Implement a Data Warehouse with SQL Server

Design and Implement Dimensions and Fact Tables

[SSMS](https://docs.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver15) - SQL Server Management Studio

# Course Modules

Implementing a Data Warehouse with SQL Server

1. **Design and Implement Dimensions and Fact Tables (Completed)**
2. Data Flow - Extract Data
3. Data Flow - Transform Data
4. Control Flow
5. Configure and Deploy SSIS
6. Manage Enterprise Data

Setting Expectations

1. Target Audience

* Data warehousing specialists who want to expand their knowledge of SQL Server Integration Services (SSIS)
* Database professionals who want to take exam 70-463 and get certified in data warehousing implementations

1. Suggested Prerequsites/Supporting Material
   1. SQL Server development experience and exposure to extract, transform, and load (ETL) processes
   2. Course 10777. Implementing a Data Warehouse with Microsoft SQL Server 2012
   3. MS Press Book: Training Kit (Exam 70-463): Implementing a Data Warehouse with Microsoft SQL Server 2012

## 

## 

## 

## 

## **Module 1: Design and Implement Dimensions and Fact Tables**

Overview

1. Schema Design: Star vs Snowflake
2. Facts and Fact Tables
3. Fact and Dimensions Granularity
4. Confirmed and Non-Conformed Dimensions
5. Time Dimensions

Topic: Schema Design: Star versus Snowflake

* Star Schema
* Snowflake Schema
* Processing and Performance Considerations

## 

## 

## 

## 

## 

## 

## 

## 

## 

## Star Schema



* A star schema has a single for each dimension
* Each table supports all attributes for that dimension
* Typically a denormalized solution - Having similar data such as state under one attribute, for example if multiple customers buy a product product from your retail store then

## 

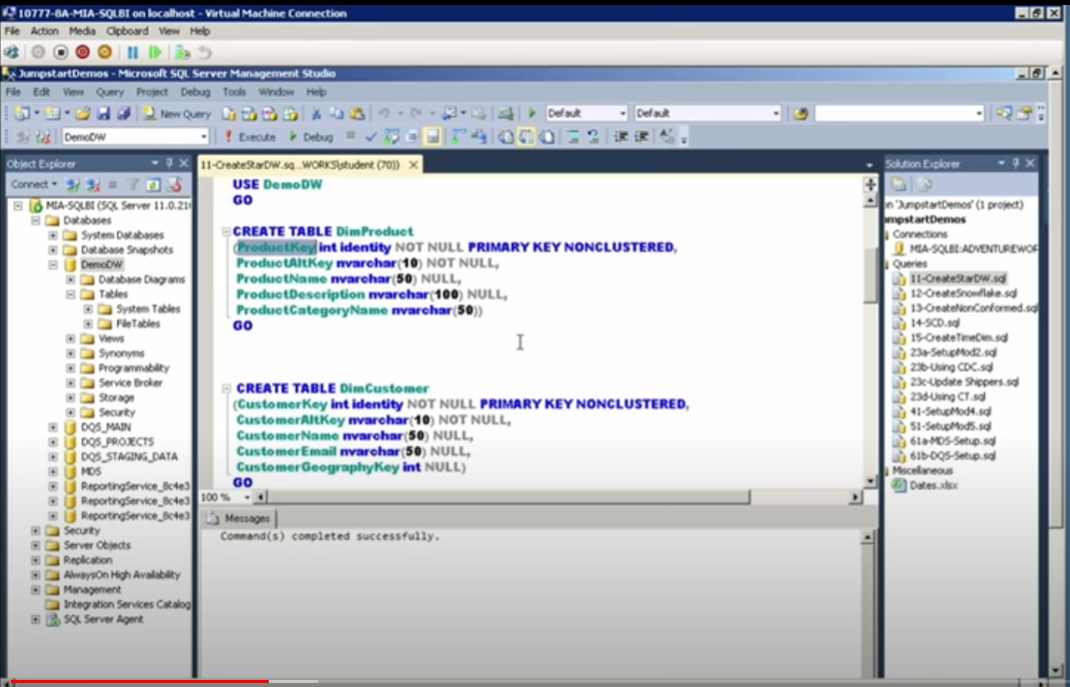
## 

## 

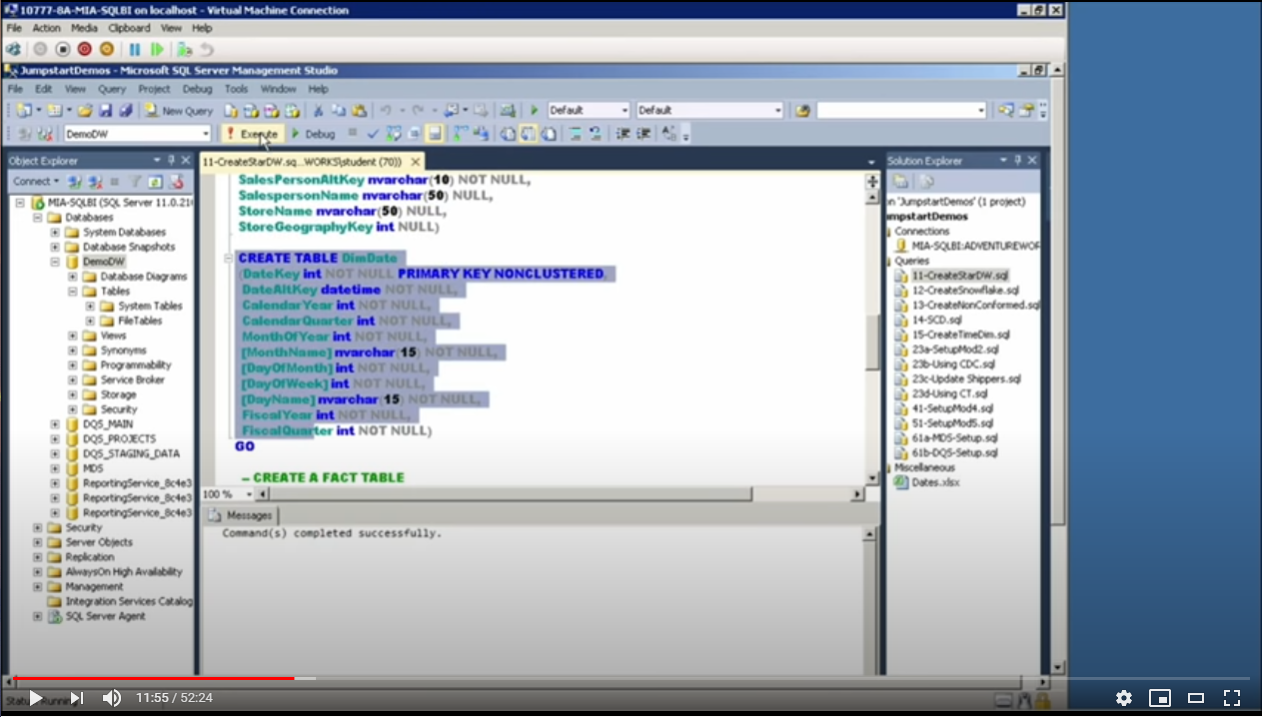
## **Demo**

**Build the Product and Customer Table**

ProductAltKey is the business key



**Build the Date Table**



## **Build the Fact Table**

Using the REFERENCE statement is important since it will relate to the other dimensions and support the attributions.

## Star Schema Visualization

## **Demo (Part 2)**

**SQL Database Tables (SQL Code)**

USE DW\_Demo

GO

CREATE TABLE DimProduct

(ProductKey int Identity NOT NULL PRIMARY KEY NONCLUSTERED,

ProductAltkey nvarchar(10) NOT NULL,

ProductName nvarchar(50) NULL,

ProductDescription nvarchar(100) NULL,

ProductCategoryName nvarchar(50))

GO

CREATE TABLE DimCustomer

(CustomerKey int Identity NOT NULL PRIMARYR KEY NONCLUSTERED,

CustomerAltKey nvarchar(10) NOT NULL,

CustomerName nvarchar(50) NULL,

CustomerEmail nvarchar(50) NULL,

CustomerGeographyKey int NULL)

GO

CREATE TABLE DimSalesPerson

(SalesPersonKey int Identity NOT NULL PRIMARY KEY NONCLUSTERED,

SalesPersonAltKey nvarchar(10) NOT NULL,

SalesPersonName nvarchar(50) NULL,

StoreName nvarchar(50) NULL,

StoreGeographyKey int NULL)

GO

CREATE TABLE DimDate

(DateKey int NOT NULL PRIMARY KEY NONCLUSTERED,

DateAltKey datetime NOT NULL,

CalenderYear int NOT NULL,

CalenderQuarter int NOT NULL,

MonthOfYear int NOT NULL,

[MonthName] nvarchar(15) NOT NULL,

[DayOfMonth] int NOT NULL,

[DayOfWeek] int NOT NULL,

[DayName] nvarchar(15) NOT NULL,

FiscalYear int NOT NULL,

FiscalQuarter int NOT NULL)

GO

/\*Create A Fact Table\*/

CREATE TABLE FactSalesOrders

(ProductKey int NOT NULL REFERENCES DimProduct(ProductKey),

CustomerKey int NOT NULL REFERENCES DimCustomer(CustomerKey),

SalespersonKey int NOT NULL REFERENCES DimSalesPerson(SalesPersonKey),

OrderDateKey int NOT NULL REFERENCES DimDate(DateKey),

OrderNo int NOT NULL,

ItemNo int NOT NULL,

Quantity int NOT NULL,

SalesAmount money NOT NULL,

Cost money NOT NULL

CONSTRAINT[PK\_ FactSalesOrder] PRIMARY KEY NONCLUSTERED

(

[ProductKey],[CustomerKey],[SalesPersonKey],[OrderDateKey],[OrderNo],[ItemNo]

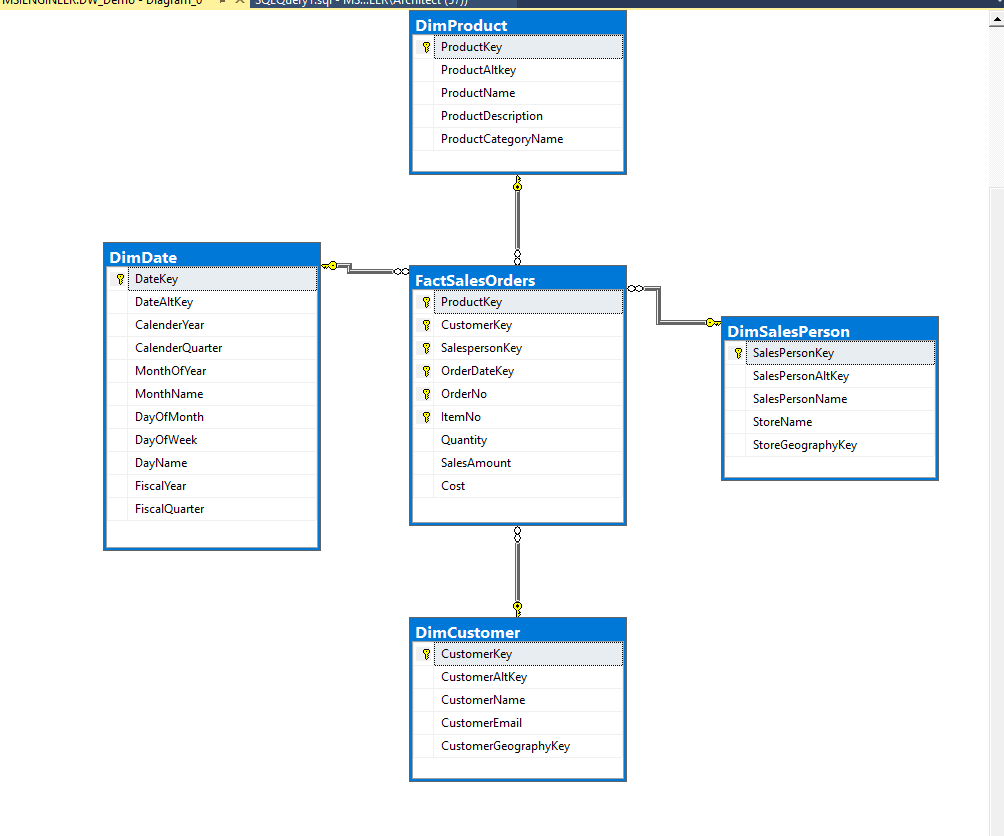
)

)

**SQL Database Tables (Diagram)**

**Columns View**

## 



**Standard View**

## 

## 

## 

## 

## 

## Snowflake Schema

* More normalized solution
* Typically contains multiple tables per dimension
* Each table contains dimension key, value, and the foreign key value for the parent



**Link a Geography Dimension to the Customer and Sales Dimension**

Create a Geography Dimension

USE DW\_Demo

GO

CREATE TABLE DimGeography

(GeographyKey int identity NOT NULL PRIMARY KEY NONCLUSTERED,

PostalCode nvarchar(15) NULL,

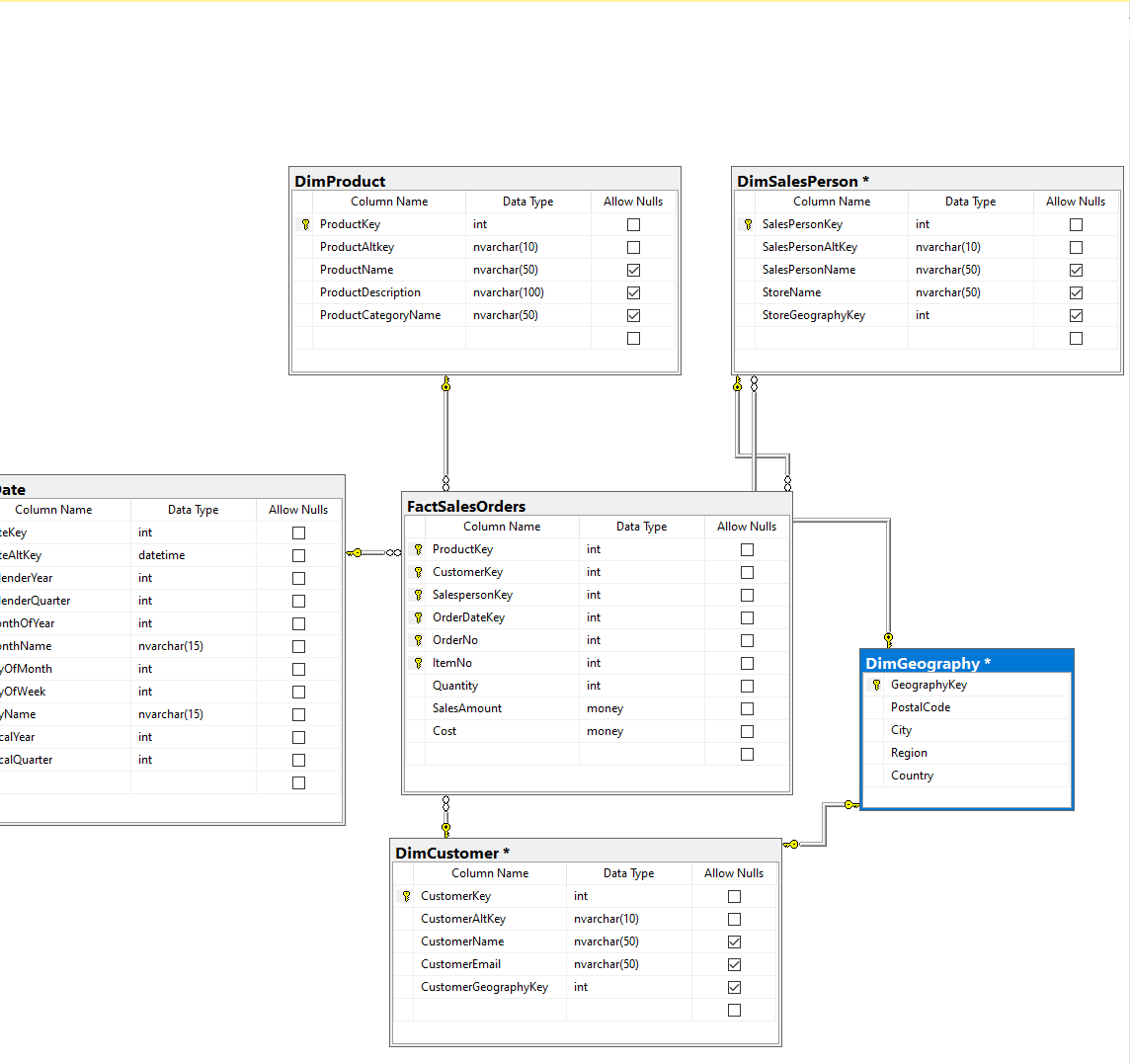
City nvarchar(50) NULL,

Region nvarchar(50) NULL,

Country nvarchar(50) NULL)

GO

**Visual**



**Processing and Performance Considerations**

* Star schema requires de-normalization during the load process
  + Can impact ETL times
* Snowflake schema can increase dimension complexity
  + Can impact Analysis Services solutions, negatively affecting cube performance
  + Querying could take longer hence dimension and table designing is very important

**Facts and Fact Tables**

* What is a Fact
* Grouping Facts
* What is granularity?
* Design Considerations

What is a Fact?

* Facts are the key metrics used to measure business
  + Sales
  + Production
  + Inventory
* Can be additive, semi-additive, or non-additive
  + Additive - Aggregate among all dimensions
  + Semi-additive - A measure is aggregatable along some dimensions but not all
  + Non-additive - Any measure that you need to calculate along a dimension at each level

Grouping Facts

* Facts are grouped into fact tables
  + Group fact tables with the same dimension relationship
* Related facts should be in the same fact table
  + It makes sense to make different fact tables for business means
* Facts with different granularity should be in different tables

What is Granularity?

* Granularity refers to the level of detail in which facts are recorded
* Facts can be at different levels of granularity

Design Considerations

* Fact tables should have all the keys relating to dimensions
* Primary key should be composite of all dimension keys
* Separate additive, semi-additive, and non-additive facts

How to Determine Fact Granularity

* Granularity is determined based on business needs
* Should be the lowest level of detail that needs to be examined
  + Should not load data to the data warehouse more than it is needed. The Fact table is going to bne so big anyways why would you need to load more than you need
* If data from transactional systems has more detail than needed for analysis, ETL should aggregate the details

How to Determine Dimension Granularity

* Dimension granularity needs to be matched with the
* Each dimension has its own granularity
* Fact tables are keyed to the granularity of the dimensions

### **Topic: Conformed and Non-Conformed Dimensions**

* What Are Conformed and Non-Conformed Dimensions?
* Shared and Degenerate Dimensions
* What Is a Slowly Changing Dimension?

**What Are Confirmed and Non-Conformed Dimensions?**

1. **Confirmed Dimensions**

* Shared by multiple fact tables
* Used when all business users have the same definitions for the dimension
  + Single categorization that all aspects of the business uses
    - One definition for a customer dimension
    - One definition for a sales dimension
    - One definition for what a financial dimension

1. **Non-Conformed Dimensions**
   * Dimension table targeted to only one fact table
   * Used when dimensions have different definitions for different business units - Uses some or different set attributes in the dimension table (examples):
     1. SalesProduct Dimension
     2. Manufacture Product Dimension

**Shared and Degenerate Dimensions**

* **Shared Dimension**
  + Used by multiple facts
  + Dimension key is stored in the fact table
  + Dimension value is stored in the dimension table with other attributes of that dimension
* **Degenerate dimension**
  + Used by a single fact table
  + Dimension value is stored directly in the fact table
  + No corresponding dimension table

**What Is a Slowly Changing Dimension?**

* When the historical attribute values are retained if the attributes are updated
* Used when the organization does not want to lose track of what actually happened
  + Example: customer moves from Connecticut to Seattle
* Slowly Changing dimensions types:
  + Type 1: Attribute history is not retained
  + Type 2: Attribute change creates a new record
  + Type 3: Original attribute value recorded and latest value recorded with an effective date

**Types of Time Dimensions**

* Based on standard calendar breakdowns
  + Year => Month => Day
  + Year => Quarter => Week => Day
* Based on fiscal calendar
  + Year => Fiscal Quarter =>Fiscal Month => Fiscal Week => Day
* Time dimension needs to contain all hierarchy elements to the lowest granularity for the fact tables
* A generated table that drives the qualification of all that information.