**DataBase**

1. It is use to store the data and maintain the data.
2. Using Database you can generated reposted easily just by using query.
3. The data stored inside database will be available for a longer period of time.
4. There are different type of databases present
   1. **Relational Database**
      1. In this database the records will be store in the structure format that is in the form of rows and columns (table format).
      2. The data can be store in a distributed way into multiple table. And those table will be related with each other by a primary and foreign key.
      3. The data store into the database is also known as structure Data.
      4. Example: **MySql**, Oracle, SQL Server, Postgrace etc.
   2. Document Database
      1. Is use to store the data in the form of document (.JSON)
      2. In this type of database there is not table structure present and cannot store data into a distributes way using primary and foreign key.
      3. Example:

MongoDB, Cassandra etc.

* 1. Graph Database
     1. The data will be store in the form of tables/rows and columns. But is represented as a graph.
     2. Example:

Neo4J

Download Database and Installation

MySql:

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



Install MySql:

<https://www.youtube.com/watch?v=OM4aZJW_Ojs>

**Important**

Note down the port number (3306), username (root) and password during the Setup.

SQL (Structure Query Language)

1. It is use to execute the queries inside database.
2. The SQL is use to interact with the database.
3. The SQL is distributed into 5 categories
   1. **DDL (Data Definition Language)**
      1. It is use to create, modify or delete the structure of the table and the different database object such as table, index, triggers, function etc.
      2. SQL Operation like **CREATE, ALTER, DROP** can be use on the Objects
   2. DML (Data Manipulation language)
      1. Using this you can insert, update or remove the data from the table. It is use to perform the data specific operation.
      2. SQL operations like **INSERT, UPDATE, DELETE** can be perform in this type.
   3. DQL (Data Query Language)
      1. Using this you can retrieve the records from the database. There can multiple options to retrieve the data such as clauses, Join etc.
      2. SQL Operation like **SELECT** can use perform in this type.
   4. TCL (Transaction Control Language)
      1. It is use to manage the transactions. Transactions are the set of queries which executes in a group and it will either complete the execution and finalized into database by commit or revert the changes if any one query fails.
      2. SQL Operations like **COMMIT, ROLLBACK and SAVEPOINT**
   5. DCL (Data Control Language)
      1. Using this you can decide which user use which functionality of the database.
      2. Here the Permissions will be granted or reverted for the database user.
      3. SQL Operations like **GRANT and REVOKE** are performed.

**Data Type**

1. String/Textual/binary data type
2. Numeric data type
3. Date and Time data type

<https://www.w3schools.com/mysql/mysql_datatypes.asp#:~:text=In%20MySQL%20there%20are%20three,numeric%2C%20and%20date%20and%20time>.



**DDL**:-

**Create Database**

1. Database is a working area where you can create a tables and records inside table.
2. Mostly for every project there will be a separate database.
3. Syntax:

CREATE <object-name> < name>;

CREATE DATABASE <db-name>;

1. Example:

CREATE DATABASE fsd23sept;

**To Enter into Database**

Syntax: USE <db-name>;

Example: USE fsd23sept;

**Create Table**

1. While creating table you have to provide the column name and its data types.
2. Syntax:

CREATE TABLE <table-name> (column-name DataType, \_ , \_ );

Example:

CREATE TABLE student (

id int,

name varchar(20),

email varchar(50),

age double

);

**Alter Table**

1. Using this you can make a modification into table structure.
2. Can perform ADD, MODIFY, DROP, RENAME operations.
3. ADD Operation
   1. Using this you can create new column into table.
   2. Syntax:

ALTER TABLE <table-name> **ADD COLUMN** <column-name> <DataType>

Example:

ALTER TABLE student ADD COLUMN dob date;

1. MODIFY Operation
   1. You can make a changes inside column data type or the size of the column etc.
   2. Syntax:

ALTER TABLE <table-name> **MODIFY COLUMN** <column-name> <datatype>

Example:

ALTER TABLE student MODIFY COLUMN city varchar(20);

1. RENAME Operation
   1. To rename a column name
   2. Syntax:

ALTER TABLE <table-name> **RENAME COLUMN** <old-column-name> TO <new-column-name>

Example:

ALTER TABLE student RENAME COLUMN city TO location;

1. DROP Operation
   1. To delete the column from the table
   2. Syntax:

ALTER TABLE <table-name> **DROP COLUMN** <column-name>

Example:

ALTER TABLE student DROP COLUMN location;

**Drop database object**

1. It is use to drop the database, table or any database object.
2. Syntax:

DROP <Object> <name>;

Example

DROP TABLE student;

**DML Queries**

1. DML queries are use to perform operation on the records from the table.
2. You can manipulate the data/records present inside table.
3. **INSERT into table:**

Syntax:

INSERT INTO <Table-name>(column, \_ , \_ ) VALUES(val, \_ , \_);

Example:

INSERT INTO student(id,name,email,age) VALUES(1, 'Abc', 'abc@gmail.com',22.1);

INSERT INTO student VALUES(2, 'Pqr', 'pqr@gmail.com',20.4);

INSERT INTO student VALUES(3, 'Lmn', 'lmn@gmail.com',21.6),(4, 'Xyz', 'xyz@gmail.com',16.2),(5, 'Stu', 'stu@gmail.com',11.6);

1. **UPDATE into table**
   1. Using UPDATE query you can update the all records from the table or you can update a specific records.
   2. To Update the specific records from the table you needs to apply clauses and conditions on the record.

Syntax:

UPDATE <table-name> SET column-name=value, column-name=value;

UPDATE <table-name> SET column-name=value, column-name=value

WHERE condition;

Example:

UPDATE student SET age=age+1;

UPDATE student SET email='xyz@yahoo.com', age='20.8' WHERE id=4;

1. **DELETE from table**
   1. You can delete the specific records from the table.
   2. DELETE will be used with a clause to delete a specific row.
   3. Syntax:

DELETE FROM <table-name> WHERE condition;

Example

DELETE FROM student WHERE id=2;

**DQL**

1. Use to get the data from the table.
2. Using SELECT operation you can perform the retrieval operation.
3. To Select the all rows and all column you can use following syntax

SELECT \* FROM <table-name>

**Clauses**

1. Clauses are used to target a specific record from the table.
2. Clauses can be used to perform SELECT, UPDATE, DELETE operation.
3. There are multiple clauses available, following are the mostly used clauses
   1. WHERE Clause
      1. In this clause you have to provide a condition so that specific rows will be targeted.
      2. Example:

UPDATE student SET email='xyz@yahoo.com', age='20.8' WHERE id=4;

DELETE FROM student WHERE id=2;

SELECT \* FROM student **WHERE** age>20;

* 1. AND/OR
     1. Using this clause, you can combine two or more conditions to target the records.
     2. It is same as && and || in java. In SQL you cannot use a symbol.
     3. Example:

SELECT \* FROM student WHERE age>20 **AND** id<4;

SELECT \* FROM student WHERE age>20 **OR** id<4;

* 1. IN
     1. Using this clause, you can select the records based on the set of values.
     2. There will be OR condition applies between the set of values.
     3. Example:

SELECT \* FROM student WHERE age **IN** (22.6, 20.8, 12.6);

* 1. BETWEEN
     1. Using this clause you can select records between an range.
     2. This can be apply in place of AND clause used for range check.
     3. Example:

select \* from student WHERE AGE BETWEEN 10 AND 21;

* 1. ORDER DY
     1. Using this clause you can arrange a records into specific order like Ascending or descending order.
     2. By default it orders by Ascending order to get it in descending order you have to explicitly mention.
     3. Example:

select \* from student ORDER BY name; // This is for Ascending order

select \* from student ORDER BY name DESC;

* 1. LIMIT
     1. Using limit clause you can apply the limit on the number of rows selected.
     2. Example:

select \* from student ORDER BY age ASC **LIMIT 2**;

* 1. LIKE
     1. It is use to get the records based on partial value.
     2. You can get the records by 3 ways

**Val%**: the records start with given value and followed with any character

**%Val**: the records ends with given value and can have any character in the start.

**%Val%**:the records start and ends with any thing and contains the given value in between.

* + 1. Example

select \* from student where email **LIKE '%gmail%'**;

select \* from student WHERE name **LIKE '%z'**;

select \* from student WHERE name **LIKE 'A%'**;

* 1. GROUP
     1. Using this clause you can create a group of records and apply an aggregate function on it like count, min, mac, avg, sum.
     2. Example:

SELECT age, count(id) AS total FROM student **GROUP BY age**;

* 1. Having 
     1. You can apply a condition on the group records.
     2. This clause has to use with a group by clause.
     3. Example:

SELECT age, count(id) AS total FROM student GROUP BY age **HAVING total>1**;

**Primary Key**

It a not null and unique value. Every record will have a unique value for this column.

This Primary Key can be use as a foreign key in to another table to make a relation between 2 or more tables.

**Foreign Key**

It is use to relate 2 or more table together so that you can link the details of 1st table with the another table. foreign key can be null and can be duplicate also.

**Entity Relational Diagram**



CREATE TABLE studinfo(

id int PRIMARY KEY AUTO\_INCREMENT,

name varchar(30),

contact varchar(10),

email varchar(40)

);

INSERT INTO studinfo VALUE(0, 'A', '9988009988','a@gmaillcom');

INSERT INTO studinfo VALUE(0, 'B', '7788009988','b@gmaillcom');

INSERT INTO studinfo VALUE(0, 'C', '9876009988','c@gmaillcom'),(0, 'D', '9876123456','d@gmaillcom'),(0, 'E', '9876543212','e@gmaillcom');

CREATE TABLE studadd(

aid int PRIMARY KEY AUTO\_INCREMENT,

city varchar(20),

pincode varchar(6),

state varchar(20),

sid int,

FOREIGN KEY(sid) REFERENCES studinfo(id)

);

INSERT INTO studadd VALUE(0, 'Pune', '998800','MH', 1);

INSERT INTO studadd VALUE(0, 'Mumbai', '551100','MH', 4);

INSERT INTO studadd VALUE(0, 'Pune', '123212','MH', 5);

INSERT INTO studadd(aid,city,pincode,state) VALUE(0, 'Mumbai', '878776','MH');

INSERT INTO studadd(aid,city,pincode,state) VALUE(0, 'Nagpur', '988767','MH');

CREATE TABLE studedu(

eid int PRIMARY KEY AUTO\_INCREMENT,

title varchar(30),

percent double,

passingyear varchar(4),

sid int,

FOREIGN KEY(sid) REFERENCES studinfo(id)

);

INSERT INTO studedu VALUE(0, 'BE', 67.56,'2020', 1);

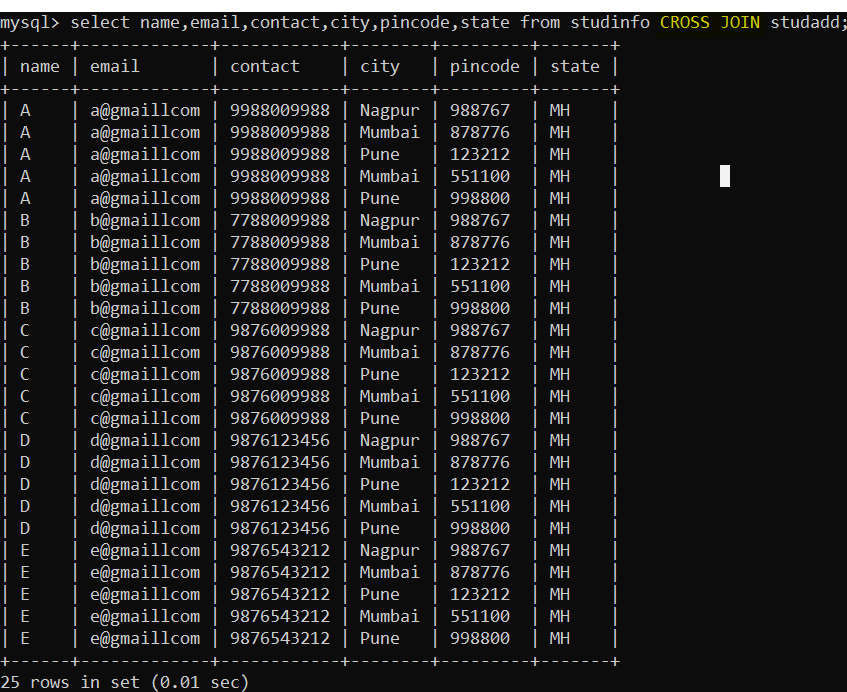
INSERT INTO studedu VALUE(0, 'ME', 71.51,'2019', 2);

INSERT INTO studedu VALUE(0, 'BCA', 78.12,'2020',4);

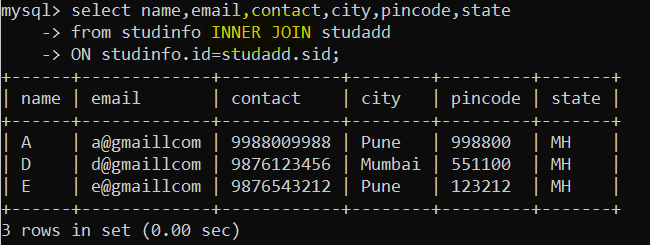
INSERT INTO studedu(eid,title,percent,passingyear) VALUE(0, 'MBA', 76.56,'2012');

**Join**

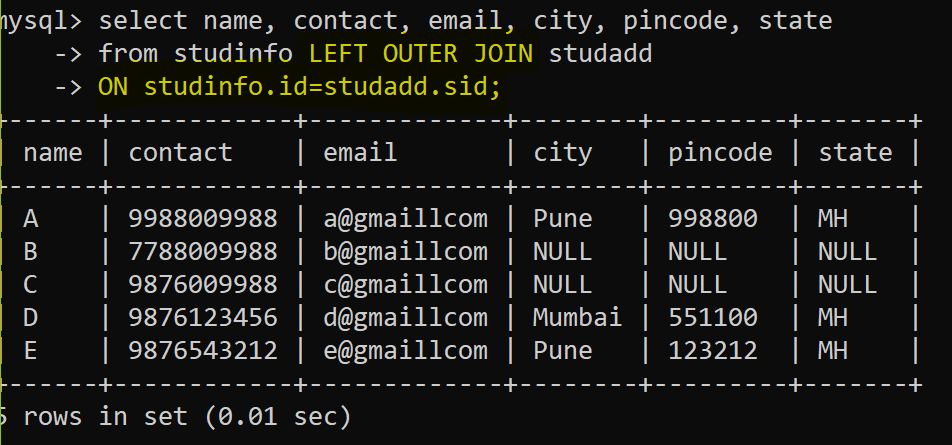
1. Join is use to work with a multiple table.
2. Using Join you can retrieve the records from the multiple table.
3. Types of Join
   1. Cross/Cartesian join
      1. Cross Join is use to get the data from multiple table.
      2. But in this case the row from one table will be mapped all the rows from another table. And hence the records will be generated multiple times.
      3. In Thus join no clause will be applied.



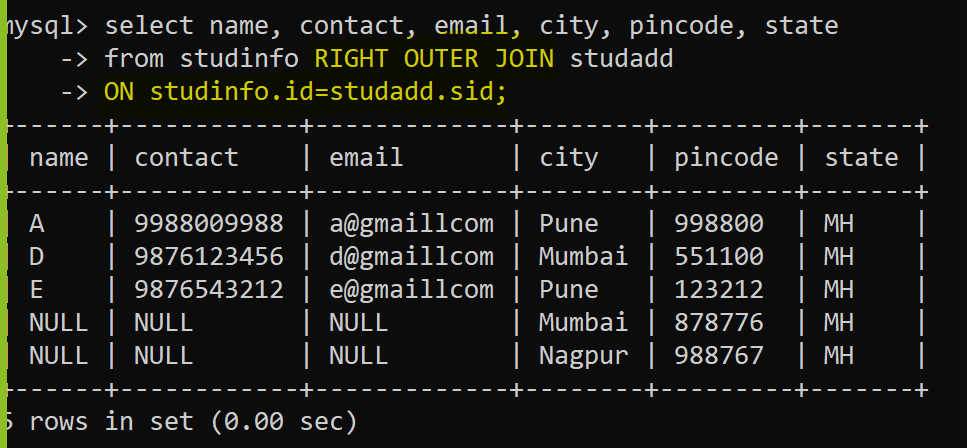
* 1. Inner Join
     1. Using Inner join you get only the matching records from the multiple tables.
     2. In this join the primary key column will be compare with the foreign key of another table.



* 1. Outer Join
     1. Left Outer Join
        1. It will select all the records from left table and only matching records from the another table



* + 1. Right Outer Join
       1. It will select all the records from right table and only matching records from the another table



* + 1. Full Join (is not supported in mysql but can be use in Oracle, Postgres etc.)



**Joining More then two table**

