**SQL**

****

There are 2 types of databases:

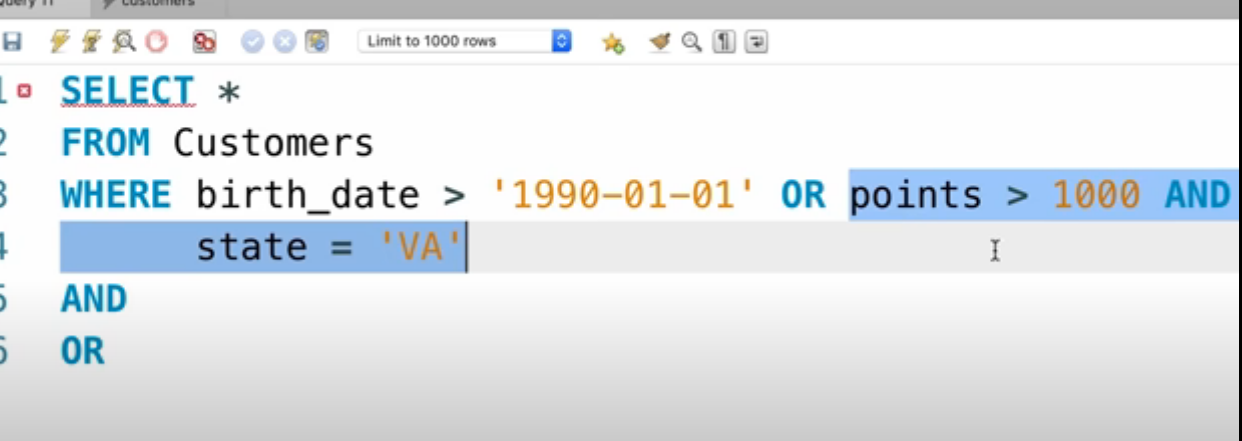
1. Relational – In this data is stored in tables, which are linked to each other using relationships. SQL is the language we use to work with relational databases. There many rdms available like MySQL, SQL Server, Oracle.
2. Non-Relational(NoSql)

SQL(Structured Query Language) or SEQUEL(Structured English Query Language)

SQL is not case-sensitive language, we may use either upper or lower case letter.

In order to use a particular database , we may use the command USE <database\_name>

AND Operator is always have higher priority then OR Operator.



In the above example, AND operator related part/expression(highlighted) is evaluated first.

In order to evaluate NOT operator, just multiply the whole expression with NOT as we usually do in mathematics. For example,

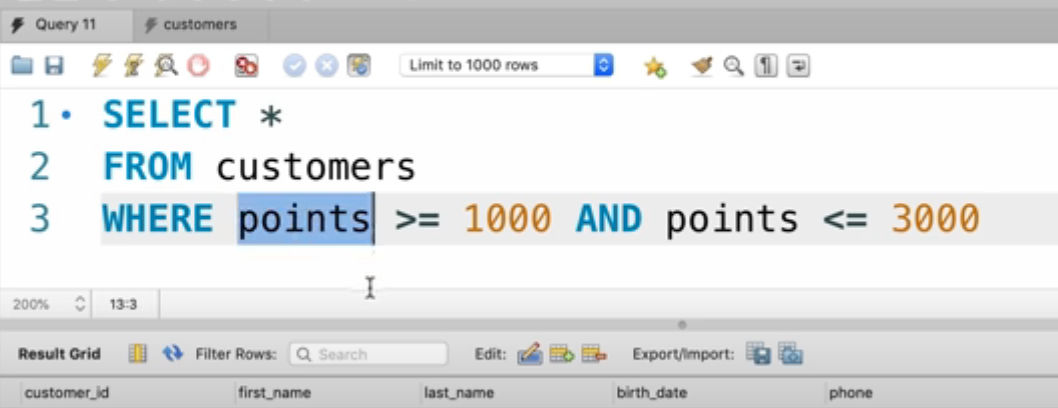
SELECT \* FROM EMPLOYEE E WHERE NOT(E.EMP\_ID < 5 OR E.SALARY < 75000 )

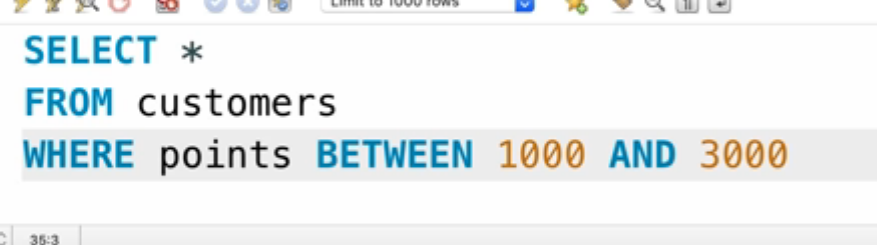
This expression can also be written as:

SELECT \* FROM EMPLOYEE E WHERE E.EMP\_ID >=5 AND E.SALARY >= 75000 )

BETWEEN OPERATOR

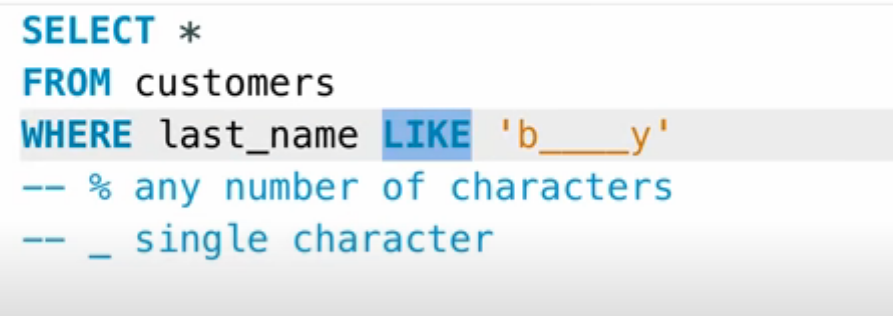
The values are inclusive in between operator.





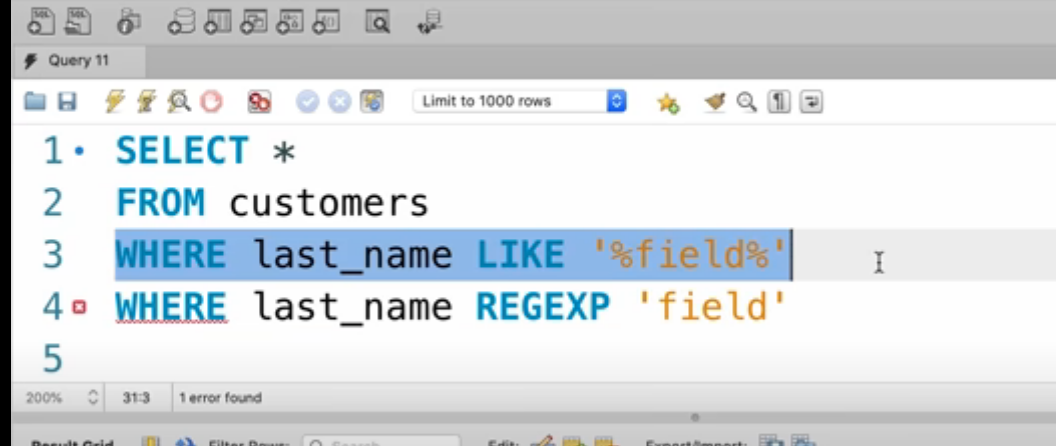
LIKE OPERATOR

Like operator can be used with %(which indicated zero or any number of characters) or with \_(which indicates a single character)



REGEX OPERATOR

Regex stands for a regular expression. We may use REGEX instead of like operator.



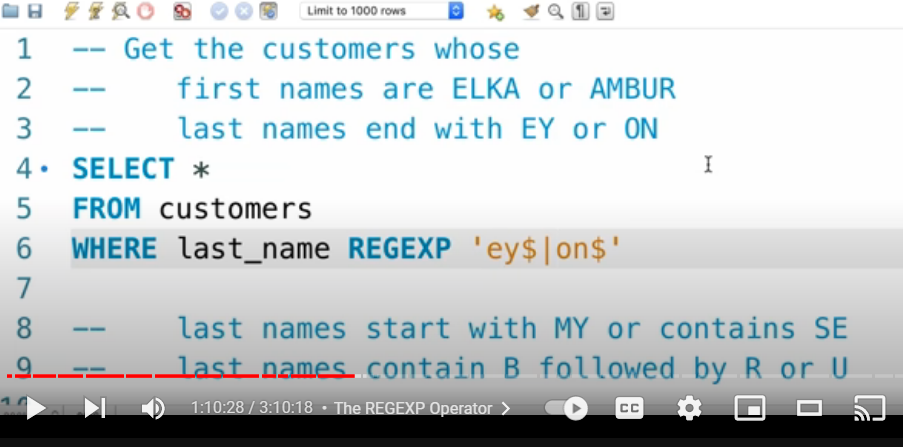
The clause in lines 3 and 4 are the same.

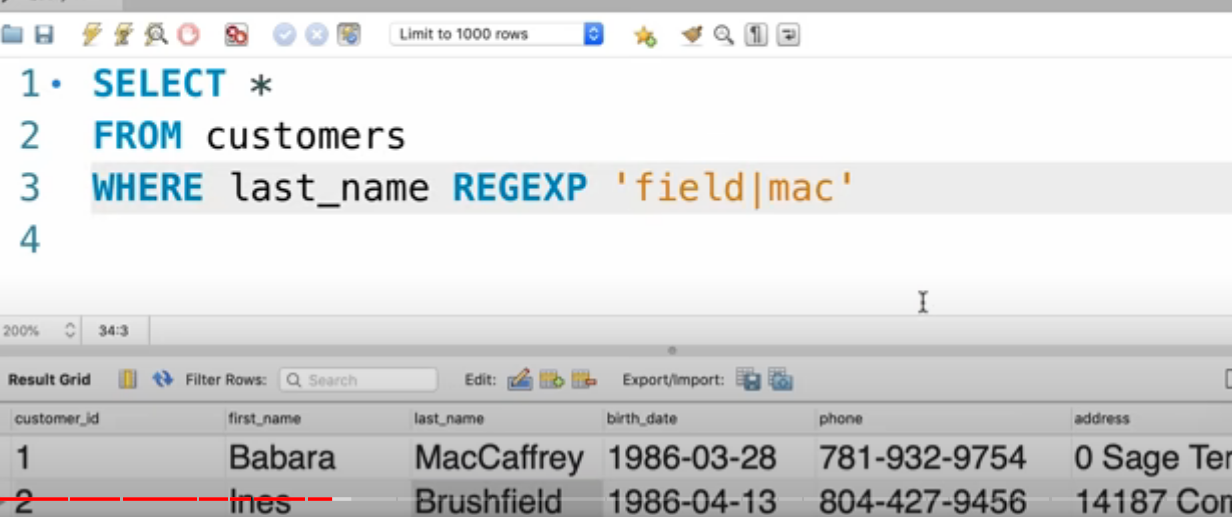
Caret sign(^) indicated that it should start with a particular String.



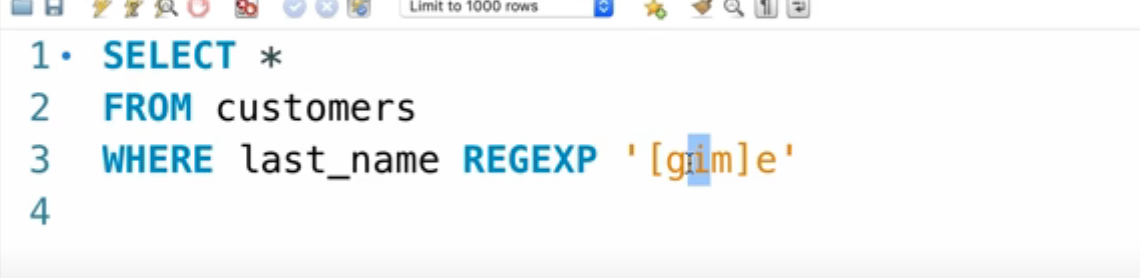
The above query returns those customers whose last\_name start with field.

Similarily, $ sign indicate it should end with string.

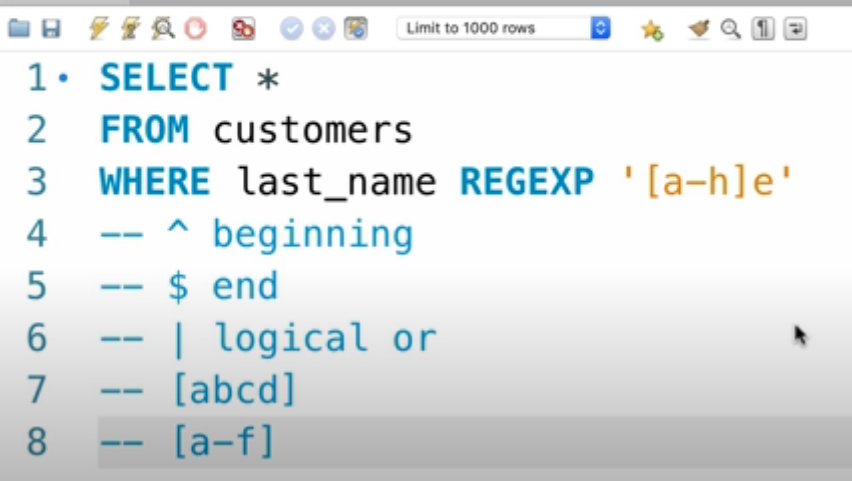




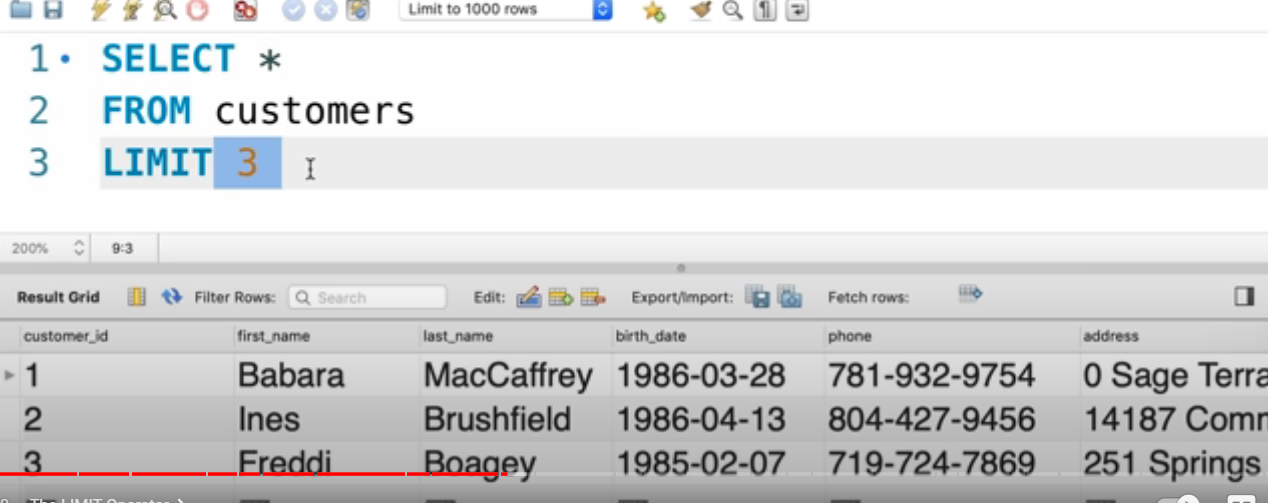
The above query returns customers whose last name either contains field or mac. Pipe(|) is used to mention that condition. It is similar to OR.



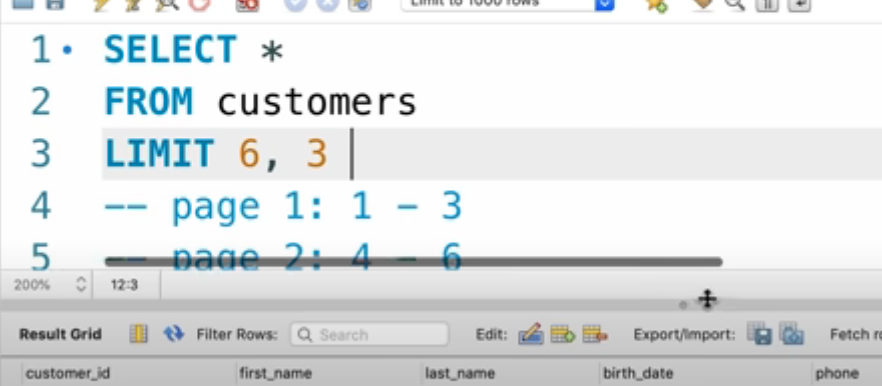
The above query returns customers whose last name has either g or I or m before e.



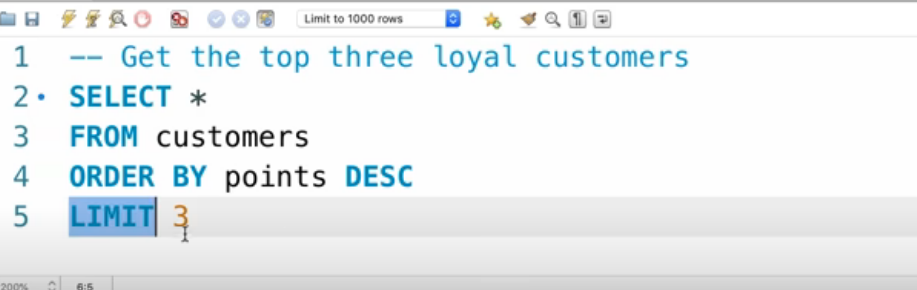
LIMIT CLAUSE



The above query return first 3 rows of a table.



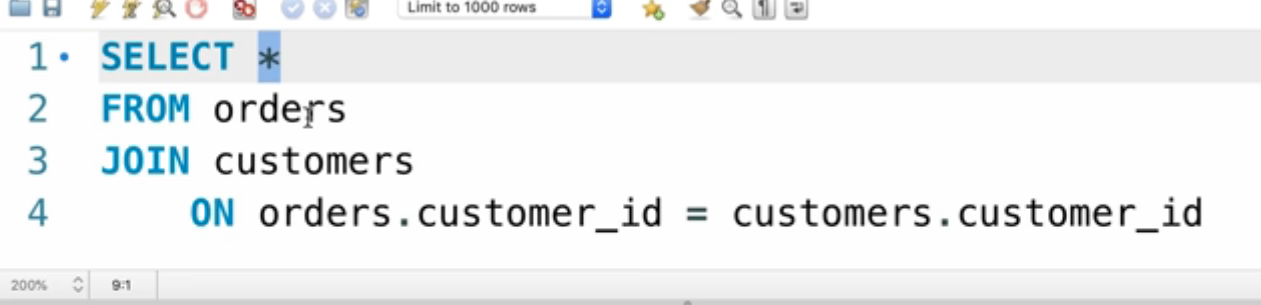
In the above query, 6 is offset. This query will skip the first 6 records and pick the next 3 records.



JOINS

Inner Join

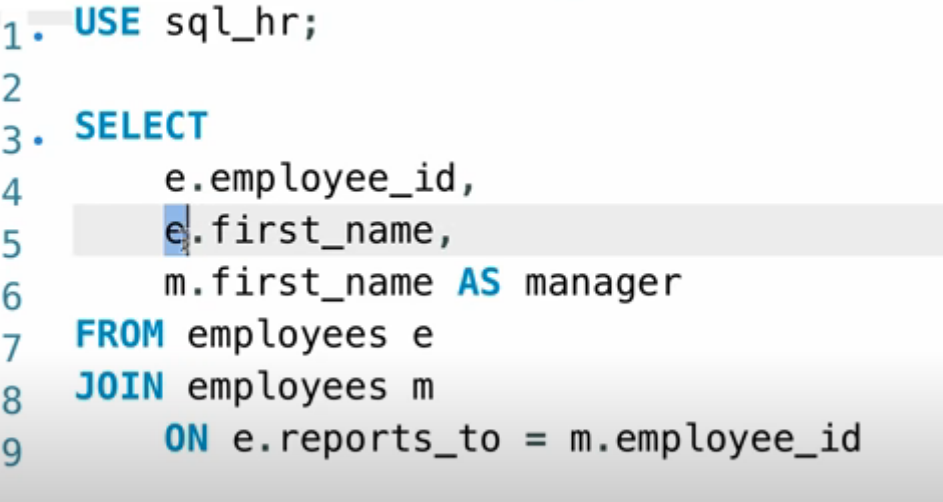
It is used to join two tables based on some criteria. We may also write inner join, but the keyword inner is optional. So whenever we use just join keyword then it indicates inner join.



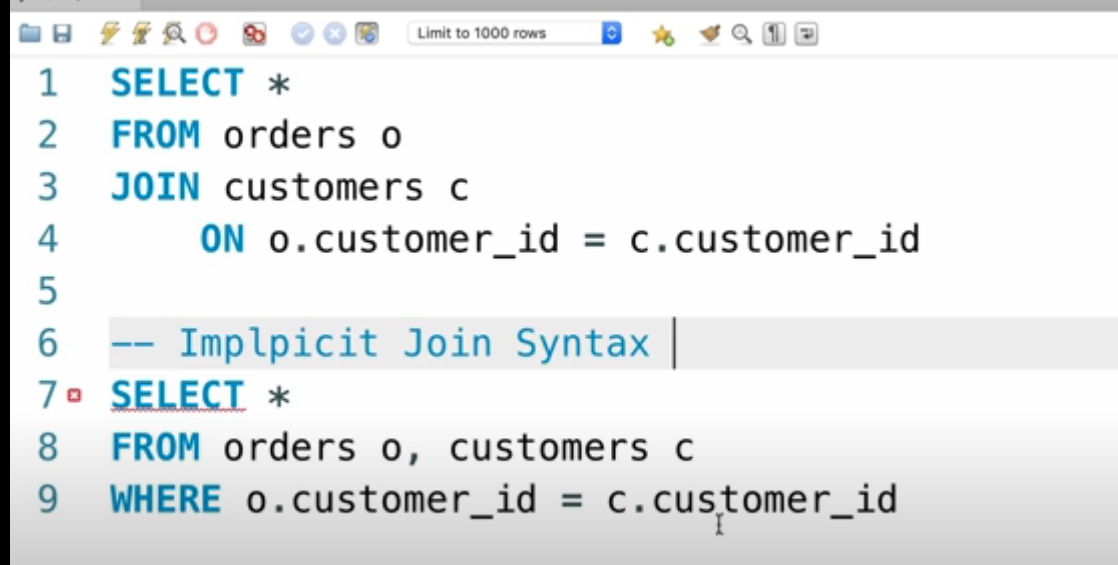
Self Join

It is used to join a table with itself. We just have to use a different alias for the same table.

In the example below, the employee and manager belong to the same table.



Implicit join Syntax



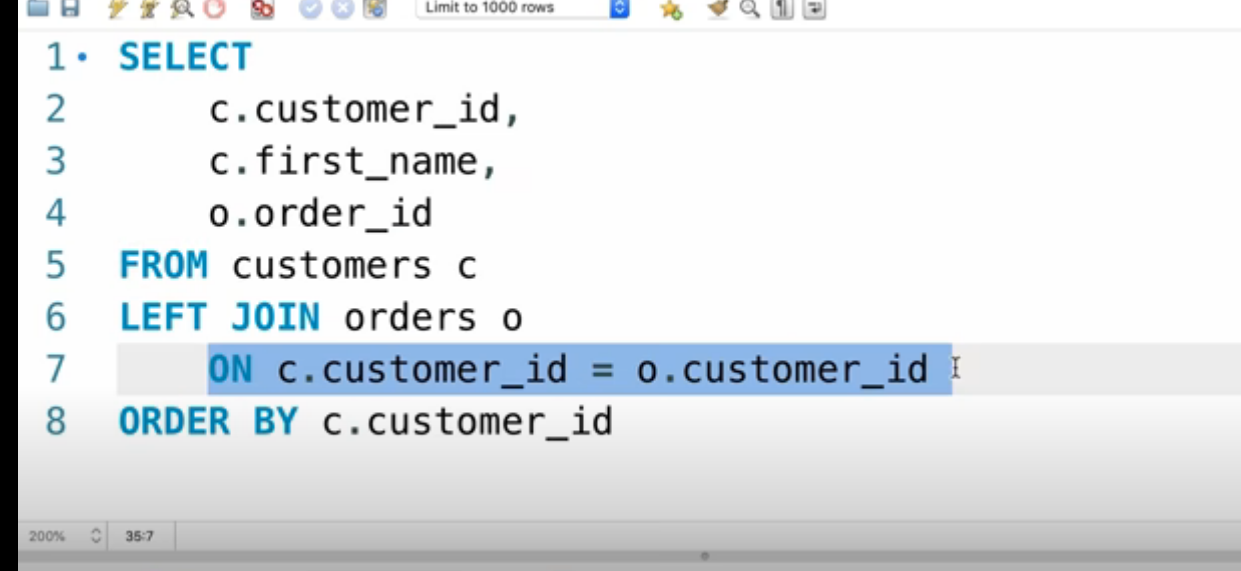
In the above-mentioned queries, the first inner join query can also be written as the second query i.e without writing the join keyword.

OUTER JOIN

Outer join returns all the entries of the table even if the mentioned condition is satisfied or not.

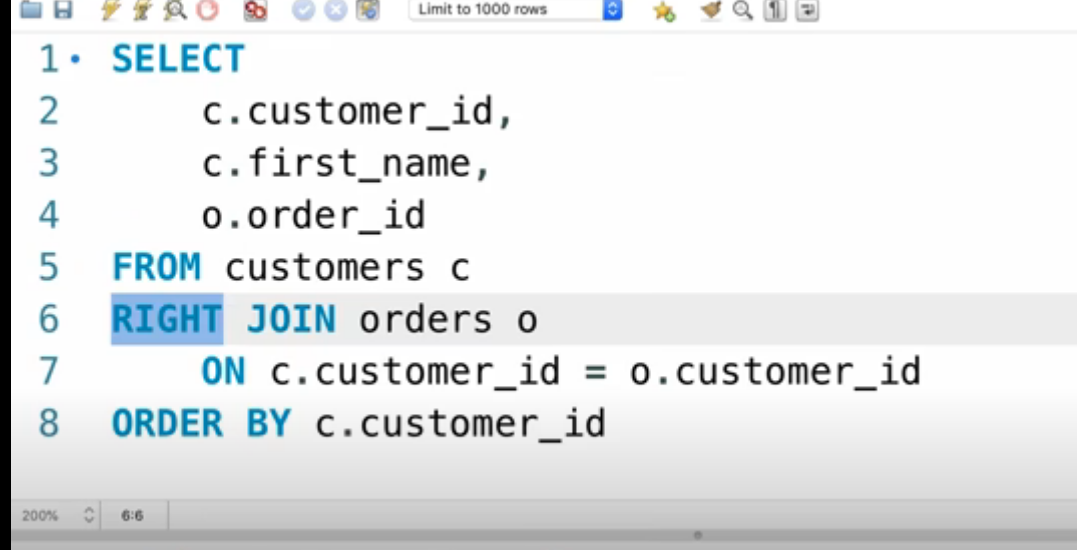
Left outer join

It returns all the entries from the left table, even if the mentioned condition is satisfied or not.



Right outer join

It returns all the entries from the right table, even if the mentioned condition is satisfied or not.



Similar to inner, outer keyword is optional we may just mention left or right with join in order to use outer join.

Self outer join

It is used to join a table with itself. Example -

SELECT e.first\_name as Employee, m.first\_name as Manger FROM

sql\_hr.employees e

left join sql\_hr.employees m on e.reports\_to = m.employee\_id;

USING clause

It can be used to simplify conditions written with join. But the name of the column should be the same in both the tables. Example –

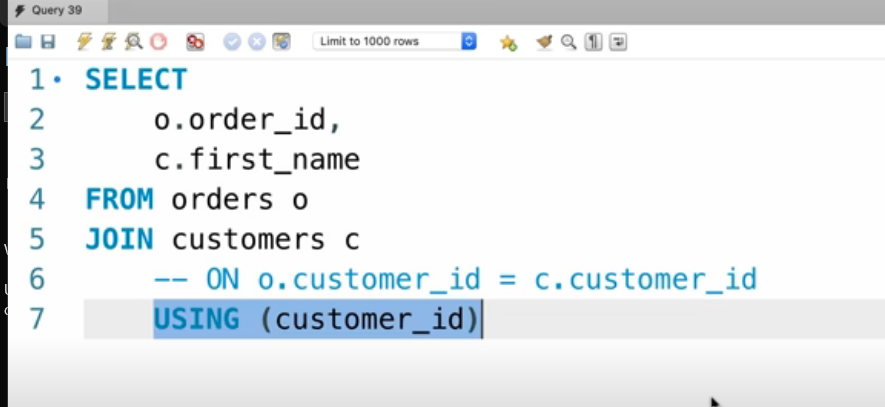
SELECT p.date,c.name,p.amount,m.name

FROM sql\_invoicing.payments p

join sql\_invoicing.clients c USING(client\_id)

join sql\_invoicing.invoices i using(invoice\_id)

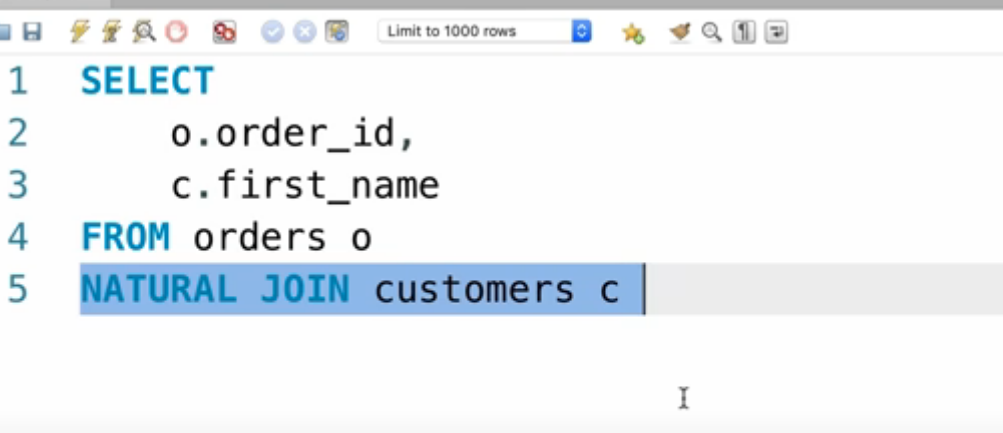
join sql\_invoicing.payment\_methods m on m.payment\_method\_id = p.payment\_method;



In the above query, expression on line 6 and 7 are same. Under the condition, that name customer\_id should be same in both the tables I,e order or customer.

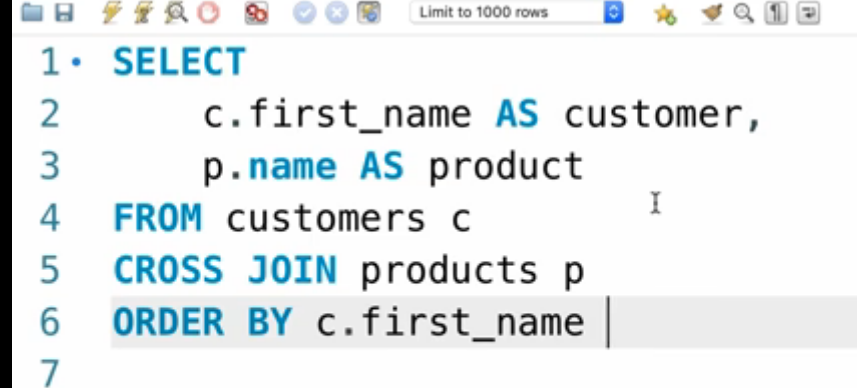
NATURAL JOIN

A NATURAL JOIN is a JOIN operation that creates an implicit join clause for you based on the common columns in the two tables being joined. It actually guesses the common columns and create join on it’s own. It may produce unexpected results in some cases so , it is generally not recommended.



CROSS JOIN

 CROSS JOINs are used to combine each row of one table with each row of another table and return the Cartesian product of the sets of rows from the tables that are joined.



UNION

It is used to combine 2 sql queries. Example –

SELECT c.customer\_id,c.first\_name,c.points, 'Bronze' as type

FROM customers c

where points <2000

UNION

SELECT c.customer\_id,c.first\_name,c.points, 'Silver' as type

FROM customers c

where points BETWEEN 2000 AND 3000

UNION

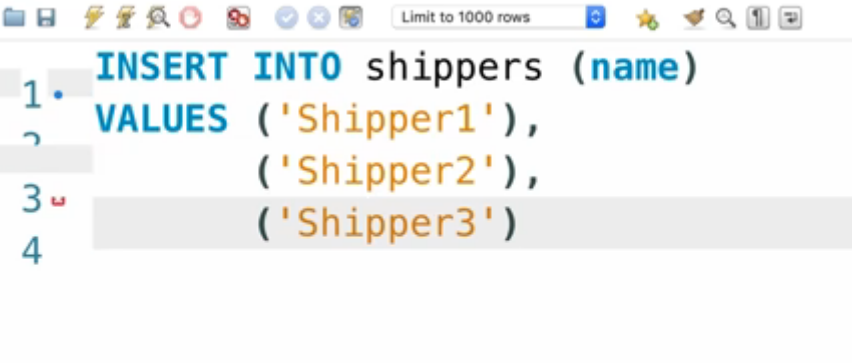
SELECT c.customer\_id,c.first\_name,c.points, 'Gold' as type

FROM customers c

where points > 3000

ORDER BY first\_name;

INSERT MULTIPLE ROWS IN THE SAME TABLE



insert into products (name,quantity\_in\_stock,unit\_price)

values (A,20,10),

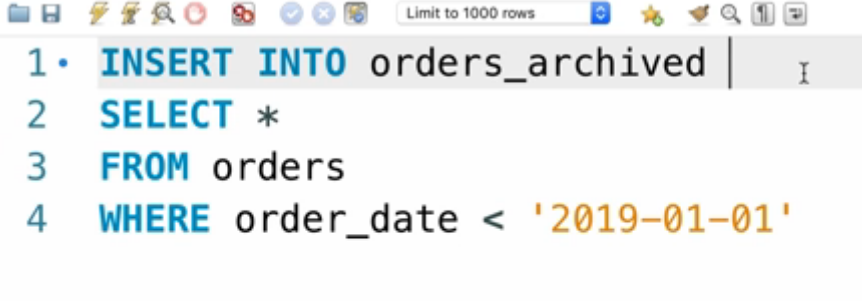
(B,22,10.5),

(C,44,100.7);

CREATING A COPY OF THE TABLE



We may also save some data in the copied table based on some conditions as mentioned below:



create table invoices\_archieve as

SELECT i.invoice\_id,i.number,c.name,i.invoice\_date,i.payment\_date,i.payment\_total FROM sql\_invoicing.invoices i

join sql\_invoicing.clients c using(client\_id);