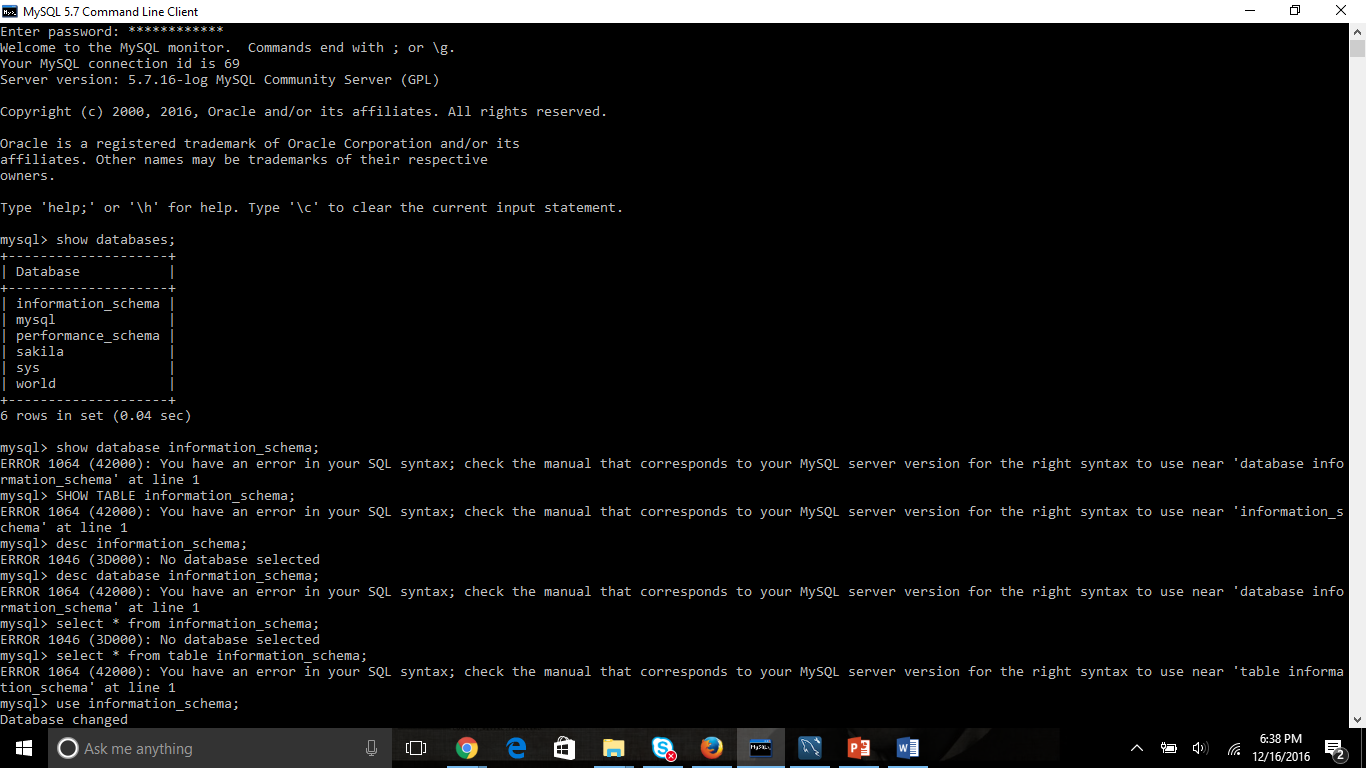
**SQL My PRACTICE**

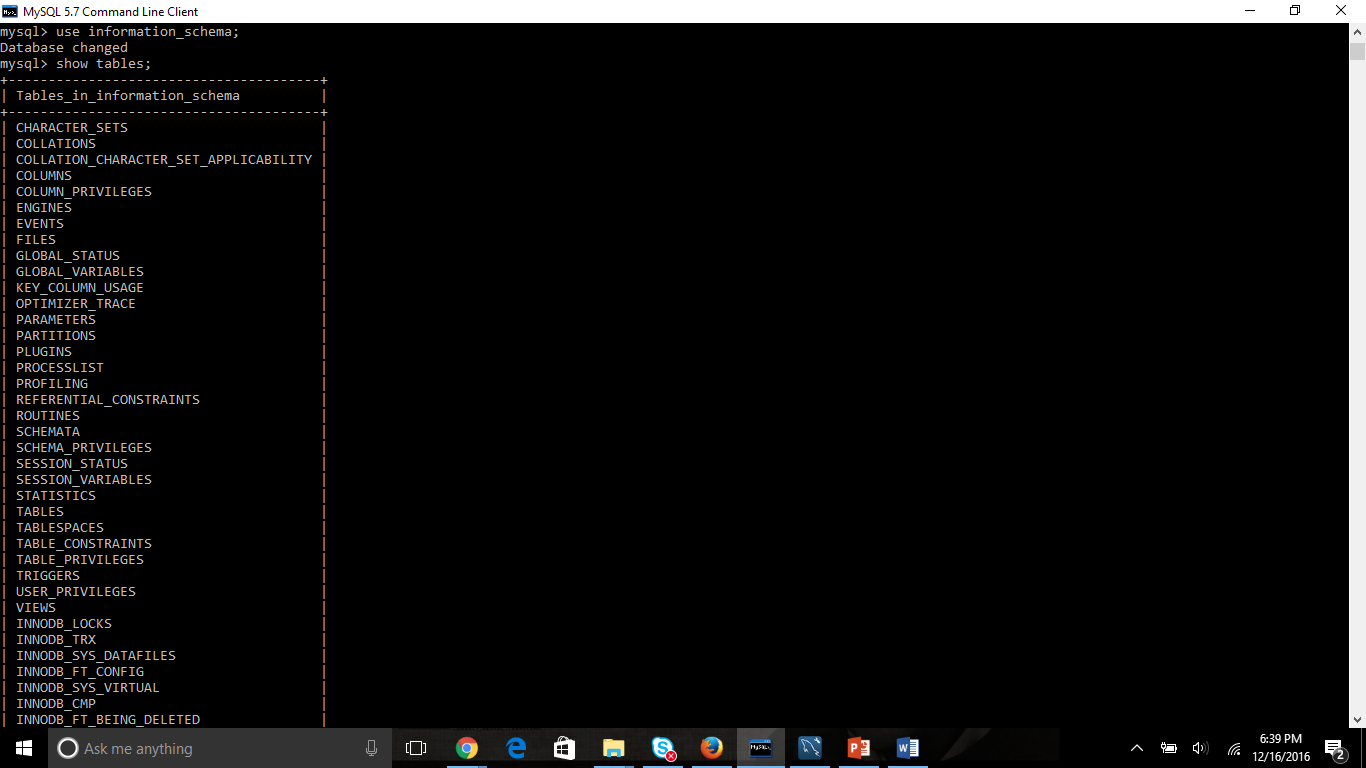
**Day I**

**How to select, choose and use database and tables**

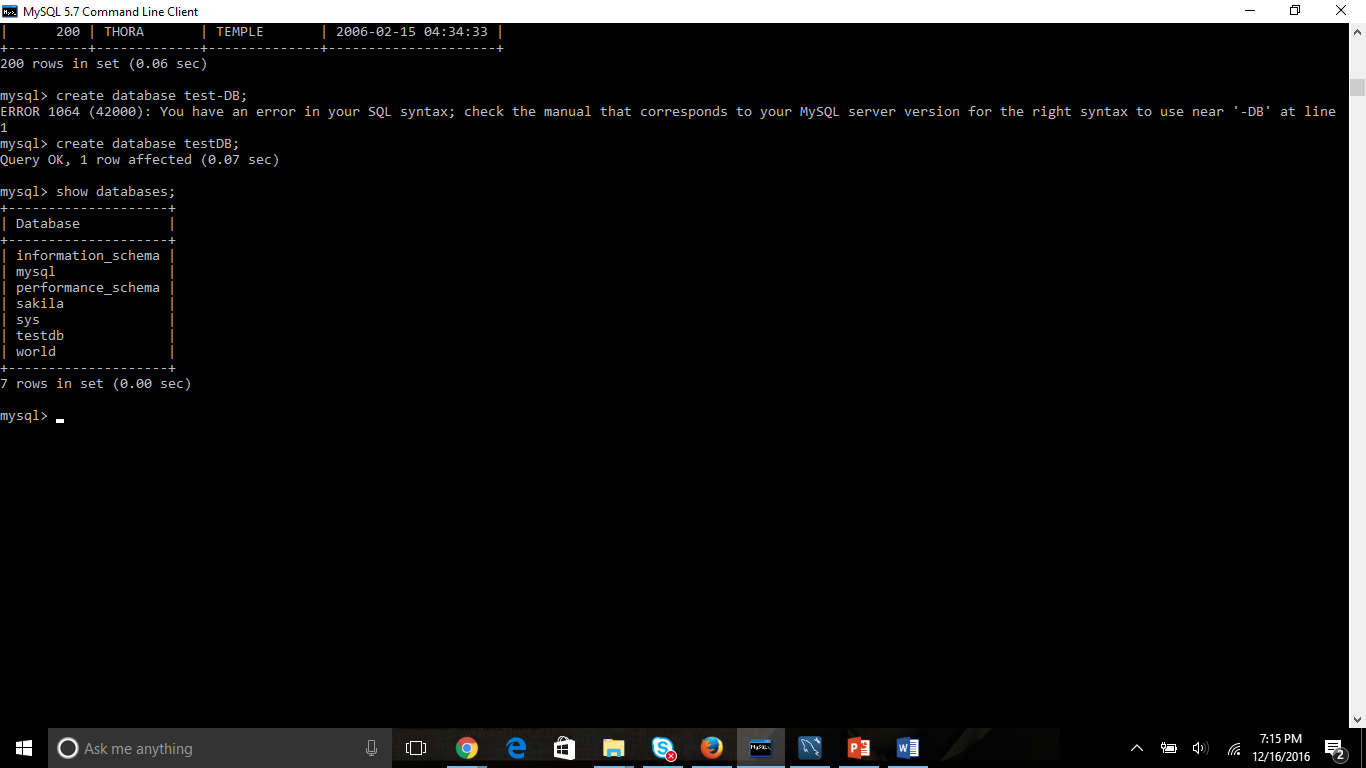
**Showing databases;**



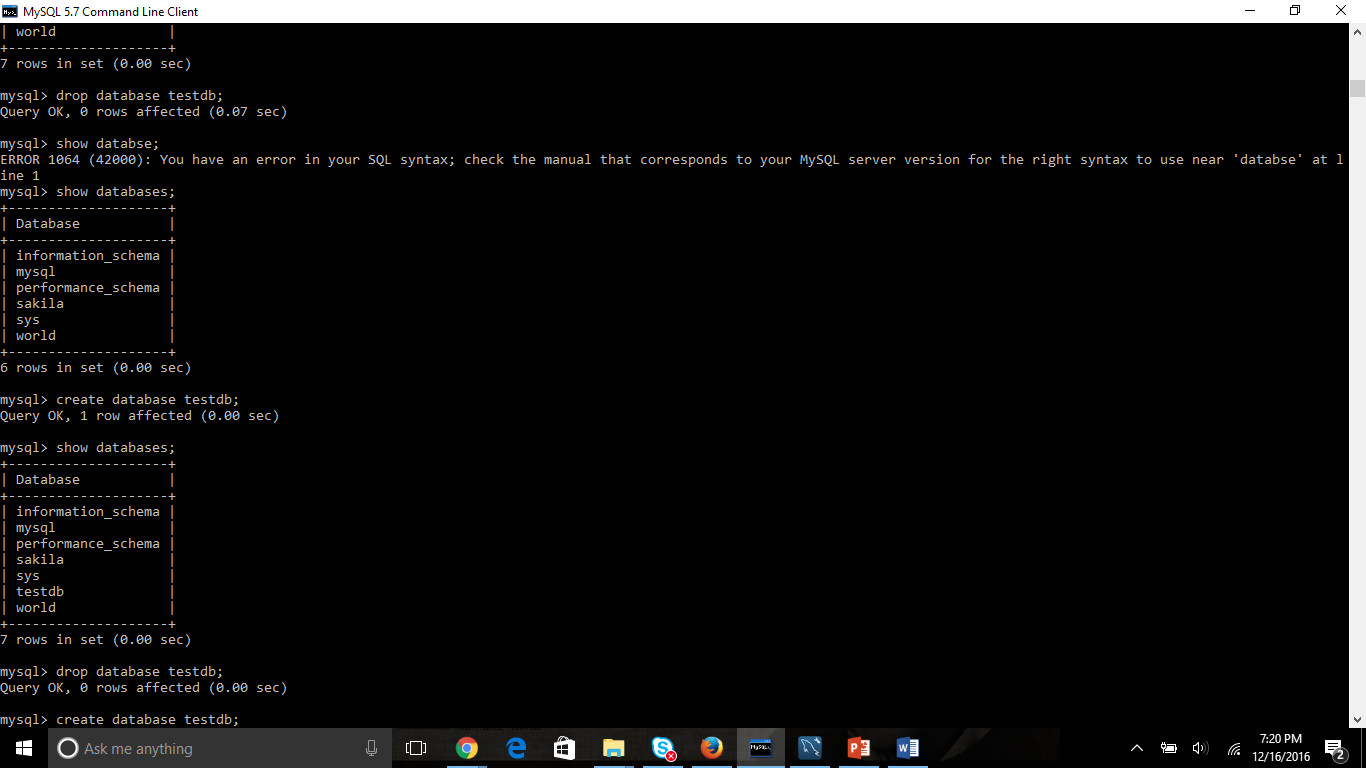
**Select database and their inside tables;**



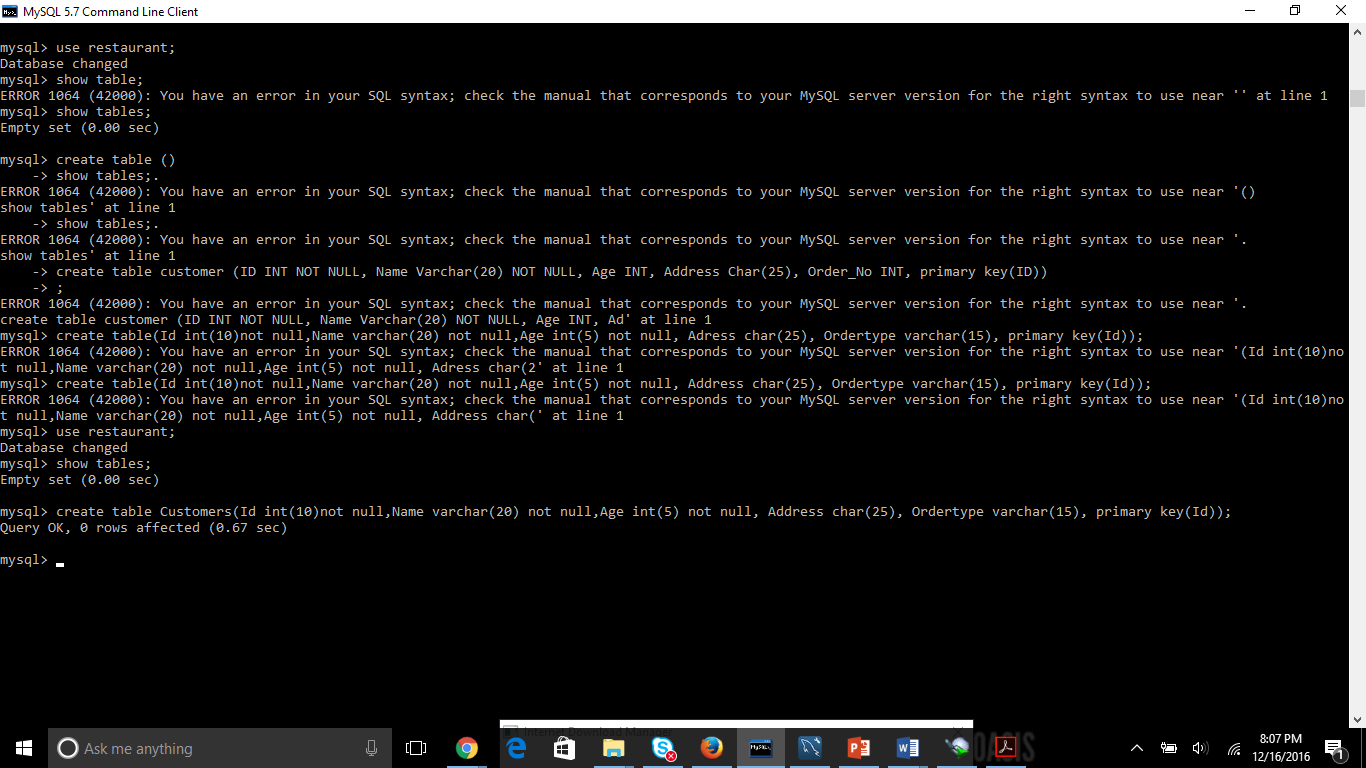
**Create Database**



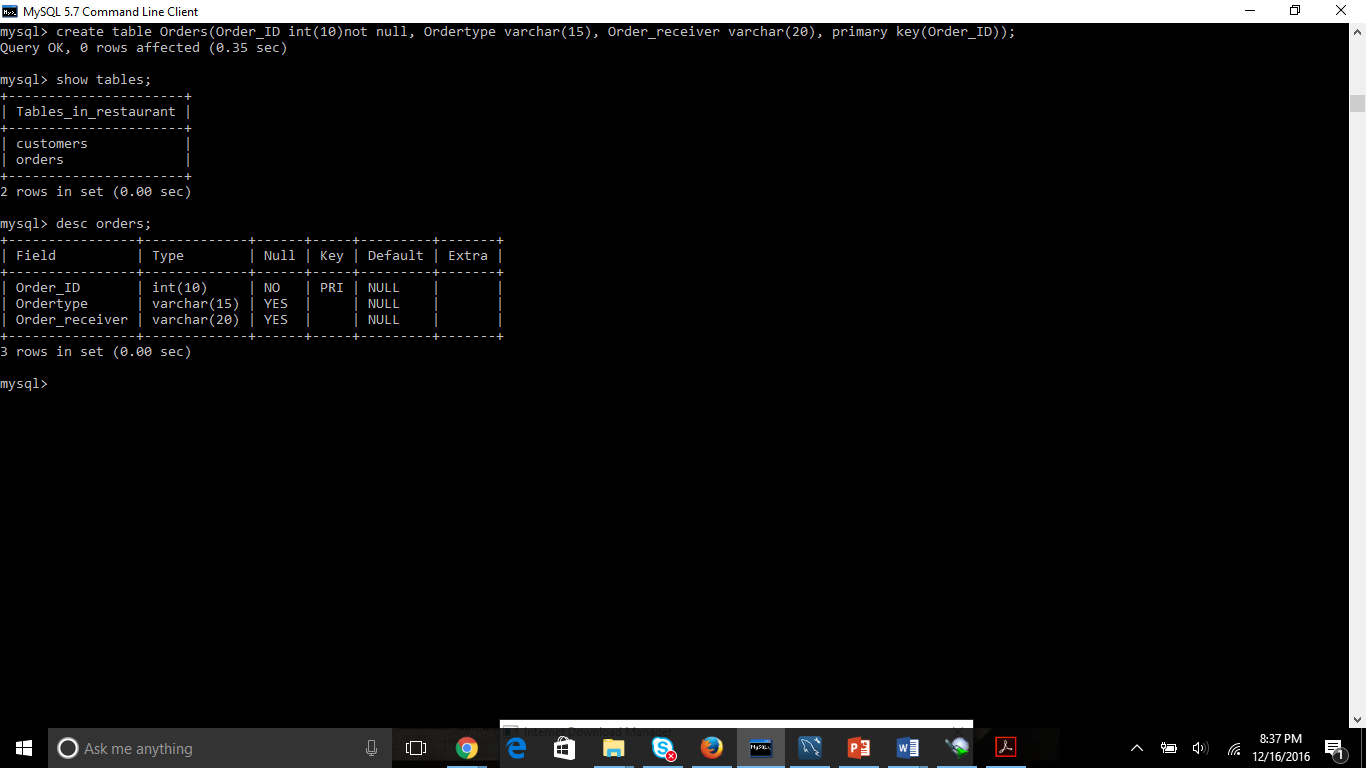
**Drop/delete database:**



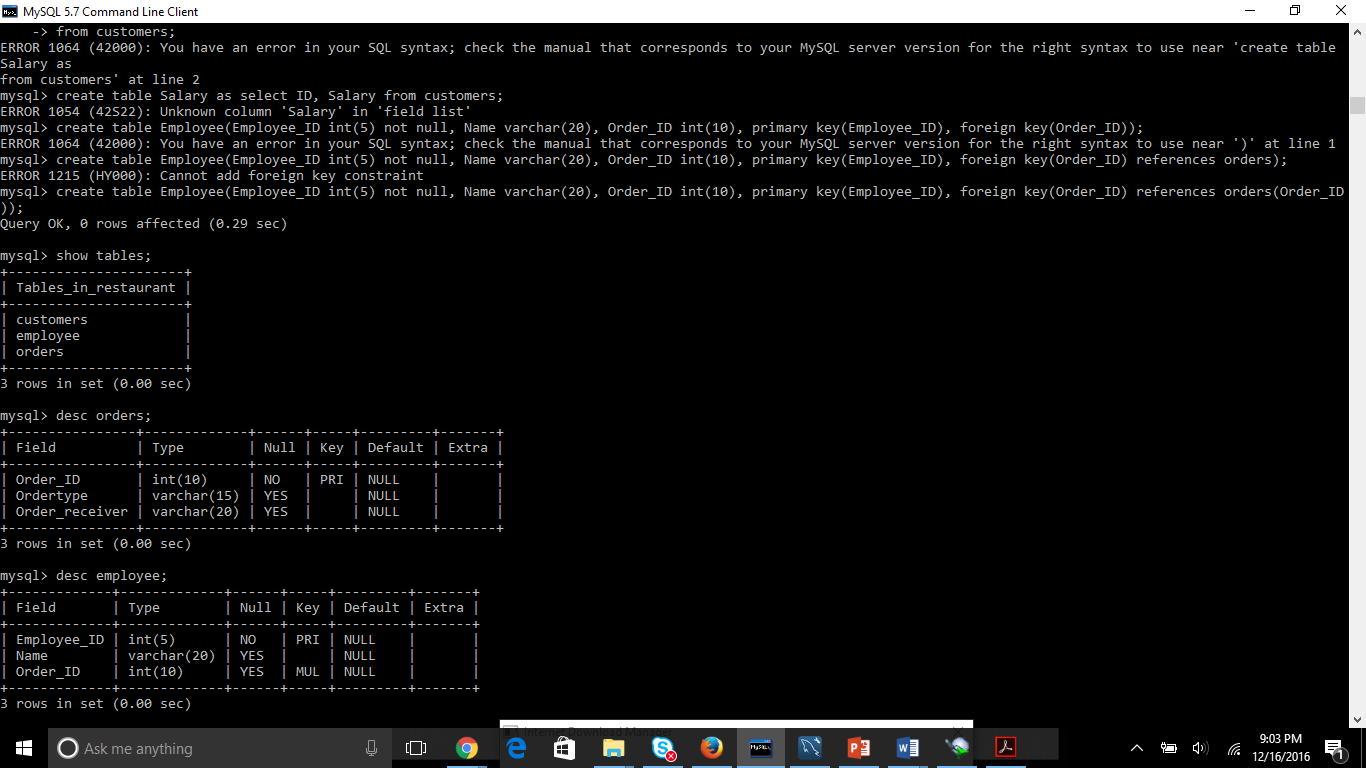
**How to create a table:**



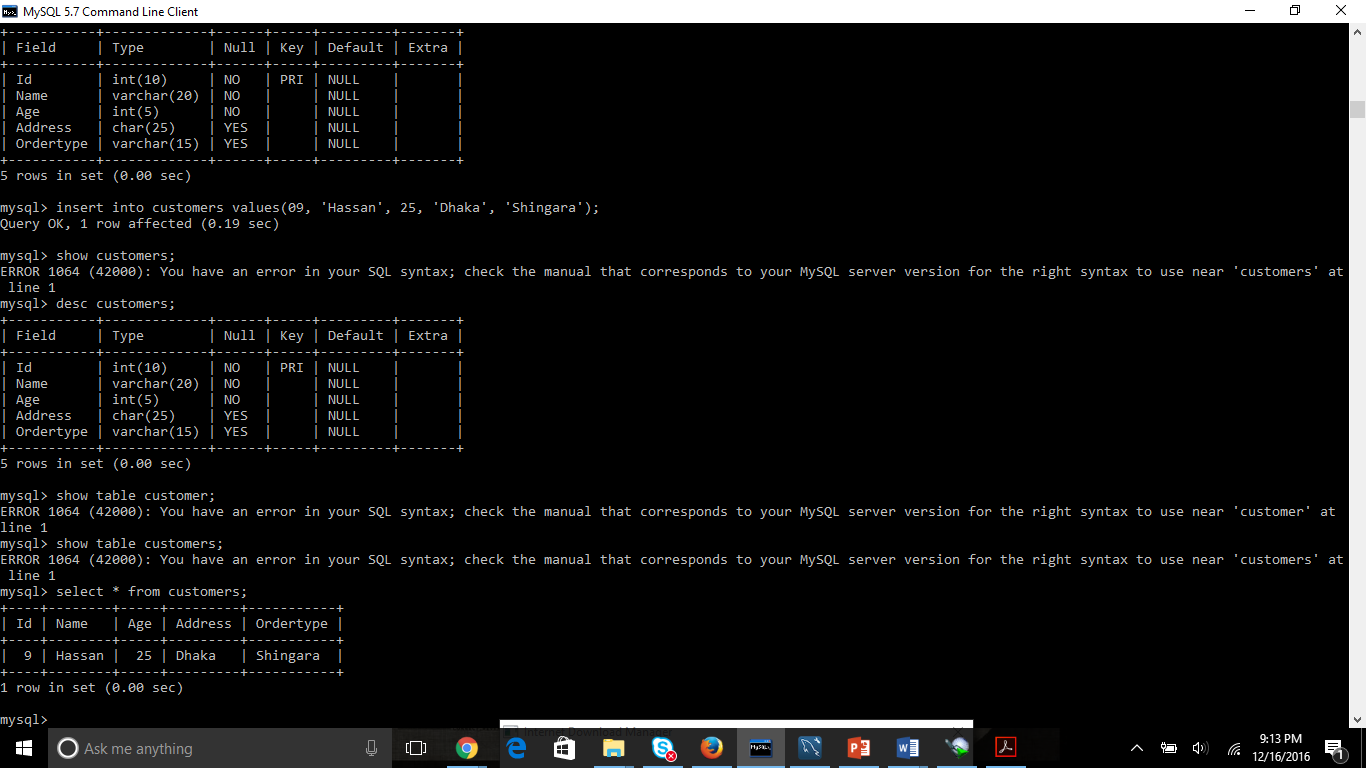
**How to view the table:**

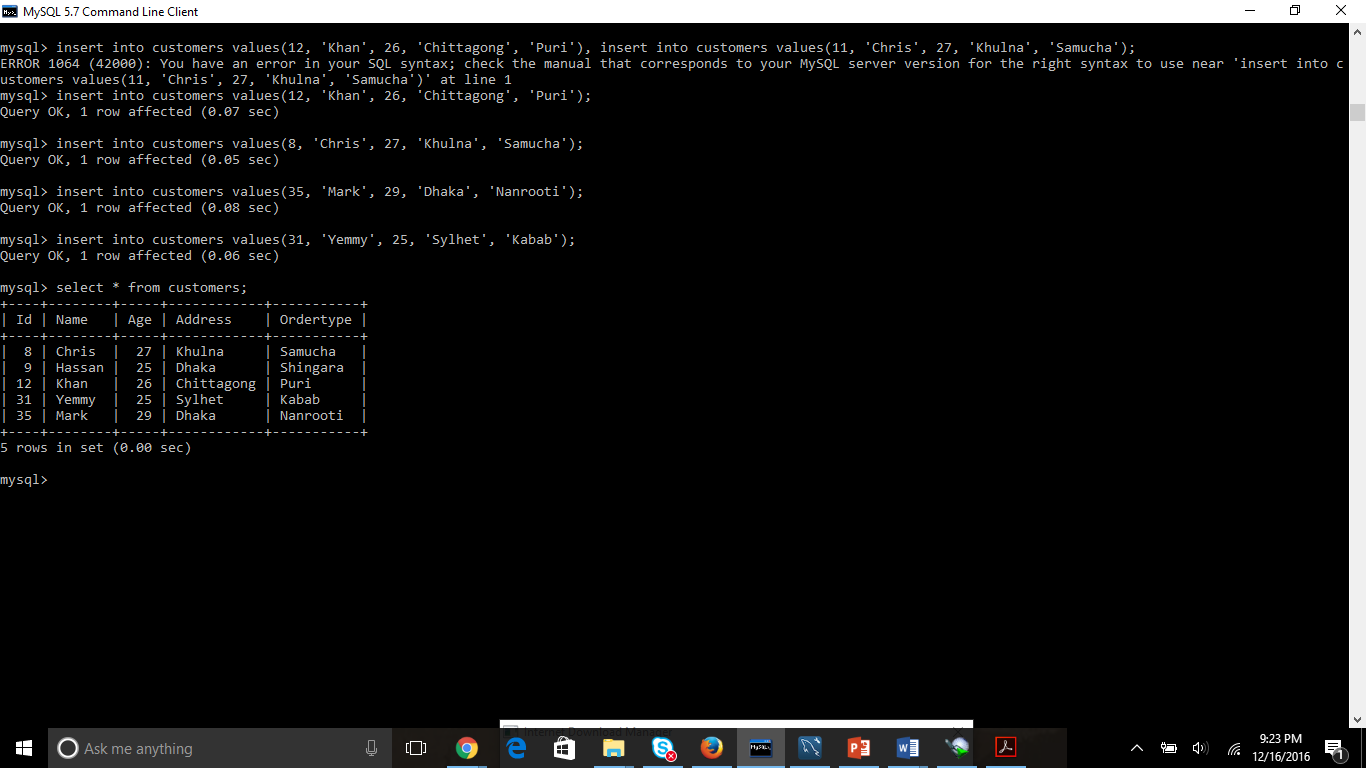


**HOW TO ADD FOREIGN KEY**

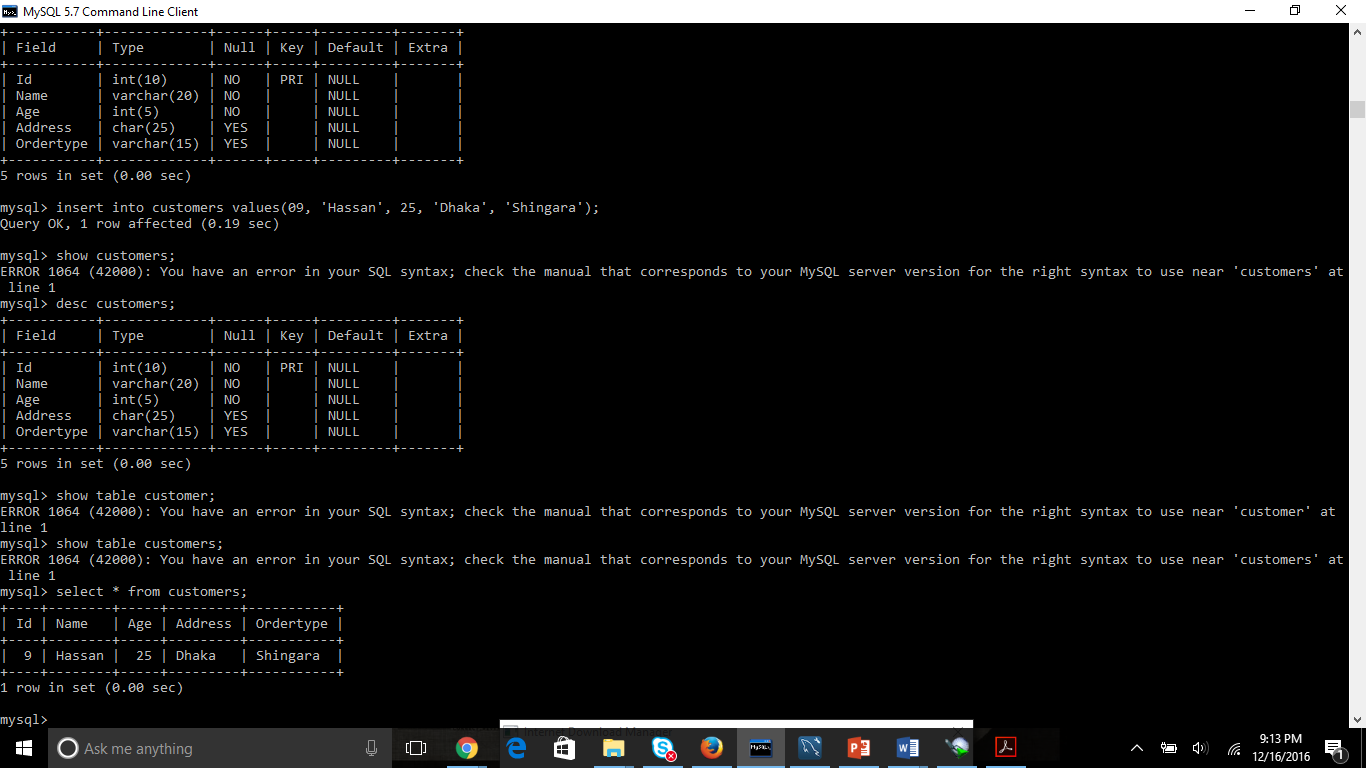


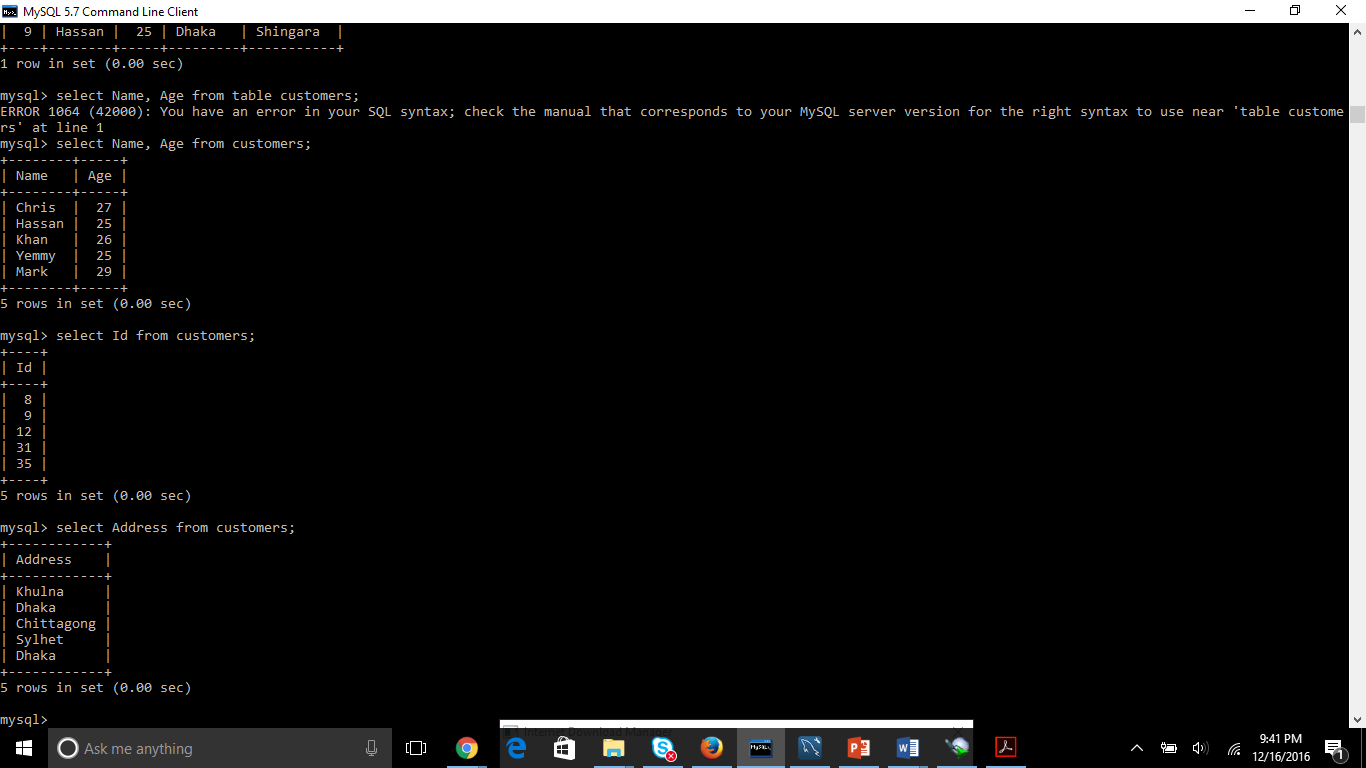
How to insert values in a table

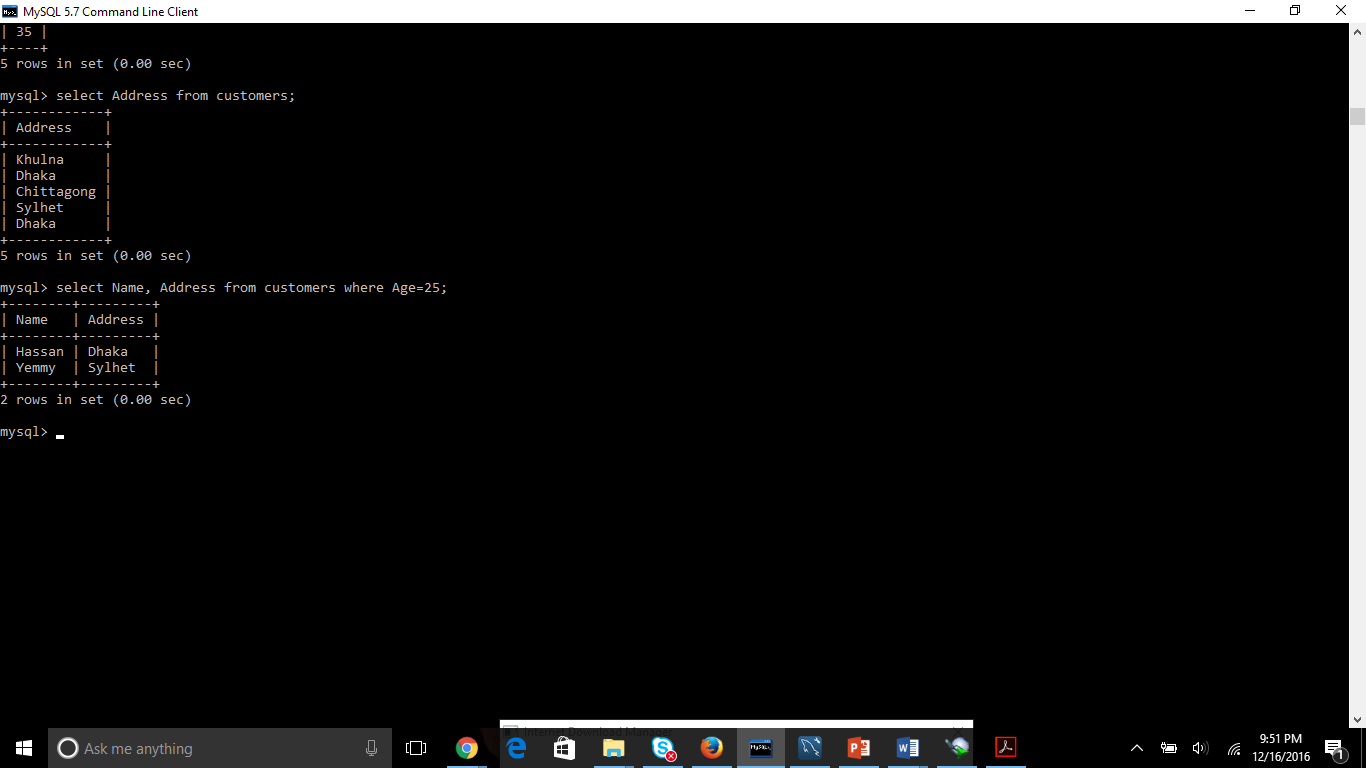




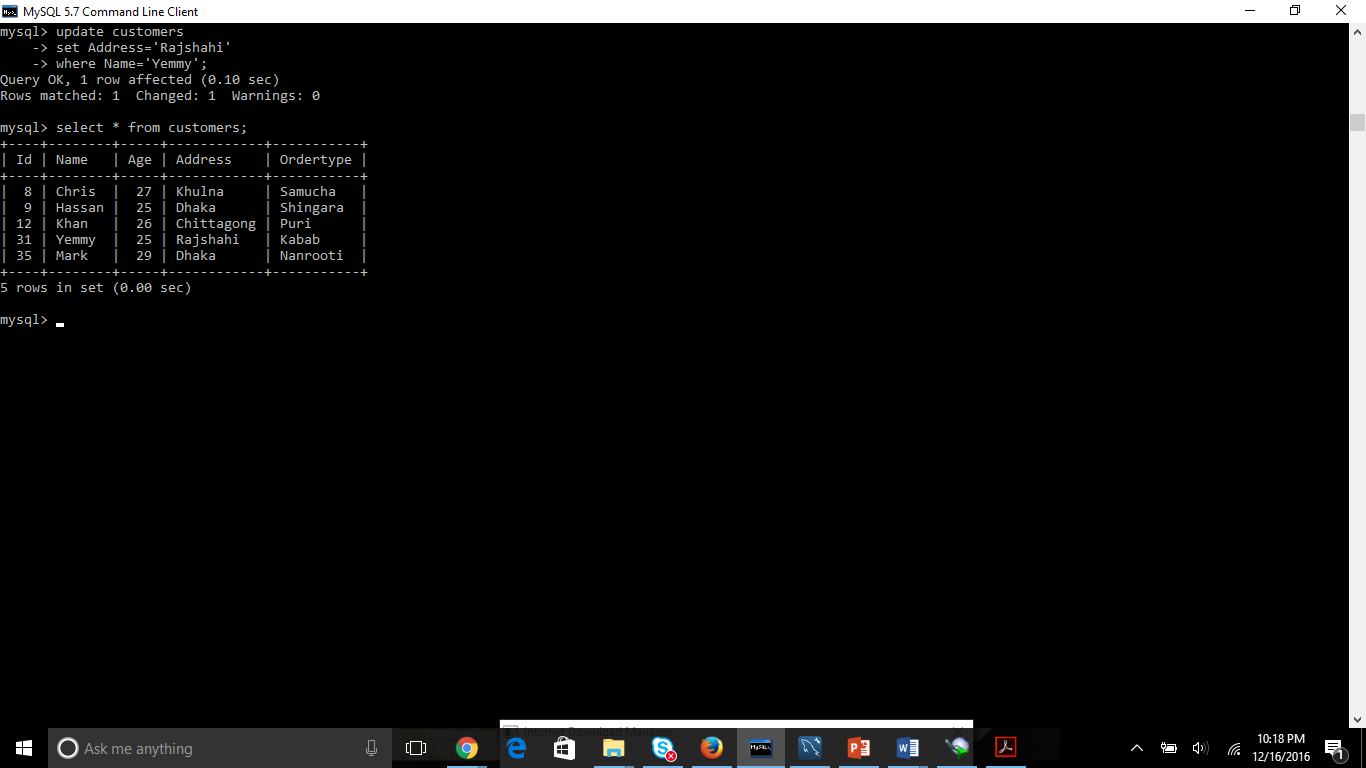
Show the data inside the table



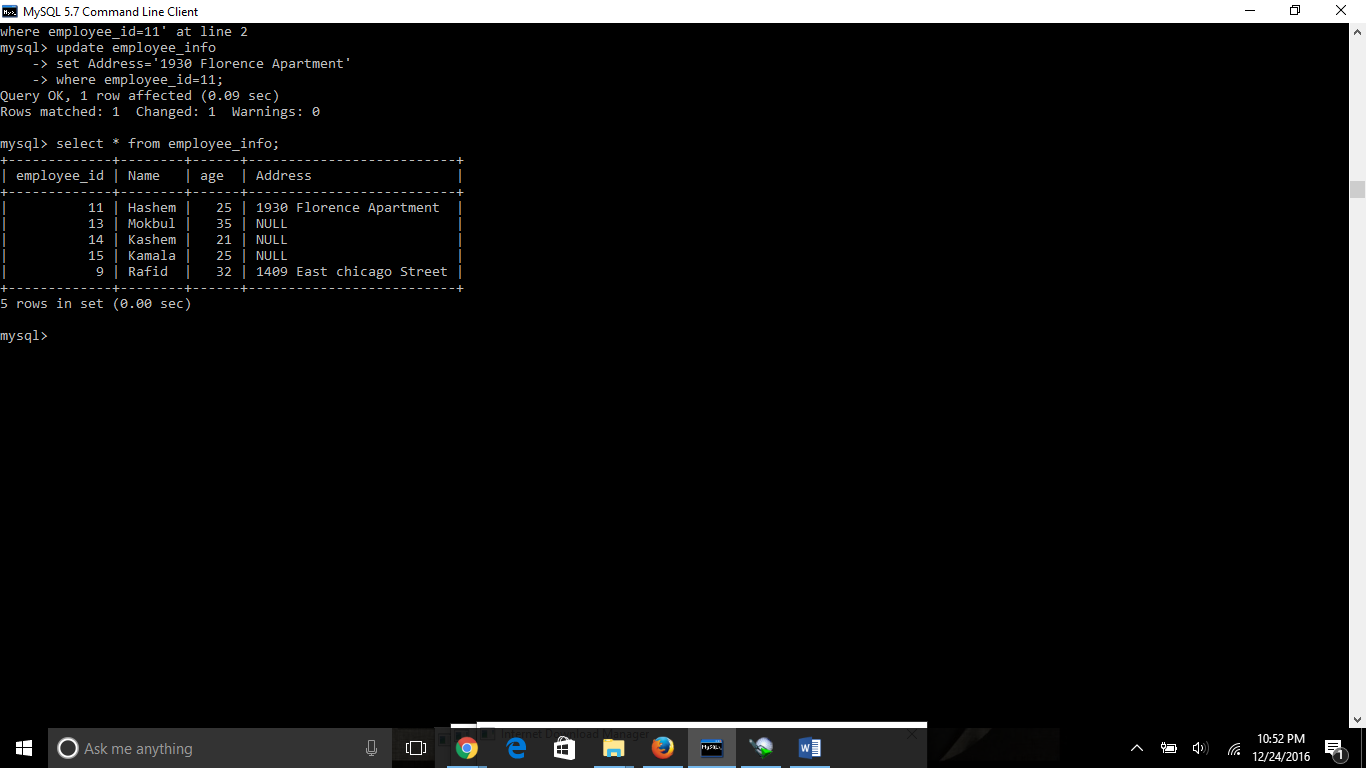




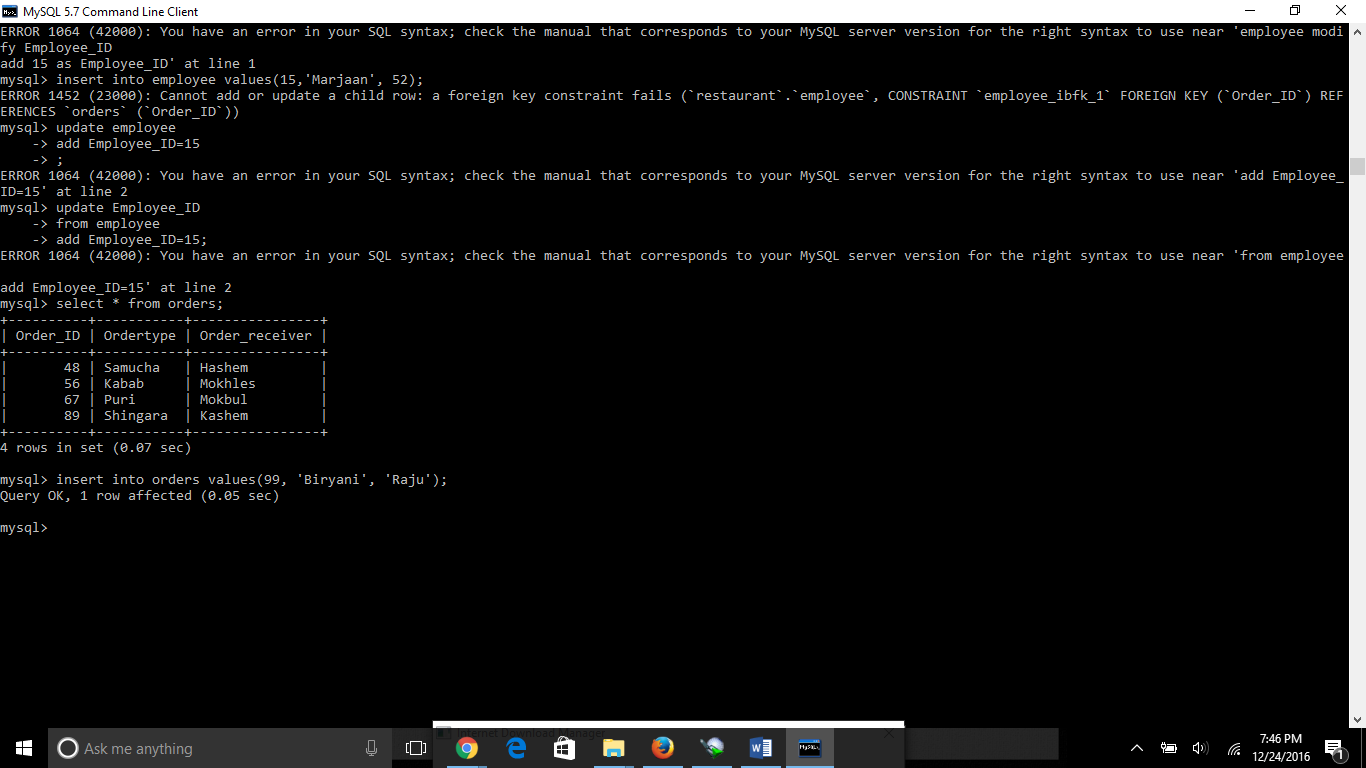
How to update a particular data in a particular table. Here I updated the address of the person whose name was yemmy.



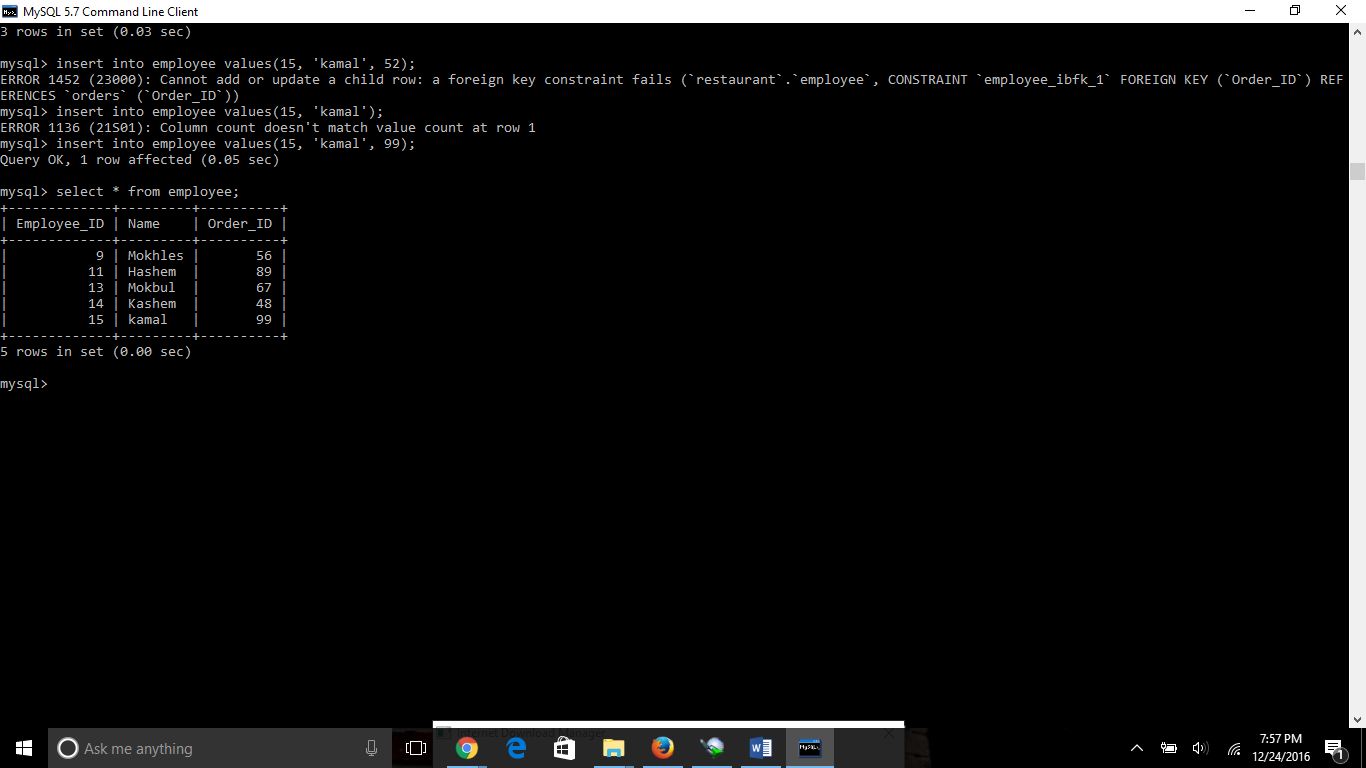
Updatae column



Updating Rows or records

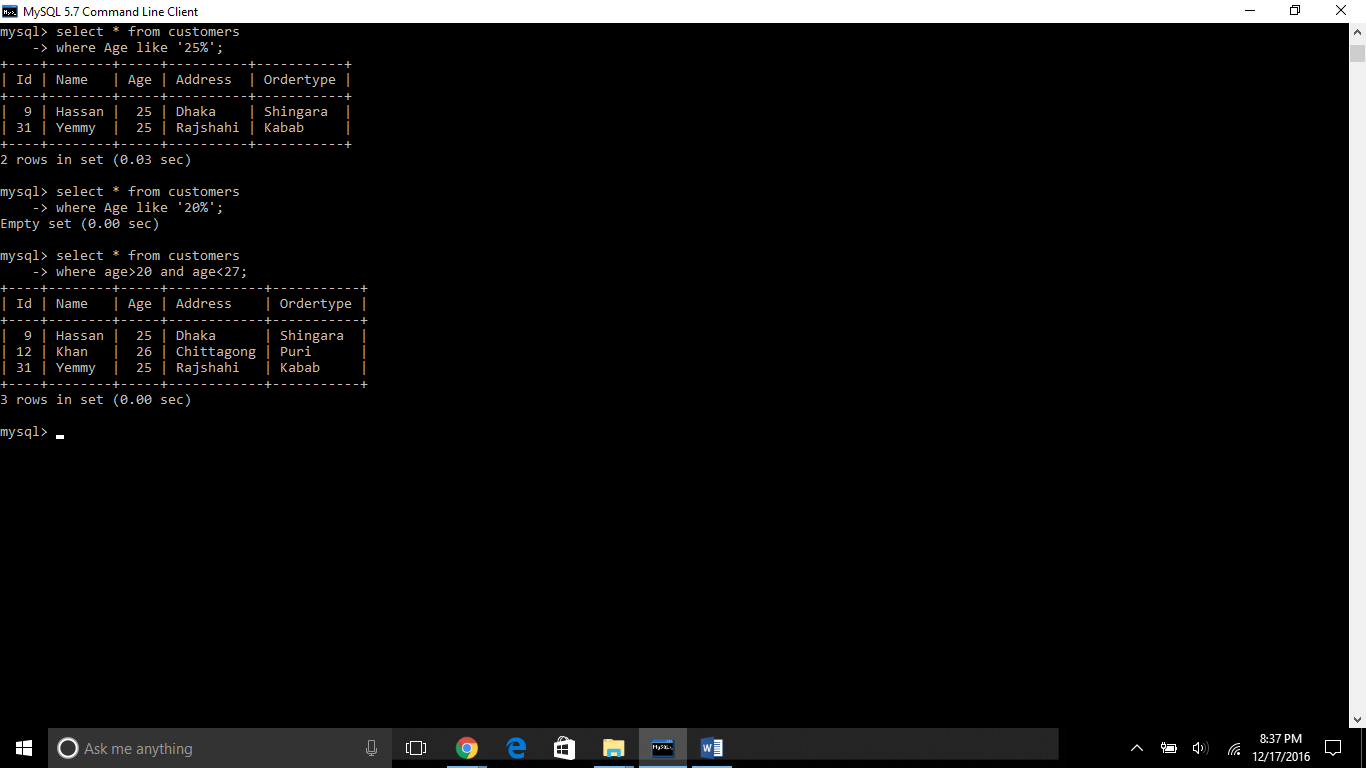


\*\*\* notes: you cannot change or update a child row or records, You have to modify the parent row or records at first then modify the child records.

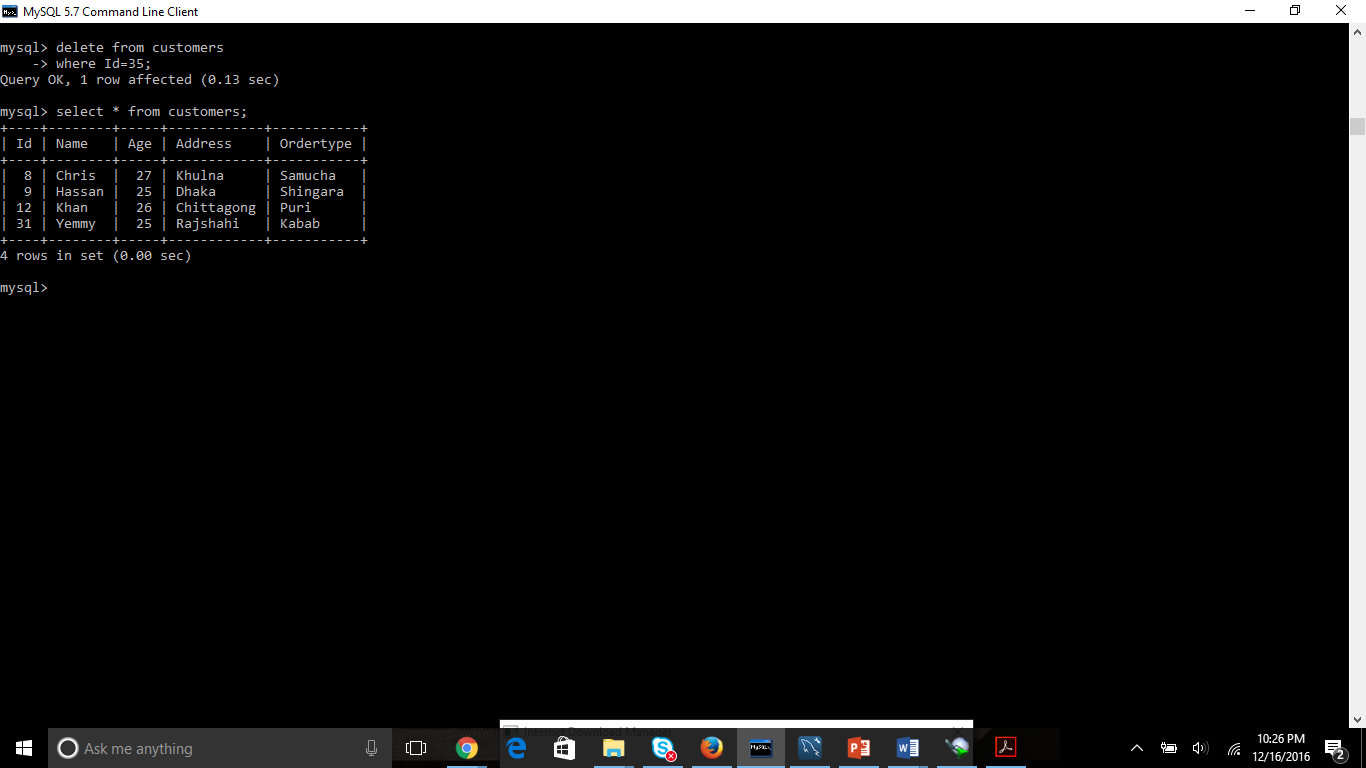


Here you can see that we change the data of a child row after updating the parent row. The mistake we made was we tried to change a different or new values which is not right. When insert any values in a child table or row always put the same value of the column which we have changed in the parent row or records.

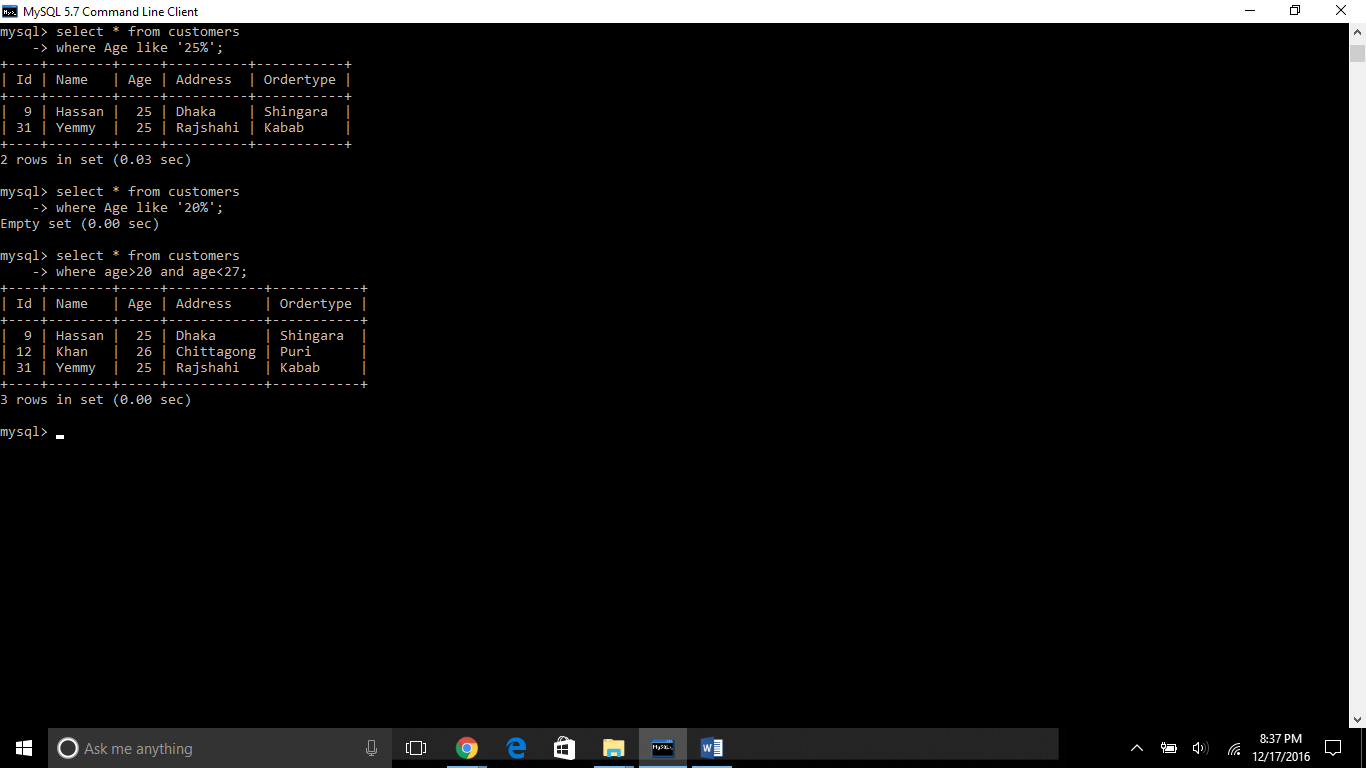
Where clause with range

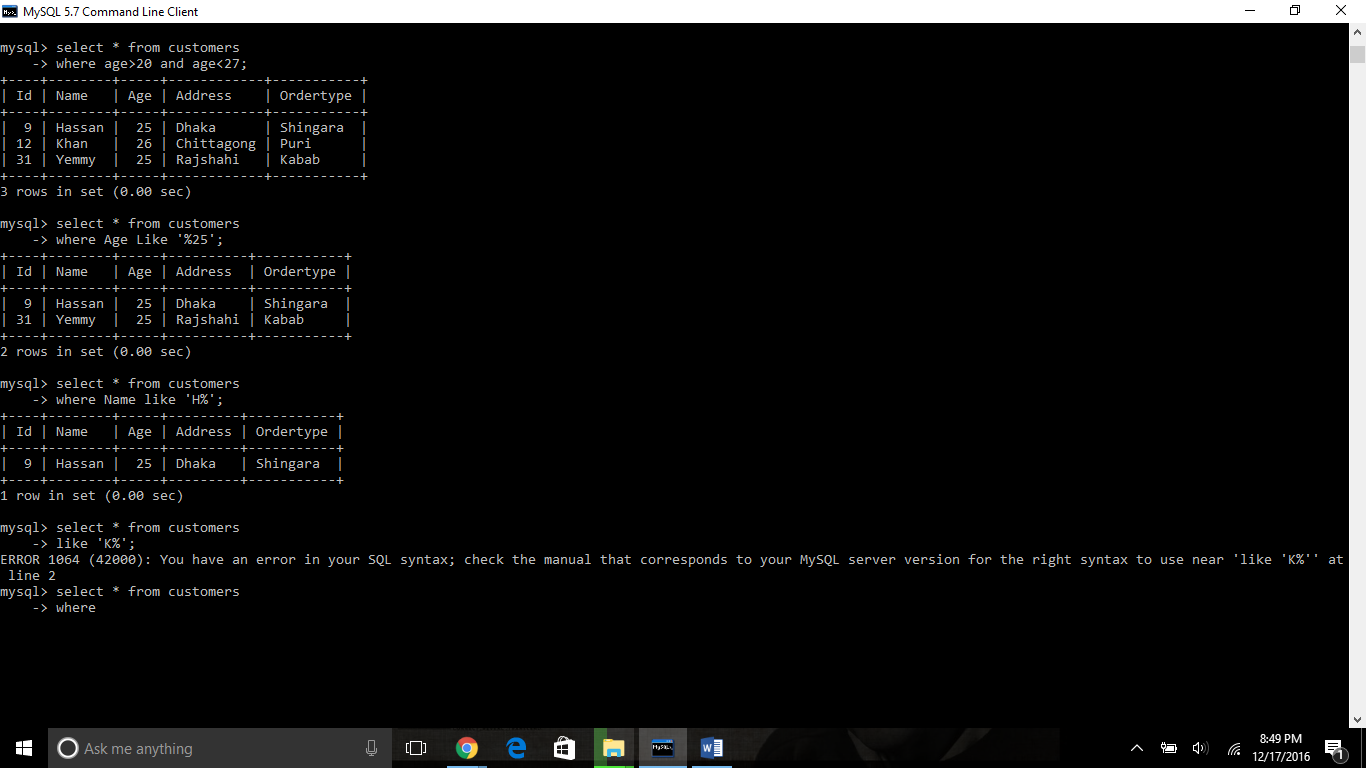


Delete query

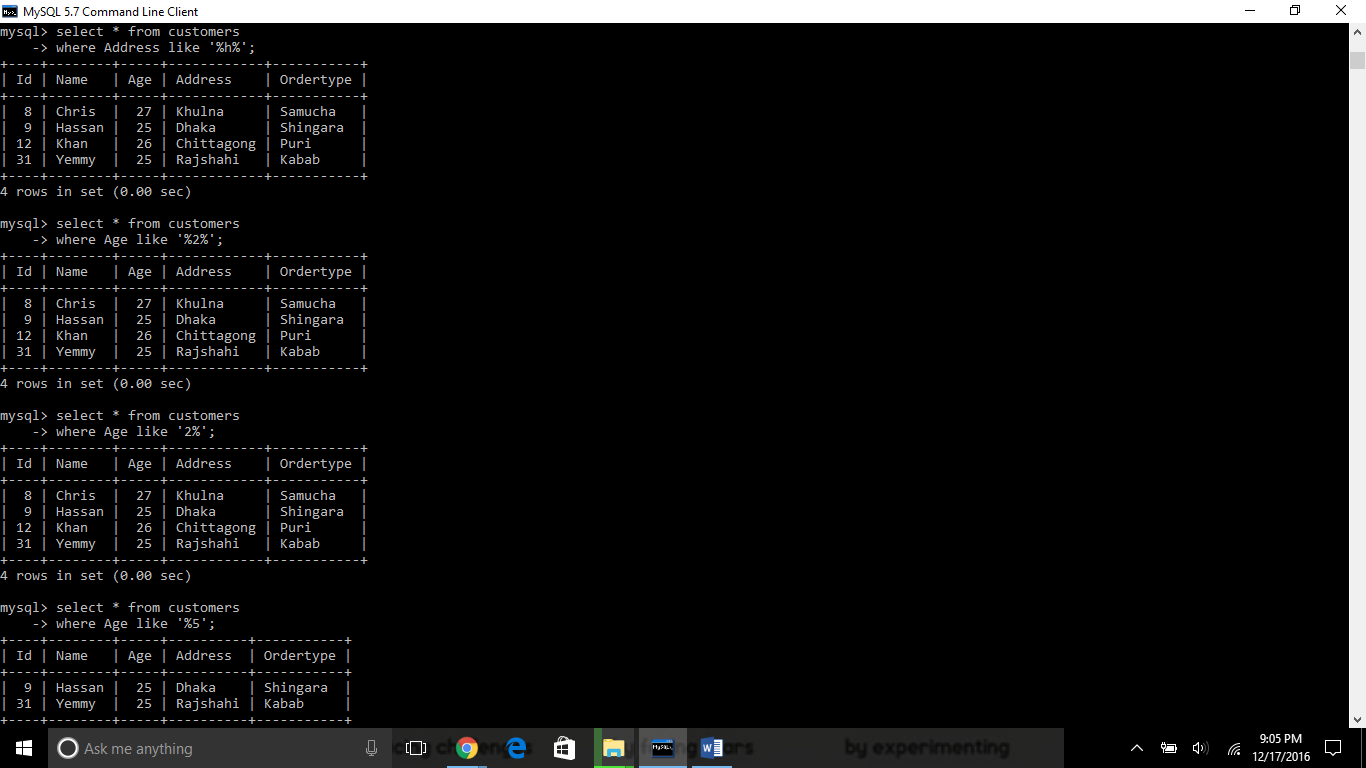


Like Clause

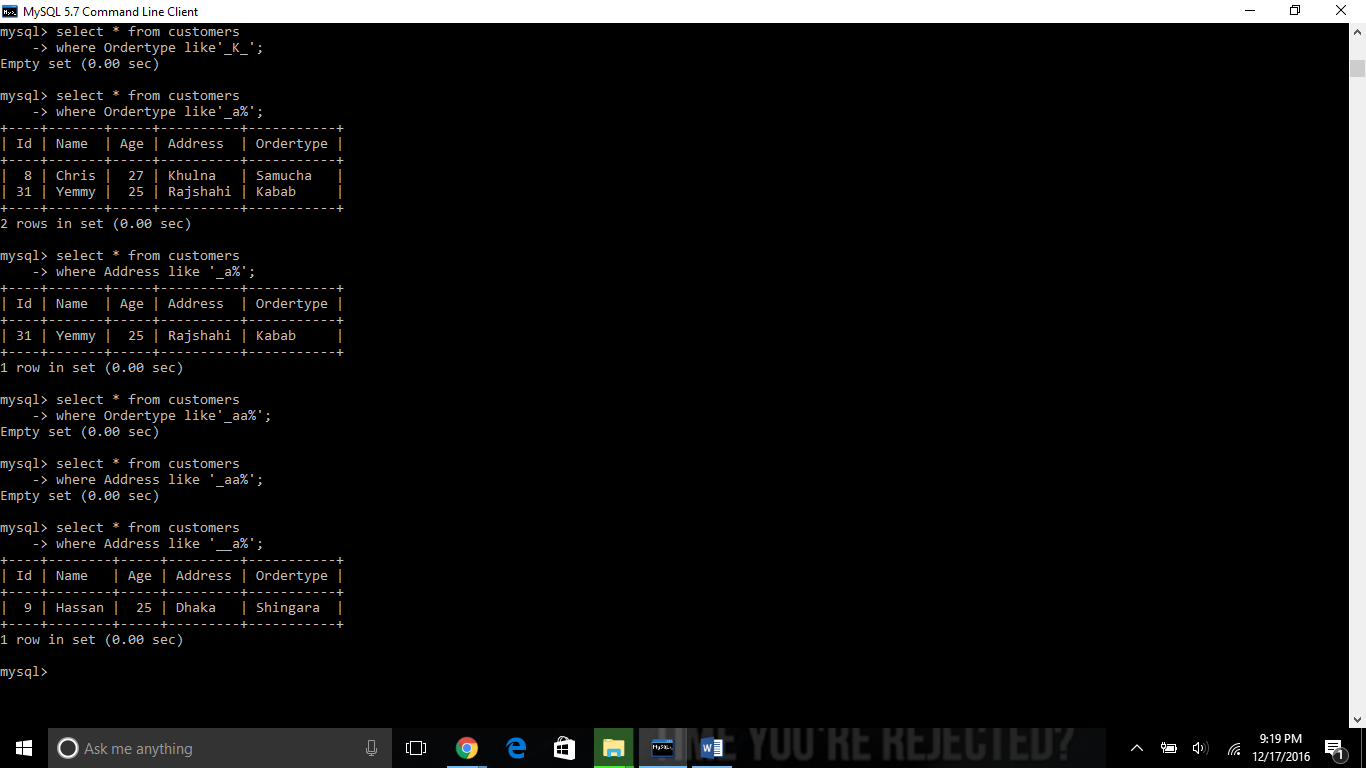




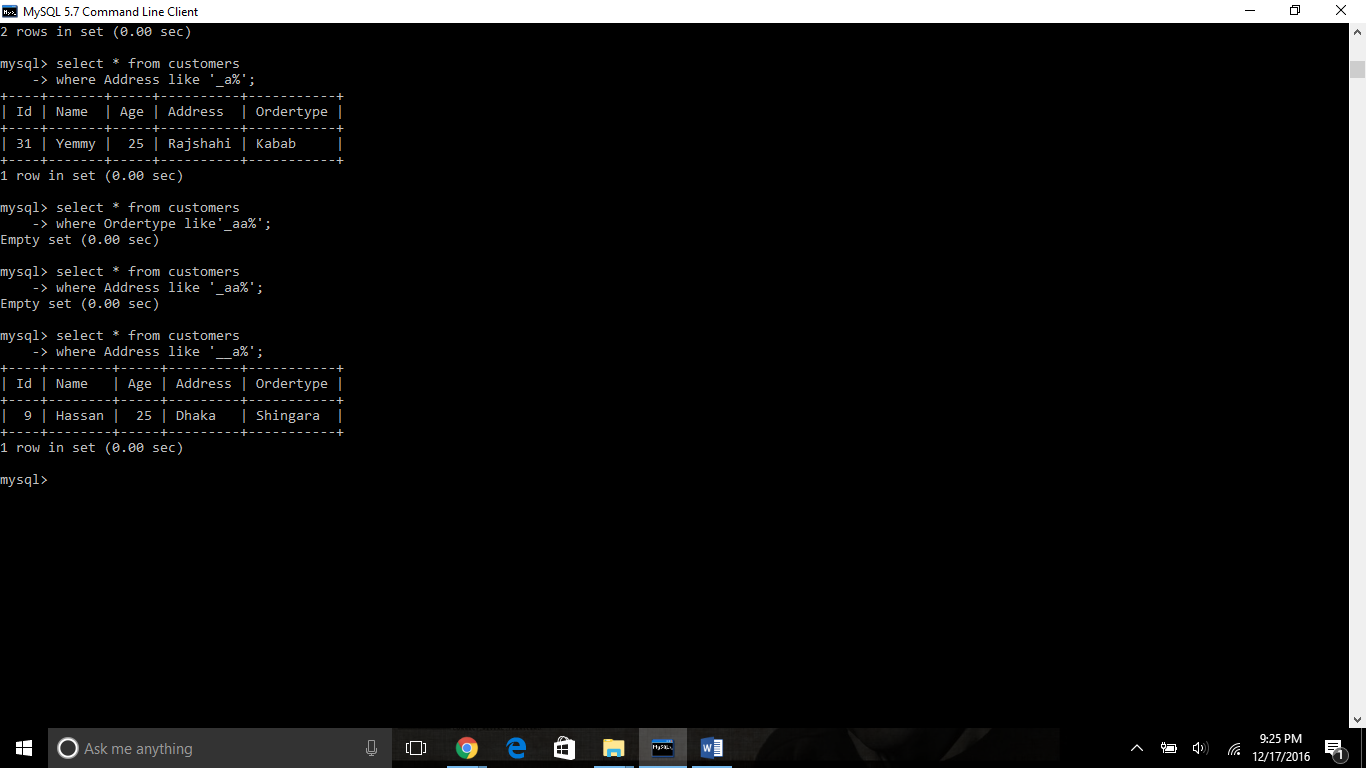


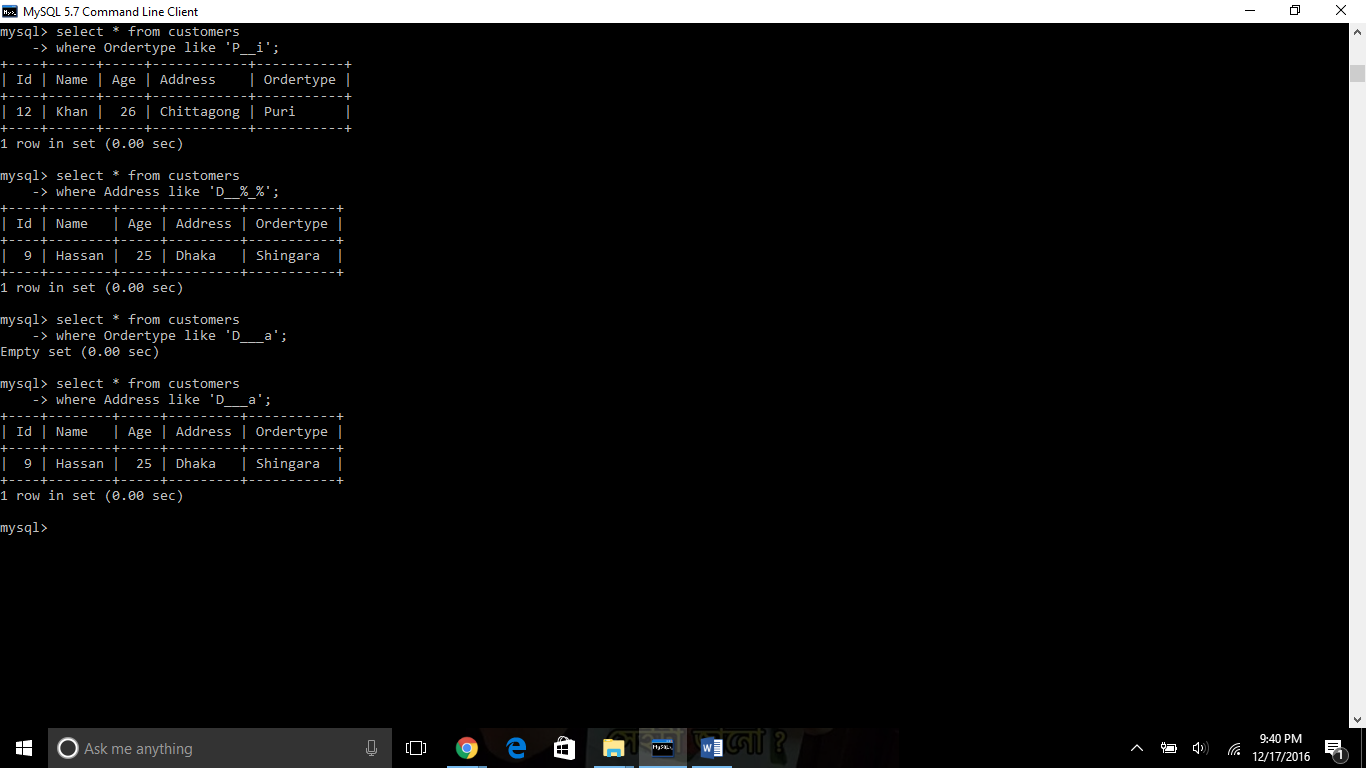


Here is the example of Like clause, where I wanted to know any character which second alphabet is a. That’s why I have used only one underscore and the % means the first alphabet starts with a.

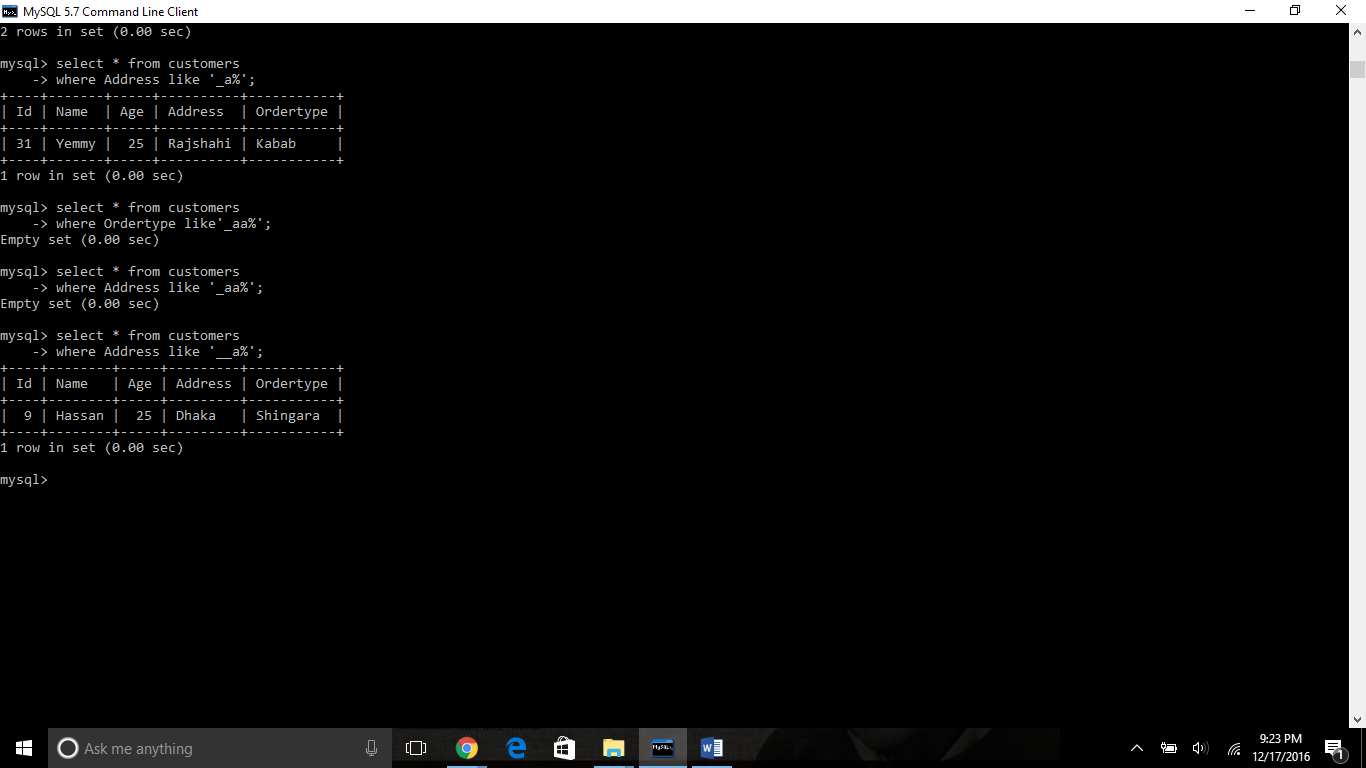


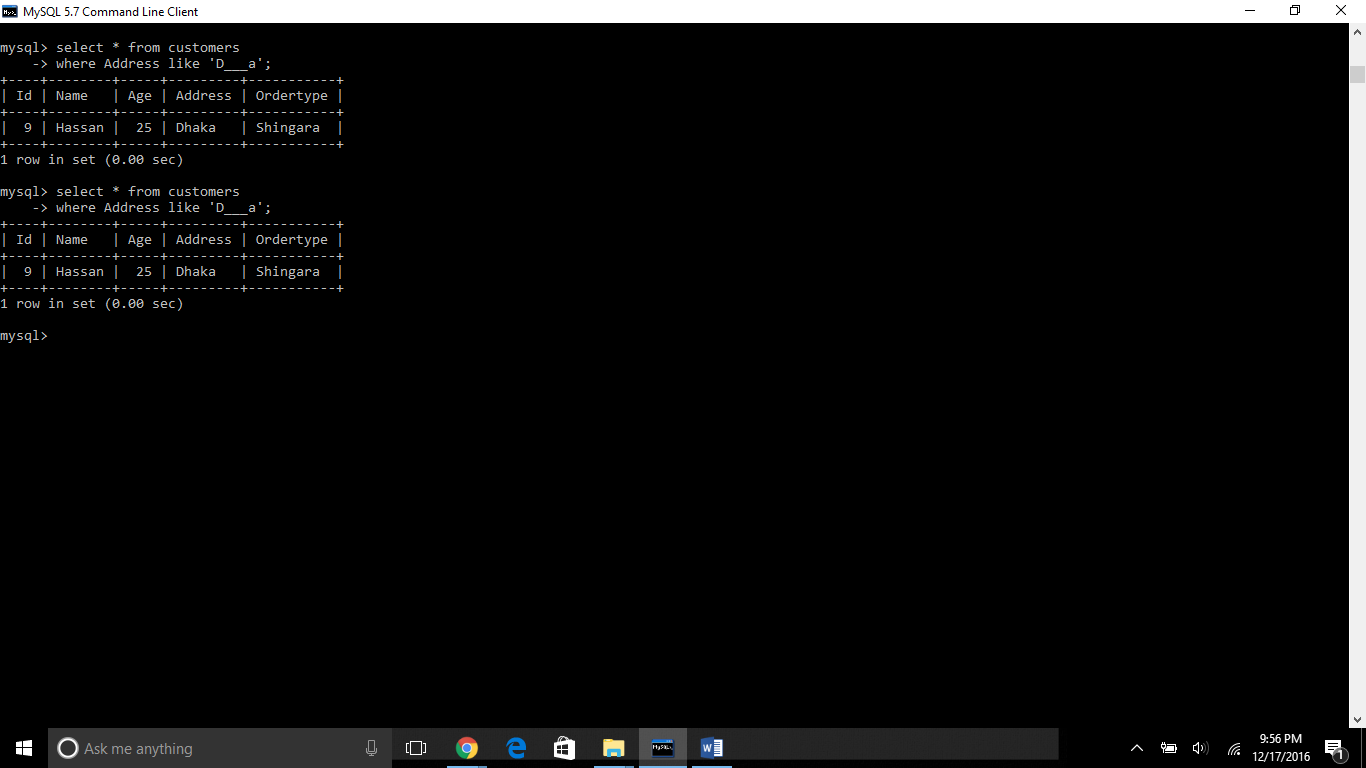
Same example byt different result when I used address insteda of ordertype.



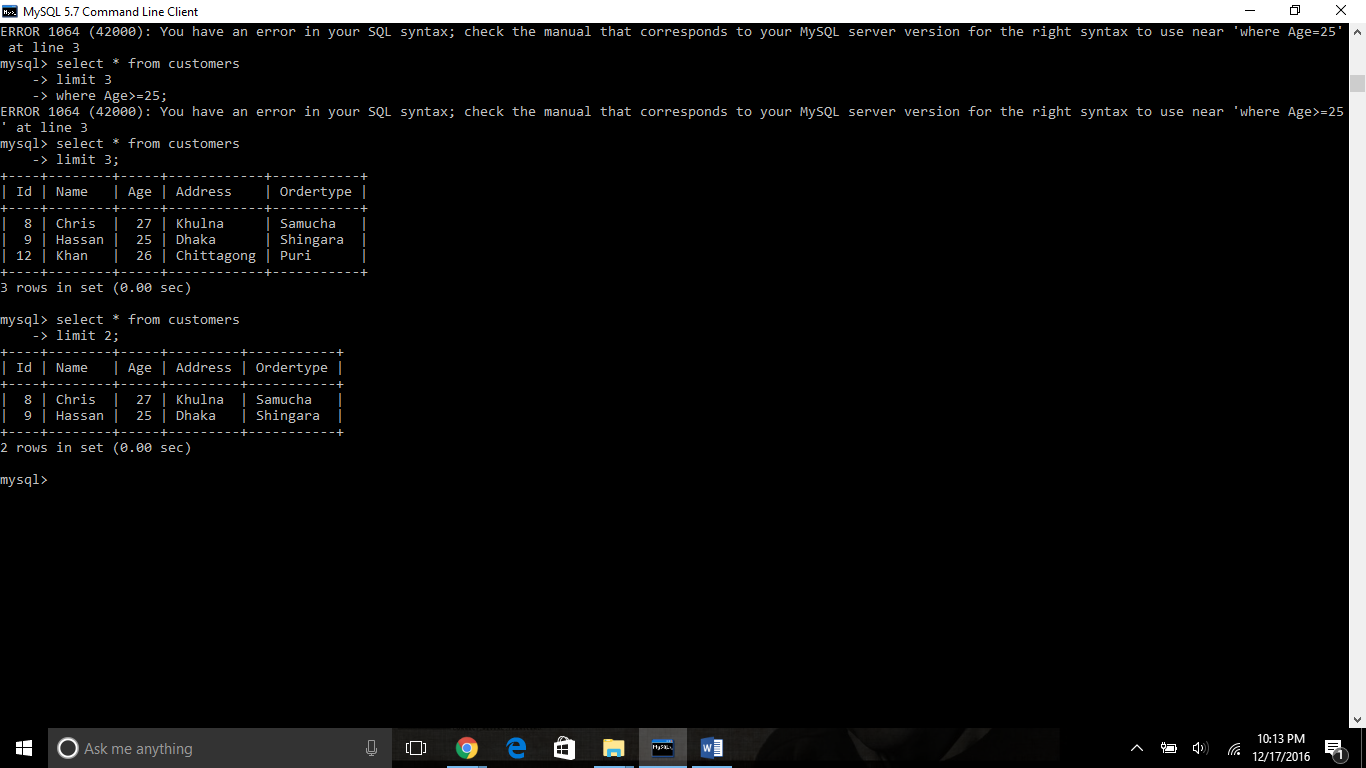


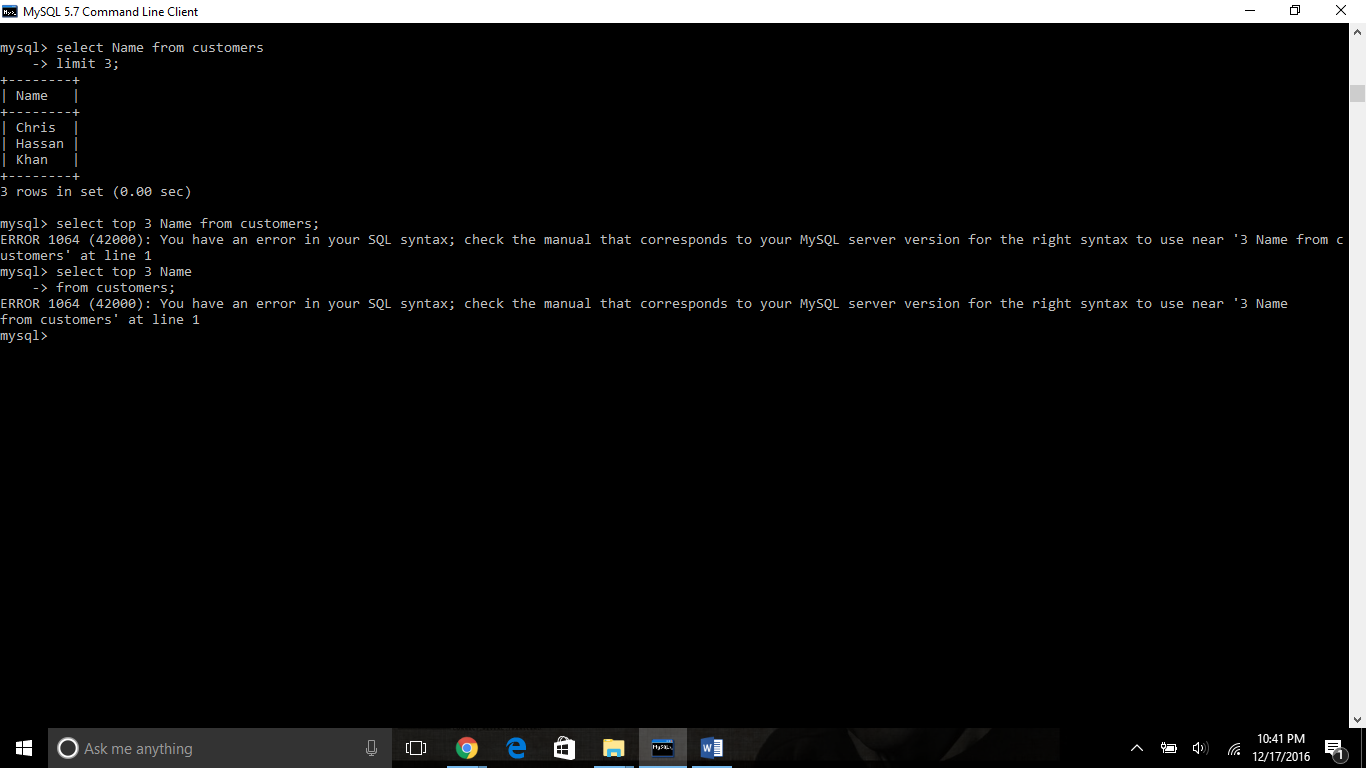
This is an example where I wanted to the third character of any address is a. That’s why I have used two underscore.



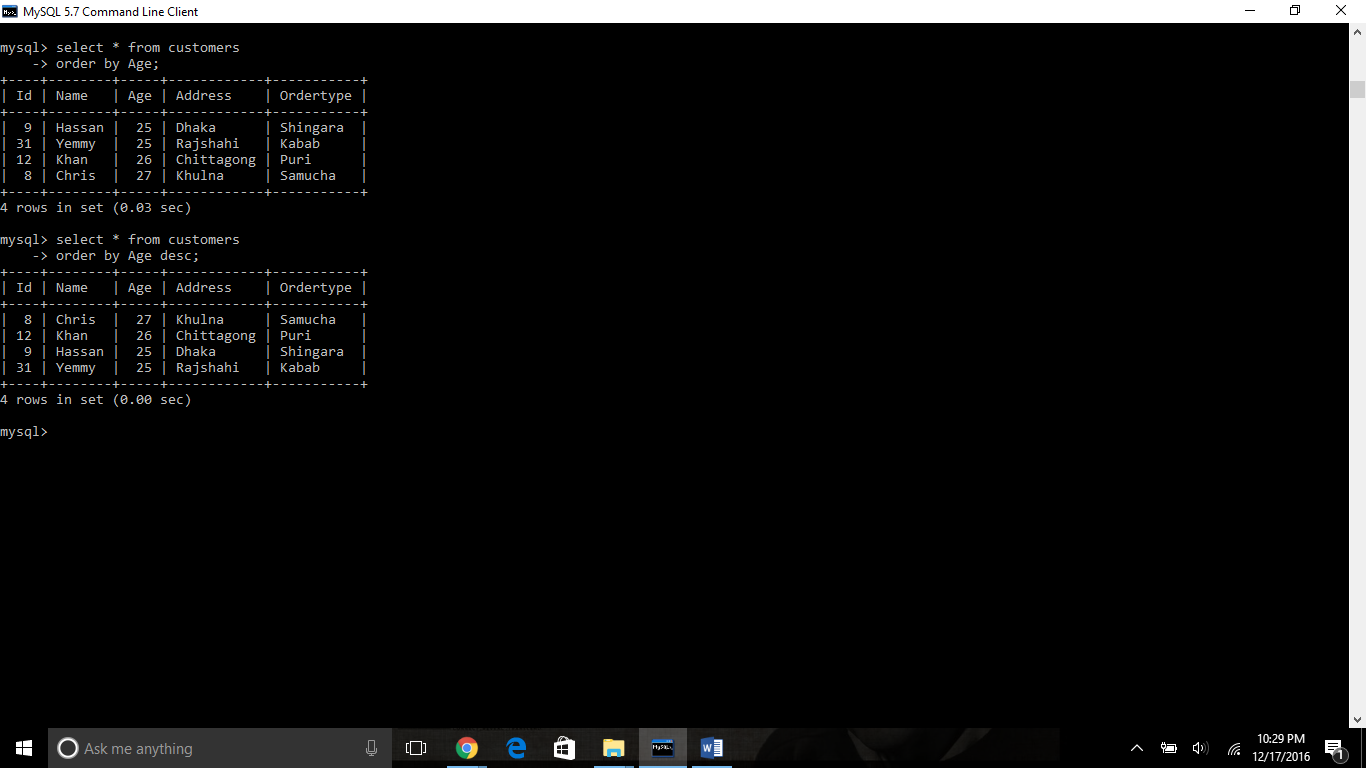


Top/Limit clause: yo choose the top n numbers of row/record

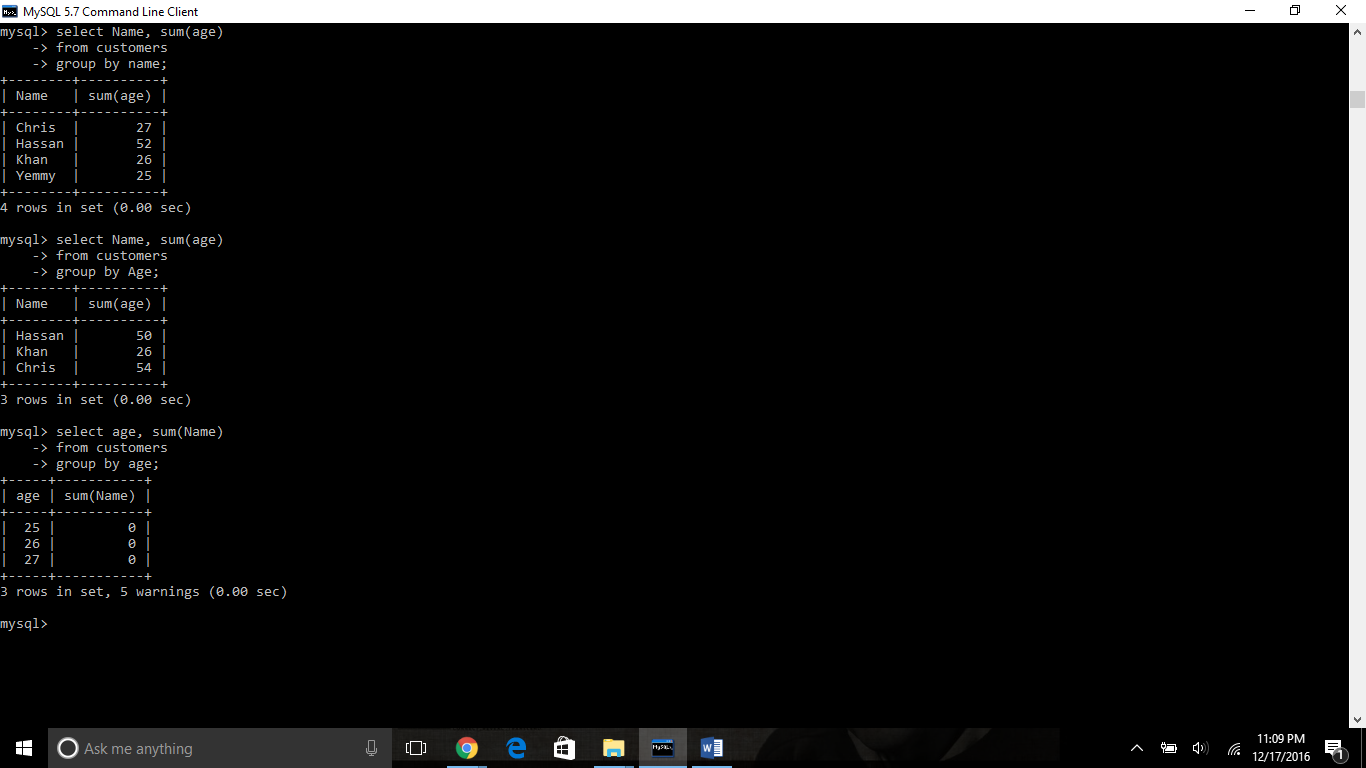




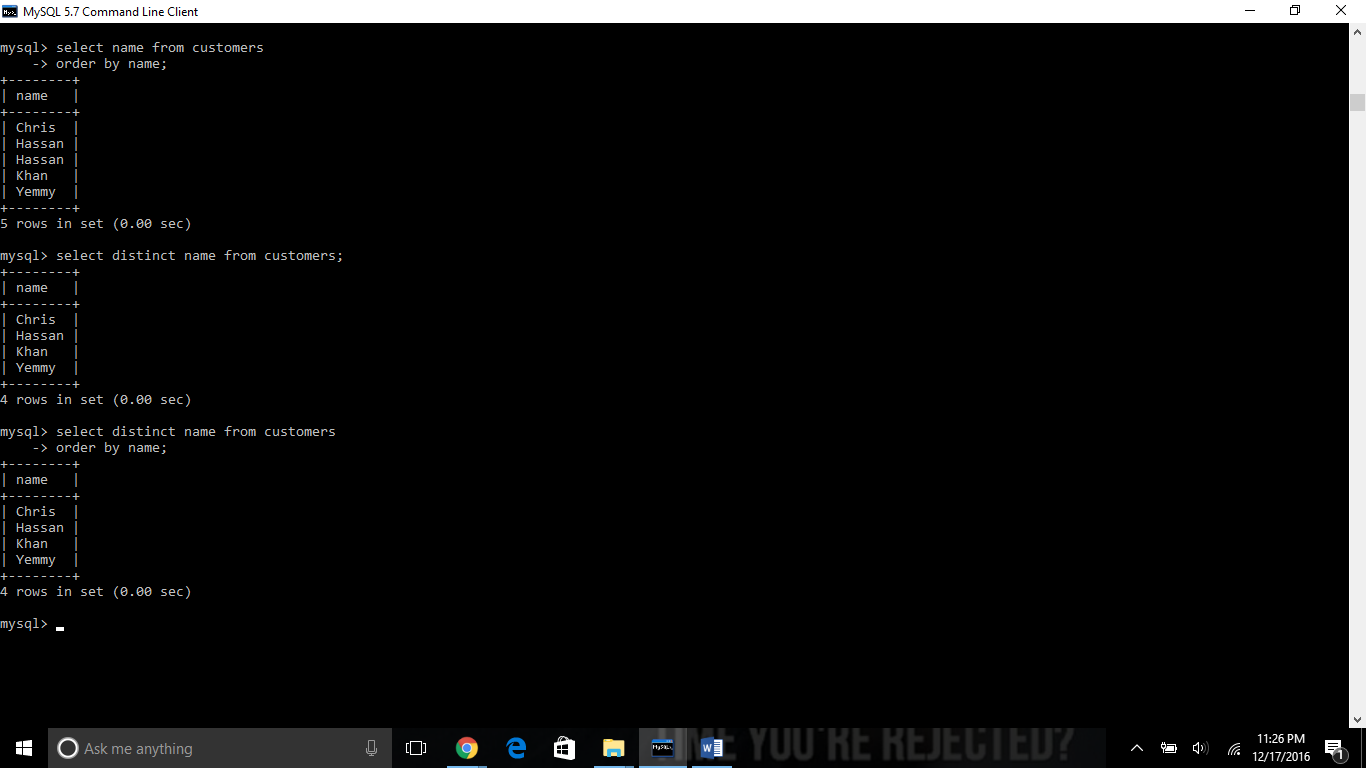
Order by clause: It is used to sort out data in ascending order or descending order

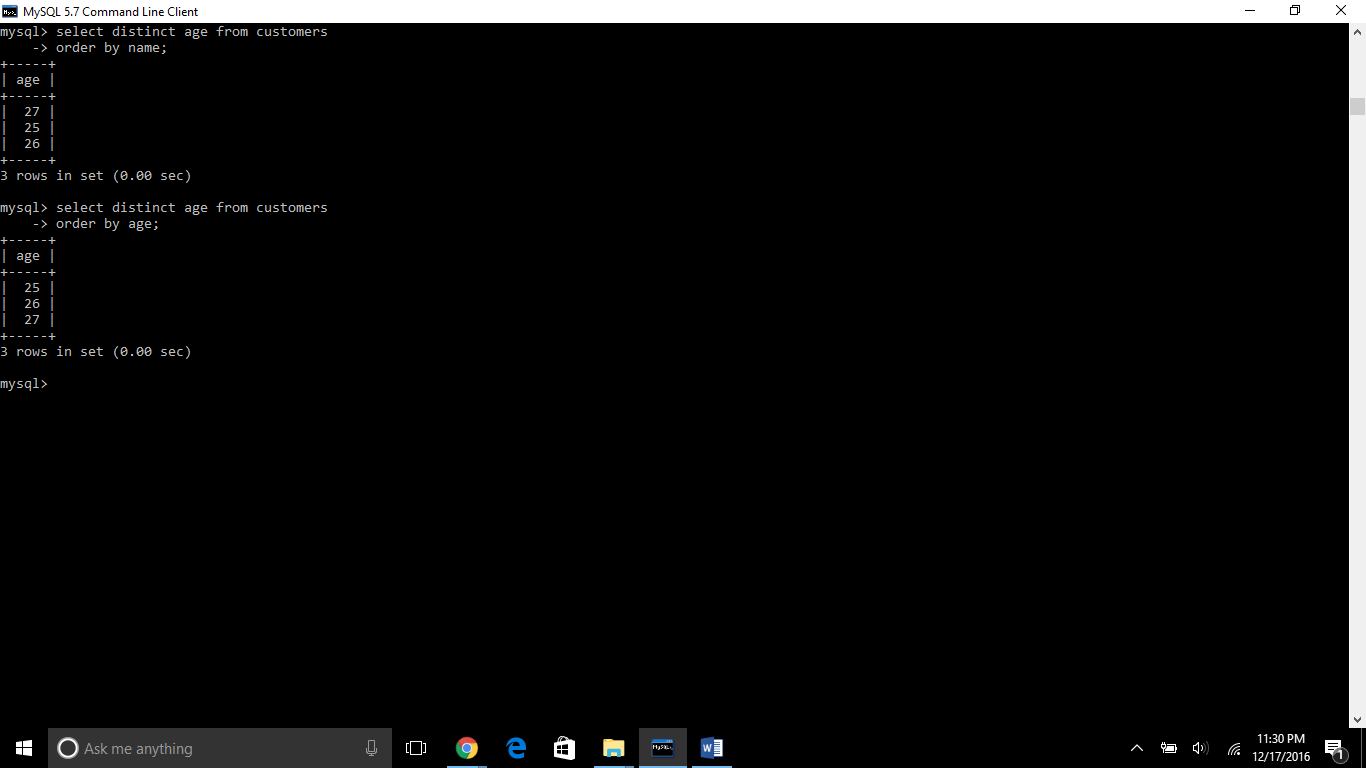


Group by clause



Distinct clause: it only choose on unique values and remove the duplicates.

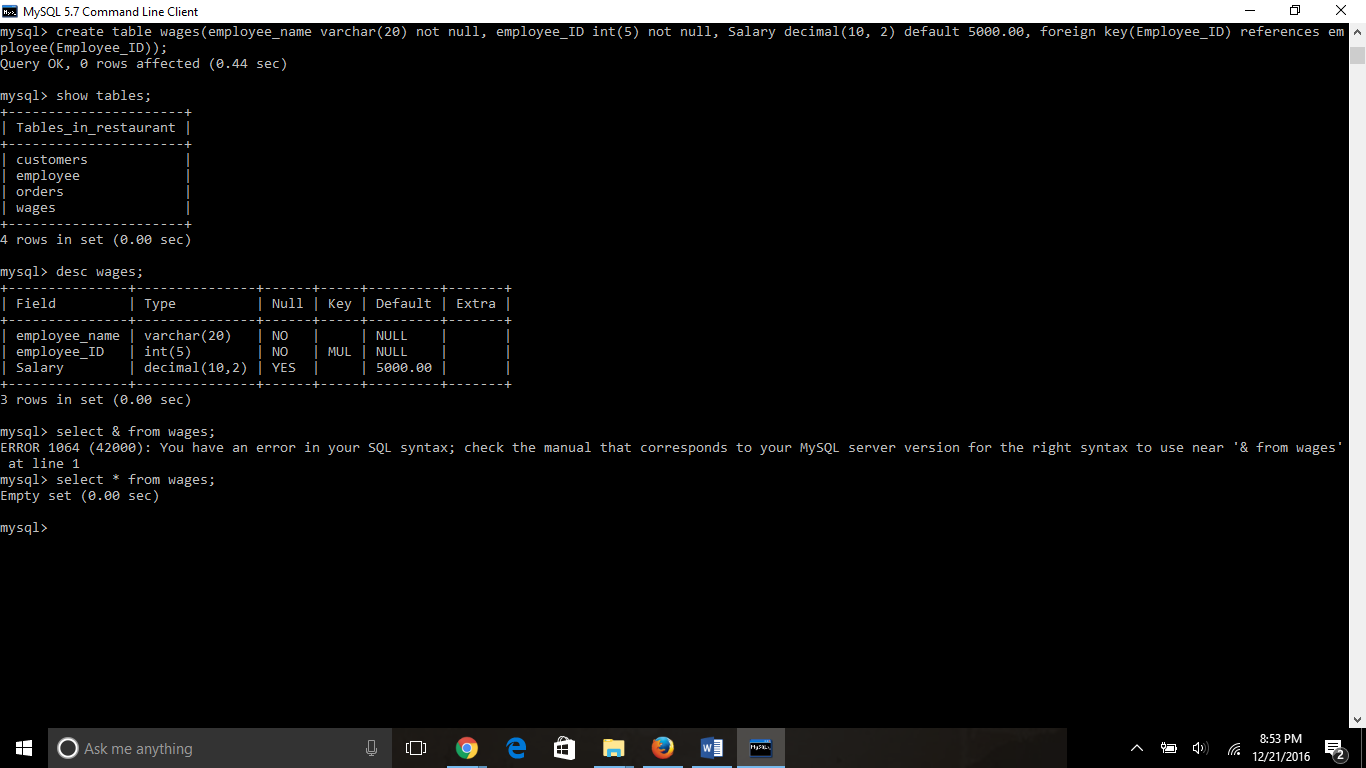




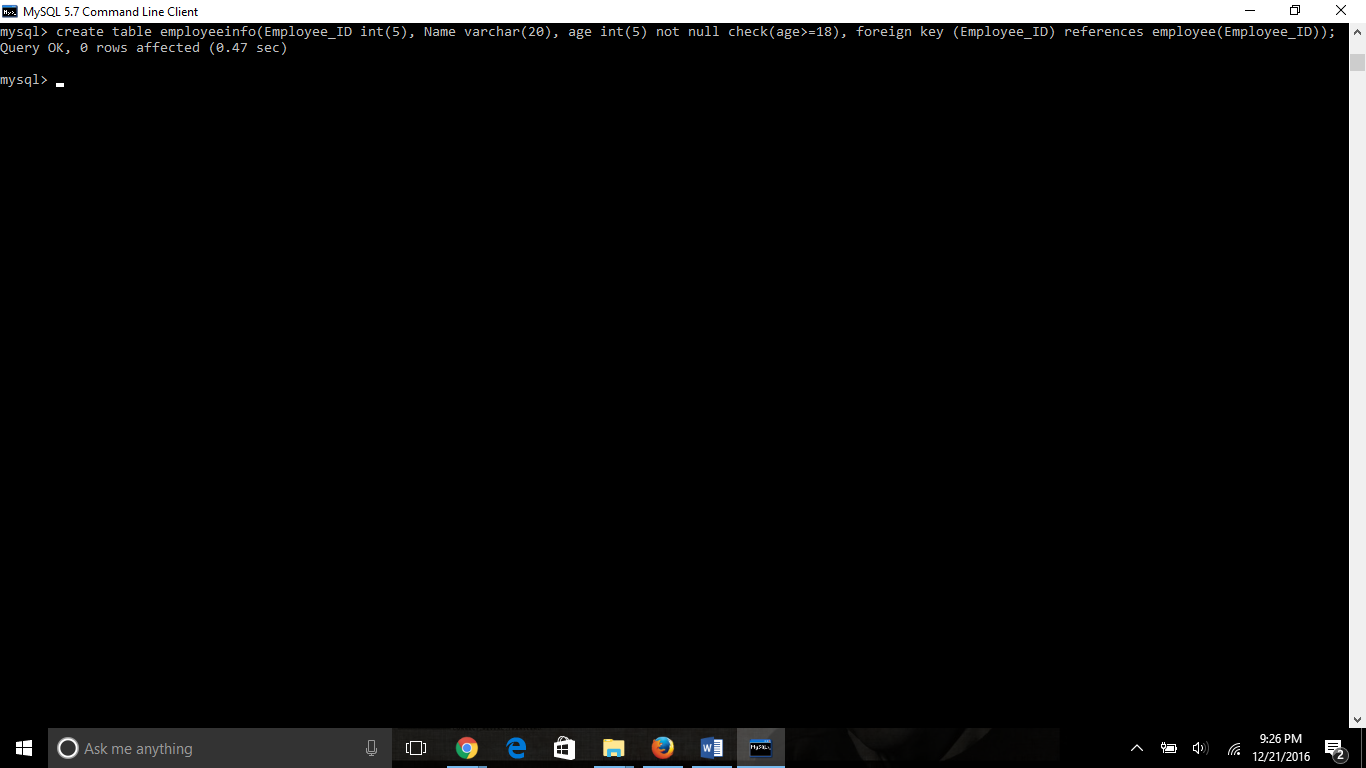
**Constraints:** Constraints are used to limit the fault in the data inside the table. Constraints are two types table level and column level. Column levels are applied only to one column table level constraint are applied to the whole table.

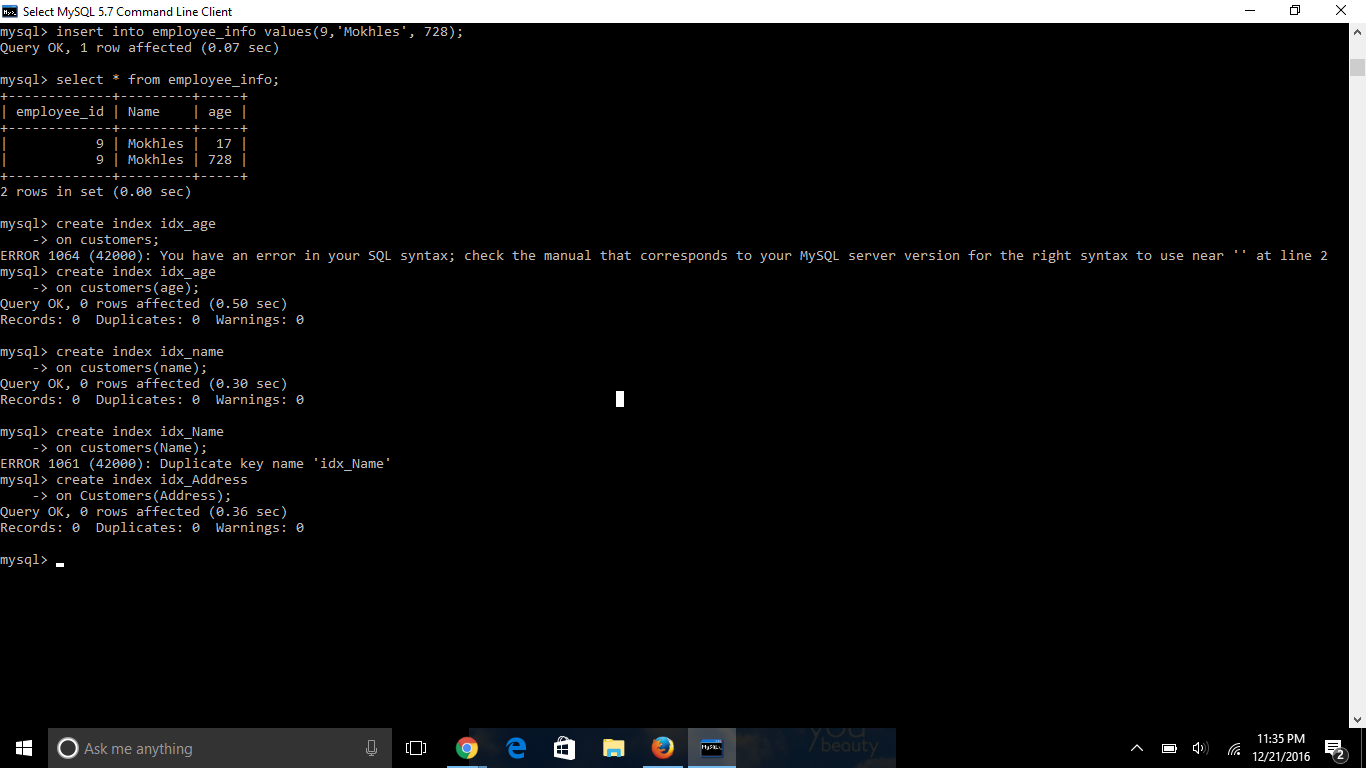
1. Not Null constraints: a data cannot be empty (Null means unknown or vacant).

2. Default Constraints: it makes the data of any tables default



Check constraints with create table

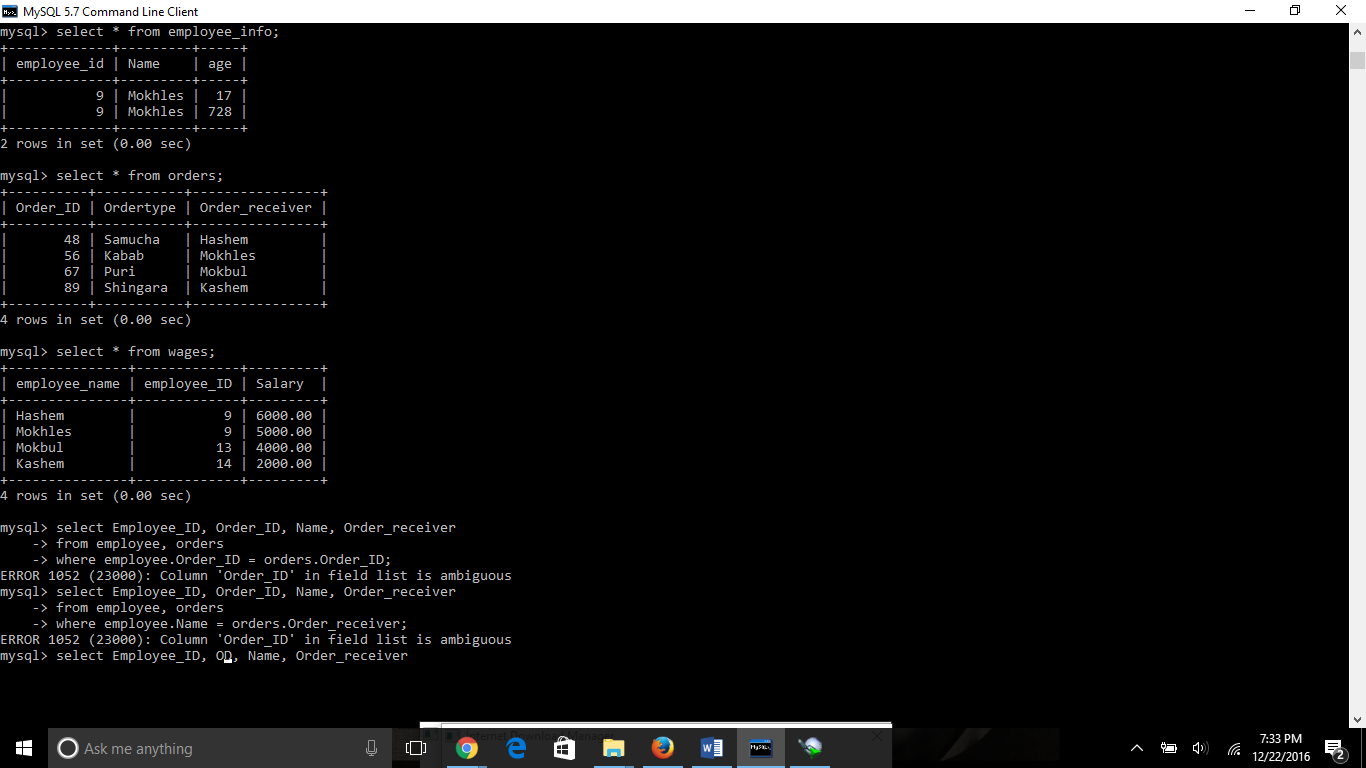




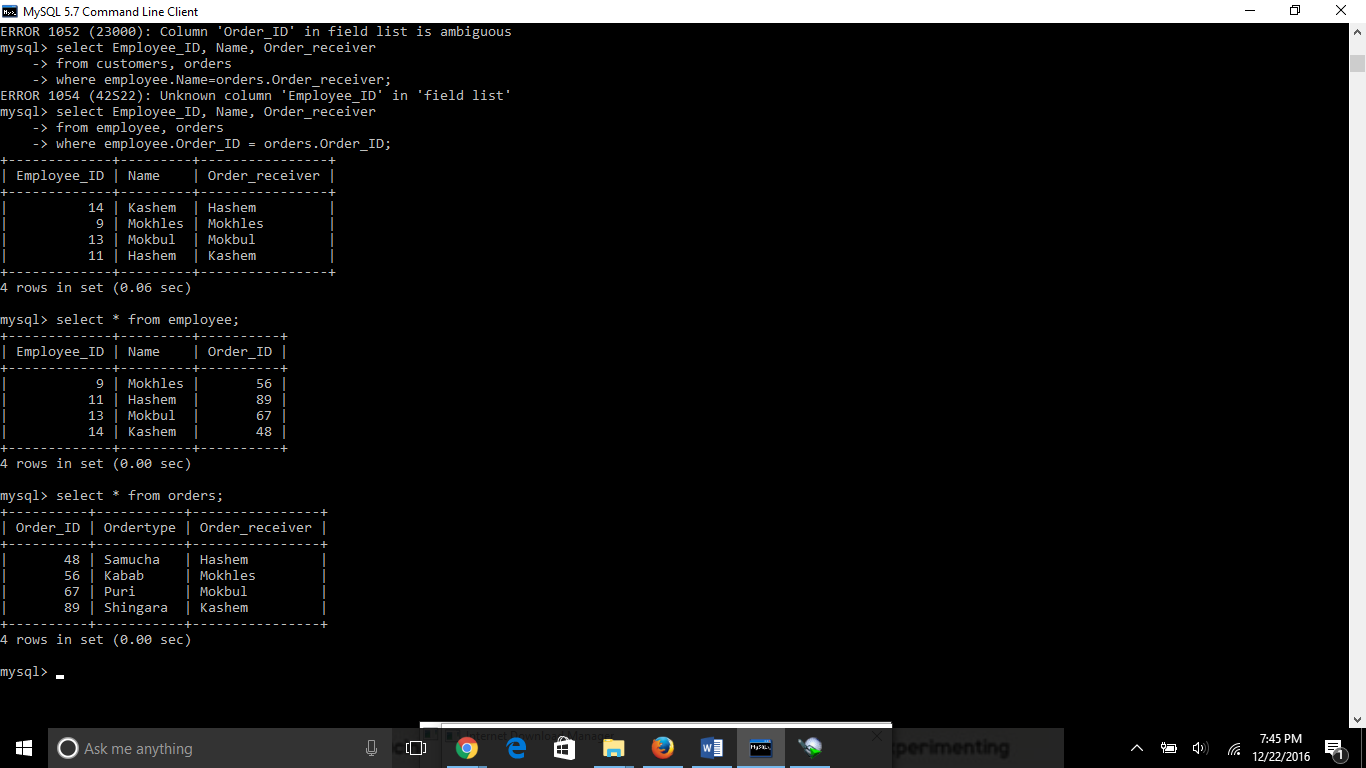
**SQL join**

Sql join is used to combine two or more tables with the same values.

Following is example of the error which shows the ambiguous of the order\_id

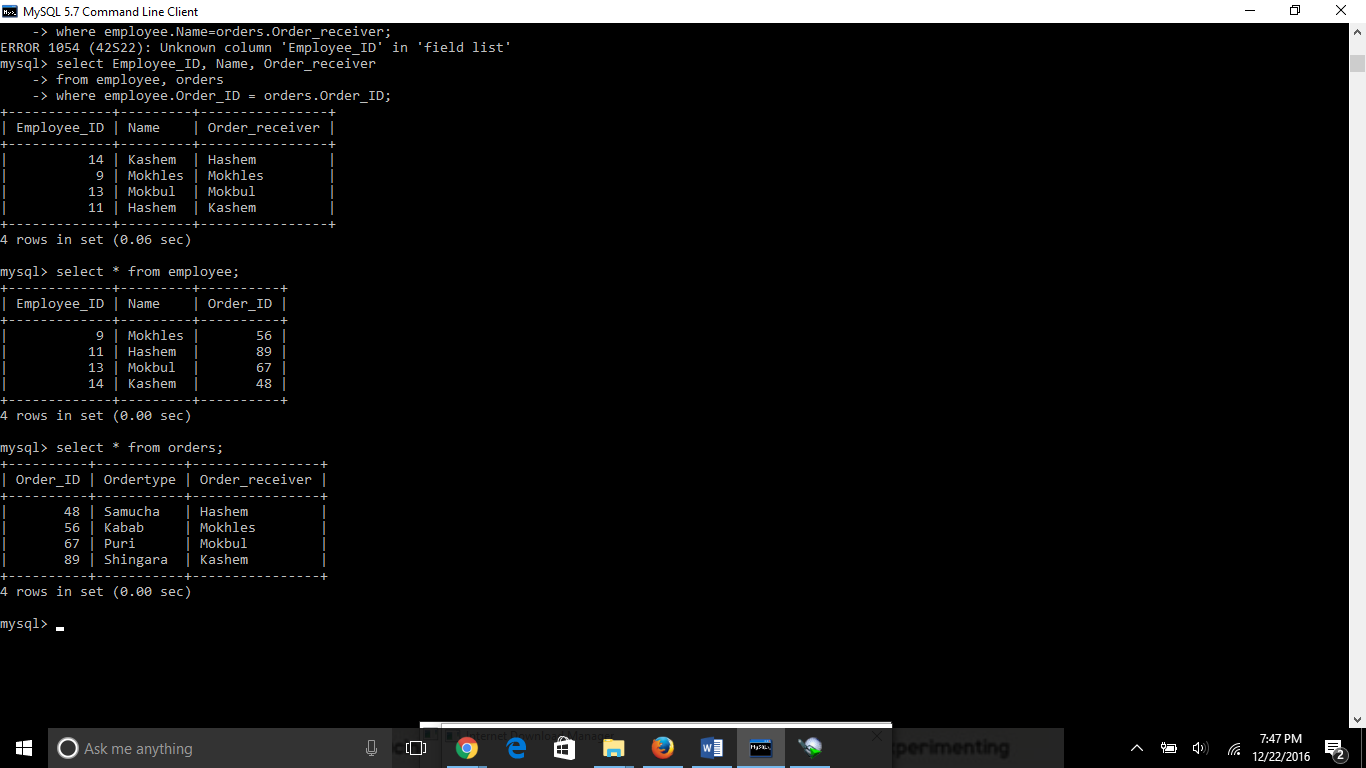


**Another mistake**

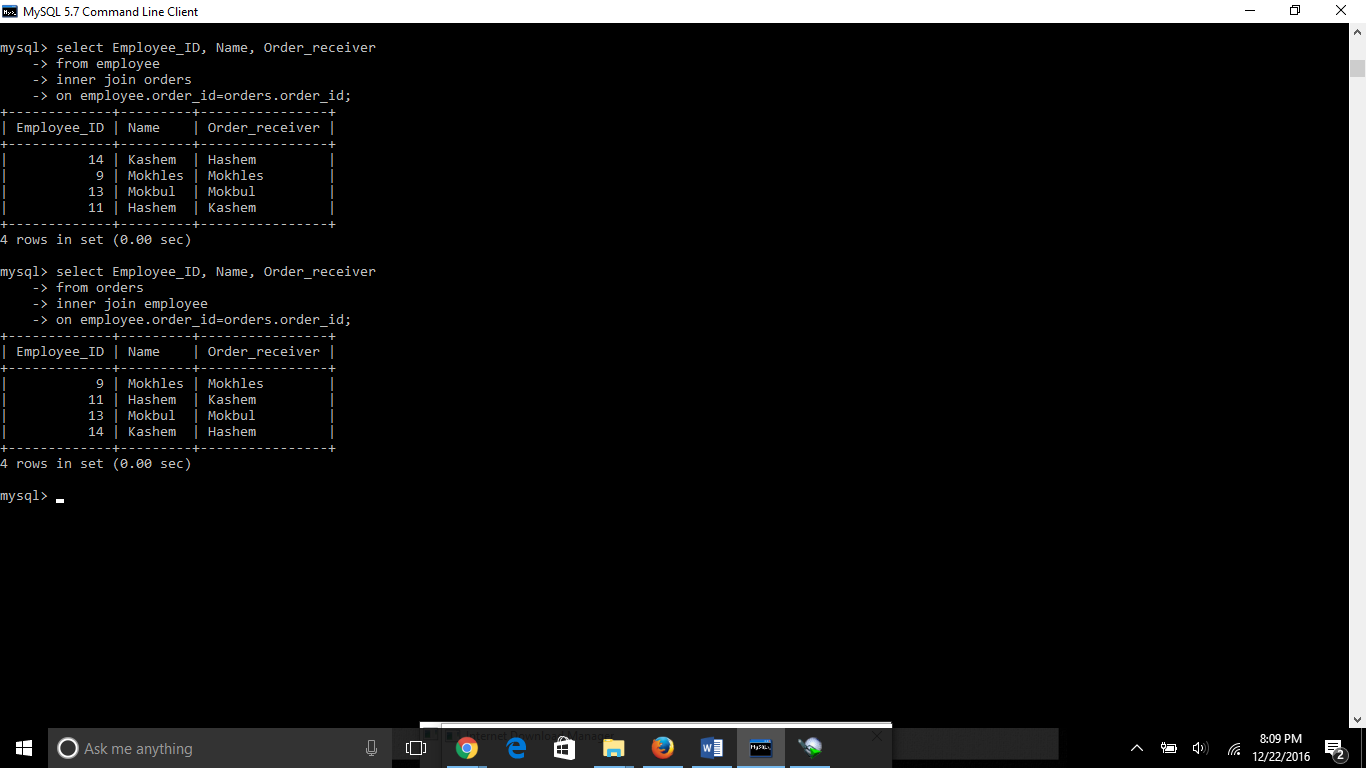


Because I choose the wrong name of the table in the join syntax. The ambigious mistake occurs when we put wrong values in the same field but in different tables.

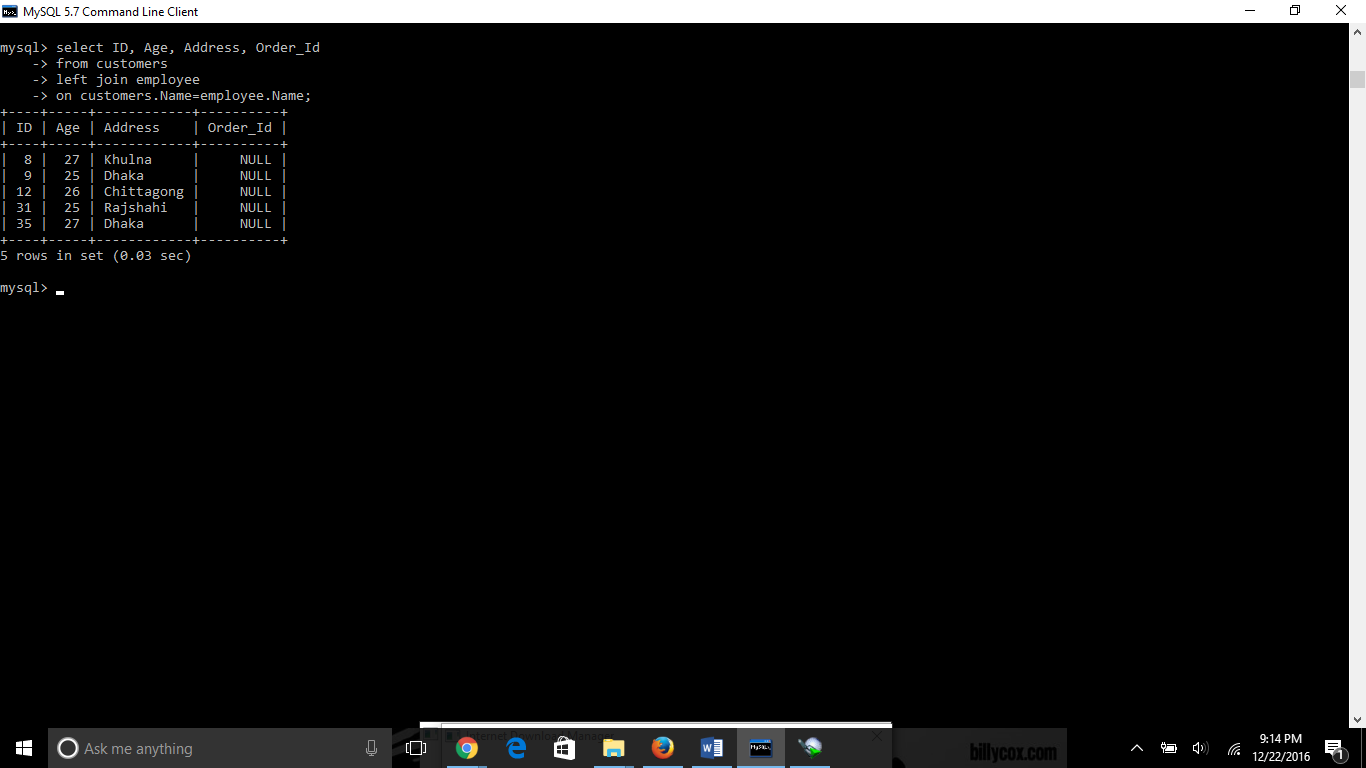
Here I choose thw right syntax



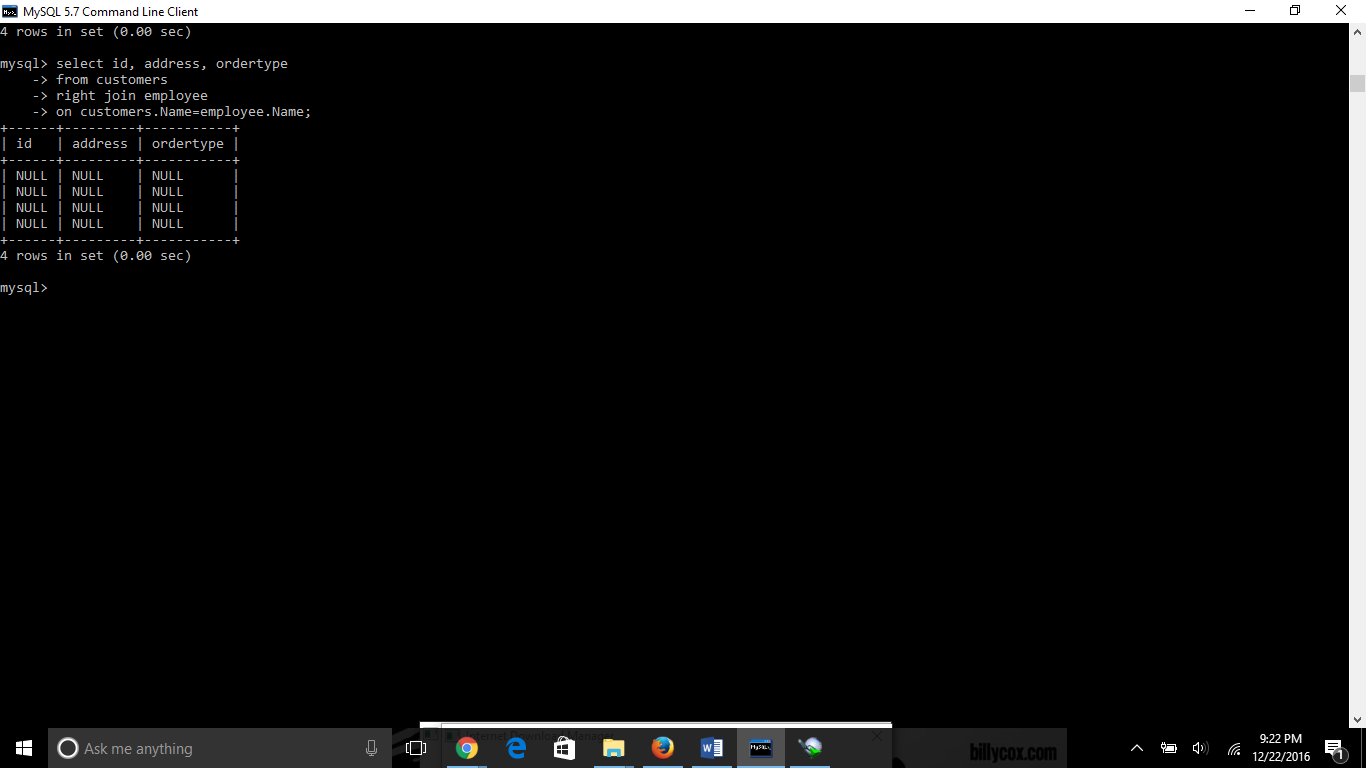
**Inner join :** there is less difference between inner join and simple equal join



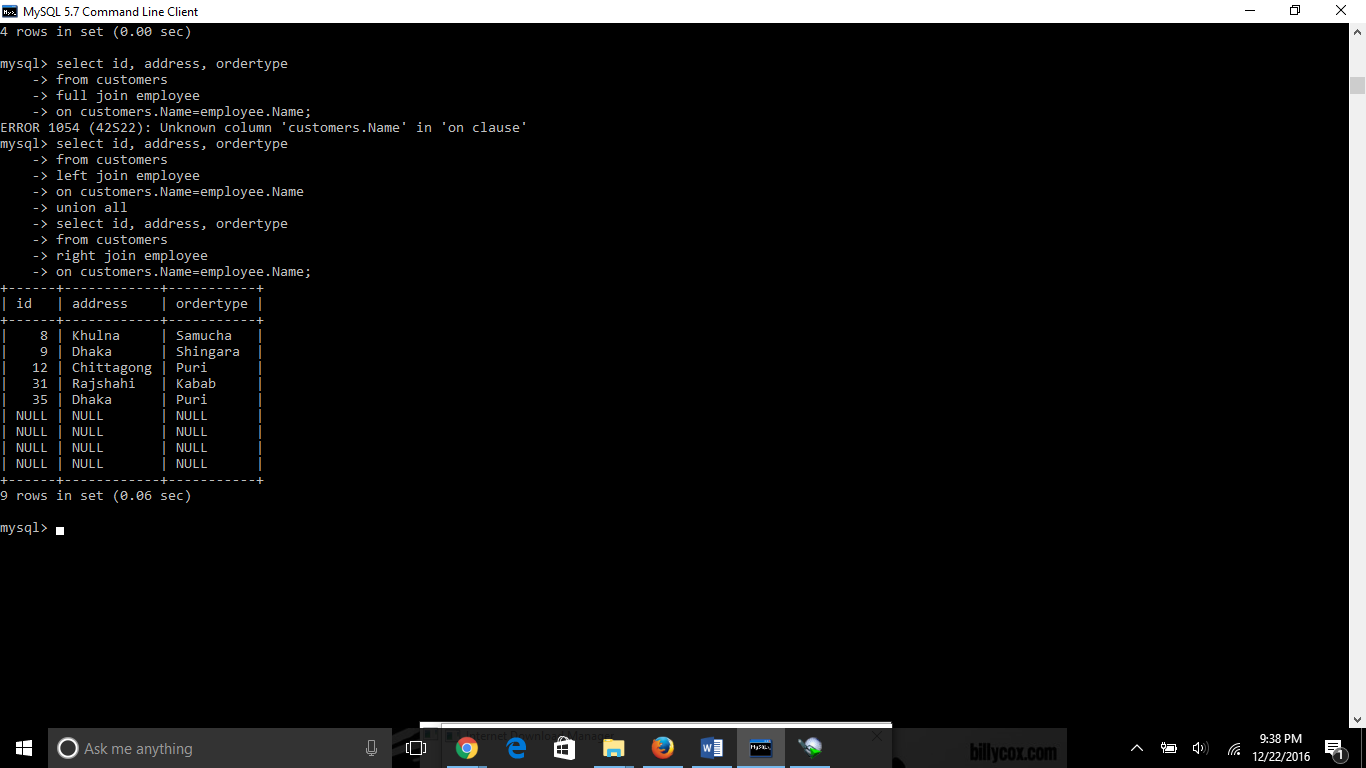
**Left join:**  when there is no match it also show values



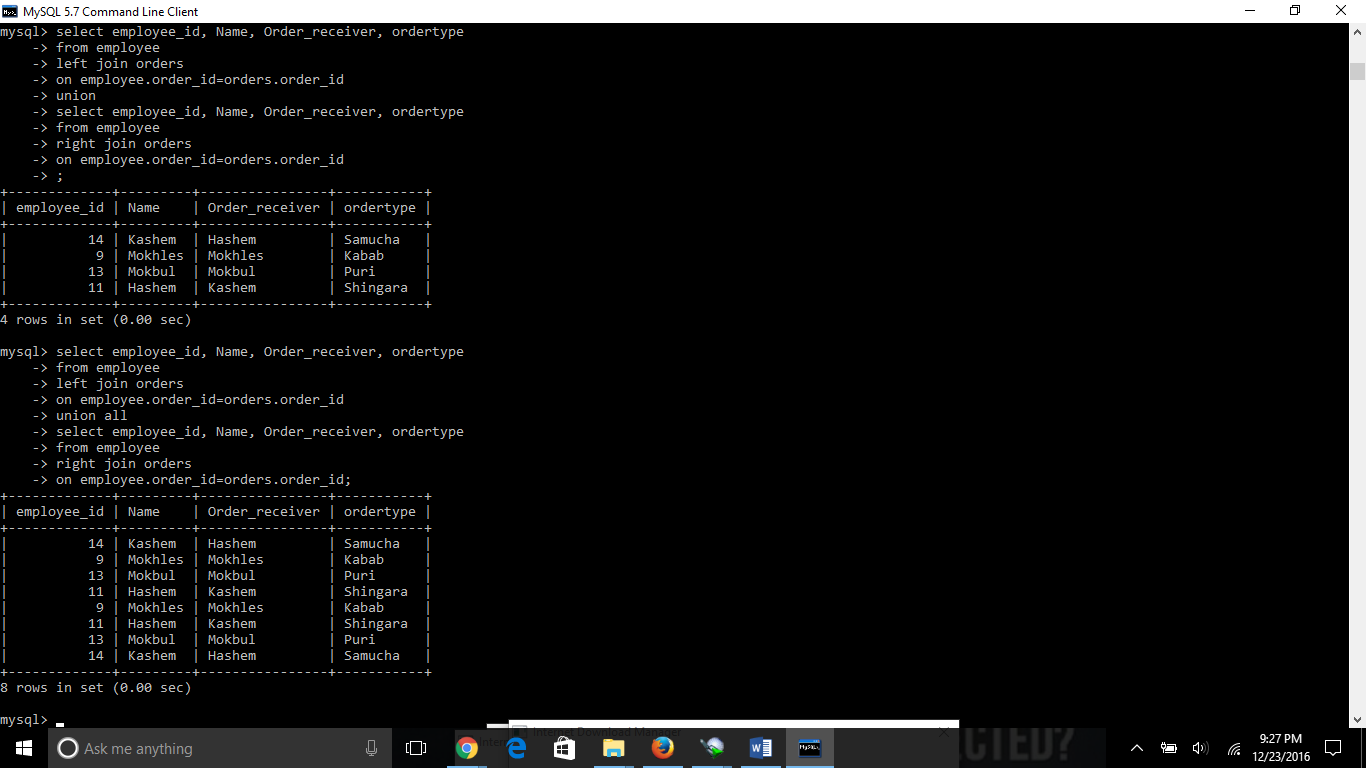
**Right join:**



**Full Join:** as we are working on My SQL instead of SQL we have to use “UNION ALL” like below



Union Clause: difference between union and union clause



Intersect and except clause: in MY sql intersect and except does not work



Null values:

+----+----------+-----+-----------+----------+

| ID | NAME | AGE | ADDRESS | SALARY |

+----+----------+-----+-----------+----------+

| 1 | Ramesh | 32 | Ahmedabad | 2000.00 |

| 2 | Khilan | 25 | Delhi | 1500.00 |

| 3 | kaushik | 23 | Kota | 2000.00 |

| 4 | Chaitali | 25 | Mumbai | 6500.00 |

| 5 | Hardik | 27 | Bhopal | 8500.00 |

| 6 | Komal | 22 | MP | |

| 7 | Muffy | 24 | Indore | |

+----+----------+-----+-----------+----------+

Now, following is the usage of **IS NOT NULL** operator:

SQL> SELECT ID, NAME, AGE, ADDRESS, SALARY

FROM CUSTOMERS

WHERE SALARY IS NOT NULL;

This would produce the following result:

+----+----------+-----+-----------+----------+

| ID | NAME | AGE | ADDRESS | SALARY |

+----+----------+-----+-----------+----------+

| 1 | Ramesh | 32 | Ahmedabad | 2000.00 |

| 2 | Khilan | 25 | Delhi | 1500.00 |

| 3 | kaushik | 23 | Kota | 2000.00 |

| 4 | Chaitali | 25 | Mumbai | 6500.00 |

| 5 | Hardik | 27 | Bhopal | 8500.00 |

+----+----------+-----+-----------+----------+

Now, following is the usage of **IS NULL** operator:

SQL> SELECT ID, NAME, AGE, ADDRESS, SALARY

FROM CUSTOMERS

WHERE SALARY IS NULL;

This would produce the following result:

+----+----------+-----+-----------+----------+

| ID | NAME | AGE | ADDRESS | SALARY |

+----+----------+-----+-----------+----------+

| 6 | Komal | 22 | MP | |

| 7 | Muffy | 24 | Indore | |

+----+----------+-----+-----------+----------+

Changing table name using alias. Alias is using for changing the table or column name temporary.

**Table alias**



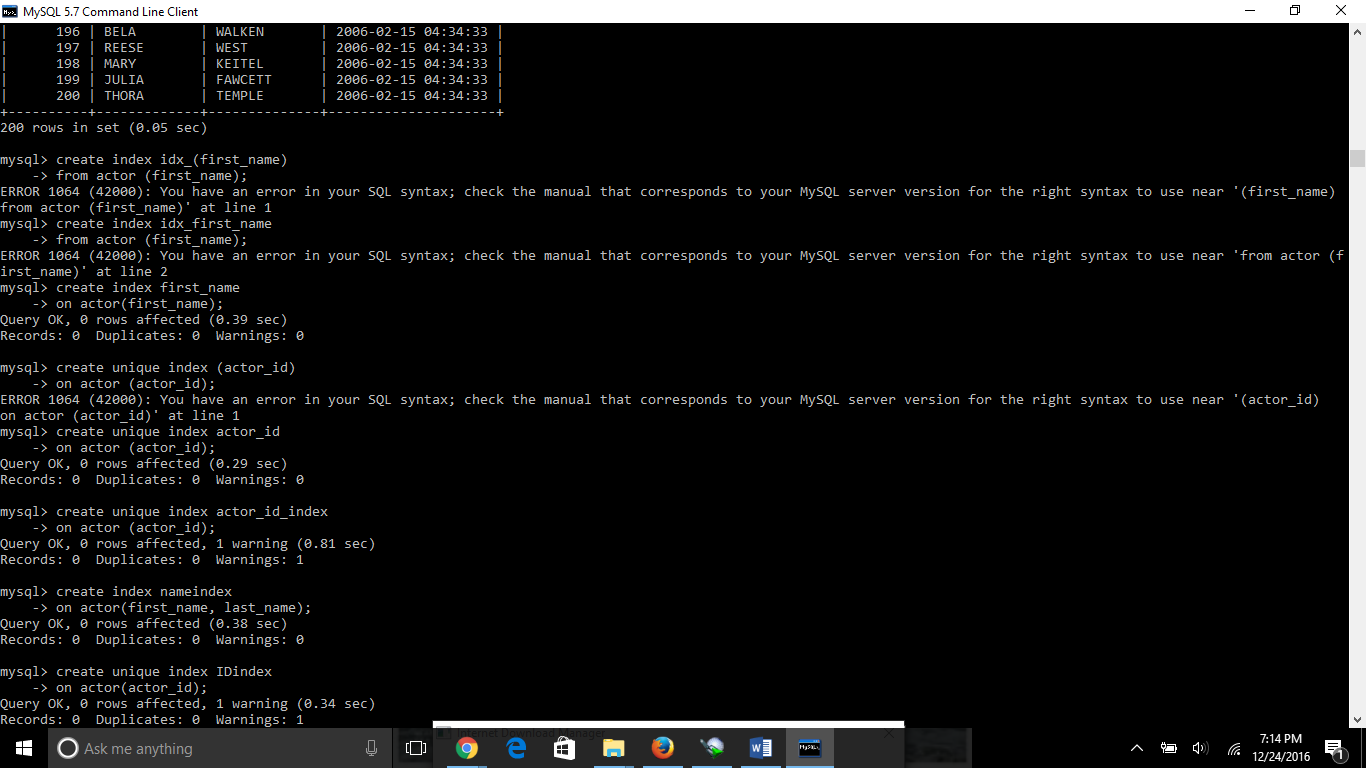
**Column alias**



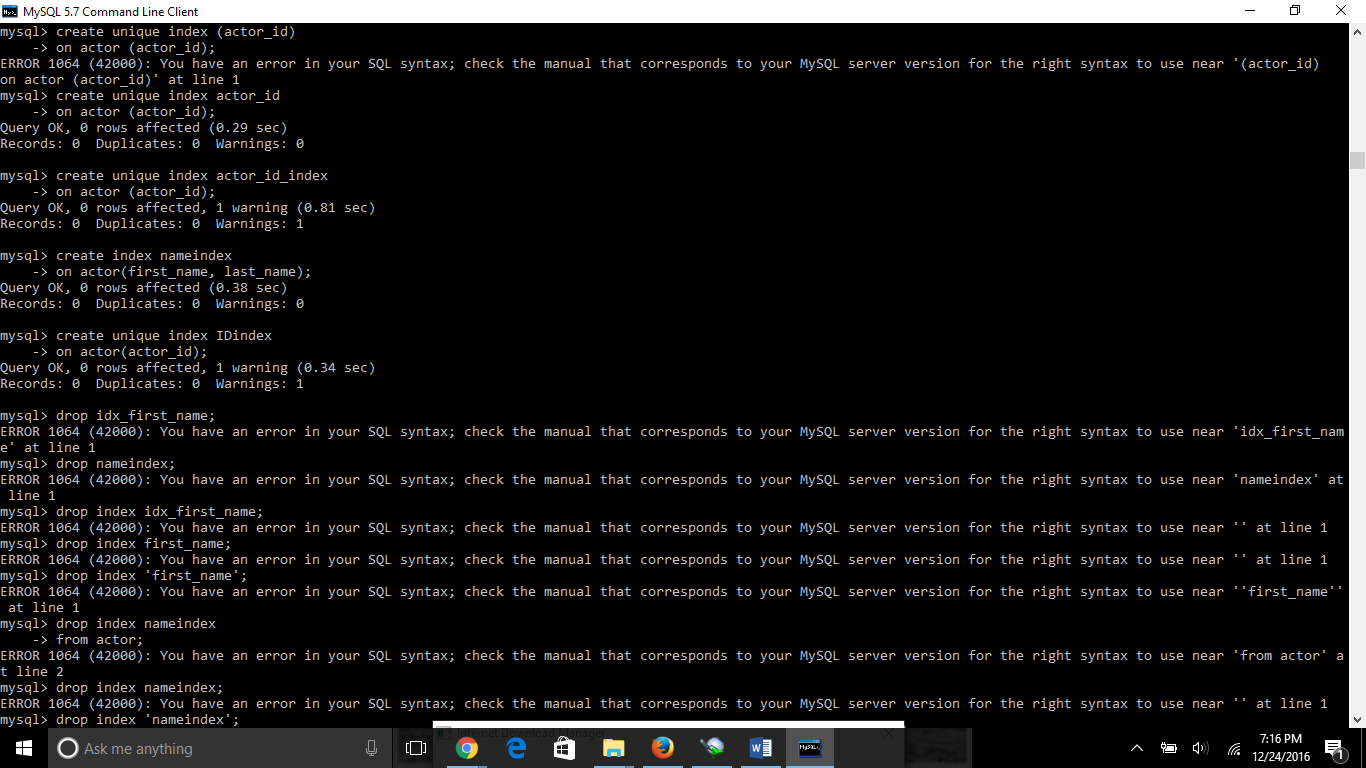
**Index:**

* Indexing is used to search and find out the data quickly.
* Always use index in a big table, and huge columns
* Use index on to those columns and tables which are more likely to be searched on frequently.
* The advantage of using index is it make faster the SELECT and WHERE clause but it slow down the UPDATE and INSERT statement.
* Usually the user cannot see the result of the index
* Updating the column with index takes more time.

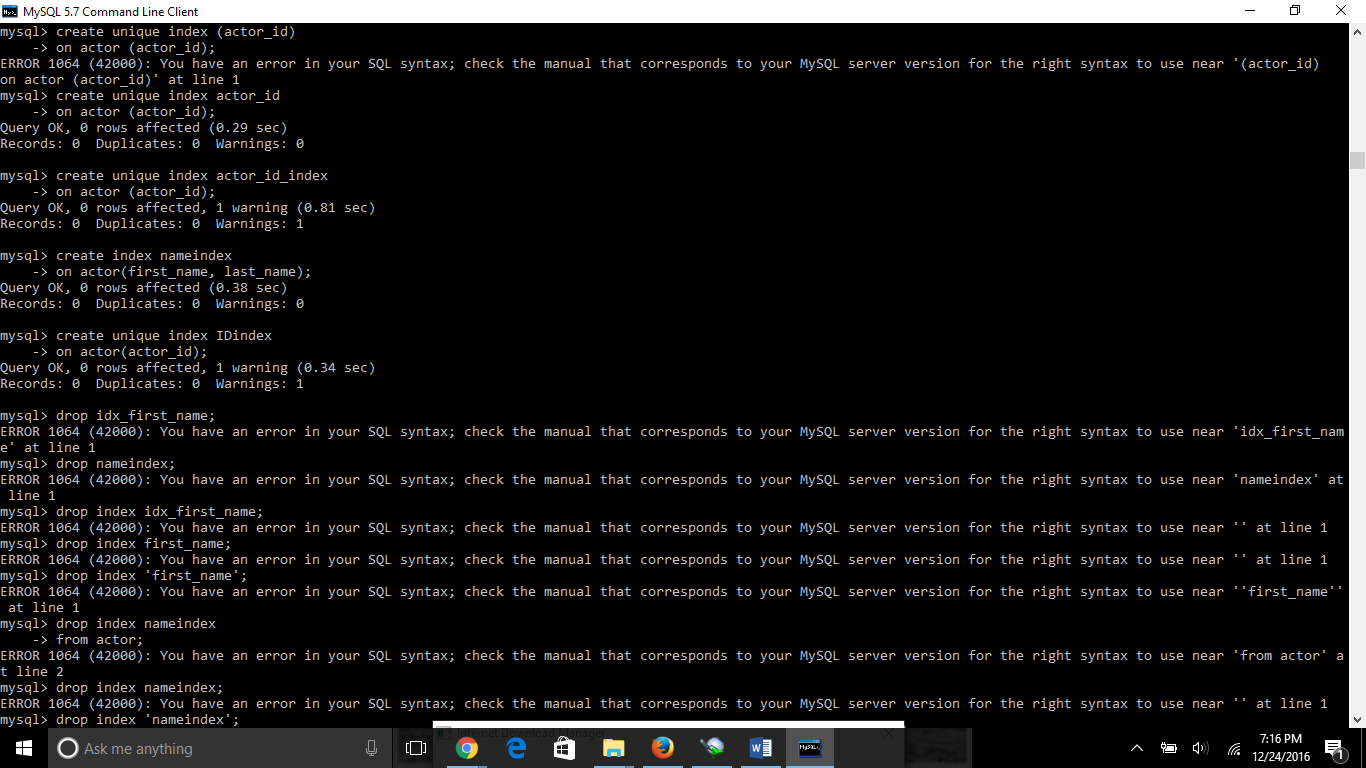
Single column index



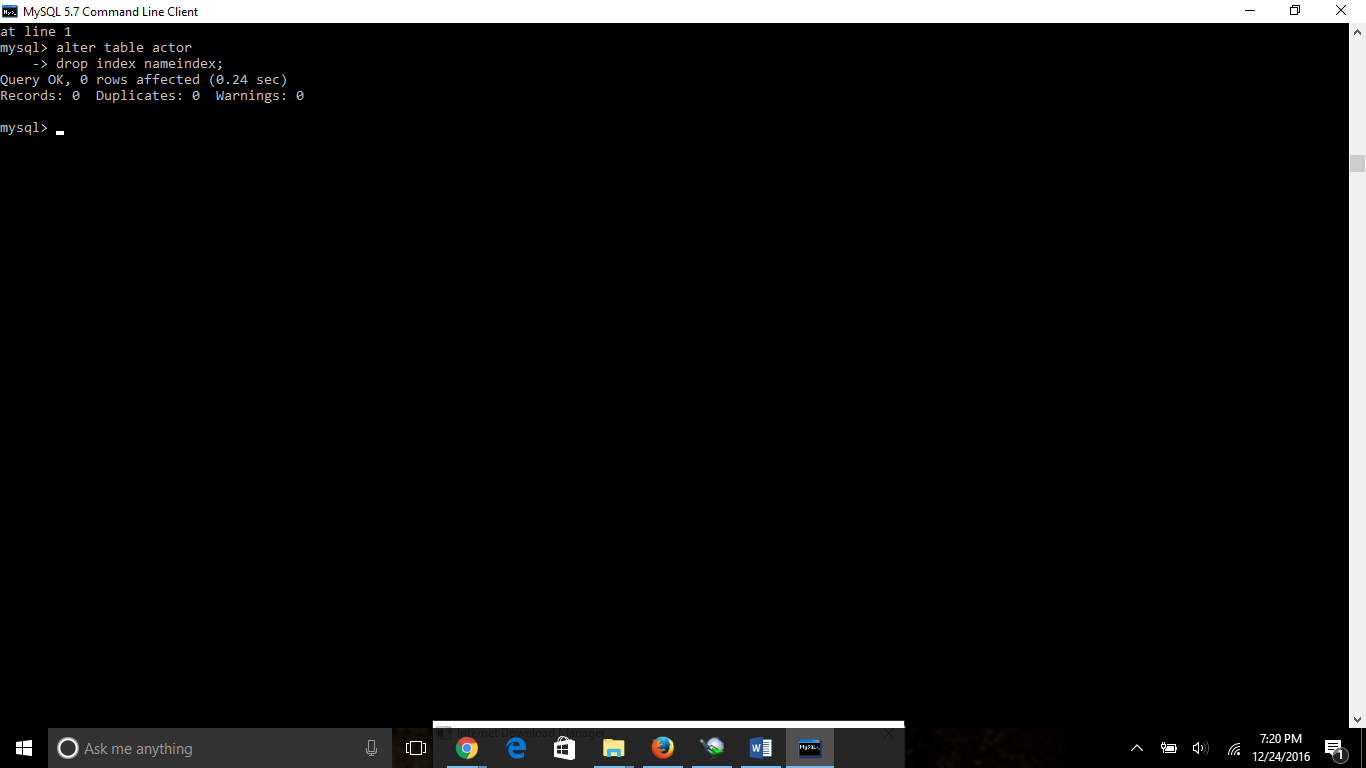
Unique index



Multiple index



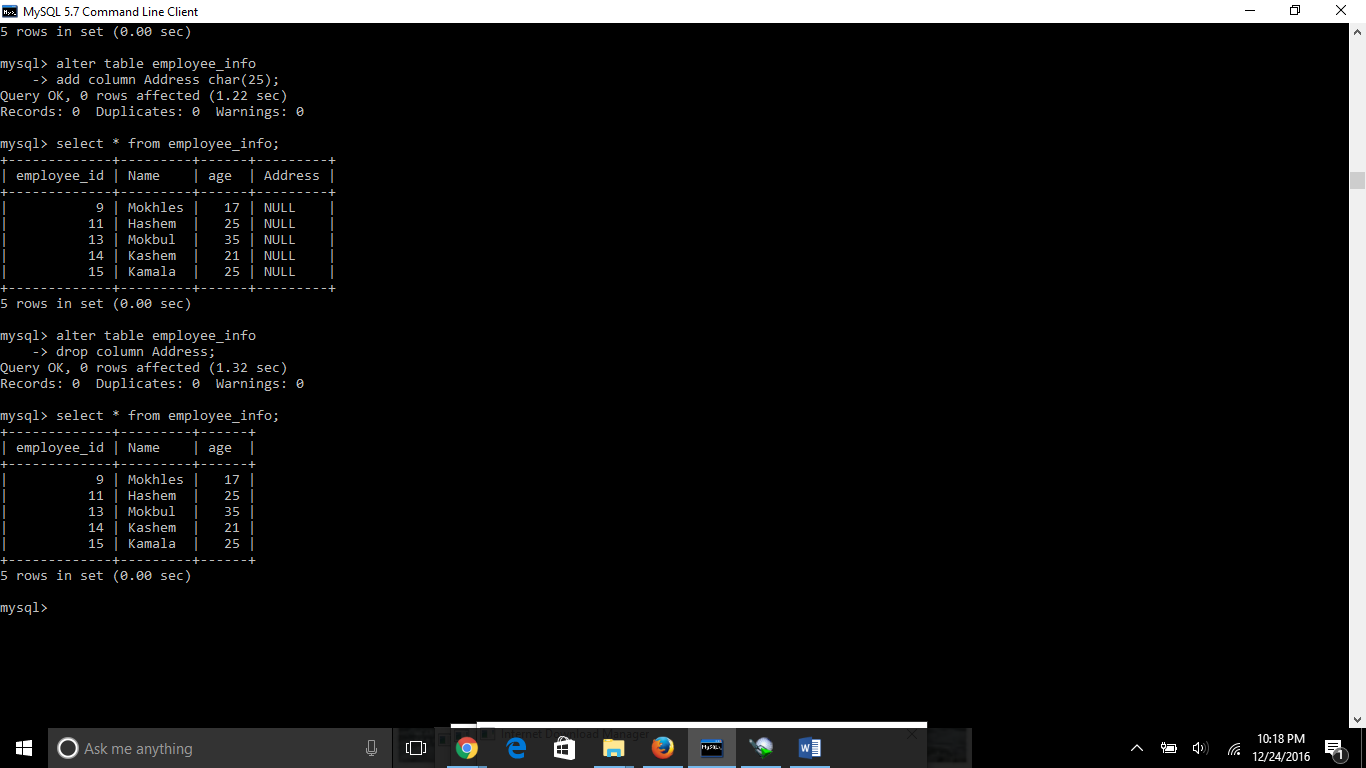
Drop index



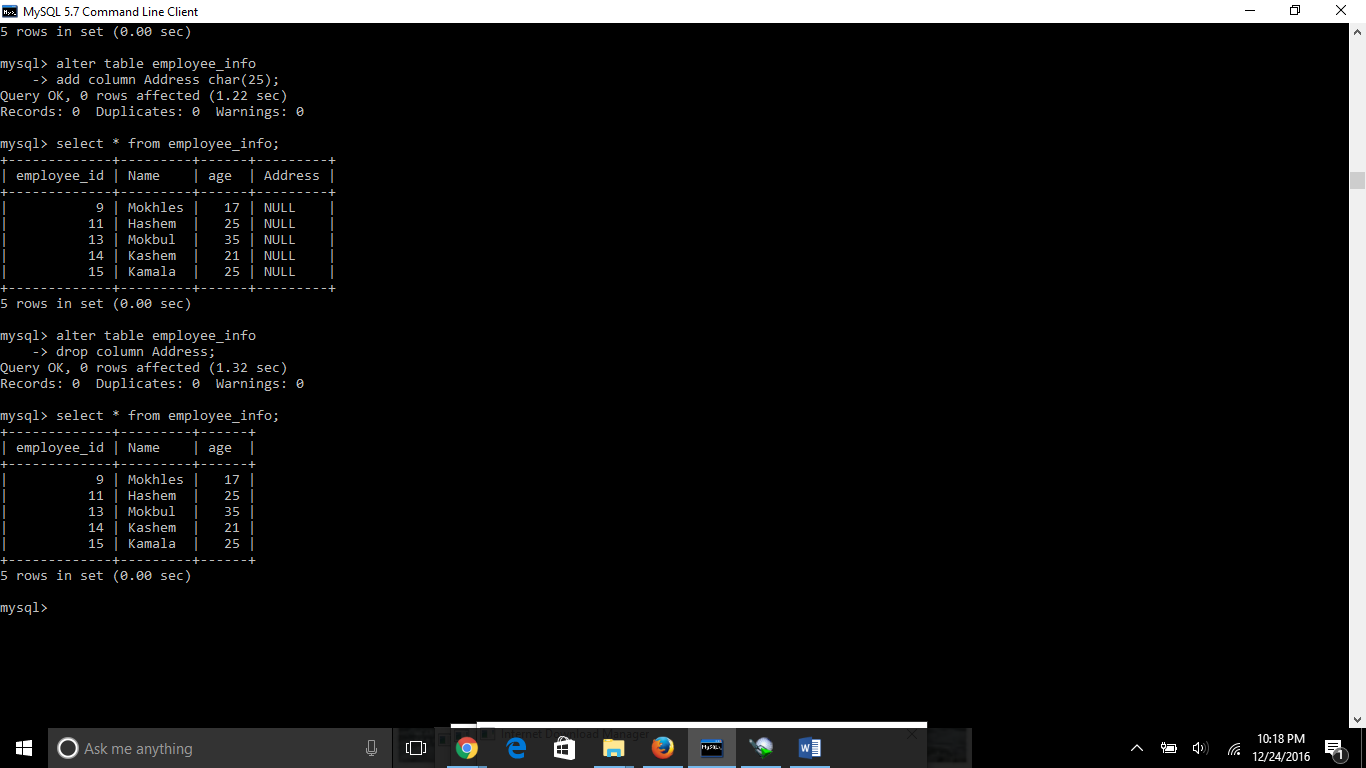
Alter Command:

Alter command is used to add, drop, modify the column in a table

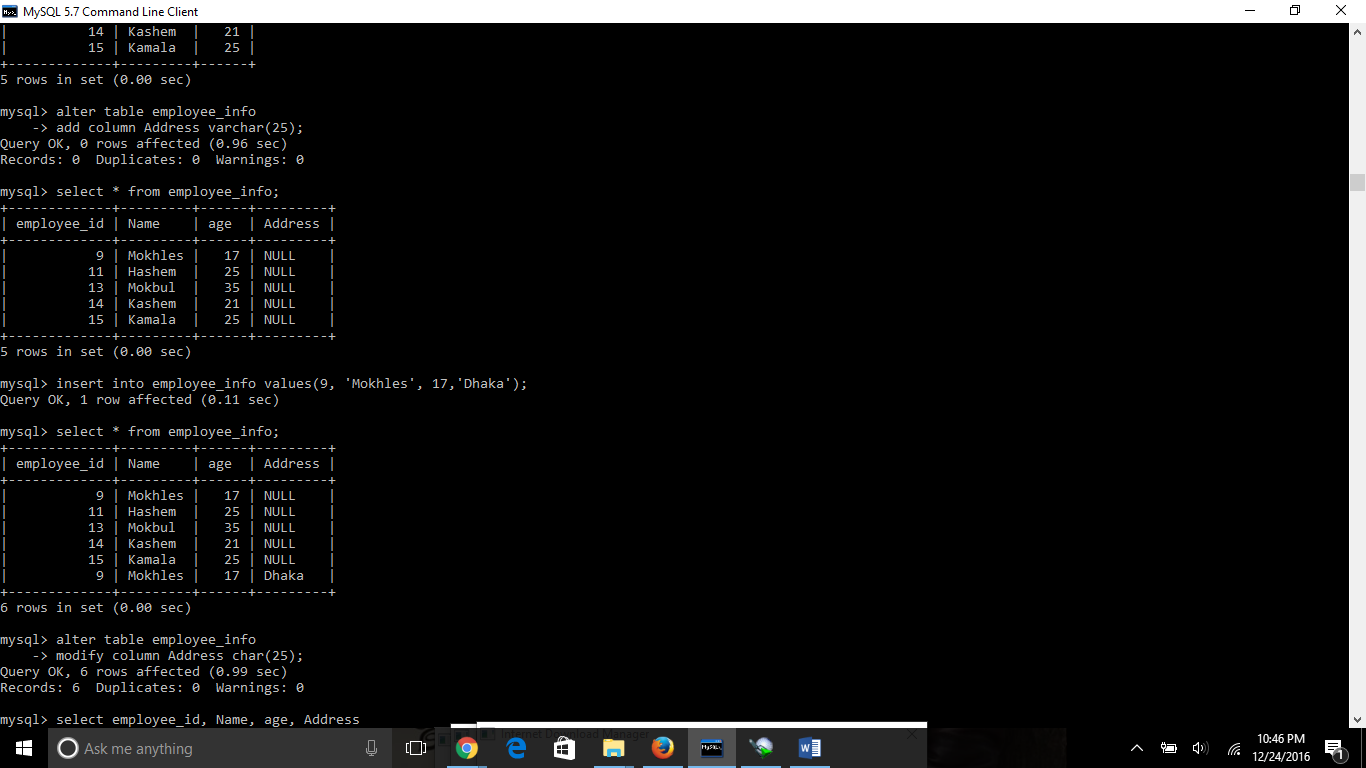
Adding column

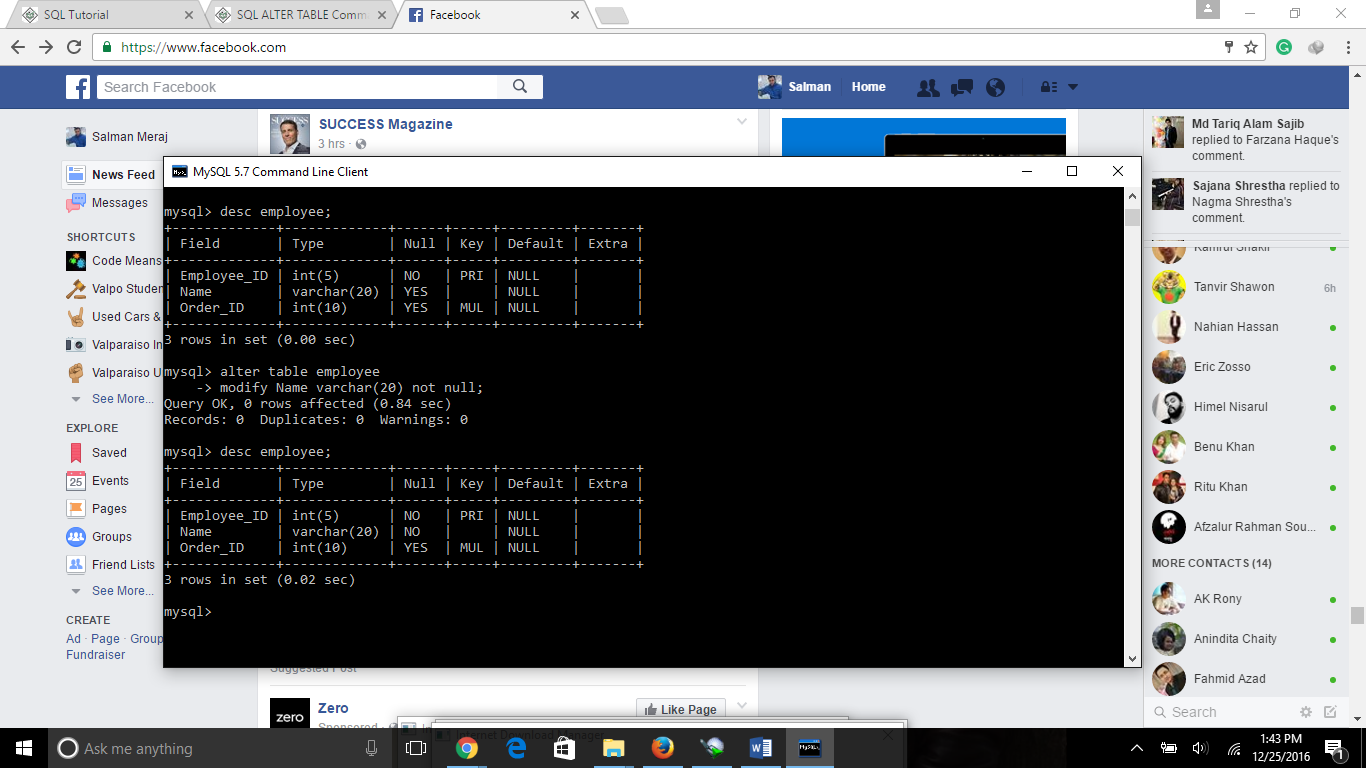


Dropping column

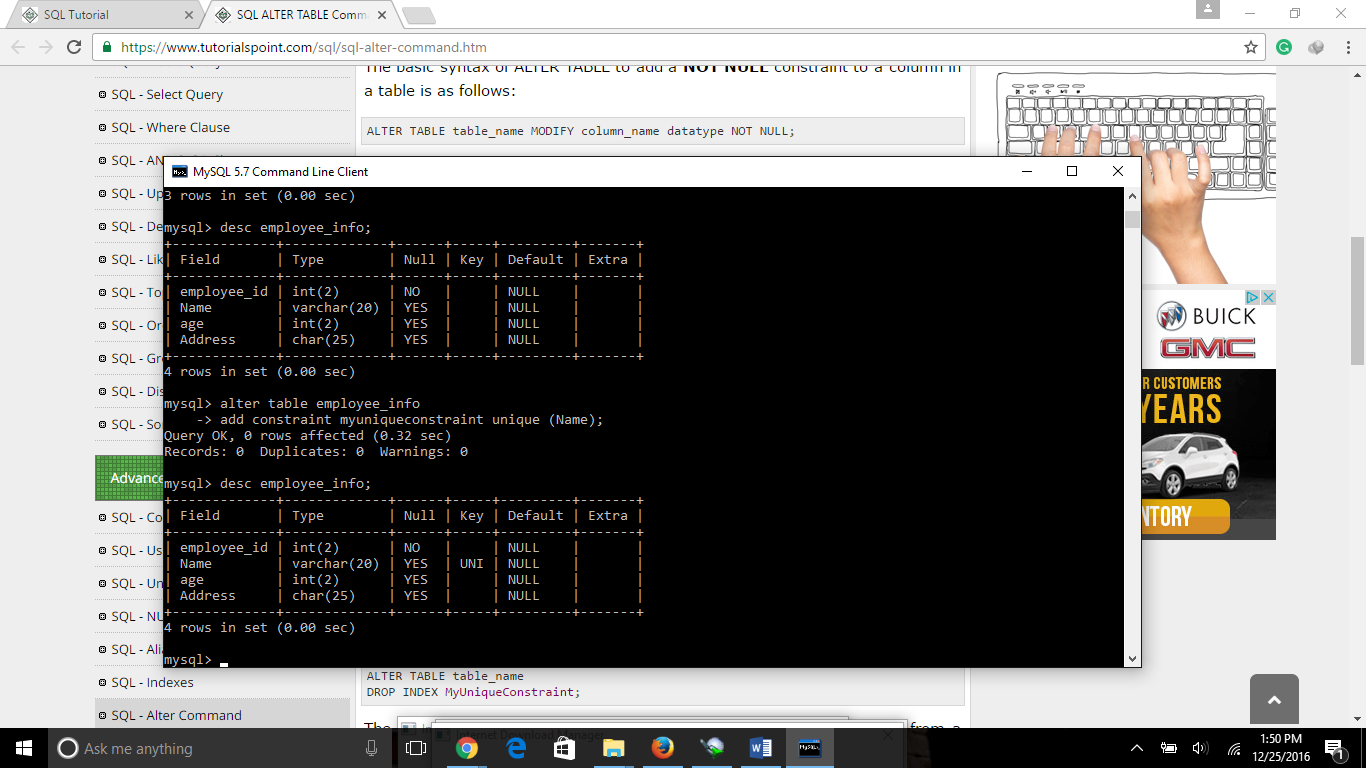


Modify datatypes of a column





Add unique constraint

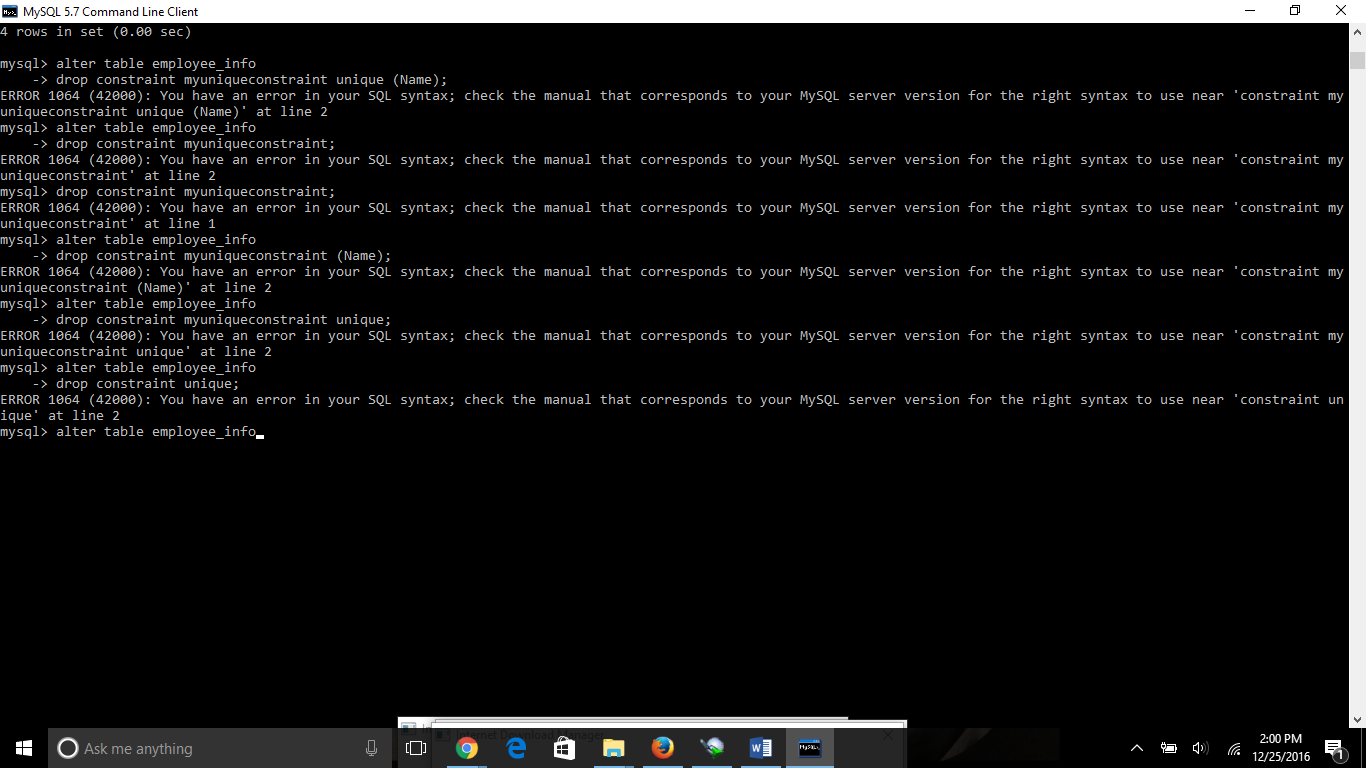


Drop constraint

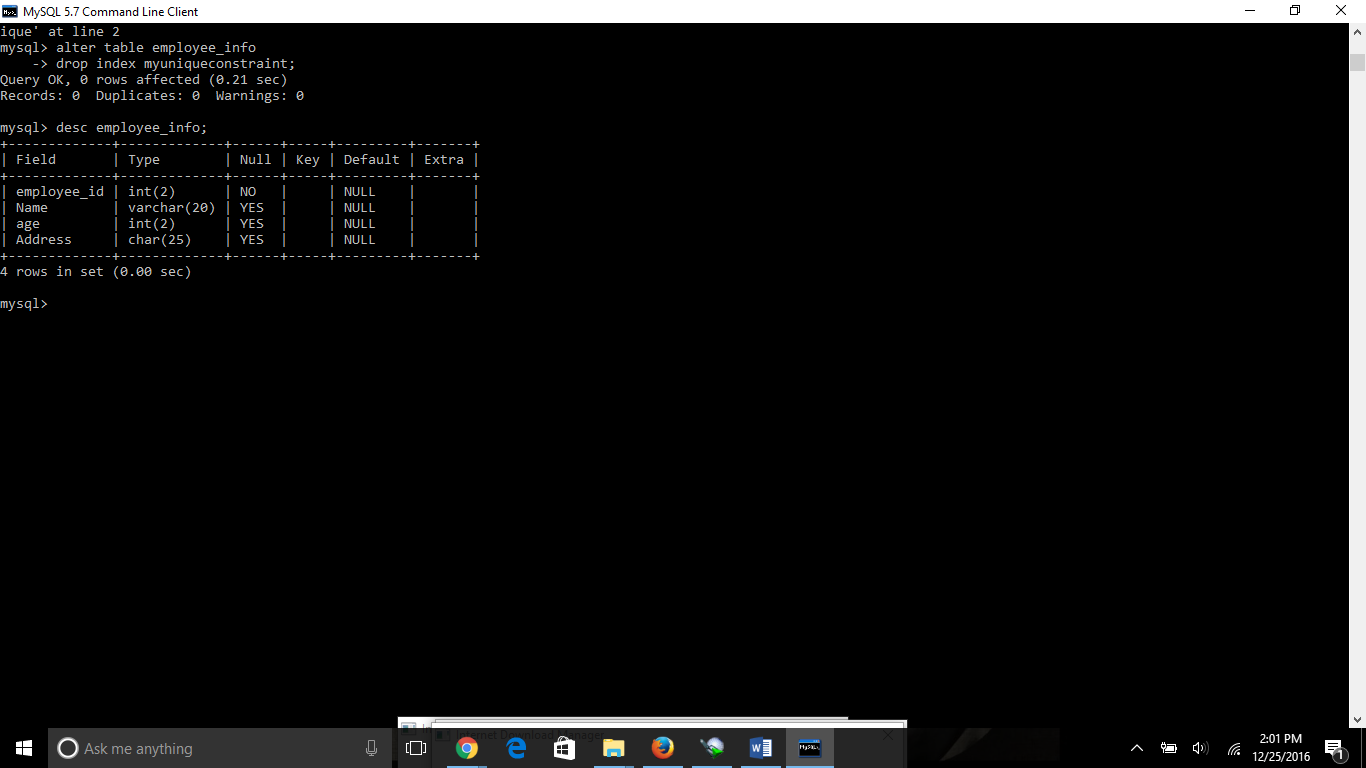
There is a slight differences in coding between SQL and MY SQL in case of drop constraint. In sql the code is as following :

ALTER TABLE table\_name

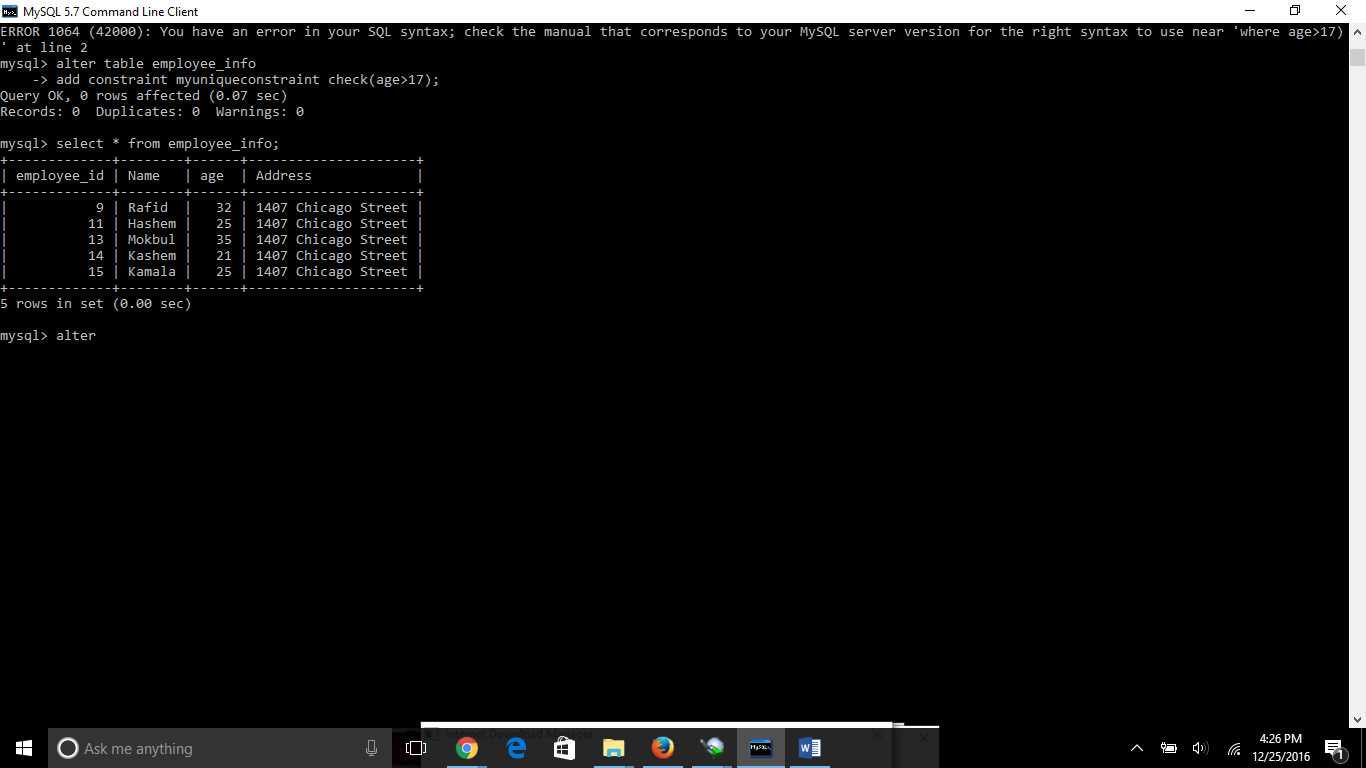
DROP CONSTRAINT MyUniqueConstraint;



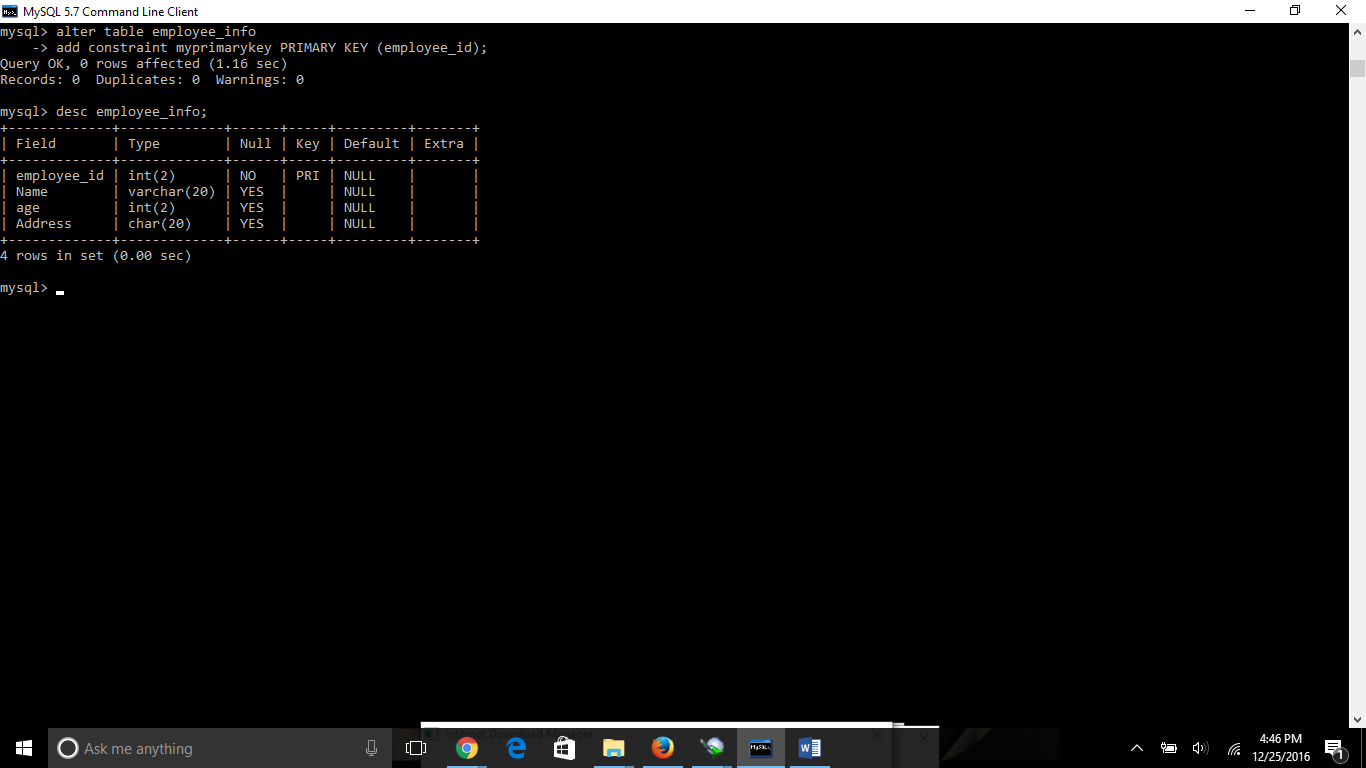
In case of My SQL it is:



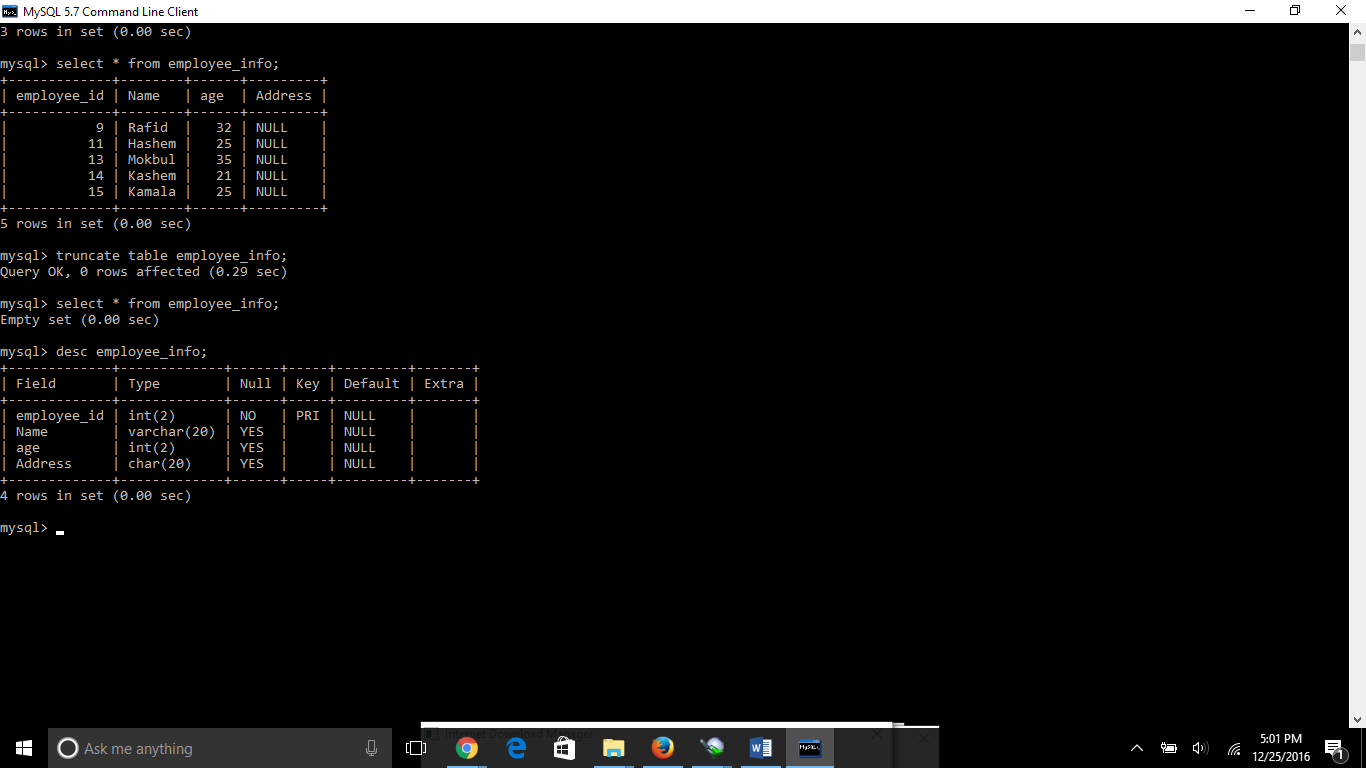
Check constraint



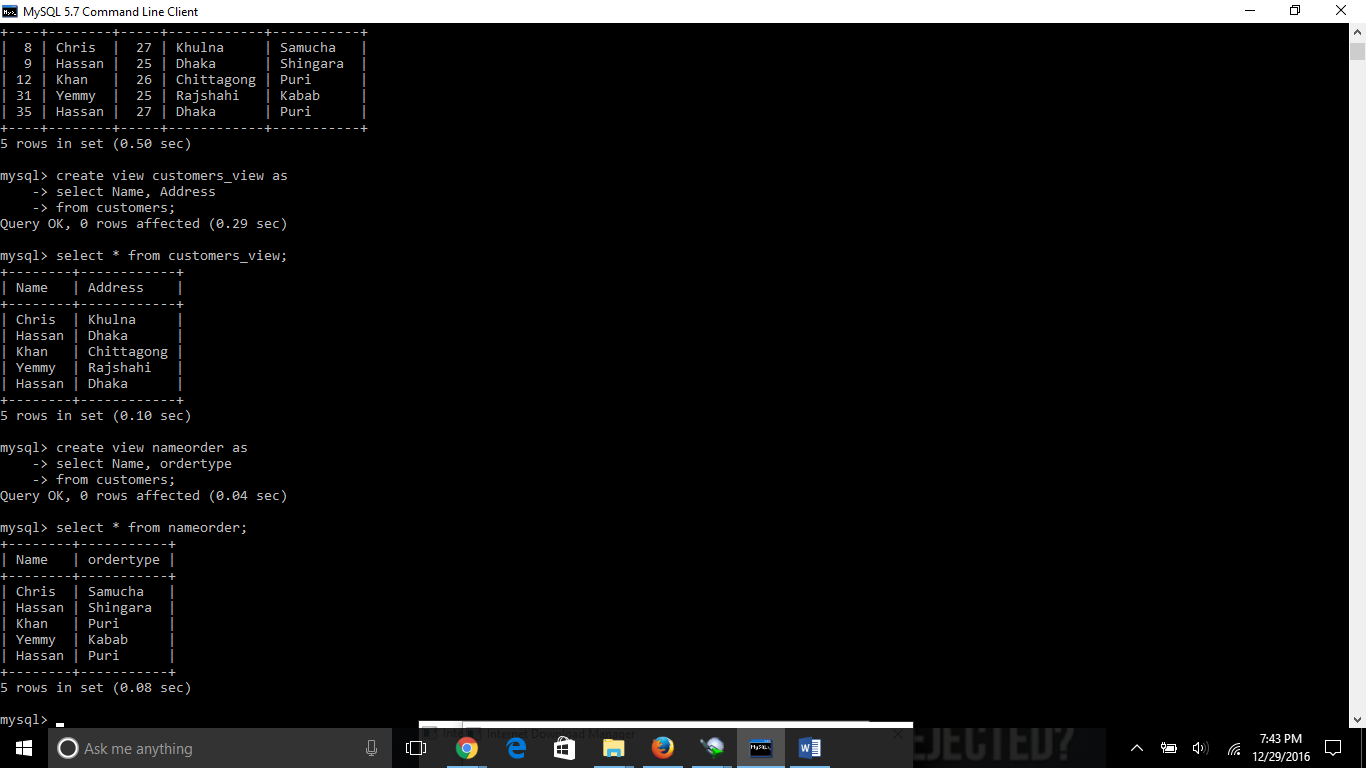
Add primary key



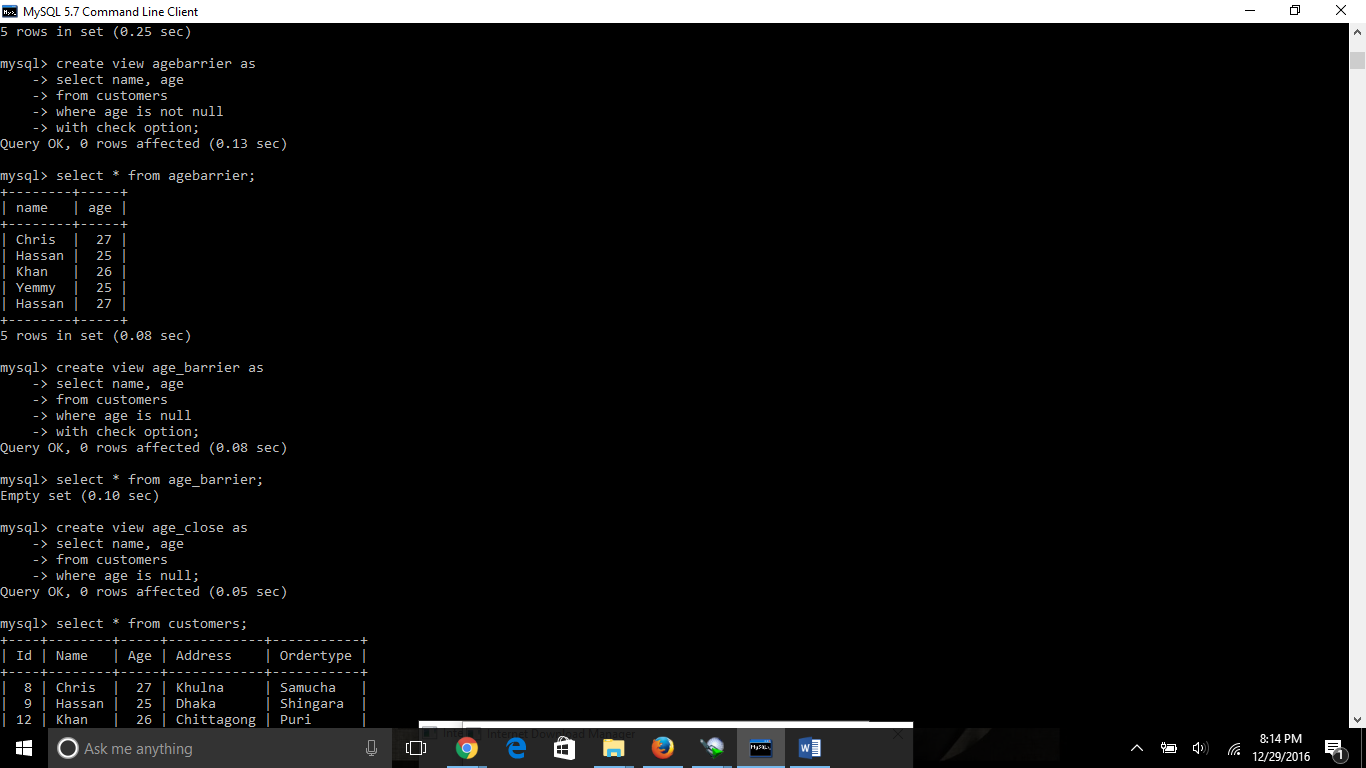
Truncate table: if you want to delete the data of a table by keeping the table then we can use truncate table syntax. The difference between truncate table and drop table is that drop table will delete the whole table from the database. So in future we need to create the whole table by create table syntax. But in truncate command it will only delete the data inside the table by keeping the table.

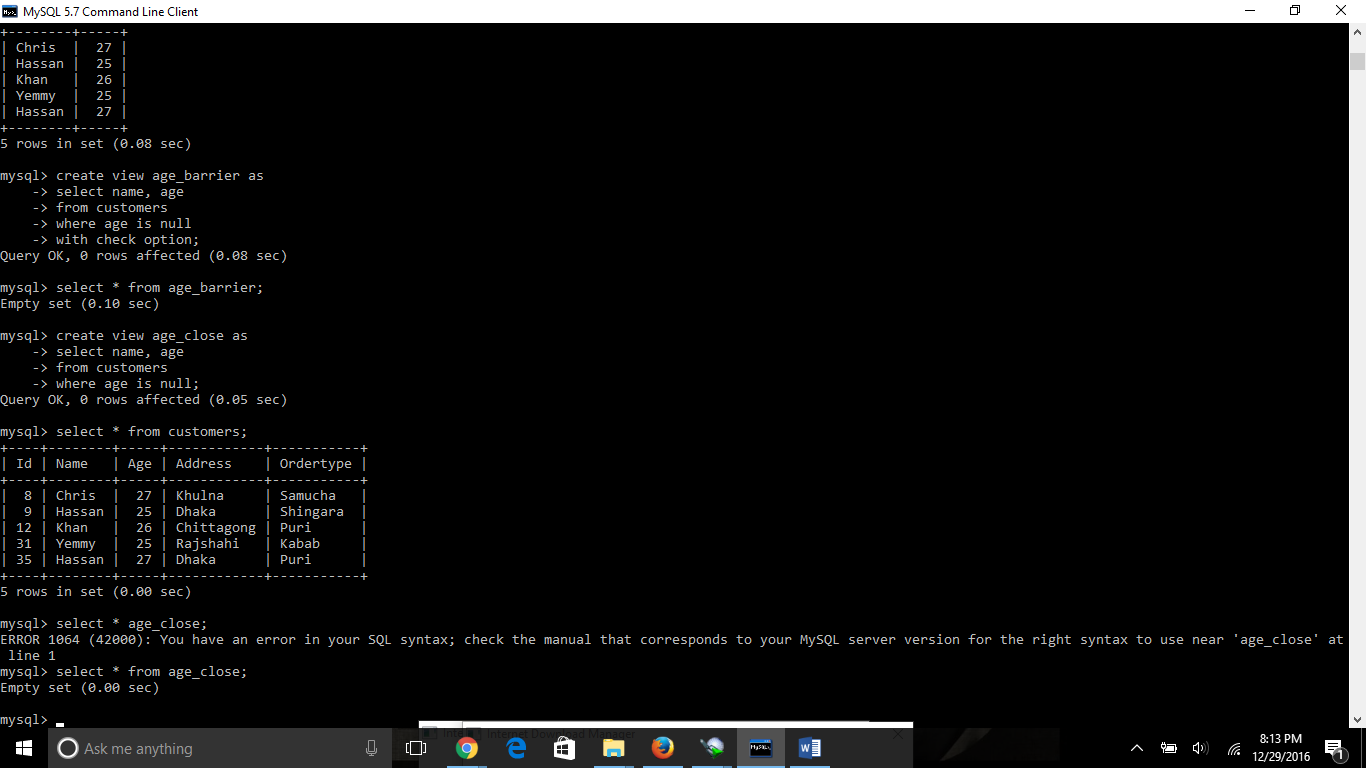


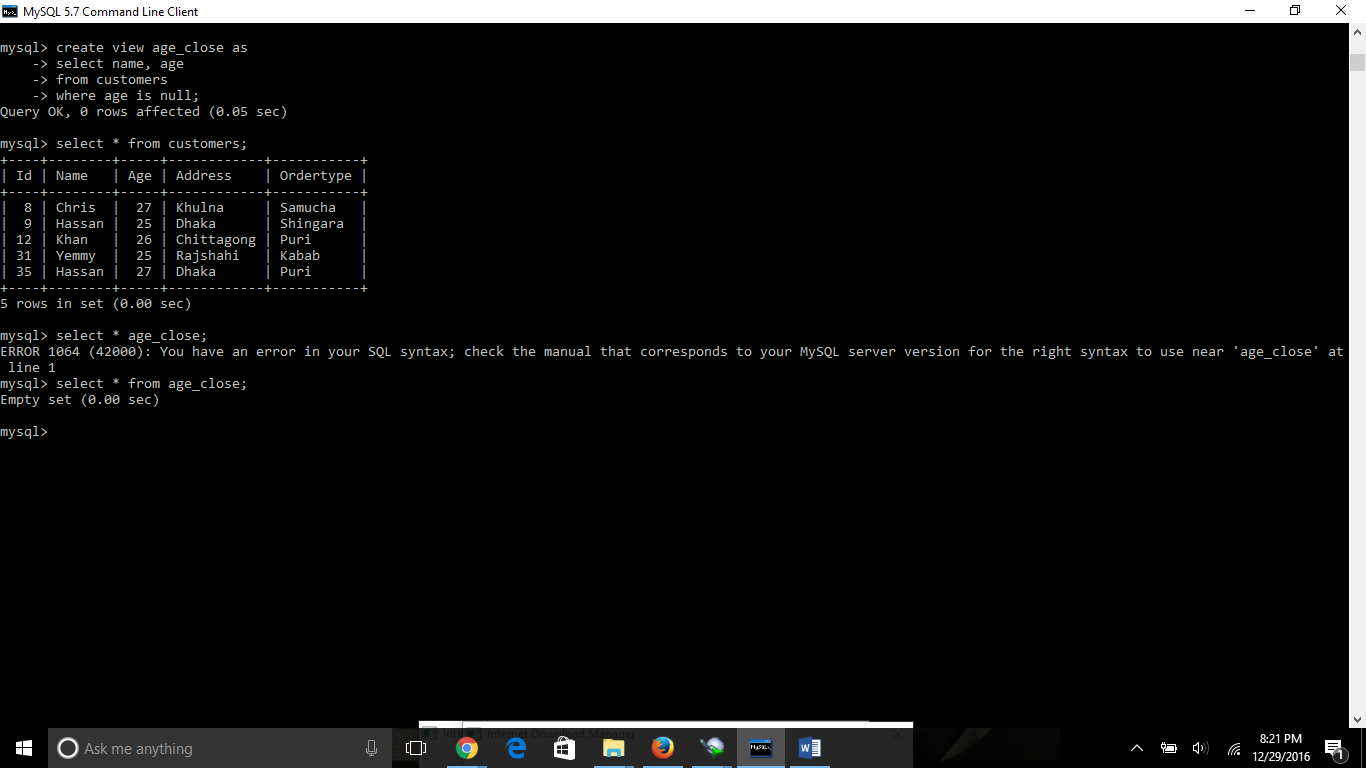
View: the main purpose of view is when we need to work on or multiple columns of a table instead of the whole table of the database. It works like we are creating a new temporary table with our selected columns.



Create view with where clause and check option. The purpose of the WITH CHECK OPTION is to ensure that all UPDATE and INSERTs satisfy the condition(s) in the view definition.

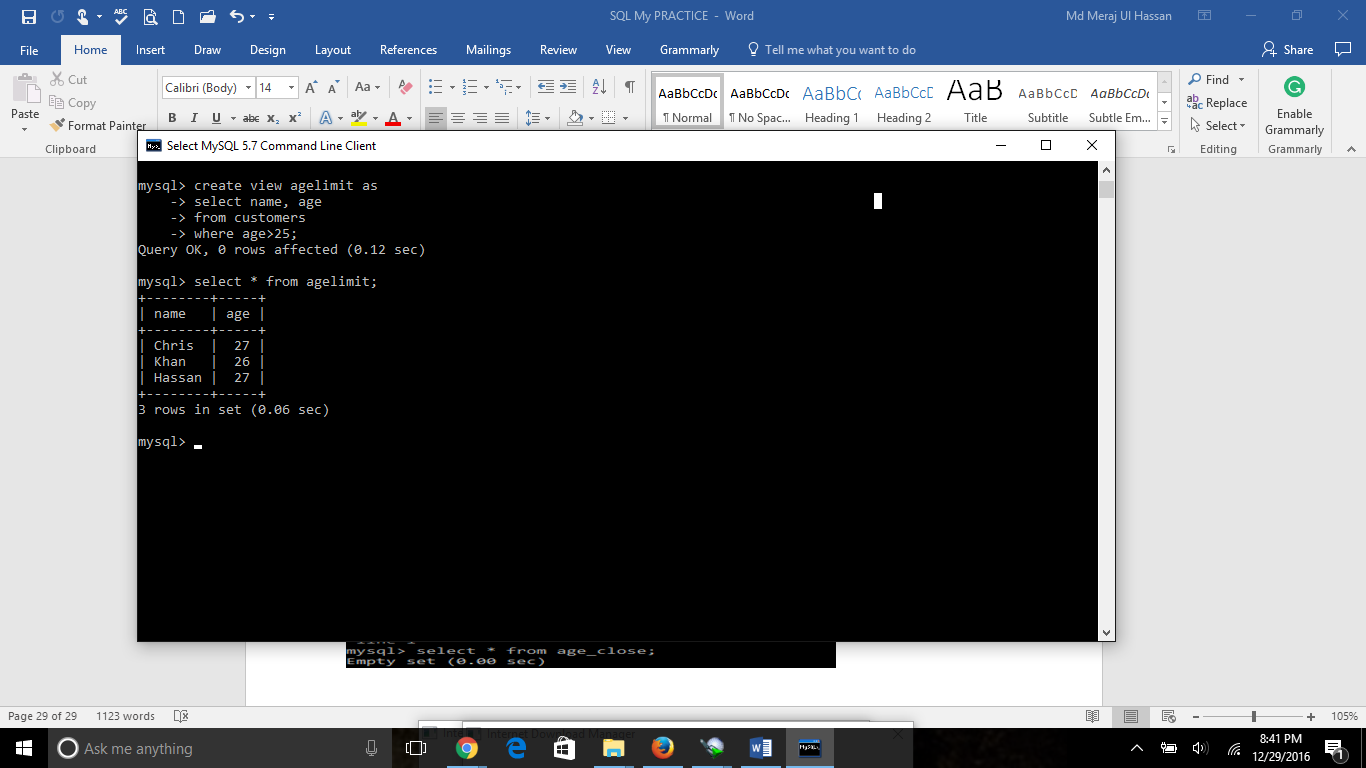


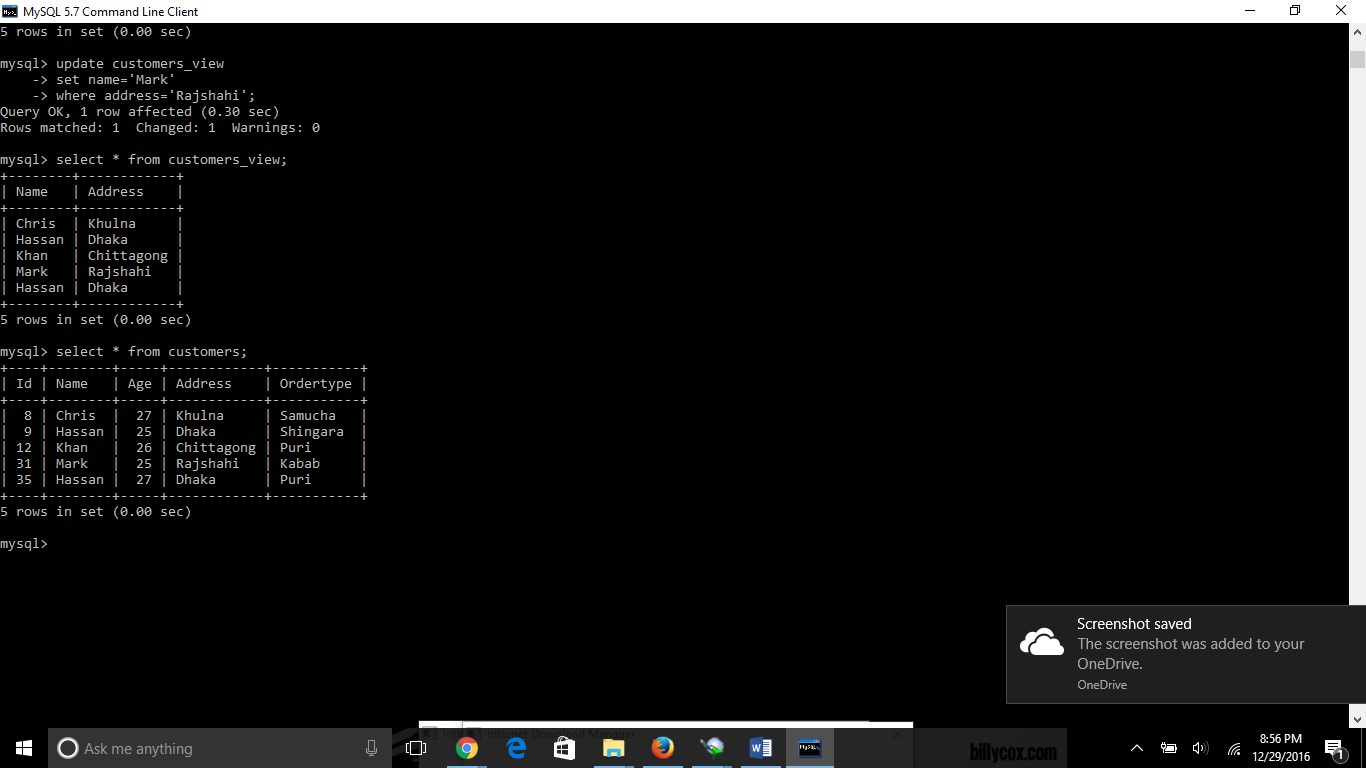




Update view: when we update a view the data of the parent column will also change with the update of the view table.

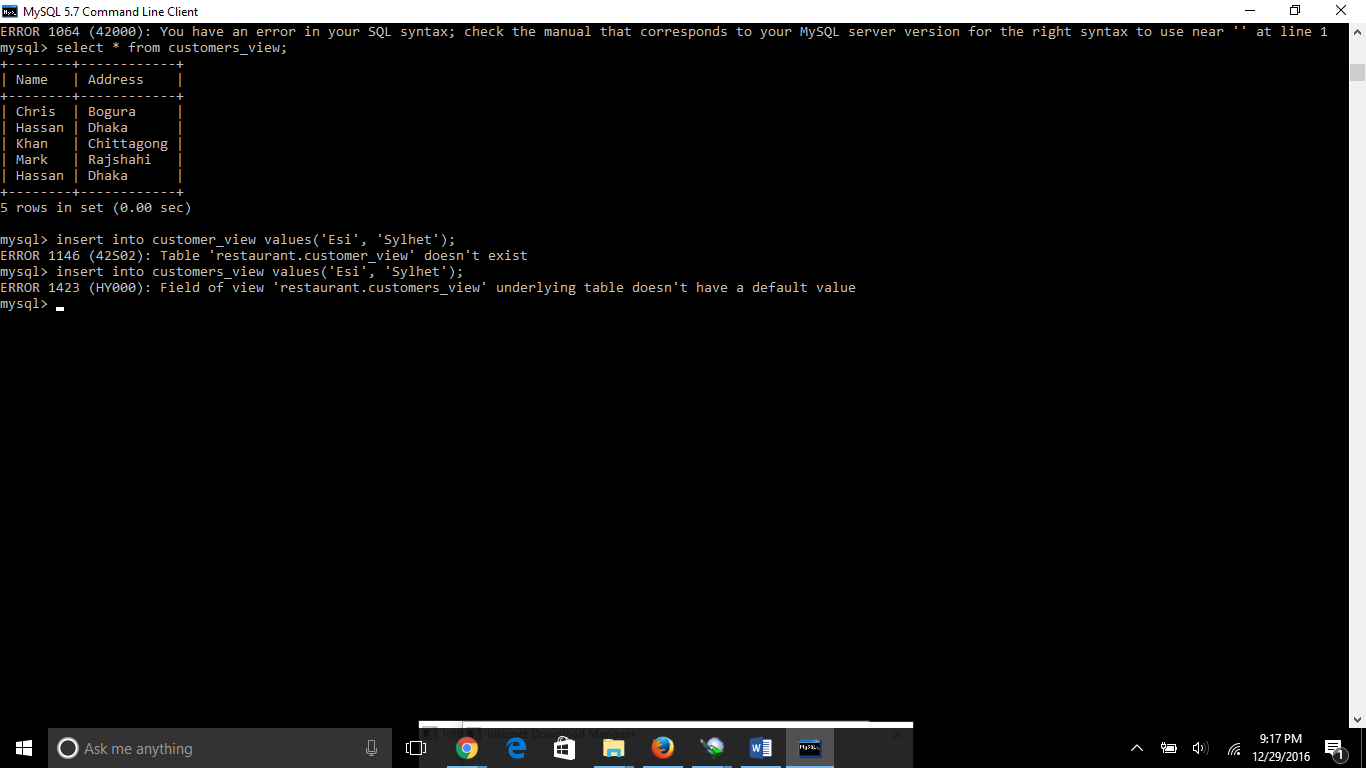
Create view with where clause condition.



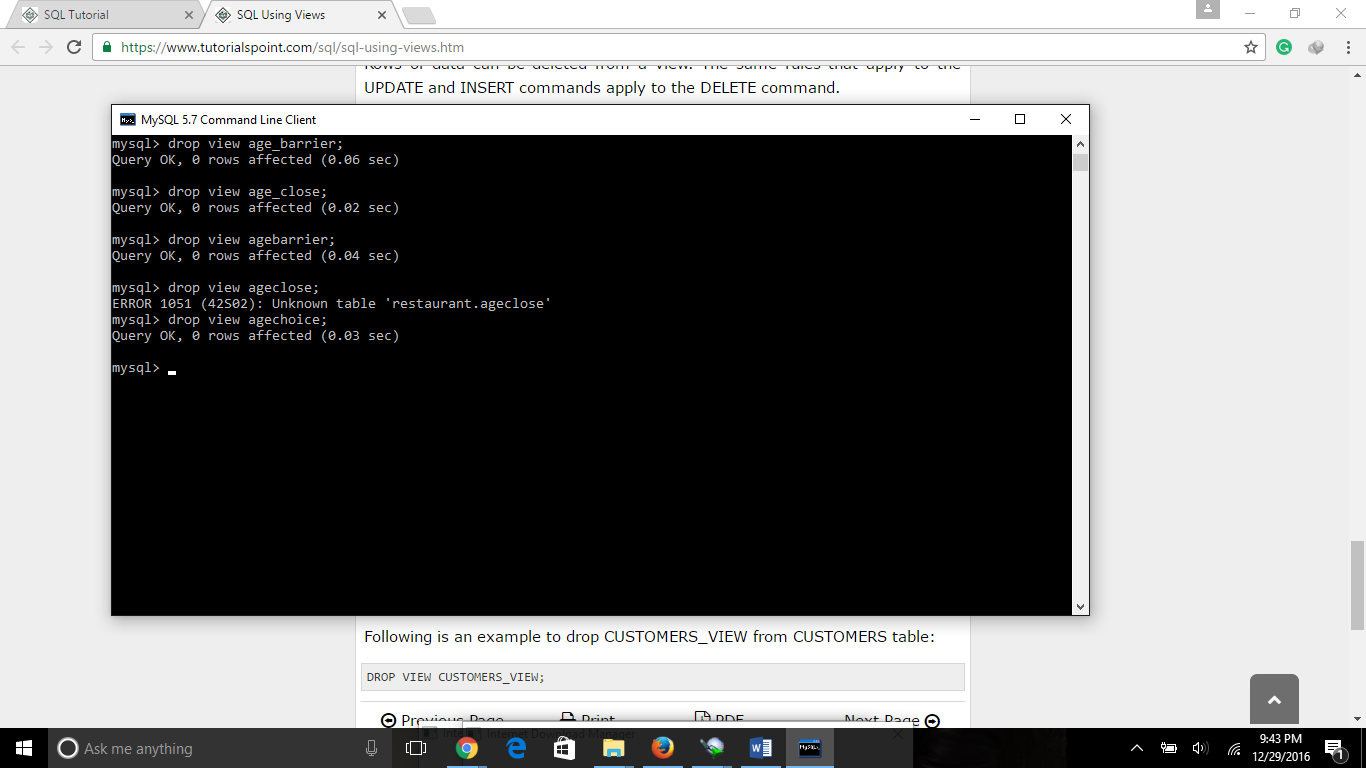


**Inserting Rows into a View:**

Rows of data can be inserted into a view. The same rules that apply to the UPDATE command also apply to the INSERT command. Here we can not insert rows in CUSTOMERS\_VIEW because we have not included all the NOT NULL columns in this view, otherwise you can insert rows in a view in similar way as you insert them in a table.



Drop view



Having Clause: the having clause is used as like as where clause but it is used only when there is any group by statement is used

Syntax: SELECT column1, column2

FROM table1, table2

WHERE [ conditions ]

GROUP BY column1, column2

HAVING [ conditions ]

ORDER BY column1, column2

If some one asks you that find out the id, name and address of the guy who’s age is more than 25 years and name is Hassan then you can use the having clause like following.

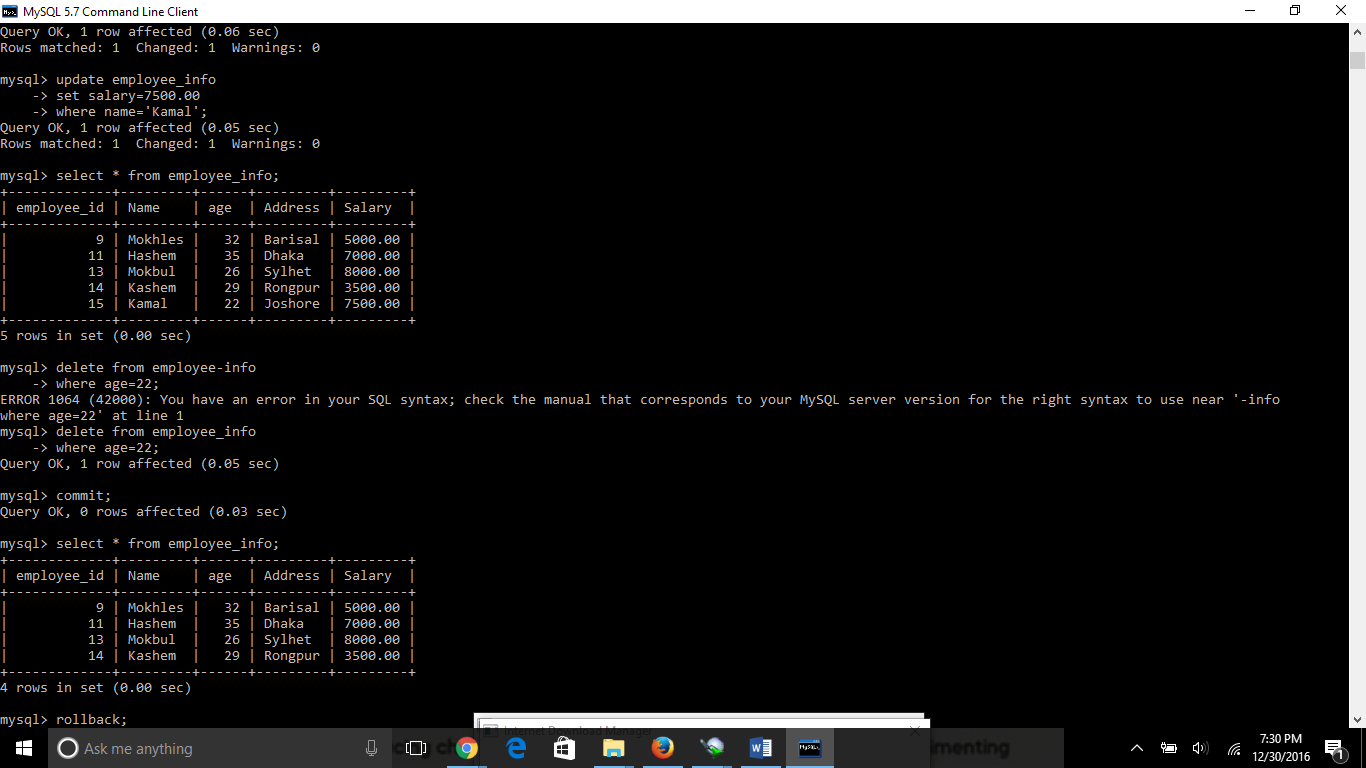


**Transaction:**

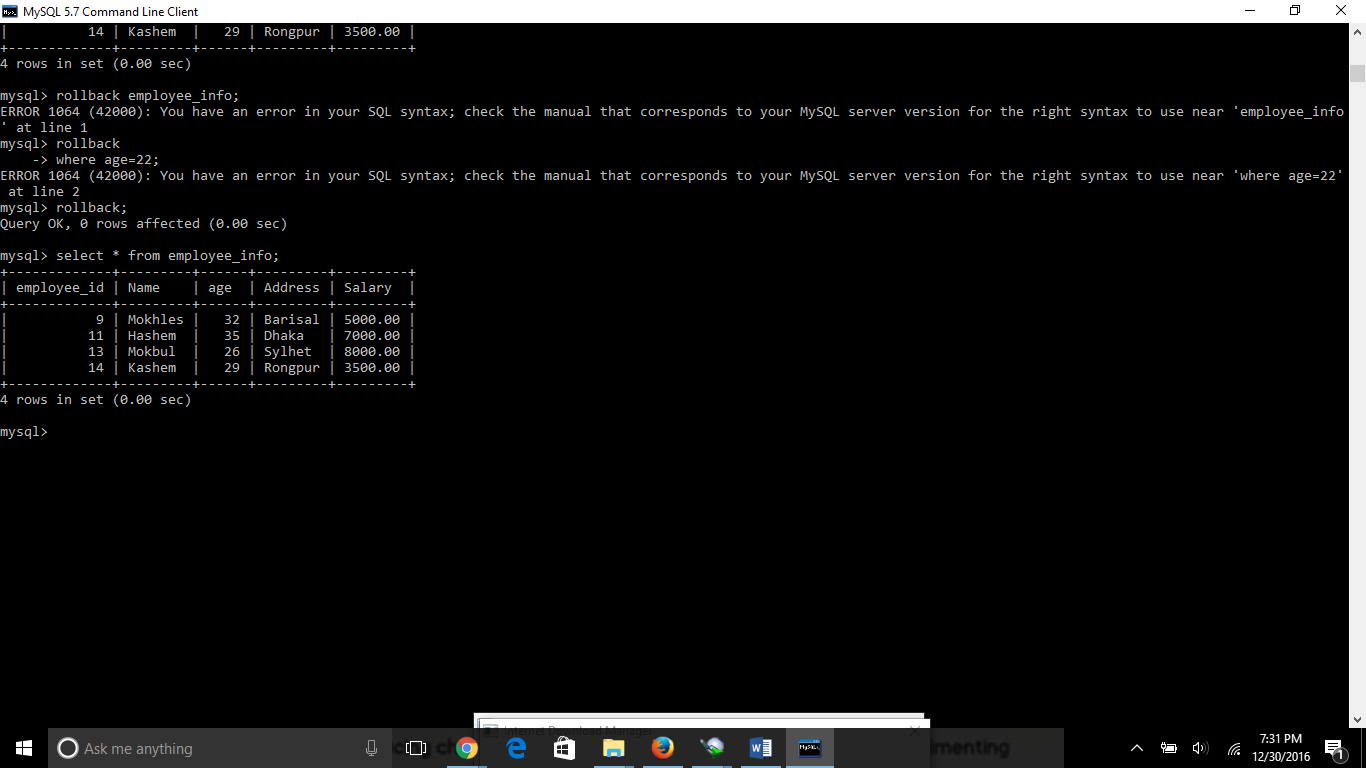
* **COMMIT:** to save the changes.
* **ROLLBACK:** to rollback the changes.
* **SAVEPOINT:** creates points within groups of transactions in which to ROLLBACK
* **SET TRANSACTION:** Places a name on a transaction.

Transactional control commands are only used with the DML commands INSERT, UPDATE and DELETE only. They can not be used while creating tables or dropping them because these operations are automatically commited in the database.

**Commit**



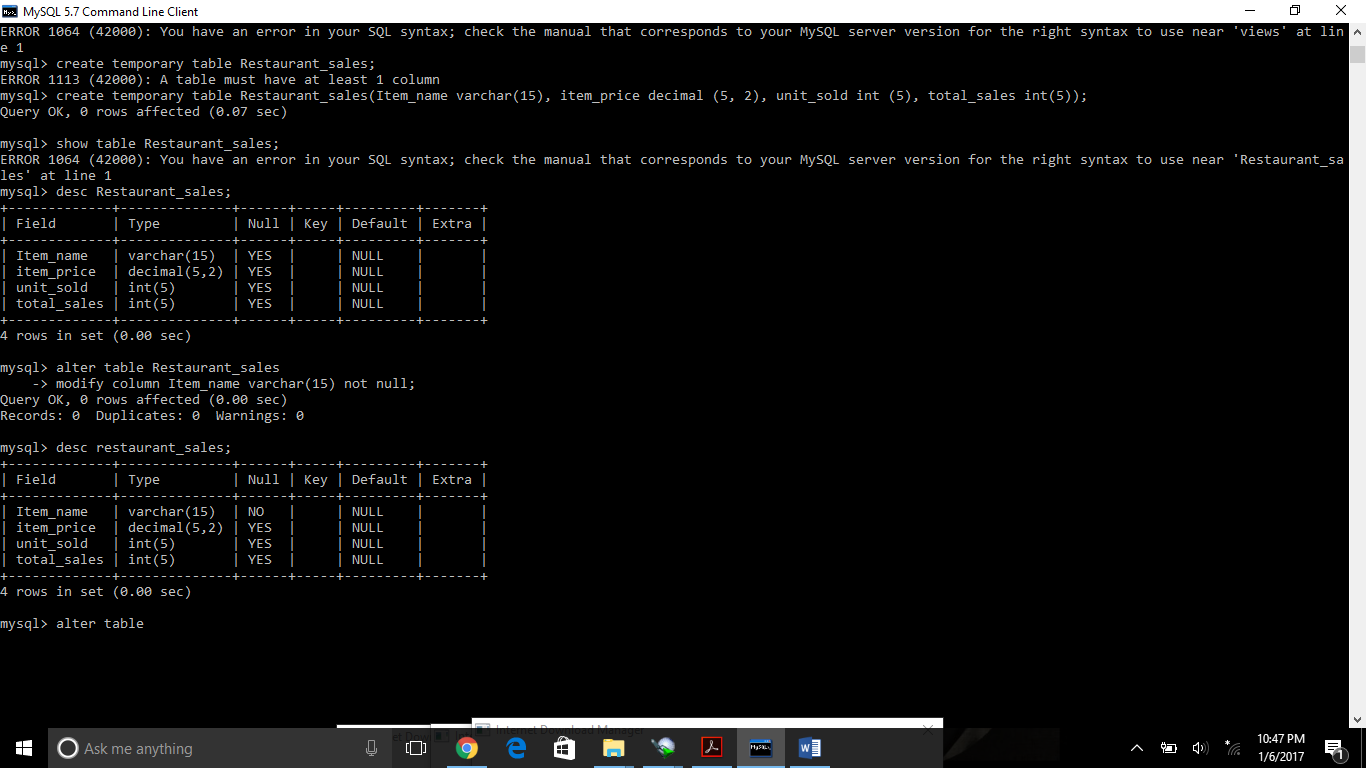
Rollback:



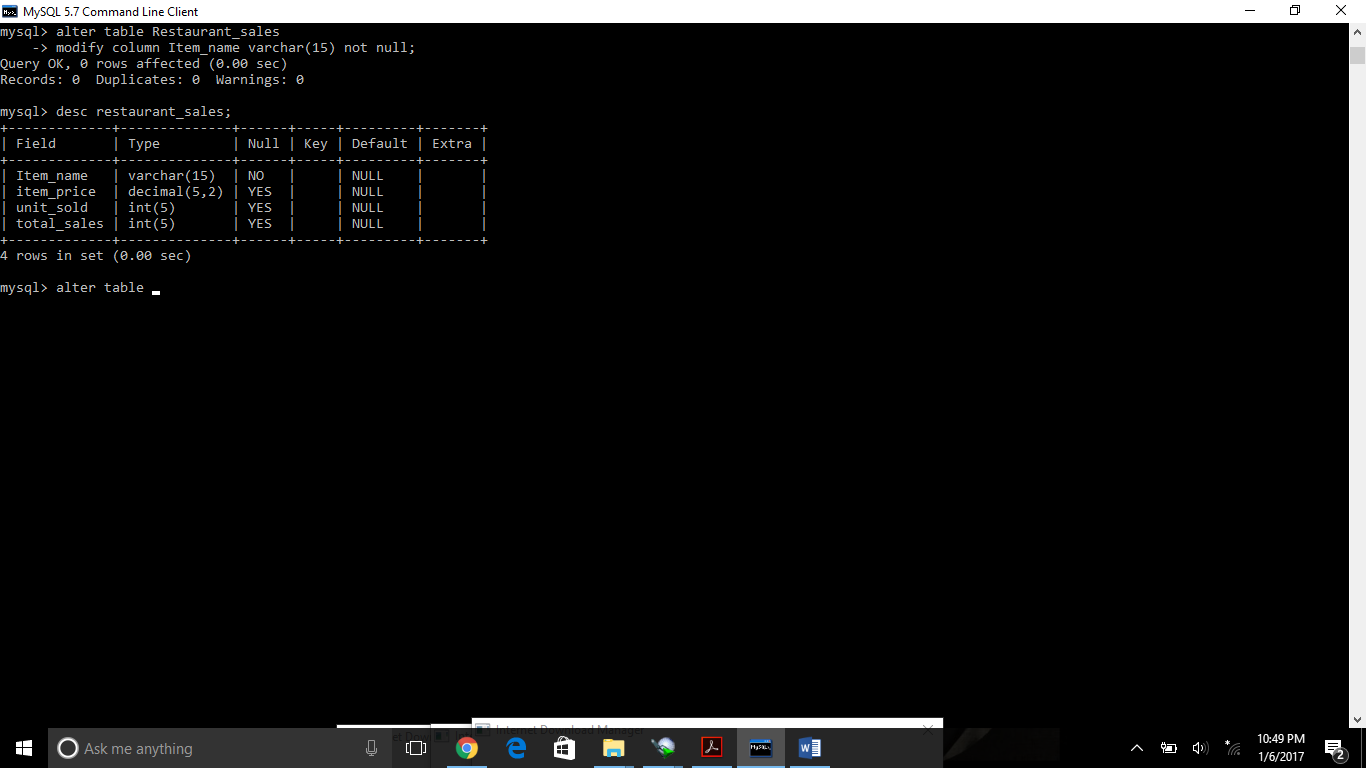
**In the upper screenshot the rollback command did not work well. It might be the reason of the MY SQL command mismatch.**

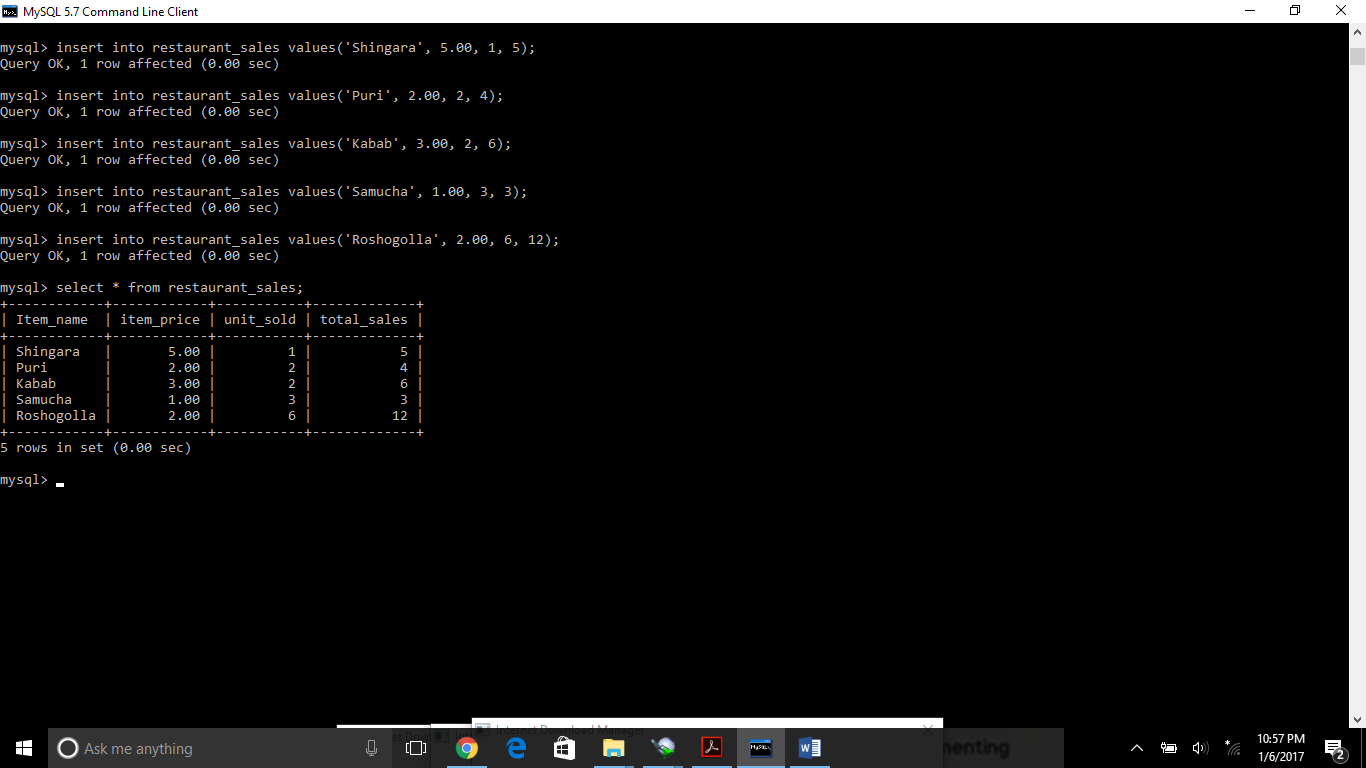
A SAVEPOINT is a point in a transaction when you can roll the transaction back to a certain point without rolling back the entire transaction.

**Temporary table:**



**Changing the not null into the column**

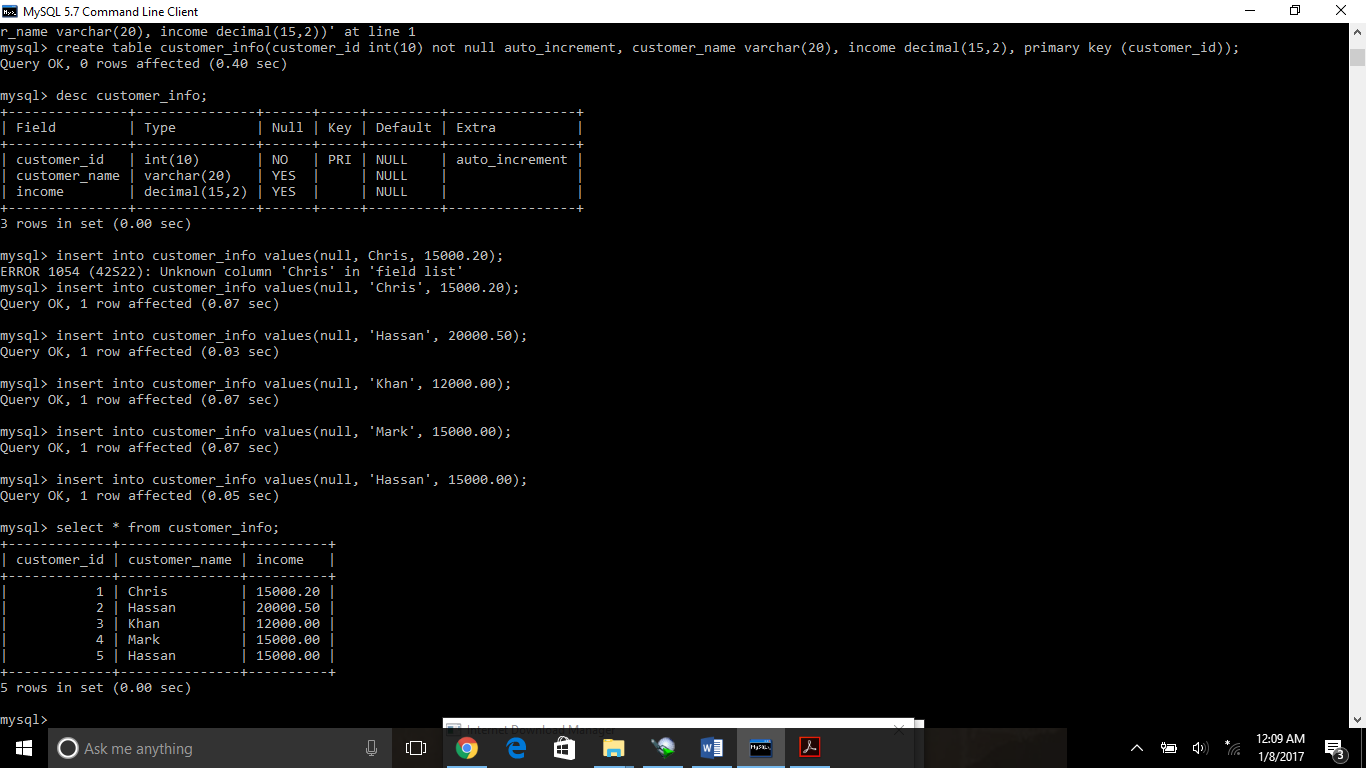


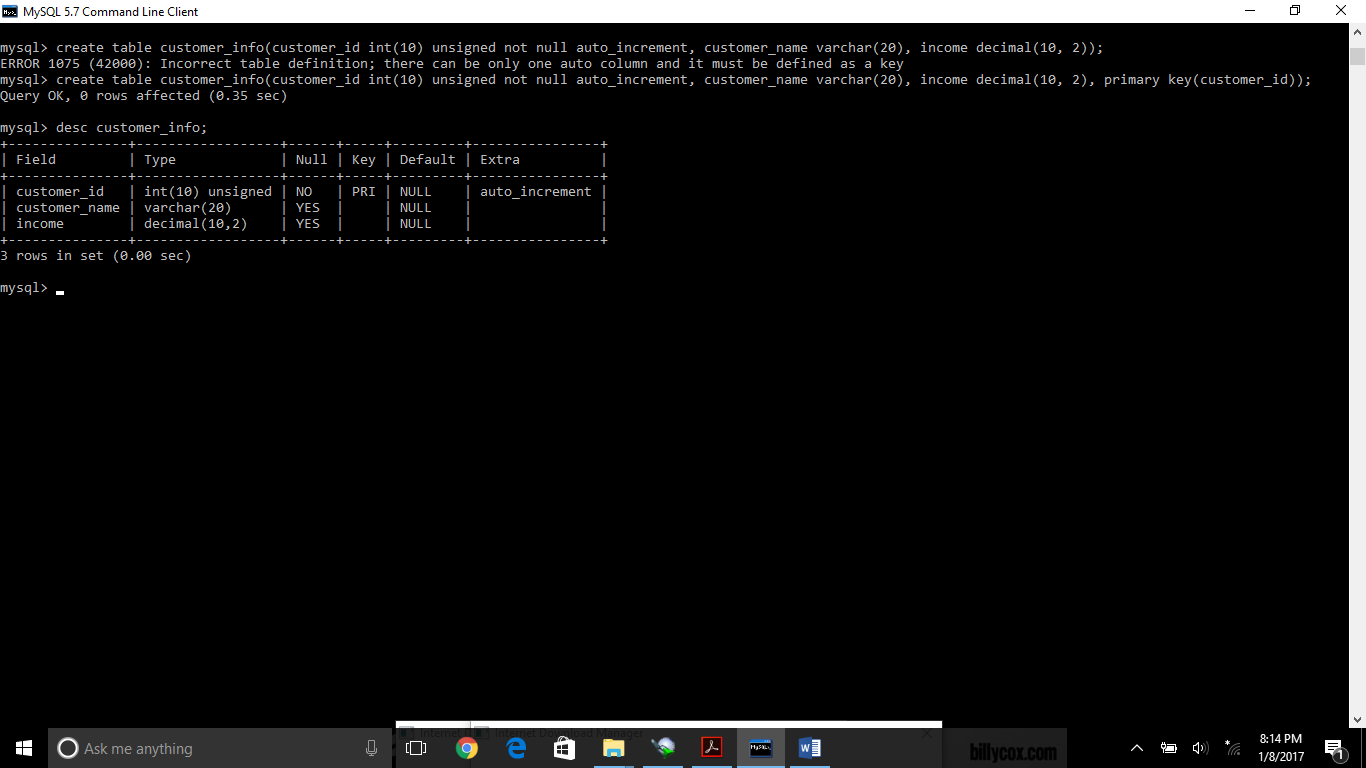


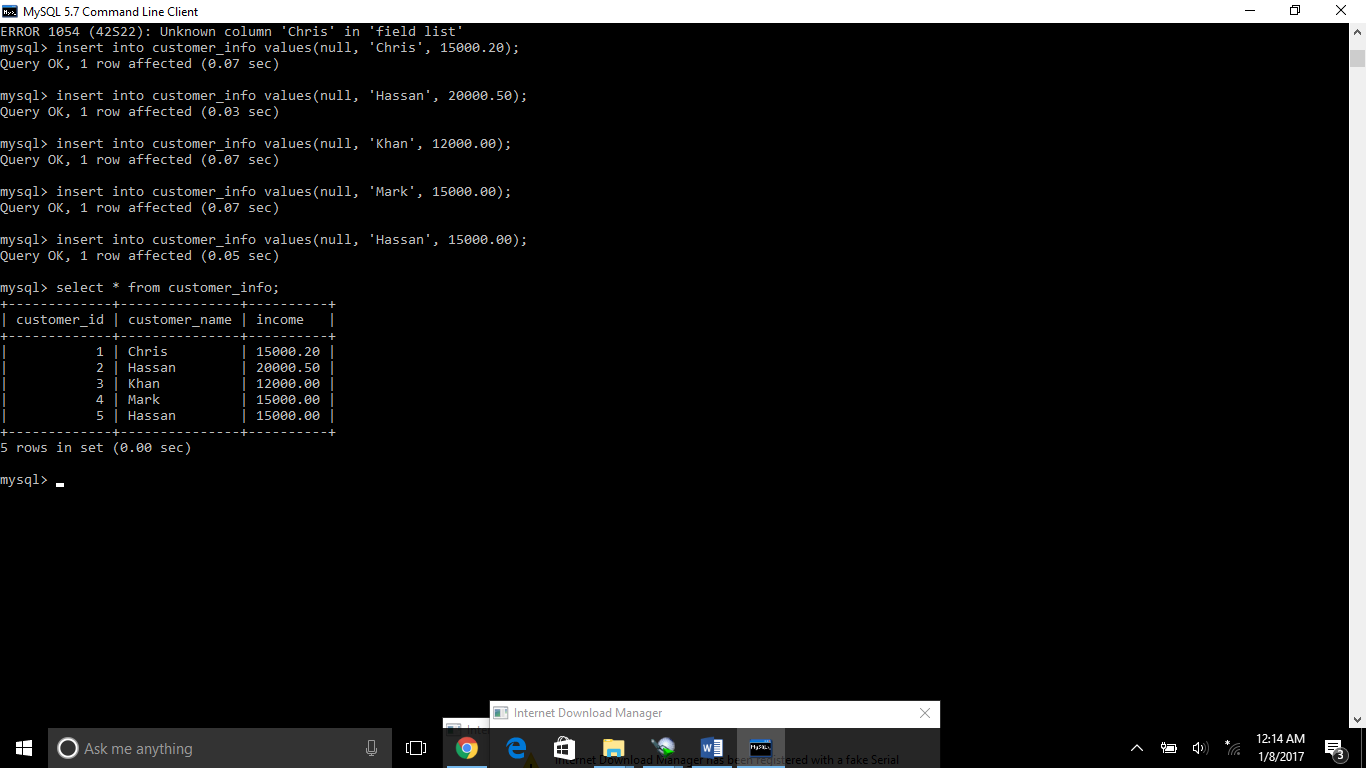
Create sequence in a table:

Frequently databases use the sequential order to call any row of the table. To do that we need to make sequence. In sequence if we do not put any order or serial number it will automatically show the result with a serial.

Syntax:







**Renumbering or resequencing a table**: there might be situation when we need to renumbering our tables. In that case at first, we need to drop that exact column from the table by using alter command and then add that column again. Before doing so we need to be very careful that our particular table should not have any join.

The syntax is as followed

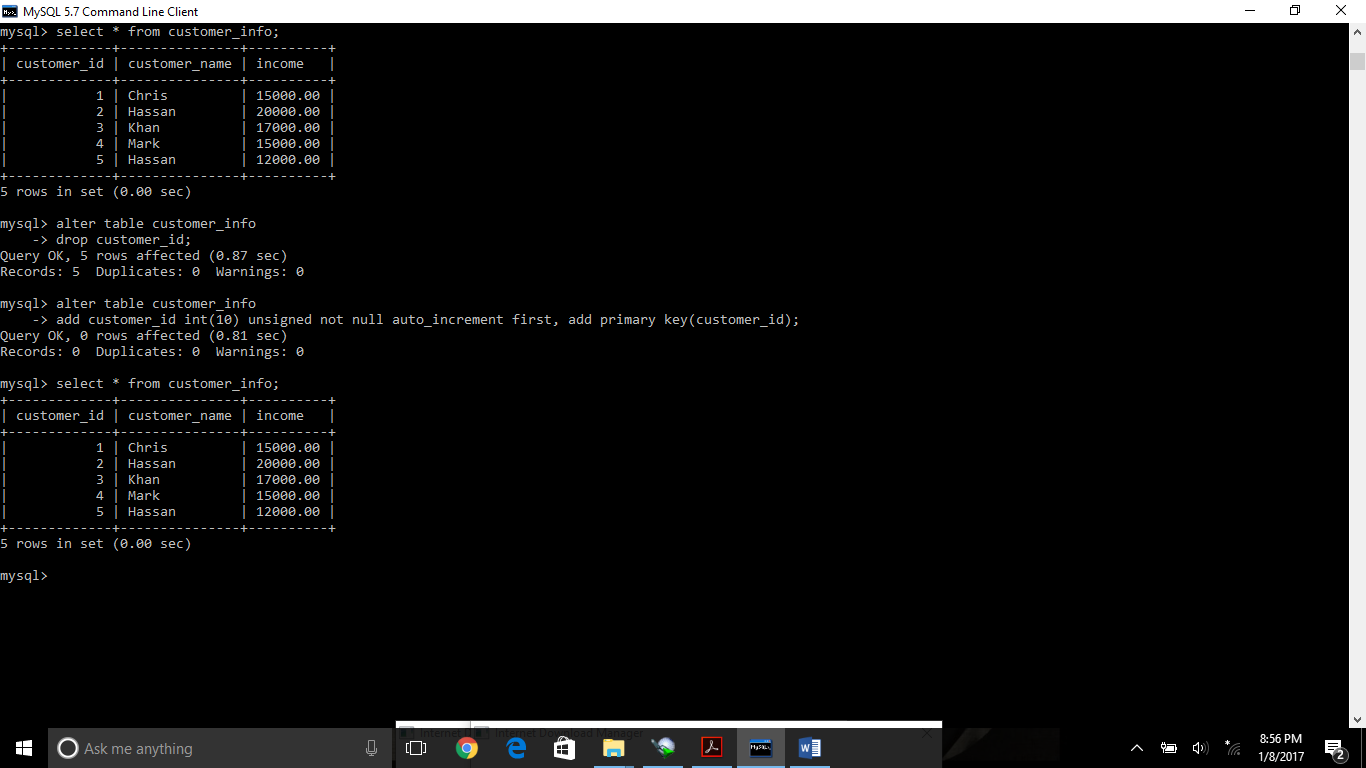
>Alter table table\_name

>drop column\_name;

Then again add that column

>alter table table\_name

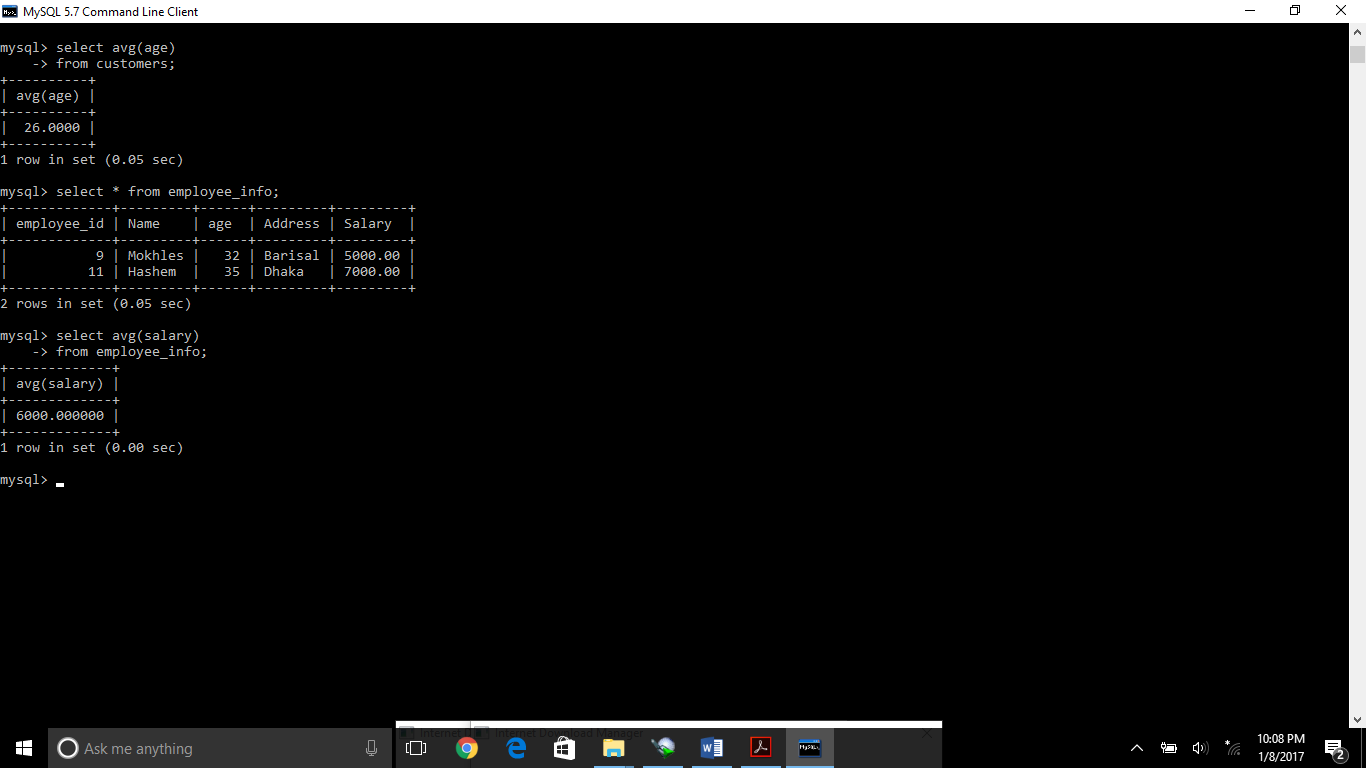
>add column\_name datatype unsigned not null auto\_increment first, add primary key(column\_name);



Average function: when you need to find out any average of any particular column then it would be like the following.

Basic syntax: >select avg(column\_name)

>from table\_name;

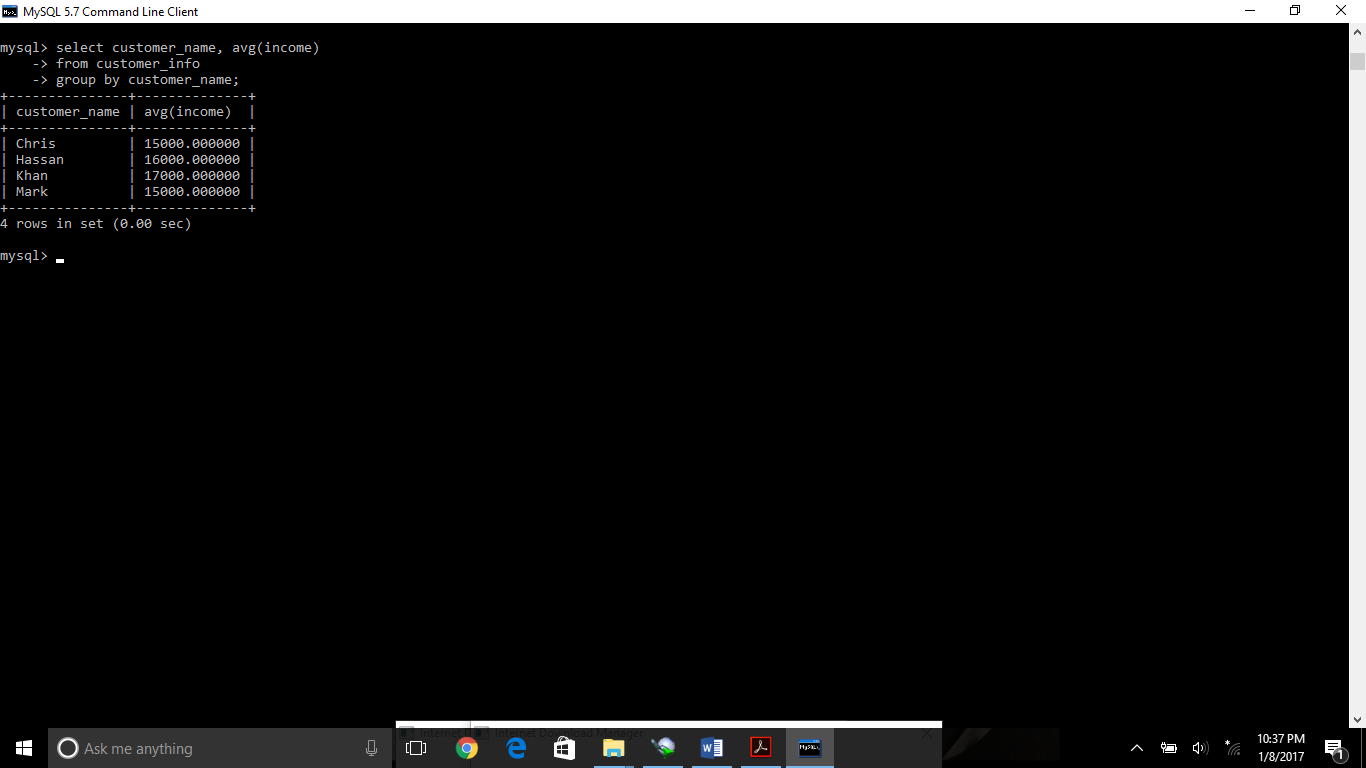


When you need to find out the averages of several records then the syntax would be as following.

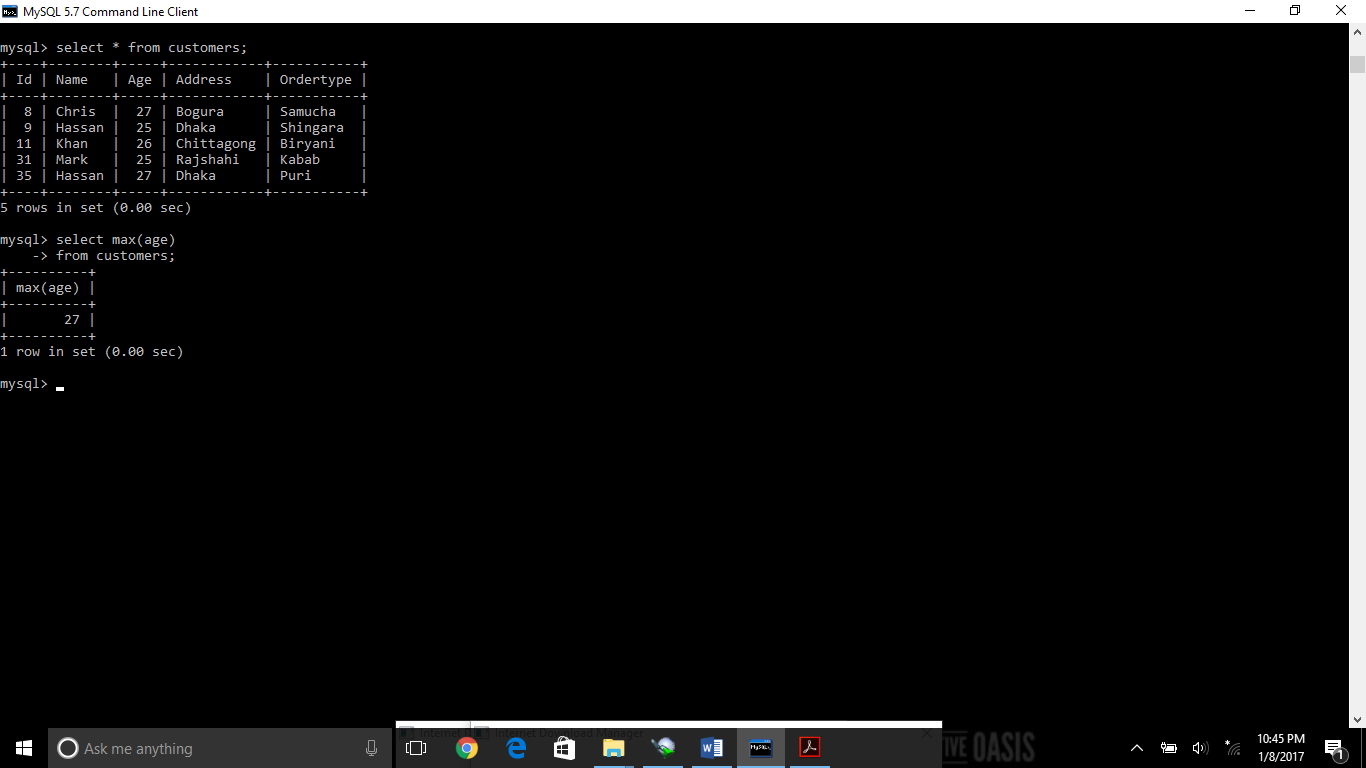
>select column\_name(\*\*this is the column which you need to do the grouping) avg(column\_name)

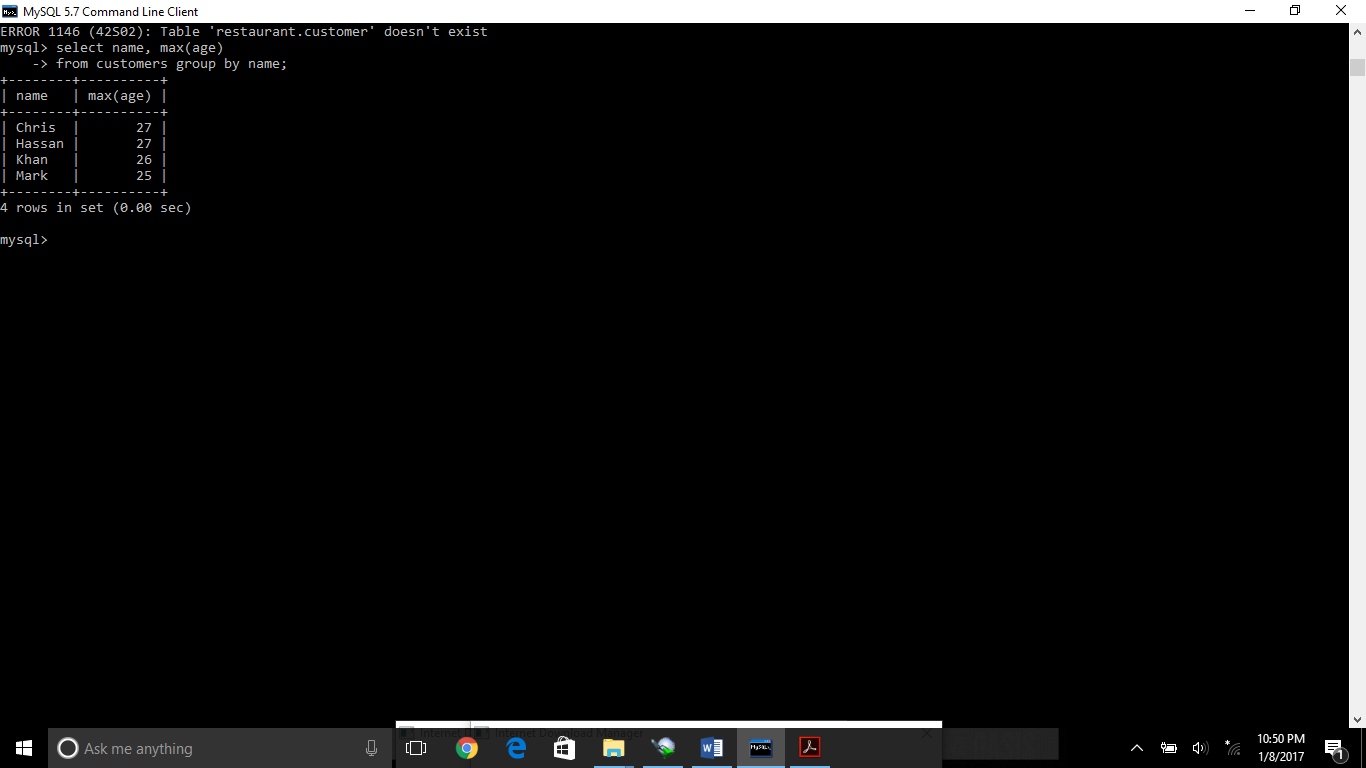
>from table\_name

>group by column\_name



Max function:

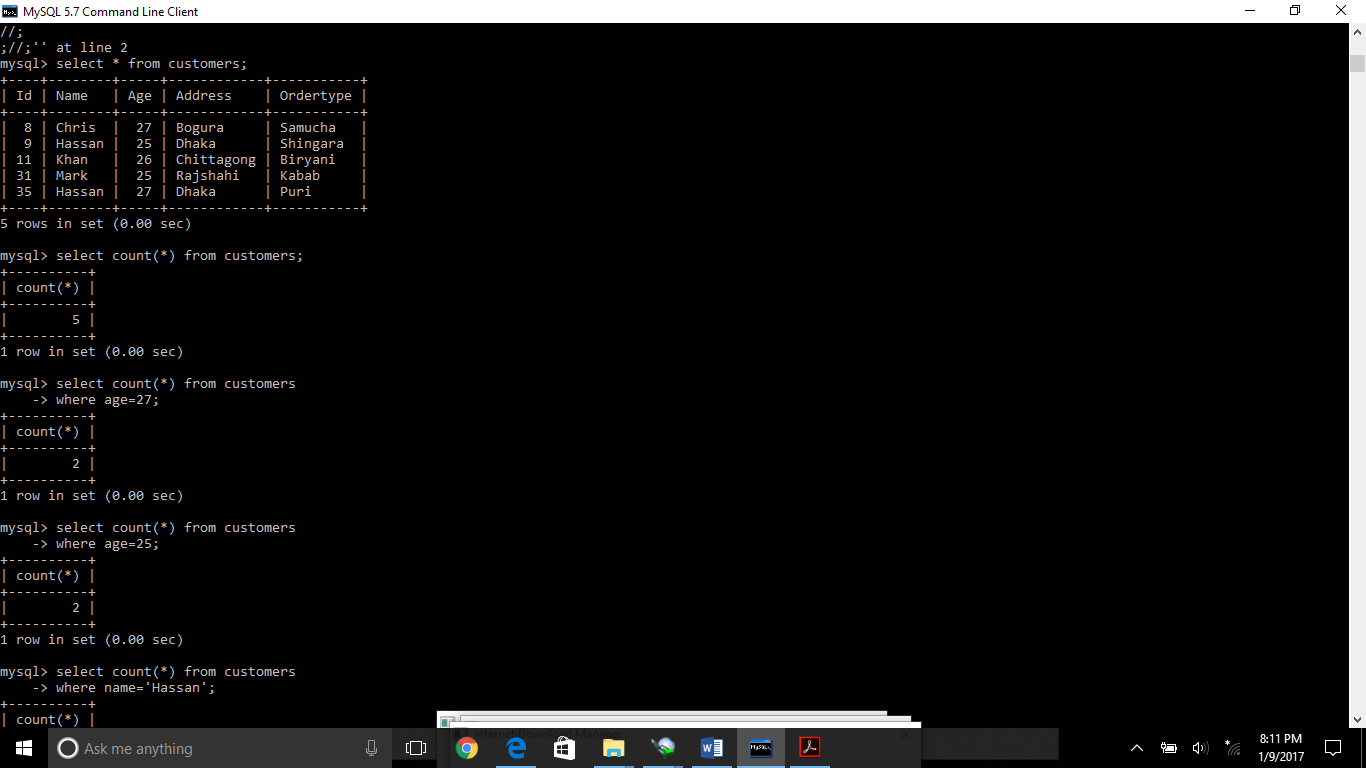




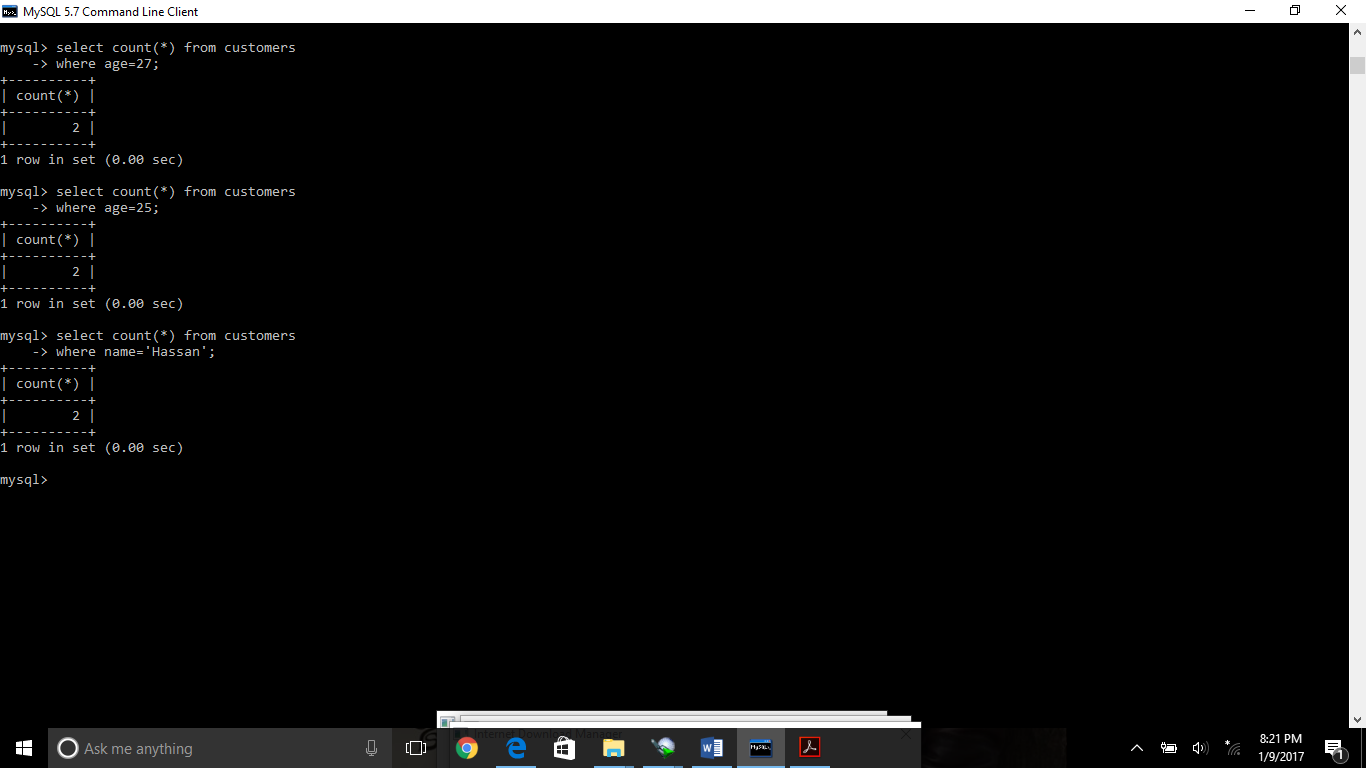
Count function: it will show us the number of all rows in a table or rows in a particular column.

No.of total rows in a table

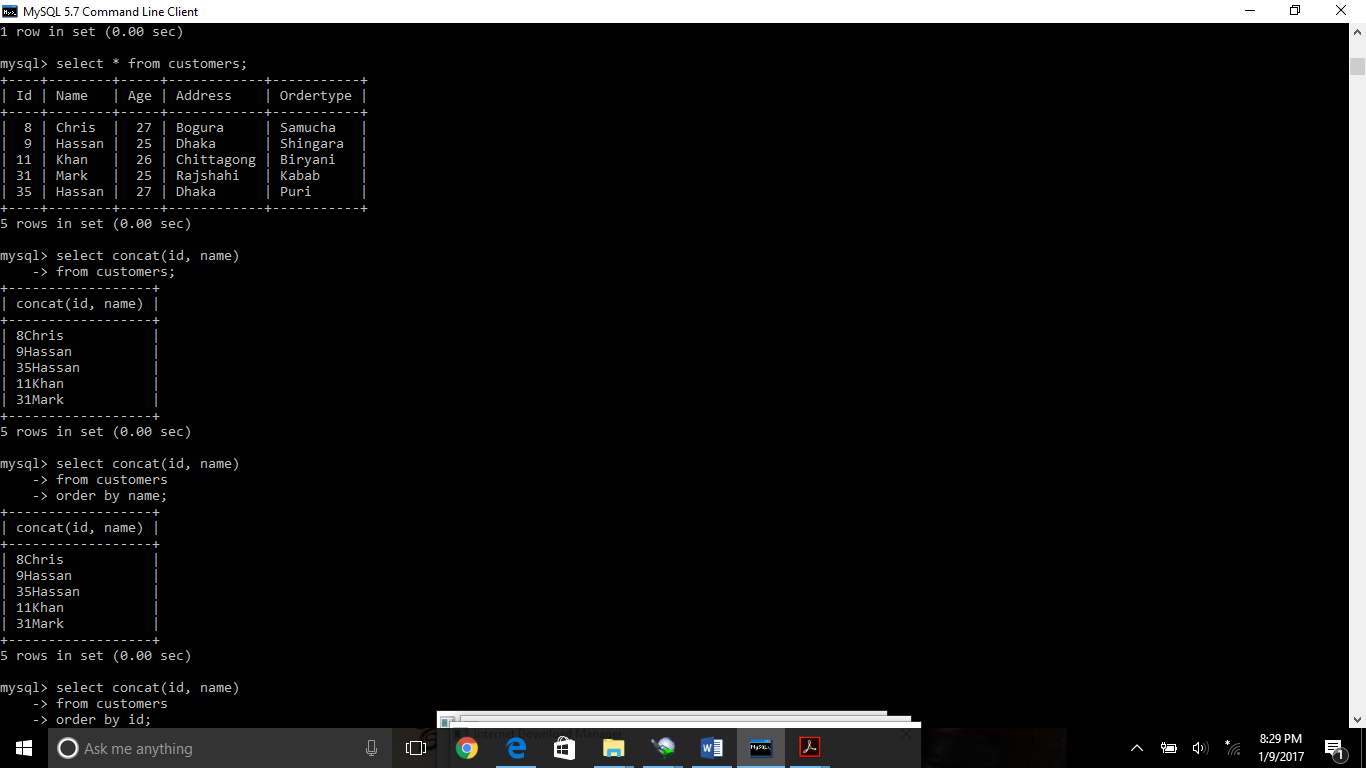
>select count(\*) from table\_name;

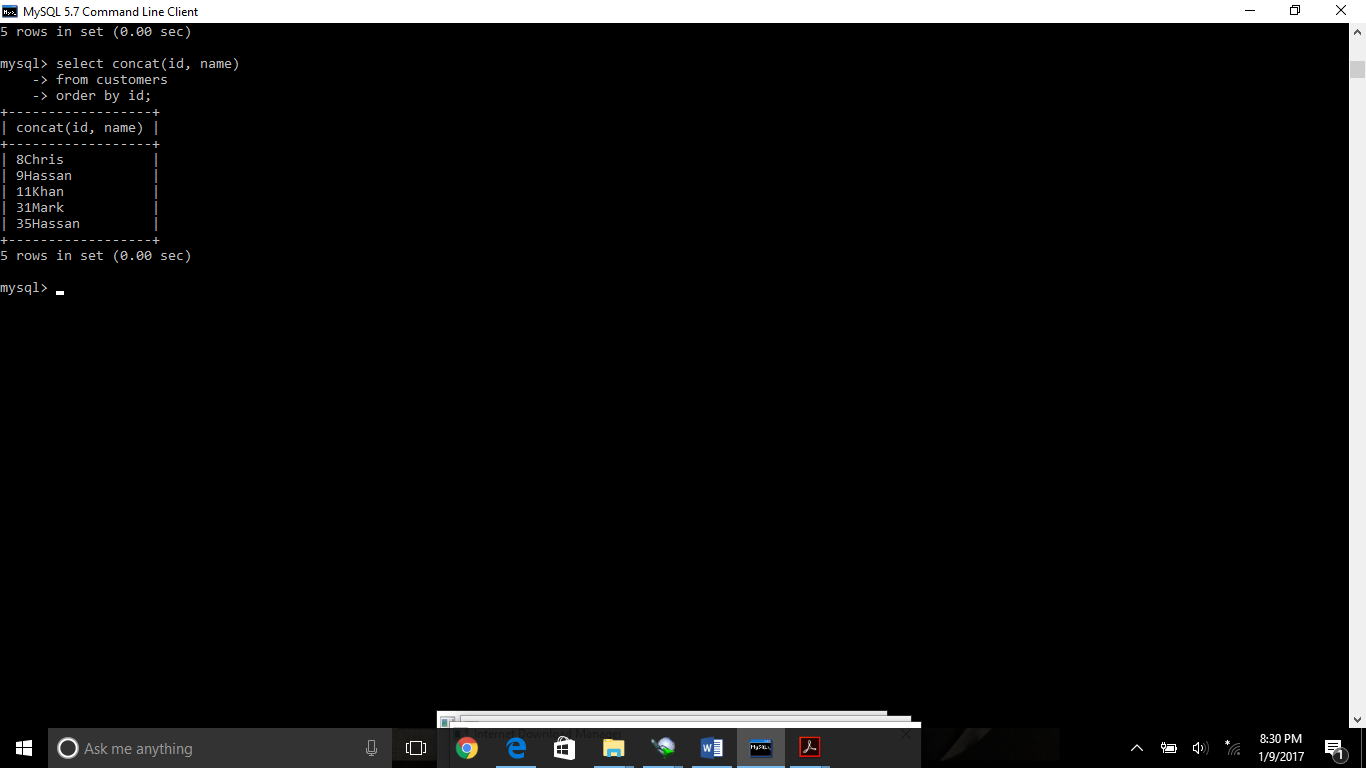


No. of rows in a particular table



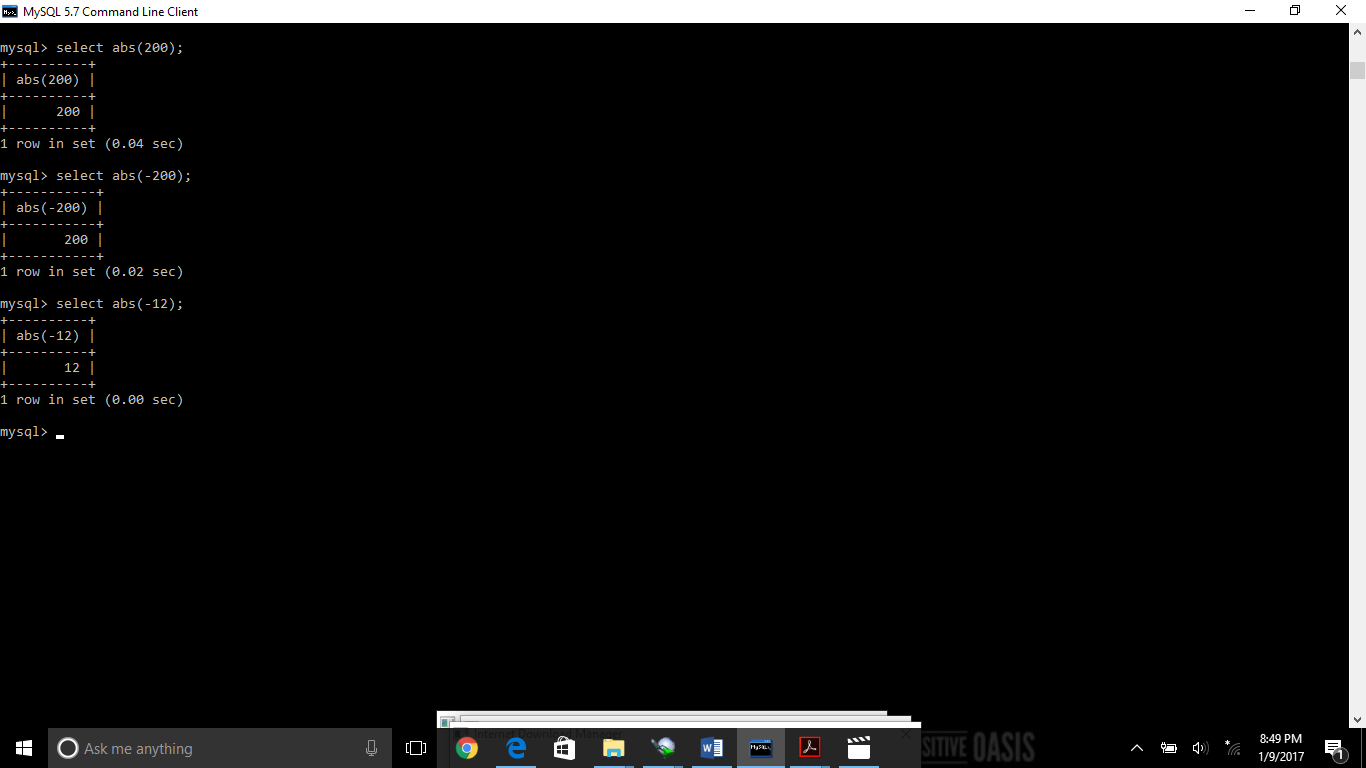
Concat Function: it is used to concatenate two different columns. We can also use order by or group by clause as needed.





Sql numeric function:

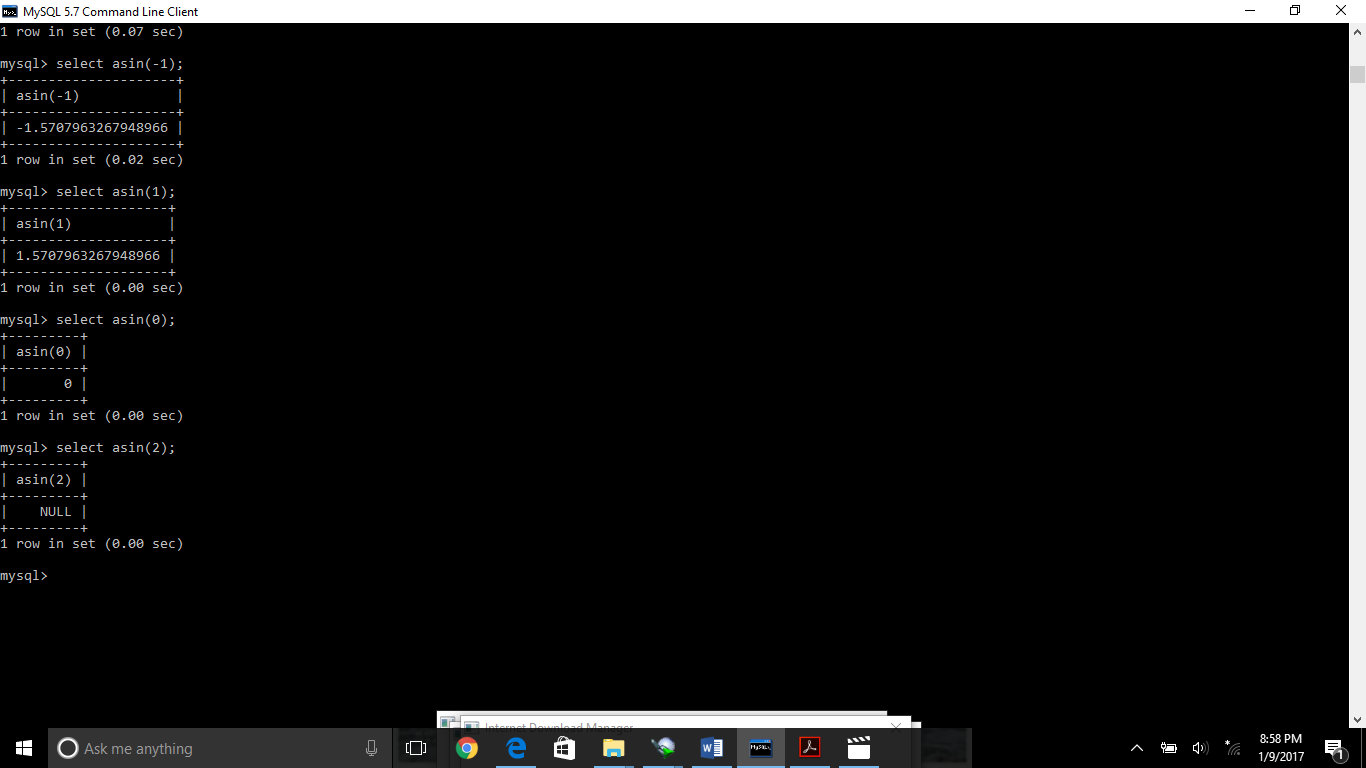
Absolute(abs) function it will give us the absolute of any numeric value.



Acos function: it will give us the accosine of any numeric value. The range should between -1 to +1 otherwise it will provide us the null value.



Asin function:



Sqrt function: to find out the square root function.

