Área de TI y Ciberseguridad

**Carrera: Ingeniería en Informática**

**Asignatura: Arquitectura y Almacenamiento de Datos**

**Sección: D-IEI-N5-P1-C3**

**Nombre del docente: Oscar Alfredo Rodríguez**

**Nombre de los integrantes del grupo: Jorge González, Nicolas Espinoza y Alessandro torriani**

**Fecha de entrega**

**Jueves 23/07/2025**

Informe de Diseño de Datawarehouse

Northwind

**Contenido**

[I. Análisis de la problemática y revisión de software necesarios 3](#_Toc200319654)

[II. Descarga de base de datos elegida y restauración con MS SQL Server Management Studio 3](#_Toc200319655)

[III. Creación de directorios y subdirectorios para prototipo de DW 5](#_Toc200319656)

[IV. Diseño de prototipo de DW 6](#_Toc200319657)

[V. Creación del prototipo de DW 8](#_Toc200319658)

[VI. Realización de pequeño ejemplo de ETL 14](#_Toc200319659)

Análisis de la problemática y revisión de software necesarios

1. Descripción del caso.

La empresa Talca-OLAP requiere mejorar su toma de decisiones mediante la construcción de un prototipo de datawarehouse (DW), consolidando datos provenientes de una base de datos OLTP que registra operaciones diarias. Se busca generar reportes estratégicos y análisis de tendencias utilizando un modelo dimensional.  
  
para esto usaremos la base de datos de Northwind y realiza operaciones como ver el total de ventas, ver los productos, los dientes etc

1. Justificación de base de datos elegida.

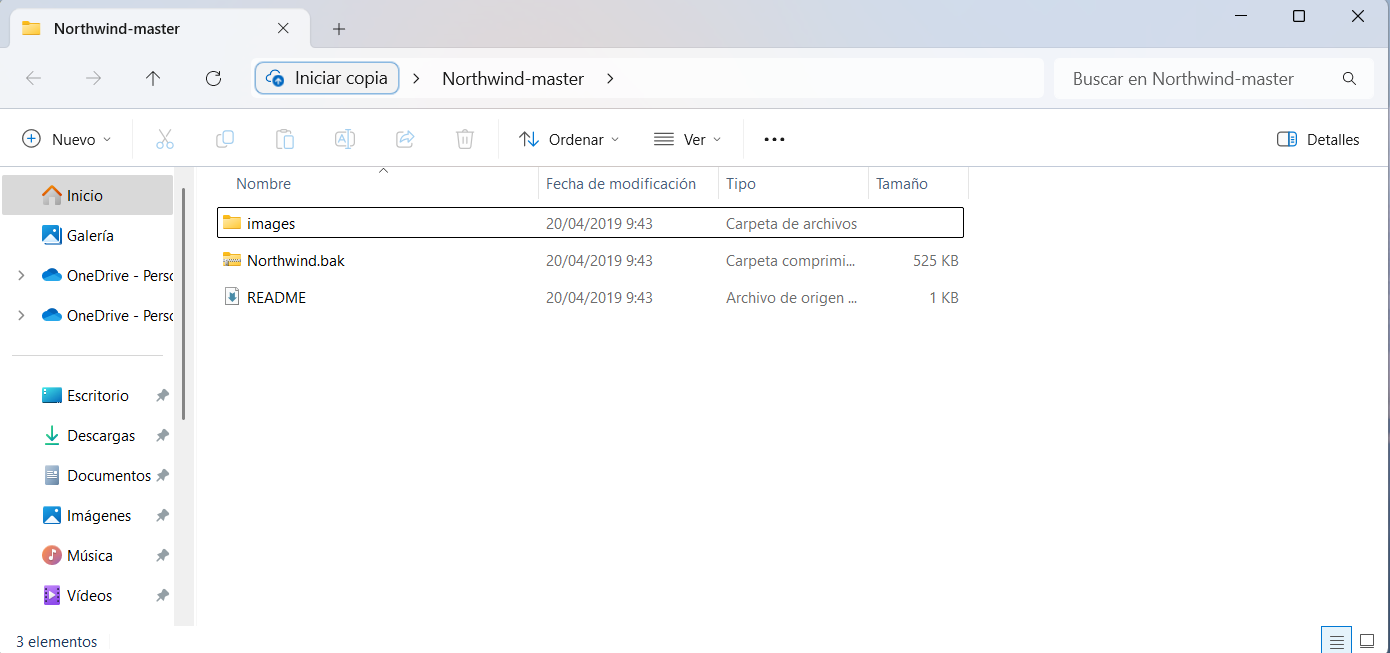
Se eligió la base de datos Northwind, ya que representa de forma completa una empresa de importación y exportación de alimentos. Contiene información esencial sobre productos, clientes, empleados, proveedores y pedidos, lo cual permite construir un esquema dimensional significativo.  
  
Esto permitiría un Análisis de negocio estratégico:

Analizar tendencias de ventas por períodos (mensual, anual)

Identificar productos más rentables y de mayor rotación

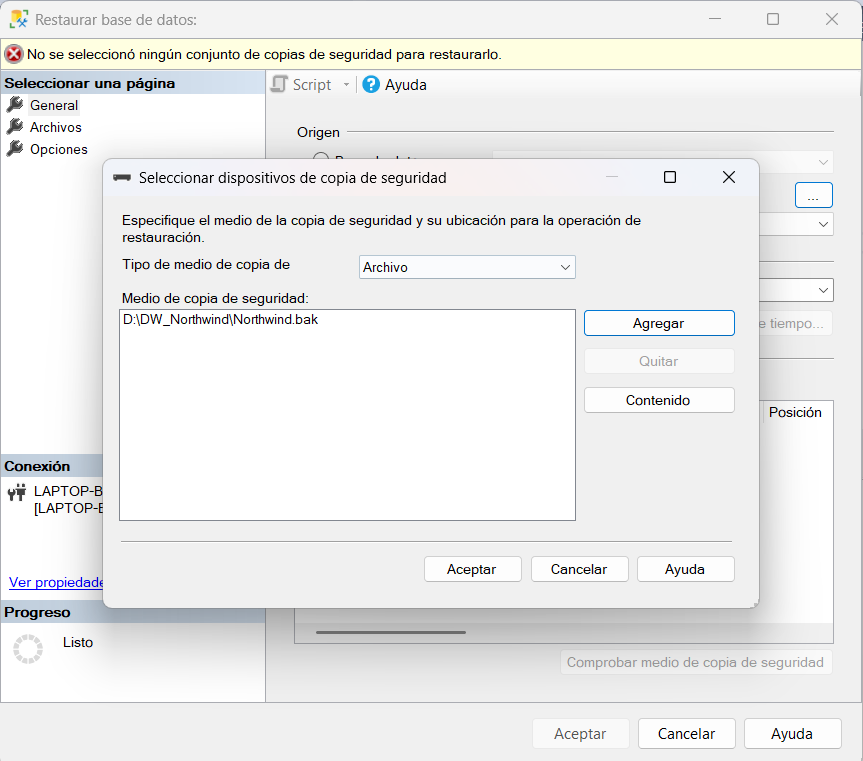
Evaluar el rendimiento de empleados y regiones geográficas

1. Descarga de base de datos elegida y restauración con MS SQL Server Management Studio



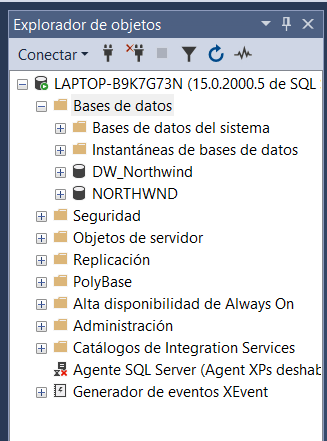
Como se puede apreciar en la captura, Estamos dentro de una carpeta llamada Northwind-master, la cual descargamos del enlace adjunto en el Word subido a la intranet, dentro de esta carpeta se encuentra el Northwind.bak que aun no esta descomprimido, lo cual haremos más adelante

1. Restauración de la base de datos con MS SQL Server Management Studio.



Para poder iniciar el proceso de restauración de la base de datos Northwind descargada previamente debemos descomprimir el archivo que se menciono en el punto anterior, el .bak que nos da al descomprimirlo, lo mandamos al disco D:/ del dispositivo en uso, antes de esto, se debió crear una carpeta la cual llamamos DW\_Northwind al cual le introducimos el archivo .bak

Luego dentro del management studio, Le damos click derecho al apartado que dice “Base de datos” Luego nos vamos a “Restaurar base de datos” el cual abrirá una interfaz donde deberemos seleccionar en donde buscar el archivo .bak, en este caso el archivo se encuentra en nuestro dispositivo, por lo cual seleccionamos esa casilla y luego buscamos en la carpeta en la cual lo introdujimos, finalmente darle en aceptar y esperar a que la base de datos sea restaurada



Luego de finalizar con el procedimiento anterior y que no haya ninguna dificultad en el proceso, debería aparecer dentro de la carpeta “Base de datos” la base de datos restaurada en este caso llamada “NORTHWND”

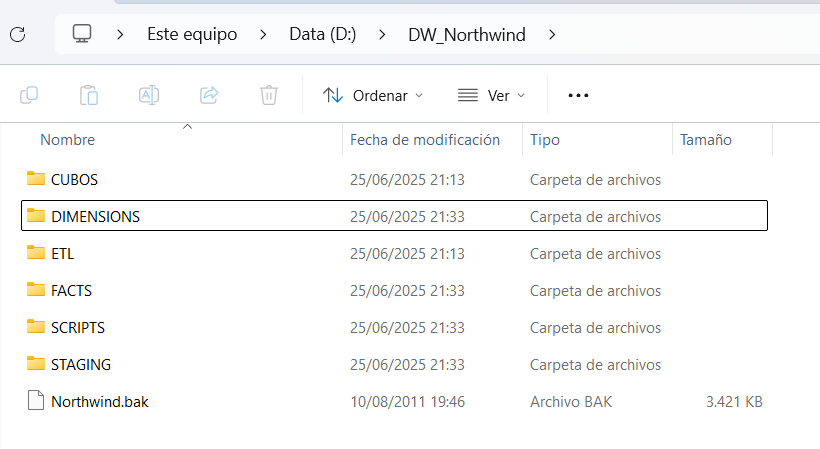
1. Creación de directorios y subdirectorios para prototipo de DW
2. Creación de directorio para prototipo de DW.

Interfaz de usuario gráfica, Texto, Correo electrónico

El contenido generado por IA puede ser incorrecto.

Como se puede apreciar, estoy en el disco D:/ de mi ordenador, dentro de el creamos el directorio llamado DW\_Northwind, para esto simplemente bastaba con ir al disco, dar click derecho y crear una nueva carpeta

1. Creación de subdirectorios para prototipo de DW.



Dentro del directorio DW\_Northwind se siguió el mismo procedimiento para los sub-directorios, Click derecho, nuevo y crear carpeta con el nombre correspondiente

1. Diseño de prototipo de DW

**Decisiones de Diseño del prototipo de DW.**

Para el diseño del prototipo de Data Waterhouse optamos por un esquema dimensional en estrella, ya que permite organizar los datos de forma clara y optimizada para análisis.

**Tabla de Hechos: FactSales**

Definimos una única tabla de hechos llamada FactSales, que representa las transacciones de ventas realizadas por la empresa. Esta tabla contiene las siguientes medidas:

Quantity: cantidad de productos vendidos por pedido.

UnitPrice: precio unitario del producto.

Total: valor total de la venta, calculado como Quantity \* UnitPrice.

La tabla también almacena claves foráneas hacia cada una de las dimensiones, permitiendo conectar los hechos con sus respectivos contextos.

**Dimensiones seleccionadas**

**DimCustomer**  
Derivada de la tabla Customers, contiene información de los clientes como CompanyName, ContactName y Country. Esta dimensión permite analizar las ventas por cliente y ubicación.

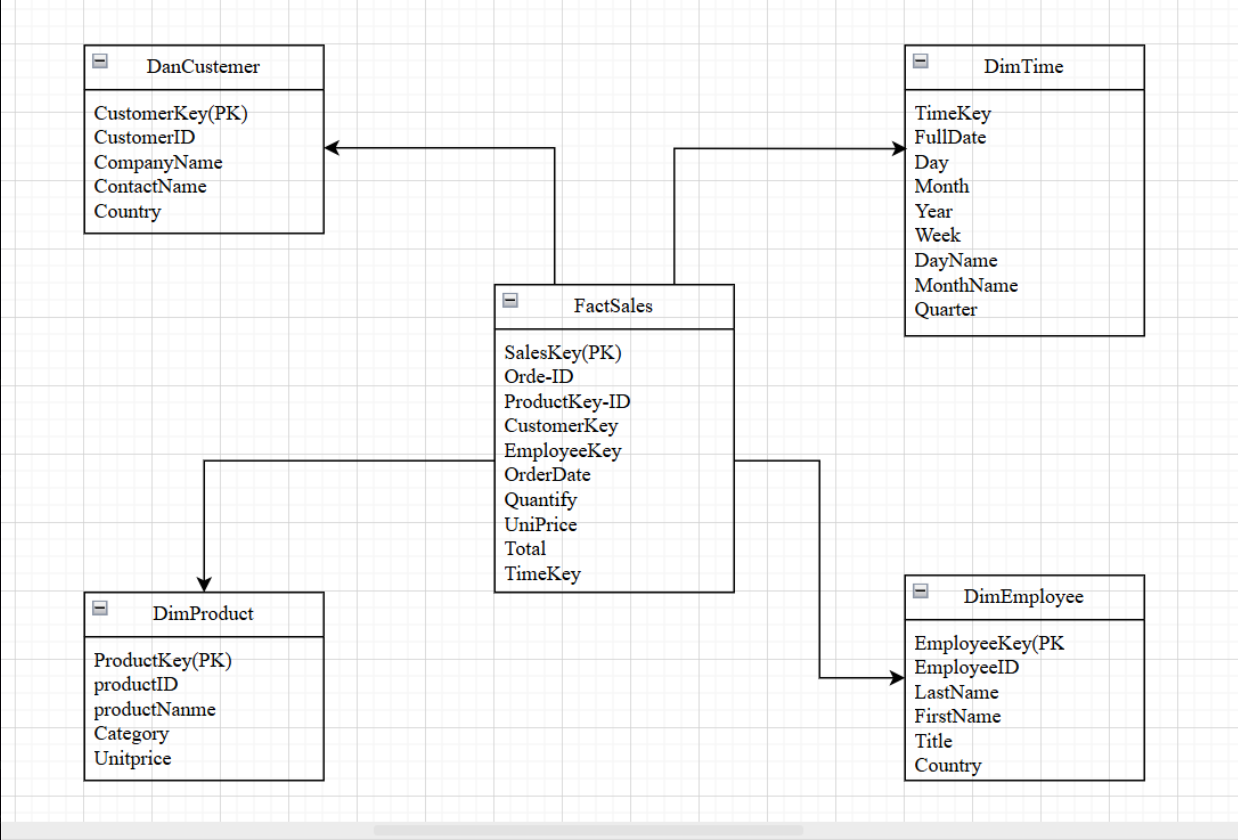
**DimProduct**:  
Fusiona las tablas Products y Categories de la base OLTP para desnormalizar los datos y simplificar el acceso. Incluye campos como ProductName, UnitPrice y CategoryName, permitiendo analizar qué productos se venden más y en qué categorías.

**DimEmployee:**  
Basada en la tabla Employees, permite conocer qué empleado gestionó cada venta. Contiene campos como LastName, Title y Country, útiles para evaluar desempeño por vendedor o por región.

**Justificación del diseño:**

La elección de estas dimensiones responde a la necesidad de analizar los datos de ventas desde las perspectivas más relevantes: quién compra (cliente), qué se vende (producto) y quién lo vende (empleado). Tambien se fusionaron tablas como Products y Categories, lo que mejora el rendimiento de las consultas analíticas.

1. Esquema estrella del prototipo de DW



1. Creación del prototipo de DW
2. **Script de eliminación de DW, creación de filegroups, configuración de modelo de recuperación, creación de particiones.**

-- Eliminar DW anterior si existe

USE master;

IF EXISTS (SELECT \* FROM sys.databases WHERE name = 'DW\_Northwind')

BEGIN

ALTER DATABASE DW\_Northwind SET SINGLE\_USER WITH ROLLBACK IMMEDIATE;

DROP DATABASE DW\_Northwind;

END

GO

-- Crear nueva base de datos DW\_Northwind

CREATE DATABASE DW\_Northwind

ON

PRIMARY (

NAME = 'DW\_Northwind\_data',

FILENAME = 'D:\DW\_Northwind\SCRIPTS\DW\_Northwind\_data.mdf',

SIZE = 50MB,

FILEGROWTH = 10MB

),

FILEGROUP DIMENSIONS (

NAME = 'DW\_Northwind\_dimensions',

FILENAME = 'D:\DW\_Northwind\DIMENSIONS\DW\_Northwind\_dimensions.ndf',

SIZE = 50MB,

FILEGROWTH = 10MB

),

FILEGROUP FACTS (

NAME = 'DW\_Northwind\_facts',

FILENAME = 'D:\DW\_Northwind\FACTS\DW\_Northwind\_facts.ndf',

SIZE = 50MB,

FILEGROWTH = 10MB

),

FILEGROUP STAGING (

NAME = 'DW\_Northwind\_staging',

FILENAME = 'D:\DW\_Northwind\STAGING\DW\_Northwind\_staging.ndf',

SIZE = 50MB,

FILEGROWTH = 10MB

)

LOG ON (

NAME = 'DW\_Northwind\_log',

FILENAME = 'D:\DW\_Northwind\SCRIPTS\DW\_Northwind\_log.ldf',

SIZE = 20MB,

FILEGROWTH = 10MB

);

GO

-- Establecer modelo de recuperación SIMPLE

ALTER DATABASE DW\_Northwind SET RECOVERY SIMPLE;

GO

1. **Script de creación de tablas de prototipo del DW (dimensiones y hecho).**

USE DW\_Northwind;

GO

-- Crear esquemas si no existen

IF NOT EXISTS (SELECT \* FROM sys.schemas WHERE name = 'DIMENSIONS')

EXEC('CREATE SCHEMA DIMENSIONS');

IF NOT EXISTS (SELECT \* FROM sys.schemas WHERE name = 'FACTS')

EXEC('CREATE SCHEMA FACTS');

-- Crear la Tabla DimCustomer

CREATE TABLE DIMENSIONS.DimCustomer (

CustomerKey INT IDENTITY(1,1) PRIMARY KEY,

CustomerID NVARCHAR(5), -- ID original de Northwind

CompanyName NVARCHAR(40),

ContactName NVARCHAR(30),

Country NVARCHAR(15)

);

GO

-- Crear la Tabla DimProduct

CREATE TABLE DIMENSIONS.DimProduct (

ProductKey INT IDENTITY(1,1) PRIMARY KEY,

ProductID INT, -- ID original de Northwind

ProductName NVARCHAR(40),

Category NVARCHAR(15),

UnitPrice MONEY

);

GO

-- Crear la Tabla DimEmployee

CREATE TABLE DIMENSIONS.DimEmployee (

EmployeeKey INT IDENTITY(1,1) PRIMARY KEY,

EmployeeID INT, -- ID original de Northwind

LastName NVARCHAR(20),

FirstName NVARCHAR(10),

Title NVARCHAR(30),

Country NVARCHAR(15)

);

GO

-- Crear la Tabla de hechos: ventas

CREATE TABLE FACTS.FactSales (

SalesKey INT IDENTITY(1,1) PRIMARY KEY,

OrderID INT,

ProductKey INT,

CustomerKey INT,

EmployeeKey INT,

OrderDate DATE,

Quantity INT,

UnitPrice MONEY,

Total MONEY,

FOREIGN KEY (ProductKey) REFERENCES DIMENSIONS.DimProduct(ProductKey),

FOREIGN KEY (CustomerKey) REFERENCES DIMENSIONS.DimCustomer(CustomerKey),

FOREIGN KEY (EmployeeKey) REFERENCES DIMENSIONS.DimEmployee(EmployeeKey)

);

GO

* 1. **Se puede utilizar el siguiente comando luego de la creación de las tablas para observar que se crearon de forma adecuada:**

SELECT TABLE\_SCHEMA, TABLE\_NAME

FROM INFORMATION\_SCHEMA.TABLES

WHERE TABLE\_SCHEMA IN ('DIMENSIONS', 'FACTS');

**Nuevo:**

**2.2 -- Crear la tabla DimTime**

USE DW\_Northwind;

GO

IF NOT EXISTS (SELECT \* FROM sys.schemas WHERE name = 'DIMENSIONS')

EXEC('CREATE SCHEMA DIMENSIONS');

GO

CREATE TABLE DIMENSIONS.DimTime (

TimeKey INT PRIMARY KEY, -- YYYYMMDD

FullDate DATE,

Day TINYINT,

Month TINYINT,

Year SMALLINT,

Week INT,

DayName NVARCHAR(10),

MonthName NVARCHAR(10),

Quarter TINYINT

);

GO

DECLARE @StartDate DATE = '1996-01-01';

DECLARE @EndDate DATE = '1998-12-31';

WHILE @StartDate <= @EndDate

BEGIN

INSERT INTO DIMENSIONS.DimTime (

TimeKey, FullDate, Day, Month, Year, Week, DayName, MonthName, Quarter

)

VALUES (

CONVERT(INT, FORMAT(@StartDate, 'yyyyMMdd')),

@StartDate,

DATEPART(DAY, @StartDate),

DATEPART(MONTH, @StartDate),

DATEPART(YEAR, @StartDate),

DATEPART(WEEK, @StartDate),

DATENAME(WEEKDAY, @StartDate),

DATENAME(MONTH, @StartDate),

DATEPART(QUARTER, @StartDate)

);

SET @StartDate = DATEADD(DAY, 1, @StartDate);

END;

GO

**2.3 Actualizar la table factSales para agregar la nueva dimension y jerarquias**

ALTER TABLE FACTS.FactSales

ADD TimeKey INT;

GO

UPDATE FACTS.FactSales

SET TimeKey = CONVERT(INT, FORMAT(OrderDate, 'yyyyMMdd'));

GO

ALTER TABLE FACTS.FactSales

ADD CONSTRAINT FK\_FactSales\_TimeKey

FOREIGN KEY (TimeKey) REFERENCES DIMENSIONS.DimTime(TimeKey);

GO

1. **Script de deshabilitación de restricciones, carga de datos en cada tabla dimensión y en la tabla de hechos y habilitación de restricciones.**

-- Deshabilitar restricciones (por si hay FOREIGN KEYS que interfieran)

ALTER TABLE FACTS.FactSales NOCHECK CONSTRAINT ALL;

ALTER TABLE DIMENSIONS.DimCustomer NOCHECK CONSTRAINT ALL;

ALTER TABLE DIMENSIONS.DimProduct NOCHECK CONSTRAINT ALL;

ALTER TABLE DIMENSIONS.DimEmployee NOCHECK CONSTRAINT ALL;

-- Insertar datos en DimCustomer desde Northwind.dbo.Customers

INSERT INTO DIMENSIONS.DimCustomer (CustomerID, CompanyName, ContactName, Country)

SELECT CustomerID, CompanyName, ContactName, Country

FROM Northwnd.dbo.Customers;

-- Insertar datos en DimProduct desde Northwind.dbo.Products y Categories

INSERT INTO DIMENSIONS.DimProduct (ProductID, ProductName, Category, UnitPrice)

SELECT

p.ProductID,

p.ProductName,

c.CategoryName,

p.UnitPrice

FROM Northwnd.dbo.Products p

JOIN Northwnd.dbo.Categories c ON p.CategoryID = c.CategoryID;

-- Insertar datos en DimEmployee desde Northwind.dbo.Employees

INSERT INTO DIMENSIONS.DimEmployee (EmployeeID, LastName, FirstName, Title, Country)

SELECT EmployeeID, LastName, FirstName, Title, Country

FROM Northwnd.dbo.Employees;

-- Insertar datos en tabla de hechos

INSERT INTO FACTS.FactSales (

OrderID, ProductKey, CustomerKey, EmployeeKey, OrderDate, Quantity, UnitPrice, Total

)

SELECT

o.OrderID,

dp.ProductKey,

dc.CustomerKey,

de.EmployeeKey,

o.OrderDate,

od.Quantity,

od.UnitPrice,

(od.Quantity \* od.UnitPrice) AS Total

FROM Northwnd.dbo.Orders o

JOIN Northwnd.dbo.[Order Details] od ON o.OrderID = od.OrderID

-- Solución collation segura:

JOIN DIMENSIONS.DimProduct dp ON dp.ProductID = od.ProductID

JOIN DIMENSIONS.DimCustomer dc ON

dc.CustomerID = o.CustomerID COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS

JOIN DIMENSIONS.DimEmployee de ON

de.EmployeeID = CAST(o.EmployeeID AS VARCHAR) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS;

-- Rehabilitar restricciones

ALTER TABLE FACTS.FactSales CHECK CONSTRAINT ALL;

ALTER TABLE DIMENSIONS.DimCustomer CHECK CONSTRAINT ALL;

ALTER TABLE DIMENSIONS.DimProduct CHECK CONSTRAINT ALL;

ALTER TABLE DIMENSIONS.DimEmployee CHECK CONSTRAINT ALL;

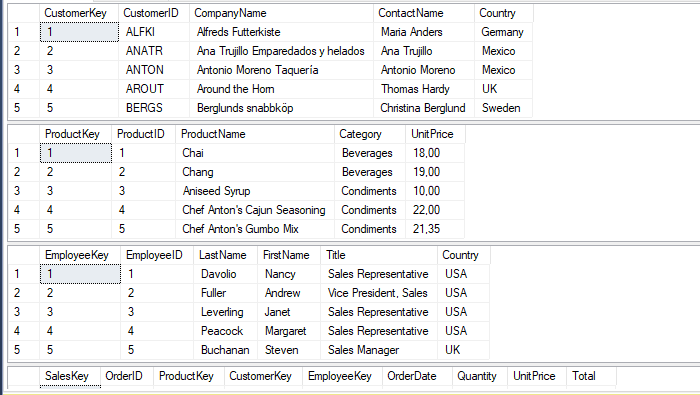
* 1. **Ingresar el siguiente comando para verificar que se agregaron valores a las tablas, en el número “5” en este caso, se podría aumentar o disminuir para que muestre más o menos datos, dependiendo lo que uno requiera, es más que nada para verificar que los datos si se muestran**

SELECT TOP 5 \* FROM DIMENSIONS.DimCustomer;

SELECT TOP 5 \* FROM DIMENSIONS.DimProduct;

SELECT TOP 5 \* FROM DIMENSIONS.DimEmployee;

SELECT TOP 5 \* FROM FACTS.FactSales;



1. **Script de manejo de restricciones en tablas de dimensiones, hechos y restricciones.**

-- Clave única para CustomerID en DimCustomer

ALTER TABLE DIMENSIONS.DimCustomer

ADD CONSTRAINT UQ\_CustomerID UNIQUE (CustomerID);

-- Clave única para ProductID en DimProduct

ALTER TABLE DIMENSIONS.DimProduct

ADD CONSTRAINT UQ\_ProductID UNIQUE (ProductID);

-- Clave única para EmployeeID en DimEmployee

ALTER TABLE DIMENSIONS.DimEmployee

ADD CONSTRAINT UQ\_EmployeeID UNIQUE (EmployeeID);

-- Validar que precios y cantidades no sean negativos

ALTER TABLE FACTS.FactSales

ADD CONSTRAINT CK\_Quantity\_Positive CHECK (Quantity >= 0);

ALTER TABLE FACTS.FactSales

ADD CONSTRAINT CK\_UnitPrice\_Positive CHECK (UnitPrice >= 0);

ALTER TABLE FACTS.FactSales

ADD CONSTRAINT CK\_Total\_Positive CHECK (Total >= 0);

-- Rehabilitar restricciones foráneas manualmente (por si alguna falla)

ALTER TABLE FACTS.FactSales

WITH CHECK CHECK CONSTRAINT ALL;

**4.1 En el siguiente comando simulamos la inserción de datos que incumplen las restricciones puestas, por lo que, al ejecutarlo, debería arrojar error**

-- Intento con Quantity negativo (debería fallar)

INSERT INTO FACTS.FactSales (OrderID, ProductKey, CustomerKey, EmployeeKey, OrderDate, Quantity, UnitPrice, Total)

VALUES (99999, 1, 1, 1, GETDATE(), -5, 10, -50);

-- Intento con UnitPrice negativo (debería fallar)

INSERT INTO FACTS.FactSales (OrderID, ProductKey, CustomerKey, EmployeeKey, OrderDate, Quantity, UnitPrice, Total)

VALUES (99998, 1, 1, 1, GETDATE(), 5, -10, -50);

-- Intento con Total negativo (debería fallar)

INSERT INTO FACTS.FactSales (OrderID, ProductKey, CustomerKey, EmployeeKey, OrderDate, Quantity, UnitPrice, Total)

VALUES (99997, 1, 1, 1, GETDATE(), 5, 10, -50);

1. **Script de creación de índices para tablas de dimensiones y de hechos.**

-- Índices en DimCustomer

CREATE NONCLUSTERED INDEX IX\_DimCustomer\_CustomerID

ON DIMENSIONS.DimCustomer (CustomerID);

-- Índices en DimProduct

CREATE NONCLUSTERED INDEX IX\_DimProduct\_ProductID

ON DIMENSIONS.DimProduct (ProductID);

-- Índices en DimEmployee

CREATE NONCLUSTERED INDEX IX\_DimEmployee\_EmployeeID

ON DIMENSIONS.DimEmployee (EmployeeID);

-- Índice por claves foráneas

CREATE NONCLUSTERED INDEX IX\_FactSales\_ProductKey

ON FACTS.FactSales (ProductKey);

CREATE NONCLUSTERED INDEX IX\_FactSales\_CustomerKey

ON FACTS.FactSales (CustomerKey);

CREATE NONCLUSTERED INDEX IX\_FactSales\_EmployeeKey

ON FACTS.FactSales (EmployeeKey);

-- Índice en OrderDate para filtros por fechas

CREATE NONCLUSTERED INDEX IX\_FactSales\_OrderDate

ON FACTS.FactSales (OrderDate);

Realización de pequeño ejemplo de ETL

Diagrama

El contenido generado por IA puede ser incorrecto.

Explicación de creación de proyecto de ETL con Analysis Services.

Para este proceso tuve que crear un nuevo proyecto en visual studio 2019, con la opción de integration services Project, luego de eso, en el administrador de conexiones  
  
Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

Seleccionar nuevo administrador de conexiones, lo cual nos abría el siguiente apartado donde debíamos buscar OLE DB el cual era el que nos serviría para los procedimientos requeridos

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Luego ahí dentro, agregar tanto la Base de datos utilizada y el data warehouse creado, luego de tener todo eso listo, buscábamos la opción para la creación de un área de flujo de datos  
  
Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Luego, arrastrar la opción dentro del cuadro a trabajar y dar doble click a la opción, eso nos llevaba al apartado de flujo de datos, en el cual debíamos meter la opción de Origen de OLE DB, luego ahí dentro, seleccionar la conexión que haríamos en este caso y a que tabla iba dirigida, luego de tener esa opción seleccionábamos un conversor de datos, para cambiar el tipo de datos de algunos parámetros por si tiraba errores, en este caso me dio varios errores con el tema de la “Photo” porque lo queríamos pasar a un Txt, entonces chocaban esas 2 partes, entonces preferiblemente, si solo queremos ver texto, desabilite la opción de la Photo y deje el resto.

Luego de todo eso, elegimos nuestra opción de destino, la cual seria el destino de archivo plano, para almacenar nuestra información en un txt  
  
Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.  
  
Dentro de esa opción, se configuraba en que parte iba a ser guardada la información, que en este caso, la dejamos guardada dentro de la carpeta del proyecto, en un lugar especialmente llamado “ETL”

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

Estos valores dados aquí son los TXT que devuelve mi procedimiento ETL con los valores de customer, employees y products

**NUEVO: creación del cubo Multidimencional**

1. **Selección del Origen de Datos**

Primero, seleccionamos el origen de datos que contiene la información que vamos a analizar. Este origen puede ser una base de datos relacional, como SQL Server, que contenga todas las tablas necesarias para nuestro modelo analítico.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

1. **Incorporación de Tablas**

A continuación, incluimos todas las tablas disponibles que forman parte del modelo. Es importante asegurarse de que estén incluidas tanto la tabla de hechos como todas las dimensiones relacionadas. Esto permite construir un modelo que refleje fielmente las relaciones entre los datos.

Interfaz de usuario gráfica, Texto

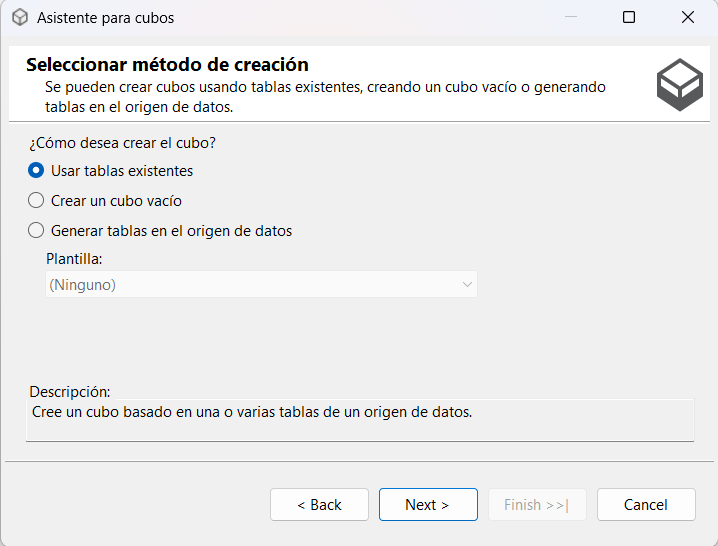
El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

1. **Creación del Cubo**

Nos dirigimos a la sección de Cubos, hacemos clic derecho y seleccionamos la opción 'Crear nuevo cubo'. Seleccionamos donde dice usar tablas existentes



Seleccionamos únicamente nuestra tabla de hechos la cual es factSales

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Luego de esto simplemente damos Next hasta que nos deje finalizar la operacion

así nuestro Cubo Multidimensional ya estaría creado

Diagrama

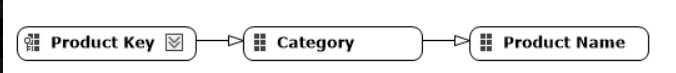
El contenido generado por IA puede ser incorrecto.

**jerarquías del cubo**

**Jerarquias de tiempo:**

****

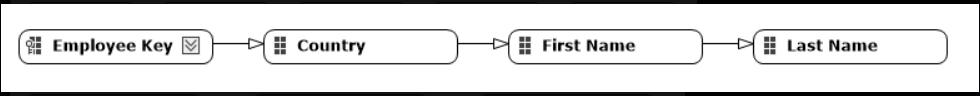
**Jerarquias de productos:**

****

**Jerarquía de costomers:**

****

**Jerarquía de employees:**

****