Proyecto Final

Arguedas Oscar

Benavides Francisco

Universidad CENFOTEC

MBD304 Bases De Datos Multidimensionales

Jose Cabezas Jaikel

Diciembre, 2023

Introducción

Este proyecto tiene como objetivo desarrollar un completo sistema de Data Warehouse utilizando como base la base de datos transaccional de World Wide Importers, una empresa ficticia utilizada como ejemplo en SQL Server. Este proyecto se organiza en varias fases clave que asegurarán la creación eficiente y eficaz de un almacén de datos robusto, escalable y de alto rendimiento, orientado a mejorar la toma de decisiones y el análisis de datos en la empresa.

Fase 1: Obtención y Aprobación de la Base de Datos Transaccional

Primero, seleccionaremos una base de datos transaccional adecuada, que será World Wide Importers en este caso. Esta base de datos ya está construida en SQL Server, y será suministrada para su aprobación antes de iniciar cualquier desarrollo. Su estructura y contenido proporcionarán el fundamento sobre el cual construiremos nuestro Data Warehouse.

Fase 2: Diseño y Creación de la Base de Datos Data Warehouse

Utilizando Azure SQL Server y SSIS, transformaremos la base de datos transaccional en una base de datos Data Warehouse. Esto incluirá:

- **Modelo Dimensional**: Seleccionaremos entre un modelo en estrella o copo de nieve para organizar nuestras tablas de dimensiones y hechos.
- **Dimensiones y Tablas de Hechos**: Crearemos al menos 10 dimensiones y definiremos claramente las tablas de hechos, enfocándonos en los aspectos clave de los negocios de World Wide Importers.
- **Diagramas**: Desarrollaremos diagramas detallados para visualizar la estructura del Data Warehouse, facilitando su comprensión y validación.

Fase 3: Implementación de Procesos ETL con SSIS

Desarrollaremos un proyecto de Integration Services (SSIS) que manejará la carga de datos de manera incremental en nuestro Data Warehouse. Este proceso incluirá:

- **Cargas Incrementales**: Estableceremos mecanismos para actualizar la base de datos con los cambios más recientes de la base de datos transaccional.

- **Tabla de Configuración**: Implementaremos una tabla de configuración que registrará automáticamente la fecha de cada carga de datos y permitirá recargas según sea necesario.

Fase 4: Proyecto de Analysis Services

En esta fase, nos enfocaremos en:

- **Base de Datos Multidimensional**: Crearemos y configuraremos una base de datos multidimensional utilizando SQL Server Analysis Services (SSAS).

- **Relevancia y Nomenclatura**: Nos aseguraremos de que todas las dimensiones, atributos y métricas sean relevantes para el cliente y estén nombradas adecuadamente.

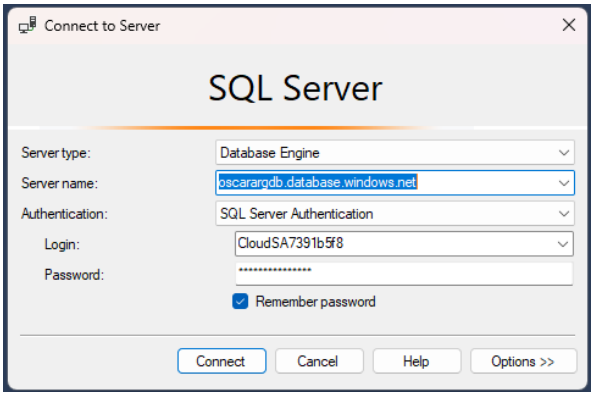
- **Perspectivas del Cubo**: Desarrollaremos perspectivas específicas del cubo, seleccionando elementos basados en su importancia para el análisis de negocios y justificando estas elecciones en nuestra documentación.

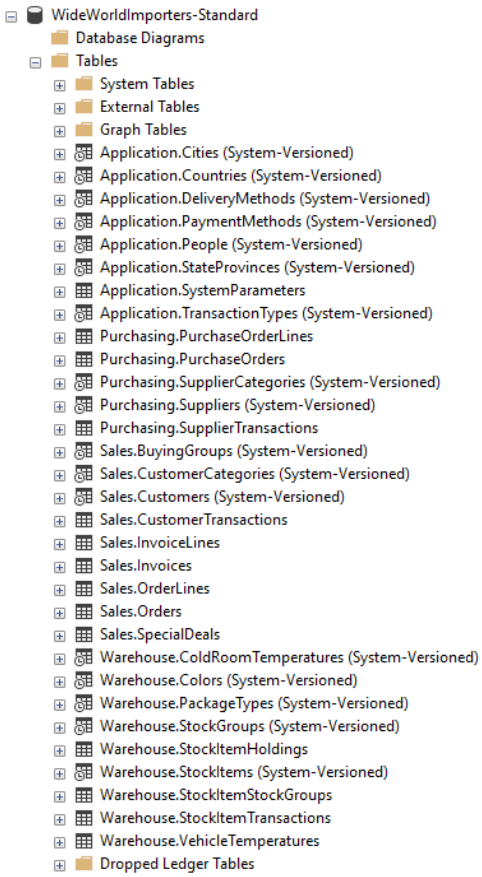
- **Traducción**: Haremos que el cubo esté disponible en inglés y español, facilitando su uso por parte de usuarios en diferentes regiones.

Tabla de contenido

Desarrollo

Para el desarrollo del proyecto se utiliza la base de datos WideWorldImporters, a continuación, se utilizó Azure para instalar la base de datos, información sobre la base de datos.





Script Creación DataWareHouse, se utiliza la tabla ControlDataLoad para llevar un control de las cargas realizadas en las dimensiones y en la tabla de hecho, la tabla se compone de ID autogenerado, una fecha de Carga y el nombre de tabla, en cada tabla de dimensiones y hecho se asocia el número de control al momento de crear un nuevo registro.

Script Base de Datos

```
--SCRIPT PROYECTO FINAL MBD304 Bases De Datos Multidimensionales

--TABLA DE CONTROL DE CARGAS
CREATE DATABASE WWI_DW;
GO

USE WWI_DW;
GO

IF OBJECT_ID('dbo.ControlDataLoad', 'U') IS NOT NULL
BEGIN
    -- Si existe, eliminar la tabla
    DROP TABLE dbo.ControlDataLoad;
END
--
CREATE TABLE ControlDataLoad (
    [ControlID]            INT            NOT NULL IDENTITY(1,1),
    [DataLoadStarted]      DATE           NOT NULL DEFAULT  GETDATE(),
    [TableName]            NVARCHAR (255) NOT NULL,
    [DataLoadCompleted]    DATETIME2 (7) NULL,
    CONSTRAINT [PK_ControlDataLoad] PRIMARY KEY CLUSTERED ([ControlID] ASC)
);
--
/*
DimCity
DimCustomer
DimDate
DimEmployee
DimPaymentMethod
DimStockItem
DimSupplier
DimTrasactionTyp

*/
/*SCRIPT DE TABLAS DE DIMENSIONES*/
IF OBJECT_ID('dbo.DimCity', 'U') IS NOT NULL
BEGIN
    -- Si existe, eliminar la tabla
    DROP TABLE dbo.DimCity;
END
CREATE TABLE [dbo].[DimCity] (
    [DimCityID]      [INT]             NOT NULL IDENTITY(1,1),
    [CityID]         [INT]             NOT NULL,
```

```
    [City]           [NVARCHAR] (255)  NOT NULL,
    [StateProvince]  [NVARCHAR] (255)  NOT NULL,
    [Country]        [NVARCHAR] (255)  NOT NULL,
    [Continent]      [NVARCHAR] (255)  NOT NULL,
    [SalesTerritory] [NVARCHAR] (255)  NOT NULL,
    [Region]         [NVARCHAR] (255)  NOT NULL,
    [Subregion]      [NVARCHAR] (255)  NOT NULL,
    [Location]       [sys].[geography] NOT NULL,
    [ControlID]      [INT]             NOT NULL  ,
    CONSTRAINT [PK_Dimension_City] PRIMARY KEY CLUSTERED ([DimCityID] ASC)
);
ALTER TABLE [dbo].[DimCity] WITH CHECK ADD CONSTRAINT
[FK_DimCity_ControlDataLoad] FOREIGN KEY([ControlID])
REFERENCES [dbo].[ControlDataLoad] ([ControlID]);

--TRIGGER
CREATE OR ALTER TRIGGER [dbo].[TRG_CONTROL_CITY] ON [dbo].[DimCity]
AFTER INSERT
AS
BEGIN

    DECLARE @ControlIDs table (id INT);
    DECLARE @ControlID INT;
    --
    INSERT INTO [dbo].[ControlDataLoad] (DATALOADSTARTED, TABLENAME)
    OUTPUT INSERTED.ControlID INTO @ControlIDs
    SELECT GETDATE(), 'DimCity';

    SET @ControlID  = (SELECT TOP 1 ID FROM @ControlIDs);
    --
    UPDATE dc
    SET dc.ControlID = @ControlID
    FROM [dbo].[DimCity] dc
    INNER JOIN INSERTED i ON dc.CityID = i.CityID;
END
GO
/*************************************************************************
**********/
--DIMENSION CUSTOMER
IF OBJECT_ID('dbo.DimCustomer', 'U') IS NOT NULL
BEGIN
    -- Si existe, eliminar la tabla
    DROP TABLE dbo.[DimCustomer];
END
CREATE TABLE [dbo].[DimCustomer] (
```

```
    [DimCustomerID]  [int] NOT NULL IDENTITY(1,1),
    [CustomerID]     [int] NOT NULL,
    [Customer]       [nvarchar](255) NOT NULL,
    [BillToCustomer] [nvarchar](255) NOT NULL,
    [Category]       [nvarchar](255) NOT NULL,
    [BuyingGroup]    [nvarchar](255) NOT NULL,
    [PrimaryContact] [nvarchar](255) NOT NULL,
    [PostalCode]     [nvarchar](255) NOT NULL,
    [ControlID]      [int]
     CONSTRAINT [PK_Dimension_Customer] PRIMARY KEY CLUSTERED
([DimCustomerID] ASC)
);
ALTER TABLE [dbo].[DimCustomer] WITH CHECK ADD CONSTRAINT
[FK_DimCustomer_ControlDataLoad] FOREIGN KEY([ControlID])
REFERENCES [dbo].[ControlDataLoad] ([ControlID]);

--TRIGGER
CREATE OR ALTER TRIGGER [dbo].[TRG_CONTROL_CUSTOMER] ON [dbo].[DimCustomer]
AFTER INSERT
AS
BEGIN

    DECLARE @ControlIDs table (id INT);
    DECLARE @ControlID INT;
    --
    INSERT INTO [dbo].[ControlDataLoad] (DATALOADSTARTED, TABLENAME)
    OUTPUT INSERTED.ControlID INTO @ControlIDs
    SELECT GETDATE(), 'DimCustomer';

    SET @ControlID  = (SELECT TOP 1 ID FROM @ControlIDs);
    --
    UPDATE dc
    SET dc.ControlID = @ControlID
    FROM [dbo].[DimCustomer] dc
    INNER JOIN INSERTED i ON dc.CustomerID = i.CustomerID;
END
GO
/**************************************************************************
**********/
--DIMENSION EMPLOYEE
IF OBJECT_ID('dbo.DimEmployee', 'U') IS NOT NULL
BEGIN
    -- Si existe, eliminar la tabla
    DROP TABLE dbo.[DimEmployee];
END
```

```
CREATE TABLE [dbo].[DimEmployee] (
    [DimEmployeeID]  [int] NOT NULL IDENTITY(1,1),
    [EmployeeID]     [int] NOT NULL,
    [Employee]       [nvarchar](255) NOT NULL,
    [PreferredName]  [nvarchar](255) NOT NULL,
    [IsSalesperson]  [bit] NOT NULL,
    [Photo]          [varbinary](max) NULL,
    [ControlID]      [int]
     CONSTRAINT [PK_Dimension_Employee] PRIMARY KEY CLUSTERED
([DimEmployeeID] ASC)
);
ALTER TABLE [dbo].[DimEmployee] WITH CHECK ADD CONSTRAINT
[FK_DimEmployee_ControlDataLoad] FOREIGN KEY([ControlID])
REFERENCES [dbo].[ControlDataLoad] ([ControlID]);
--TRIGGER
CREATE OR ALTER TRIGGER [dbo].[TRG_CONTROL_EMPLOYEE] ON [dbo].[DimEmployee]
AFTER INSERT
AS
BEGIN

    DECLARE @ControlIDs table (id INT);
    DECLARE @ControlID INT;
    --
    INSERT INTO [dbo].[ControlDataLoad] (DATALOADSTARTED, TABLENAME)
    OUTPUT INSERTED.ControlID INTO @ControlIDs
    SELECT GETDATE(), 'DimEmployee';

    SET @ControlID  = (SELECT TOP 1 ID FROM @ControlIDs);
    --
    UPDATE dc
    SET dc.ControlID = @ControlID
    FROM [dbo].[DimEmployee] dc
    INNER JOIN INSERTED i ON dc.EmployeeID = i.EmployeeID;
END
GO
/*************************************************************************
************************/
--DIMENSIONES FECHA
IF OBJECT_ID('dbo.DimDate', 'U') IS NOT NULL
BEGIN
    -- Si existe, eliminar la tabla
    DROP TABLE dbo.[DimDate];
END

CREATE TABLE DimDate (
```

```
    [DateID]      [INT] NOT NULL,
    [Date]        [DATE],
    [Day]         [INT],
    [Month]       [INT],
    [Year]        [INT],
    [Weekday]     [INT],
    [WeekdayName] [NVARCHAR](255),
    [MonthName]   [NVARCHAR](255),
    [IsWeekend]   [BIT]
      CONSTRAINT [PK_Dimension_Date] PRIMARY KEY CLUSTERED ([DateID] ASC)
);
--llenar tabla dimension fechas
CREATE OR ALTER PROCEDURE PopulateDimDate
AS
BEGIN
    DECLARE @StartDate DATE = '2010-01-01';
    DECLARE @EndDate   DATE = '2023-12-31';

    WHILE @StartDate <= @EndDate
    BEGIN
        INSERT INTO DimDate (
            DateID,
            Date,
            Day,
            Month,
            Year,
            Weekday,
            WeekdayName,
            MonthName,
            IsWeekend
        )
        VALUES (
            CONVERT(INT, FORMAT(@StartDate, 'yyyyMMdd')),
            @StartDate,
            DAY(@StartDate),
            MONTH(@StartDate),
            YEAR(@StartDate),
            DATEPART(WEEKDAY, @StartDate),
            FORMAT(@StartDate, 'dddd'),
            FORMAT(@StartDate, 'MMMM'),
            CASE WHEN DATEPART(WEEKDAY, @StartDate) IN (1, 7) THEN 1 ELSE 0
END
        );

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

```
    END;
END;
--
EXEC PopulateDimDate;
/*************************************************************************
************************/

--DIMENSION PAYMENT METHOD
IF OBJECT_ID('dbo.DimPaymentMethod', 'U') IS NOT NULL
BEGIN
    -- Si existe, eliminar la tabla
    DROP TABLE dbo.DimPaymentMethod;
END
CREATE TABLE [dbo].[DimPaymentMethod] (

    [DimPaymentMethodID] [INT]           NOT NULL IDENTITY(1,1),
    [PaymentMethodID]    [INT]           NOT NULL,
    [Payment Method]     [nvarchar](255) NOT NULL,
    [ControlID]          [INT]           NOT NULL,
    CONSTRAINT [PK_Dimension_PaymentMethod] PRIMARY KEY CLUSTERED
([DimPaymentMethodID] ASC)
);
ALTER TABLE [dbo].[DimPaymentMethod] WITH CHECK ADD CONSTRAINT
[FK_DimPaymentMethod_ControlDataLoad] FOREIGN KEY([ControlID])
REFERENCES [dbo].[ControlDataLoad] ([ControlID]);
--TRIGGER
CREATE OR ALTER TRIGGER [dbo].[TRG_CONTROL_PAYMENTMETHOD] ON
[dbo].[DimPaymentMethod]
AFTER INSERT
AS
BEGIN

    DECLARE @ControlIDs table (id INT);
    DECLARE @ControlID INT;
    --
    INSERT INTO [dbo].[ControlDataLoad] (DATALOADSTARTED, TABLENAME)
    OUTPUT INSERTED.ControlID INTO @ControlIDs
    SELECT GETDATE(), 'DimPaymentMethod';

    SET @ControlID  = (SELECT TOP 1 ID FROM @ControlIDs);
    --
    UPDATE dc
    SET dc.ControlID = @ControlID
    FROM [dbo].[DimPaymentMethod] dc
    INNER JOIN INSERTED i ON dc.PaymentMethodID = i.PaymentMethodID;
```

```
END
GO
/*************************************************************************
************************/
--DIMENSION DimStockItem
IF OBJECT_ID('dbo.DimStockItem', 'U') IS NOT NULL
BEGIN
    -- Si existe, eliminar la tabla
    DROP TABLE dbo.DimStockItem;
END
CREATE TABLE [dbo].[DimStockItem] (

    [DimStockItemID]         [INT]           NOT NULL IDENTITY(1,1),
    [StockItemID]            [INT]           NOT NULL,
    [StockItem]              [nvarchar](100) NOT NULL,
    [Color]                  [nvarchar](20)  NOT NULL,
    [SellingPackage]         [nvarchar](50)  NOT NULL,
    [BuyingPackage]          [nvarchar](50)  NOT NULL,
    [Brand]                  [nvarchar](50)  NOT NULL,
    [Size]                   [nvarchar](20)  NOT NULL,
    [LeadTimeDays]           [int] NOT NULL,
    [QuantityPerOuter]       [int] NOT NULL,
    [IsChillerStock]         [bit] NOT NULL,
    [Barcode]                [nvarchar](50)   NULL,
    [TaxRate]                [decimal](18, 3) NOT NULL,
    [UnitPrice]              [decimal](18, 2) NOT NULL,
    [RecommendedRetailPrice] [decimal](18, 2) NULL,
    [TypicalWeightPerUnit]   [decimal](18, 3) NOT NULL,
    [Photo]                  [varbinary](max) NULL,
    [ControlID]              [INT]            NOT NULL,
    CONSTRAINT [PK_Dimension_StockItem] PRIMARY KEY CLUSTERED
([DimStockItemID] ASC)
);
ALTER TABLE [dbo].[DimStockItem] WITH CHECK ADD CONSTRAINT
[FK_DimStockItem_ControlDataLoad] FOREIGN KEY([ControlID])
REFERENCES [dbo].[ControlDataLoad] ([ControlID]);
--TRIGGER
CREATE OR ALTER TRIGGER [dbo].[TRG_CONTROL_STOCKITEM] ON
[dbo].[DimStockItem]
AFTER INSERT
AS
BEGIN

    DECLARE @ControlIDs table (id INT);
    DECLARE @ControlID INT;
```

```
    --
    INSERT INTO [dbo].[ControlDataLoad] (DATALOADSTARTED, TABLENAME)
    OUTPUT INSERTED.ControlID INTO @ControlIDs
    SELECT GETDATE(), 'DimStockItem';

    SET @ControlID  = (SELECT TOP 1 ID FROM @ControlIDs);
    --
    UPDATE dc
    SET dc.ControlID = @ControlID
    FROM [dbo].[DimStockItem] dc
    INNER JOIN INSERTED i ON dc.StockItemID = i.StockItemID;
END
GO
/*************************************************************************
************************/
--DIMENSION DimSupplier

IF OBJECT_ID('dbo.DimSupplier', 'U') IS NOT NULL
BEGIN
    -- Si existe, eliminar la tabla
    DROP TABLE dbo.DimSupplier;
END

CREATE TABLE [dbo].[DimSupplier] (
    [DimSupplierID]     [INT]            NOT NULL IDENTITY(1,1),
    [SupplierID]        [INT]            NOT NULL,
    [Supplier]          [nvarchar](255)  NOT NULL,
    [Category]          [nvarchar](255)  NOT NULL,
    [PrimaryContact]    [nvarchar](255)  NOT NULL,
    [SupplierReference] [nvarchar](255)  NULL,
    [PaymentDays]       [int]            NOT NULL,
    [PostalCode]        [nvarchar](255)  NOT NULL,
    [ControlID]         [INT]            NOT NULL,
    CONSTRAINT [PK_Dimension_Supplier] PRIMARY KEY CLUSTERED
([DimSupplierID] ASC)
);
ALTER TABLE [dbo].[DimSupplier] WITH CHECK ADD CONSTRAINT
[FK_DimSupplier_ControlDataLoad] FOREIGN KEY([ControlID])
REFERENCES [dbo].[ControlDataLoad] ([ControlID]);
--TRIGGER
CREATE OR ALTER TRIGGER [dbo].[TRG_CONTROL_SUPPLIER] ON [dbo].[DimSupplier]
AFTER INSERT
AS
BEGIN
```

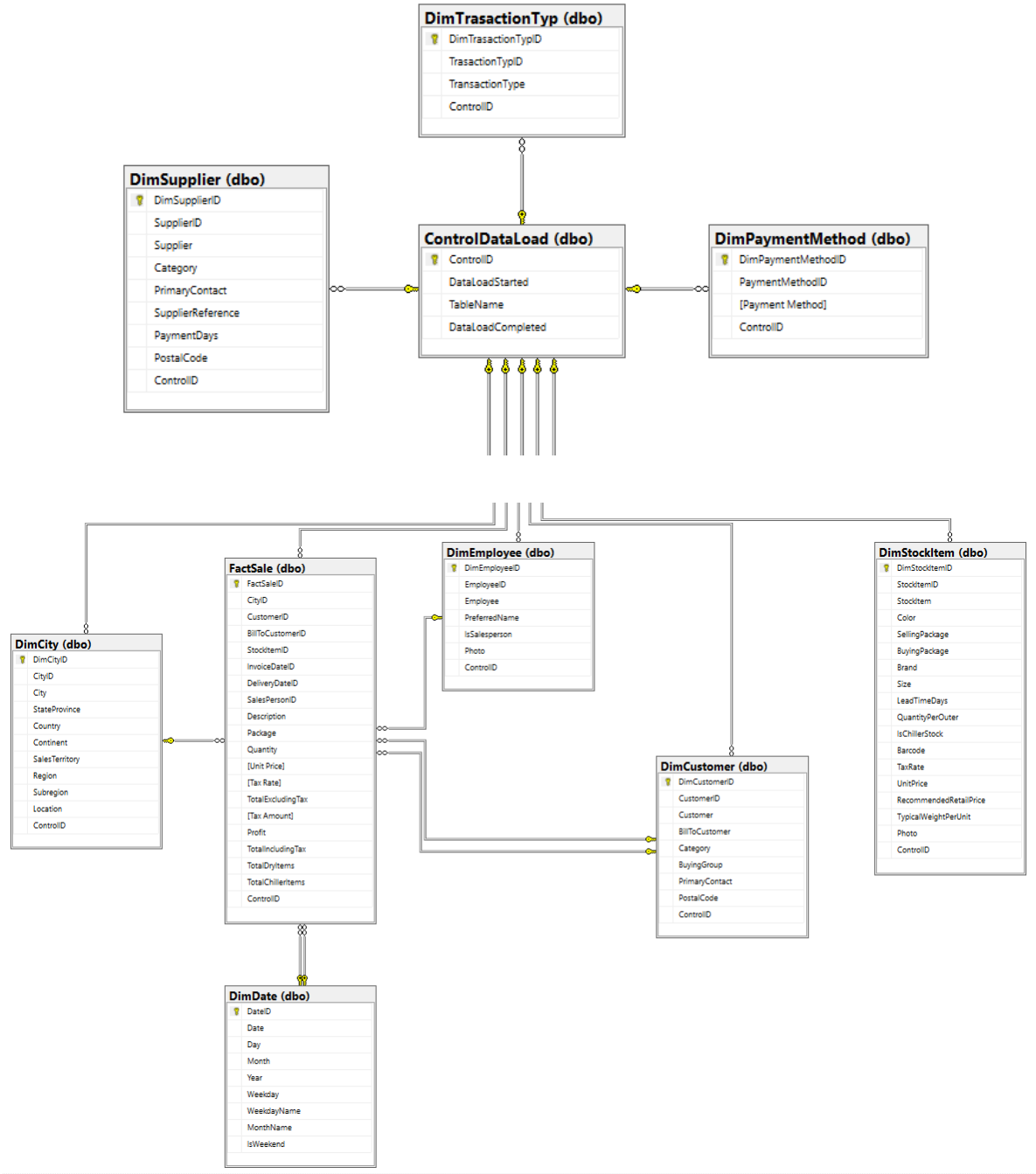
```
    DECLARE @ControlIDs table (id INT);
    DECLARE @ControlID INT;
    --
    INSERT INTO [dbo].[ControlDataLoad] (DATALOADSTARTED, TABLENAME)
    OUTPUT INSERTED.ControlID INTO @ControlIDs
    SELECT GETDATE(), 'DimSupplier';

    SET @ControlID  = (SELECT TOP 1 ID FROM @ControlIDs);
    --
    UPDATE dc
    SET dc.ControlID = @ControlID
    FROM [dbo].[DimSupplier] dc
    INNER JOIN INSERTED i ON dc.SupplierID = i.SupplierID;
END
GO
/*************************************************************************
*************************/
--DIMENSIONS DimTrasactionTyp
IF OBJECT_ID('dbo.DimTrasactionTyp', 'U') IS NOT NULL
BEGIN
    -- Si existe, eliminar la tabla
    DROP TABLE dbo.DimTrasactionTyp;
END

CREATE TABLE [dbo].[DimTrasactionTyp] (

    [DimTrasactionTypID] [INT]          NOT NULL IDENTITY(1,1),
    [TrasactionTypID]    [INT]          NOT NULL,
    [TransactionType]    [nvarchar](255) NOT NULL,
    [ControlID]          [INT]          NOT NULL,
    CONSTRAINT [PK_Dimension_TrasactionTyp] PRIMARY KEY CLUSTERED
([DimTrasactionTypID] ASC)
);
ALTER TABLE [dbo].[DimTrasactionTyp] WITH CHECK ADD CONSTRAINT
[FK_DimTrasactionTyp_ControlDataLoad] FOREIGN KEY([ControlID])
REFERENCES [dbo].[ControlDataLoad] ([ControlID]);
--TRIGGER
CREATE OR ALTER TRIGGER [dbo].[TRG_CONTROL_TRANSACTIONTYP] ON
[dbo].[DimTrasactionTyp]
AFTER INSERT
AS
BEGIN

    DECLARE @ControlIDs table (id INT);
    DECLARE @ControlID INT;
```

```
    --
    INSERT INTO [dbo].[ControlDataLoad] (DATALOADSTARTED, TABLENAME)
    OUTPUT INSERTED.ControlID INTO @ControlIDs
    SELECT GETDATE(), 'DimTrasactionTyp';

    SET @ControlID  = (SELECT TOP 1 ID FROM @ControlIDs);
    --
    UPDATE dc
    SET dc.ControlID = @ControlID
    FROM [dbo].[DimTrasactionTyp] dc
    INNER JOIN INSERTED i ON dc.TrasactionTypID = i.TrasactionTypID;
END
GO
/*************************************************************************
************************/
/*************************************************************************
************************/

--tablas de hecho
--DIMENSIONS FactSale
IF OBJECT_ID('dbo.FactSale', 'U') IS NOT NULL
BEGIN
    -- Si existe, eliminar la tabla
    DROP TABLE dbo.FactSale;
END

CREATE TABLE [dbo].[FactSale] (

    [FactSaleID]        [BIGINT] NOT NULL IDENTITY(1,1),
    [CityID]            [int]    NOT NULL,
    [CustomerID]        [int]    NOT NULL,
    [BillToCustomerID]  [int]    NOT NULL,
    [StockItemID]       [int]    NOT NULL,
    [InvoiceDateID]     [int]    NOT NULL,
    [DeliveryDateID]    [int]    NOT NULL,
    [SalesPersonID]     [int]    NOT NULL,
    [Description]       [nvarchar](100) NOT NULL,
    [Package]           [nvarchar](50) NOT NULL,
    [Quantity]          [int] NOT NULL,
    [Unit Price]        [decimal](18, 2) NOT NULL,
    [Tax Rate]          [decimal](18, 3) NOT NULL,
    [TotalExcludingTax] [decimal](18, 2) NOT NULL,
    [Tax Amount]        [decimal](18, 2) NOT NULL,
    [Profit]            [decimal](18, 2) NOT NULL,
    [TotalIncludingTax] [decimal](18, 2) NOT NULL,
```

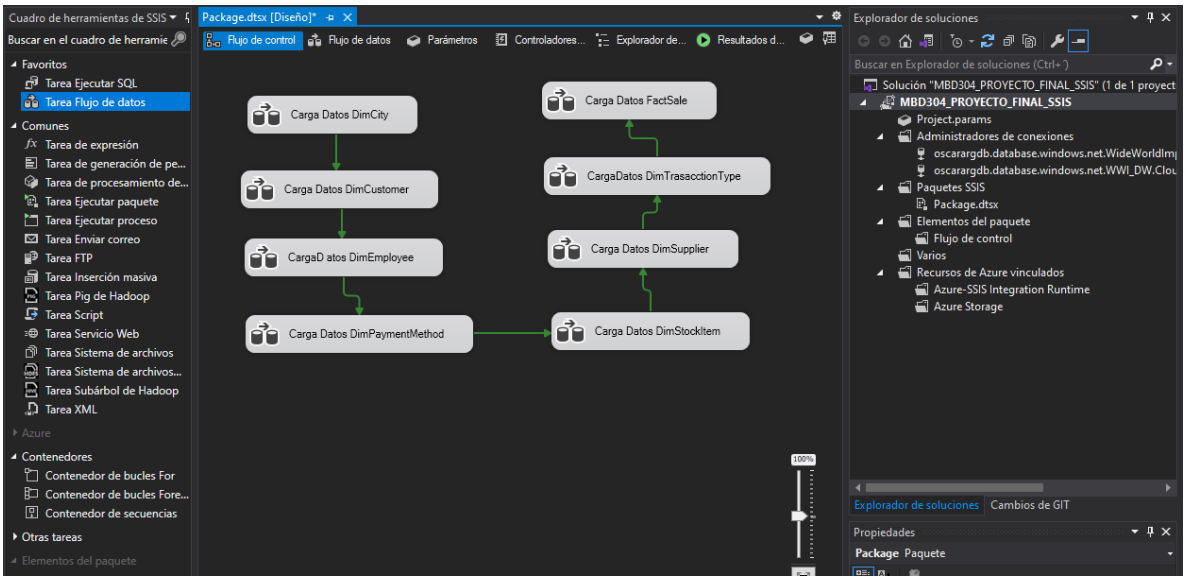
```
    [TotalDryItems]     [int] NOT NULL,
    [TotalChillerItems] [int] NOT NULL,
    [ControlID]        [INT]          NOT NULL,
    CONSTRAINT [PK_Fact_Sale] PRIMARY KEY CLUSTERED ([FactSaleID] ASC)
);
--
ALTER TABLE [dbo].[FactSale] WITH CHECK ADD CONSTRAINT
[FK_FactSale_ControlDataLoad] FOREIGN KEY([ControlID])
REFERENCES [dbo].[ControlDataLoad] ([ControlID]);
--
ALTER TABLE [dbo].[FactSale] WITH CHECK ADD CONSTRAINT
[FK_FactSale_DimCustomer_CDimCity] FOREIGN KEY([CityID])
REFERENCES [dbo].[DimCity] ([DimCityID]);
--
ALTER TABLE [dbo].[FactSale] WITH CHECK ADD CONSTRAINT
[FK_FactSale_DimCustomer_CustomerID] FOREIGN KEY([CustomerID])
REFERENCES [dbo].[DimCustomer] ([DimCustomerID]);
--
ALTER TABLE [dbo].[FactSale] WITH CHECK ADD CONSTRAINT
[FK_FactSale_DimCustomer_BillToCustomer] FOREIGN KEY([BillToCustomerID])
REFERENCES [dbo].[DimCustomer] ([DimCustomerID]);
--
ALTER TABLE [dbo].[FactSale] WITH CHECK ADD CONSTRAINT
[FK_FactSale_DimDate_InvoiceDateID] FOREIGN KEY([InvoiceDateID])
REFERENCES [dbo].[DimDate] ([DateID]);
--
ALTER TABLE [dbo].[FactSale] WITH CHECK ADD CONSTRAINT
[FK_FactSale_DimDate_DeliveryDateID] FOREIGN KEY([DeliveryDateID])
REFERENCES [dbo].[DimDate] ([DateID]);
--
ALTER TABLE [dbo].[FactSale] WITH CHECK ADD CONSTRAINT
[FK_FactSale_DimDate_DimEmployee] FOREIGN KEY([SalesPersonID])
REFERENCES [dbo].[DimEmployee] ([DimEmployeeID]);


--TRIGGER
CREATE OR ALTER TRIGGER [dbo].[TRG_CONTROL_FactSale] ON [dbo].[FactSale]
AFTER INSERT
AS
BEGIN

    DECLARE @ControlIDs table (id INT);
    DECLARE @ControlID INT;
    --
    INSERT INTO [dbo].[ControlDataLoad] (DATALOADSTARTED, TABLENAME)
```

```
    OUTPUT INSERTED.ControlID INTO @ControlIDs
    SELECT GETDATE(), 'FactSale';

    SET @ControlID  = (SELECT TOP 1 ID FROM @ControlIDs);
    --
    UPDATE x
    SET x.ControlID = @ControlID
    FROM [dbo].[FactSale] x
    INNER JOIN INSERTED i ON x.CityID = i.CityID AND
                             x.CustomerID = i.CustomerID AND
                             x.BillToCustomerID = i.BillToCustomerID AND
                             x.StockItemID = i.StockItemID AND
                             x.InvoiceDateID = i.InvoiceDateID AND
                             x.DeliveryDateID = i.DeliveryDateID AND
                             x.SalesPersonID = i.SalesPersonID;
END
GO
```

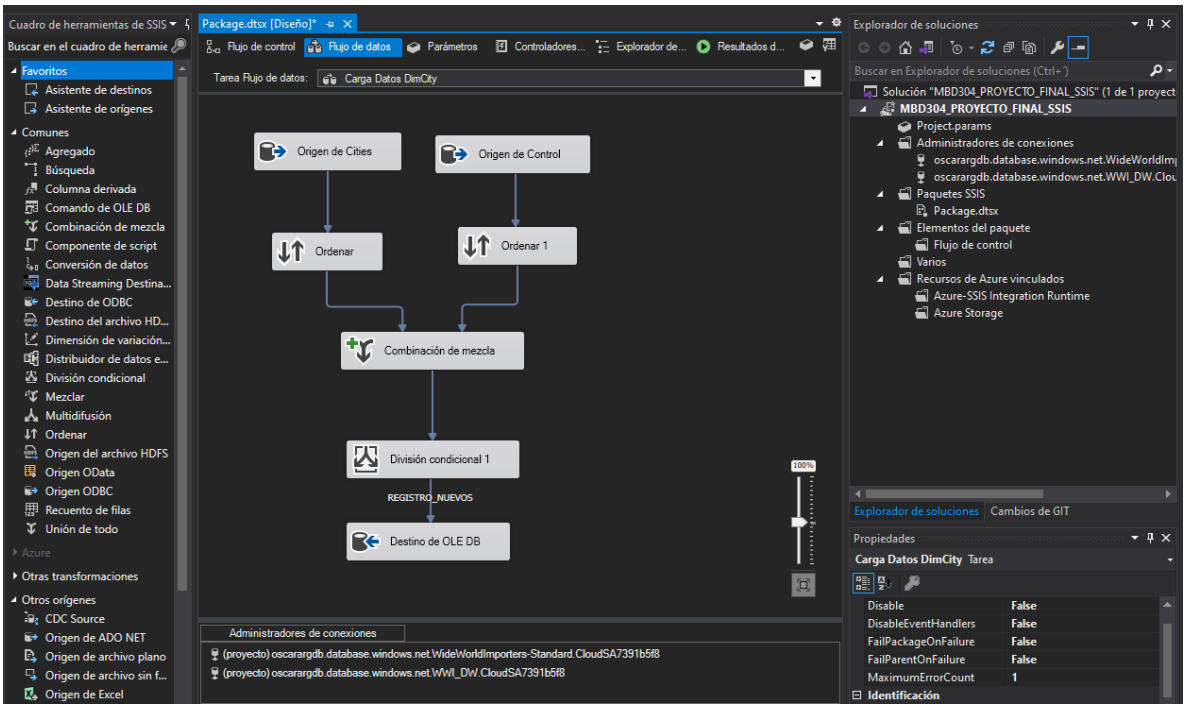
Diagrama Base de Datos



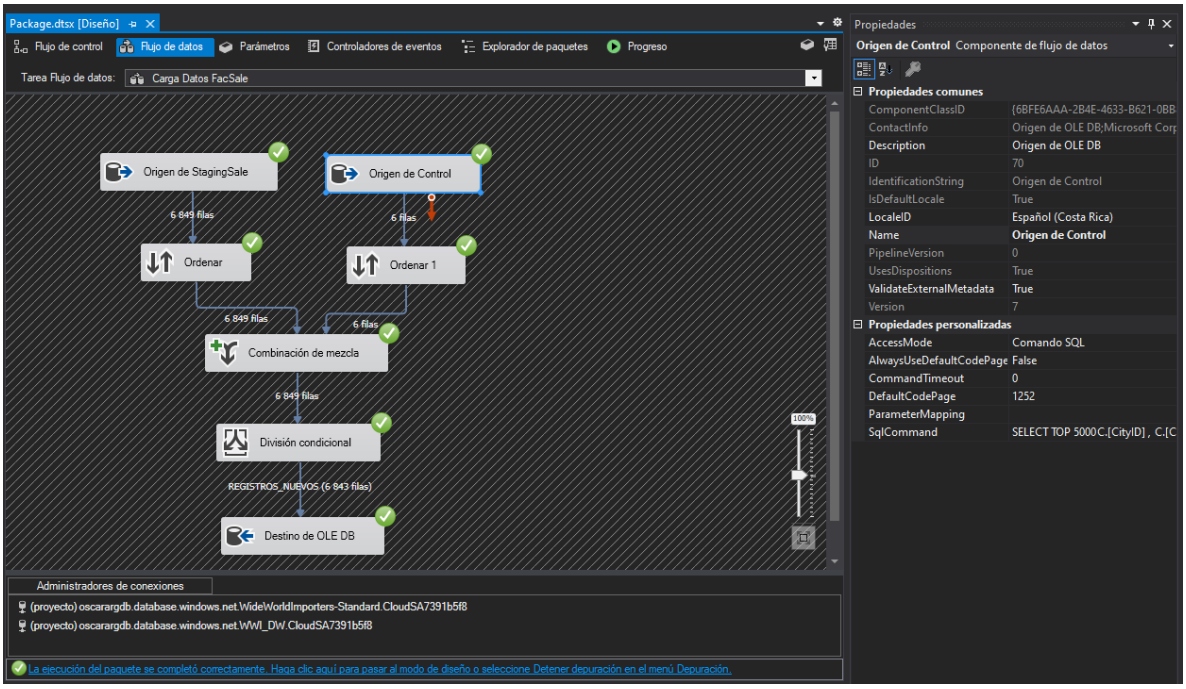
Creación del SSIS

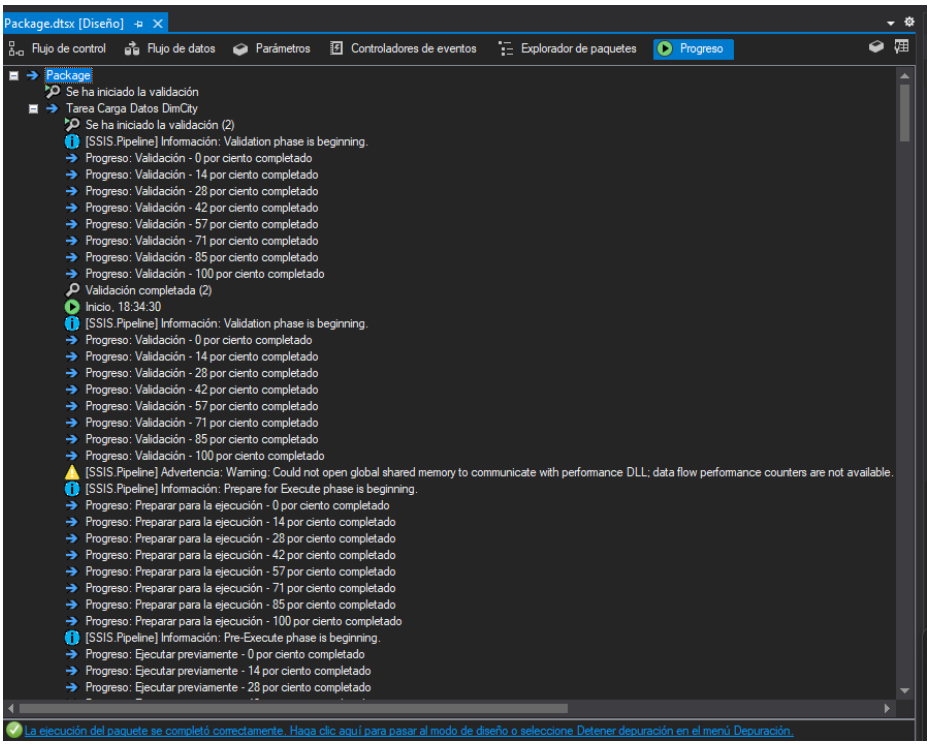


En cada Flujo de carga de datos se realiza una validación de los datos que ya están cargados, y solo se van a cargar aquellos que no existan en las dimensiones, se utilizo 2 origen de datos, uno para cargar la data que se a cargar a la dimensión, y en el otro se carga información de lo que ya se encuentra cargado en el sistema, seguido se realiza un ordenamiento de los datos de cada consulta y hace un combinación de mezcla del tipo INNER JOIN para excluir aquellos datos que no han sido cargados aun.

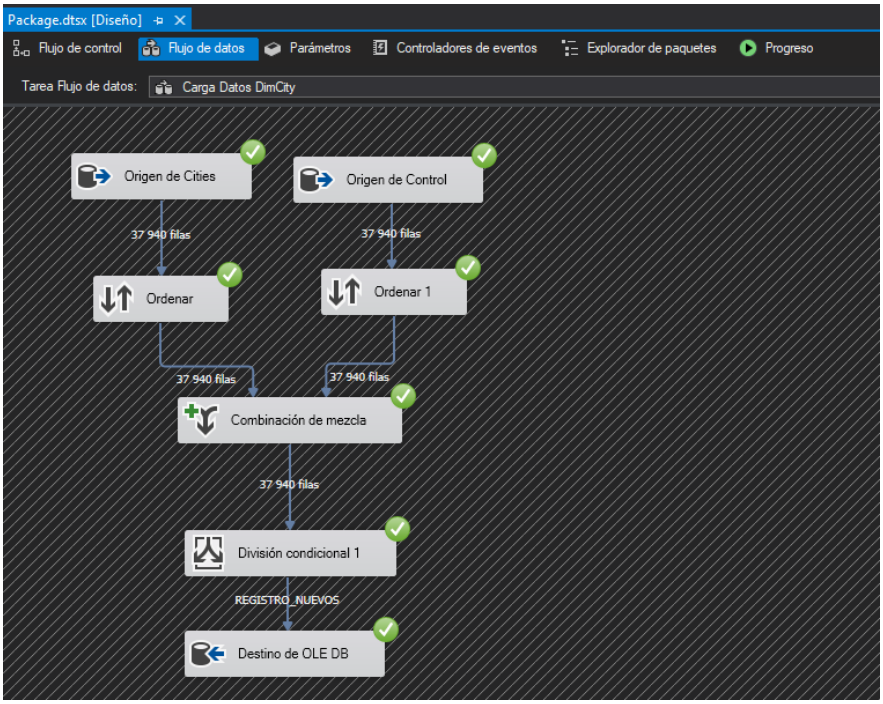


Proceso de Carga





En caso de volver a ejecutar el proceso de carga se validan que fue cargado y que no, y solo se va a cargar los registros nuevos

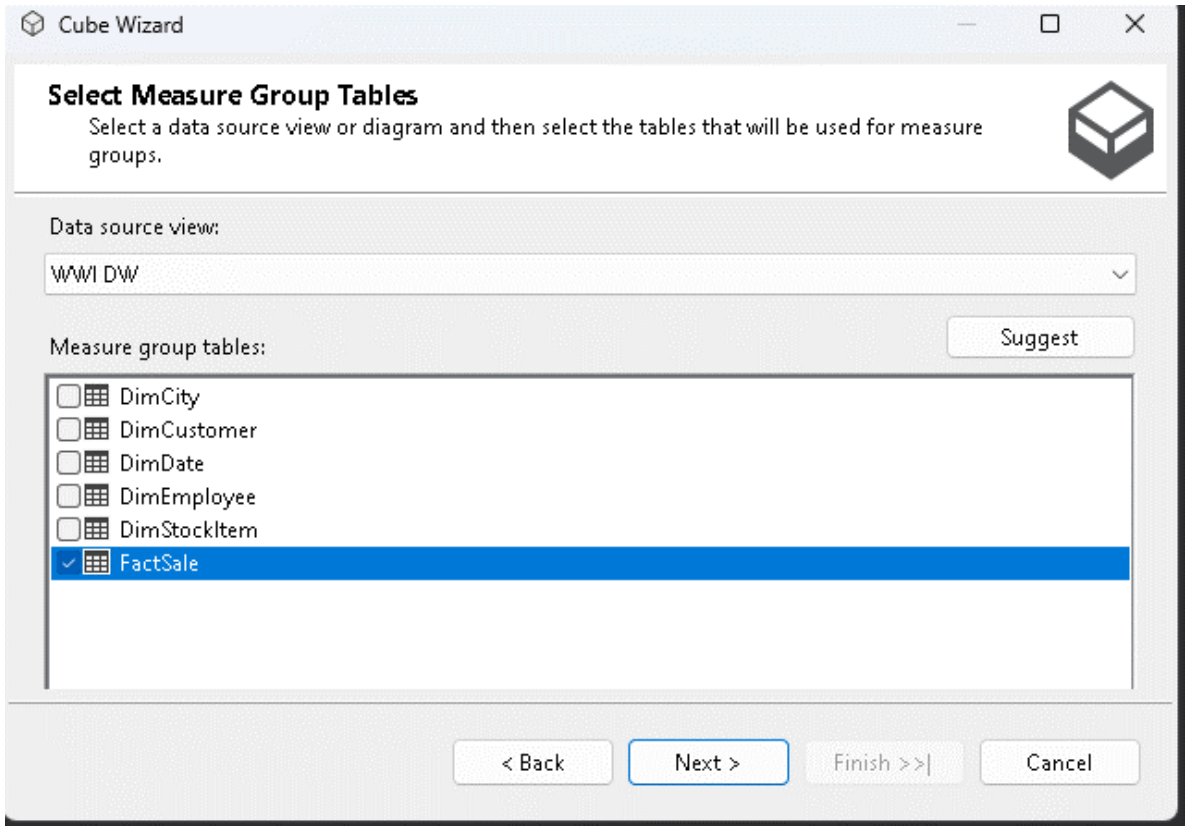


Creación del Cubo

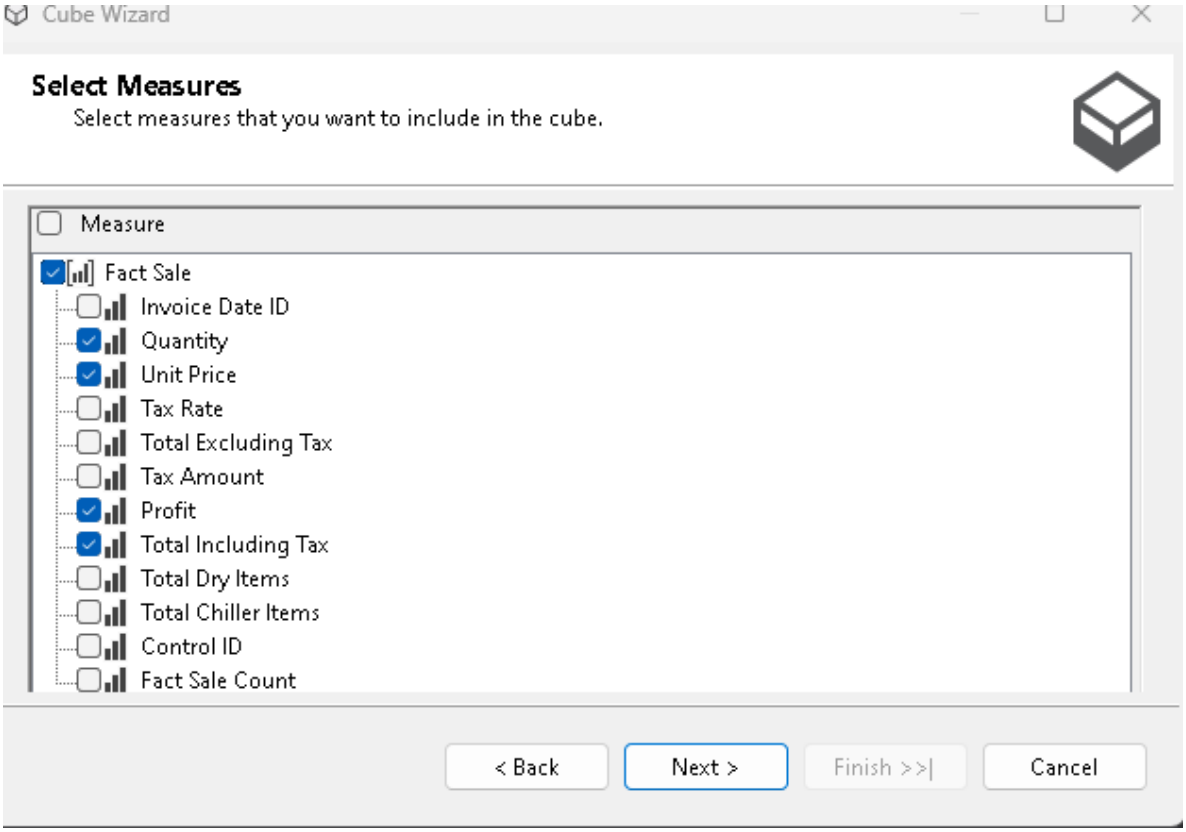
Para la creación del cubo hemos incorporado un proyecto de SSAS al ya existente. A continuación, se muestran los pasos para la creación de este cubo:

Una vez ajustamos la fuente de datos y su vista procedemos con el cubo

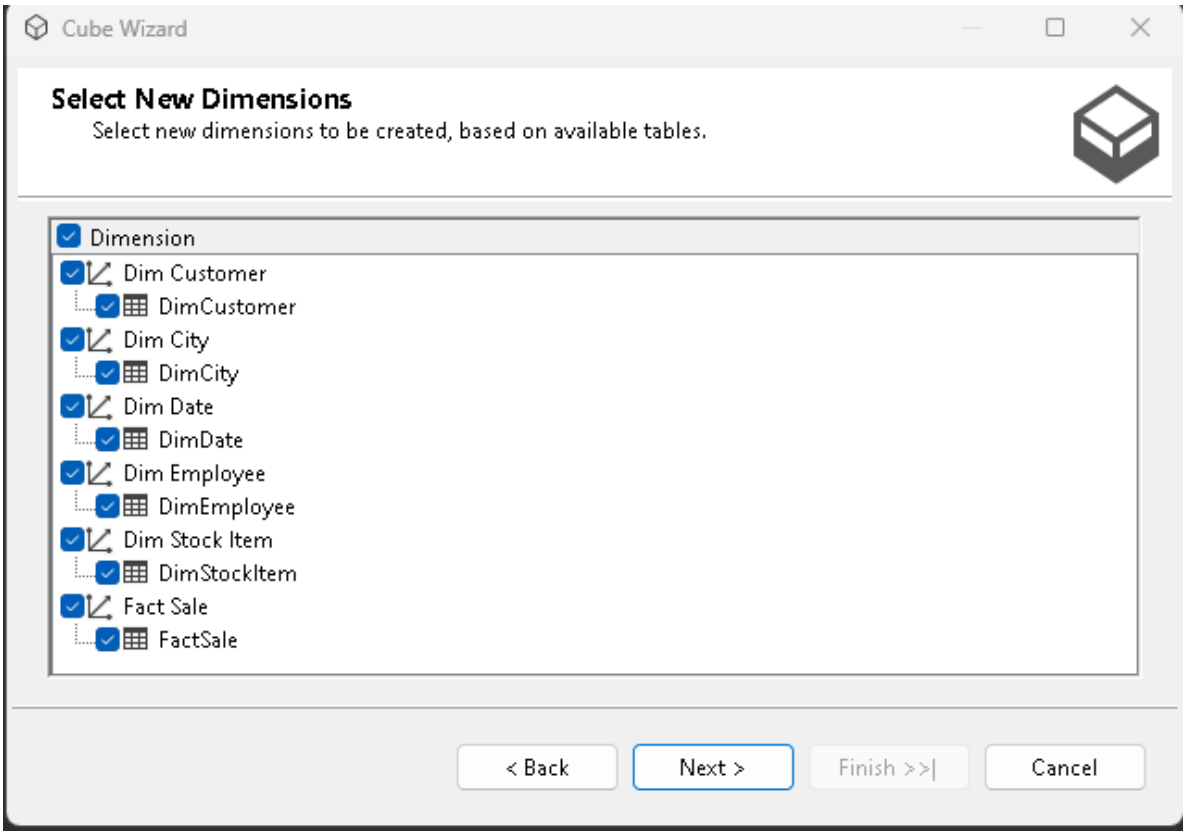
Como medida utiliza su tabla de hechos:



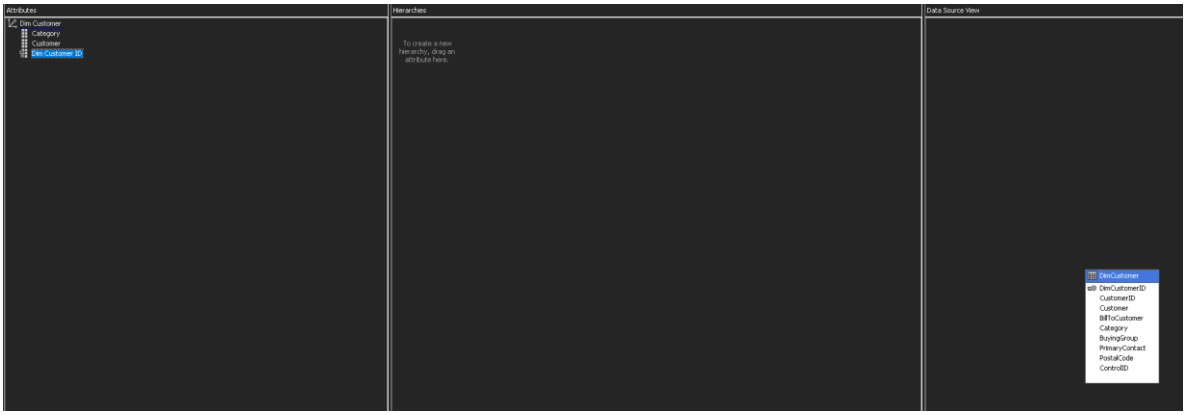
Con sus respectivas medidas:



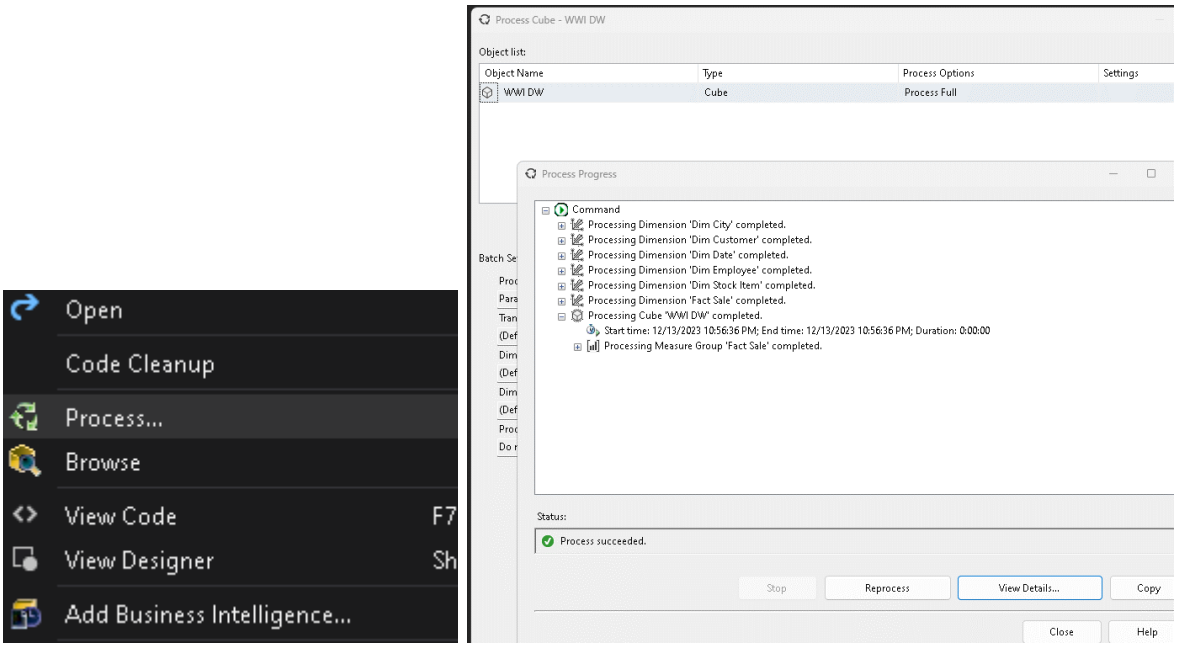
Y todas las dimensiones asociadas:



En cada una de las dimensiones vamos a cargar los atributos:



Una vez hecho el proceso en todas las tablas estamos listos para procesar el cubo:



Ahora podemos utilizar el cubo para analisis de datos, en el caso de esta base de datos se centra principalmente en Ventas

Repositorio de Codigo

[OscarArguedasB/MBD304-FranciscoBenavides-OscarArguedas (github.com)]

Conclusión

La culminación de este proyecto de Data Warehouse, basado en la base de datos transaccional de World Wide Importers, representa un avance significativo con respecto a inicios de curso. Hemos logrado no solo consolidar información dispersa en una única fuente coherente y de alta calidad, sino también hemos establecido un sistema que es escalable y flexible, adaptándose a las necesidades cambiantes del negocio.

El desarrollo e implementación de procesos ETL con SSIS han sido elementos cruciales de este proyecto, optimizando la extracción, transformación y carga de datos. La eficiencia y automatización proporcionadas por SSIS han permitido una gestión de datos más fluida y controlada, una base para futuras expansiones y adaptaciones.

Más allá de la implementación técnica, este proyecto subraya la importancia de considerar la gestión de datos como un proceso continuo y evolutivo. La infraestructura establecida requiere un monitoreo constante, mantenimiento regular y actualizaciones de seguridad para garantizar su relevancia y eficacia a largo plazo. Además, la capacitación y el desarrollo continuos del personal son esenciales para maximizar las capacidades del sistema y asegurar que la organización aproveche al máximo su inversión en tecnología.

La integración de un Data Warehouse robusto con herramientas de análisis e inteligencia empresarial coloca a cualquier organización en una posición ventajosa. También hemos enriquecido nuestra habilidad para convertir los datos utilizados en conocimientos significativos y decisiones informadas.