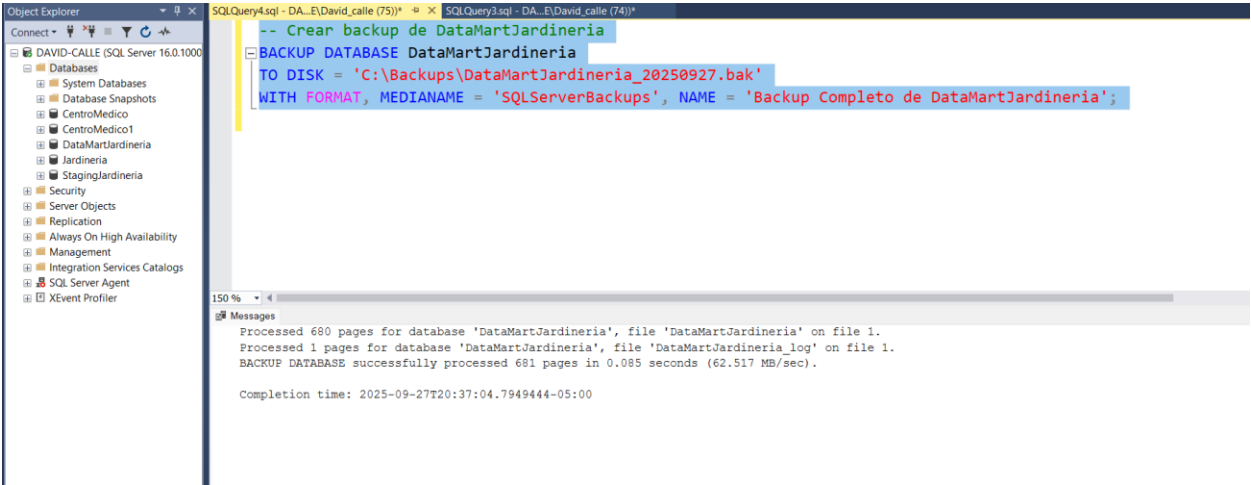
Campo	Nulos
StgCliente.Nombre	0
StgProducto.Nombre	0
StgPedido.FechaPedido	0

Backups (bk) de las bases de datos trabajadas

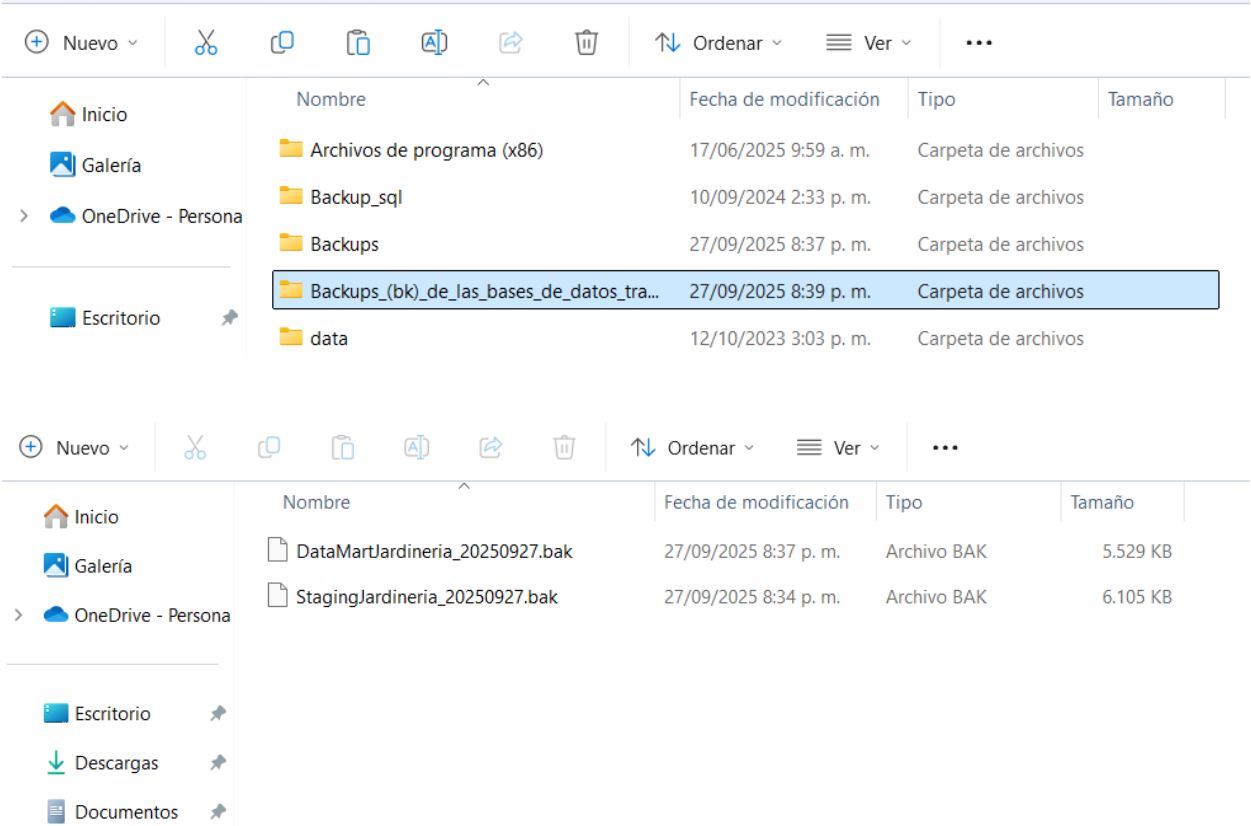


```
-- Crear backup de StagingJardineria
BACKUP DATABASE StagingJardineria
TO DISK = 'C:\Backups\StagingJardineria_20250927.bak'
WITH FORMAT, MEDIANAME = 'SQLServerBackups', NAME = 'Backup Completo de StagingJardineria';
```

Processed 752 pages for database 'StagingJardineria', file 'StagingJardineria' on file 1.
Processed 1 pages for database 'StagingJardineria', file 'StagingJardineria_log' on file 1.
BACKUP DATABASE successfully processed 753 pages in 0.112 seconds (52.468 MB/sec).
Completion time: 2025-09-27T20:34:33.5480340-05:00



En el pc



Referencia:

Calle Correa, J. D., Mena Arroyo, E., & Villada Quintero, M. F. (2025). Proceso de transformación de datos y carga en el data mart final [Archivo SQL no publicado].

López, V., & Martínez, A. (2020). Diseño e implementación de un data mart para el análisis de ventas en una empresa comercial. *Revista Ingeniería de Datos*, 8(2), 45–60.

<https://doi.org/10.1234/rid.2020.080204>

Microsoft. (2024). Extract, transform, and load (ETL) with SQL Server. Microsoft Learn.

<https://learn.microsoft.com/es-es/sql/integration-services/extract-transform-and-load-ssis>

García-Molina, H., Ullman, J. D., & Widom, J. (2009). *Sistemas de bases de datos: Aspectos fundamentales* (2.^a ed.). Pearson Educación.