Lab Week 4

Designing a new Data Warehouse

In the last week's lab, we learn how to configure a sample data warehouse and understand about the designing of AdventureWorksDW. Finally, we have visualized the some of the table contents using PowerBI. In this Lab, we will (1) go through the process of designing a new data warehouse using SSMS, (2) build a cube for drill-down and roll up analysis in SSDT, and (3) visualize the data using Power BI. The detailed descriptions of the tasks are given below:

Task 1: Design a data warehouse, create fact table(s) and dimension tables, and populate the tables with data.

Task 2: Build a multi-dimensional analysis service solution using SSDT. The key steps include:

- Connect to a data source
- Create the data source view
- Build a cube
- Generate and fine tune the dimension hierarchies

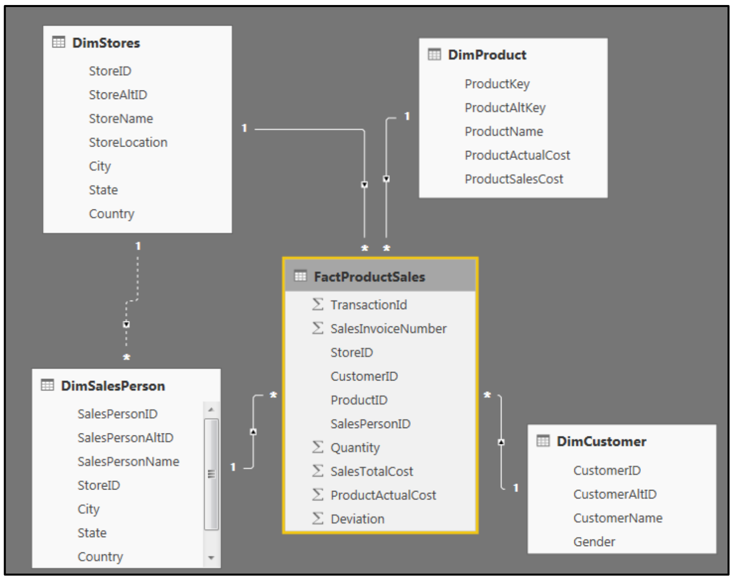
Task 3: Connect to the solution through Power BI for visualization

Note: Please check whether your environment setup is completed already. The SQL Server instance in the local machine/VM should be in running mode (refer Week3 lab). In this lab, we need SQL Server Analysis Service (SSAS) instance (part of SSDT tool). Analysis Services will be used for deployment of multi-dimensional cubes. For the installation process of SSAS, please refer Week2 lab.

Business Requirement

Woolworths have different shopping centers in Perth, where daily sales take place for various products. Higher management is facing an issue while decision making due to non-availability of integrated data they can't do study on their data as per their requirement. So they need to design a system which can help them quickly in decision making and provide Return on Investment (ROI).

Let us start designing of the data warehouse, we need to follow a few steps before we start our data warehouse design. First, try to identify the dimension table, fact tables and the primary key, foreign key relationships. One of the possible designs is shown in the following figure. You need to identify the Fact table and the dimension tables with their relationships from the below diagram.



Task 1.1: Data Warehouse Creation using SQL Scripting

In this task, we will follow the below instructions to create the Fact and Dimension tables:

a. Create a new database **NewSalesDW**
b. Create the four dimension tables **DimCustomer, DimProduct, DimStores, DimSalesPerson** with respective primary keys (CustomerId, ProductKey, StoreId, SalesPersonId). At the same time, populate the dimension tables with sample data.
c. Create the Fact table **FactProductSales** (Primary Key - TransactionId)
d. Add relation between fact table foreign keys to Primary keys of Dimension Tables (i.e. Foreign key constraints)
e. Populate the Fact table with sample data.

The bellow steps, in general, should be followed to create the database and tables as mentioned above.

1. Open SQL Server Management Studio (SSMS) and connect to Database Engine.
2. Open "New Query" editor.
3. Copy and paste scripts given below in new query editor window one by one. These scripts are following the instructions mentioned above.
4. To run the given SQL script, click "Execute" or press **F5** key.

Step 1

Run the following script to create a database '**NewSalesDW**' and then use the database.

```
PRINT '';
PRINT '*** Dropping Database';
GO

IF EXISTS (SELECT [name] FROM [master].[sys].[databases] WHERE [name] = N'NewSalesDW')
DROP DATABASE NewSalesDW;
GO

PRINT '';
PRINT '*** Creating Database';
GO
```

```
Create database NewSalesDW
Go

Use NewSalesDW
Go
```

Step 2

Run the following script to create **Customer dimension** table in Data Warehouse which will hold the customers' personal details.

```
PRINT '';
PRINT '*** Creating Table DimCustomer';
GO

Create table DimCustomer
(
CustomerID int primary key identity,
CustomerAltID varchar(10) not null,
CustomerName varchar(50),
Gender varchar(20)
)
Go
```

Run the following script to fill the DimCustomer table with sample Values

```
Insert into DimCustomer (CustomerAltID,CustomerName,Gender)
values
('IMI-001','Henry Ford','M'),
('IMI-002','Bill Gates','M'),
('IMI-003','Muskan Shaikh','F'),
('IMI-004','Richard Thrubin','M'),
('IMI-005','Emma Wattson','F');
Go
```

Step 3: Run the following script to create basic level of **Product Dimension** table without considering any Category or Subcategory

```
PRINT '';
PRINT '*** Creating Table DimProduct';
GO

Create table DimProduct
(
ProductKey int primary key identity,
ProductAltKey varchar(10)not null,
ProductName varchar(100),
ProductActualCost money,
ProductSalesCost money
)
Go
```

Run the following script to insert some sample values in DimProduct table.

```
Insert into DimProduct (ProductAltKey, ProductName, ProductActualCost, ProductSalesCost)
values
('ITM-001','Wheat Floor 1kg',5.50,6.50),
('ITM-002','Rice Grains 1kg',22.50,24),
('ITM-003','SunFlower Oil 1 ltr',42,43.5),
('ITM-004','Nirma Soap',18,20),
('ITM-005','Arial Washing Powder 1kg',135,139);
Go
```

Step 4

Run the following script to create **Store Dimension** table which will hold details related stores available across various places.

```
PRINT '';
PRINT '*** Creating Table DimStores';
GO

Create table DimStores
(
StoreID int primary key identity,
StoreAltID varchar(10)not null,
StoreName varchar(100),
StoreLocation varchar(100),
City varchar(100),
State varchar(100),
Country varchar(100)
)
Go
```

Run the following script to insert some sample values in table DimStores.

```
Insert into DimStores (StoreAltID, StoreName, StoreLocation, City, State, Country) values
('LOC-A1','X-Mart','S.P. RingRoad','Ahmedabad','Guj','India'),
('LOC-A2','X-Mart','Maninagar','Ahmedabad','Guj','India'),
('LOC-A3','X-Mart','Sivranjani','Ahmedabad','Guj','India');
Go
```

Step 5

Run the following script to create **Dimension Sales Person** table which will hold details related stores available across various places and add some sample data into the table

```
PRINT '';
PRINT '*** Creating Table DimSalesPerson';
GO

Create table DimSalesPerson
(
SalesPersonID int primary key identity,
SalesPersonAltID varchar(10)not null,
SalesPersonName varchar(100),
StoreID int,
City varchar(100),
State varchar(100),
Country varchar(100)
)
Go
```

Run the following script to insert some sample values.

```
Insert into DimSalesPerson(SalesPersonAltID,SalesPersonName,StoreID,City,State,Country)
Values
('SP-DMSPR1','Ashish',1,'Ahmedabad','Guj','India'),
('SP-DMSPR2','Ketan',1,'Ahmedabad','Guj','India'),
('SP-DMNGR1','Srinivas',2,'Ahmedabad','Guj','India'),
('SP-DMNGR2','Saad',2,'Ahmedabad','Guj','India'),
('SP-DMSVR1','Jasmin',3,'Ahmedabad','Guj','India'),
('SP-DMSVR2','Jacob',3,'Ahmedabad','Guj','India');
Go
```

Step 6: Run the following script to create Fact Table

```
PRINT '';
PRINT '*** Creating Table FactProductSales';
GO

Create Table FactProductSales
(
TransactionId bigint primary key identity,
SalesInvoiceNumber int not null,
StoreID int not null,
CustomerID int not null,
ProductID int not null,
SalesPersonID int not null,
Quantity float,
SalesTotalCost money,
ProductActualCost money,
Deviation float
)
Go
```

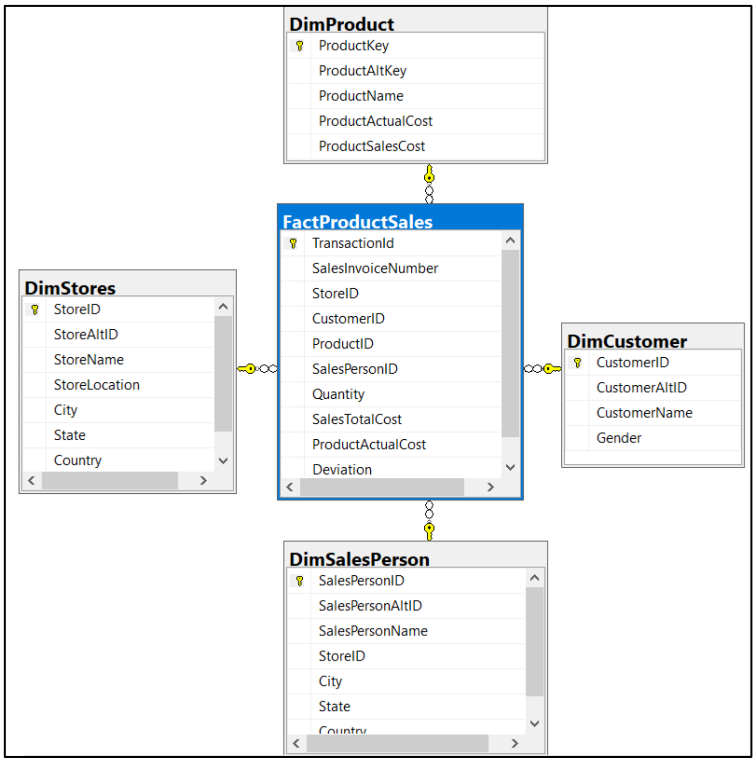
Run the following script to add relations between Fact table and dimension tables:

```
PRINT '';
PRINT '*** Add relation between fact table foreign keys to Primary keys of Dimensions';
GO

AlTER TABLE FactProductSales ADD CONSTRAINT
FK_StoreID FOREIGN KEY (StoreID)REFERENCES DimStores(StoreID);
AlTER TABLE FactProductSales ADD CONSTRAINT
FK_CustomerID FOREIGN KEY (CustomerID)REFERENCES Dimcustomer(CustomerID);
AlTER TABLE FactProductSales ADD CONSTRAINT
FK_ProductKey FOREIGN KEY (ProductID)REFERENCES Dimproduct(ProductKey);
AlTER TABLE FactProductSales ADD CONSTRAINT
FK_SalesPersonID FOREIGN KEY (SalesPersonID)REFERENCES Dimsalesperson(SalesPersonID);
Go
```

Now, populate the Fact table FactProductSales

```
Insert into FactProductSales(SalesInvoiceNumber,StoreID,CustomerID,ProductID,
SalesPersonID,Quantity,ProductActualCost,SalesTotalCost,Deviation)
values
(1,1,1,1,1,2,11,13,2),
(1,1,1,2,1,1,22.50,24,1.5),
(1,1,1,3,1,1,42,43.5,1.5),

(2,1,2,3,1,1,42,43.5,1.5),
(2,1,2,4,1,3,54,60,6),

(3,1,3,2,2,2,11,13,2),
(3,1,3,3,2,1,42,43.5,1.5),
(3,1,3,4,2,3,54,60,6),
(3,1,3,5,2,1,135,139,4),

(4,1,1,1,1,2,11,13,2),
(4,1,1,2,1,1,22.50,24,1.5),

(5,1,2,3,1,1,42,43.5,1.5),
(5,1,2,4,1,3,54,60,6),

(6,1,3,2,2,2,11,13,2),
(6,1,3,5,2,1,135,139,4),
(7,2,1,4,3,3,54,60,6),
(7,2,1,5,3,1,135,139,4),

(8,1,1,3,1,2,84,87,3),
(8,1,1,4,1,3,54,60,3),

(9,1,2,1,1,1,5.5,6.5,1),
(9,1,2,2,1,1,22.50,24,1.5),

(10,1,3,1,2,2,11,13,2),
(10,1,3,4,2,3,54,60,6),

(11,2,1,2,3,1,5.5,6.5,1),
(11,2,1,3,3,1,42,43.5,1.5)
Go
```

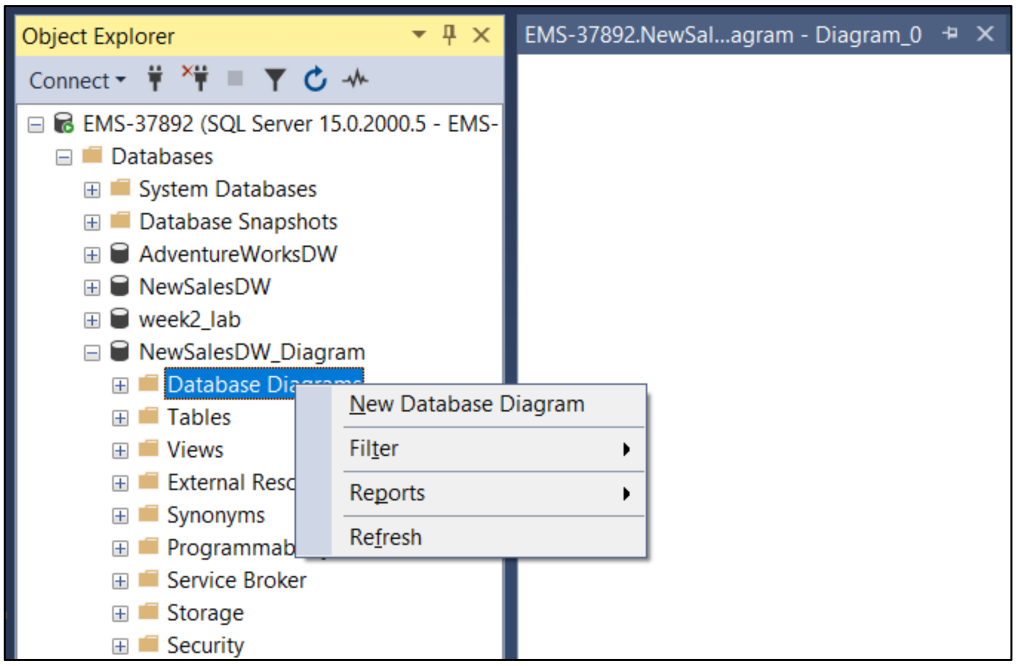
Step 7: Now, execute all the above scripts (if not yet executed) and refresh the Object Explorer at the left pane of SSMS. Generate the database diagram by expanding “Database Diagrams” to look at the overall design of the newly created database **NewSalesDW**. Note, some SSMS can not automatically generate the database diagram, therefore, generate it manually by adding the Fact table and dimension tables (refer, week 3 lab). Note this database diagram down (e.g. through screenshot). You should have a database diagram similar to the one below.



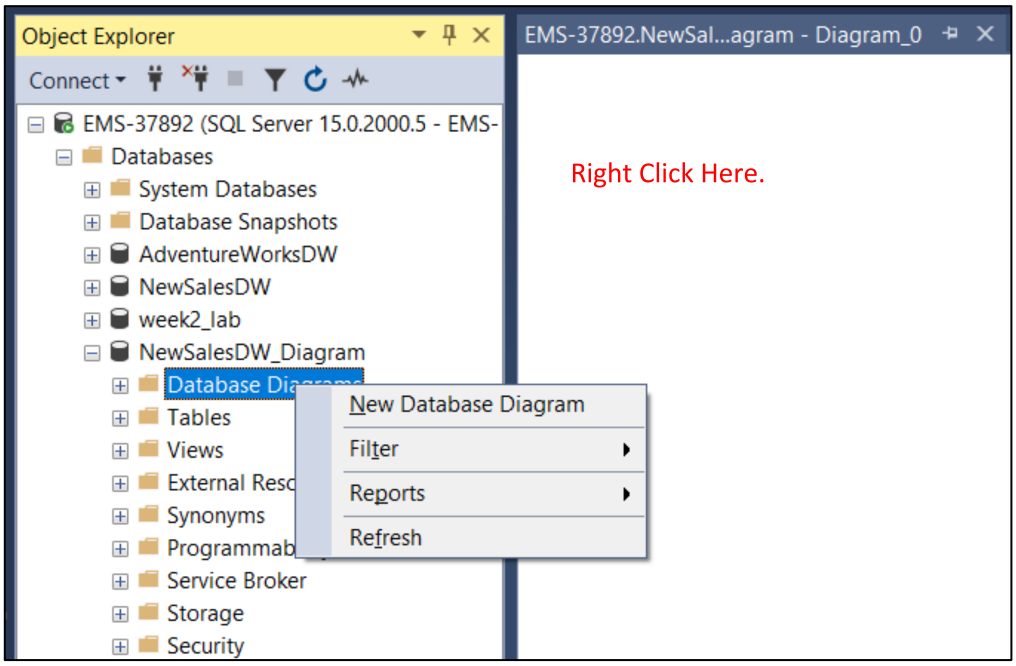
Task 1.2: Data Warehouse Creation using Database Diagram

Step 1: Open SQL Server Management Studio (SSMS) and Connect to Database Engine

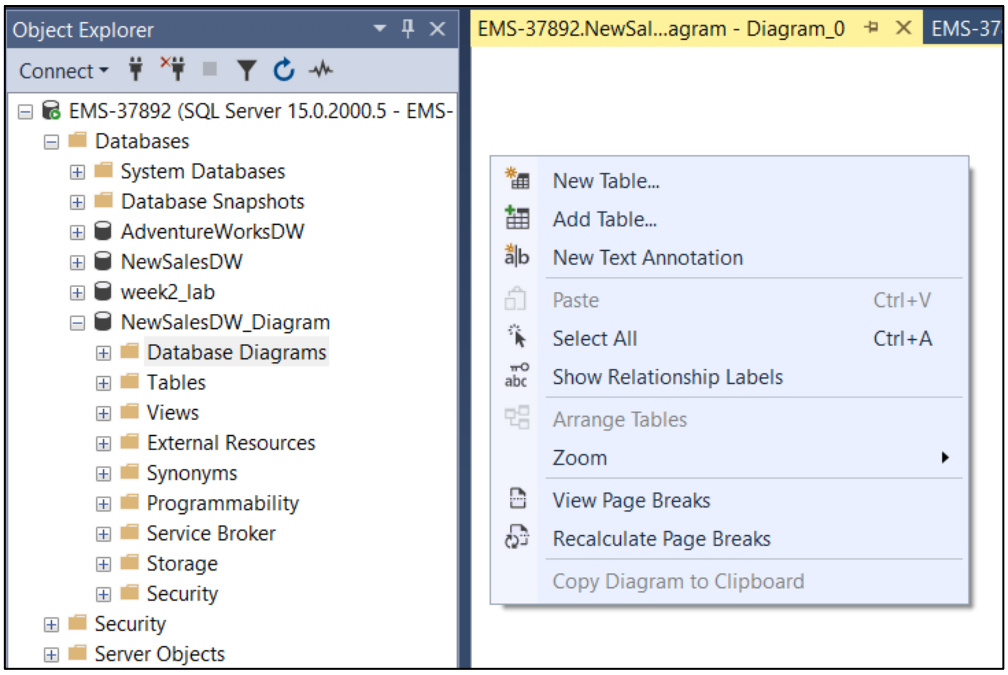
Step 2: Create Database '**NewSalesDW_Diagram**' (right click on "Databases" in Object Explorer and create New Database). Expand '**NewSalesDW_Diagram**' in the Object Explorer pane to see the following screen, and click "New Database Diagram".



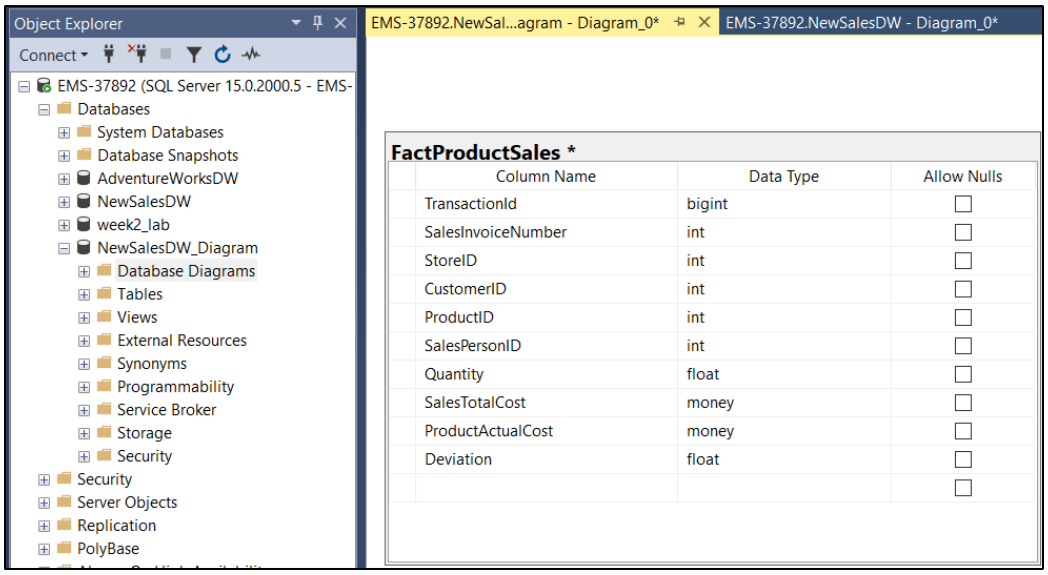
Step 3: Right click anywhere in the "Diagram" space.



Step 4: You should see the following screen.



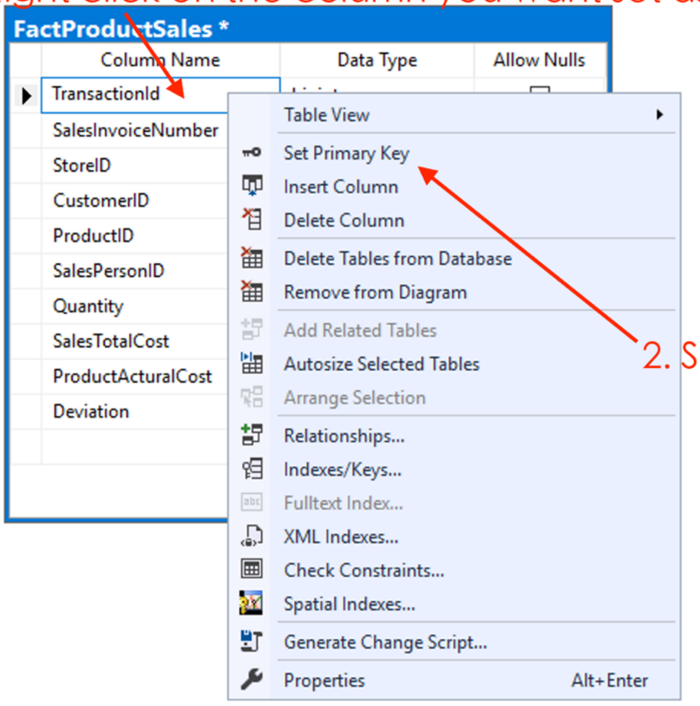
Step 5: Use the database diagram generated from **Task 1** as a template, create a diagram and generate the tables and their relations using the Database Diagram tool. Create the Fact and Dimension tables as below. Also, set Primary key and **specify it as Identity**.



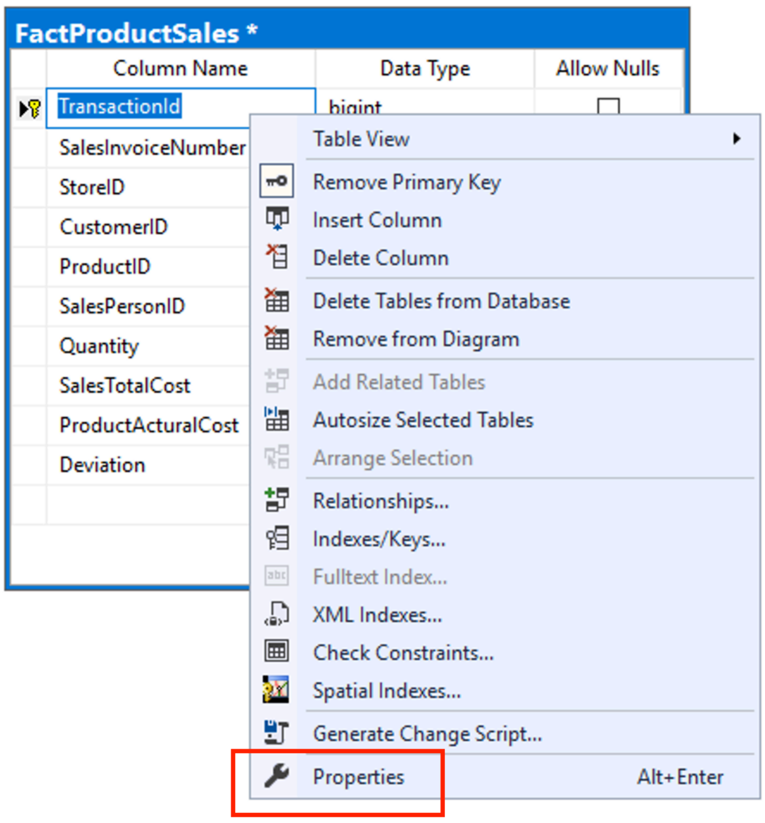
FactProductSales *

Column Name	Data Type	Allow Nulls
TransactionId	bigint	☐
SalesInvoiceNumber	int	☐
StoreID	int	☐
CustomerID	int	☐
ProductID	int	☐
SalesPersonID	int	☐
Quantity	float	☐
SalesTotalCost	money	☐
ProductActualCost	money	☐
Deviation	float	☐
		☐

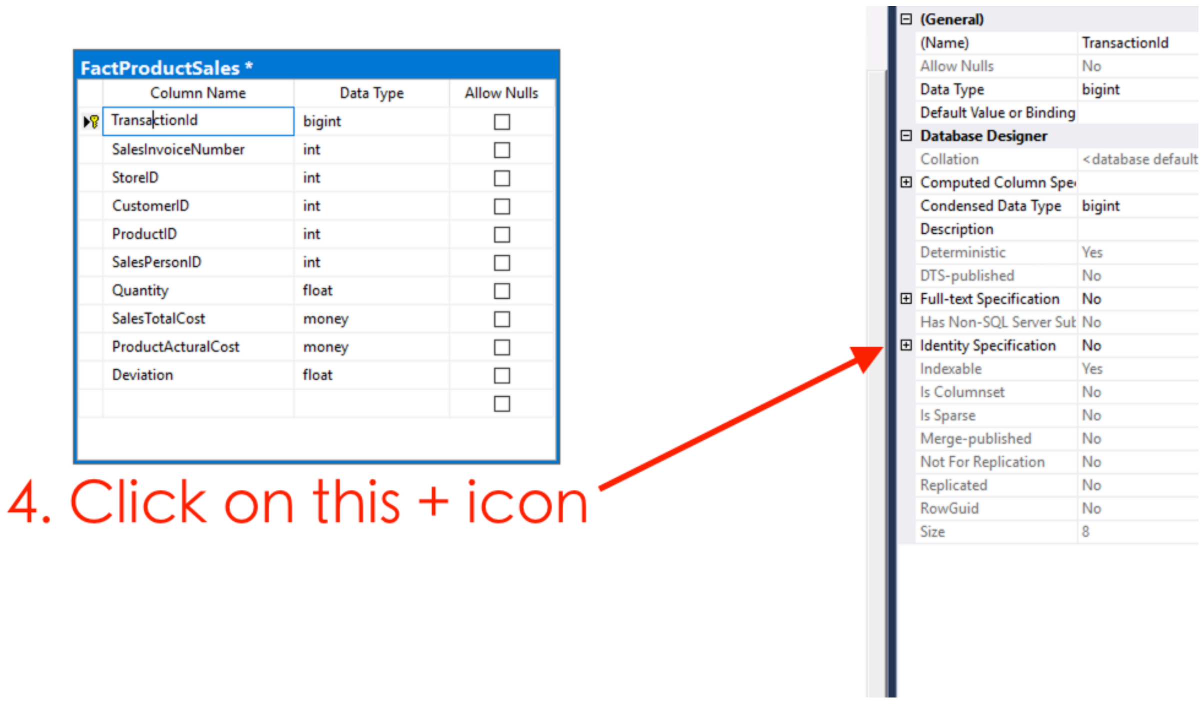
1. Right click on the column you want set as Primary Key



2. Set as Primary Key



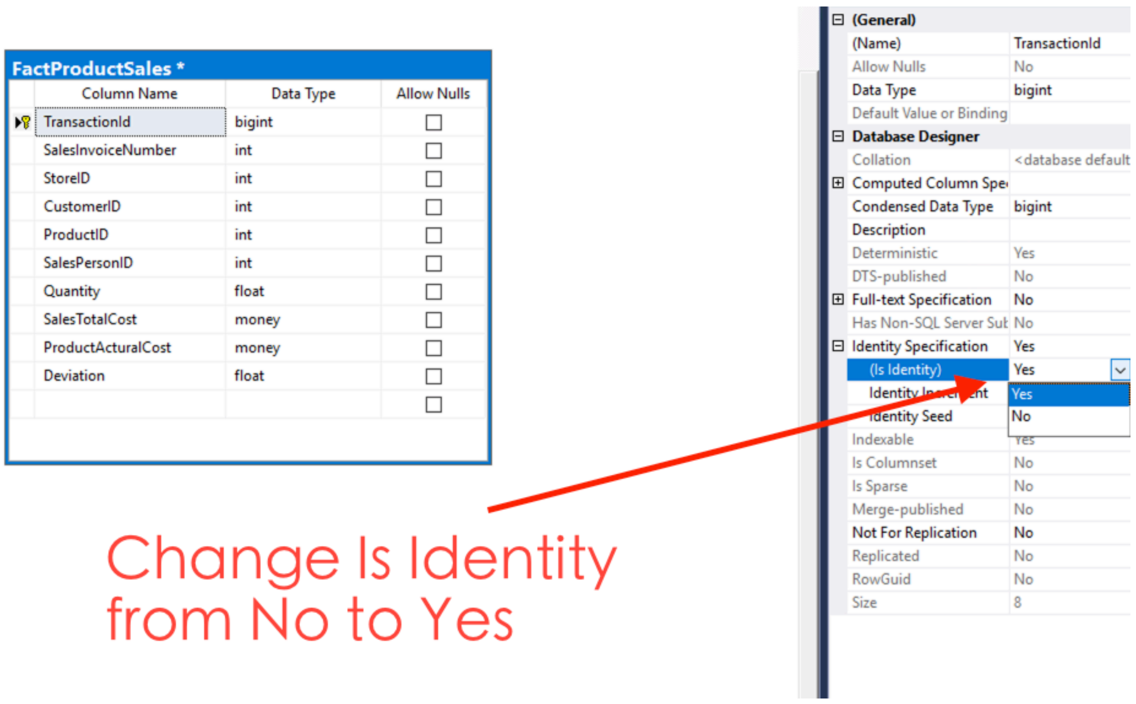
3. Right click this column again and click Properties.



FactProductSales *

Column Name	Data Type	Allow Nulls
TransactionId	bigint	☐
SalesInvoiceNumber	int	☐
StoreID	int	☐
CustomerID	int	☐
ProductID	int	☐
SalesPersonID	int	☐
Quantity	float	☐
SalesTotalCost	money	☐
ProductActuralCost	money	☐
Deviation	float	☐
		☐

⊟ **(General)**	
(Name)	TransactionId
Allow Nulls	No
Data Type	bigint
Default Value or Binding	
⊟ **Database Designer**	
Collation	<database default
⊞ Computed Column Spe	
Condensed Data Type	bigint
Description	
Deterministic	Yes
DTS-published	No
⊞ Full-text Specification	No
Has Non-SQL Server Sul	No
⊞ Identity Specification	No
Indexable	Yes
Is Columnset	No
Is Sparse	No
Merge-published	No
Not For Replication	No
Replicated	No
RowGuid	No
Size	8

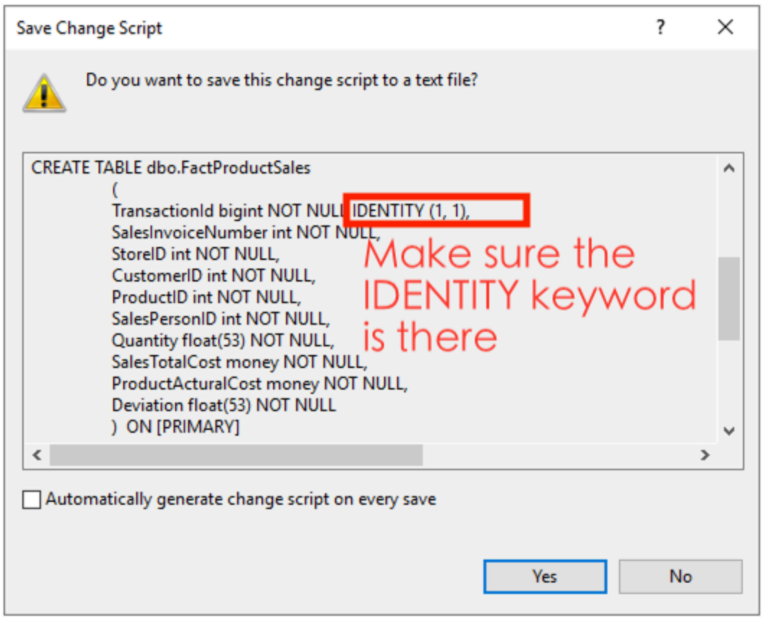


FactProductSales *

Column Name	Data Type	Allow Nulls
TransactionId	bigint	☐
SalesInvoiceNumber	int	☐
StoreID	int	☐
CustomerID	int	☐
ProductID	int	☐
SalesPersonID	int	☐
Quantity	float	☐
SalesTotalCost	money	☐
ProductActuralCost	money	☐
Deviation	float	☐
		☐

⊟ **(General)**	
(Name)	TransactionId
Allow Nulls	No
Data Type	bigint
Default Value or Binding	
⊟ **Database Designer**	
Collation	<database default
⊞ Computed Column Spe	
Condensed Data Type	bigint
Description	
Deterministic	Yes
DTS-published	No
⊞ Full-text Specification	No
Has Non-SQL Server Sul	No
⊟ Identity Specification	Yes
(Is Identity)	Yes
Identity Increment	Yes
Identity Seed	No
Indexable	Yes
Is Columnset	No
Is Sparse	No
Merge-published	No
Not For Replication	No
Replicated	No
RowGuid	No
Size	8

One can generate the script by right click on created table and click “Generate Change Script..”



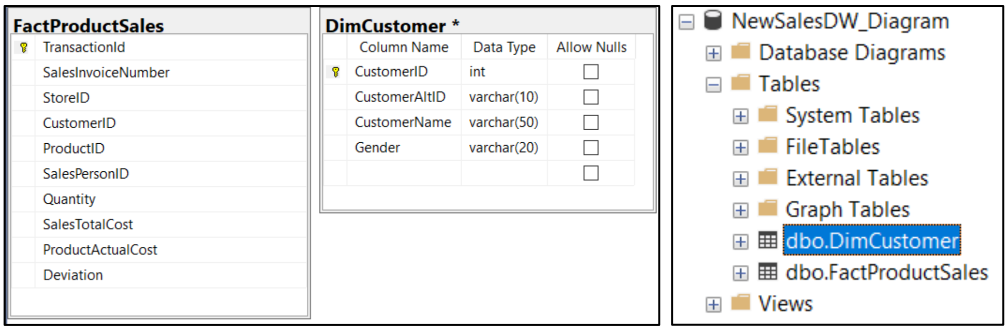
Do you want to save this change script to a text file?

```
CREATE TABLE dbo.FactProductSales
	(
	TransactionId bigint NOT NULL IDENTITY (1, 1),
	SalesInvoiceNumber int NOT NULL,
	StoreID int NOT NULL,
	CustomerID int NOT NULL,
	ProductID int NOT NULL,
	SalesPersonID int NOT NULL,
	Quantity float(53) NOT NULL,
	SalesTotalCost money NOT NULL,
	ProductActuralCost money NOT NULL,
	Deviation float(53) NOT NULL
	)  ON [PRIMARY]
```

Make sure the IDENTITY keyword is there

☐ Automatically generate change script on every save

Similarly, add other tables (e.g., dim tables) and save the diagram(s). Refresh the object explorer and can see that new tables are available under "Tables". In the below figure, we show two tables FactProductSales and DimCustomer in the center pane (e.g., diagram). You need to create FactProductSales and the 'four' dimension tables and save the diagram as "DiagramTest".



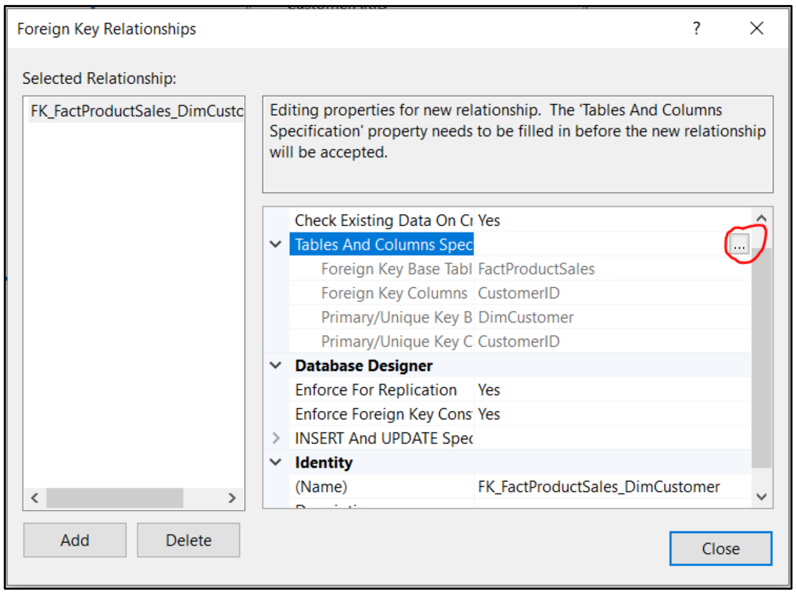
FactProductSales

TransactionId
SalesInvoiceNumber
StoreID
CustomerID
ProductID
SalesPersonID
Quantity
SalesTotalCost
ProductActualCost
Deviation

DimCustomer *

Column Name	Data Type	Allow Nulls
CustomerID	int	☐
CustomerAltID	varchar(10)	☐
CustomerName	varchar(50)	☐
Gender	varchar(20)	☐
		☐

Now, right click on FactProductSales and click on "Relationships…". Click on "Add" to establish a new relationship between Fact table and dimension tables. Below, we give an example to establish the relationships between FactProductSales and DimCustomer table.

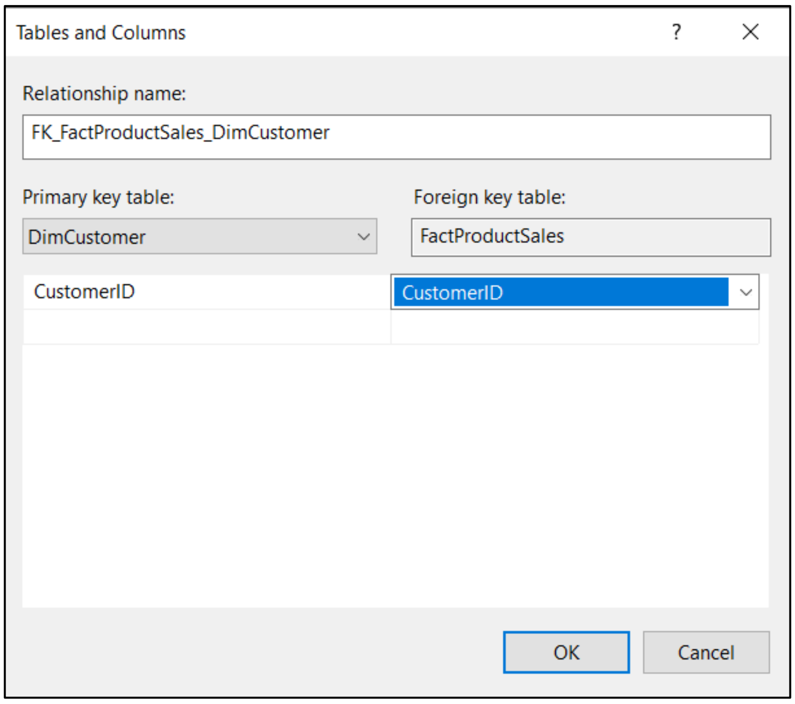

Foreign Key Relationships

Selected Relationship:

FK_FactProductSales_DimCustc

Editing properties for new relationship. The 'Tables And Columns Specification' property needs to be filled in before the new relationship will be accepted.

Check Existing Data On Cı	Yes
Tables And Columns Spec	
Foreign Key Base Tabl	FactProductSales
Foreign Key Columns	CustomerID
Primary/Unique Key B	DimCustomer
Primary/Unique Key C	CustomerID
Database Designer	
Enforce For Replication	Yes
Enforce Foreign Key Cons	Yes
INSERT And UPDATE Spec	
Identity	
(Name)	FK_FactProductSales_DimCustomer


Tables and Columns

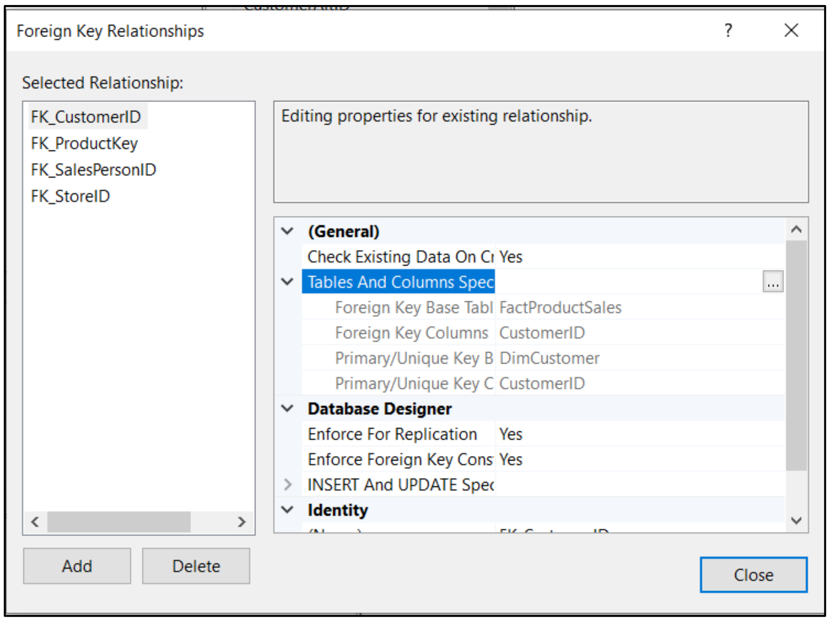
Relationship name: FK_FactProductSales_DimCustomer

Primary key table: DimCustomer

Foreign key table: FactProductSales

Primary key table	Foreign key table
CustomerID	CustomerID

The final relationship screen will be as below. One can change the identity name also. For example, we rename “FK_FactProductSales_DimCustomer” with simple identity “FK_CustomerId”. Similarly, establish the relationships between FactProductSales and the remaining three dimension tables.


Foreign Key Relationships

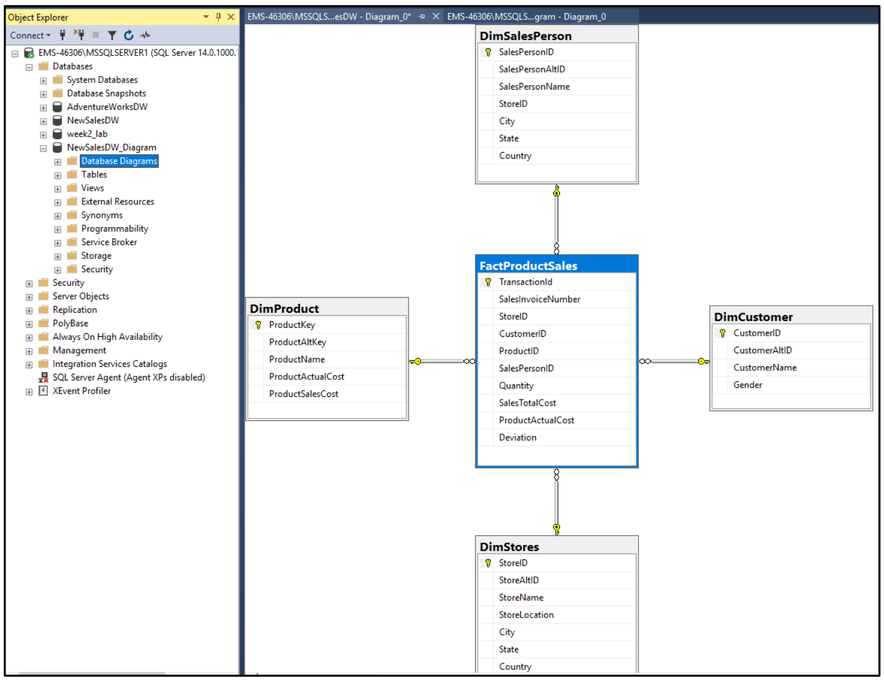
Selected Relationship:

FK_CustomerID
FK_ProductKey
FK_SalesPersonID
FK_StoreID

Editing properties for existing relationship.

(General)	
Check Existing Data On Cı	Yes
Tables And Columns Spec	
Foreign Key Base Tabl	FactProductSales
Foreign Key Columns	CustomerID
Primary/Unique Key B	DimCustomer
Primary/Unique Key C	CustomerID
Database Designer	
Enforce For Replication	Yes
Enforce Foreign Key Cons	Yes
INSERT And UPDATE Spec	
Identity	

You should design a database diagram as follows.



Step 6: Download the **NewSalesDW_Diagram.zip**, which contain the data in **csv** files and also an installation script: CreateTables.sql and InsertData.sql. Note that in the csv files, the first column is empty, they are reserved for the auto-increment surrogate key values. You table should have been created by the Database Diagram tool in **Step 5**, so **no need to run the** CreateTables.sql script.

Step 7: Populate your NewSalesDW_Diagram database with data stored in the csv files. To get yourself prepared for the loading your project data, we strongly suggest you to only make use of the csv files, but try to write your own data loading script. The two scripts are provided for you to check against if things do not work out.

A sample bulk insert statement is included below for you to note that the construct of KEEPIDENTITY needs to be commented out for the auto-increment key to work.

```
BULK INSERT [dbo].[DimCustomer] FROM '$(SqlSamplesSourceDataPath)DimCustomer.csv'
WITH (
   CHECK_CONSTRAINTS,
   --CODEPAGE='ACP',
   DATAFILETYPE='char',
   FIELDTERMINATOR=',',
   ROWTERMINATOR='\n',
   --KEEPIDENTITY,
   TABLOCK
);
```

Task 1.3: Use SQL Queries for Rollup and Cube Aggregation

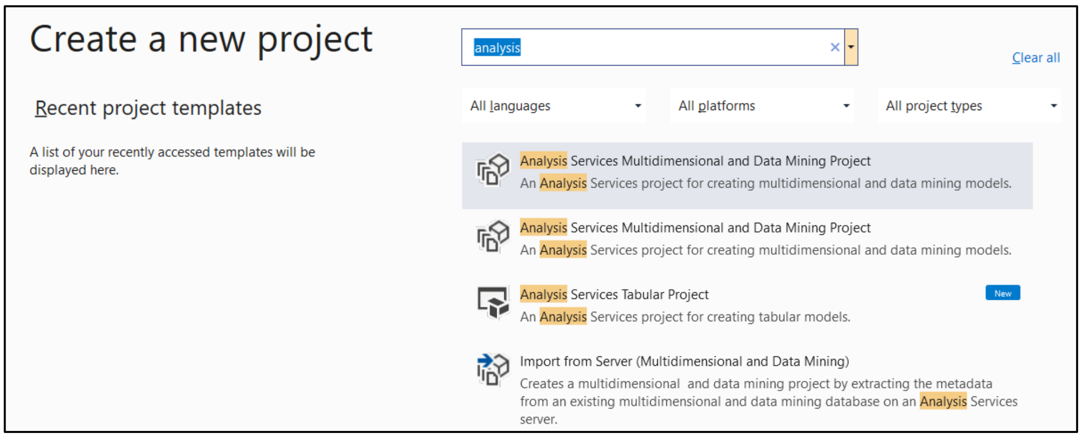
Write the SQL Queries in a similar fashion to the slides in Lecture 3 to see the differences between the *rollup* and *cube* construct. For example, you can use the following query.

```
Select ProductID, StoreID, SUM(Quantity) from dbo.FactProductSales group by cube (ProductID, StoreID)
```

Task 2: Using SSDT to create a multidimensional analysis service project

Step 1: Create a new project

Open Microsoft Visual Studio, Click on File -> New -> Project to create an *Analysis Service Multidimensional and Data Mining Project*. [If your Visual Studio doesn't have *Analysis Service Multidimensional and Data Mining Project*, please install the extensions using instructions given in Week 2 lab.



Create a new project

analysis

Clear all

Recent project templates

All languages | All platforms | All project types

A list of your recently accessed templates will be displayed here.

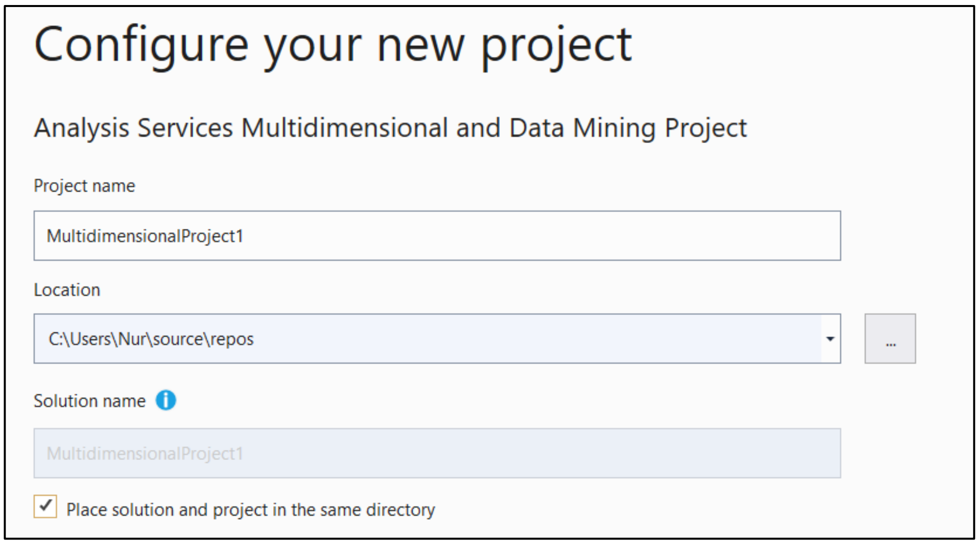
Analysis Services Multidimensional and Data Mining Project
An Analysis Services project for creating multidimensional and data mining models.

Analysis Services Multidimensional and Data Mining Project
An Analysis Services project for creating multidimensional and data mining models.

Analysis Services Tabular Project — New
An Analysis Services project for creating tabular models.

Import from Server (Multidimensional and Data Mining)
Creates a multidimensional and data mining project by extracting the metadata from an existing multidimensional and data mining database on an Analysis Services server.

Enter the project name and click "Create".



Configure your new project

Analysis Services Multidimensional and Data Mining Project

Project name: MultidimensionalProject1

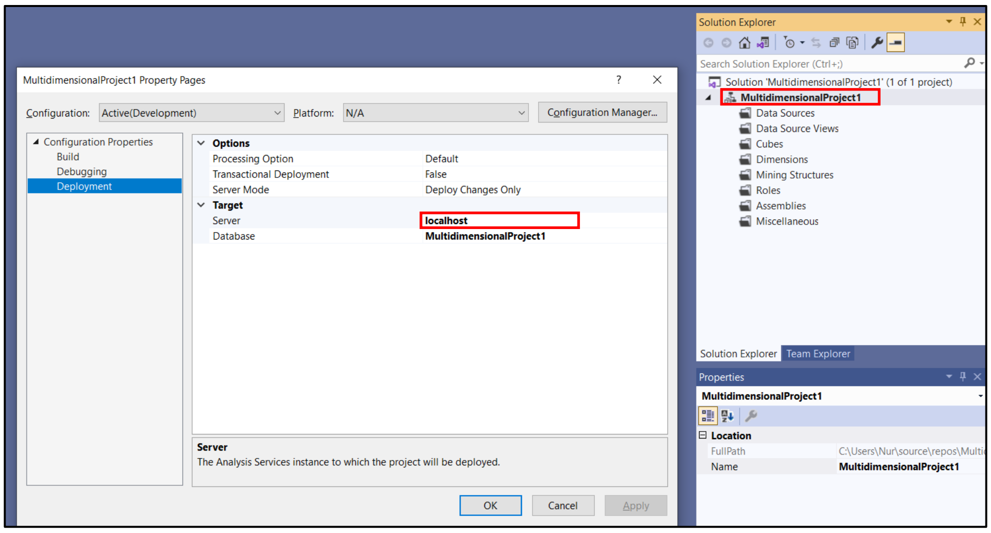
Location: C:\Users\Nur\source\repos

Solution name: MultidimensionalProject1

☑ Place solution and project in the same directory

Step 2: Set the correct SQL Server Analysis Service Instance for deployment

This will open the Solution Explorer of a new project on the right-hand side of the window. Right click on the project name, and select "Properties", and replace the Deployment server (localhost) with the right SQL Server Analysis Service Instance.



MultidimensionalProject1 Property Pages

Configuration: Active(Development) Platform: N/A

Options	
Processing Option	Default
Transactional Deployment	False
Server Mode	Deploy Changes Only
Target	
Server	localhost
Database	MultidimensionalProject1

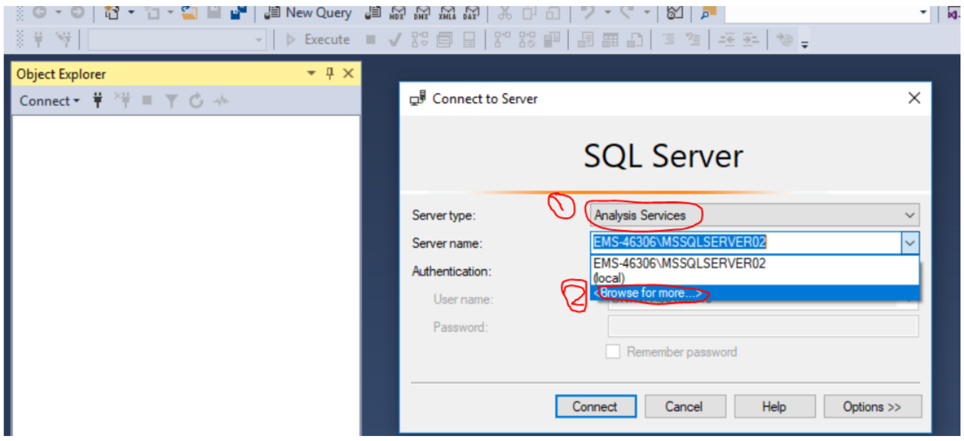
Server
The Analysis Services instance to which the project will be deployed.

Solution 'MultidimensionalProject1' (1 of 1 project)
MultidimensionalProject1
Data Sources
Data Source Views
Cubes
Dimensions
Mining Structures
Roles
Assemblies
Miscellaneous

Properties: MultidimensionalProject1

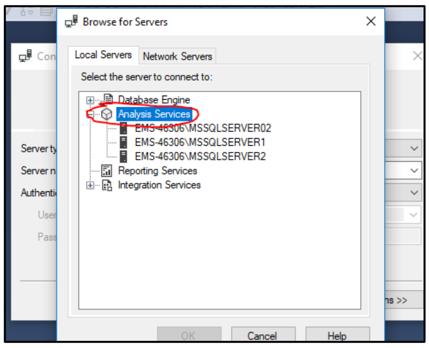
Location	
FullPath	C:\Users\Nur\source\repos\Multi
Name	MultidimensionalProject1

To find the SQL Server Analysis Instance installed on your machine, open SSMS (SQL Server Management Studio), select "Analysis Services" from the "Server type" drop-down list, and select "Browse from more…" from the "Server name" drop-down list. Please see screenshot below.



Server type: Analysis Services
Server name: EMS-46306\MSSQLSERVER02
EMS-46306\MSSQLSERVER02
(local)
<Browse for more...>
Authentication:
User name:
Password:
Remember password

Select a server under "Analysis Services". [If no server is displayed, this means your system has no SQL Analysis Service installed.]

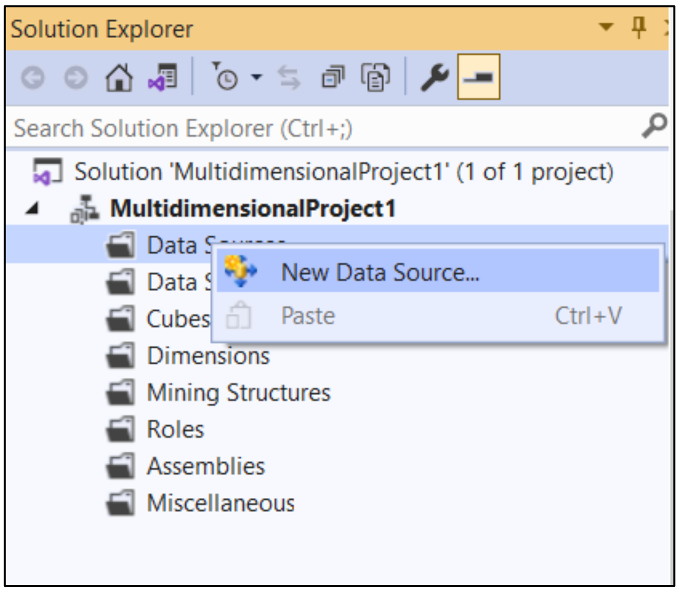


Select the server to connect to:
Database Engine
Analysis Services
EMS-46306\MSSQLSERVER02
EMS-46306\MSSQLSERVER1
EMS-46306\MSSQLSERVER2
Reporting Services
Integration Services

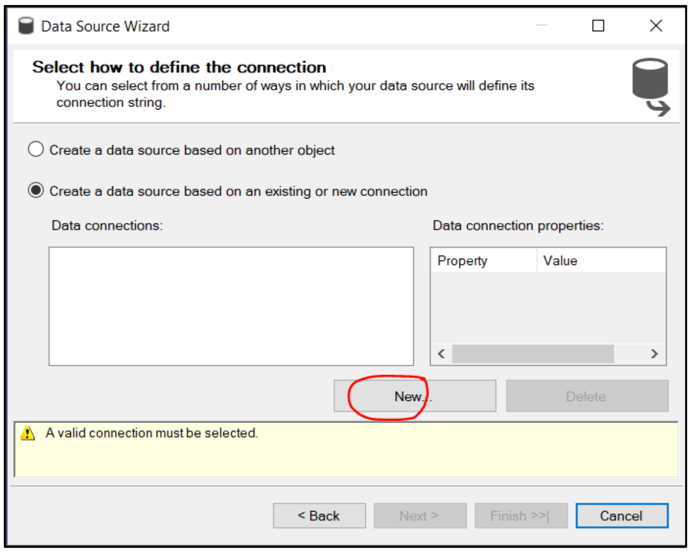
Note: If you are working on your own machine (locally installed SQL server) with **single** SQL server instance, then no need to follow Step 2 (keep the server as default localhost). However, we recommend to replace the localhost with the exact server instance.

Step 3: Create the Data Source

Right click on the "Data Sources" using MS Visual Studio, and select "New Data Source…."



Follow the wizard to connect the NewSalesDW database:


Data Source Wizard

Select how to define the connection
You can select from a number of ways in which your data source will define its connection string.

○ Create a data source based on another object

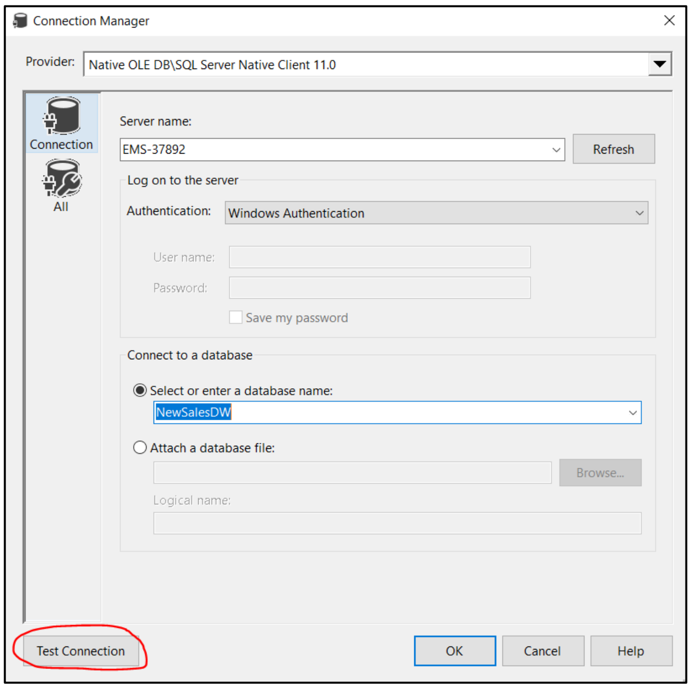
● Create a data source based on an existing or new connection

Data connections:

Data connection properties:

Property	Value

A valid connection must be selected.


Connection Manager

Provider: Native OLE DB\SQL Server Native Client 11.0

Server name: EMS-37892

Log on to the server

Authentication: Windows Authentication

User name:

Password:

☐ Save my password

Connect to a database

● Select or enter a database name: NewSalesDW

○ Attach a database file:

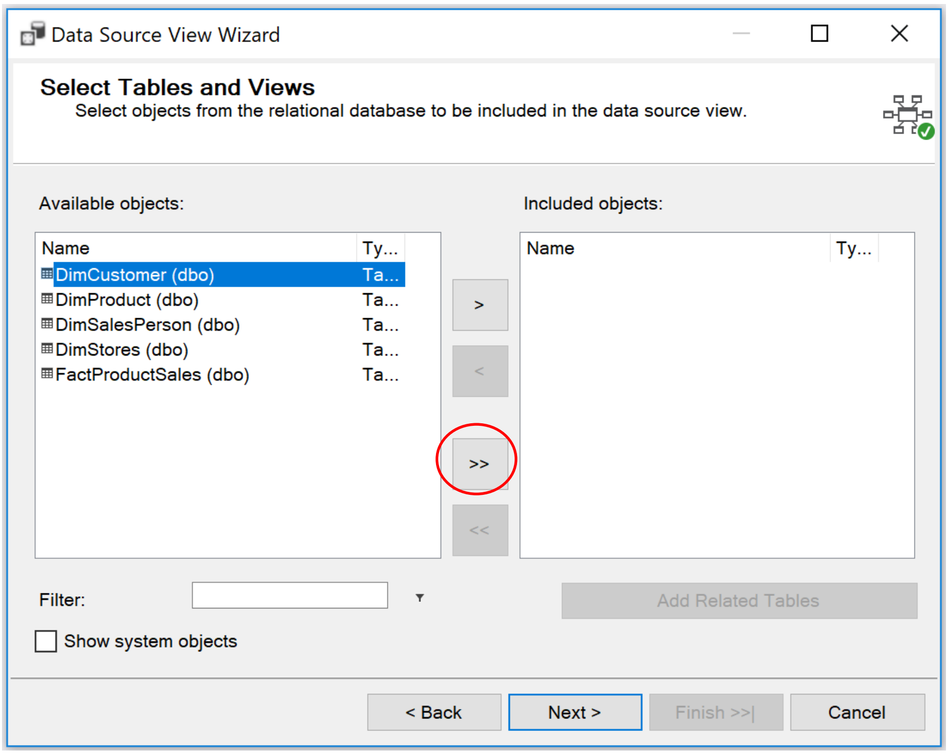
Logical name:

And test connection and click “OK” and Finish.

Step 4: Create a Data Source View

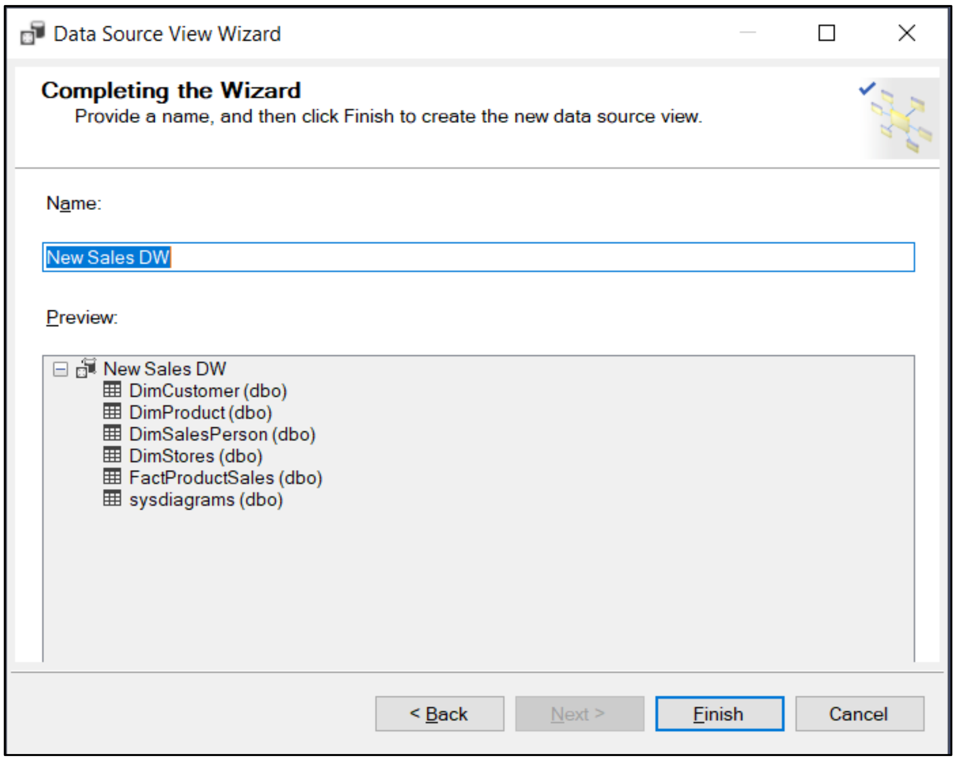
Right click on “Data Source Views” in the Solution Explorer and click on “New Data Source View…”

After bringing up the “Data Source View Wizard”, follow the prompt to the following screen, select all tables (Fact and dimension tables) to be included on the right hand side.



Name	Ty...
DimCustomer (dbo)	Ta...
DimProduct (dbo)	Ta...
DimSalesPerson (dbo)	Ta...
DimStores (dbo)	Ta...
FactProductSales (dbo)	Ta...

Then click “Next” and “Finish”

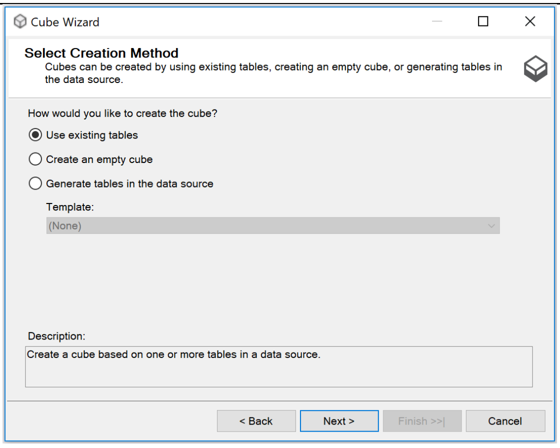


Name: New Sales DW

```
New Sales DW
    DimCustomer (dbo)
    DimProduct (dbo)
    DimSalesPerson (dbo)
    DimStores (dbo)
    FactProductSales (dbo)
    sysdiagrams (dbo)
```

Step 5: Create the Cubes

We can now use the Data Source View to create a cube. Right click on “Cubes” in the Solution Explorer to bring up the Cube Wizard, the following screenshots demonstrate how to


Cube Wizard

Select Creation Method
Cubes can be created by using existing tables, creating an empty cube, or generating tables in the data source.

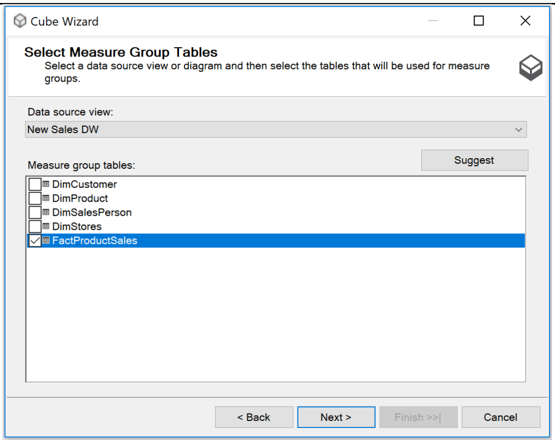
How would you like to create the cube?
● Use existing tables
○ Create an empty cube
○ Generate tables in the data source

Template:
(None)

Description:
Create a cube based on one or more tables in a data source.

< Back | Next > | Finish >>| | Cancel

(1) Use Existing Tables


Cube Wizard

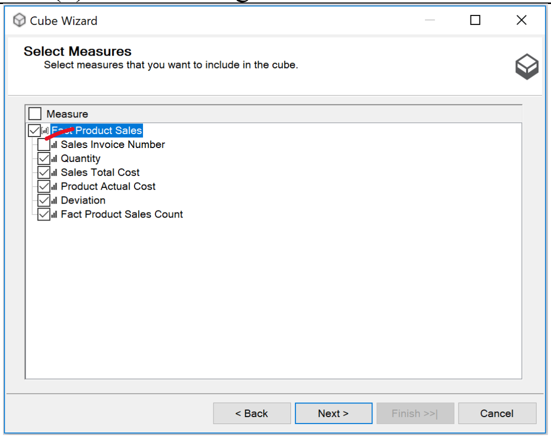
Select Measure Group Tables
Select a data source view or diagram and then select the tables that will be used for measure groups.

Data source view:
New Sales DW

Measure group tables: Suggest
☐ DimCustomer
☐ DimProduct
☐ DimSalesPerson
☐ DimStores
☑ FactProductSales

< Back | Next > | Finish >>| | Cancel

(2) Fact Table as Measure Group Table

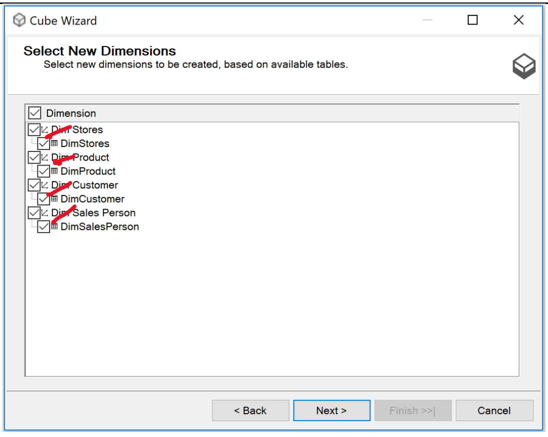

Cube Wizard

Select Measures
Select measures that you want to include in the cube.

☐ Measure
☑ ~~Fact~~ Product Sales
☐ Sales Invoice Number
☑ Quantity
☑ Sales Total Cost
☑ Product Actual Cost
☑ Deviation
☑ Fact Product Sales Count

< Back | Next > | Finish >>| | Cancel

(3) Tick the measures (rename the measure group as Product Sales, untick Sales Invoice Number)

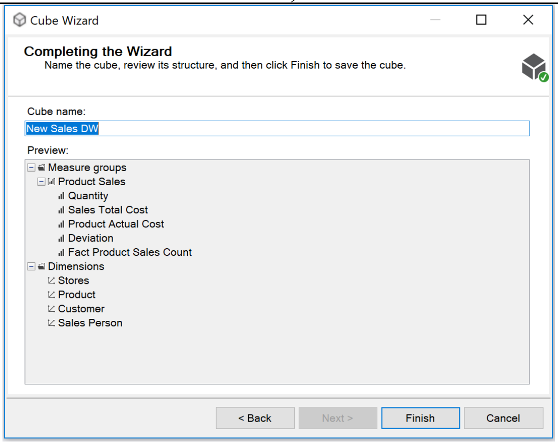

Cube Wizard

Select New Dimensions
Select new dimensions to be created, based on available tables.

☑ Dimension
☑ ~~Dim~~Stores
☑ DimStores
☑ ~~Dim~~Product
☑ DimProduct
☑ ~~Dim~~Customer
☑ DimCustomer
☑ ~~Dim~~Sales Person
☑ DimSalesPerson

< Back | Next > | Finish >>| | Cancel

(4) Rename each dimension name by removing the word Dim.


Cube Wizard

Completing the Wizard
Name the cube, review its structure, and then click Finish to save the cube.

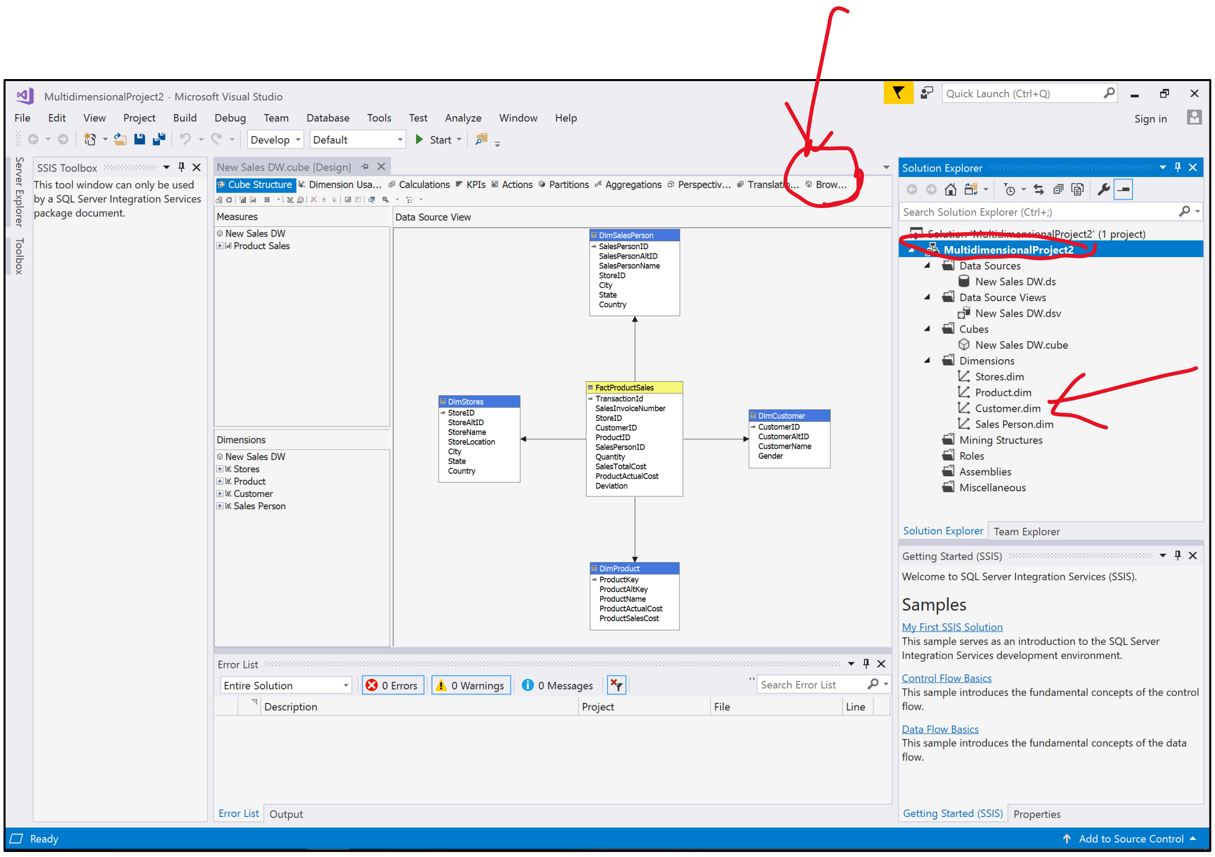
Cube name:
New Sales DW

Preview:
- Measure groups
  - Product Sales
    - Quantity
    - Sales Total Cost
    - Product Actual Cost
    - Deviation
    - Fact Product Sales Count
- Dimensions
  - Stores
  - Product
  - Customer
  - Sales Person

< Back | Next > | Finish | Cancel

(5) Once created, this is the final view of your cube

After clicking on Finish, you should see the Cube Structure Pane populated with information about the measures, dimensions and also a diagram showing the data source view. Note that the Dimensions in the Solution Explorer is populated as well.

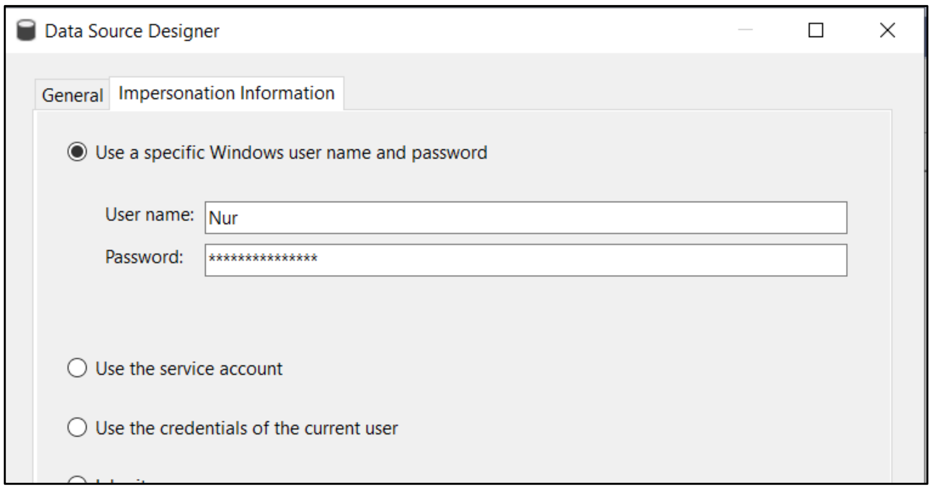


So far, we have only carried out the design, the actual values of the cube have not been extracted out of the relational database yet. So the next step is to extract data and load to the cube, and then deploy the cube on the SQL Server Analysis Services Instance.

Right click on the Project Name (in the above example screenshot - MultiDimensionalProject2), and click "Process", then a server content out date information box pops up, asking for permission to build and deploy, click "Yes".

If some error exists in deployment screen, follow the below steps to solve it.

a. Double click the Data Source Object (e.g., New Sales DW.ds) and change the credential option in Impersonation Information. Use your login information.



(●) Use a specific Windows user name and password

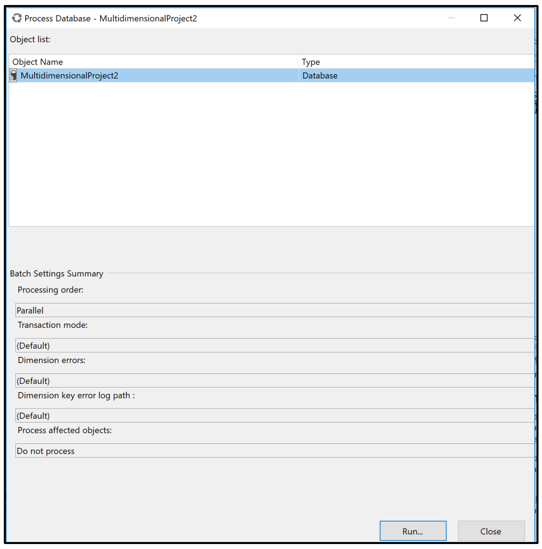
User name: Nur

Password: ***************

( ) Use the service account

( ) Use the credentials of the current user

After successful deployment, you should see the following screen for Process Databases:



Process Database - MultidimensionalProject2

Object list:

Object Name	Type
MultidimensionalProject2	Database

Batch Settings Summary

Processing order: Parallel

Transaction mode: (Default)

Dimension errors: (Default)

Dimension key error log path : (Default)

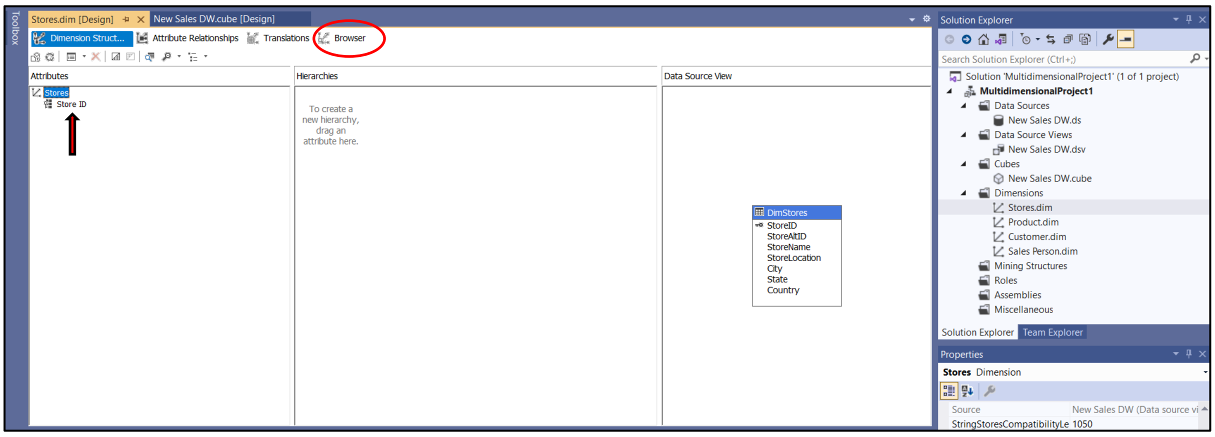
Process affected objects: Do not process

After processing, go back to the Cube Pane, and select Browse to look at a tabular view

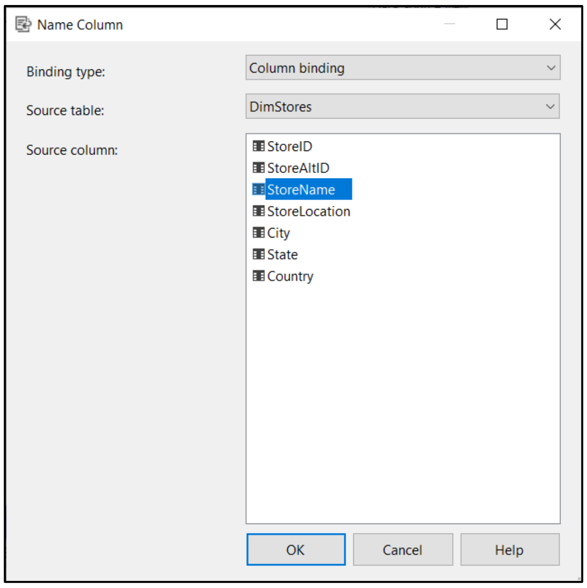
Step 6: Generate the Dimensions and fine tune the hierarchies.

Processing automatically populated the dimensions with unique values of each dimension, by default, the dimension keys. This is not very useful when we do business queries, so it is better to associate meaningful names to the keys, and also take the opportunity to create concept hierarchies if the data supports such hierarchical relations.

Take the Stores dimension for example, click on “Stores.dim” in the Solution Explorer, brings up the following screen. Click on “Browser” will show you the following screen.



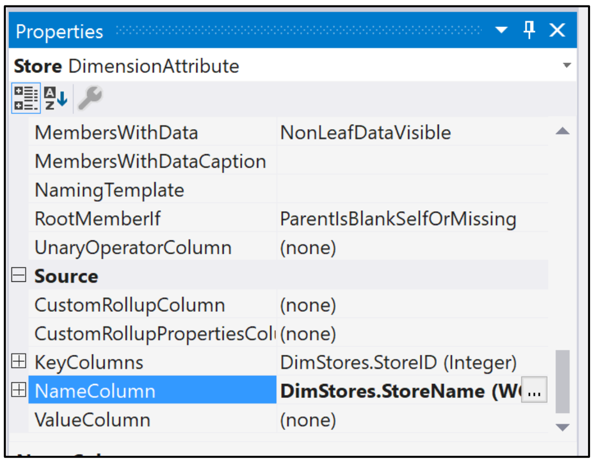
First, in the Properties pane, change the dimension’s Basic Name from Store ID to Store. Click “Store ID” on left pane. Then scroll down to find the Source of NameColumn to StoreName. This allows the display of unique store names rather than Store IDs.



Binding type: Column binding

Source table: DimStores

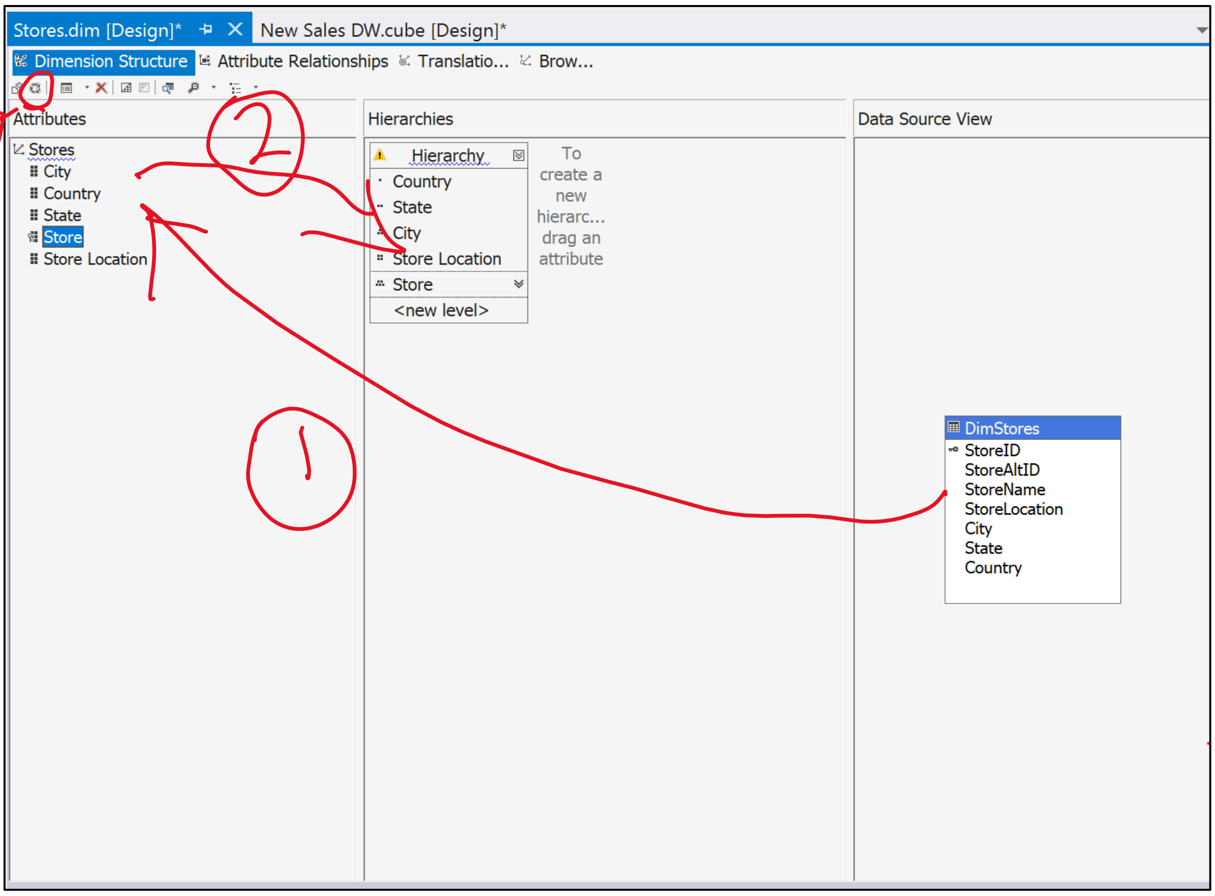
Source column:
- StoreID
- StoreAltID
- StoreName
- StoreLocation
- City
- State
- Country



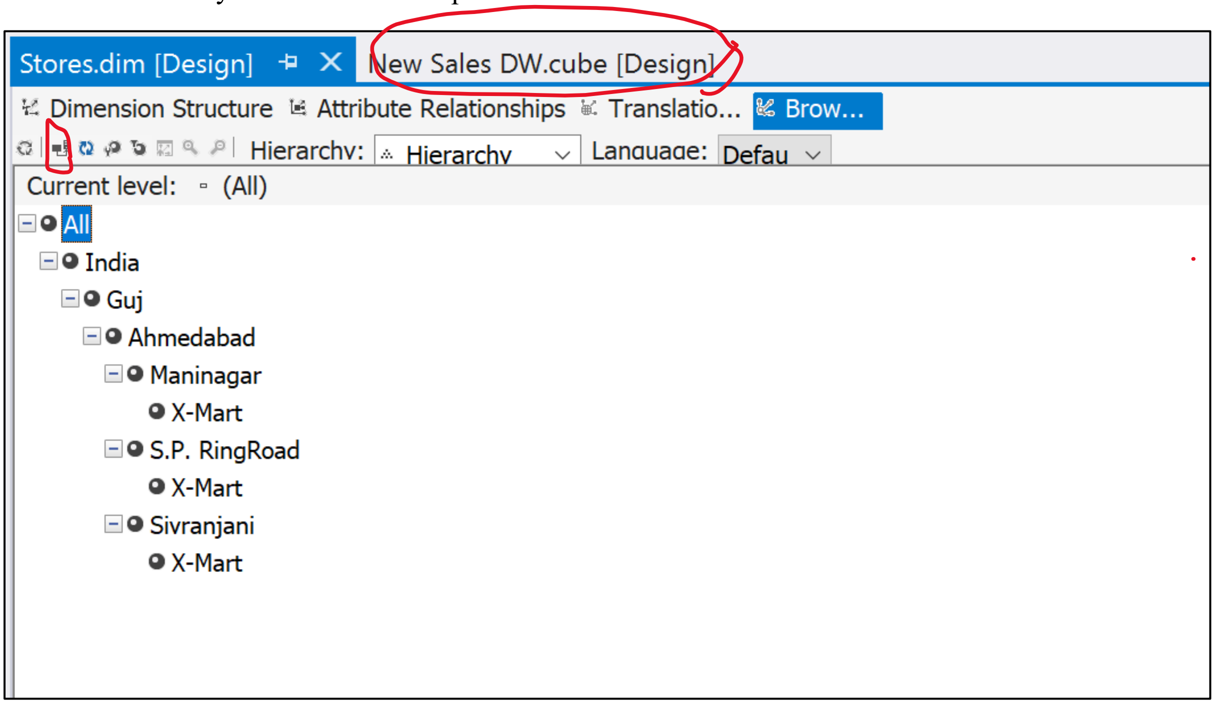
Store DimensionAttribute

MembersWithData	NonLeafDataVisible
MembersWithDataCaption	
NamingTemplate	
RootMemberIf	ParentIsBlankSelfOrMissing
UnaryOperatorColumn	(none)
Source	
CustomRollupColumn	(none)
CustomRollupPropertiesCol	(none)
KeyColumns	DimStores.StoreID (Integer)
NameColumn	**DimStores.StoreName (W**
ValueColumn	(none)

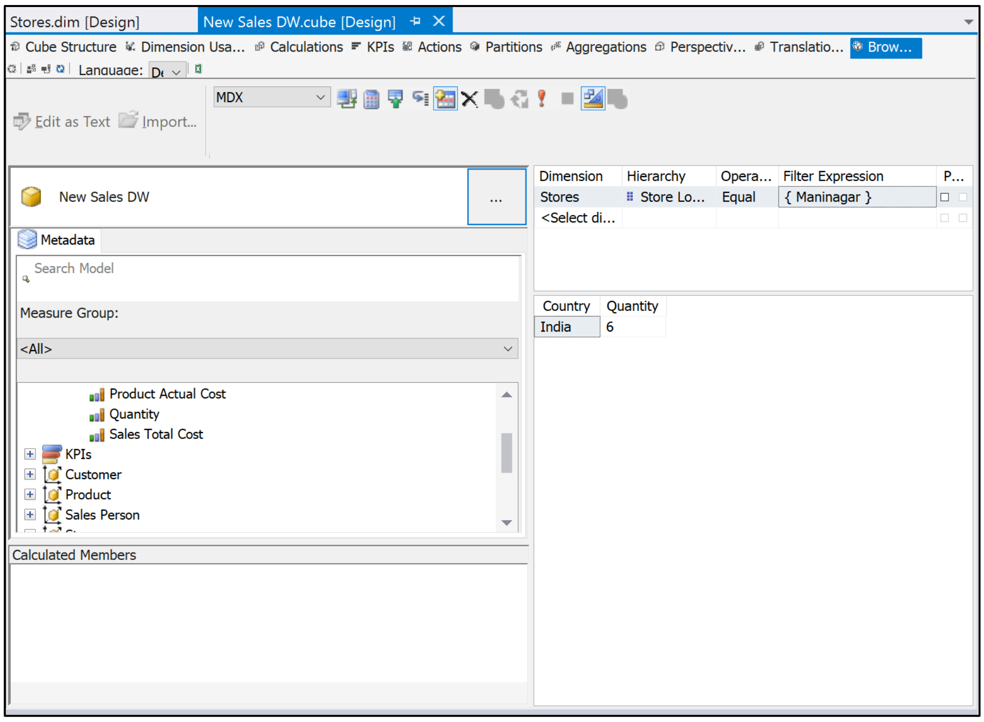
To build concept hierarchies, (1) we need to drag more attributes of the dimensions from the **Data Source View pane** to the **Attribute pane** in the dimension design window. (2) Arrange them by dragging Country, State, City, Store Location, and then Store into the **Hiearchies** pane in order. Finally, all these changes will only take place after clicking Process button (3), which extracts data from the original relational database.



After successful processing, you should see the following Hierarchy. You may need to click on the Reconnect icon if you do not see the update.



We can now switch to the *New Sales DW.cube* design Tab to have a quick play of drill down and roll up analysis of the cube.



Dimension	Hierarchy	Opera...	Filter Expression	P...
Stores	Store Lo...	Equal	{ Maninagar }	
<Select di...				

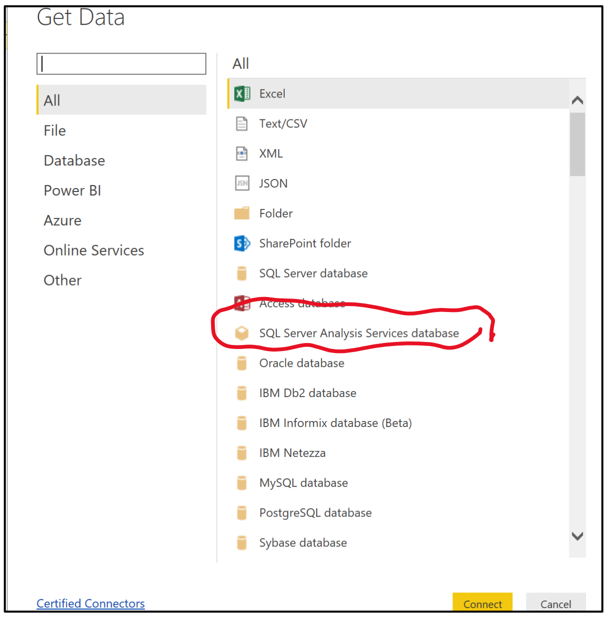
Country	Quantity
India	6

Change the store information to be more Australian related to have a better understanding of how it all works if so desired.

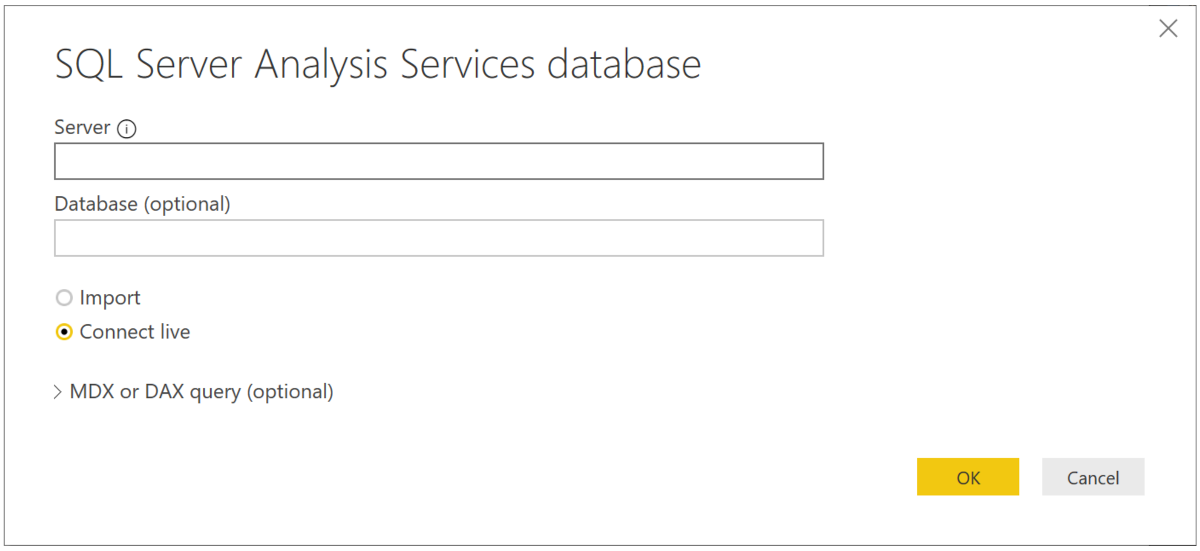
Do the same for other dimensions, namely, display meaningful unique names rather than IDs, and generate hierarchies when applicable.

Task 3: Visualising the Multidimensional Solution using Power BI

Open **Power BI Desktop**, and click on Get Data from the Home menu tool bar, this opens up the Get Data Window.



Select the “SQL Server Analysis Services database” from the list. Check the radio button “Connect live”.


SQL Server Analysis Services database

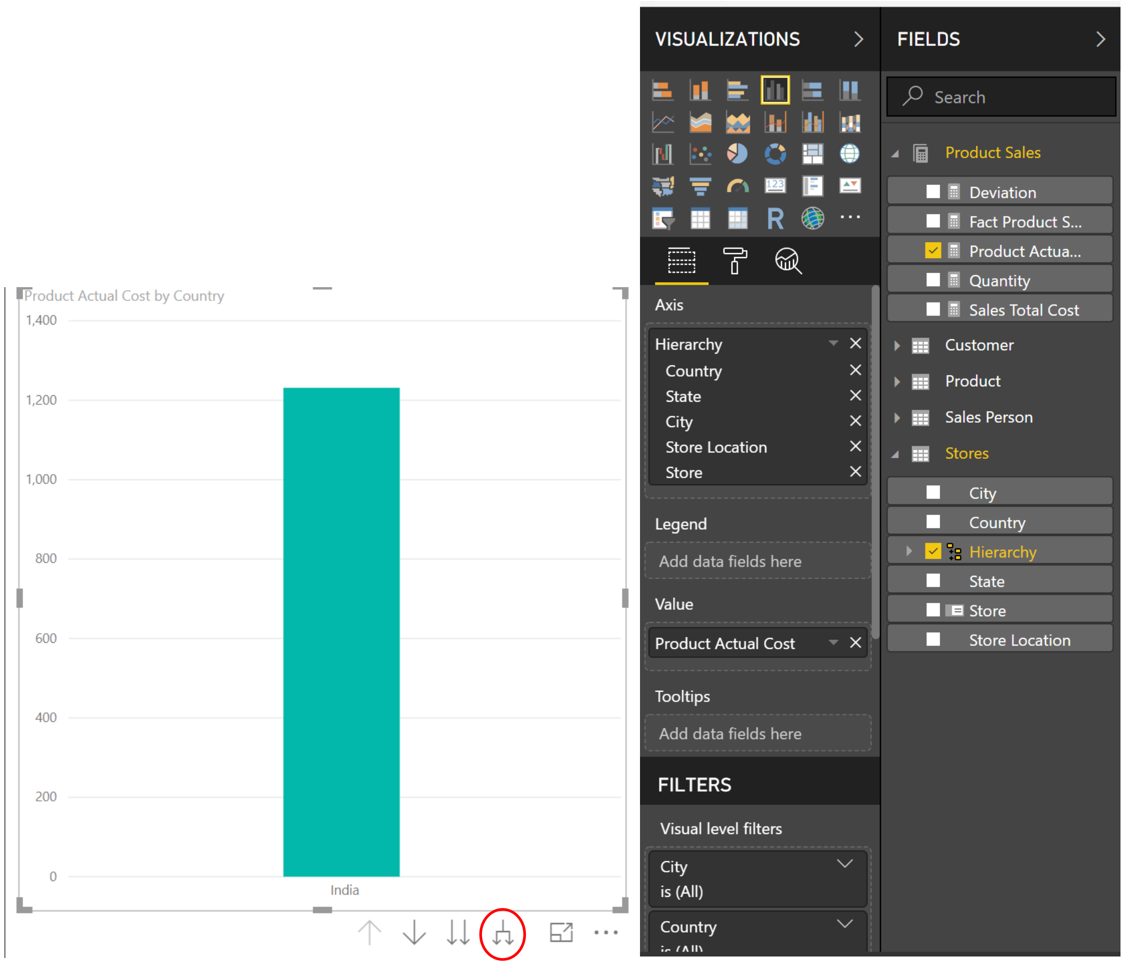
Server

Database (optional)

Import [✓ Connect live]

> MDX or DAX query (optional)

Enter the same analysis services SQL Server instance name and click OK, and follow the wizard to connect to the multi-dimensional project we have just created. A successful connection will show the “FIELDS” of fact table and dimension tables as shown on the right pane. Then select a chart type from “VISUALIZATIONS”, tick the fields you want to display for Axis and Value. You can also consider use field value for Legends and Tooltips.



VISUALIZATIONS

Axis
Hierarchy
Country
State
City
Store Location
Store

Legend
Add data fields here

Value
Product Actual Cost

Tooltips
Add data fields here

FILTERS

Visual level filters
City
is (All)
Country
is (All)

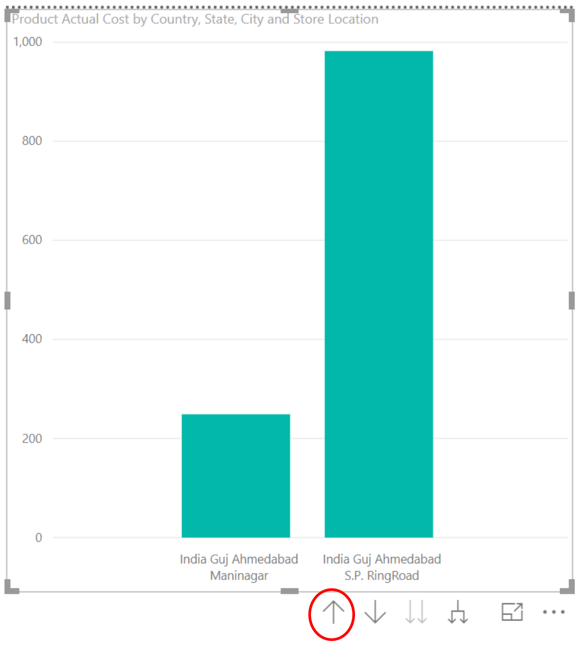
FIELDS

Search

Product Sales
- [ ] Deviation
- [ ] Fact Product S...
- [x] Product Actua...
- [ ] Quantity
- [ ] Sales Total Cost

Customer
Product
Sales Person
Stores
- [ ] City
- [ ] Country
- [x] Hierarchy
- [ ] State
- [ ] Store
- [ ] Store Location

Clicking on the Hiearchy Icon, we can drill down to see more finer detailed values of the selected measure. Similiarly, the Up Arrow is intuitively for roll up.


Product Actual Cost by Country, State, City and Store Location
1,000
800
600
400
200
0
India Guj Ahmedabad Maninagar
India Guj Ahmedabad S.P. RingRoad

Explore PowerBI interface to try other visualisations, for example, a geographical map. Find out how to insert pictures and text onto the dashboard to document your design process. Export a PDF file as report.