**LAB 1:**

**TITLE: Data Definition Language (DDL) commands**

**OBJECTIVE:**

**Implement the DDL commands and constraints**

**THEORY:**

DDL (Data Definition Language) is a type of SQL command used to define data structures and modify data. It creates, alters, and deletes database objects such as tables, views, indexes, and users. Examples of DDL statements include CREATE, ALTER, DROP and TRUNCATE.

**Commands:**

1. CREATE:

Create database mydb;

create table book (

book\_id int primary key,

book\_name varchar(50),

price decimal

);

1. INSERT

Insert multiple tuple in table.

Insert into book (book\_id, book\_name, price)

values

(100,'Crystal',1000),

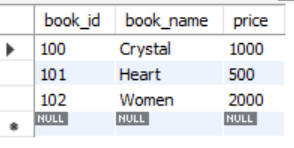
(101,'Heart',500),

(102,'Women',2000);

1. SELECT

To simply select or display the table.

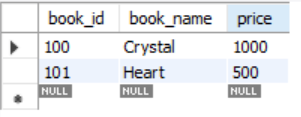
**select \* from book;**



1. DELETE:

To delete a row with id 102.

**delete from book where book\_id = 102;**

****

1. DROP:

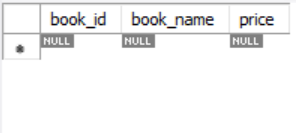
To delete entire table.

**drop table book;**

1. TRUNCATE:

To delete entire data from table.

**truncate table book;**



**Constraints:**

1. NOT NULL:

the column must have a value.

**create table book (**

**book\_id int primary key,**

**book\_name varchar(50) NOT NULL,**

**price decimal**

**);**

1. UNIQUE :

the column value must be unique in the table.

**create table book (**

**book\_id int primary key unique,**

**book\_name varchar(50) ,**

**price decimal**

**);**

1. CHECK

the column value conforms to an arbitrary condition.

**create table book (**

**book\_id int primary key unique,**

**book\_name varchar(50) ,**

**price decimal ,**

**check (price between 100 and 2000)**

**);**

**LAB 2:**

**TITLE: Select and update data in table.**

**OBJECTIVE:**

**To implement different select and update query in mysql.**

1. Create table of name employee and insert data.

CREATE TABLE employees (

employee\_id int PRIMARY KEY,

first\_name VARCHAR(50),

last\_name VARCHAR(50),

salary int,

address varchar(50)

);

INSERT INTO employees VALUES

(1, "Shyam", "Shrestha", 30000, "Kathmandu"),

(2, "Ram", "Bhandari", 20000, "Sindhupalchowk"),

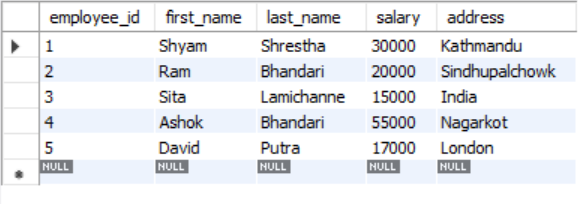
(3, "Sita", "Lamichanne", 15000, "India"),

(4, "Ashok", "Bhandari", 55000, "Nagarkot"),

(5, "David", "Putra", 17000, "London");

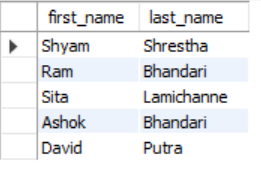
1. Select all the inserted data.

**SELECT \* FROM employees;**



1. Select only first name and last name.

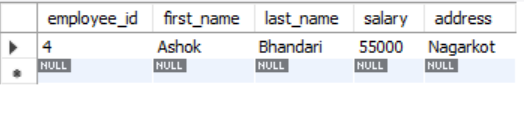
**SELECT first\_name, last\_name FROM employees;**



1. Select whose last name is Bhandari whose salary is greater than 40000.

SELECT \* FROM employees

WHERE last\_name = "Bhandari" and salary > 40000;

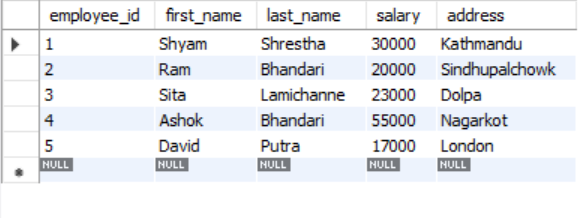


1. Update address and salary whose employee\_id is 3.

**UPDATE employees SET address = "Dolpa", salary =23000**

**WHERE employee\_id = 3;**

**SELECT \* FROM employees;**

****

**LAB 3:**

**TITLE: Union, Intersection and Rename operation in table.**

**OBJECTIVE:**

**Implement the union, intersection and rename operation in table.**

1. Create table of name std1 and insert data.

create table std1 (

std\_id int primary key,

std\_name varchar(50) ,

fee decimal

);

Insert into std1 (std\_id, std\_name, fee)

values

(100,'Ram',10000),

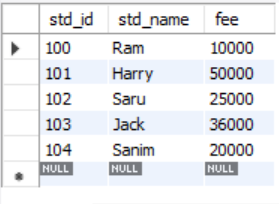
(101,'Harry',50000),

(102,'Saru',25000),

(103,'Jack',36000),

(104,'Sanim',20000);

select \* from std1;



1. Now, Create a table of name std2 and insert data.

create table std2 (

std\_id int primary key,

std\_name varchar(50) ,

fee decimal

);

Insert into std2 (std\_id, std\_name, fee)

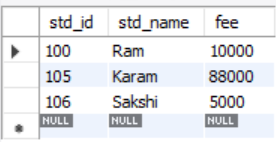
values

(100,'Ram',10000),

(105,'Karam',88000),

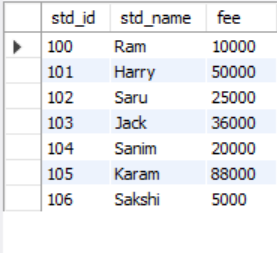
(106,'Sakshi',5000);

select \* from std2;



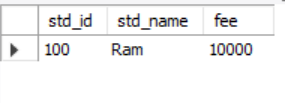
1. Perform “UNION” command.

select \* from std1 union select \* from std2;



1. Perform “INTERSECTION” command.

select \* from std1 intersect select \* from std2;



1. Perform “RENAME” command for std1 to student1.

rename table std1 to student1;

**OR**

alter table std1

rename to student1;

**LAB 4:**

**TITLE: Primary key, Foreign key and joining of 2 tables.**

**OBJECTIVE:**

**Implementation of primary key, foreign key and joining two tables.**

1. Create table of name course and insert data.

create table course

(

c\_id int primary key,

c\_name varchar(50)

);

Insert into course (c\_id, c\_name)

values

(101, 'DBMS'),

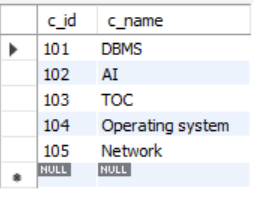
(102, 'AI'),

(103, 'TOC'),

(104, 'Operating system'),

(105, 'Network');

select \* from course;



1. Create table of name student and insert data.

create table student

(

std\_id int primary key,

std\_name varchar(50),

co\_id int,

foreign key (co\_id) references course (c\_id)

);

Insert into student (std\_id, std\_name, co\_id)

values

(301, "Ram", 101),

(302, "Sita", 105),

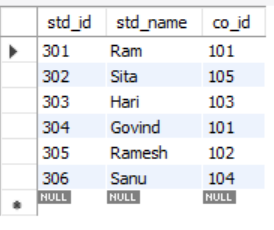
(303, "Hari", 103),

(304, "Govind", 101),

(305, "Ramesh", 102),

(306, "Sanu", 104);

select \* from student;

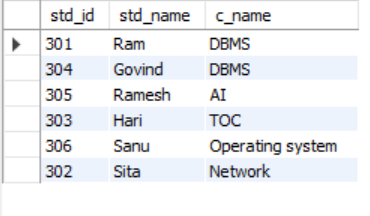


1. Join the two table with common keys.

select student.std\_id, student.std\_name, course.c\_name

from course Inner Join student

on student.co\_id = course.c\_id;



**LAB 5:**

**TITLE: CARTESIAN JOIN/** **CROSS JOIN.**

**OBJECTIVE:**

**To perform cartesian product between the tables.**

1. Create table named student and insert data.

CREATE DATABASE lab\_5;

USE lab\_5;

CREATE TABLE student (

std\_id INT PRIMARY KEY,

std\_name VARCHAR(50),

program VARCHAR(50)

);

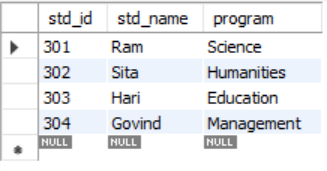
INSERT INTO student (std\_id, student, program) VALUES

(301, "Ram", 'Science'),

(302, "Sita", 'Humanities'),

(303, "Hari", 'Education'),

(304, "Govind", 'Management');



1. Create table named semester and student courses and insert the data.

CREATE TABLE sem (

std\_id INT,

semester INT,

PRIMARY KEY (std\_id, semester),

FOREIGN KEY (std\_id) REFERENCES student(std\_id)

);

INSERT INTO sem (std\_id, semester)

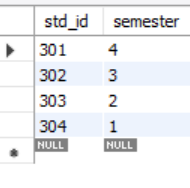
VALUES

(301, 4),

(302, 3),

(303, 2),

(304, 1);



CREATE TABLE std\_C (

std\_id INT,

sem INT,

PRIMARY KEY (std\_id, sem),

FOREIGN KEY (std\_id) REFERENCES student(std\_id)

);

INSERT INTO std\_C (std\_id, sem)

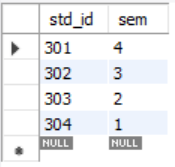
VALUES

(301, 4),

(302, 3),

(303, 2),

(304, 1);



1. Cross-Join the tables with common keys.

SELECT

student.std\_id,

student.std\_name,

sem.semester,

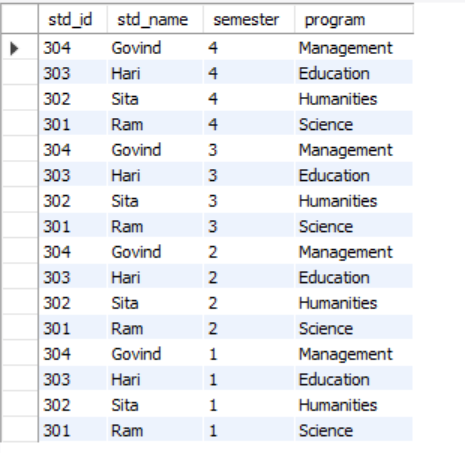
student.program

FROM

student

CROSS JOIN

sem;



**LAB 6:**

**OBJECTIVE:**

Consider the following database and write SQL as given:

Customer (Cno, Cname, Caddress, Ccontact)

Purchase (Cno, Pid)

Product (Pid, Pname, price, quantity)

a. Find the names of all products having price 1000.

b. Find the name of those customers who purchased Dell Laptop.

c. Find the total number of products purchased by customer ‘Ram’.

d. Increase price of all products by 5% .

e. Find total price of Dell Laptops.

1. Create table named customer, purchase, product and insert data.

create database lab\_6;

use lab\_6;

create table customer(

Cno int primary key,

Cname varchar(50),

Caddress varchar(50),

Ccontact int

);

insert into customer(Cno, Cname, Caddress, Ccontact) values

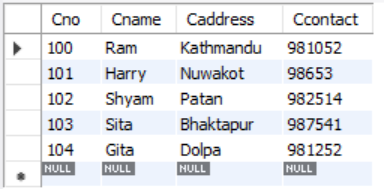
(100,'Ram', 'Kathmandu', 981052),

(101,'Harry', 'Nuwakot', 98653),

(102,'Shyam', 'Patan', 982514),

(103,'Sita', 'Bhaktapur', 987541),

(104,'Gita', 'Dolpa', 981252);



create table Product(

Pid int primary key,

Pname varchar(50),

Price int,

quantity int

);

insert into Product(Pid, Pname, Price, quantity) values

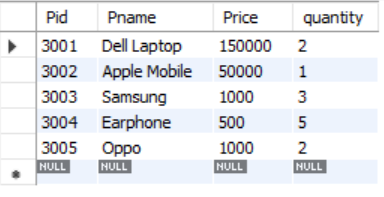
(3001,'Dell Laptop', 150000 , 2),

(3002,'Apple Mobile', 50000, 1),

(3003,'Samsung', 1000 , 3),

(3004,'Earphone', 500 , 5),

(3005,'Oppo', 1000 , 2);



create table Purchase(

Pu int primary key,

Cno int,

foreign key (Cno) references customer (Cno),

Pid int ,

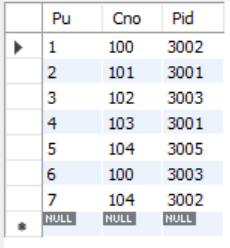
foreign key (Pid) references Product (Pid)

);

insert into Purchase(Pu, Cno, Pid) values

(1,100,3002),

(2,101,3001),

****(3,102,3003),

(4,103,3001),

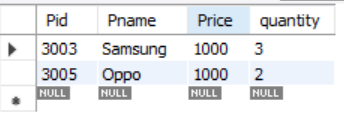
(5,104,3005),

(6,100,3003),

(7,104,3002);

1. Find the names of all products having price 1000.

**select \* from product where price = 1000;**



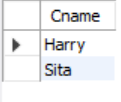
1. Find the name of those customers who purchased Dell Laptop.

**select c.Cname from customer c**

**join Purchase p on p.Cno = c.Cno**

**join product pr on p.Pid = pr.Pid**

**where pr.Pname = "Dell Laptop"**



1. Find the total number of products purchased by customer ‘Ram’.

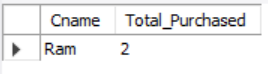
**SELECT c.Cname,**

**COUNT(pu.Pid) AS Total\_Purchased FROM customer c**

**JOIN Purchase pu ON c.Cno = pu.Cno**

**JOIN Product pr ON pu.Pid = pr.Pid**

**WHERE c.Cname = 'Ram'**

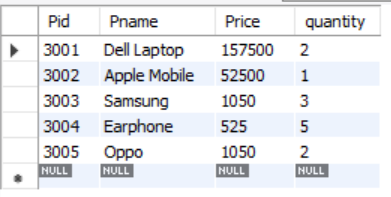


1. Increase price of all products by 5%.

**UPDATE Product**

**SET Price = Price \* 1.05;**

**select \* from Product;**

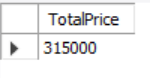


1. Find total price of Dell Laptops.

**SELECT SUM(p.Price \* p.quantity) AS TotalPrice**

**FROM Product p**

**WHERE p.Pname = 'Dell Laptop';**



**LAB 7:**

**OBJECTIVE:**

Retrive the TName, SName, SPhone for “ABC” school using SQL from given relation as below:

TEACHER (TID, TName, TAddress, TQualification)

SCHOOL (SID, SName, SAddress, SPhone)

SCHOOL\_TEACHER (SID, TID, No\_of Period)

1. Create table named teacher, school, school teacher and insert data.

create database lab\_7;

use lab\_7;

create table teacher(

Tid int primary key,

Tname varchar(50),

Taddress varchar(50),

Tqualification varchar(50)

);

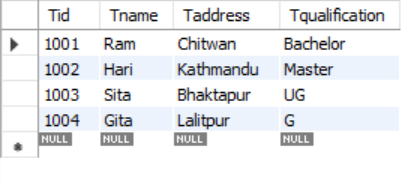
insert into teacher(Tid, Tname, Taddress, Tqualification) values

(1001,'Ram', 'Chitwan', 'Bachelor'),

(1002,'Hari', 'Kathmandu', 'Master'),

(1003,'Sita', 'Bhaktapur', 'UG'),

(1004,'Gita', 'Lalitpur', 'G');



create table school(

Sid int primary key,

Sname varchar(50),

Saddress varchar(50),

Sphone varchar(15)

);

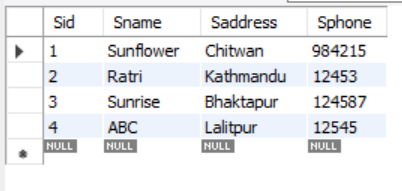
insert into school(Sid, Sname, Saddress, Sphone) values

(1,'Sunflower', 'Chitwan', 984215),

(2,'Ratri', 'Kathmandu', 012453),

(3,'Sunrise', 'Bhaktapur', 124587),

(4,'ABC', 'Lalitpur', 12545);



create table Steacher(

Sid int,

foreign key (Sid) references school(Sid),

Tid int,

foreign key (Tid) references teacher(Tid),

No\_Period int

);

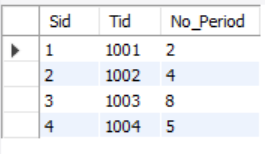
insert into Steacher(Sid, Tid, No\_Period) values

(1,1001, 2),

(2,1002, 4),

(3,1003, 8),

(4,1004, 5);



1. Display the TName, SName, SPhone for “ABC” school

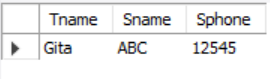
**Select T.Tname, S.Sname, S.Sphone**

**From Steacher St**

**Join school S on S.Sid=St.Sid**

**JOin teacher T on T.Tid=St.Tid**

**where S.SName='ABC'**



**LAB 8:**

**OBJECTIVE:**

Consider a banking database with three labels and primary key underlined as given

below:

Customer (CustomerID , CustomerName, Address, Phone, Email)

Borrows (CustomerID, LoanNumber )

Loan ( LoanNumber , LoanType, Amount )

Write both relational algebra and SQL queries:

1. To display name of all customers who live in “Lalitpur” in ascending order of name.
2. To count total number of customers having loan at the bank.
3. To find name of those customers who have loan amount greater than or equal to 500000.
4. To find average loan amount of each account type.
5. Create table named customer, borrows, loan and insert data.

create database lab\_8;

use lab\_8;

create table customer(

Cid int primary key,

Cname varchar(50),

Address varchar(50),

phone int,

email varchar(50)

);

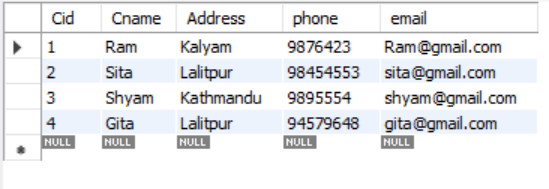
insert into customer(Cid, Cname, Address, phone,email) values

(1,'Ram', 'Kalyam', 9876423, 'Ram@gmail.com'),

(2,'Sita', 'Lalitpur', 98454553, 'sita@gmail.com'),

(3,'Shyam', 'Kathmandu', 9895554, 'shyam@gmail.com'),

(4,'Gita', 'Lalitpur', 94579648, 'gita@gmail.com');



create table loan(

Lid int primary key,

Ltype varchar(50),

Amount int

);

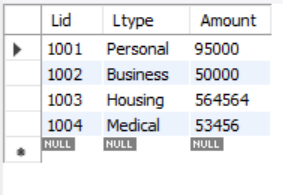
insert into loan(Lid, Ltype, Amount) values

(1001,'Personal', 95000),

(1002,'Business', 50000),

(1003,'Housing', 564564),

(1004,'Medical', 53456);



create table borrow(

Bid int primary key,

Cid int,

Foreign key (Cid) references customer(Cid),

Lid int,

Foreign key (Lid) references loan(Lid)

);

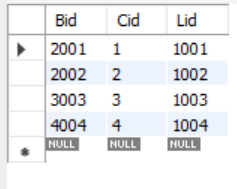
insert into borrow(Bid, Cid, Lid) values

(2001, 1, 1001),

(2002, 2, 1002),

(3003, 3, 1003),

(4004, 4, 1004);

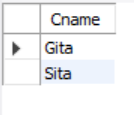


1. To display name of all customers who live in “Lalitpur” in ascending order of name.

**Select Cname from customer**

**where Address = 'Lalitpur'**

**order by 1 asc;**



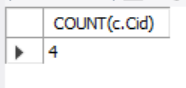
1. To count total number of customers having loan at the bank.

**SELECT COUNT(c.Cid)**

**FROM borrow b**

**JOIN customer c ON b.Cid = c.Cid**

**JOIN loan l ON b.Lid = l.Lid;**



1. To find name of those customers who have loan amount greater than or equal to 500000.

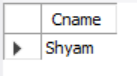
**SELECT c.Cname**

**FROM borrow b**

**JOIN customer c ON c.Cid = b.Cid**

**JOIN loan l ON l.Lid = b.Lid**

**WHERE l.Amount >= 500000;**



1. To find average loan amount of each account type.

**select distinct Ltype, avg(Amount)**

**From loan**

**group by Ltype;**

