****

**Samriddhi College**

**Lokanthali-1, Bhaktapur**

**B.Sc. CSIT Fourth Semester**

2078 Batch

Lab Report

on

**Database Management System (DBMS)**

Submitted by

**Name of the Student**

Symbol No.: **123456(for sample)**

TU Regd. #: **3-2-23-150-2020(for sample)**

**Chaitra, 2080**

**Index**

|  |  |  |  |
| --- | --- | --- | --- |
| ***Lab #*** | ***Lab Title*** | ***Page No.*** | ***Signature*** |
| 1 | Basic Introduction to SQL |  |  |
| 2 | Installation of MySQL Community Edition on MS Windows |  |  |
| 3 | SQL Queries Set 1 |  |  |
| 4 | SQL Queries Set 2 |  |  |
| 5 | SQL Queries Set 3 |  |  |
| 6 | SQL Queries Set 4 |  |  |
| 7 | SQL Queries Set 5 |  |  |
| 8 | Relational Database Design using ER diagram |  |  |
| 9 | Mini Project |  |  |

Lab 1

**Basic Introduction to SQL**

In this section students are supposed to write basic theory, commands and their syntax.

You are suggested to independently write this section.

**Refer to the resources uploaded to the drive folder. All your theories must be based on MYSQL.**

Lab 2

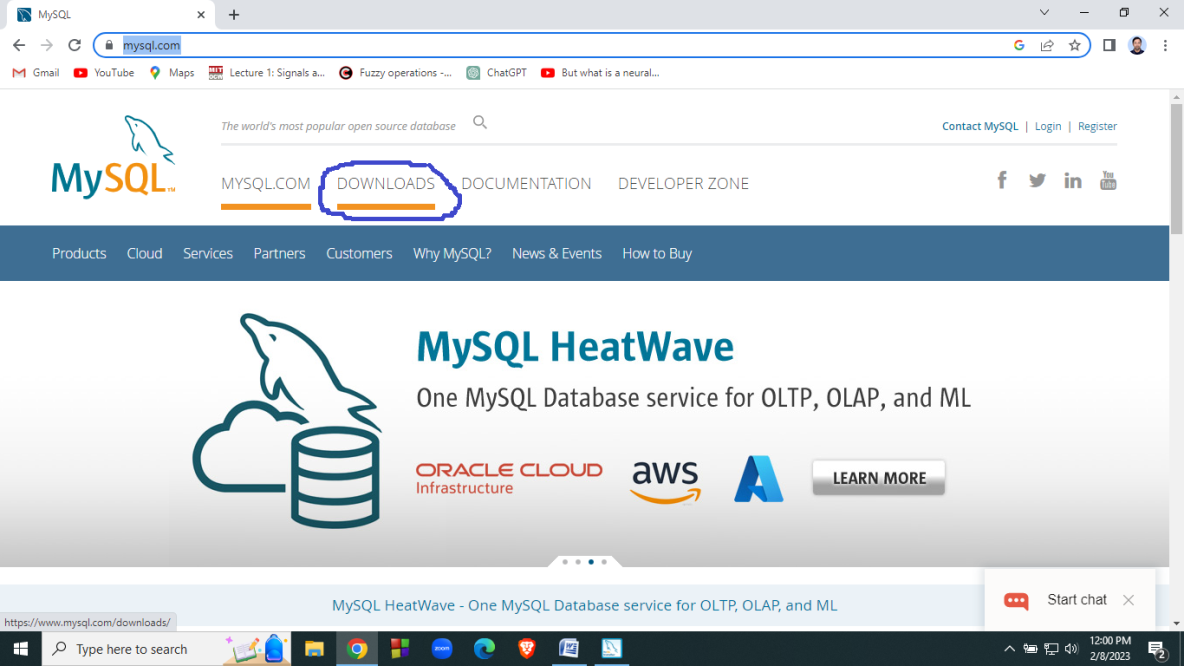
**Installation of MySQL Community Edition (GPL) on Windows**

In this lab, students are expected to learn the process of installing MySQL Installer and configuring MySQL server for running DBMS queries in MySQL Workbench.

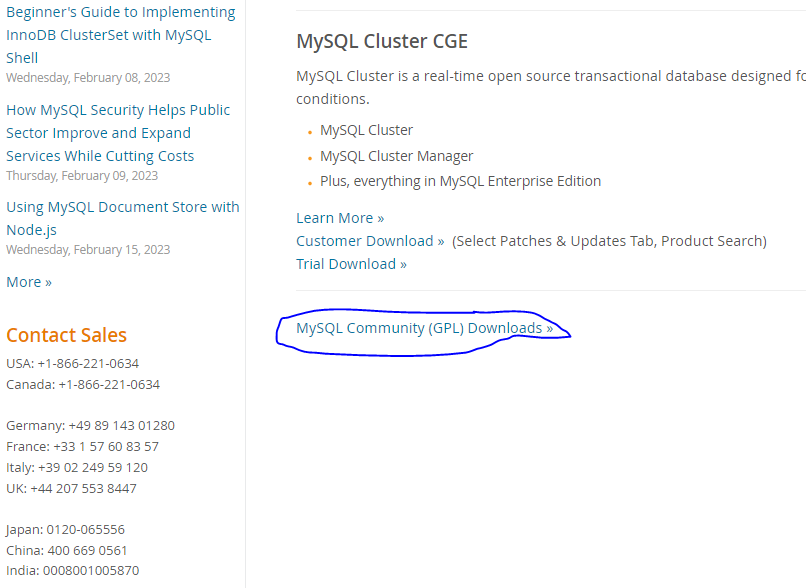
Following are the steps to be followed:

**Steps for downloading MySQL Community Edition:**

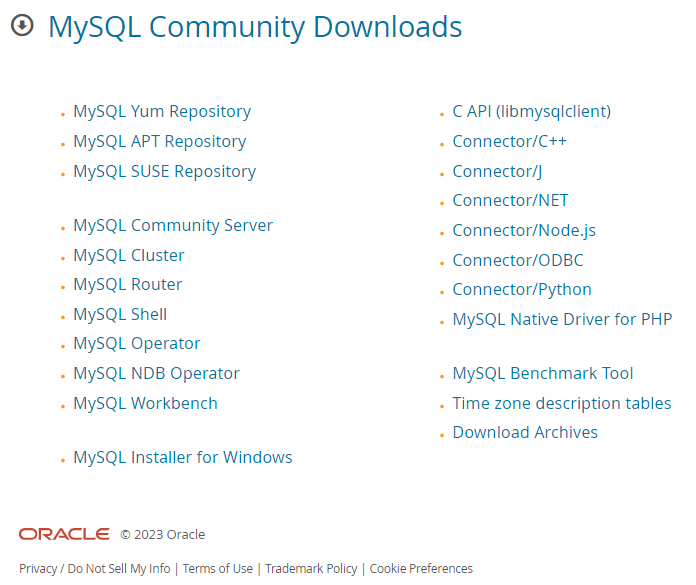
1. Go to <https://www.mysql.com/>
2. Click on Downloads



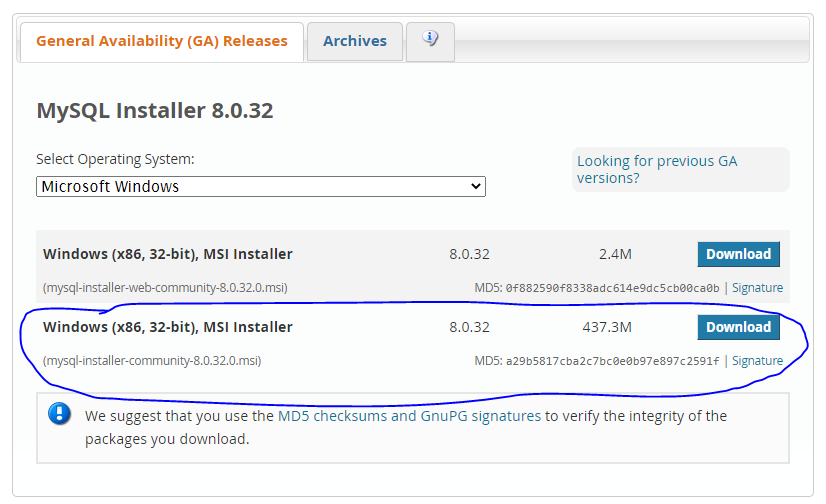
1. After clicking on Downloads, scroll the webpage and locate the link for MySQL Community(GPL) Downloads



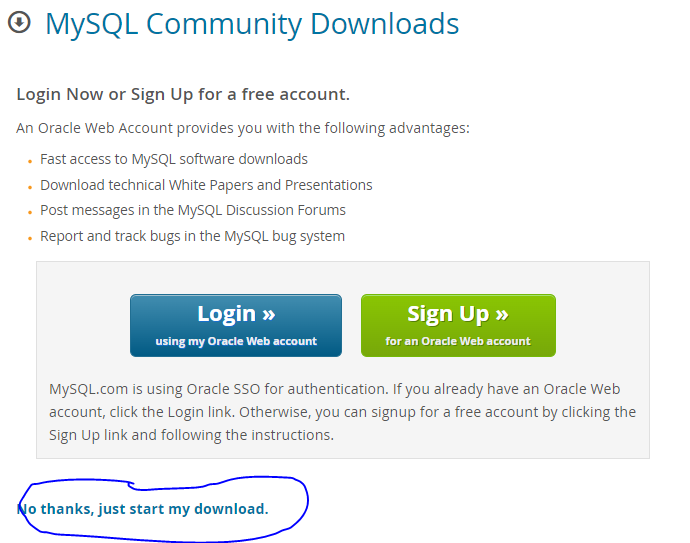
1. Upon click on the aforementioned link, following list of downloads will be shown:



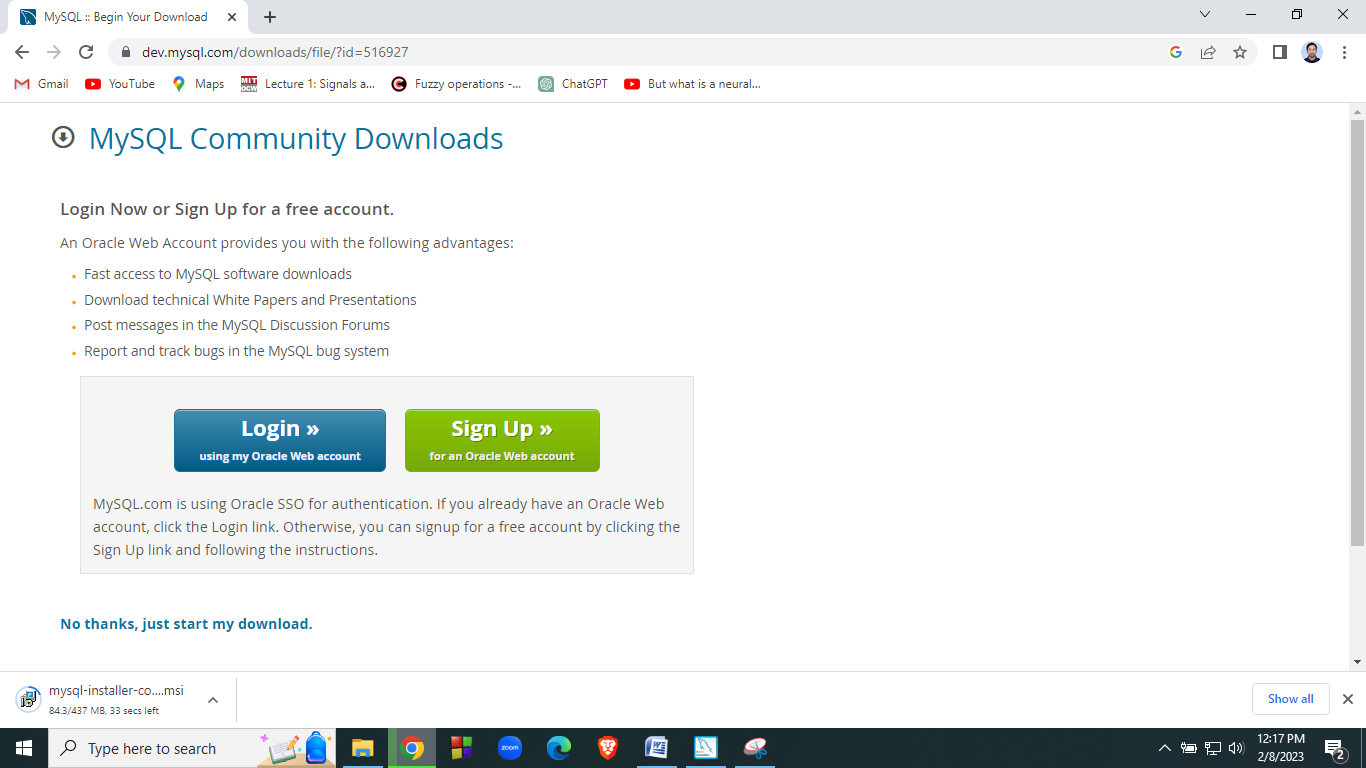
1. Click on MySQL Installer for Windows. Following download links will be displayed:



1. Click on Windows(x86, 32bit), MSI Installer for offline installation.
2. Before starting download, the web page suggests you to login or signup for oracle web account. If you like, you can open one. However, if you want to download directly, click on **No thanks, just start my download** as shown in the following image.



1. Now, the browser will start the download of installer file.



This completes the download procedure.

**Steps for installing MySQL Community Edition:**

1. Double click on the downloaded MSI installer file for MySQL community edition.

*Complete remaining steps yourself*!

Lab 3

**SQL Queries Set 1**

In this lab, students are expected to learn basic MySQL queries to create a database, use existing database, create tables with a set of attributes, insert values, set constraints on attributes etc.

**Q1: Perform the following taks:**

Task #1: Create a database called **DBMS\_CSIT**

Task #2: Use the database **DBMS\_CSIT**

Task #3: Create a table **student** with following schema

**student(name, roll, marks, address)**

Task #4: Populate the table with following data

|  |  |  |  |
| --- | --- | --- | --- |
| ***name*** | ***roll*** | ***marks*** | ***address*** |
| Ram | 12 | 98 | KTM |
| Shyam | 13 | 99 | PKR |
| Hari | 14 | 95 | BKT |
| Rita | 15 | 85 | TNU |
| Sita | 16 | 78 | KTM |

Task #5: Write SQL queries to display the records of students in the order of marks (both ascending and descending).

Task #6: Write SQL query to display the records of students in alphabetical order (both forward and reverse alphabetical order)

Task #7: Write SQL query to display details of a student with roll no 12.

Task #8: Write SQL query to display details of students whose name is “Ram”

Task #9: Write SQL query to add an attribute phone\_no

Task #10: Write SQL query to drop the attribute address.

**Task #1 Solution:**

create database DBMS\_CSIT;

**Task #2 Solution:**

use DBMS\_CSIT;

**Task #3 Solution:**

create table student

(

name varchar(50),

roll int,

marks int,

address varchar(50)

);

**Task #4 Solution:**

**insert into student values("Ram",12,98,"A");**

**insert into student values("Hari",13,77,"B");**

**insert into student values("Shyam",14,78,"C");**

**insert into student values("Gita",15,79,"D");**

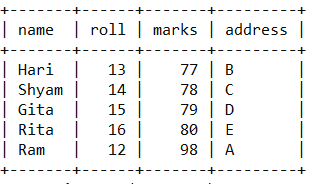
**insert into student values("Rita",16,80,"E");**

**Task #5 Solution:**

**select \* from student**

**order by marks;**

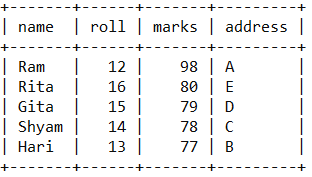
**Output:**

****

**select \* from student**

**order by marks desc;**

**Output:**

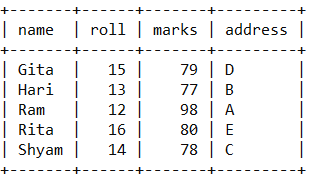
****

**Task #6 Solution:**

**select \* from student**

**order by name;**

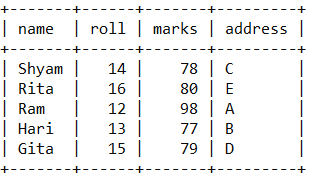
**Output:**

****

**select \* from student**

**order by name desc;**

**Output:**

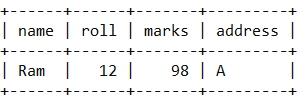
****

**Task #7 Solution:**

**select \* from student**

**where roll = 12;**

**Output:**

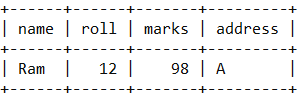
****

**Task #8 Solution:**

**select \* from student**

**where name = 'Ram';**

**Output:**

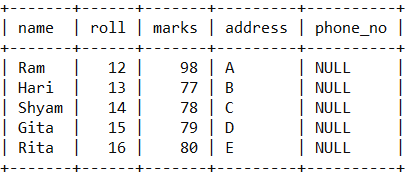
****

**Task #9 Solution:**

**alter table student**

**add column phone\_no varchar(10) default NULL;**

**Output:**

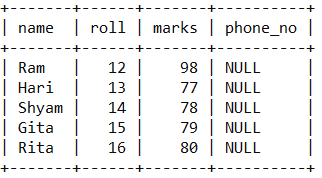
****

**Task #10 Solution:**

**alter table student**

**drop column address;**

**Output:**

****

**Design two additional SQL questions yourself with at least 5-7 tasks. Questions should be different from other questions in this lab report.**

Lab 4

**SQL Queries Set 2**

In this lab students are expected to learn SQL queries related to aggregate functions, setting integrity constraints(such as primary key, foreign key etc)

Q1. Create a table employee with following schema:

**employee(name, eid, designation, salary)**

Perform following tasks:

1. Populate employee table with 10 or more records
2. Write SQL query to retrieve all records from the table.
3. Write SQL query to set salary of all employees whose designation is "Supervisor"
4. Write SQL query to change the name of employee with eid=50 to "Hari"
5. Write SQL query to delete the record of a employee with eid=10
6. Write SQL query to display average salary of employees
7. Write SQL query to display the no. of employees
8. Write SQL query to display the total salary paid by the company.
9. Write SQL query to increase the salary of all employees by 10%.

***Solution*:**

**create table employee(**

**eid int not null primary key,**

**name varchar(50),**

**salary int,**

**designation varchar(50)**

**);**

**Task #1 Solution:**

**insert into employee values(12,"Ram",30000,"Peon");**

**insert into employee values(13,"Hari",12000,"Supervisor");**

**insert into employee values(14,"Shyam",13000,"Store Keeper");**

**insert into employee values(15,"Rita",14000,"Librarian");**

**insert into employee values(16,"Gita",15000,"Cook");**

**insert into employee values(17,"Sita",18000,"Gate Keeper");**

**insert into employee values(18,"Dinesh",12000,"Supervisor");**

**insert into employee values(19,"Nabin",12000,"Supervisor");**

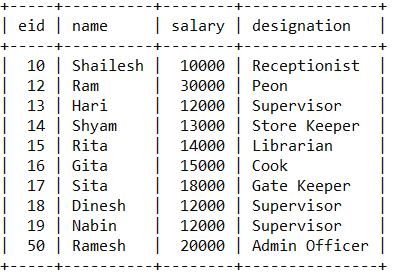
**insert into employee values(50,"Ramesh",20000,"Admin Officer");**

**insert into employee values(10,"Shailesh",10000,"Receptionist");**

**Task #2 Solution:**

**select \*from employee;**

Output:

****

**Task #3 Solution:**

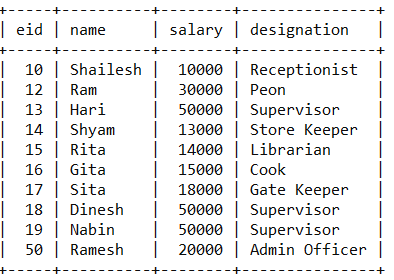
**SET SQL\_SAFE\_UPDATES = 0;**

**update employee**

**set salary=50000**

**where designation="Supervisor";**

Output:

****

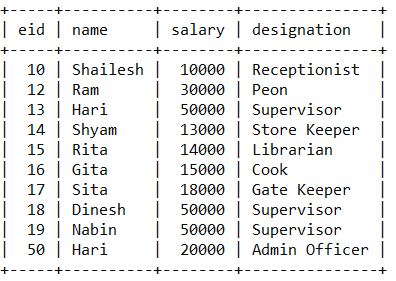
**Task #4 Solution:**

**update employee**

**set name="Hari"**

**where eid=50;**

Output:

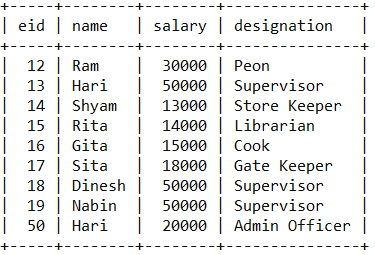
****

**Task #5 Solution:**

**delete from employee**

**where eid=10;**

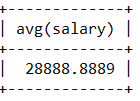
Output:

****

**Task #6 Solution:**

**select avg(salary) from employee;**

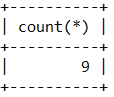
Output:

****

**Task #7 Solution:**

**select count(\*) from employee;**

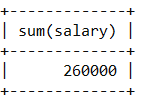
Output:

****

**Task #8 Solution:**

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

Output:

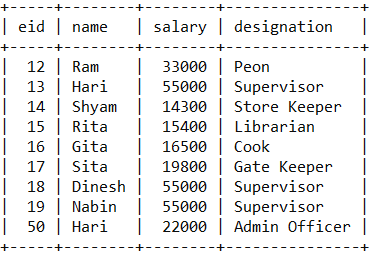
****

**Task #9 Solution:**

**update employee**

**set salary=1.1\*salary;**

Output:

****

**Q2. Consider the following tables:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Student** |  |  |  | **Course** |  |
| ***Name*** | ***Roll*** | ***CID*** |  | ***CID*** | ***Cname*** |
| Ram | 1 | S001 |  | S001 | DBMS |
| Shyam | 2 | S002 |  | S002 | TOC |
| Hari | 3 | S003 |  | S003 | CN |
| Rita | 4 | S001 |  | S004 | OS |
| Sita | 5 | S002 |  | S005 | Extra |
| Gita | 6 | S003 |  | S006 | AI |

**Perform the following tasks**

|  |  |
| --- | --- |
| 1 | Create two tables with following schema: |
|  | **Student(Name, Roll, CID)** |
|  | **Course(CID, Cname)** |
| 2 | Set CID of relation Student as foreign key which references CID of relation Course. |
| 3 | Populate the tables with records. |
| 4 | Write SQL query to retrieve records of all students along with course they took. |
| 5 | Write SQL query to display details of all students who took DBMS course. |
| 6 | Write SQL query to delete the table Course and comment on the result. |
| 7 | Write SQL query to insert a record ('Kartik', 7,'S007') into student table and comment on the result. |

**Solution:**

|  |
| --- |
| **Task #1 and 2 Solution:** |
| **create database dbms\_csit;** |
| **use dbms\_csit;** |
|  |
| **create table course** |
| **(** |
| **CID varchar(10),** |
| **Cname varchar(50),** |
| **primary key(CID)** |
| **);** |
|  |
| **create table student** |
| **(** |
| **name varchar(50),** |
| **roll int primary key,** |
| **CID varchar(10) ,** |
| **foreign key(CID) references Course(CID)** |
| **);** |
| **Task #3 Solution:** |
| **insert into course values('S001','DBMS');** |
| **insert into course values('S002','TOC');** |
| **insert into course values('S003','CN');** |
| **insert into course values('S004','OS');** |
| **insert into course values('S005','Extra');** |
| **insert into course values('S006','AI');** |
|  |
|  |
| **insert into student values('Ram',1,'S001');** |
| **insert into student values('Shyam',2,'S002');** |
| **insert into student values('Hari',3,'S003');** |
| **insert into student values('Rita',4,'S001');** |
| **insert into student values('Sita',5,'S002');** |
| **insert into student values('Gita',6,'S003');** |
| **Task # 4 Solution:** |
| **select \* from student natural join course;**  **Output:** |
| **Task #5 Solution:** |
| **select \* from student natural join course where cname='DBMS';** |
| **Output:**    **Task #6 Solution:** |
| **drop table course;** |
| **Output:**  **# generates following error message** |
| **Cannot drop table 'course' referenced by a foreign key constraint 'student\_ibfk\_1' on table 'student'.** |
| **Task #7 Solution:** |
| insert into student values('Kartik',7,'S007');  **Output:** |
| **#Foreign key constraint fails** |

Lab 5

**SQL Queries Set 3**

In this lab students are expected to learn SQL queries related to nested query, aggregate function, as, like and having clause.

**Q1.** **Consider the following COURSE table given below:**

|  |  |  |  |
| --- | --- | --- | --- |
| **CourseID** | **CourseName** | **CourseFee** | **Instructor** |
| 11 | Programming | 10000 | Ravi |
| 12 | C# | 15000 | Jiban |
| 13 | Java | 18000 | Janak |
| 14 | XML | 5000 | Ravi |
| 15 | Database | 12500 | Han |
| 16 | ASP.net | 10000 | Shyam |

Now answer the following questions:

1. Write SQL syntax to create the given table and insert few records in it.

**create table COURSE**

**(**

**CourseID integer primary key,**

**CourseName varchar(50),**

**CourseFee integer,**

**Instructor varchar(50)**

**);**

**insert into COURSE values(11,'Programming',10000,'Ravi');**

**insert into COURSE values(12,'C#',15000,'Jiban');**

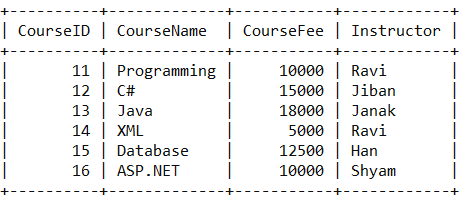
**insert into COURSE values(13,'Java',18000,'Janak');**

**insert into COURSE values(14,'XML',5000,'Ravi');**

**insert into COURSE values(15,'Database',12500,'Han');**

**insert into COURSE values(16,'ASP.NET',10000,'Shyam');**

**Output:**

****

1. Write SQL syntax to update the instructor to Ramesh whose CourseID is 12.

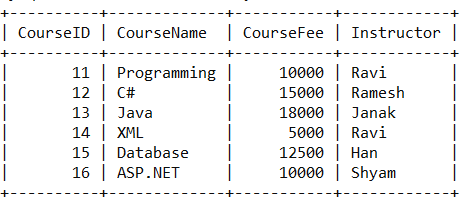
***Solution:***

**update COURSE**

**set Instructor='Ramesh'**

**where CourseID=12;**

Output:

****

1. Write SQL query to retrieve all information of courses that have more than one instructor.

***Solution:***

select count(instructor), instructor from Course

group by instructor

having count(instructor)>1;

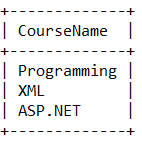
1. Write SQL query to find the name of course whose fee is less than the average fee of all the courses.

***Solution:***

**select CourseName from COURSE**

**where CourseFee<(Select avg(CourseFee) from COURSE);**

Output:

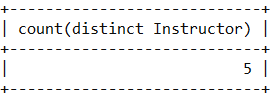
****

1. Write SQL query to count distinct number of instructors in the course table.

***Solution:***

**select count(distinct Instructor) from COURSE;**

Output:

****

Q2. Consider the following relation and attributes

**PRODUCT**

PID Varchar(5)  
ProductName Varchar(40)

Unit Price number(5)

1. Develop DDL in SQL to implement above schema.

***Solution:***

create table PRODUCT

(

PID Varchar(5),

ProductName Varchar(40),

UnitPrice numeric(5)

);

1. Develop SQL Queries to insert a new product named Smartphone with PID 12345 and price of 25000.

***Solution:***

insert into PRODUCT(PID,ProductName,UnitPrice)

values(12345,'Smartphone',25000);

1. Develop SQL queries to list product with unit price greater than 200.

***Solution:***

select \* from Product

where UNITPRICE>200

1. Develop SQL queries to list products sorted by the “ProductName” column.

***Solution:***

select \* from Product

order by ProductName;

1. Develop SQL queries to list details of product whose price I sgreater than the average price of all products.

***Solution:***

select \* from PRODUCT

where UNITPRICE>(select avg(unitprice) from PRODUCT);

1. Develop SQL queries to delete all rows in a table without deleting the table.

***Solution:***

delete from Product;

1. Develop SQL queries to delete the table named product from the database.

***Solution:***

**drop table PRODUCT;**

**Q3. Consider the relational database where the primary keys are highlighted. Give an expression in SQL for each of the following queries:**

Employee(**person\_name**, street, city)

Works(**person\_name**, company\_name, salary)

Company(**company\_name**, city)

Manages(**person\_name**,manager\_name)

1. Implement DDL for the given relation.

***Solution:***

create table Employee

(

person\_name varchar(30) primary key,

street varchar (50),

city varchar(30)

);

create table Works

(

person\_name varchar(30) primary key,

company\_name varchar (50),

salary numeric

);

create table Company

(

company\_name varchar(50) primary key,

city varchar(50)

);

create table Manages

(

person\_name varchar(30) primary key,

manager\_name varchar(30)

);

1. Find the names of all employees who work for the First Bank Corporation.

***Solution:***

select person\_name from WORKS where

company\_name='First Bank Corporation';

1. Find the names of all employees who live in the same city and on the same street as do their managers.

***Solution:***

Select E1.person\_name

From Employee as E1, Employee as E2, Manages as M

Where E1.person\_name=M.person\_name and E2.person\_name=M.manager\_name

and E1.stree=E2.street and E1.city=E2.city

1. Find the names, street address and cities of residence of all employees who work for First Bank Corporation and earn more than $10,000 per annum.

***Solution:***

select \*from

Employee

inner join WORKS

on Employee.person\_name=Works.PERSON\_NAME

where Works.COMPANY\_NAME='First Bank Corporation' and Works.salary>10000;

1. Give all employees of First Bank Corporation a 10 percent salary raise.

***Solution:***

update WORKS SET SALARY=1.1\*SALARY

WHERE COMPANY\_NAME='First Bank Corporation'

1. Delete all the tuples in the works relation for employees of Small Bank Corporation

***Solution:***

delete from Works where COMPANY\_NAME='Small Bank Corporation';

Q4. Create a student table with following schema

STUDENT(name, roll, marks, address);

1. Write SQL query to create the table.

*Solution:*

create table student

(

name varchar(20),

roll integer primary key,

marks integer,

address varchar(50)

);

1. Write SQL queries to populate the table with 10 records.

*Solution:*

insert into student values('Ram',12,98,'Palpa');

insert into student values('Shyam',13,99,'KTM');

insert into student values('Hari',14,88,'PKR');

insert into student values('Rita',15,57,'BRT');

insert into student values('Sita',16,66,'BKT');

insert into student values('Gita',17,29,'KTM');

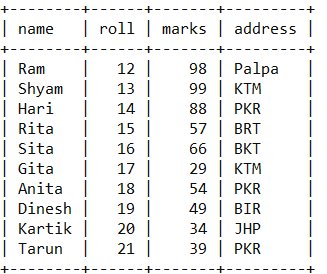
insert into student values('Anita',18,54,'PKR');

insert into student values('Dinesh',19,49,'BIR');

insert into student values('Kartik',20,34,'JHP');

insert into student values('Tarun',21,39,'PKR');

After the execution of above commands the state of the database is:



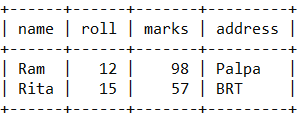
1. Write SQL queries to list the details of all the student whose name starts with ‘R’

*Solution:*

select \* from student

where name like 'R%';

Output:



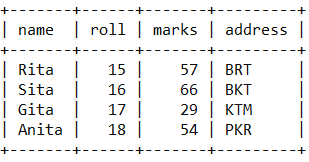
1. Write SQL queries to display the details of all the student whose name ends with ‘ita’

*Solution:*

select \* from student

where name like '%ita';

Output:

****

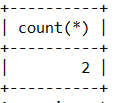
1. Write SQL queries to count the no. of students whose name starts with ‘R’

*Solution:*

select count(\*) from student

where name like 'R%';

**Output:**

****

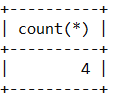
1. Write SQL queries to count the no. of students whose name ends with ‘ita’

*Solution:*

select count(\*) from student

where name like '%ita';

Output:

****

**For Questions 2 and 3 generate the output yourself.**

Lab 6

**SQL Queries Set 4**

In this lab, students are expected to learn queries related to cartesian product, join( natural join, theta join, equi join, left outer join, right outer join, full outer join etc), and set operations(such as union, intersection and difference).

**Q1. Consider following tables:**



1. **Create the schema for the tables Students and Courses.**

**Solution:**

Create table Students(

studno integer primary key,

name varchar(50),

course varchar(4)

);

create table Courses(

courseno varchar(4) primary key,

name varchar(50)

);

1. **Populate the tables with above indicated values.**

**Solution:**

insert into Students values(100,'Fred','PH');

insert into Students values(200,'Dave','CM');

insert into Students values(300,'Bob','CM');

insert into Courses values('PH','Pharmacy');

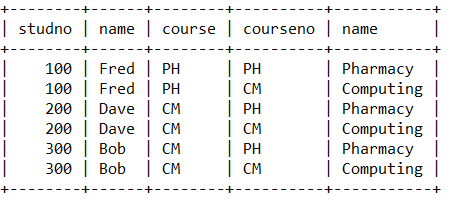
insert into Courses values('CM','Computing');

1. **Write SQL query to display the Cartesian product of two tables.**

**Solution:**

select \* from Students, Courses;

**Output:**

****

1. **Write SQL query to display the result of the theta join operation**

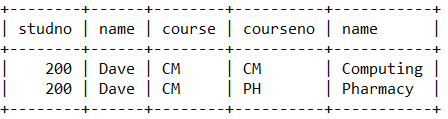
*Students* **⋈stud#=200 Courses**

**Solution:**

select \* from Students, Courses

where studno = 200;

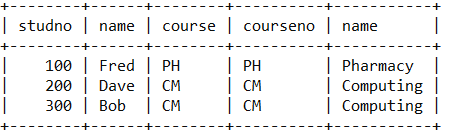
Output:



1. **Write SQL query to display the result of the equi join operation**

*Students* **⋈course=course# Courses**

**Solution:**

****

**Q2. Consider following tables:**



1. **Write SQL query to create the schemas for tables *r* and *s* and to populate the indicated values.**

**Solution:**

create table r(

a varchar(4),

b varchar(4)

);

create table s(

b varchar(4),

c varchar(4)

);

insert into r values('a1','b1');

insert into r values('a2','b2');

insert into r values('a3','b3');

insert into s values('b1','c1');

insert into s values('b2','c2');

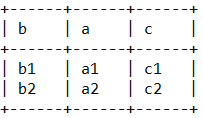
insert into s values('b4','c4');

1. **Write SQL query to display the result of the natural join operation r ⋈ s**

**Solution:**

select \* from r natural join s;

Output:

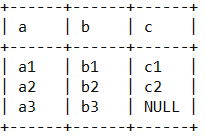


1. **Write SQL query to display the result of the left outer join operation r ⟕ s**

***Solution:***

select a,r.b,s.c from r left join s on r.b=s.b;

**Output:**

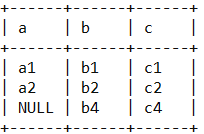
****

1. **Write SQL query to display the result of the right outer join operation r ⟖ s**

***Solution:***

select a,s.b,c from r right join s on r.b=s.b;

Output:



1. **Write SQL query to display the result of the full outer join operation r ⟗ s**

***Solution:***

create view g as

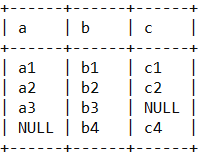
(select a,r.b,s.c from r left join s on r.b=s.b);

create view h as

(select a,s.b,c from r right join s on r.b=s.b);

select \* from g union select \* from h;

Output:



**Q3. Consider following tables:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***First*** | |  | ***Second*** | |
| *id* | *name* |  | *id* | *Name* |
| 1 | A |  | 2 | B |
| 2 | B |  | 3 | C |
| 3 | C |  | 5 | E |
| 4 | D |  | 6 | F |

1. **Write SQL query to create schema for the tables First and Second.**

***Solution:***

create table First(

id integer,

name varchar(4)

);

create table Second(

id integer,

name varchar(4)

);

1. **Write SQL query to populate the indicated values.**

**Solution:**

insert into First values(1,'A');

insert into First values(2,'B');

insert into First values(3,'C');

insert into First values(4,'D');

insert into Second values(2,'B');

insert into Second values(3,'C');

insert into Second values(5,'E');

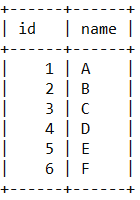
insert into Second values(6,'F');

1. **Write SQL query to find the union of two tables.**

***Solution:***

select \* from First union Select \* from Second;

**Output:**

****

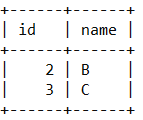
1. **Write SQL query to find the intersection of the two tables.**

***Solution:***

select \* from First intersect Select \* from Second;

or alternatively following query can be written to generate same output:  
select \* from First where id in(Select id from Second);

**Output:**

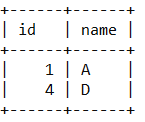
****

1. **Write SQL query to find the difference of the two tables.**

***Solution:***

select \* from First where id not in(Select id from Second);

**Output:**

****

Lab 7

**SQL Queries Set 5**

Design at least five SQL questions related to SQL clauses such as *group by*, *having*, *as*, *exists*, *some*, *all* etc.

Design one SQL question related to creating a view, displaying records from a view, and dropping a view.

Lab 8

**Relational Database Design using ER diagram**

In this lab, students are expected to learn how to draw ER diagram using draw.io

Following are the tasks covered in this lab:

Task #1: Draw an ER diagram for **COMPANY** database.

Task #2: Draw an ER diagram for **MOVIE** database.

Task #3: Draw an ER diagram for **AIRLINE RESERVATION SYSTEM** database.

Task #4: Draw an ER diagram for **HOSPITAL MANAGEMENT SYSTEM** database.

Task #5: Draw an ER diagram for **LIBRARY MANAGEMENT SYSTEM** database.

Task #6: Draw an ER diagram for **UNIVERSITY** database.

Task #7: Draw an ER diagram for **BANK** database.

Task #8: Draw an ER diagram of any system involving **Specialization and Generalization**.

**It is to be noted that your image must be in vector format.**

Lab 9

**Mini Project**

In this lab, students are expected to design ER diagram for any system other than that covered in Lab 8. Then the concept of mapping ER diagram components to relational model must be applied. Finally draw a detailed schema diagram of the design outlining key attributes and foreign key constraints.