Ex no: 01. **DDL and DML commands**

AIM:

To implement basic DDL and DML commands in sql.

DESCRIPTION:

SQL:

SQL is a database computer language designed for the retrieval and management of data in a relational database. SQL stands for Structured Query Language.
SQL is Structured Query Language, which is a computer language for storing, manipulating and retrieving data stored in a relational database.SQL is the standard
language for Relational Database System. All the Relational Database Management Systems (RDMS) like MySQL, MS Access, Oracle, Sybase, Informix, Postgres and SQL
Server use SQL as their standard database language.

DDL:

DDL stands for "Data Definition Language. A DDL is a language used to define data structures and modify data. For example, DDL commands can be used to
add, remove, or modify tables within in a database. DDLs used in database applications are considered a subset of SQL, the Structured Query Language.

Examples of DDL commands:

CREATE – is used to create the database or its objects (like table, index, function, views, store procedure and triggers).
DROP – is used to delete objects from the database.
ALTER-is used to alter the structure of the database.
TRUNCATE–is used to remove all records from a table, including all spaces allocated for the records are removed.
COMMENT –is used to add comments to the data dictionary.
RENAME –is used to rename an object existing in the database.

DQL (Data Query Language) :

DML statements are used for performing queries on the data within schema objects. The purpose of DQL Command is to get some schema relation based on the
query passed to it.

Example of DQL:

SELECT – is used to retrieve data from the a database.

DML(Data Manipulation Language) :

The SQL commands that deals with the manipulation of data present in the database belong to DML or Data Manipulation Language and this includes most of
the SQL statements.

Examples of DML:

INSERT – is used to insert data into a table.
UPDATE – is used to update existing data within a table.
DELETE – is used to delete records from a database table.

DCL(Data Control Language) :

DCL includes commands such as GRANT and REVOKE which mainly deals with the rights, permissions and other controls of the database system.

Examples of DCL commands:

GRANT-gives user’s access privileges to database.
REVOKE-withdraw user’s access privileges given by using the GRANT command.

TCL(transaction Control Language) :

TCL commands deals with the transaction within the database.

Examples of TCL commands:

COMMIT- commits a Transaction.
ROLLBACK- rollbacks a transaction in case of any error occurs.
SAVEPOINT-sets a savepoint within a transaction.
SET TRANSACTION-specify characteristics for the transaction.

EXERCISE:

Question: 1

Create a student database with table "student_details" and include the attributes of Reg_no, Name, Dept with appropriate constraints.

Query:

```
      create database student;
use student;
create table student_details(
Reg_no int(10),
Name varchar(25),
Dept varchar(10),
primary key(Reg_no)
);

desc student_details;
```

Answer:

```
+--------+-------------+------+-----+---------+-------+
| Field  | Type        | Null | Key | Default | Extra |
+--------+-------------+------+-----+---------+-------+
| Reg_no | int(10)     | YES  | PRI | NULL    |       |
| Name   | varchar(25) | YES  |     | NULL    |       |
| Dept   | varchar(10) | YES  |     | NULL    |       |
      +--------+-------------+------+-----+---------+-------+
```

Question: 2

create duplicate of student_details table named student_detail_new.

Query:

```
      create table student_detail_new like student_details;

      desc student_detail_new;
```

Answer:

```
      +--------+-------------+------+-----+---------+-------+
      | Field  | Type        | Null | Key | Default | Extra |
      +--------+-------------+------+-----+---------+-------+
      | Reg_no | int(10)     | YES  | PRI | NULL    |       |
      | Name   | varchar(25) | YES  |     | NULL    |       |
      | Dept   | varchar(10) | YES  |     | NULL    |       |
      +--------+-------------+------+-----+---------+-------+
```

Question: 3

Add the following attributes with appropriate attribute.
m1_dbms (mark1_database management system),
m2_os   (mark2_Operating System),
m3_cd   (mark3_Circuit Design),
Total.

Query:

```
      alter table student_details add(
      m1_dbms int,
      m2_os int,
      m3_cd int,
      total int);

      desc student_details;
```

Answer:

```
      +---------+-------------+------+-----+---------+-------+
      | Field   | Type        | Null | Key | Default | Extra |
      +---------+-------------+------+-----+---------+-------+
      | Reg_no  | int(10)     | YES  | PRI | NULL    |       |
      | name    | varchar(25) | YES  |     | NULL    |       |
      | dept    | varchar(10) | YES  |     | NULL    |       |
      | m1_dbms | int(11)     | YES  |     | NULL    |       |
      | m2_os   | int(11)     | YES  |     | NULL    |       |
      | m3_cd   | int(11)     | YES  |     | NULL    |       |
      | total   | int(11)     | YES  |     | NULL    |       |
      +---------+-------------+------+-----+---------+-------+
```

Question: 4

Insert atleast 10 records in the table with Reg_no,Name,Dept,m1_dbms,m2_os,m3_cd.

Query:

```
      insert into student_details (Reg_no,Name,Dept,m1_dbms,m2_os,m3_cd)
values ('1','Vijay','CSE',45,85,70);

      insert into student_details (Reg_no,Name,Dept,m1_dbms,m2_os,m3_cd)
values ('2','Fahi','ECE',55,75,60);

      insert into student_details (Reg_no,Name,Dept,m1_dbms,m2_os,m3_cd)
values ('3','Edwin','EEE',95,55,70);

      insert into student_details (Reg_no,Name,Dept,m1_dbms,m2_os,m3_cd)
values ('4','Siva','EIE',65,25,30);

      insert into student_details (Reg_no,Name,Dept,m1_dbms,m2_os,m3_cd)
values ('5','Sasi','IBT',75,75,40);

      insert into student_details (Reg_no,Name,Dept,m1_dbms,m2_os,m3_cd)
values ('6','Sanjay','IBT',60,80,45);

      insert into student_details (Reg_no,Name,Dept,m1_dbms,m2_os,m3_cd)
values ('7','Joseph','IT',40,60,65);

      insert into student_details (Reg_no,Name,Dept,m1_dbms,m2_os,m3_cd)
values ('8','Teejay','IT',95,55,70);

      insert into student_details (Reg_no,Name,Dept,m1_dbms,m2_os,m3_cd)
values ('9','Ajay','MECH',65,35,80);

      insert into student_details (Reg_no,Name,Dept,m1_dbms,m2_os,m3_cd)
values ('10','Karthi','PROD',85,75,70);

      select * from student_details;
```

Answer:

```
      +--------+---------+------+---------+-------+-------+--------+
      | Reg_no | name    | dept | m1_dbms | m2_os | m3_cd |  total |
      +--------+---------+------+---------+-------+-------+--------+
      |      1 | vijay   | CSE  |   45    |   85  |   70  |   NULL |
      |      2 | fahi    | ECE  |   55    |   75  |   60  |   NULL |
      |      3 | edwin   | EEE  |   95    |   55  |   70  |   NULL |
      |      4 | siva    | EIE  |   65    |   25  |   30  |   NULL |
      |      5 | sasi    | IBT  |   75    |   75  |   40  |   NULL |
      |      6 | sanjay  | IBT  |   60    |   80  |   45  |   NULL |
      |      7 | joseph  | IT   |   40    |   60  |   85  |   NULL |
      |      8 | teejay  | IT   |   95    |   55  |   70  |   NULL |
      |      9 | ajay    | MECH |   65    |   35  |   80  |   NULL |
      |     10 | karthi  | PROD |   85    |   75  |   70  |   NULL |
      +--------+---------+------+---------+-------+-------+--------+
```

Question: 5

Update the column table with the sum of mark1,mark2 and mark3.

Query:

```
update student_details set total = (m1_dbms+m2_os+m3_cd);

select * from student_details;
```

Answer:

```
+--------+---------+------+---------+-------+-------+--------+
| Reg_no | name    | dept | m1_dbms | m2_os | m3_cd |  total |
+--------+---------+------+---------+-------+-------+--------+
|     1  | vijay   | CSE  |   45    |   85  |   70  |   200  |
|     2  | fahi    | ECE  |   55    |   75  |   60  |   190  |
|     3  | edwin   | EEE  |   95    |   55  |   70  |   220  |
|     4  | siva    | EIE  |   65    |   25  |   30  |   120  |
|     5  | sasi    | IBT  |   75    |   75  |   40  |   190  |
|     6  | sanjay  | IBT  |   60    |   80  |   45  |   185  |
|     7  | joseph  | IT   |   40    |   60  |   85  |   185  |
|     8  | teejay  | IT   |   95    |   55  |   70  |   220  |
|     9  | ajay    | MECH |   65    |   35  |   80  |   180  |
|    10  | karthi  | PROD |   85    |   75  |   70  |   230  |
+--------+---------+------+---------+-------+-------+--------+
```

Question: 6

Retrive the records for the following:

a) where the total is less than 150

Query:

```
Select * from student_details where total<150;
```

Answer:

```
+--------+---------+------+---------+-------+-------+---
----+
| Reg_no | name    | dept | m1_dbms | m2_os | m3_cd |
total |
+--------+---------+------+---------+-------+-------+---
----+
|     4  | siva    | EIE  |   65    |   25  |   30  |
120 |
+--------+---------+------+---------+-------+-------+---
----+
```

b) who scored < 50 in dbms

Query:

```
select * from student_details where m1_dbms < 50;
```

Answer:

```
      +--------+---------+------+---------+-------+-------+---
----+
      | Reg_no | name    | dept | m1_dbms | m2_os | m3_cd |
total |
      +--------+---------+------+---------+-------+-------+---
----+
      |     1  | vijay   | CSE  |   45    |   85  |   70  |
200 |
      |     7  | joseph  | IT   |   40    |   60  |   85  |
185 |
      +--------+---------+------+---------+-------+-------+---
----+
```

c) who scored <50 in OS and CD

Query:

```
      Select * from student_details where m1_os < 50 and m2_cd
<50;
```

Answer:

```
      +--------+---------+------+---------+-------+-------+---
----+
      | Reg_no | name    | dept | m1_dbms | m2_os | m3_cd |
total |
      +--------+---------+------+---------+-------+-------+---
----+
      |     4  | siva    | EIE  |   65    |   25  |   30  |
120 |
      +--------+---------+------+---------+-------+-------+---
----+
```

d) who got less than 50 in any of the subjects

Query:

```
      Select * from student_details where m1_dbms <50 or m2_os
<50 or m3_cd < 50;
```

Answer:

```
      +--------+---------+------+---------+-------+-------+---
----+
      | Reg_no | name    | dept | m1_dbms | m2_os | m3_cd |
total |
      +--------+---------+------+---------+-------+-------+---
----+
      |     1  | vijay   | CSE  |   45    |   85  |   70  |
200 |
                |     4  | siva    | EIE  |   65    |   25  |   30  |
120 |
                |     5  | sasi    | IBT  |   75    |   75  |   40  |
190 |
                |     6  | sanjay  | IBT  |   60    |   80  |   45  |
185 |
                |     7  | joseph  | IT   |   40    |   60  |   85  |
185 |
                |     9  | ajay    | MECH |   65    |   35  |   80  |
180 |
                +------- +---------+------+---------+-------+-------+---
----+
```

Question: 7
Insert three records by single insert statement.

Query:

```
     insert into student_details (Reg_no,Name,Dept,m1_dbms,m2_os,m3_cd)
values
     (11,'mahesh','civil',70,70,65,0),
     (12,'ananda','CSE',70,80,65,0),
     (13,'lalith','CSE',80,75,90,0);

     update student_details set total = (m1_dbms+m2_os+m3_cd);

     select * from student_details;
```

Answer:

```
+--------+---------+------+---------+-------+-------+-------+
| Reg_no | name    | dept | m1_dbms | m2_os | m3_cd | total |
+--------+---------+------+---------+-------+-------+-------+
|     1  | vijay   | CSE  |   45    |   85  |   70  |   200 |
|     2  | fahi    | ECE  |   55    |   75  |   60  |   190 |
|     3  | edwin   | EEE  |   95    |   55  |   70  |   220 |
|     4  | siva    | EIE  |   65    |   25  |   30  |   120 |
|     5  | sasi    | IBT  |   75    |   75  |   40  |   190 |
|     6  | sanjay  | IBT  |   60    |   80  |   45  |   185 |
|     7  | joseph  | IT   |   40    |   60  |   85  |   185 |
|     8  | teejay  | IT   |   95    |   55  |   70  |   220 |
|     9  | ajay    | MECH |   65    |   35  |   80  |   180 |
|    10  | karthi  | PROD |   85    |   75  |   70  |   230 |
|    11  | mahesh  | civil|   70    |   70  |   65  |   205 |
|    12  | ananda  | CSE  |   70    |   80  |   65  |   215 |
|    13  | lalith  | CSE  |   80    |   75  |   90  |   245 |
+--------+---------+------+---------+-------+-------+-------+
```

Question: 8
Insert a record in student_details table to ensure no duplicate value should be register in Reg_no.

Query:

```
      INSERT into student_details values(11,'anand','IT',45,100,90,0);
```

Answer:

```
      ERROR 1062 (23000): Duplicate entry '11' for key 'PRIMARY'
```

Question: 9
Retrive records for who get same mark of any two subjects.

Query:
```
      select * from student_details where m1_dbms==m2_os or m2_os==m3_cd
or m3_cd==m1_dbms;
```

Answer:

```
      +--------+---------+------+---------+-------+-------+-------+
      | Reg_no | name    | dept | m1_dbms | m2_os | m3_cd | total |
      +--------+---------+------+---------+-------+-------+-------+
      |     5  | sasi    | IBT  |   75    |   75  |   40  |   190 |
      |    11  | mahesh  | civil|   70    |   70  |   65  |   205 |
      +--------+---------+------+---------+-------+-------+-------+
```

Question: 10
Retrive records who got >90 in any subject.

Query:
```
      select * from student_details where m1_dbms>90 or m2_os >90 or
m3_cd>90;
```

Answer:

```
      +--------+---------+------+---------+-------+-------+-------+
      | Reg_no | name    | dept | m1_dbms | m2_os | m3_cd | total |
      +--------+---------+------+---------+-------+-------+-------+
      |     3  | edwin   | EEE  |   95    |   55  |   70  |   220 |
      |     8  | teejay  | IT   |   95    |   55  |   70  |   220 |
      +--------+---------+------+---------+-------+-------+-------+
```

Question: 11
Retrive all student name starts with 'f'.

Query:
```
      Select * from student_details where name like 'f%';
```

Answer:

```
      +--------+---------+------+---------+-------+-------+-------+
      | Reg_no | name    | dept | m1_dbms | m2_os | m3_cd | total |
      +--------+---------+------+---------+-------+-------+-------+
      |     2  | fahi    | ECE  |   55    |   75  |   60  |   190 |
      +--------+---------+------+---------+-------+-------+-------+
```

Question: 12

Retrive records with unique from student_details table.

Query:

```
      Select name, dept, count(dept) from student_details group by having
count(dept) = 1;
```

Answer:

```
+--------+------+-------------+
| Name   | dept | count(dept) |
+--------+------+-------------+
| mahesh | civi |           1 |
| fahi   | ECE  |           1 |
| edwin  | EEE  |           1 |
| siva   | EIE  |           1 |
| ajay   | MECH |           1 |
| karthi | PROD |           1 |
+--------+------+-------------+
```

Question: 13

Return all student records that ends with 'ay'.

Query:

```
      select * from student_details where name like '%ay';
```

Answer:

```
+--------+---------+------+---------+-------+-------+-------+
| Reg_no | name    | dept | m1_dbms | m2_os | m3_cd | total |
+--------+---------+------+---------+-------+-------+-------+
|     1  | vijay   | CSE  |   45    |   85  |   70  |   200 |
|     6  | sanjay  | IBT  |   60    |   80  |   45  |   185 |
|     8  | teejay  | IT   |   95    |   55  |   70  |   220 |
|     9  | ajay    | MECH |   65    |   35  |   80  |   180 |
+--------+---------+------+---------+-------+-------+-------+
```

Question: 14

Return all student records with name that has the patter 'ja' in it.

Query:

```
      select * from student_details where name like '%ja%';
```

Answer:

```
+--------+---------+------+---------+-------+-------+-------+
| Reg_no | name    | dept | m1_dbms | m2_os | m3_cd | total |
+--------+---------+------+---------+-------+-------+-------+
|     1  | vijay   | CSE  |   45    |   85  |   70  |   200 |
|     6  | sanjay  | IBT  |   60    |   80  |   45  |   185 |
|     8  | teejay  | IT   |   95    |   55  |   70  |   220 |
|     9  | ajay    | MECH |   65    |   35  |   80  |   180 |
+--------+---------+------+---------+-------+-------+-------+
```

Question: 15

Retrive student name that start and end with vowels.

Query:

```
      select * from student_details where name rlike
'^[aeiouAEIOU].*[aeiouAEIOU]$';
```

Answer:

```
      +--------+---------+------+---------+-------+-------+-------+
      | Reg_no | name    | dept | m1_dbms | m2_os | m3_cd | total |
      +--------+---------+------+---------+-------+-------+-------+
      |    12  | ananda  | CSE  |   70    |   80  |   65  |   215 |
      +--------+---------+------+---------+-------+-------+-------+
```

Question: 16

Retrive records Name that 'a' as their third character.

Query:

```
      select * from student_details where name like '__a%';
```

Answer:

```
      +--------+---------+------+---------+-------+-------+-------+
      | Reg_no | name    | dept | m1_dbms | m2_os | m3_cd | total |
      +--------+---------+------+---------+-------+-------+-------+
      |     9  | ajay    | MECH |   65    |   35  |   80  |   180 |
      |    12  | ananda  | CSE  |   70    |   80  |   65  |   215 |
      +--------+---------+------+---------+-------+-------+-------+
```

Quesion: 17

Retrive records whose total is between 150 and 200.

Query:

```
      select * from student_details where total>150 and total<200;
```

Answer:

```
      +--------+---------+------+---------+-------+-------+-------+
      | Reg_no | name    | dept | m1_dbms | m2_os | m3_cd | total |
      +--------+---------+------+---------+-------+-------+-------+
      |     2  | fahi    | ECE  |   55    |   75  |   60  |   190 |
      |     5  | sasi    | IBT  |   75    |   75  |   40  |   190 |
      |     6  | sanjay  | IBT  |   60    |   80  |   45  |   185 |
      |     7  | joseph  | IT   |   40    |   60  |   85  |   185 |
      |     9  | ajay    | MECH |   65    |   35  |   80  |   180 |
      +--------+---------+------+---------+-------+-------+-------+
```

Question: 18

Display student name in ascending order and in upper case.

Query:

```
      select upper(name) from student_details order by name;
```

Answer:

```
+--------+
| Name   |
+--------+
| AJAY   |
| ANANDA |
| EDWIN  |
| FAHI   |
| JOSEPH |
| KARTHI |
| LALITH |
| MAHESH |
| SANJAY |
| SASI   |
| SIVA   |
| TEEJAY |
| VIJAY  |
+--------+
```

Question: 19
Add a column age to the table as a first column and insert default value for age.

Query:
```
alter table student_details add Age int(5) default 18 first;

desc student_details;
```

Answer:

```
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| Age        | int(5)      | YES  |     | 18      |       |
| Reg_no     | int(10)     | NO   | PRI | NULL    |       |
| Name       | varchar(20) | YES  |     | NULL    |       |
| dept       | varchar(10) | YES  |     | NULL    |       |
| m1_dbms    | int(11)     | YES  |     | NULL    |       |
| m2_os      | int(11)     | YES  |     | NULL    |       |
| m3_cd      | int(11)     | YES  |     | NULL    |       |
| Total      | int(11)     | YES  |     | NULL    |       |
+------------+-------------+------+-----+---------+-------+
```

Question: 20
Update the record with name to 'Anandan' whose register number is '4'.

Query:
```
update student_details set Name='Anandan' where Reg_no = 4;

select * from student_details where Reg_no=4;
```
Answer:

```
      +------+--------+---------+-----+---------+-------+------+-------
+
      | age  | Reg_no | name    | dept | m1_dbms | m2_os | m3_cd | total
|
      +------+--------+---------+-----+---------+-------+------+-------
+
      |   18 |     4  | siva    | EIE  |   65    |   25  |   30  |   120
|
      +------+--------+---------+-----+---------+-------+------+-------
+
```

Question: 21
      Start a transaction and create a savepoint.

Query:
```
      start transaction;
      savepoint A;
```

Question: 22
      Delete a record of register number 8.
Query:
```
      delete from student_details where Reg_no=8;

      select * from student_details where Reg_no=8;
```

Answer:
```
      Empty set.
```

Question: 23
      Retrieve a database to last committed statement.

Query:
```
      rollbackto A;

      select * from student_details where Reg_no = 8;
```

Answer:

```
      +------+--------+---------+-----+---------+-------+------+-------
+
      | age  | Reg_no | name    | dept | m1_dbms | m2_os | m3_cd | total
|
      +------+--------+---------+-----+---------+-------+------+-------
+
      |   18 |     8  | teejay  | IT   |   95    |   55  |   70  |   220
|
      +------+--------+---------+-----+---------+-------+------+-------
+
```

Question: 24
      Save the transacion.

Query:

```
      commit;
```

Question: 25
Update the age column of student table with not available for all student.

Query:

```
      update student_details set age=18;

      select distinct age from student_details;
```

Answer:

```
      +------+
      | age  |
      +------+
      |   18 |
      +------+
```

Question: 26
Display the records sorted by total marks.

Query:

```
      select * from student_details order by total;
```

Answer:

```
      +-------+-------+
      | Reg_no| total |
      +-------+-------+
      |     4 |   120 |
      |     9 |   180 |
      |     7 |   185 |
      |     6 |   185 |
      |     2 |   190 |
      |     5 |   190 |
      |     1 |   200 |
      |    11 |   205 |
      |    12 |   215 |
      |     8 |   220 |
      |     3 |   220 |
      |    10 |   230 |
      |    13 |   245 |
      +-------+-------+
```

Question: 27
Retrive records from 'CSE' department.

Query:

```
      select * from student_details where dept='CSE';
```

Answer:

```
      +------+--------+---------+------+---------+-------+-------+-------
+
      | age  | Reg_no | name    | dept | m1_dbms | m2_os | m3_cd | total
|
      +------+--------+---------+------+---------+-------+-------+-------
+
      |   18 |     1  | VIJAY   | CSE  |   45    |   85  |   70  |   200
|
      |   18 |    12  | KABILAN | CSE  |   70    |   80  |   65  |   215
|
      |   18 |    13  | LALITH  | CSE  |   80    |   75  |   90  |   245
|
      +------+--------+---------+------+---------+-------+-------+-------
+
```

Question: 28
Rename the table name to 'std'.

Query:
```
      alter table student_details rename to std;

      show tables;
```

Answer:
```
      +----------------------+
      | Tables_in_student    |
      +----------------------+
      | student_details_new  |
      | std                  |
      +----------------------+
```

RESULT:
Thus, basic commands in DDL and DML are executed and output is verified.

Exp. No : 2 **Form Design and report generation using PHP/Java/Django**

Aim:

To design a Form using PHP and make queries with the database.

Description:

Forms:

An HTML form is used to collect user input. The user input can then be sent to a server for processing.

PHP:

PHP: Hypertext Preprocessor. PHP is a server side scripting language that is used to develop Static websites or Dynamic websites or Web applications.The client computers accessing the PHP scripts require a web browser. PHP is open source and cross platform.

Database Connectivity:

MySQLi is an API that uses a connector function to link the backend of the PHP app to the MySQL database. It provides a better set of functions and extensions.

PHP Data Objects (PDO) extension is a Database Abstraction Layer. It is like an interface for the backend to interact with the MySQL database and make changes without making any change in the PHP code.

Form Code:

```
<?php


$username = "";
```

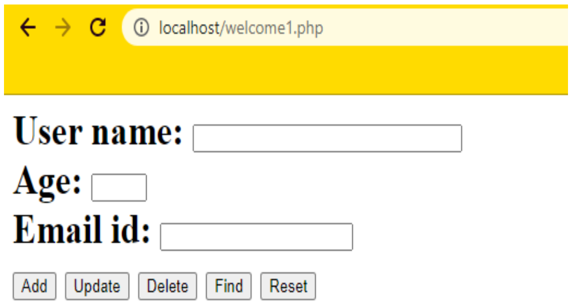
```
$age = "";
$email = "";

$host = "localhost";
$dbUsername = "root";
$dbPassword = "";
$dbname = "form";

mysqli_report(MYSQLI_REPORT_ERROR | MYSQLI_REPORT_STRICT);

try{
	$conn = new mysqli($host, $dbUsername, $dbPassword, $dbname);
}catch (Exception $ex){
	echo 'ERROR';
}

function getPosts()
{
	$post = array();
	$post[0]= $_POST['username'];
	$post[1]= $_POST['age'];
	$post[2]= $_POST['email'];
	return $post;
}

if(isset($_POST['search']))
{
	$data = getPosts();
	$searchq = "SELECT * FROM register WHERE name = '$data[0]'";
	$searchr = mysqli_query($conn, $searchq);
```

```
        if($searchr)
        {
                if(mysqli_num_rows($searchr))
                {
                        while($row = mysqli_fetch_array($searchr))
                        {
                                $username = $row['name'];
                                $age = $row['Age'];
                                $email = $row['email'];
                        }
                }
                else{echo 'not found for username';}
        }
        else{
                echo 'result Error';
        }
}

//insert
if(isset($_POST['insert']))
{
        $data= getPosts();
        $insertq = "INSERT INTO `register` (`name`, `Age`, `email`) VALUES ('$data[0]', '$data[1]',
'$data[2]')";
        try{
                $insertr = mysqli_query($conn, $insertq);

                if($insertr)
                {
                        if(mysqli_affected_rows($conn)>0)
```

```
                    {
                        echo 'Data inserted';
                    }
                    else{
                        echo 'Data not inserted';
                    }
                }
            }catch(Exception $ex){
                echo 'Error Insert '.$ex->getMessage();
            }
}

//delete

if(isset($_POST['delete']))
{
  $data = getPosts();
  $delete_Query = "DELETE FROM `register` WHERE `register`.`name` = '$data[0]';";
  try{
    $delete_Result = mysqli_query($conn, $delete_Query);

    if($delete_Result)
    {
      if(mysqli_affected_rows($conn) > 0)
      {
        echo 'Data Deleted';
      }else{
        echo 'Data Not Deleted';
      }
    }
    } catch (Exception $ex) {
        echo 'Error Delete '.$ex->getMessage();
    }
}

//update

if(isset($_POST['update']))
{
    $data = getPosts();
    $update_Query = "UPDATE `register` SET `Age`='$data[1]',`name` = '$data[0]' WHERE
`email`='$data[2]'";
    try{
        $update_Result = mysqli_query($conn, $update_Query);

        if($update_Result)
        {
            if(mysqli_affected_rows($conn) > 0)
            {
                echo 'Data Updated';
            }else{
                echo 'Data Not Updated';
            }
        }
    } catch (Exception $ex) {
        echo 'Error Update '.$ex->getMessage();
    }
}

?>
```

```
<html>
  <head>
    <title>REGISTER</title>
  </head>
  <body>
     <h1>
        <form name="registration" action="welcome1.php" method="POST">
        User name:
        <input type="text" name="username" size="30" value="<?php echo $username; ?>">
        <br/>
                    Age:
                    <input type="text" name="age" size="2" value="<?php echo $age; ?>">
                    <br/>
        Email id:
        <input type="email" id="email" name="email" value="<?php echo $email; ?>">
                    <div>
                      <input type="submit" name="insert" value="Add">
                      <input type="submit" name="update" value="Update">
                      <input type="submit" name="delete" value="Delete">
                      <input type="submit" name="search" value="Find">
                      <input type="reset" name="reset">
                    </div>

  </body>
</html>
```

Output:

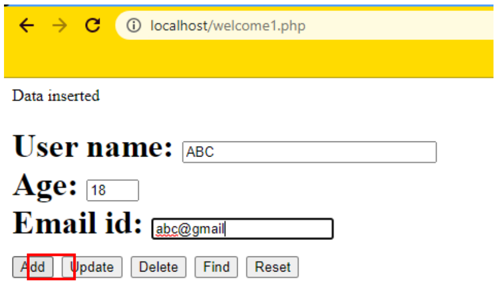

localhost/welcome1.php

User name:

Age:

Email id:

Add | Update | Delete | Find | Reset


localhost/welcome1.php

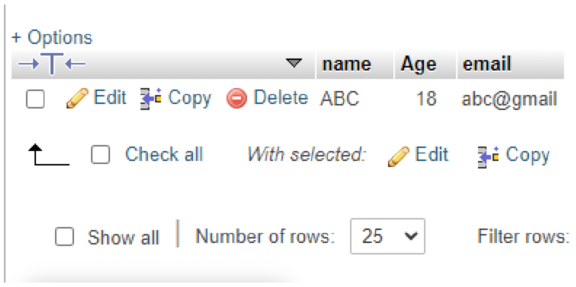
Data inserted

User name: ABC

Age: 18

Email id: abc@gmail

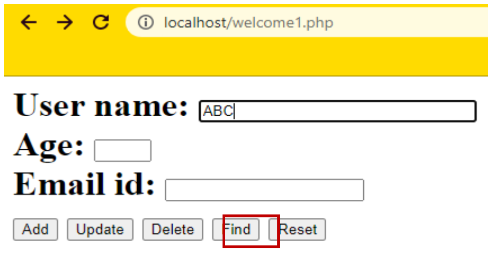
Add | Update | Delete | Find | Reset


+ Options

	name	Age	email
Edit Copy Delete	ABC	18	abc@gmail

Check all *With selected:* Edit Copy

Show all | Number of rows: 25 Filter rows:

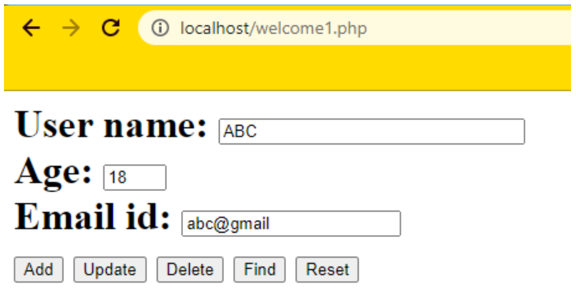

localhost/welcome1.php

User name: ABC

Age:

Email id:

Add | Update | Delete | Find | Reset

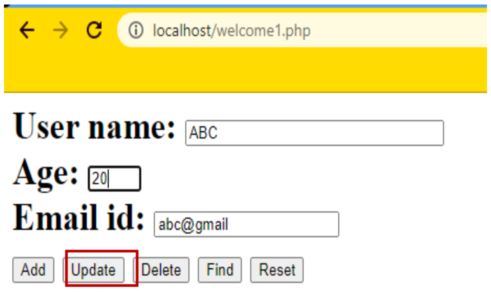

localhost/welcome1.php

User name: ABC

Age: 18

Email id: abc@gmail

Add | Update | Delete | Find | Reset

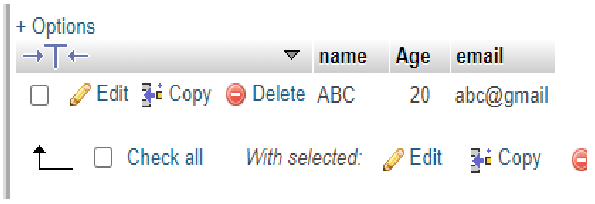

localhost/welcome1.php

User name: ABC

Age: 20

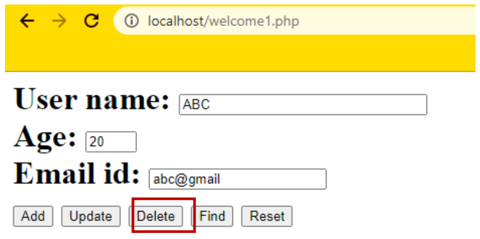
Email id: abc@gmail

Add | Update | Delete | Find | Reset


+ Options

	name	Age	email
Edit Copy Delete	ABC	20	abc@gmail

Check all *With selected:* Edit Copy

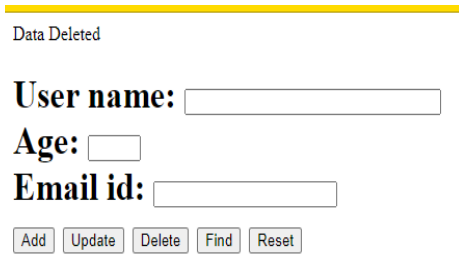

localhost/welcome1.php

User name: ABC

Age: 20

Email id: abc@gmail

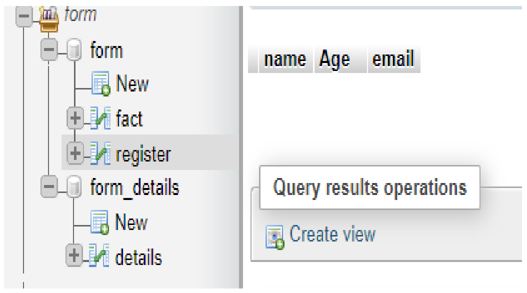
Add | Update | Delete | Find | Reset


Data Deleted

User name:

Age:

Email id:



Result:

Form was designed using HTML and PHP, connection with database was established using MySQL server and Queries were executed with the database.

Ex.no:03

3.VIEWS & SUBQURIES

AIM:

To implement views and subqueries .

MySQL view:

In Mysql view,a virutual table is created by a query by joining one more table

MySQL create view:

A view is created by select commands. Select statements are used to take data from the source table to make view.

CREATE VIEW Syntax

```
CREATE VIEW view_name AS
SELECT column1, column2, ...
FROM table_name
WHERE condition;
```

Example:

```
CREATE VIEW [  Indian Customers] AS
SELECT CustomerName, ContactName
FROM Customers
WHERE Country = ' Indian ';
```

MySQL Updating a View

A view can be updated with the ALTER VIEW command.

Updating View syntax:

```
  ALTER VIEW view_name AS
  SELECT column1, column2, ...
 FROM table_name
  WHERE condition;
```

Example

```
  ALTER VIEW [ Customers 1] AS
  SELECT CustomerName, ContactName, City
 FROM Customers
 WHERE Country = 'India';
```

MySQL drop view :

Create the drop view using drop view statement.

MySQL drop view syntax:

```
DROP VIEW[is exists] view name ;
```

Example

```
DROP view trainers;
```

Subqueries

A subqueries is a select query that is contained inside query. The result of the order select query.

Subqueries syntax

```
SELECT column_name [, column_name ]
FROM   table1 [, table2 ]
```

Types of subqueries

1.Scalar subquery

2.Row subquery

3.Table subquery

Scalar subquery

A scalar subquery returns a single row single column

Row subquery

They can have more than one column but return only one row.

Table subquery

Table queries can return multiple rows as well as columns.

create a table employee with following fields Id, Name, Date of joining, Designation, Department, Salary. And create another table emp_project with following fields Id, Project_name, no of hours worked.

```
create table employee(id int(10),name varchar(20),do
b date, doj date, designation varchar(11),dept
varchar(11),salary int(20))
        describe employee
+------------    + ---------------- + ----- + ----- + --------- + -------
+
| Field          | Type             | Null | Key | Default | Extra
|
+------------     + ---------------- + ------ + ----- + --------- + ------
+
| id             | int(10)          | YES  |     | NULL    |
|
| name           | varchar(20)      | YES  |     | NULL    |
|
| dob            | date             | YES  |     | NULL    |
|
| doj            | date             | YES  |     | NULL    |
|
| desig          | varchar(11)      | YES  |     | NULL    |        |
| dept           | varchar(11)      | YES  |     | NULL    |        |
| salary         | int(20)          | YES  |     | NULL    |        |
+ --------------- +---------------- + ----- + ---- + --------- + -------
+
```

create table empl_project(id int(11),project_name varchar(10),working_hours int(10))

describe empl_project

```
 + ----------------- + -------------- + ----- + ----- + ---------- +-------+
| Field              | Type           | Null | Key | Default | Extra |
+ ------------------ + -------------- + ----- + ----- + ---------- +-------+
|p_ id               | int(11)        | YES  |      | NULL    |        |
| project_name  | varchar(10) | YES  |      | NULL    |        |
| working_hours | int(10)        | YES  |      | NULL    |        |
+ ------------------- + -------------- + ----- + ----- + -------- + ------- +
```

insert into employee values('101','sindhu','2001/04/22', '2019/02/5','hr','ece','70000')

insert into employee values('102','laxmi','2000/03/24',' 2019/04/7','pr','ece','60000')

insert into employee values('103','arthi','2001/01/1','2 019/02/5','rep','cse','50000')

insert into employee values('104','jothika','2000/07/8',

```
'2015/02/9','vo','it','40000')
insert into  employee values('105','divya','2000/05/7','2
017/02/9','js','it','55000')
   select *from employee


 + ------ + --------- + ----------------+---------------+---------+--------+---------+
|  id     | name    | dob            | doj            | desig  | dept | salary |
+ ------ + --------- + --------------- + ---------------- + ------- + ----- + -------- +
|  101 | sindhu  | 2001-04-22 | 2019-02-05 |   hr     | ece  |  70000 |
|  102 | laxmi    | 2000-03-24 | 2019-04-07 |    pr    | ece  |  60000 |
|  103 | arthi     | 2001-01-01 | 2019-02-05 |     rep  | cse  |  50000 |
|  104 | jothika | 2000-07-08 | 2015-02-09 |    vo    | it     |  40000 |
|  105 | divya   | 2000-05-07 | 2017-02-09 |    js      | it     |  55000 |
+--------+---------+----------------+---------------+---------+--------+---------+
 insert into empl_project values('101','ijkl','9')
  insert into empl_project values('102','ijkl','10')
 insert into empl_project values('103','mnop','13')
 insert into empl_project values('104','mnop','10')
 insert into empl_project values('105','qrst','11')
     select *from empl_project
```

```
+-----------+------------------+---------------------+
| p_id      | project_name | working_hours |
+-----------+------------------+---------------------+
|  101      | ijkl               |           9          |
|  102      | ijkl               |          10          |
|  103      | mnop             |          13           |
|  104    | mnop             |          10           |
|  105    | qrst               |         11          |
+---------+-------------------+--------------------+
```

1.Display the details of the employee working in project ‘ijkl’

```
    select *from employee where id in(select id from emp
l_project where project_name='ijkl')
+-------+----------+--------+---------+
| id     | name    | dept  | salary |
+-------+----------+--------+---------+
|  101 | sindhu  | ece   |  60000 |
|  102 | laxmi   | cse    |  50000 |
+-------+----------+--------+----------+
```

2.Display the project where ece department employees are involved

select *from empl_project where id in(select id from employee where dept='ece')

```
+-----------+-------------------+--------------------+
|p _ id     | project_name | working_hours |
+-----------+------------------+---------------------+
|  101      | ijkl               |          9          |
|  103     | mnop           |         13          |
|  105     | qrst              |         11          |
+----------+-----------------+--------------------+
```

3.Display the second highest salary

select max(salary) as salary from employee where sal
ary<(select max(salary) from employee)

```
  +-----------+
  | salary   |
  +-----------+
  |  55000  |
 +-----------+
```

4.Display the second highest salary in each department

select dept,max(salary)as salary from info where(dep t,salary)NOT IN(select dept,max(salary)from info group by dept)group by dept

```
+---------+---------+
| dept  | salary  |
+--------+----------+
| ece   |  60000 |
| it     |  40000  |
+-------+-----------+
```

5.Reterive the employee who joined last in the organization

select name,max(date_of_joning)from employee

```
 +----------+-----------------+
|name    | doj               |
+----------+------------------+
|  laxmi  | 2019-04-07    |
+----------+------------------+
```

6.Display the details of the employee who works for more no.of hours

select id,max(working_hours)as working_hours from em

pl_project where working_hours<(select max(working_hours)from empl_project)

```
+--------+--------------------+
| id      | working_hours  |
+--------+---------------------+
|  101  |          11          |
+--------+---------------------+
```

7.Create a view which has information about employee Id, employeename and project_name

```
  Create view workers  as select id,name,project_name from
employee,empl_project where id=p_id

      Select * from workers
```

```
+-------+---------------+------------------+
| id    | name          | project_name |
+-------+--------------+------------------+
|  101 |    sindhu    |   ijkl              |
|  102 |  laxmi       |    ijkl             |
|  103 |   arthi       |     mnop         |
|  104 |   jothika    |    mnop          |
|  105 |   divya       |      qrst          |
+-------+--------------+------------------+
```

Result :

Thus the queries for views and subqueries are verified.

EX NO:03 RELATIONAL ALGEBRA OPERATIONS

AIM

To implement the relational algebra operations.

DESCRIPTION

PRIMARY KEY:

Primary key constraints identifiers each record in a table. Primary key must contain the values and cannot contain NULL values. A table have only one primary key which may consists of single or multiple fields.

RELATIONAL ALGEBRA OPERATIONS:

The fundamental operations of relational algebra are as follows:

- Select
- Project
- Union
- Intersect
- Set difference
- Cartesian product
- Rename
- Join

1. SELECT OPERATION(σ):

It selects tuples that satisfy the given predicate from a relation.

2. PROJECT OPERATOR(π)

It projects columns that satisfy a given predicate.

3. UNION OPERATOR(u):

It performs binary union between two given relations.

4. CARTESIAN PRODUCT(×):

It combines information of two different relations into one notation.

5. RENAME OPERATION:

The result of relational algebra are also relations but without any name the rename operation rename the output relation.

6. MYSQL CROSS JOIN:

The cross join makes a Cartesian product of rows for multiple tables.

7. LEFT JOIN (R∞S):

To form on inner join, you need a condition which is known as a join prediction. An inner join requires rows in two joined tables to have matching column values.

8. RIGHT JOIN (R∞S):

The right join keyword returns all record from the right table and the matched records from the left table**.**

9. INSERT SIMULATION:

It returns rows only from the first statement that are identical to a row in select record statement.

10. SET DIFFERENCE:

The result is tuples which are present in relation but was not in the relation***.***

QUERIES

1. Create five tables and attributes of the following:

Table name 1: student

Table name attributes: student id, student name

```
MariaDB [(none)]> create database college;
 Query OK, 1 row affected (0.001 sec)


MariaDB [(none)]> use college;
```

```
 Database changed

MariaDB [college]> create table student(stu_id int(20),stu_name varchar(30));

 Query OK, 0 rows affected (0.124 sec)


 MariaDB [college]> insert into student values('101','dhanam'),('102','bhuvana'),
('103','pavi'),('104','deepika'),('105','rama'),('106','keerthi');

Query OK, 6 rows affected (0.036 sec)

 Records: 6  Duplicates: 0  Warnings: 0


MariaDB [college]> desc student;

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

  | Field        | Type           | Null  | Key | Default | Extra |

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

  | stu_id       |   int(20)        | YES  |      | NULL     |       |

  | stu_name | varchar(30)  | YES  |      | NULL     |       |

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

2 rows in set (0.004 sec)


MariaDB [college]> select *from student;

+--------+----------+ | stu_id | stu_name | +--------+----------+

                         |    101 | Dhanam   |

                          |    102 | Bhuvana  |

                        |    103 | Pavi       |

                        |    104 | Deepika  |

                        |   105  | Rama       |
|    106 | Keerthi    |
+--------+----------+ 6 rows in set (0.000 sec)
```

Table name 2 : course

Table attributes : course id, course name

```
MariaDB [college]> create table course(cour_id int(20),cour_name varchar(40));

Query OK, 0 rows affected (0.149 sec)


MariaDB [college]> insert into course values('1','dbms'),('2','os'),('3','ca'),(
'4','tm'),('5','coi'),('1','dbms');

Query OK, 6 rows affected (0.053 sec)

Records: 6  Duplicates: 0  Warnings: 0


MariaDB [college]> select *from course;

+---------+-----------+   | cour_id | cour_name | +---------+-----------+
                          |       1 | dbms      |
                          |       2 | os        |
                          |     3   | ca        |
                          |       4 | tm        |
                          |       5 | coi       |
                          |     6   | ds        |
                            +---------+-----------+ 6 rows in set (0.000 sec)
```

Table name 3 : club

Table attributes : club id, club name

```
MariaDB [college]> create table club(club_id int(10),club_name varchar(20));
Query OK, 0 rows affected (0.137 sec)
```

```
MariaDB [college]> insert into club values('01','lds'),('02','rotaract'),('03','
sports'),('04','yrc'),('05','ncc'),('06','finearts');
Query OK, 6 rows affected (0.041 sec)
Records: 6  Duplicates: 0  Warnings: 0

 MariaDB [college]> select *from club;
+---------+-----------+
| club_id | club_name |
+---------+-----------+
|       1 | lds       |
|       2 | rotaract  |
|       3 | sports    |
|       4 | yrc       |
|       5 | ncc       |
|       6 | finearts  |
+---------+-----------+
6 rows in set (0.001 sec)
```

Table name 4 : course enrolment

Table attributes : student id, course id, date of joining

```
MariaDB [college]> create table course_entrollment(stu_id int(20),cour_id int(30
),doj date);
Query OK, 0 rows affected (0.143 sec)
```

```
MariaDB [college]> insert into course_entrollment values('101','1','2015/4/5'),(
'102','2','2015/3/19'),('103','3','2018/6/14'),('104','4','2017/2/19'),('105','5
','2018/7/4'),('106','6','2019/9/20');
Query OK, 6 rows affected (0.044 sec)
Records: 6  Duplicates: 0  Warnings: 0


MariaDB [college]> select *from course_entrollment;
                    +--------+---------+------------+
                    | stu_id | cour_id|      doj      |
                     +--------+---------+------------+
                     |    101 |      1    | 2015-04-05 |
                     |    102 |      2    | 2015-03-19 |
                     |    103 |      3    | 2018-06-14 |
                     |    104 |      4    | 2017-02-19 |
                     |    105 |      5    | 2018-07-04 |
                     |    106 |      6    | 2019-09-20 |
+--------+---------+------------+
6 rows in set (0.000 sec)
```

Table name 5 : club involvement

Table attributes : student id, club id, date of joining

```
MariaDB [college]> create table club_involvement(stu_id int(20),club_id int(10),
doj date);
Query OK, 0 rows affected (0.125 sec)


MariaDB [college]> insert into club_involvement values('101','1','2018/7/5'),('1
03','3','2017/4/5'),('106','6','2019/9/20');
```

```
Query OK, 3 rows affected (0.033 sec)

Records: 3  Duplicates: 0  Warnings: 0


MariaDB [college]> insert into club_involvement values('101','1','2018/7/5'),('1
03','3','2017/4/5'),('106','6','2019/9/20');

 Query OK, 3 rows affected (0.040 sec)

Records: 3  Duplicates: 0  Warnings: 0


MariaDB [college]> select *from club_involvement;

                    +--------+---------+------------+
                     | stu_id  | club_id |     doj       |
                        +--------+---------+------------+
                    |    101  |       1      | 2018-07-05 |
                    |    102  |       2     | 2017-04-05 |
                    |    103  |       3      | 2019-09-20 |
                    |    104  |       4      | 2018-07-05 |
                    |    105  |       5      | 2017-04-05 |
                    |    106  |       6      | 2019-09-20 |
                       +--------+---------+------------+

6 rows in set (0.000 sec)
```

2. select the student whose names starts with the letter 'D'

```
MariaDB [college]> select *from student where stu_name like'd%';

                    +--------+----------+
                     | stu_id | stu_name |
                      +--------+----------+
                     |    101 | Dhanam     |
                     |    104 | Deepika     |
                       +--------+----------+
```

2 rows in set (0.001 sec)

3. select roll number, name of the student whose names ends with the letter ‘i’

MariaDB [college]> select *from student where stu_name like'%i';

```
+--------+----------+
| stu_id | stu_name |
+--------+----------+
|     103| pavi         |
|     106 | keerthi    |
+--------+----------+
```

2 rows in set (0.001 sec)

4. Perform Cartesian products for course and clubs

MariaDB [relations]> select *from course,club;

```
+------+--------+---------+-----------+
| c_id | c_name | club_id | club_name |
+------+--------+---------+-----------+
|    1 | dbms   |       1 | lds       |
|    2 | os     |       1 | lds       |
|    3 | ca     |       1 | lds       |
|    4 | tm     |       1 | lds       |
|    5 | coi    |       1 | lds       |
|    6 | ds     |       1 | lds       |
|    1 | dbms   |       2 | rotaract  |
|    2 | os       |      2 | rotaract  |
|    3 | ca       |      2 | rotaract  |
|    4 | tm       |      2 | rotaract  |
|    5 | coi      |      2 | rotaract  |
|    6 | ds       |      2 | rotaract  |
|    1 | dbms  |      3 | sports     |
|    2 | os       |      3 | sports     |
|    3 | ca       |      3 | sports     |
|    4 | tm       |      3 | sports     |
|    5 | coi      |      3 | sports     |
|    6 | ds       |      3 | sports     |
|    1 | dbms  |      4 | yrc          |
|    2 | os      |      4 | yrc          |
|    3 | ca      |      4 | yrc          |
|    4 | tm     |      4 | yrc          |
|    5 | coi     |      4 | yrc          |
|    6 | ds     |      4 | yrc          |
|    1 | dbms |   5    | ncc          |
|    2 | os     |      5 | ncc          |
|    3 | ca     |      5 | ncc          |
|    4 | tm     |      5 | ncc          |
|    5 | coi       |      5 | ncc      |
|    6 | ds        |      5 | ncc      |
|    1 | dbms   |       6 | sjc       |
|    2 | os     |       6 | sjc       |
|    3 | ca     |       6 | sjc       |
|    4 | tm     |       6 | sjc       |
|    5 | coi    |       6 | sjc       |
|    6 | ds     |       6 | sjc       |
+------+--------+---------+-----------+
36 rows in set (0.001 sec)
```

5. Display all the students enrolled for course or involved in club activities

```
MariaDB [relations]> select stu_id from course_enrollment union select stu_id
from club_involvement;
+--------+
| stu_id |
+--------+
|    101 |
|    102 |
|    103 |
|    104 |
|    105 |
|    106 |
+--------+
6 rows in set (0.001 sec)
```

6. Display the student names who are all involved in LDS club and enrolled for DBMS

```
MariaDB[relations]>select stu_id from club_involvement where club_id='1'
intersect select stu_id from course_enrollment where c_id='1';

+--------+

| stu_id |

+--------+

|    101 |

+--------+
```

7. Display the student names who are enrolled for DBMS but not involved in club activities

```
MariaDB [relations]> select stu_id from course_enrollment where c_id in(select
c_id from course where c_name='dbms' and stu_id not in(select stu_id from
club_involvement));

+--------+

| stu_id |

+--------+

|    101 |

+--------+

1 row in set (0.002 sec)
```

8. Display the details of student who are not involved in any club activities

```
MariaDB [relations]> select *from student where stu_id not in (select stu_id
from club_involvement);

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

| stu_id | stu_name |

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

|    102 | Bhuvana    |

|    106 | keerthi       |
 +--------+----------+
```

2 rows in set (0.001 sec)

9. Display all the details of student along with club activities they are involved in

```
MariaDB [relations]> select *from club_involvement left join club on club_involvement.club_id=club.club_id;

+--------+---------+------------+---------+-----------+
| stu_id | club_id | doj        | club_id | club_name |
+--------+---------+------------+---------+-----------+
|    101 |       1 | 2018-07-15 |       1 | lds       |
|    103 |       2 | 2017-04-05 |       2 | rotaract  |
|    104 |       4 | 2018-07-05 |       4 | yrc       |
|    105 |       5 | 2016-07-09 |       5 | ncc       |
+--------+---------+------------+---------+-----------+
4 rows in set (0.009 sec)
```

RESULT:

Thus the queries of relational algebra operations are implemented.

EX.NO:05 STORED PROCEDURES AND CURSORS

Aim

To perform fundamental operations of stored procedures and cursors in mysql.

DESCRIPTION

Stored procedure

stored procedure is a prepared SQL code that you can save, so the code can be reused over and over again. So if you have an SQL query that you write over and over again, save it as a stored procedure, and then just call it to execute it.

1)Create procedure

Use the create procedure statement to create a stand alone stored procedure or call specification. the stored procedure more flexible and useful in mysql.a parameter has end of three modes IN,OUT,INOUT.

IN is the default mode. When you define an IN parameter in a stored procedure, the calling program has to pass an argument to the stored procedure. In addition, the value of an IN parameter is protected. It means that even the value of the IN parameter is changed inside the stored procedure, its original value is retained after the stored procedure ends. In other words, the stored procedure only works on the copy of the IN parameter.

The value of an OUT parameter can be changed inside the stored procedure and its new value is passed back to the calling program. Notice that the stored procedure cannot access the initial value of the OUT parameter when it starts.

An INOUT parameter is a combination of IN and OUT parameters. It means that the calling program may pass the argument, and the stored procedure can modify the INOUT parameter, and pass the new value back to the calling program.

2)Cursor

A cursor is a pointer to this context area. PL/SQL controls the context area through a cursor. A cursor holds the rows (one or more) returned by a SQL statement. The set of rows the cursor holds is referred to as the **active set**.
You can name a cursor so that it could be referred to in a program to fetch and process the rows returned by the SQL statement, one at a time. There are two types of cursors −

1) Implicit cursors
2) Explicit cursors

1)Create a employee table , insert values

```
create table emp(emp_id int(11),name varchar(20),age int(4),designation varchar(20),salary int(5));

insert into emp
values(101,'mani','27','manager','75500'),(102,'rani','30','hr','80000'),(103,'felina','29','project
manager','60000'),(104,'arifa','25','project
head','55000'),(105,'shireen','27','hr','67000'),(106,'mathi','23','assistant hr','56000');

select*from emp;
```

```
+-----------+-----------+-------+----------------------+---------+
| emp_id | name    | age  | designation       |salary |
+-----------+-----------+-------+---------------------+----------+
|    101    | mani     |   27  | manager            |  75500 |
|    102    | rani      |   30  | hr                      |  80000 |
|    103    | felina    |   29  | project manager |  60000 |
|    104    | arifa      |   25  | project head       |  55000 |
|    105    | shireen |   27  | hr                      |  67000 |
|    106   | mathi     |   23  | assistant hr        |  56000 |
+----------+-----------+--------+---------------------+-----------+
```

2)Display the name and designation using cursor

```
SET namelist=CONCAT(s_name,";",namelist);

SET desiglist=CONCAT(desig,";"' at line 13

DELIMITER $$

CREATE PROCEDURE list(INOUT namelist varchar(4000),INOUT desiglist varchar(4000))

    BEGIN

    DECLARE is_done INTEGER DEFAULT 0;

    DECLARE s_name varchar(100) DEFAULT " ";

    DECLARE desig varchar(100) DEFAULT " ";

    DECLARE emp_cursor CURSOR FOR

    SELECT name,designation FROM emp;
    DECLARE CONTINUE HANDLER FOR NOT FOUND SET is_done=1;
   OPEN emp_cursor;
    get_list:LOOP
    FETCH emp_cursor INTO s_name,desig;
   IF is_done=1 THEN
    LEAVE get_list;
    END IF;
    SET namelist=CONCAT(s_name,";",namelist);
    SET desiglist=CONCAT(desig,";",desiglist);
    END LOOP get_list;
    CLOSE emp_cursor;
    END $$
set @namelist="",@desiglist="";
call list(@namelist,@desiglist);
MariaDB [employee_database]> select @namelist,@desiglist;
+----------------------------------+---------------------------------------------------------+
| @namelist                        | @desiglist                                              |
+----------------------------------+---------------------------------------------------------+
| mathi;shireen;arifa;felina;rani;mani; | assistant hr;hr;project head;project manager;hr;manager; |
+----------------------------------+---------------------------------------------------------+
```

RESULT:

Thus,the fundamental operations of stored procedure and cursor is implemented and output is displayed.

Ex.no:06 STORED FUNCTIONS

Aim:

To perform factorial operations by using stored function in mysql.

Description:

A stored function is a special kind stored program that returns a single value. Typically, you use stored functions to encapsulate common formulas or business rules that are reusable among SQL statements or stored programs.

Different from a stored procedure, you can use a stored function in SQL statements wherever an expression is used. This helps improve the readability and maintainability of the procedural code.

A stored function (also called a user function or user-defined function) is a set of PL/SQL statements you can call by name. Stored functions are very similar to procedures, except that a function returns a value to the environment in which it is called. User functions can be used as part of a SQL expression.

Syntax for stored functions:

```
DELIMITER $$

CREATE FUNCTION function_name(
   param1,
   param2.
)
RETURNS datatype

[NOT] DETERMINISTIC
```

```
BEGIN

 -statements

END $$

DELIMITER ;
```

To create a stored function ,you use the create function statement

Syntax for create statement:

```
CREATE_FUNCTION function_name(func_parameter1, func_parameter2, ..)

        RETURN datatype [characteristics]

        func_body
```

1.Factorial of given number using stored functions.

Query:

```
mysql> Delimiter //

mysql> CREATE PROCEDURE fact(IN x INT)

    -> BEGIN

    -> DECLARE result INT;

    -> DECLARE i INT;

    -> SET result = 1;

    -> SET i = 1;

    -> WHILE i <= x DO

    -> SET result = result * i;

    -> SET i = i + 1;

    -> END WHILE;

    -> SELECT x AS Number, result as Factorial;

    -> END//
```

Output:

```
mysql> Delimiter ;

mysql> CALL Fact(5);


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

| Number | Factorial |

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

|    5   |   120   |

+-------------+---------------+
mysql> CALL Fact(6);

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

| Number  | Factorial |

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

|    6    |   720   |

+---------------+---------------+
```

Result:

Thus, the implementation of stored functions in mysql is done experimentally and the output is verified.

EXP NO:7 TRIGGERS

AIM:

To perform fundamental operations of triggers in mysql.

DESCRIPTION:

TRIGGERS:

A trigger is a stored procedure in database which automatically involves whenever a special event in the database occurs. For example, a trigger can be involved when arrow is inserted into a specified table or when certain table columns are being updated.

Trigger can also contain INSERT and DELETE logic within itself , so when the trigger is fired because of data modification it can also cause another data modification, thereby fixing another trigger. A trigger that contains data modification logic within itself is called a nested trigger.

TYPES OF TRIGGERS:

1.AFTER INSERT:

Activated after data is inserted into the table.

2.AFTER UPDATE:

Activated after the data in the table is updated.

3.AFTER DELETE:

Activated after the data is deleted or removed from the table.

4.BEFORE INSERT:

Activated before data is inserted into the table.

5.BEFORE UPDATE:

Activated before the data in the table is modified.

6.BEFORE DELETE:

Activated before the data is deleted or removed from a table.

BEFORE AND AFTER TRIGGER:

Before trigger run the trigger action before the triggering statement is run.

After trigger run the trigger action after the triggering statement is run.

SYNTAX:

Create trigger[trigger_name]

```
[before/after]

{insert/update/delete}

on[table_name]

[for each row]

[trigger_body]
```

PROGRAM:

1.Create employee table with columns(emp no,name,age,job,salary)

```
Create database employee;

Use employee;

Create table emp(emp_no int(20),empname varchar(20),age int(20),job
varchar(20),salary int(6));

Select * from emp;
```

```
+------------+---------------+-----------+-----------------------+--------------+
| empno | empname | age  | job         | salary |
+-------+---------+------+-------------+--------+
|     1 | a       |   25 | worker      |  15000 |
|     2 | b       |   34 | manager     |  34000 |
|     3 | c       |   29 | hr          |  28000 |
|     4 | d       |   30 | worker      |  20000 |
|     5 | e       |   32 | supervisior |  30000 |
|     6 | f       |   28 | hr          |  20000 |
+-------+---------+------+-------------+--------+
```

2.Write a trigger to ensure that NO employees of age less than 25 can be inserted in the table.

```
delimiter $$

CREATE TRIGGER  Check_age  BEFORE INSERT ON employee

 FOR EACH ROW

BEGIN

IF NEW.age < 25 THEN
```

```
        SIGNAL SQLSTATE '45000'

        SET MESSAGE_TEXT = 'ERROR:   AGE MUST BE ATLEAST 25 YEARS!';

       END IF;

       END; $$

       delimiter;
```

```
   =>Insert into emp values(7,"g",20,"worker",15000);
```

ANSWER: ERROR: AGE MUST BE ATLEAST 25 YEARS!

3.Create a trigger which will work before deletion in employee table and create a duplicate copy of the record in another table employee_backup.

```
                    create table employee_backup (employee_no int(90),      employee_name
varchar(40),age int(90), job varchar(40), salary int(90));
```

```
     delimiter $$
     CREATE TRIGGER Backup BEFORE DELETE ON vv
      -> FOR EACH ROW
      -> BEGIN
      -> INSERT INTO employee_backup
      -> VALUES (OLD.empno, OLD.empname, OLD.age,OLD.job, OLD.salary);
      -> END; $$
 Query OK
  -> delete from emp where empno=6;
 Query OK

   select *from emp;
```

```
+-----------+--------------+-------+---------------+---------+
| empno | empname | age  | job          | salary |
+-----------+------------ --+-------+--------------+---------+
|    1.      | a             |   25 | worker     |  15000 |
|    2       | b             |   34 | manager.  | 34000 |
|    3.      | c             |   29 | hr             |  28000 |
|    4       | d             |   30 | worker      |  20000 |
|    5       | e             |   32 | supervisior|  30000 |
+-----------+---------------+------+---------------+-----------+
```

select *from employee_backup;

```
+-------------------+------------------------+-------+------+---------+
| employee_no | employee_name | age  | job  | salary |
+-------------------+------------------------+-------+------+---------+
|          6          | f                          |   28  | hr    |  20000|
+-------------------+------------------------+-------+-------+---------+
```

4.Write a trigger to count number of new tuples inserted using each insert statement.

```
                SET @COUNT=0;
                CREATE TRIGGER Count_tupples
                   ->   AFTER INSERT ON emp
                  -> FOR EACH ROW
                  -> BEGIN
                  -> SET @count = @count + 1;
                 -> END; $$
       Query OK
    => insert into emp values(7,'g',32,'md',50000);
    -> $$
   => select @count;
+------------+
| @count |
+------------+
|     1       |
+------------+
      => insert into emp values(8,'h',32,'worker',25000);

      => select @count;

+------------+
| @count |
+------------+
|      2      |

+------------+
```

RESULT:

Thus the trigger is created and output is verified successfully.

EX.NO:08 DCL AND TCL COMMANDS

AIM

To execute DCL and TCL commands in mysql.

DESCRIPTION

I. DCL:

DCL refers to data control language. DCL includes commands such as GRANT and REVOKE which mainly deals with the rights, permissions and other controls of database system.

GRANT ->Gives user's access privileges to database.

REVOKE -> withdraw user's access privileges given by using the grant command.

II. TCL:

TCL refers to transaction control language. TCL commands deals with the transaction within the database.

COMMIT ->Commits a transaction.

ROLLBACK -> Rollbacks a transaction in case of any error occurs.

SAVEPOINT ->Sets a savepoint within a transaction.

SET TRANSACTION ->Specify characteristics for the transaction.

1) Create employee table with columns (emp id, name, age, designation, salary)

```
MariaDB [employee]> create table emp_details(emp_id int(3), name varchar(15),age int(3),designation varchar(25),salary int(5));
```

2) Insert values into employee table

```
insert into emp_details values(100,'kunal' , 26,'JE',250000),(101,'mishti',28,'architect',40000),(102,'karthick',25,'junior architect',35000);
```

```
select *from emp_details;

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

| emp_id | name     | age  | designation      | salary   |
+--------+----------+------+----------------+----------+
|   100  | kunal    |  26  | JE             | 250000 |
|   101  | mishti   |  28  | architect      |  40000  |
|   102  | karthick |  25  | junior architect |  35000  |
+--------+----------+------+----------------+----------+
```

3) Using DCL commands

I. **Give permission to the user "ABC" to access all the columns in employee table**

```
select user from mysql.user;
```

```
+------+
| User |
+------+
| root  |
| root  |
| pma  |
| root  |
+------+
```

```
create user ABC@localhost;
select user from mysql.user;
```

```
+--------+
| User    |
+--------+
| root    |
| root    |
| ABC  |
| pma    |
| root    |
+--------+
show grants for ABC@localhost;
+-----------------------------------------------------+
| Grants for ABC@localhost                            |
+-----------------------------------------------------+
| GRANT USAGE ON *.* TO 'ABC'@'localhost'   |
+-----------------------------------------------------+
grant all on employee.emp_details to ABC@localhost;
show grants for ABC@localhost;
+-------------------------------------------------------------------------------------
+
| Grants for ABC@localhost                                                            |
+-------------------------------------------------------------------------------------
+
| GRANT USAGE ON *.* TO 'ABC'@'localhost'                              |
| GRANT ALL PRIVILEGES ON `employee`.`emp_details` TO
'ABC'@'localhost' +-------------------------------------------------------------------
--------------------+
```

II. Again deny the permission given to the user "ABC"

```
 revoke all, grant option from ABC@localhost;
show grants for ABC@localhost;
+-----------------------------------------------------+
| Grants for ABC@localhost                            |
+-----------------------------------------------------+
| GRANT USAGE ON *.* TO 'ABC'@'localhost' |
+--------------------------------------------------------+
```

1) Create student table with columns (student id, student name, location)

```
create table student_details(stud_id int(3),stud_name varchar(20),location varchar(20));
```

2)Insert into the values in student table

Student id	**student name**	**location**
001	**x**	**Chennai**
002	**y**	**Mumbai**
003	**z**	**Delhi**
004	**u**	**Bangalore**
005	**v**	**Kolkata**

```
insert into student_details values(1,'x','chennai'),(2,'y','mumbai'),(3,'z','delhi'),(4,'u','bangalore'),(5,'v','kolkata');
```

```
 select *from student_details;
+---------+--------------+--------------+
| stud_id | stud_name | location  |
+---------+--------------+-------------+
|       1     | x                 | chennai     |
|       2     | y                 | mumbai    |
|       3     | z                 | delhi         |
|       4     | u                 | bangalore |
|       5     | v                 | kolkata     |
+---------+--------------+-------------+
```

3)Display the student table

```
select *from student_details;
+---------+--------------+--------------+
| stud_id | stud_name | location  |
+---------+--------------+-------------+
|       1     | x                 | chennai     |
|       2     | y                 | mumbai    |
|       3     | z                 | delhi         |
|       4     | u                 | bangalore |
|       5     | v                 | kolkata     |
+---------+--------------+-------------+
```

3)Display the student table

```
select *from student_details;
+---------+--------------+--------------+
| stud_id | stud_name | location       |
+---------+--------------+--------------+
|       1     | x                 | chennai      |
|       2     | y                 | mumbai     |
|       3     | z                 | delhi          |
|       4     | u                 | bangalore  |
|       5     | v                 | kolkata      |
+---------+--------------+-------------+
```

4)Perform rollback command

```
start transaction;
delete from student_details where location='bangalore';
 select *from student_details;
```

```
+---------+-----------+------------+
| stud_id | stud_name | location |
Yo+---------+-----------+-------------+
|       1     | x              | chennai    |
|       2     | y              | mumbai   |
|       3     | z              | delhi         |
|       5     | v              | kolkata    |
+---------+-----------+------------+
 rollback;
select *from student_details;
+---------+-----------+------------+
| stud_id | stud_name | location  |
+---------+-----------+------------+
|       1     | x              | chennai     |
|       2     | y              | mumbai    |
|       3     | z              | delhi         |
|       4     | u             | bangalore  |
|       5     | v             | kolkata      |
+---------+-----------+-------------+
```

5)Perform rollback with commit command

```
update student_details set location='hydrabad' where stud_id=4;
  select *from student_details;
+---------+-----------+----------+
| stud_id | stud_name | location |
+---------+-----------+----------+
|      1    | x             | chennai    |
|      2    | y             | mumbai   |
|      3    | z             | delhi        |
|      4    | u             | hydrabad |
|      5    | v             | kolkata    |
+---------+-----------+----------+
commit;
 rollback;
 select *from student_details;
+---------+-----------+----------+
| stud_id | stud_name | location |
+---------+-----------+----------+
|      1    | x             | chennai    |
|      2    | y             | mumbai   |
|      3    | z             | delhi        |
|      4    | u             | hydrabad |
|      5     | v            | kolkata    |
+---------+-----------+----------+
```

6)Delete the details of the student where id is "003"

```
delete from student_details where stud_id=3;
select *from student_details;
+---------+--------------+----------+
| stud_id | stud_name | location |
+---------+--------------+----------+
|      1    | x                 | chennai   |
|       2    | y                 | mumbai  |
|       4    | u                | hydrabad |
|       5    | v                 | kolkata   |
+---------+-----------+-------------+
```

7)Delete details of the student where location is “Delhi”

```
delete from student_details where location='delhi';
 select *from student_details;
+---------+-------------+----------+
| stud_id | stud_name | location |
+---------+--------------+----------+
|       1    | x                 | chennai  |
|       2    | y                 | mumbai |
|       4    | u                 | hydrabad |
|       5    | v                 | kolkata  |
```

8)Use “SAVEPOINT” command after deleting (deleting details of the student where location is “Delhi”

```
start transaction;
savepoint sp;
```

9)Delete details of the student where name is ‘u’

```
delete from student_details where stud_name='u';
Query OK, 1 row affected (0.050 sec)


MariaDB [studentdb]> select *from student_details;
+---------+-----------+----------+
| stud_id | stud_name | location |
+---------+-----------+----------+
|      1.  | x          | chennai  |
|      2  | y          | mumbai   |
|      5  | v          | kolkata  |
+---------+-----------+----------+
```

10)Perfotm rollback with savepoint command

```
rollback  to  sp;

  select  *from  student_details;

+------------+-----------------+-------------+
|  stud_id  |  stud_name  |  location  |
+------------+-----------------+-------------+
|        1    |  x                |  chennai    |
|        2    |  y                |  mumbai     |
|        4    |  u                |  hydrabad  |
|        5    |  v                |  kolkata    |
```

RESULT

Thus the queries for dcl and tcl commands are verified.