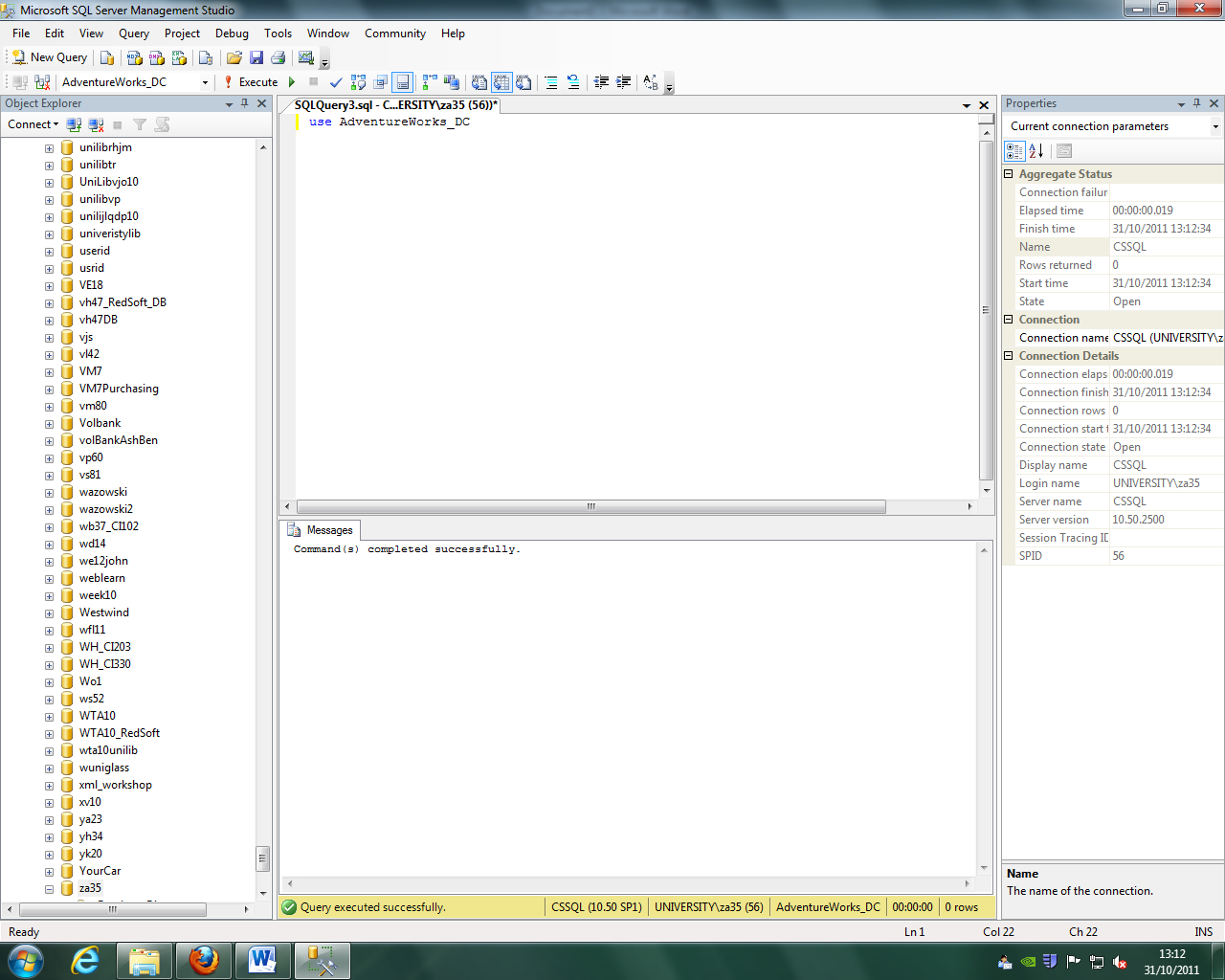
Week 5: Joins, keys and views

**Ensure that you are using the AdventureWorks\_DC**

To enter the AdventureWorks\_DC you do the following:

1. Start a new query
2. Type: use AdventureWorks\_DC
3. Then you Execute
4. After you execute, you will be connected to the AdventureWorks\_DC database. After this is achieved, you then can delete the command “use AdventureWorks\_DC”



**JOINGING TABLES**

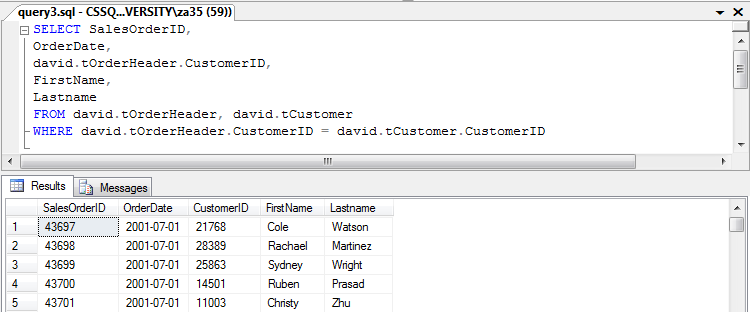
There are two different ways of joining tables, (two different ways of writing the syntax for doing a joining command). They are:

1. WHERE

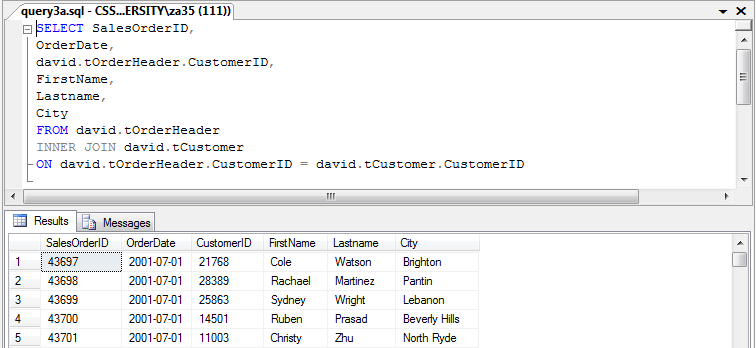
2. INNER JOIN

(Run the join sql in query3 AND query3a using AdventureWorks\_DC) as done below, then you will demonstrate the two different syntax:

**Running query 3** join tables using “WHERE” syntax

(OBS! WHERE is the old way of doing joint queries. The INNER JOIN is the current preferrd industry standard syntax ) 

**Running query 3 a** join tables using “INNER JOIN” syntax (OBS! The INNER JOIN is the current preferrd industry standard syntax )



**Syntax explained:**

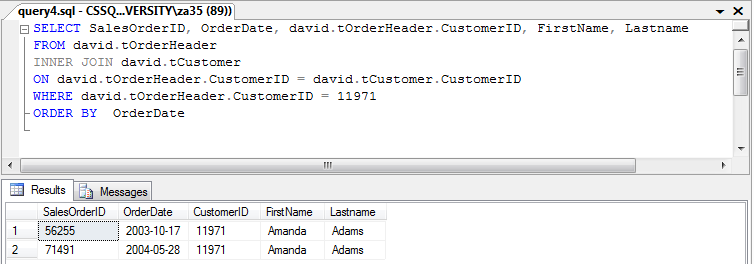
1. You start a new query
2. You type the attributes (columns) you want to have/joined with the other table ( in this case: SalesOrderID,OrderDate,david.tOrderHeader.CustomerID,FirstName,Lastname,City
3. You specify form which table the columns come from? In this case: FROM david.tOrderHeader
4. Then you type the name of **the other table** which you want it to join together with the above requested columns. In this case: INNER JOIN david.tCustomer
5. Now if you refer to step 2., you will notice that for CustomerID, the name of the table (david.tOrderHeader.) preceded the ‘column name’: (david.tOrderHeader.CustomerID). The reason you do this is because you have to tell SQL where from you want to retrieve the CustomerID data. If you don’t specify, it will not be able to do the join, as CustomerID exists in both david.tOrderHeader AND david.tCustomer tables. If you don’t specify where from the CustomerID should come from, you will get an ERROR message saying: (Ambiguous column name 'CustomerID'.)
6. Now you instruct SQL which data (column) you want to join. The syntax to do this is:

ON david.tOrderHeader.CustomerID = david.tCustomer.CustomerID

(OBS! Please notice that CustomerID in tCustomer is a uinique key, (not necessarily a primary key, but must be unique. Otherwise the join will not work)

1. Now execute, and you should have the above image (table).

**Running query 4** below is a good demonstration of how to use WHERE and ORDER BY

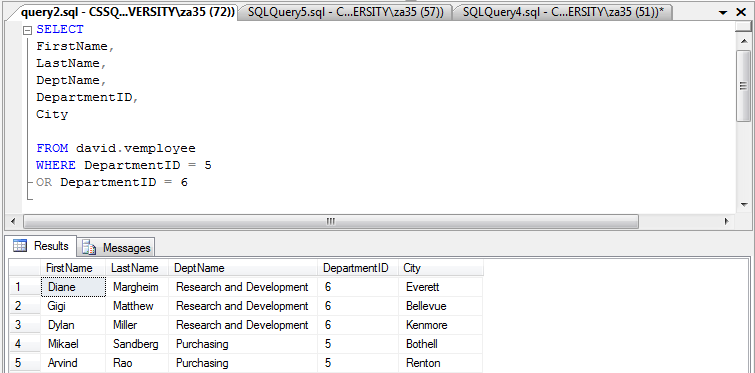
****

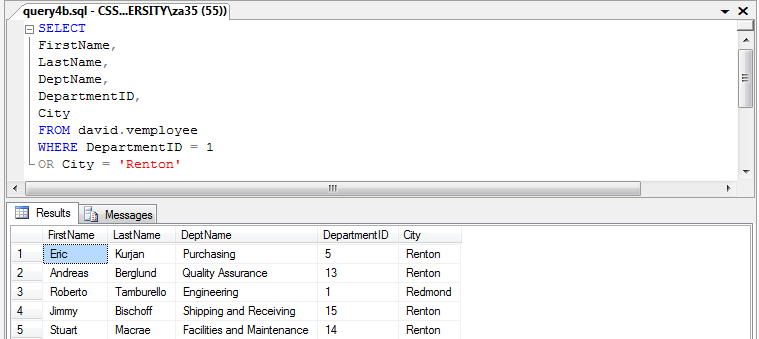
Week 6: Queries using AND, OR, BETWEEN and IN

**Using OR:**

You usually use ‘OR’ if you want to open more than one data, like in below example. You opened (DepartmentID = 5) using the ‘WHERE’, and then you also opened (DepartmentID = 6) using ‘OR’

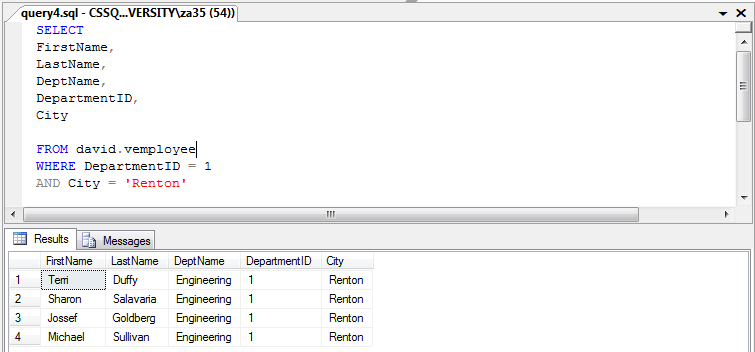
**(OBS! If you want to open more than one data [DepartmentID] then you should use ‘IN’)**

****

Lets do another **OR** example. Now we want to open data from (DepartmentID = 1) and city ‘Renton’. Again using ‘WHERE’ and ‘OR’:

**Using AND:**

You usually use ‘AND’ if you want to open data matching with certain other data. Like in the below example, we want to open data from (DepartmentID = 1) and then we only want to see people living in city ‘Renton’ working in Department 1. Anyone else not living in ‘Renton’ but working in (DepartmentID = 1) will not be shown on the table. You achive this by using ‘WHERE’ and ‘AND’:

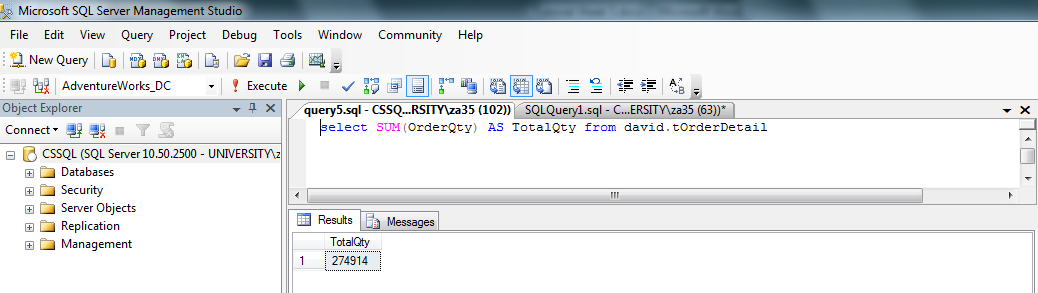
**(You can use ‘AND’ together with other commands like ‘WHERE’)**

Week 7: Aggregation, Group by and Totals

**Using SUM**

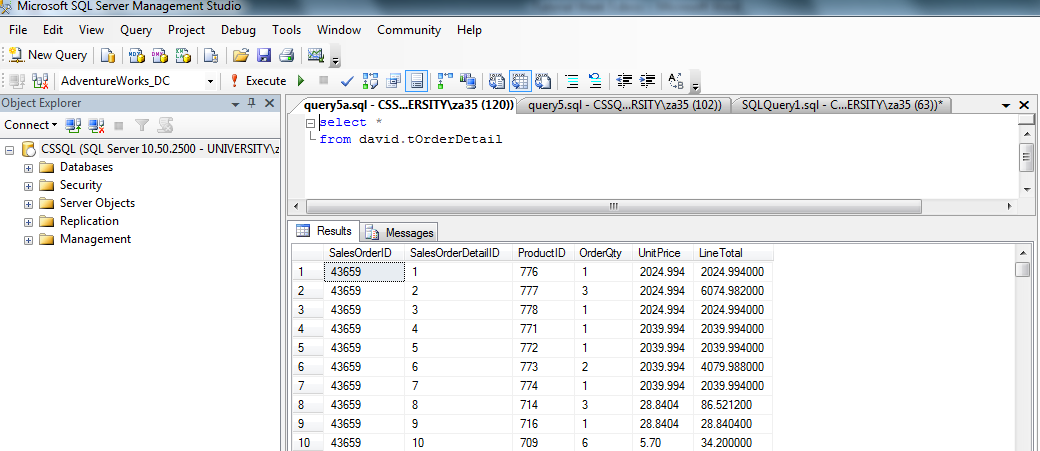
Using SUM allows you to work out the total sum of a certain coloumn. In this case, below example, we want to have the total numner of orders for OrderQty

The total order quantity is 274914

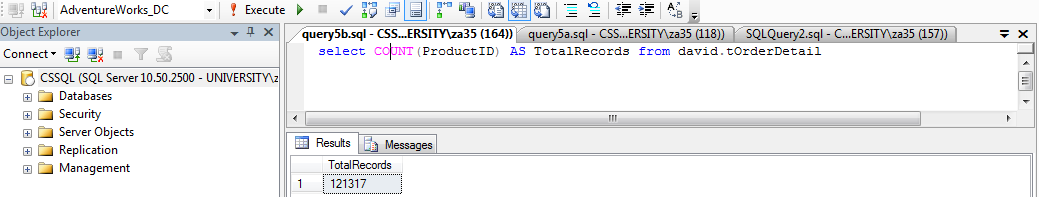


**Using COUNT**

You can use this method to work out the total number of records (rows) in a table (or for a specific column), as demonstrated in below example, we want to work out the total number of rows for ProductID



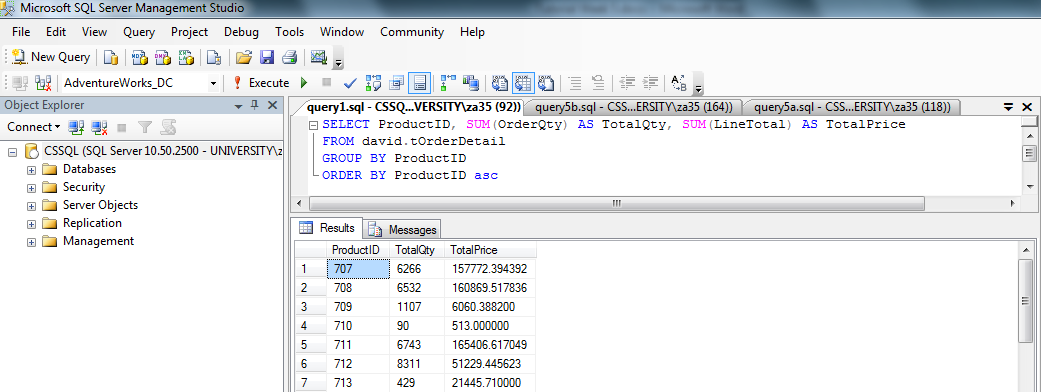
The total number of records (rows) for ProductID is 121317



**Using GROUP BY Syntax**

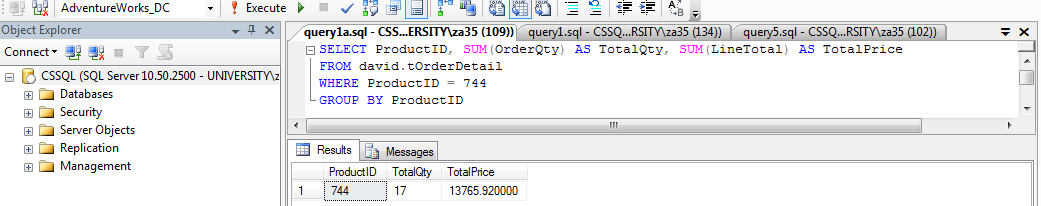
The GROUP BY keyword supports the generation of total, averages and basic counting based on groups from the data.

Example below, we want to have the totals for both (OrderQty) and (LineTotal) grouped by ProductID

This will give you the *total number of orders*, and *the total sum of price* for EACH PRODUCT

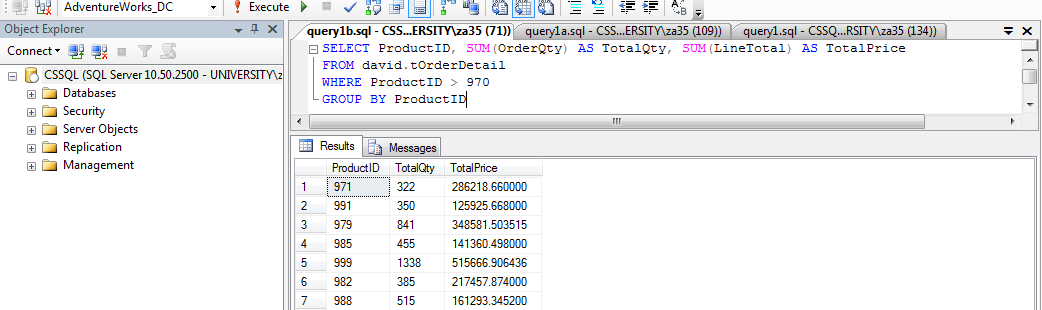
You can also use this method for a 1 specific data, as demonstrated below, we want the totals for ProductID 744 only:

(We achieve this by using WHERE clause)



You can use other commands like greater than, or less than, as done in below example:

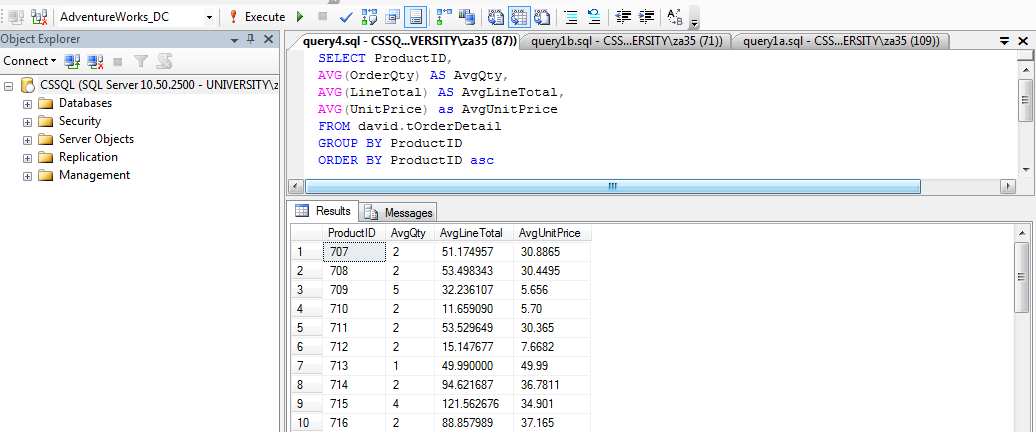
We want to group by ProductID but only for ProductID greater than >970



**Using Average feature**

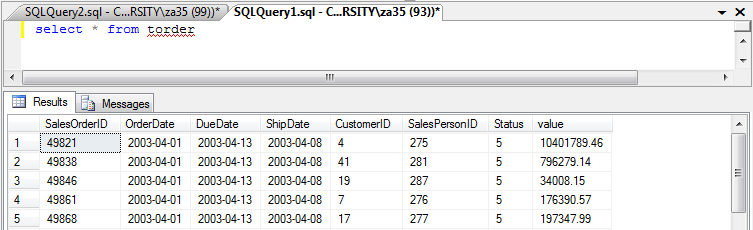
The average feature (AVG) is used in the pretty same way as the sum feature (SUM), but the only difference is as the names suggests, average feature will only give you the average, whereas the sum will give you the total sum of the data.

Example of using the Average feature:



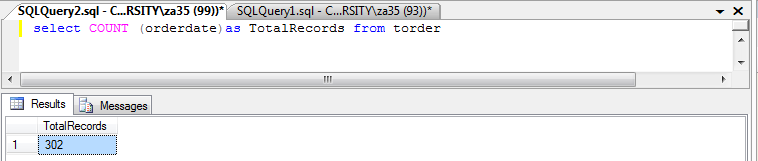
**Task 2. (week7)** Use the table tOrder in your own database and display: **2a.2b.2c.2d.2e**

First open the table (from the relevant database, in this case ZA35):

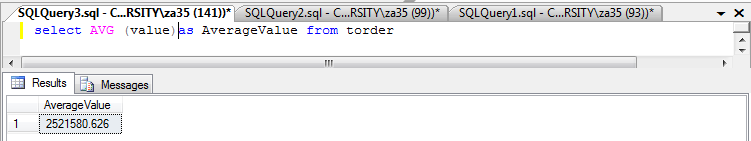


**2a The total number of records in this table:**

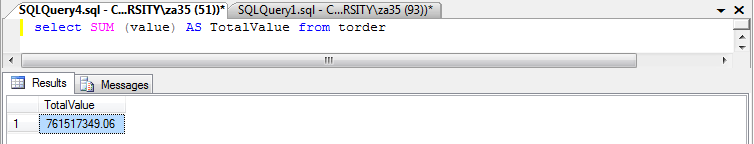
Use the COUNT to work out the total number of records. Total number of records is **302.**



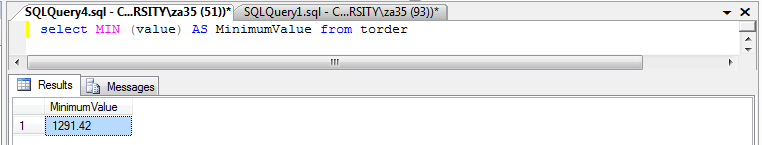
**2b. The average value across all the orders**

****

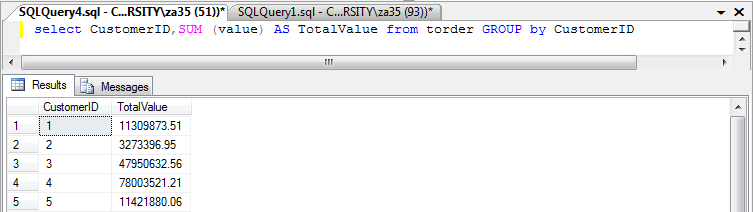
**2C. The maximum value ordered**

****

**2d. The minimum value ordered**

****

**2e. CutomerID and total value for each customer**

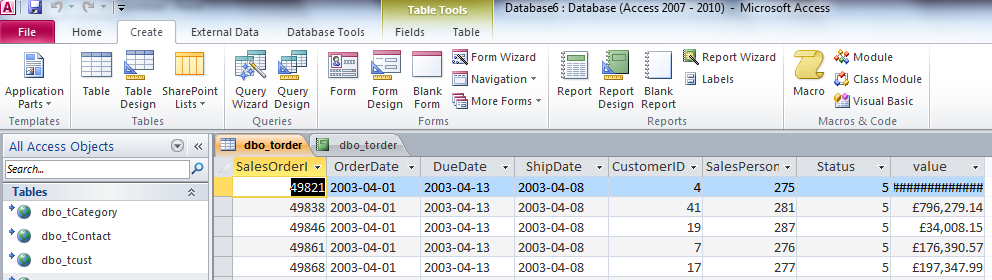
****

Week 8: Middleware and front-ends!

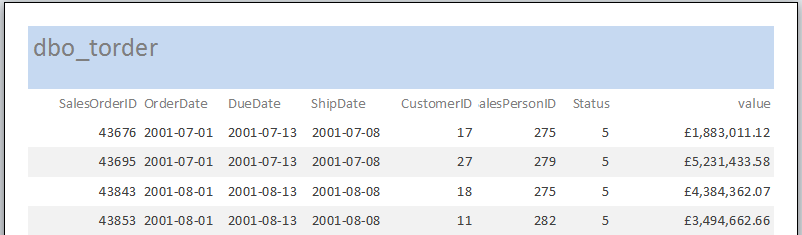
**Task 1 and 2, just follow the handout file named (odbcGuide.docx) in folder “Data Basa Lab W8/** **week\_8”**

**Task 3: Now create some simple reports against your SQL Server tables.**

**What you do, now you have your tables opened in Access, click on create, then click on Report Wizard. Then just make the report, it’s very easy to follow!**

****

**The report will look something like this:**

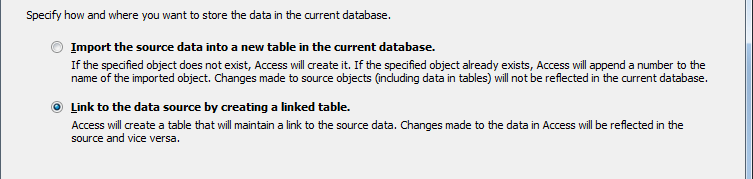
****

Week 9: Reporting tools

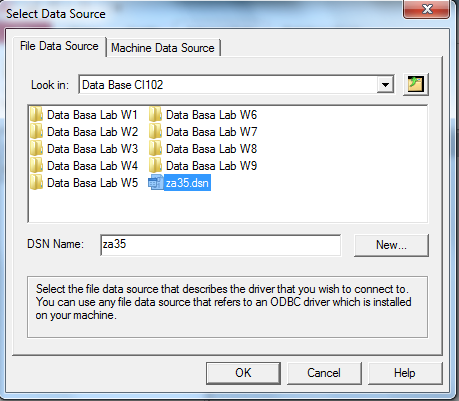
**Task 2**

**Q: Create your own Access reports against views in your CSSQL database using the ODBC middleware you created last week.**

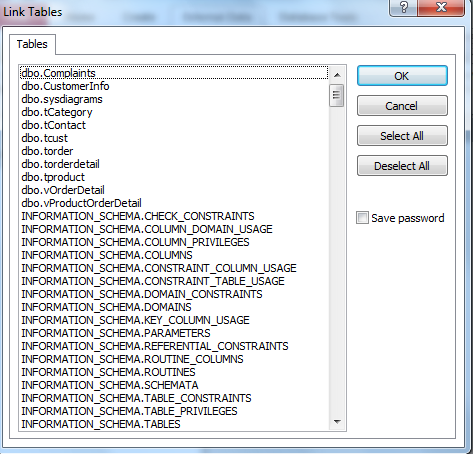
**A:** open access, (then click blank database, then close it), then click **External Data**, then click on **ODBC Database**, then **click on (Link to the data source….) as demonstrated in below image:**

****

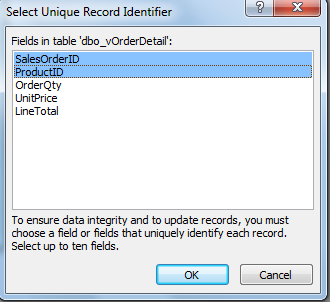
**Then chose your file source za35dsn (which you created in week 8):**

****

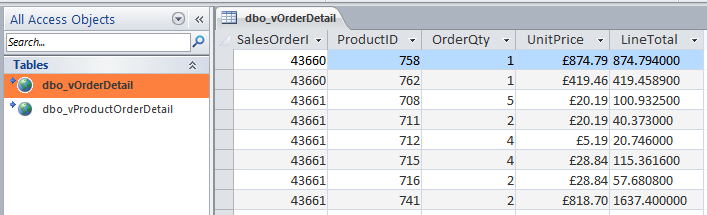
**Then chose the tables or views you want:**

****

**Then you have to specify unique keys:**

****

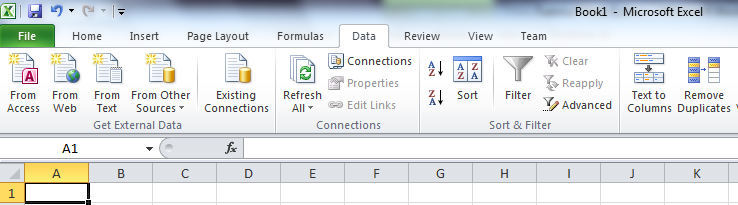
**Then job done: just click on the views you want to see:**

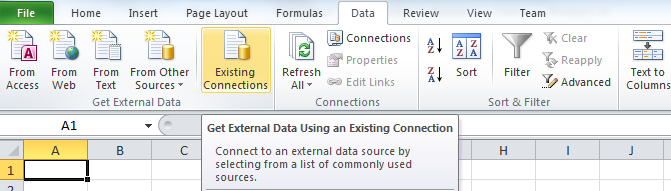
****

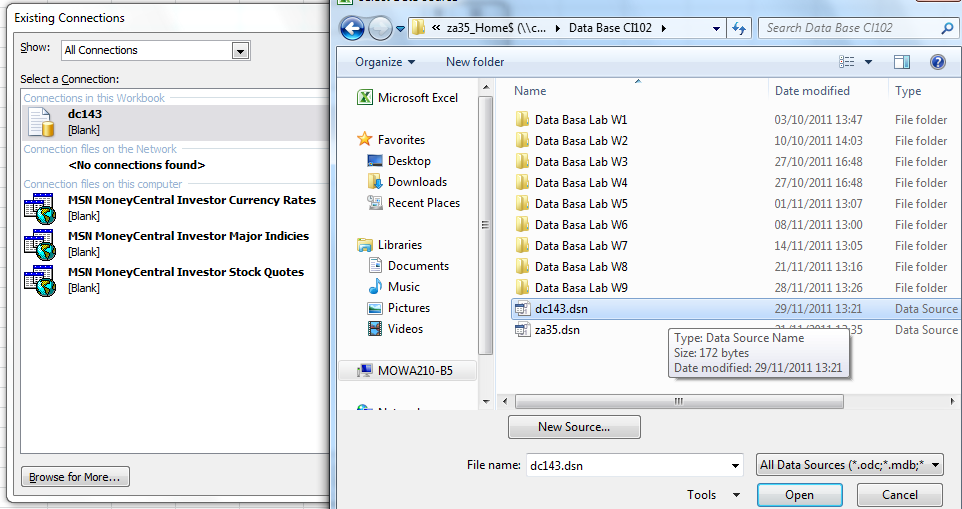
**Task 3**

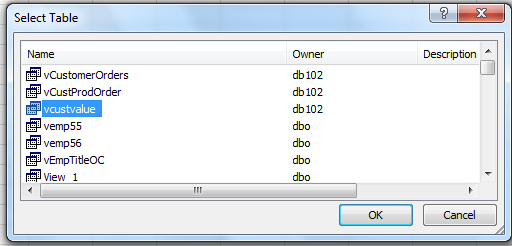
**Q: Create charts…perhaps using my view db102.vcustvalue. To use my view you will need another ODBC file this time pointing at my database dc143. [Notice the power of retrieving data from a range of database]**

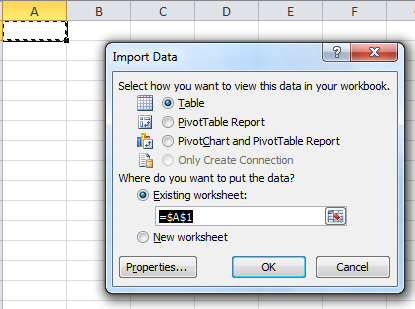
**A: first, create a new ODBC file to connect to David’s database (dc143.dsn), then open it in Excel:**

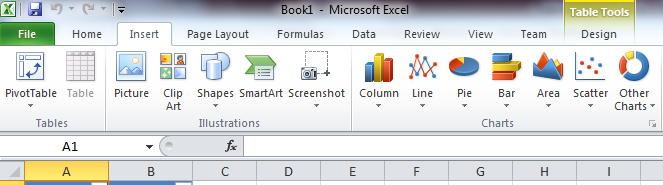
**1. Click on data:**

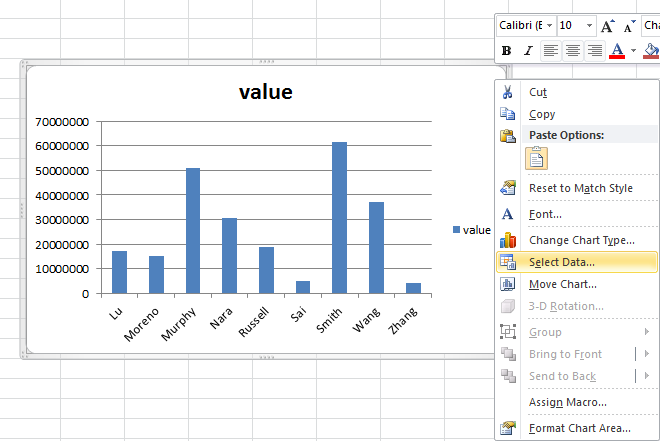
**2. Click on Existing Connections:**

**3. Then click on browse for more… then chose the desired ODBC file, in this case (dc143.dsn)**

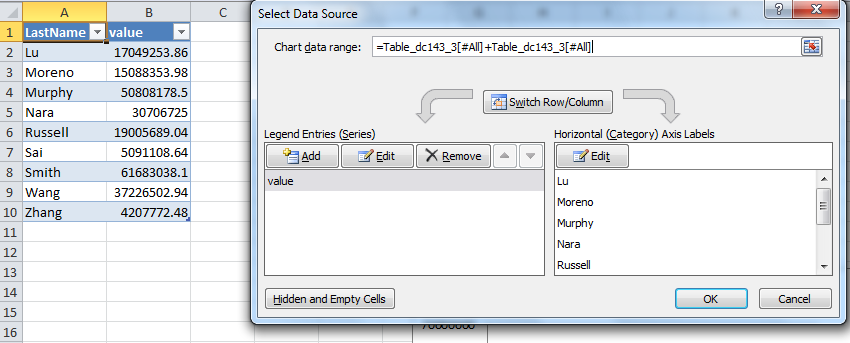
**Then chose the file you desire, in this case vcustvalue:**

**Then choose where you want it to be placed in Excel:**

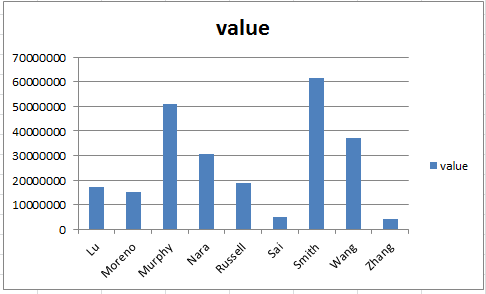
**Then click on Insert:**

**Then chose the type of chart you want to have**

**Then right click on the chart, and click on select data you want to turn into a chart.**

****

**Then press ok, then you’ll have the chart:**

****