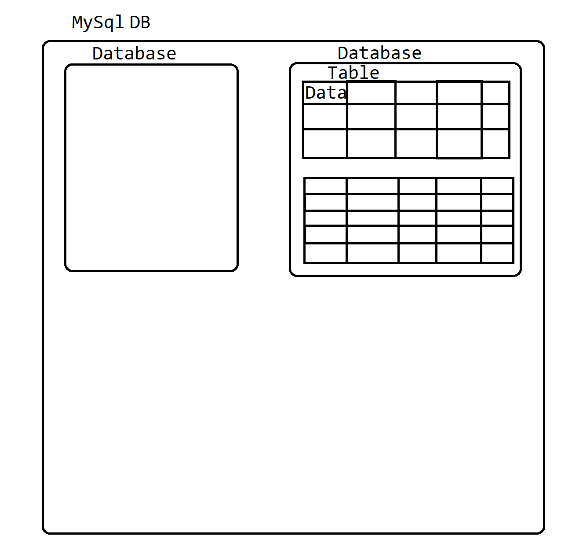
MySql Installation:

<https://dev.mysql.com/downloads/installer/>

<https://www.youtube.com/watch?v=7_dv1R-3p1c>

**Data Base**

1. Data Base is collection of data which is store in the structure manner.
2. In the data base Data is store in the form of row and column (table)
3. You can interact with the database using SQL (Structure Query Language)
4. SQL is a language using which you can execute a queries to work on the data.
5. These queries will be common for all type of Database except some minor changes.
6. There are different of the Data base
   1. Relational Database
      1. The data store in a table format (Row and Column)
      2. There will be a Primary key and Foreign Key relation in these table
      3. Example: Mysql, Oracle, Mysql server, H2, Darby, Postgrace, DB2
   2. Document Database
      1. The data store in the form of JSON.
      2. There is no primary or foreign key relation in this JSON.
      3. Example: MangoDB
   3. Graph Database
      1. The Data store in form of table or JSON.
      2. The data can be view in a visualized was by Tree structure.
      3. Example: Noe4j
7. Databases are used to store and manage the data easily and more meaningful way and you can easily get the report for analysis using queries.



**SQL (Structure Query Language)**

1. Using this Language you can interact with Database.
2. You can execute a query using CMD or UI
3. Topics included in SQL
   1. Data Type
      1. <https://www.w3schools.com/sql/sql_datatypes.asp>
   2. Different Categories Queries

DDL, DML, DCL, TCL, DQL

* 1. Constraints
  2. Different type of Joins

**Queries:**

**DDL (Data Definition Language)**

1. To Get the list of all the database in MySql

**show databases;**

1. To select a specific Database in MySql

**Use <Database\_Name>**

1. To Get the list of table in the MYSQL

**show tables;**

1. To create your own database

**Create <ObjectName> <DB\_Name>;**

Example: **create database fsd19jan;**

* 1. To create any object inside Mysql you can use a Create command
  2. The Object in database are Database, Table, Constraints, Index, View, Tigger, Procedure, Function, Cursor.

1. Create table query

Syntax: create table <table\_name> (column1 datatype, column2 datatype, column2 datatype, ….);

Example: **CREATE TABLE employee (**

**id int primary key,**

**name varchar(20) not null,**

**contact varchar(10) Unique,**

**salary double check (salary>=100000 and salary<=5000000),**

**gender char(1) check (gender in ('M','F','O')),**

**isActive Boolean,**

**doj date);**

1. Alter table
   1. Is use to modify the table or column.
   2. 3 types of alter query
      1. Add

ALTER TABLE employee **ADD** age int;

* + 1. Modify

ALTER TABLE employee **MODIFY** COLUMN age float;

* + 1. Drop

ALTER TABLE employee **DROP** COLUMN age;

1. Rename Table and column

Rename Table: ALTER TABLE employee **RENAME TO** empInfo;

Rename Column: ALTER TABLE empinfo **RENAME COLUMN** contact **TO** phone;

1. Drop Object in DB
   1. Drop is nothing but deleting the object from the DB.
   2. Wjen you drop Database and table, the data in the table will also be deleted.
   3. Syntax

Drop <object> <Objectname>

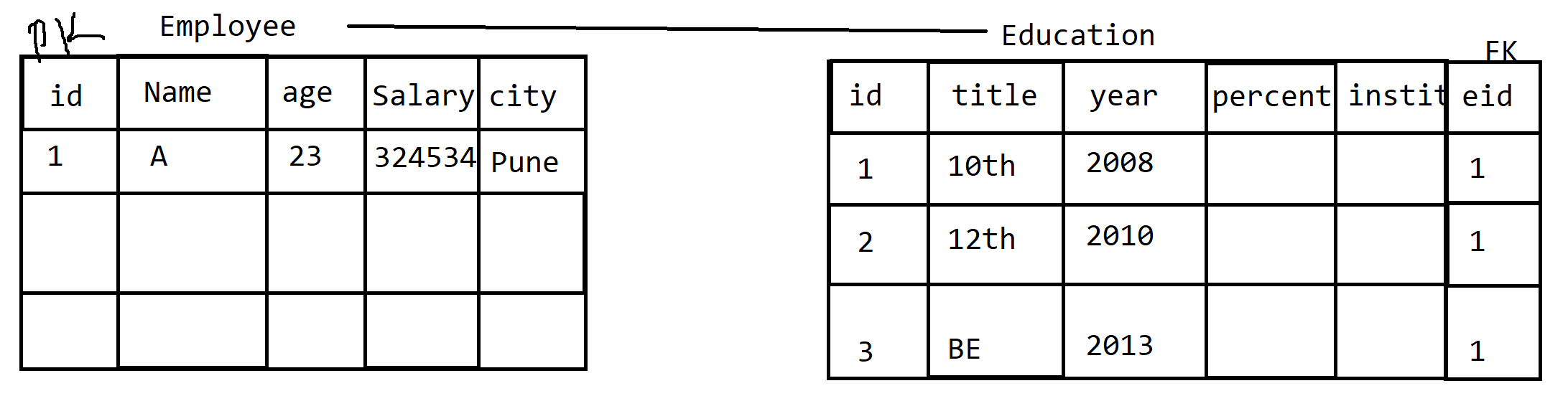
* 1. Example

Drop database fsd19jan;

Drop table empinfo;

**Sql Constraints**

1. Constraints are the rules that can be apply on the data in the table.
2. These rules are applied on the data at time of storing into table.
3. Following are the SQL contains
   1. **Not Null**: The values will never be null. Null values cannot insert into column.
   2. **Unique**: All values from the column will be unique and duplicate values are not allowed
   3. **Primary Key**: It is a combination of NotNull and Unique. By Making any column as Primary Key internally index will created for the faster search. Also This primary key column can be use as a foreign key to relate multiple table with each other.
   4. **Foreign Key:**  The values can be null and can be duplicate, these values will be linked with another table primary key.
   5. **Check:** To Set the condition on the Data like range check, Data list check.
   6. **Default:** To set the default value for the column, By default the default values will be null in the SQL.

****

**DML (Data Manipulation Language)**

1. In this type command you can work with the data.
2. Using these types of queries you create new records/data, modify existing data, delete the data, get/retrieve the data.
3. This is also known as CURD (Create Update Retrieve Delete)
4. There are different type of queries
   1. Insert: You can create or add new data into the table
   2. Update: Can update/modify existing data
   3. Delete: Can delete the records
   4. Select: can get the records
5. **Insert Query**
   1. Syntax

INSERT INTO <tableName>(Column1, Column2,….) VALUES(val1, val2, val3,…);

* 1. Example:

INSERT INTO employee(id, name, contact, salary, gender, isActive, doj) VALUES(1,'A','9988778787',3124534.43,'M',true,'2022-01-12');

INSERT INTO employee VALUES(3,'C','9943124432',1124534.43,'F',true,'2022-03-12');

In the above syntax column name is not provided, it is optional only in the case where the sequence and the values for all columns are provided.

INSERT INTO employee VALUES(6,'F','8923224432',1824534.43,'M',true,'2002-05-10'),

(7,'G','7713224432',2824534.43,'F',false,'2012-01-22'),

(8,'H','6723224432',1224534.43,'M',true,'2017-02-11');

You can insert multiple records a t a time using this syntax.

1. Select Query
   1. This query is use to get the data store into the database.
   2. Select query can be use with clauses, operators, function(Date, String, arithmetic function), Joins.
   3. Syntax:

Select column1, column2, column3,… from <TableName>

Select \* from <TableName>

1. Update Query
   1. Using update query you can update the data from existing row.
   2. You can update multiple rows or single rows at a time
   3. Syntax:

UPDATE <tableName> SET column=value;

UPDATE <tableName> SET column=value where condition;

1. Delete Query
   1. Using delete query you can delete a specific record or all the records.
   2. Syntax:

DELETE FROM <TABLENAME>;

DELETE FROM <TABLENAME> WHERE condition;

* 1. Example:

DELETE FROM employee WHERE id = 2;

**Clause**

1. Clause are use to add a filtration while retrieving rows/data from table.
2. Operators are mostly use to write a condition on the data.
3. Some of the commonly used clauses
   1. Where Clause
      1. In the where clause you can write a conditions on the column so that only a specific records will be selected. These is one of the way using which you can filter the records from the database.
   2. Like Clause
      1. In this clause you can get the record based on partial values.
      2. ‘%’ in this query denotes any match.
      3. Example:   
         select \* from employee where contact like '99%';

select \* from employee where contact like '%8877%';

select \* from employee where contact like '%8877';

* 1. Limit clause
     1. Limit clause is use to limit the number of rows getting selected by query.
     2. Example:

select \* from employee limit 5;

* 1. Order by Clause
     1. You can select record in the order.
     2. Records can be selected by Ascending (asc) and Descending (desc) order.
     3. Example

select \* from employee ORDER BY salary desc;

select \* from employee ORDER BY salary desc;

select \* from employee ORDER BY salary,name desc;

select \* from employee ORDER BY salary desc, name asc;

* 1. Distinct clause
     1. You can select only the unique values for a specific column
     2. Example:

SELECT DISTINCT salary FROM employee;

SELECT DISTINCT gender FROM employee;

* 1. Group By Clause
     1. Is use to group the records using the functionality.
     2. Group by clause is mostly used with an aggregate functions.
     3. Group By Cluse is use with select query.
     4. Syntax:

SELECT column, function(column) from Table where Condition GROUP BY column

* 1. Having Clause
     1. Is use to apply the condition on the group records.
     2. This has to apply with the GROUP By clause.
     3. Syntax:

SELECT column, function(column) from Table where Condition

GROUP BY column HAVING condition

mysql> select count(isActive) as count, sum(salary), isActive from employee **group by** isActive;

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

| count | sum(salary) | isActive |

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

| 6 | 11229473.795 | 1 |

| 3 | 5808510.176 | 0 |

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

2 rows in set (0.00 sec)

mysql> select count(isActive) as count, sum(salary), isActive from employee **group by** isActive **HAVING count>5**;

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

| count | sum(salary) | isActive |

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

| 6 | 11229473.795 | 1 |

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

1 row in set (0.00 sec)

**Operators**

* + - 1. AND, OR operator
         1. These operators are use to combine multiple condition together.
      2. Between operator
         1. Is use to select a record which is in the range of two values.
         2. Syntax:

SELECT \* FROM <TableName> where column BETWEEN value1 AND value2;

Example :

select \* from employee where salary BETWEEN 1000000 and 2000000;

* + - 1. In Operator
         1. Is use to select a records base on the list of values
         2. Example:

select \* from employee where name in ('A','B','C','D');

* + - 1. Not Operator
         1. Is use to reverse the condition.
         2. Example:

select \* from employee where NOT isActive = 0;

**Function**

* + - 1. Functions are pre define implementation provided by SQL.
      2. There are different categories of function
         1. Aggregate Function
         2. String Function
         3. Date Function

**Aggregate Function**

In this type values are group together and the single value as an output

Count()

**mysql> select count(id) from employee;**

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

| count(id) |

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

| 7 |

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

1 row in set (0.01 sec)

**mysql> select count(id) from employee where isActive = 1;**

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

| count(id) |

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

| 5 |

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

1 row in set (0.01 sec)

**mysql> select count(id) as totalEmp from employee where isActive = 1;**

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

| totalEmp |

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

| 5 |

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

1 row in set (0.00 sec)

Sum()

**mysql> select sum(salary) from employee;**

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

| sum(salary) |

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

| 14488915.111 |

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

1 row in set (0.00 sec)

**mysql> select sum(salary) as totalSalary from employee;**

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

| totalSalary |

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

| 14488915.111 |

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

1 row in set (0.00 sec)

**mysql> select sum(salary) as totalSalary from employee where isActive=0;**

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

| totalSalary |

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

| 4783975.746 |

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

1 row in set (0.00 sec)

Avg()

**mysql> select avg(salary) as averageSalary from employee where isActive=1;**

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

| averageSalary |

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

| 1940987.8730000001 |

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

1 row in set (0.00 sec)

**mysql> select avg(salary) as averageSalary from employee;**

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

| averageSalary |

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

| 2069845.0158571429 |

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

1 row in set (0.00 sec)

Min()

**mysql> select min(salary) as minimumSalary from employee;**

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

| minimumSalary |

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

| 1236987.873 |

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

1 row in set (0.01 sec)

**mysql> select min(salary) as minimumSalary from employee where isActive=1;**

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

| minimumSalary |

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

| 1236987.873 |

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

1 row in set (0.00 sec)

Max()

**mysql> select max(salary) as maximumSalary from employee where isActive=1;**

**mysql> select max(salary) as maximumSalary from employee;**

**String function**

1. Upper()

Is use to convert string value in to Capital case

mysql> select id, **upper(name)** as name, contact from employee;

1. Lower()

Is use to convert string into small case.

mysql> select id, **lower(name)** as name, contact from employee;

1. Concat()

Is use to combine the multiple column into a single column

select id, **concat(name,'-' ,contact)** from employee;

1. length()

Is use to get the total count of character in the string

mysql> select id, contact, **length(contact)** from employee;

1. trim()

Is use to remove the spaces from both the side (left and right) of the string. It will not remove the space which is in between the word.

select id,**trim(name)** from employee;

1. ltrim() and rtrim()

is use to remove the spaces from left side (ltrim) and right side (rtrim) of the string

mysql> select id,**rtrim(name)** from employee;

mysql> select id,**ltrim(name)** from employee;

**Date Function**

1. now()

To get the system data and time

**mysql> select now();**

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

| now() |

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

| 2022-04-07 08:02:07 |

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

1 row in set (0.01 sec)

1. curdate()

to get the current system date

**mysql> select curdate();**

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

| curdate() |

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

| 2022-04-07 |

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

1 row in set (0.00 sec)

1. curtime()

can get the current system time

**mysql> select curtime();**

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

| curtime() |

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

| 08:02:56 |

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

1 row in set (0.00 sec)

1. **year()**

Is use to get the year from the given date

**mysql> select id,name,year(doj) from employee;**

1. month()

Is use to get the month from the given date

**mysql> select id,name,month(doj) from employee;**

1. monthname()

Is use to get the month in word from the given date

**mysql> select id,name,monthname(doj) from employee;**

1. day()

is use to print the day in the given date

**mysql> select id,name,day(doj) from employee;**

1. datediff()

to Get the difference in two dates in day.

**mysql> select id,name,doj, datediff(now(), doj) from employee;**

1. date\_format()

is use to fomrat the date

**mysql> select id,name,date\_format(doj,'%d %M %Y') from employee;**

**Queries:**

mysql> select \* from employee where isActive=1;

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

| id | name | contact | salary | gender | isActive | doj |

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

| 1 | A | 9988778787 | 3436987.873 | M | 1 | 2022-01-12 |

| 3 | C | 9943124432 | 1236987.873 | F | 1 | 2022-03-12 |

| 4 | D | 9913124432 | 1676987.873 | M | 1 | 2021-03-15 |

| 6 | F | 8923224432 | 2006987.873 | M | 1 | 2002-05-10 |

| 8 | H | 6723224432 | 1346987.873 | M | 1 | 2017-02-11 |

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

5 rows in set (0.01 sec)

mysql> select \* from employee where isActive=1 AND gender='M';

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

| id | name | contact | salary | gender | isActive | doj |

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

| 1 | A | 9988778787 | 3436987.873 | M | 1 | 2022-01-12 |

| 4 | D | 9913124432 | 1676987.873 | M | 1 | 2021-03-15 |

| 6 | F | 8923224432 | 2006987.873 | M | 1 | 2002-05-10 |

| 8 | H | 6723224432 | 1346987.873 | M | 1 | 2017-02-11 |

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

4 rows in set (0.00 sec)

mysql> select \* from employee where isActive=1 AND gender='F';

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

| id | name | contact | salary | gender | isActive | doj |

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

| 3 | C | 9943124432 | 1236987.873 | F | 1 | 2022-03-12 |

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

1 row in set (0.00 sec)

mysql> select \* from employee;

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

| id | name | contact | salary | gender | isActive | doj |

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

| 1 | A | 9988778787 | 3436987.873 | M | 1 | 2022-01-12 |

| 3 | C | 9943124432 | 1236987.873 | F | 1 | 2022-03-12 |

| 4 | D | 9913124432 | 1676987.873 | M | 1 | 2021-03-15 |

| 5 | E | 7713124432 | 1676987.873 | M | 0 | NULL |

| 6 | F | 8923224432 | 2006987.873 | M | 1 | 2002-05-10 |

| 7 | G | 7713224432 | 3106987.873 | F | 0 | 2012-01-22 |

| 8 | H | 6723224432 | 1346987.873 | M | 1 | 2017-02-11 |

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

7 rows in set (0.00 sec)

mysql> select \* from employee where salary>1000000 AND salary<2000000;

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

| id | name | contact | salary | gender | isActive | doj |

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

| 3 | C | 9943124432 | 1236987.873 | F | 1 | 2022-03-12 |

| 4 | D | 9913124432 | 1676987.873 | M | 1 | 2021-03-15 |

| 5 | E | 7713124432 | 1676987.873 | M | 0 | NULL |

| 8 | H | 6723224432 | 1346987.873 | M | 1 | 2017-02-11 |

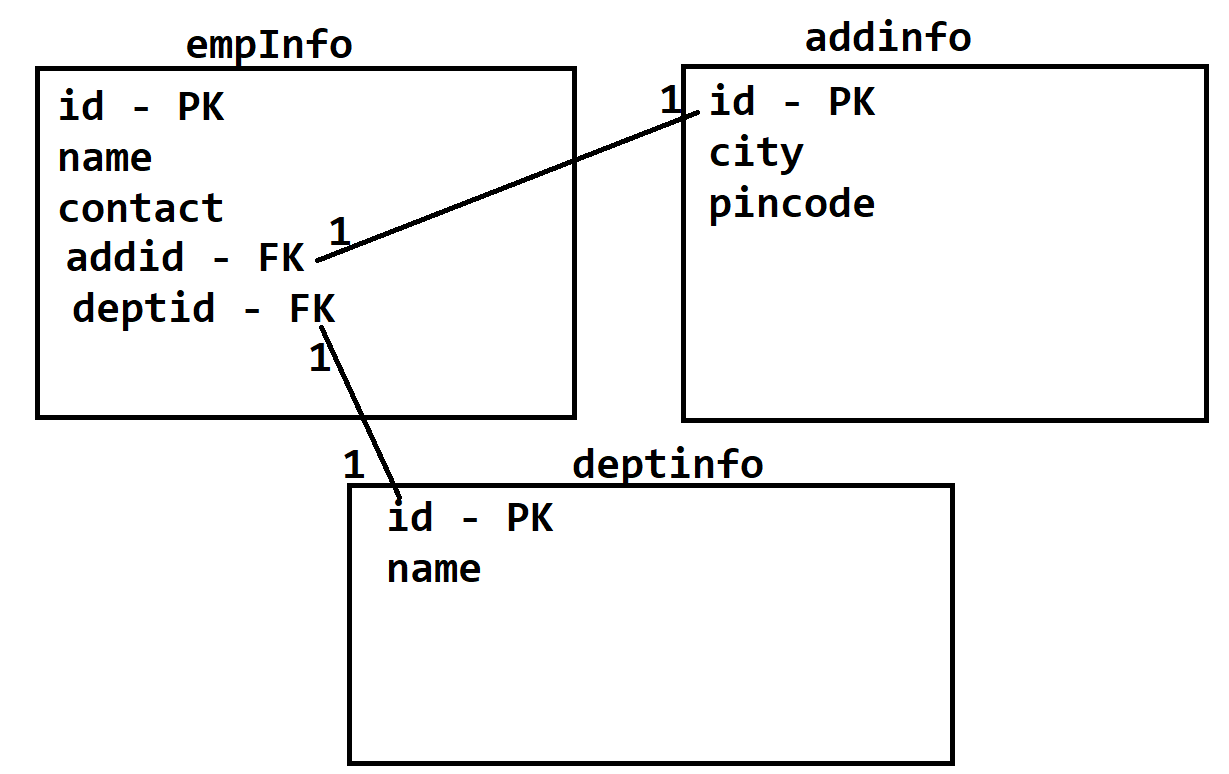
+----+------+------------+-------------+--------+----------+------------+

4 rows in set (0.00 sec)

**Join**

1. Is use to combine two or multiple tables together and generate the result.
2. Joins always use with a Select Query.
3. Types Of Join
   1. Cross Join (Cartesian Join)
   2. Inner Join
   3. Outer Join
      1. Left Outer Join
      2. Right Outer Join
      3. Full Join





create table **empinfo**(

id int primary key,

name varchar(20),

contact varchar(10),

addid int,

deptid int,

FOREIGN key(addid) references addinfo(id),

FOREIGN KEY(deptid) references deptinfo(id));

insert into deptinfo values(1,'Development'),(2,'Testing'),(3,'HR'),(4,'Account');

insert into addinfo values(1,'Pune','456454'),(2,'Pune','336454'),(3,'Mumbai','445554');

insert into empinfo values(1, 'A', '9988776656',2 ,1);

insert into empinfo values(2, 'C', '7788776656',1 ,1),(3, 'D', '7888776656',3 ,2);

insert into empinfo(id,name,contact,addid) values(4, 'X', '7788773234',2 ),(5, 'Y', '7888776655',3);

mysql> select \* from deptinfo;

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

| id | name |

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

| 1 | Development |

| 2 | Testing |

| 3 | HR |

| 4 | Account |

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

4 rows in set (0.00 sec)

mysql> select \* from addinfo;

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

| id | city | pincode |

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

| 1 | Pune | 456454 |

| 2 | Pune | 336454 |

| 3 | Mumbai | 445554 |

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

3 rows in set (0.00 sec)

mysql> select \* from empinfo;

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

| id | name | contact | addid | deptid |

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

| 1 | A | 9988776656 | 2 | 1 |

| 2 | C | 7788776656 | 1 | 1 |

| 3 | D | 7888776656 | 3 | 2 |

| 4 | X | 7788773234 | 2 | NULL |

| 5 | Y | 7888776655 | 3 | NULL |

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

1. rows in set (0.00 sec)

**Cross Join:**

1. Join with every row of one table with another table.
2. There is no where clause in the cross join.

mysql> select \* from empinfo CROSS JOIN addinfo;

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

| id | name | contact | addid | deptid | id | city | pincode |

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

| 1 | A | 9988776656 | 2 | 1 | 3 | Mumbai | 445554 |

| 1 | A | 9988776656 | 2 | 1 | 2 | Pune | 336454 |

| 1 | A | 9988776656 | 2 | 1 | 1 | Pune | 456454 |

| 2 | C | 7788776656 | 1 | 1 | 3 | Mumbai | 445554 |

| 2 | C | 7788776656 | 1 | 1 | 2 | Pune | 336454 |

| 2 | C | 7788776656 | 1 | 1 | 1 | Pune | 456454 |

| 3 | D | 7888776656 | 3 | 2 | 3 | Mumbai | 445554 |

| 3 | D | 7888776656 | 3 | 2 | 2 | Pune | 336454 |

| 3 | D | 7888776656 | 3 | 2 | 1 | Pune | 456454 |

| 4 | X | 7788773234 | 2 | NULL | 3 | Mumbai | 445554 |

| 4 | X | 7788773234 | 2 | NULL | 2 | Pune | 336454 |

| 4 | X | 7788773234 | 2 | NULL | 1 | Pune | 456454 |

| 5 | Y | 7888776655 | 3 | NULL | 3 | Mumbai | 445554 |

| 5 | Y | 7888776655 | 3 | NULL | 2 | Pune | 336454 |

| 5 | Y | 7888776655 | 3 | NULL | 1 | Pune | 456454 |

| 6 | Z | 9988771122 | NULL | NULL | 3 | Mumbai | 445554 |

| 6 | Z | 9988771122 | NULL | NULL | 2 | Pune | 336454 |

| 6 | Z | 9988771122 | NULL | NULL | 1 | Pune | 456454 |

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

18 rows in set (0.01 sec)

**Inner Join**

1. The rows from both the table will be selected if they are matches with the condition
2. Syntax:

SELECT column(s)

from Table1 INNER JOIN Table1

ON Condition

INNER JOIN table3

ON Condition

INNER JOIN table3

ON Condition

mysql> select \* from empinfo **INNER JOIN** addinfo ON empinfo.addid=addinfo.id;

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

| id | name | contact | addid | deptid | id | city | pincode |

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

| 2 | C | 7788776656 | 1 | 1 | 1 | Pune | 456454 |

| 1 | A | 9988776656 | 2 | 1 | 2 | Pune | 336454 |

| 4 | X | 7788773234 | 2 | NULL | 2 | Pune | 336454 |

| 3 | D | 7888776656 | 3 | 2 | 3 | Mumbai | 445554 |

| 5 | Y | 7888776655 | 3 | NULL | 3 | Mumbai | 445554 |

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

5 rows in set (0.00 sec)

mysql> select name,contact,city,pincode from empinfo **INNER JOIN** addinfo ON empinfo.addid=addinfo.id AND empinfo.name='A';

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

| name | contact | city | pincode |

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

| A | 9988776656 | Pune | 336454 |

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

1 row in set (0.00 sec)

mysql> select emp.id,emp.name,contact,city,pincode,d.name as department

-> from empinfo emp **INNER JOIN** addinfo a

-> ON emp.addid=a.id

-> **INNER JOIN** deptInfo d

-> ON emp.deptId = d.id;

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

| id | name | contact | city | pincode | department |

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

| 2 | C | 7788776656 | Pune | 456454 | Development |

| 1 | A | 9988776656 | Pune | 336454 | Development |

| 3 | D | 7888776656 | Mumbai | 445554 | Testing |

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

3 rows in set (0.00 sec)

**Left Join**

1. In the LEFT JOIN all the (matching and non-matching) records selected from the left side table and only the matching records of the right side table will be selected.
2. Syntax:

SELECT column(s)

from Table1 **LEFT JOIN** Table2

ON Condition

**LEFT JOIN** table3

ON Condition

**LEFT JOIN** table3

ON Condition

mysql> select emp.id,emp.name,contact,city,pincode

-> from empinfo emp LEFT JOIN addinfo a

-> ON emp.addid=a.id;

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

| id | name | contact | city | pincode |

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

| 1 | A | 9988776656 | Pune | 336454 |

| 2 | C | 7788776656 | Pune | 456454 |

| 3 | D | 7888776656 | Mumbai | 445554 |

| 4 | X | 7788773234 | Pune | 336454 |

| 5 | Y | 7888776655 | Mumbai | 445554 |

| 6 | Z | 9988771122 | NULL | NULL |

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

1. rows in set (0.00 sec)

**Right Join**

1. In the RIGHT JOIN all the (matching and non-matching) records selected from the right side table and only the matching records of the left side table will be selected.
2. Syntax:

SELECT column(s)

from Table1 **RIGHT JOIN** Table2

ON Condition

**RIGHT JOIN** table3

ON Condition

**RIGHT JOIN** table3

ON Condition

mysql> select emp.id,emp.name,contact,d.name as department

-> from empinfo emp RIGHT JOIN deptinfo d

-> ON emp.deptid = d.id;

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

| id | name | contact | department |

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

| 2 | C | 7788776656 | Development |

| 1 | A | 9988776656 | Development |

| 3 | D | 7888776656 | Testing |

| NULL | NULL | NULL | HR |

| NULL | NULL | NULL | Account |

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

5 rows in set (0.00 sec)

**Full Join**

1. In the FULL JOIN all the (matching and non-matching) records selected from both the tables.
2. Syntax:

SELECT column(s)

from Table1 **FULL JOIN** Table2

ON Condition

**FULL JOIN** table3

ON Condition

**FULL JOIN** table3

ON Condition

Example: **(FULL OUTER join is not supported in MYSQL it can be use in Oracle, Postgrese)**

select emp.id,emp.name,contact,d.name as department

from empinfo emp **FULL JOIN** deptinfo d

ON emp.deptid = d.id;