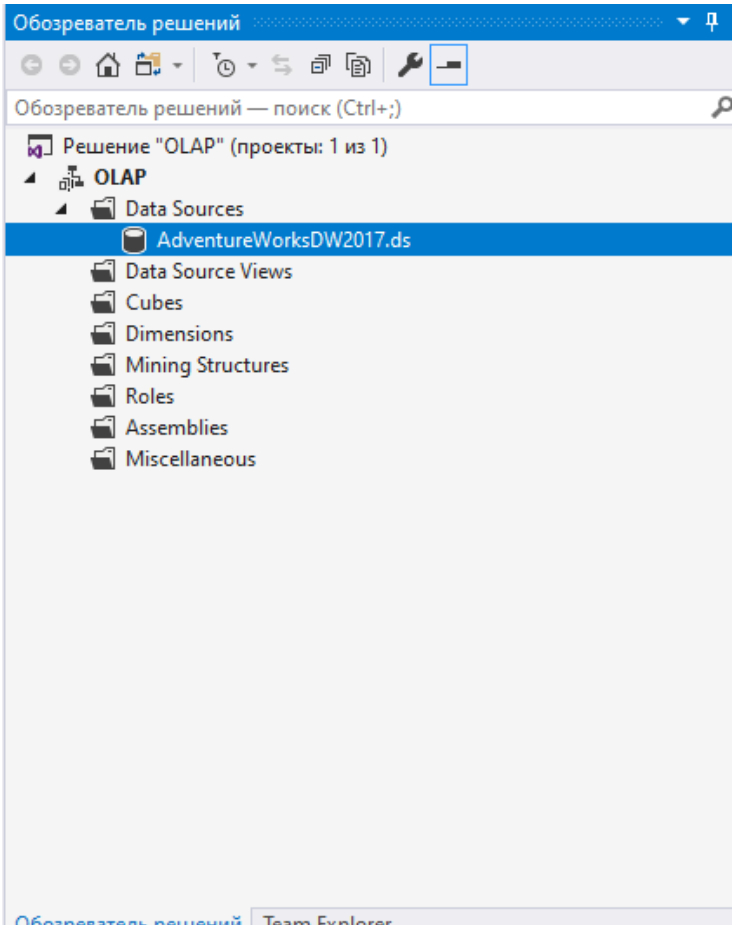
LECTURE 13

OLAP CUBES AND MDX

CREATING OLAP CUBES

It is necessary to use Analysis services to create OLAP cube. We use Adventure works 2017 database

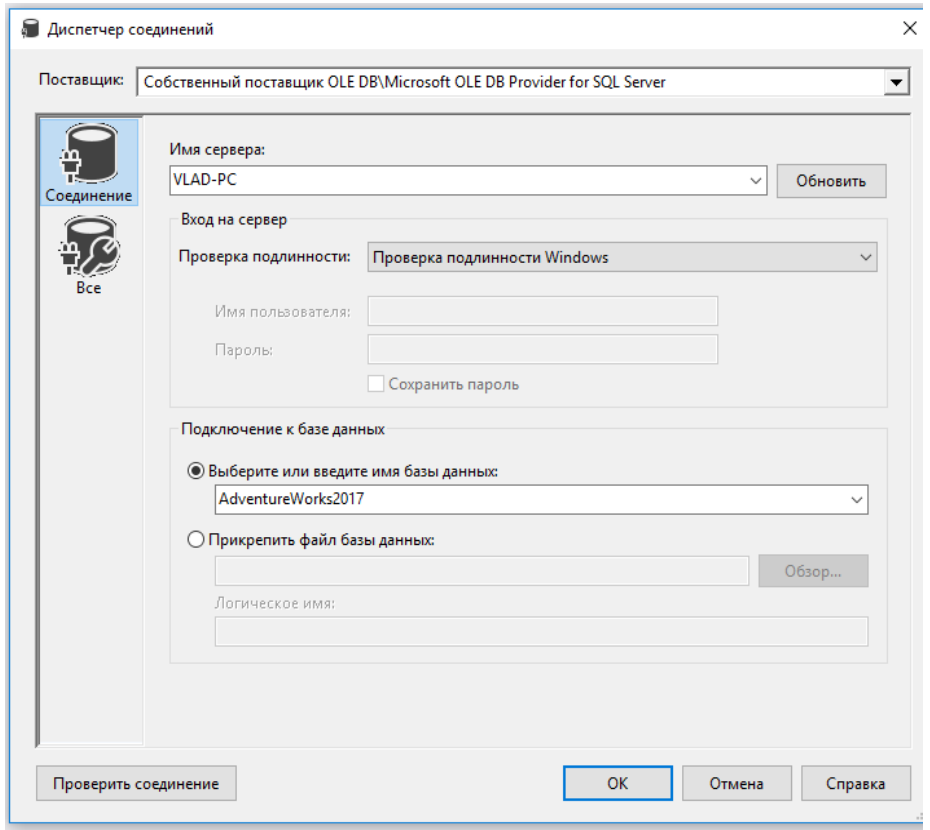


CREATING OLAP CUBES

Имя	Состояние	Режим запуска	Использовать для входа	Идентифик...	Тип службы
Службы SQL Server ...	Работает	Авто	NT Service\MsDtsServer140	22536	
SQL Full-text Filter D...	Работает	Вручную	NT Service\MSSQLFDLauncher	8744	
SQL Server Launchp...	Работает	Авто	NT Service\MSSQLLaunchpad	18184	
SQL Server (MSSQLS...	Работает	Авто	NT Service\MSSQLSERVER	13032	SQL Server
Службы SQL Server ...	Работает	Авто	NT Service\MSSQLServerOLAPService	6728	Analysis Server
Обозреватель SQL ...	Работает	Авто	NT AUTHORITY\LOCALSERVICE	4512	
SQL Server PolyBase...	Работает	Авто	NT AUTHORITY\NETWORKSERVICE	12988	SQL Server Polybase Data Movement Service
SQL Server PolyBase...	Работает	Авто	NT AUTHORITY\NETWORKSERVICE	20352	SQL Server Polybase Engine
Агент SQL Server (M...	Работает	Авто	NT Service\SQLSERVERAGENT	12788	SQL Agent

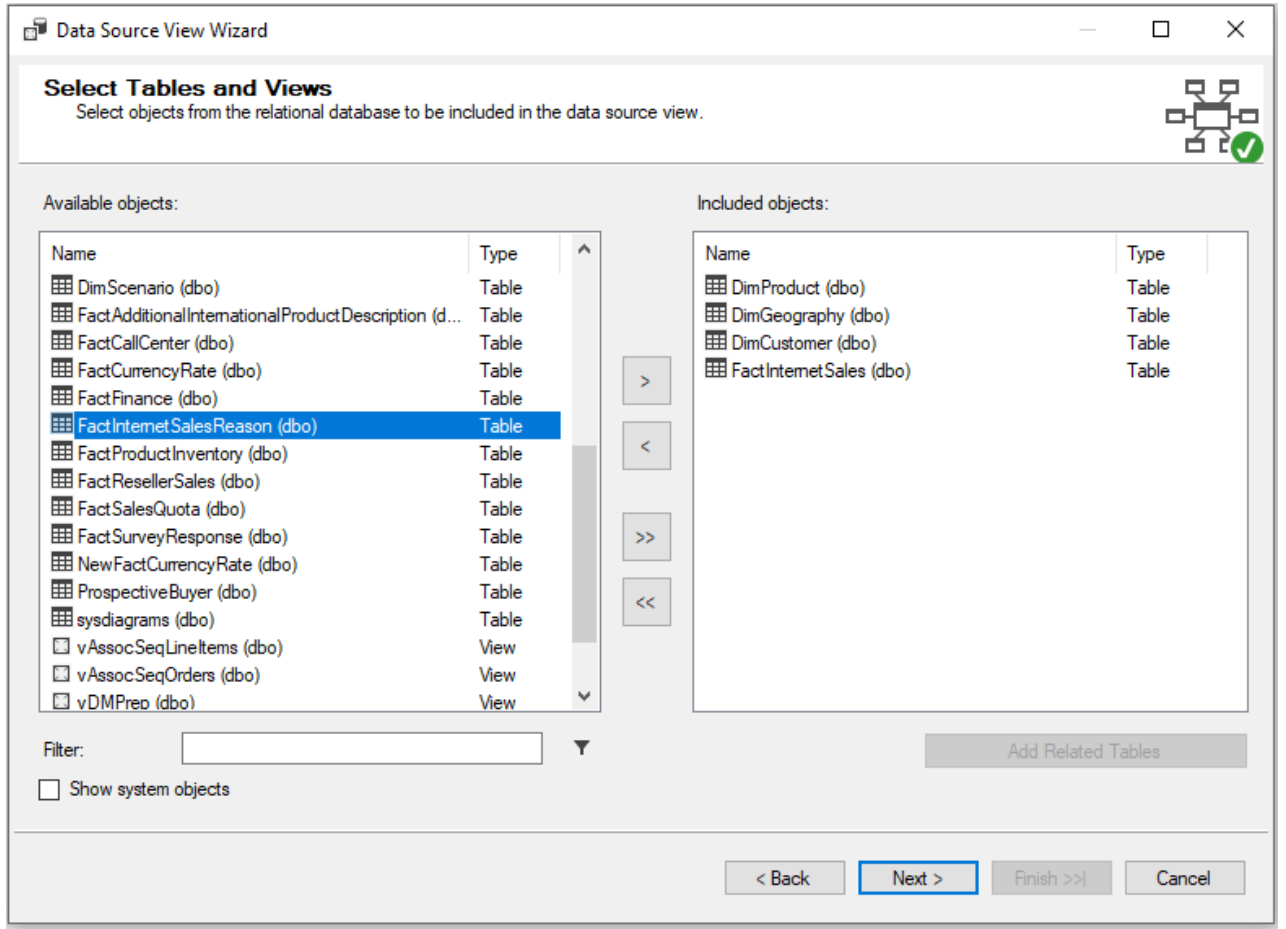
CREATING OLAP CUBES

- To create an OLAP cube, you will need to create a Business Intelligence project in Microsoft Visual Studio. Install the SSDT tool and you can create this type of project in Visual Studio.

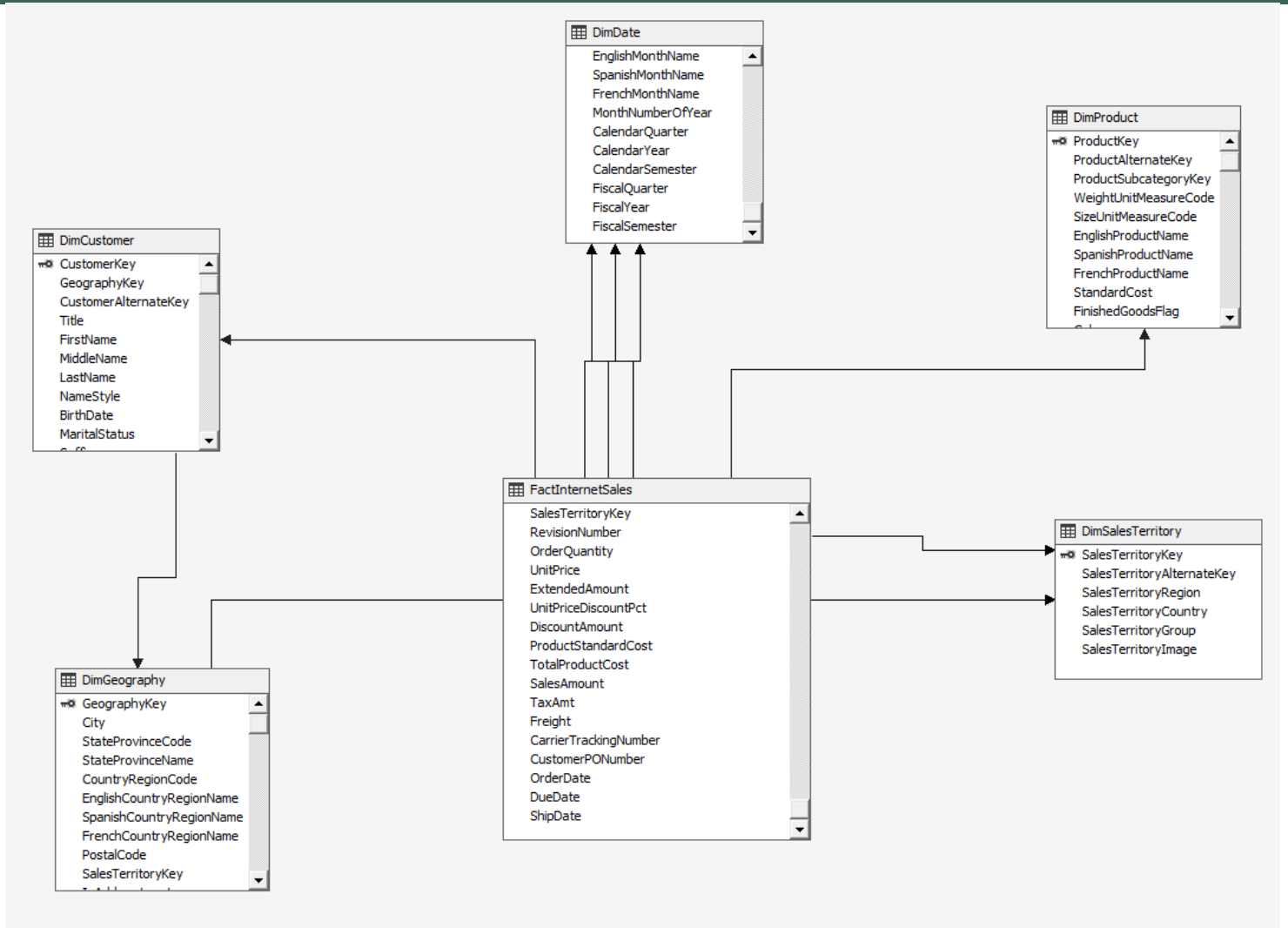


CREATING OLAP CUBES

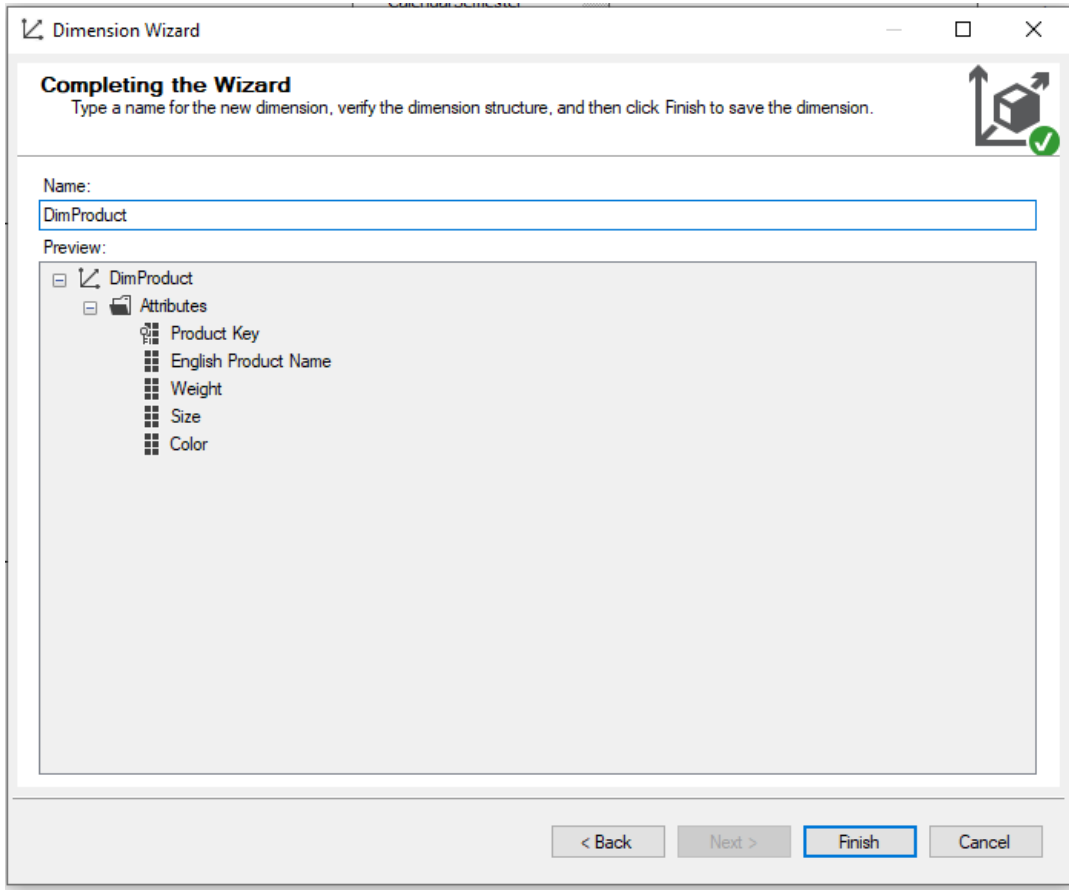
- In the data source view, select the desired tables from which the OLAP cube will be built.



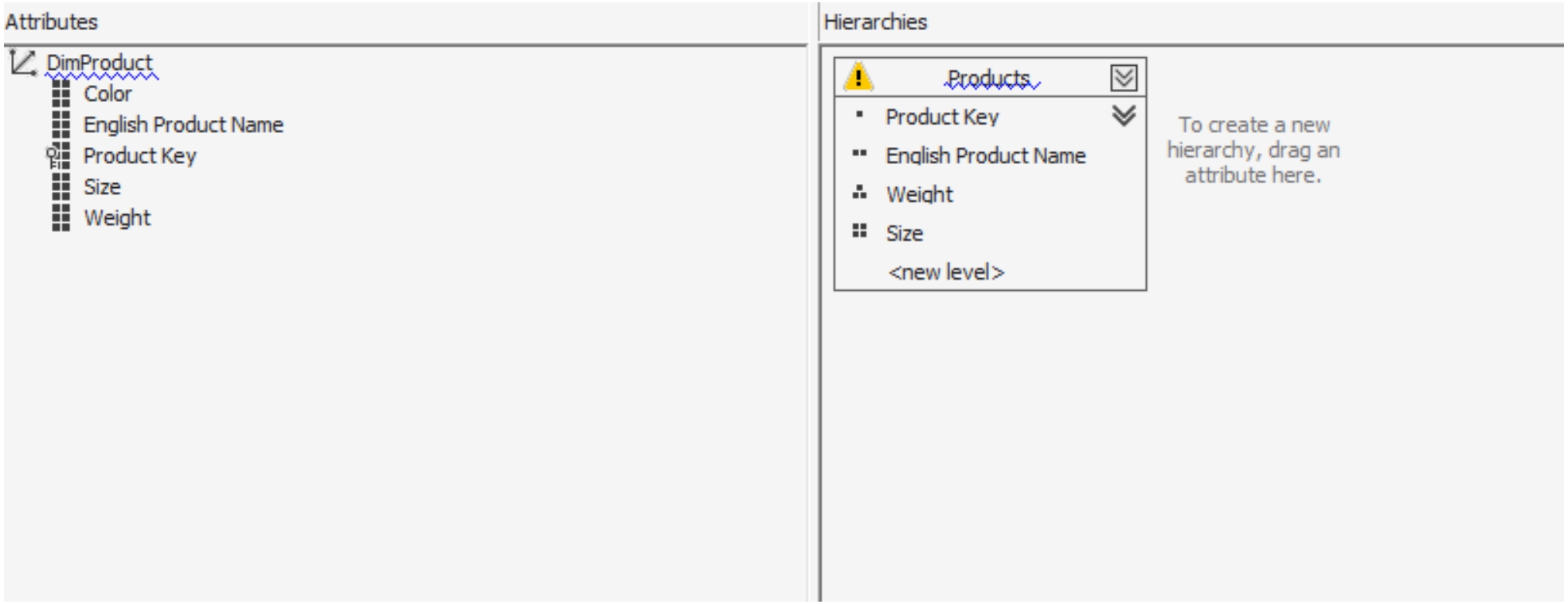
CREATING OLAP CUBES



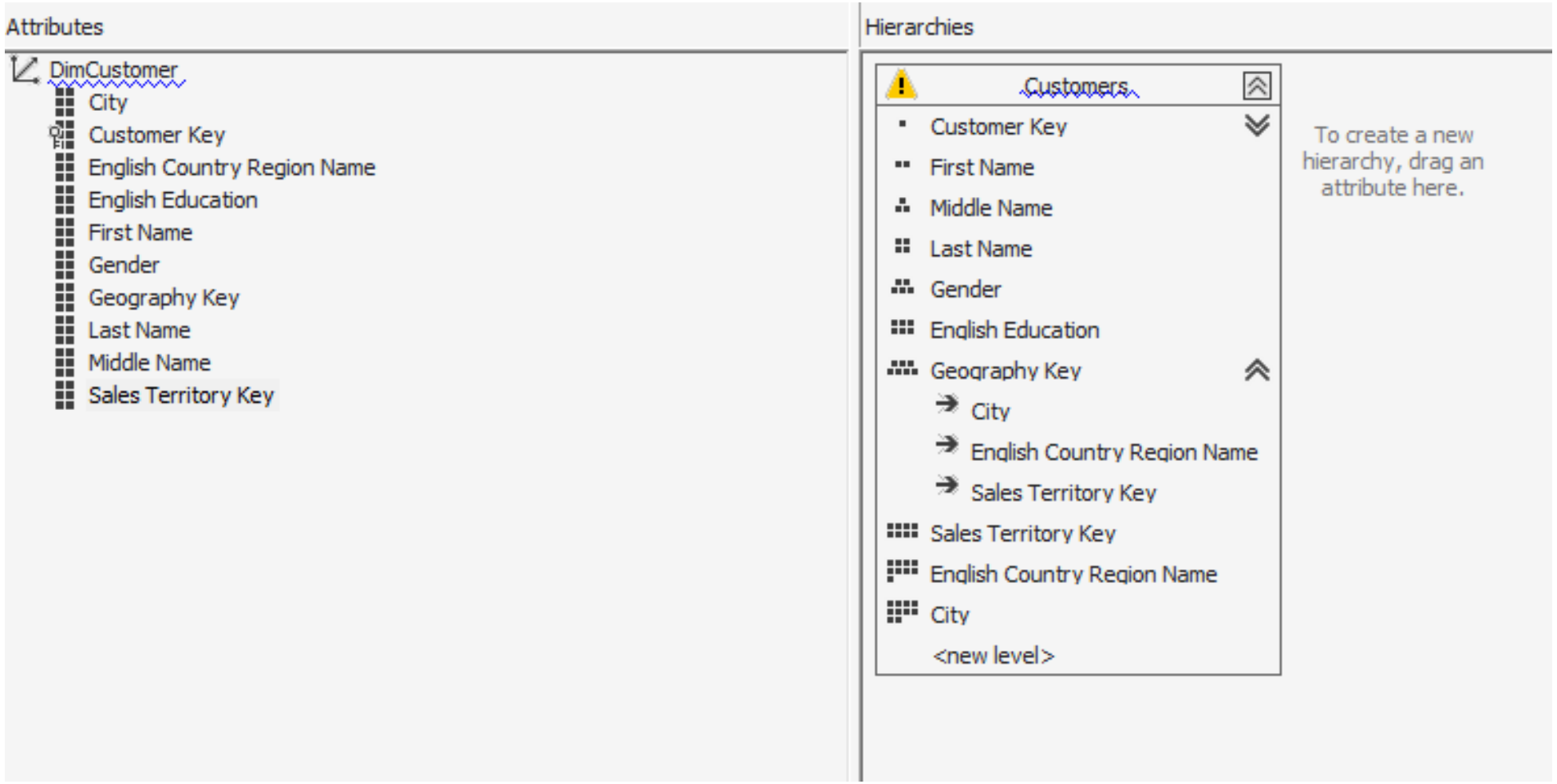
CREATE DIMENSIONS



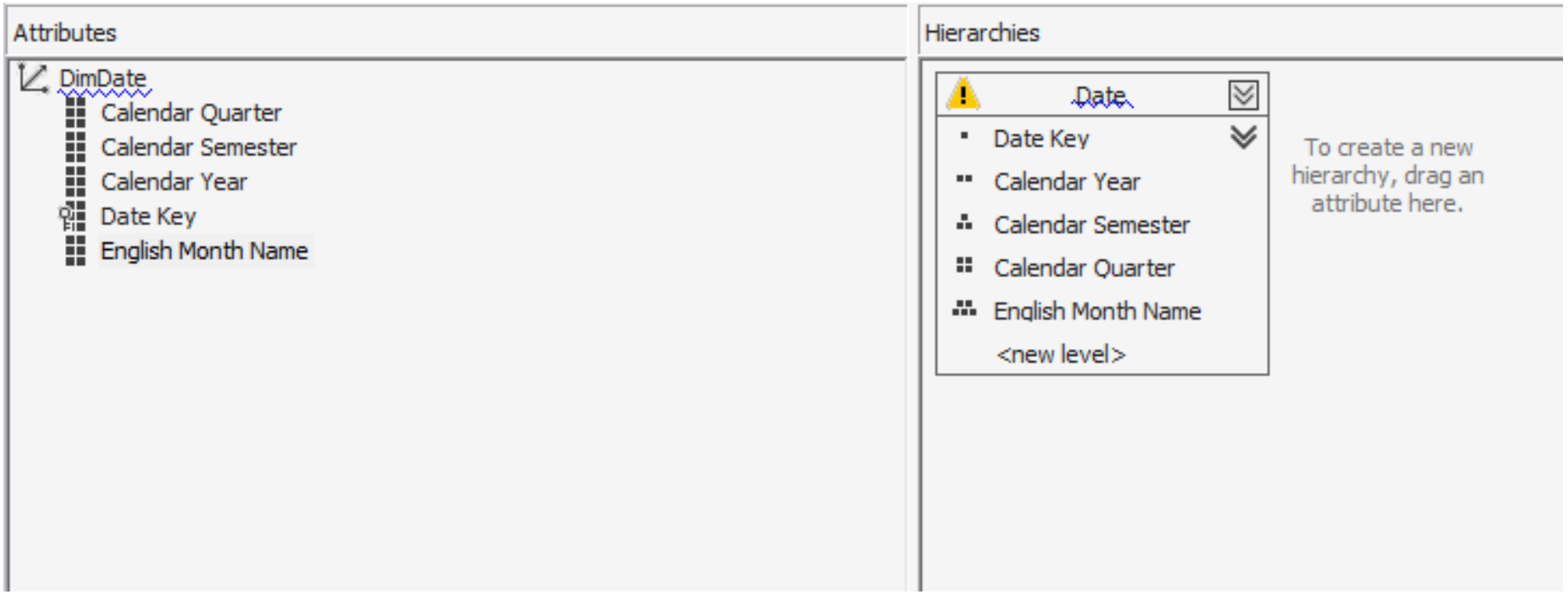
CREATE DIMENSIONS



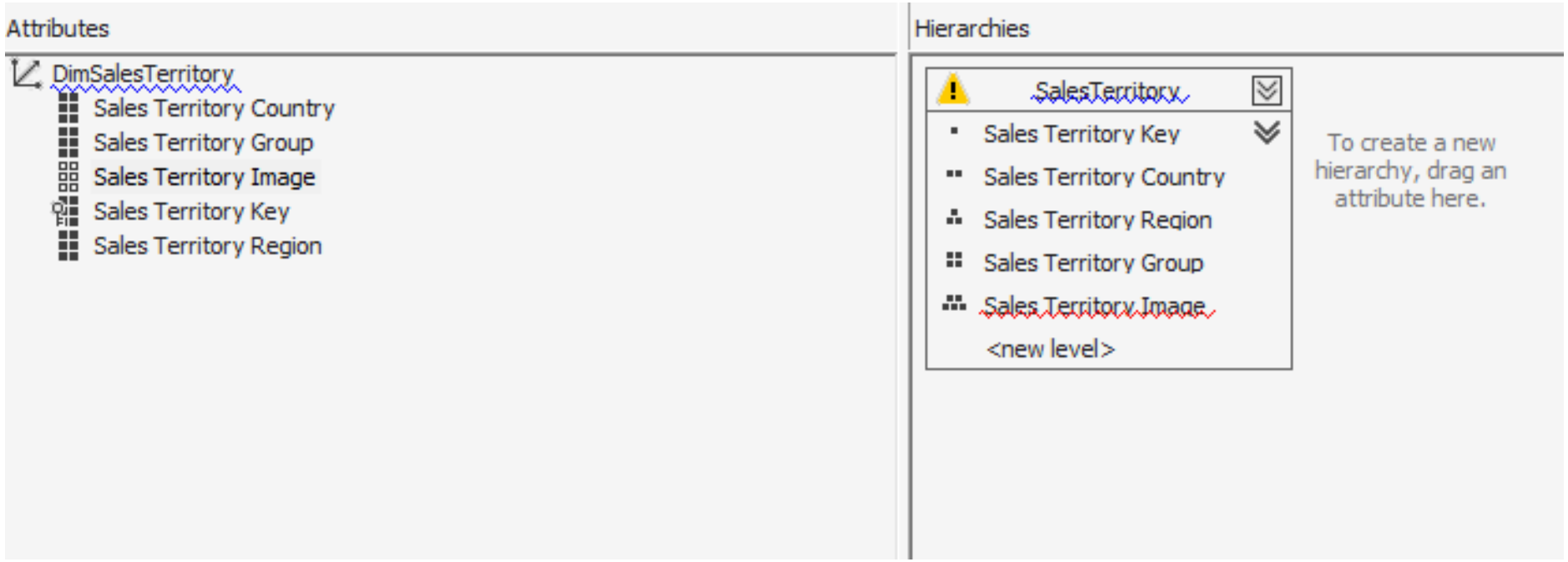
CREATE DIMENSIONS



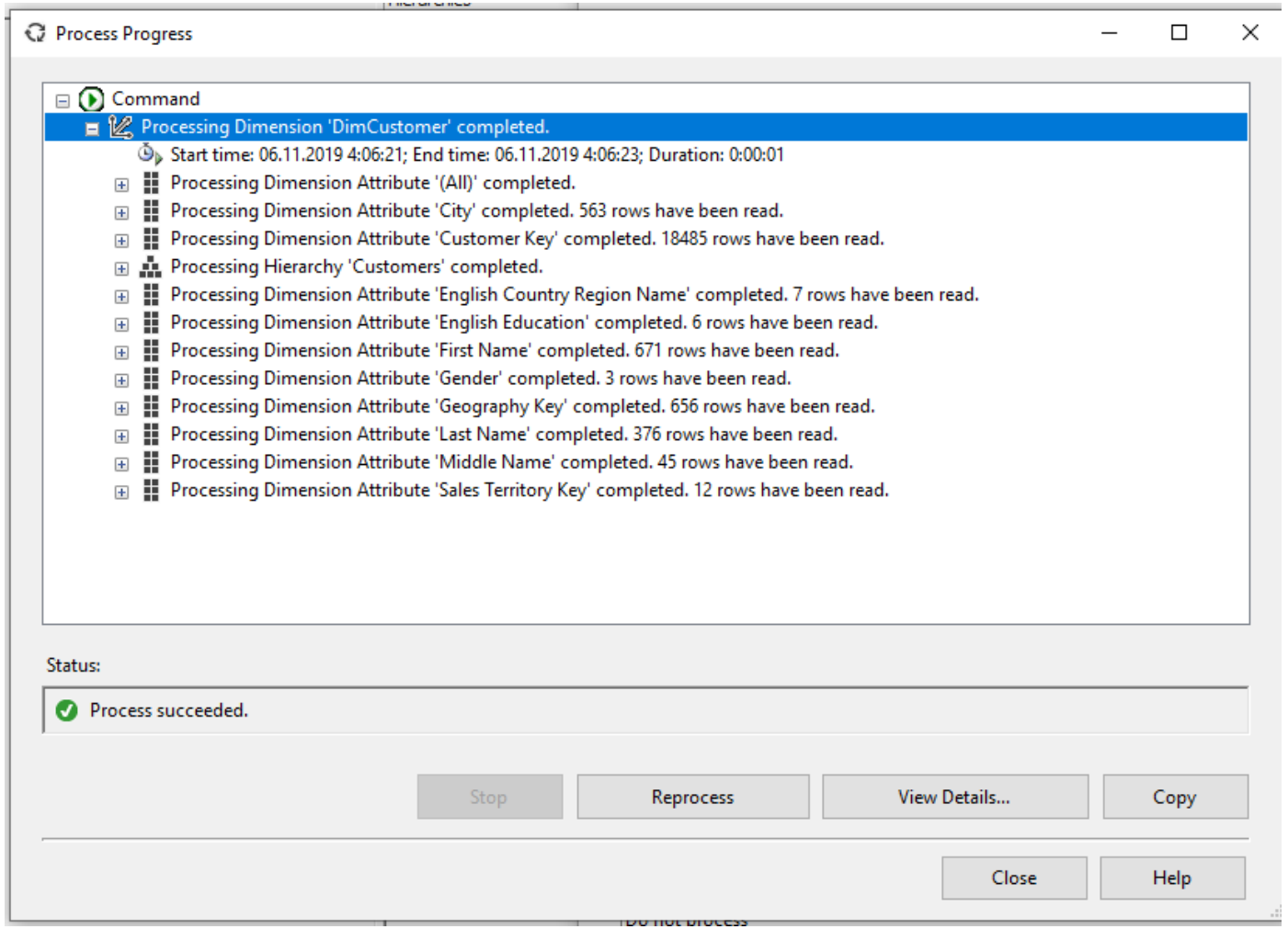
CREATE DIMENSIONS



CREATE DIMENSIONS

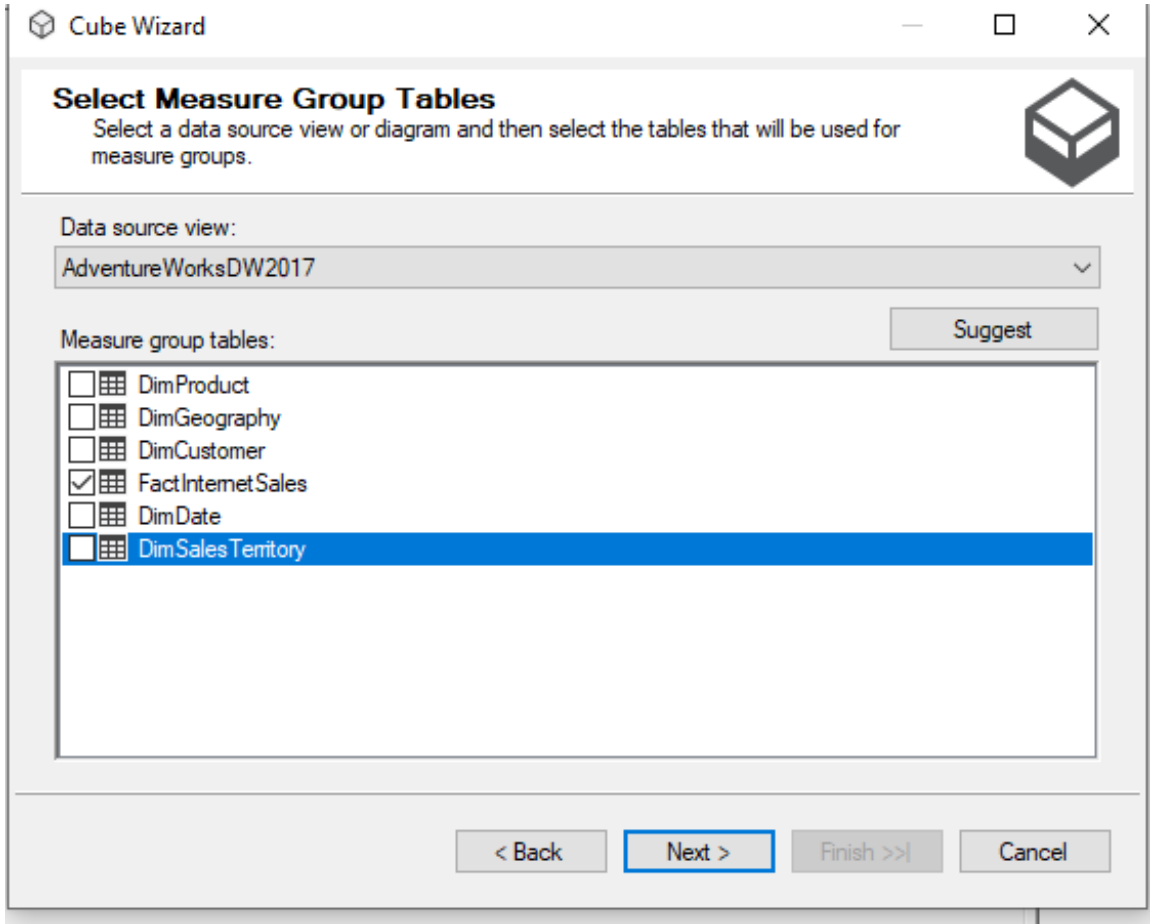


DEPLOY DIMENSIONS

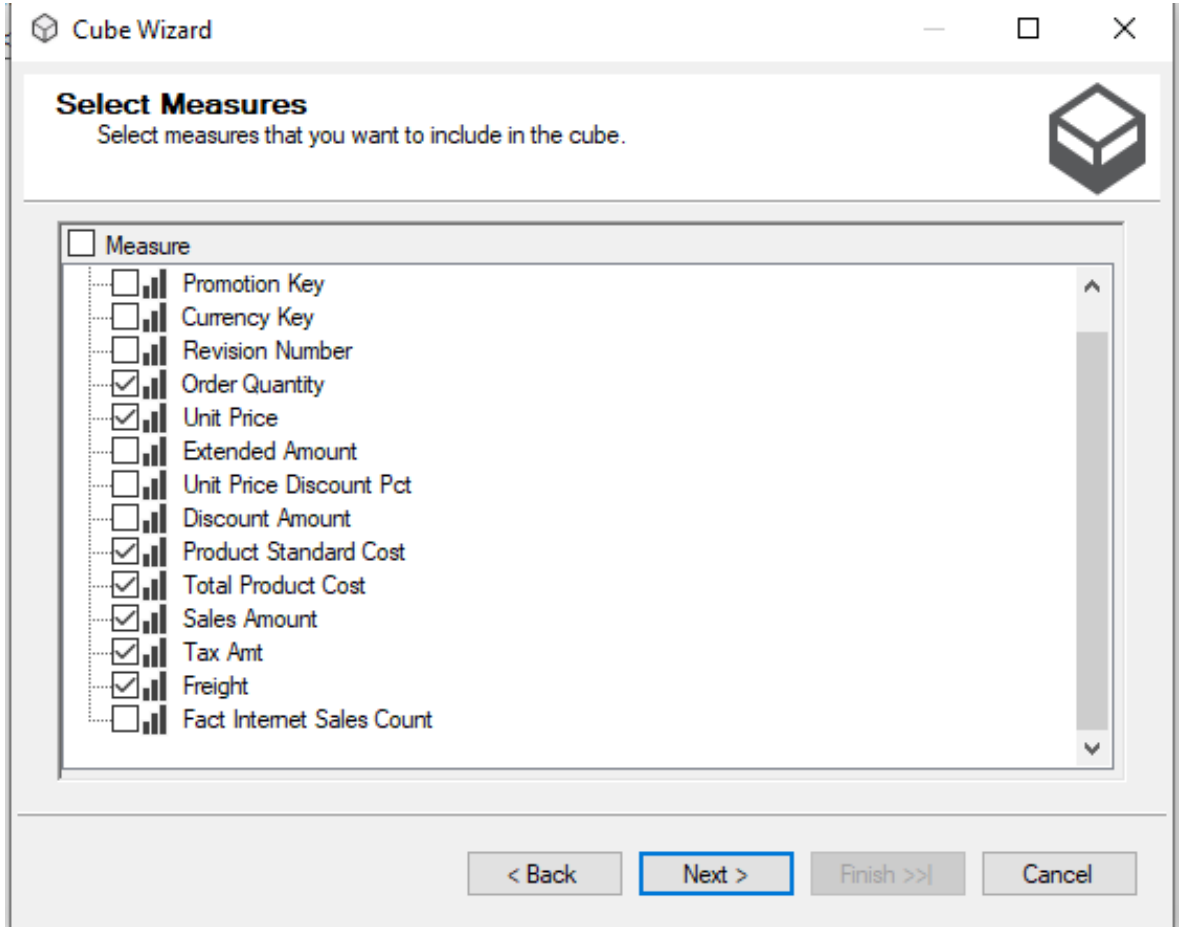


CREATING THE CUBE

- After deploying dimensions, it is necessary to create the cube itself. We choose measures from one table

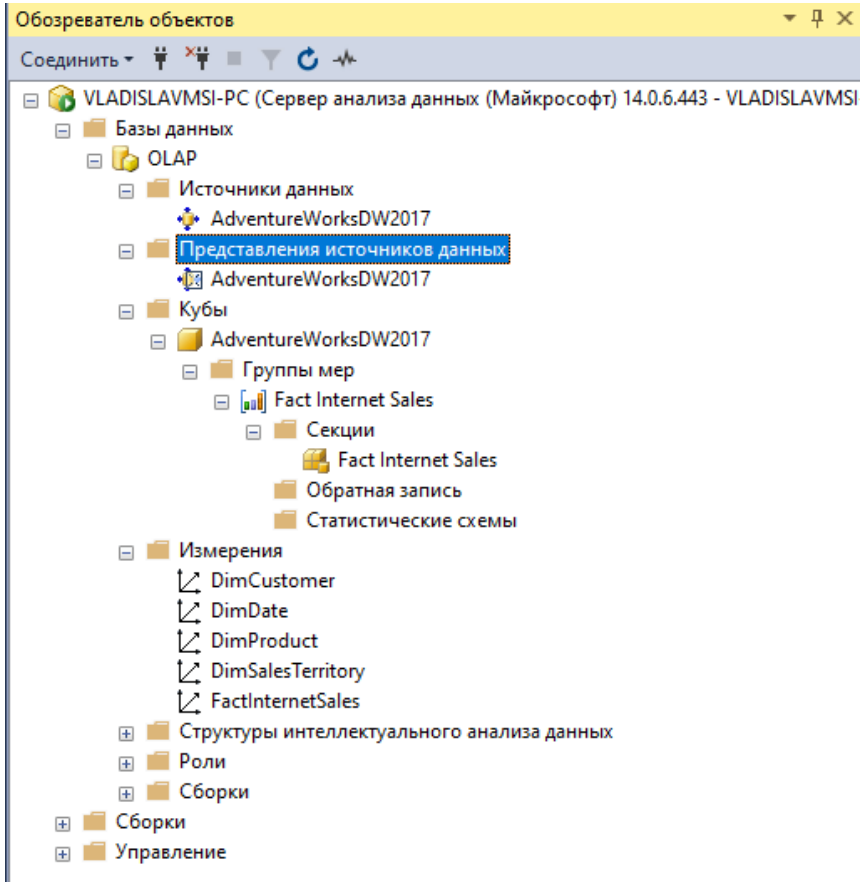


CREATING THE CUBE



OLAP CUBE

- After the deployment of the cube, connect to Microsoft SQL Analysis Services

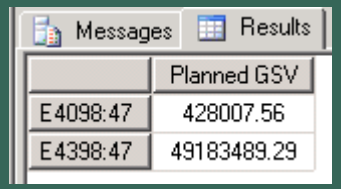


MDX DIMENSIONALITY

MDX QUERIES ARRANGE CUBE DIMENSIONS ON THE REPRESENTATION DIMENSIONS (HEREAFTER REFERRED AS 'AXIS' TO AVOID CONFUSION)

E.G. RETRIEVE PLANNED GSV MEASURE FOR ACCOUNTS E4098,E4398.

```
SELECT
        {
        [MEASURES].[PLANNED GSV]
       } ON 0,
       {
       [ACCOUNT].[ACCOUNT CODE].[E4098:47]
       ,[ACCOUNT].[ACCOUNT CODE].[E4398:47]
        }  ON 1
       FROM [CUBE REPORTING]
```



MDX QUERY – AXIS FRAMEWORK

MDX QUERIES ARRANGE CUBE DIMENSIONS ON THE REPRESENTATION DIMENSIONS (HEREAFTER REFERRED AS 'AXIS' TO AVOID CONFUSION)

E.G. RETRIEVE PLANNED GSV MEASURE FOR ACCOUNTS E4098,E4398.

```
SELECT
        {
        [MEASURES].[PLANNED GSV]
       } ON  COLUMNS,
       {
       [ACCOUNT].[ACCOUNT CODE].[E4098:47]
       ,[ACCOUNT].[ACCOUNT CODE].[E4398:47]
        } ON ROWS
       FROM [CUBE REPORTING]
```

MDX provides names for each axis (till 4)

0	Columns
1	Rows
2	Pages
3	Chapters
4	Sections

MDX QUERY – AXIS FRAMEWORK

MDX QUERIES PRIMARILY DEFINE AXIS'S

```
SELECT
        {
        SOMETHING
        } ON  AXIS(0),
        {
        SOMETHING ELSE
        } ON AXIS(1),
        FROM [CUBE NAME]
```

	E4098:47	E4398:47
Planned GSV	428007.56	49183489.29

```
SELECT
{
 [ACCOUNT].[ACCOUNT CODE].[E4098:47]
 ,[ACCOUNT].[ACCOUNT CODE].[E4398:47]
  }

 ON AXIS(0),
 {
[MEASURES].[PLANNED GSV]  }  ON AXIS(1)
FROM [CUBE REPORTING]
```

MDX QUERY – AXIS FRAMEWORK

MDX QUERIES PRIMARILY DEFINE AXIS'S

Something ?

Something = set or tuple

```
SELECT
        {
        SOMETHING
        } ON  AXIS(0),
        {
        SOMETHING ELSE
         } ON AXIS(1),
        FROM [CUBE NAME]
```

Also note the structure of the basic MDX query

MDX QUERY – TUPLE AND SETS

TUPLES

A *TUPLE* IS A COMBINATION OF MEMBERS FROM ONE OR MORE DIMENSIONS

-> NOT MORE THAN 1 MEMBER FROM A DIMENSION (SAME RULE AS CO-ORDINATE GEOMETRY) *

-> MANY WAYS ARE THERE TO SPECIFY A MEMBER.

() – for tuples
{} – for sets

([MEASURES].[PLANNED GSV] ,[TIME].[2010 HALFYEAR 1])

([MEASURES].[PLANNED GSV] ,[TIME].&[64])

SETS

A SET IS AN ORDERED COLLECTION OF TUPLES.

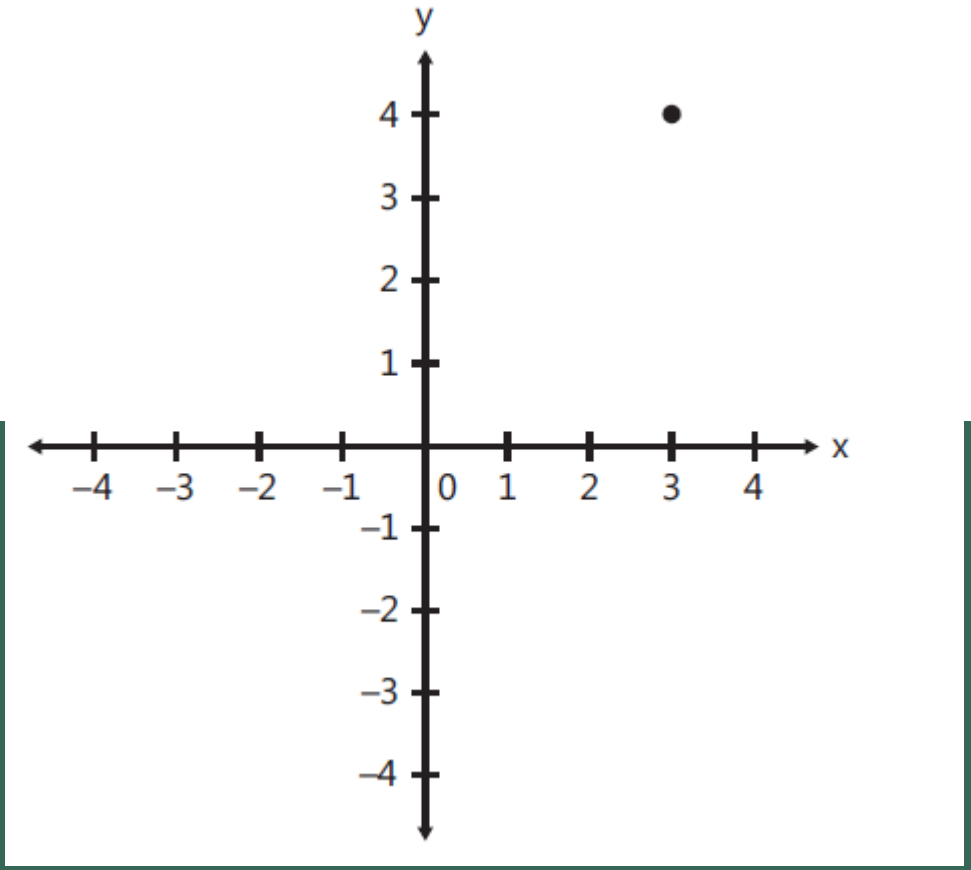
```
SELECT

{
 [ACCOUNT].[ACCOUNT CODE].[E4098:47]
 ,[ACCOUNT].[ACCOUNT CODE].[E4398:47]
  }

 ON AXIS(0),
```

```
 {
( [Measures].[Planned GSV] ,time.[2010 JAN])
,( [Measures].[Planned GSV] ,[Time].[2010 HalfYear 1]  )
  }

 on Axis(0)
```



Understanding tuples are key to thinking in MDX .

We will stop here till all tuple related queries are clarified.

Understanding tuple

Best analogy - coordinate geometry *

2D –space

Tuple of the form *(x_1,y_1) e.g. (3,4)*

Tuple like *(x_1,y_1 , y_2) or (x_1,x_2 , y_2) are invalid*

Now apply same concept to n- dimensional cube

What are valid tuples ?

(time.year.[2011] , Product.brand.[B1])

(time.year.[2011] , Product.brand.[B1] , time.year.[2010])

(time.year.[2011] , Product.brand.[B1], Geo.[India])

(time.year.[2011] , Product.brand.[B1], Product.brand.[B2])

* This analogy holds good except for hierarchies. Hierarchies in cube space can be considered as dimensions in Co-ordinate geometry

MDX QUERY – TUPLE AND SETS

SOMETHING TO REMEMBER

1. THIS IS A SET CONTAINING A SINGLE TUPLE

{

[ACCOUNT].[ACCOUNT CODE].[E4098:47]

}

2. THIS IS A SINGLE TUPLE

[ACCOUNT].[ACCOUNT CODE].[E4098:47]

BOTH ARE DIFFERENT THINGS CONCEPTUALLY AND PROGRAMMATICALLY (THOUGH SSAS OVERLOOKS IT).

MDX QUERY – TUPLE CONSTRUCTS

A CLOSER LOOK AT TUPLE

- ELEMENTS SEPARATED BY ' , '
- MULTI PART IDENTIFIER ' . '
- CAN BE OF THE FORM

DIMENSION.HIERARCHY.LEVEL.MEMBER

[TIME].[TIME HIERARCHY].[MONTH].&[421]

[TIME].[TIME HIERARCHY].[MONTH].[2010 JAN]

DIMENSION.MEMBER

[TIME].[2010 JAN]

- COLON ' : ' USED TO REPRESENT A SERIES OF MEMBERS

([TIME].[2010 JAN]:[TIME].[2010 DEC]) - ALL MONTHS FROM JAN 2010 TO DEC 2010

	E4398:47
2010 JAN	338733
2010 FEB	601503
2010 MAR	394195
2010 APR	437042
2010 MAY	578184
2010 JUN	715195
2010 JUL	707481
2010 AUG	1007624
2010 SEP	550235
2010 OCT	743799
2010 NOV	604130
2010 DEC	223204

```
SELECT
{[ACCOUNT].[E4398:47]}
 ON 0,
{([TIME].[2010 JAN]:[TIME].[2010 DEC]) }
ON 1
FROM [CUBE REPORTING]
```

MDX QUERY – THE CHILDREN FUNCTION

.CHILDREN

-- USED TO EXPRESS THE CHILDREN OF A MEMBER

E.G. AS PER TIME HIERARCHY – MONTH IS THE CHILD OF QUARTER

```
SELECT { TIME.[2010 QUARTER 1].CHILDREN }ON 0,
{[ACCOUNT].[E4398:47]} ON 1
FROM [CUBE REPORTING]
```

	E4398:47
2010 JAN	338733
2010 FEB	601503
2010 MAR	394195

MDX QUERY – THE DESCENDATNS FUNCTION

DESCENDANTS (*MEMBER* [, [*LEVEL*] [, *FLAG*]])

-- USED TO EXPRESS THE DESCENDANT OF A *MEMBER* AT A *LEVEL*

FLAG ALLOWS DISPLAY

```
SELECT {[ACCOUNT].[E4398:47]}
 ON 0,
{
DESCENDANTS(TIME.[2010],MONTH,SELF)
} ON 1
FROM [CUBE REPORTING]

SELECT {[ACCOUNT].[E4398:47]}
 ON 0,
{
DESCENDANTS(TIME.[2010],MONTH,SELF_AND_BEFORE)
} ON 1
FROM [CUBE REPORTING]
```

	E4398:47
2010 JAN	338733
2010 FEB	601503
2010 MAR	394195
2010 APR	437042
2010 MAY	578184
2010 JUN	715195
2010 JUL	707481
2010 AUG	1007624
2010 SEP	550235
2010 OCT	743799
2010 NOV	604130
2010 DEC	223204

	E4398:47
2010	6901325
2010 HalfYear 1	3064852
2010 Quarter 1	1334431
2010 JAN	338733
2010 FEB	601503
2010 MAR	394195
2010 Quarter 2	1730421
2010 APR	437042
2010 MAY	578184
2010 JUN	715195
2010 HalfYear 2	3836473
2010 Quarter 3	2265340
2010 JUL	707481
2010 AUG	1007624
2010 SEP	550235
2010 Quarter 4	1571133
2010 OCT	743799
2010 NOV	604130
2010 DEC	223204

ASSIGNMENT - I

ASSUME THE FOLLOWING DIMENSIONS :

TIME : YEAR <- QUARTER <- MONTH <- DAY

PRODUCT : DOLLAR SALES, UNIT SALES

GEO : COUNTRY <- STATE <- CITY

CROSS JOINS

CONCEPT :

TWO SETS - A ,B

A = {1,2,3} B = {X,Y}

CROSS JOIN AXB

AXB = { (1,X), (2,X), (3,X), (1,Y), (2,Y), (3,Y),}

PLANNED GSV FOR 2 ACCOUNTS FOR 3 MONTHS --

```
SELECT
{
(ACCOUNT.[ACCOUNT CODE].[E4098:47]),
(ACCOUNT.[ACCOUNT CODE].[E4400:47])
}
*  -- CROSS JOIN
{
TIME.[2010 JAN] : TIME.[2010 MAR]
}
 ON 0,
{
[MEASURES].[PLANNED GSV]
}ON 1
FROM
[CUBE REPORTING]
```

	E4098:47	E4098:47	E4098:47	E4400:47	E4400:47	E4400:47
	2010 JAN	2010 FEB	2010 MAR	2010 JAN	2010 FEB	2010 MAR
Planned GSV	31594.08	38569.92	38481.04	1481540	3209700	2112100

FILTER

CONCEPT :

FILTER (SET , EXPRESSION)

ACCOUNTS WHICH HAVE MORE THAN GSV

.members gives all members of that level

```
SELECT
FILTER
(
(ACCOUNT.[ACCOUNT CODE].MEMBERS),
[MEASURES].[PLANNED GSV] > 10000000
)
 ON 0,
{
[MEASURES].[PLANNED GSV]
}ON 1
FROM
[CUBE REPORTING]
```

	All	DTTot3:47	E4398:47	E4400:47	E4401:47	E4414:47	E4982:47
Planned GSV	886166048.7521	13611790	49183489.29	11623040	637964831.993	35195728.33	76271218.992

FILTER

WHAT IS THE MEANING OF THIS ?

```
ELECT
FILTER
(
(ACCOUNT.[ACCOUNT CODE].MEMBERS),
([MEASURES].[PLANNED GSV] ) > 10000000
)*  -- CROSS JOIN
{
TIME.[2010 JAN] :TIME.[2010 MAR]
}
 ON 0,
{
[MEASURES].[PLANNED GSV]
}ON 1
FROM
[CUBE REPORTING]
```

	All	All	All	DTTot3:47	DTTot3:47	DTTot3:47	E4398:47	E4398:47	E4398:47	E4400:47	E4400:47
	2010 JAN	2010 FEB	2010 MAR	2010 JAN	2010 FEB	2010 MAR	2010 JAN	2010 FEB	2010 MAR	2010 JAN	2010 FEB
Planned GSV	59230786.49708	83152066.09	70867123.5314	(null)	(null)	(null)	3593022.88	5276412.38	2256812.5	1481540	3209700

FILTER

AND THIS ?

```
SELECT
FILTER
(
(ACCOUNT.[ACCOUNT CODE].MEMBERS),
([MEASURES].[PLANNED GSV] ,TIME.[2010 JAN]) > 10000000
)*  -- CROSS JOIN
{
TIME.[2010 JAN] :TIME.[2010 MAR]
}
 ON 0,
{
[MEASURES].[PLANNED GSV]
}ON 1
FROM
[CUBE REPORTING]
```

Tuple reference is one of the powerful concepts in MDX

	All	All	All	E4401:47	E4401:47	E4401:47
	2010 JAN	2010 FEB	2010 MAR	2010 JAN	2010 FEB	2010 MAR
Planned GSV	59230786.49708	83152066.09	70867123.5314	49120998.089	71687757.826	61112877.531

ORDER

CONCEPT :

ORDER (SET1, EXPRESSION [,ASC | DESC | BASC | BDESC])

ACCOUNTS ORDERED BY GSV

```
SELECT
NON EMPTY (
ORDER
(
(ACCOUNT.[ACCOUNT CODE].MEMBERS),
([MEASURES].[PLANNED GSV] )
)
)
 ON 0,
{
[MEASURES].[PLANNED GSV]
}ON 1
FROM
[CUBE REPORTING]
```

	All	E4616:47	E4612:47	E4598:47	E4602:47	E4547:47	E4626:47	E4546:47	E4533:47	E4627:47	E4548:47	E4619:47
Planned GSV	886166048.7521	13408	20880	23360	40493.3	96924	115602.5	126750	144464.7	148253.5	183610	202138

ACCOUNTS ORDERED BY ???

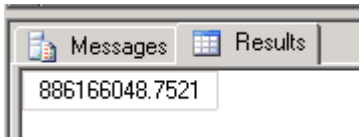
```
SELECT
NON EMPTY (
ORDER
(
(ACCOUNT.[ACCOUNT CODE].MEMBERS),
([MEASURES].[PLANNED GSV],TIME.[2011 JAN] )
)
)
 ON 0,
{
[MEASURES].[PLANNED GSV]
}ON 1
FROM
[CUBE REPORTING]
```

	All	DTTot3:47	E4098:47	E4398:47	E4400:47	E4401:47	E4402:47	E4407:47	E4413:47	E4414:47
Planned GSV	886166048.7521	13611790	428007.56	49183489.29	11623040	637964831.993	6288527.17226	1974200	8884200	35195728.33

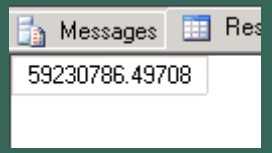
THE WHERE CLAUSE

Tuple instead of expression used

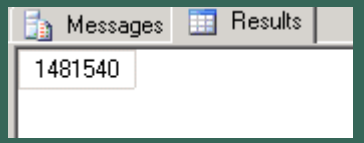
```
SELECT
FROM
[CUBE REPORTING]
WHERE
([MEASURES].[PLANNED GSV] )
```




```
SELECT
FROM
[CUBE REPORTING]
WHERE
([MEASURES].[PLANNED GSV],TIME.[2010 JAN] )
```




```
SELECT
FROM
[CUBE REPORTING]
WHERE
([MEASURES].[PLANNED GSV],TIME.[2010 JAN] ,ACCOUNT.[ACCOUNT CODE].[E4400:47] )
```



NAMED SETS : EASE OF REFERENCE

```
WITH SET [GREAT ACCOUNTS] AS
{
[ACCOUNT].[ACCOUNT CODE].[E1373:47],
[ACCOUNT].[ACCOUNT CODE].[E40301:47]
}

SELECT
{
[MEASURES].[PLANNED GSV]
}
ON 0,
 NON EMPTY
{
[GREAT ACCOUNTS]
} ON 1
FROM [CUBE REPORTING]
```

	Planned GSV
E1373:47	31173548.79884
E40301:47	41130.25381

ALSO POSSIBLE TO CREATE PERSISTENT NAMED SETS

```
CREATE SET [CUBE REPORTING].[TEST ACCOUNTS] AS
{
[ACCOUNT].[ACCOUNT CODE].[E1373:47],
[ACCOUNT].[ACCOUNT CODE].[E40301:47]
}
```

CALCULATED MEMBERS : THE POWER OF MDX !!!!

1. SIMPLE CALCULATED MEMBERS

FIND THE AVERAGE SALES PRICE (I.E. TOTAL DOLLAR SALES / NUMBER OF UNITS SOLD) FOR THE QUARTERS 2005 Q1 AND Q2.

Simple Division used to calculate a new measure

```
WITH
MEMBER [MEASURES].[AVG SALES PRICE] AS
[MEASURES].[DOLLAR SALES] / [MEASURES].[UNIT SALES]

SELECT
{ [MEASURES].[DOLLAR SALES]
[MEASURES].[UNIT SALES]
, [MEASURES].[AVG SALES PRICE] }
ON COLUMNS,
{ [TIME].[Q1, 2005]
, [TIME].[Q2, 2005]
}
ON ROWS
FROM SALES
```

	Dollar Sales	Unit Sales	Average Sales price
Q1, 2005	100	5	20
Q2, 2005	120	8	15

2. CALCULATED MEMBERS OF MEDIUM COMPLEXITY

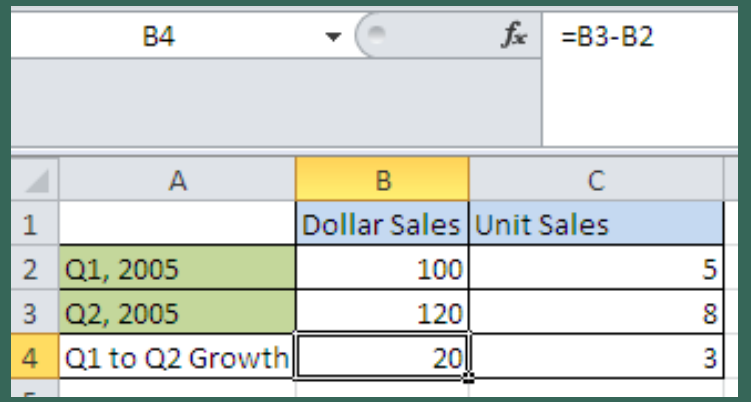
FIND THE QUARTER ON QUARTER GROWTH FOR DOLLAR SALES AND UNIT SALES FOR THE QUARTERS 2005 Q2.

GROWTH IN DOLLAR SALES = 2005 Q2 DOLLAR SALES - 2005 Q1 DOLLAR SALES

GROWTH IN UNIT SALES = 2005 Q2 UNIT SALES - 2005 Q1 UNIT SALES

```
WITH
MEMBER [TIME].[Q1 TO Q2 GROWTH] AS
[TIME].[Q2, 2005] - [TIME].[Q1, 2005]
SELECT
{ [MEASURES].[DOLLAR SALES]
, [MEASURES].[UNIT SALES] }
ON COLUMNS,
{ [TIME].[Q1, 2005]
, [TIME].[Q2, 2005]
, [TIME].[Q1 TO Q2 GROWTH] }
ON ROWS
FROM SALES
WHERE ([CUSTOMER].[MA])
```

How does this take care of both subtractions ?

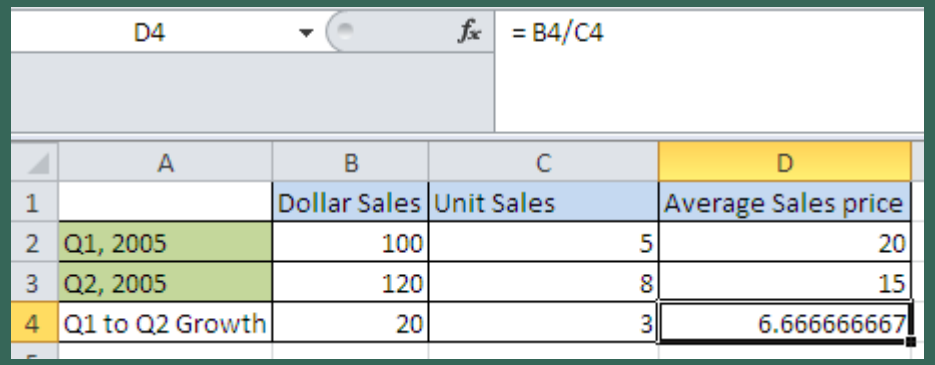


CALCULATED MEMBERS : THE POWER OF MDX !!!!

PRECEDENCE RESOLUTIONS

COMBINING PREVIOUS TWO PROBLEMS ,WRITE MDX TO CALCULATE Q1 TO Q2 GROWTH IN AVERAGE SALES PRICES

```
WITH
MEMBER [MEASURES].[AVG SALES PRICE] AS
[MEASURES].[DOLLAR SALES] / [MEASURES].[UNIT SALES]
MEMBER [TIME].[Q1 TO Q2 GROWTH] AS
[TIME].[Q2, 2005] - [TIME].[Q1, 2005]
SELECT
{ [MEASURES].[DOLLAR SALES], [MEASURES].[UNIT SALES],
[MEASURES].[AVG SALES PRICE]
}
ON COLUMNS,
{ [TIME].[Q1, 2005], [TIME].[Q2, 2005], [TIME].[Q1 TO Q2 GROWTH] }
ON ROWS
FROM [SALES]
```



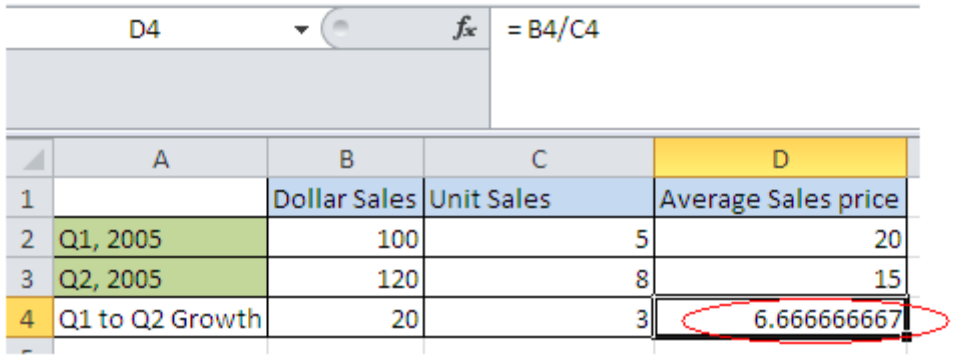
	A	B	C	D
1		Dollar Sales	Unit Sales	Average Sales price
2	Q1, 2005	100	5	20
3	Q2, 2005	120	8	15
4	Q1 to Q2 Growth	20	3	6.666666667

CALCULATED MEMBERS : THE POWER OF MDX !!!!

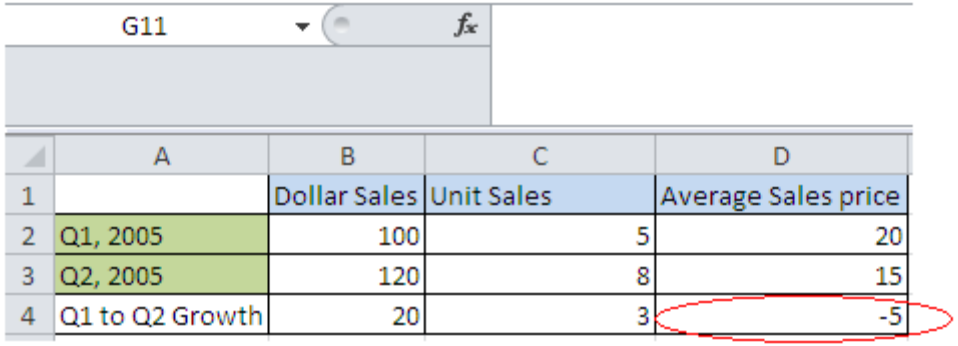
PRECEDENCE RESOLUTIONS

COMBINING PREVIOUS TWO PROBLEMS ,WRITE MDX TO CALCULATE Q1 TO Q2 GROWTH IN AVERAGE SALES PRICES

```
WITH
MEMBER [MEASURES].[AVG SALES PRICE] AS
[MEASURES].[DOLLAR SALES] / [MEASURES].[UNIT SALES],
SOLVE_ORDER = 0
MEMBER [TIME].[Q1 TO Q2 GROWTH] AS
[TIME].[Q2, 2005] - [TIME].[Q1, 2005],
SOLVE_ORDER = 1
SELECT
{ [MEASURES].[DOLLAR SALES], [MEASURES].[UNIT SALES],
[MEASURES].[AVG SALES PRICE]
}
ON COLUMNS,
{ [TIME].[Q1, 2005], [TIME].[Q2, 2005], [TIME].[Q1 TO Q2 GROWTH] }
ON ROWS
FROM [SALES]
```



	A	B	C	D
1		Dollar Sales	Unit Sales	Average Sales price
2	Q1, 2005	100	5	20
3	Q2, 2005	120	8	15
4	Q1 to Q2 Growth	20	3	6.666666667



	A	B	C	D
1		Dollar Sales	Unit Sales	Average Sales price
2	Q1, 2005	100	5	20
3	Q2, 2005	120	8	15
4	Q1 to Q2 Growth	20	3	-5

CALCULATED MEMBERS : THE POWER OF MDX !!!!

WRITE AN MDX TO GET THE FOLLOWING RESULT

	Sale Amount	Total Cost	Profit	Percentage Margin
Actual				
Planned				
Amount of Variance				
Percentage of Variance				

MEASURES -> MEASURES.[SALES AMOUNT], MEASURES.[TOTAL COST]

DIMENSIONS -> PHASE.ACTUAL , PHASE.PLANNED

USE NORMAL FORMULAS FOR PROFIT , PERCENTAGE MARGIN , AMOUNT OF VARIANCE, PERCENTAGE OF VARIANCE.