LABORATORY MANUAL

OF

DBMS AND MySQL

(IT 311)

Department Of Information Technology,
Gauhati University

Submitted By- Riyanka Chakraborty

Roll Number-180103009

Branch- CSE

INDEX

Date of expt: 12/11/2020

EXPERIMENT NO: 1 Introduction to MYSQL

AIM:To study about Mysql database

THEORY:

MySQL, is one of the most popular Open Source SQL database management systems.

MySQL is a fast, easy-to-use RDBMS being used for many small and big businesses.

MySQL is developed, marketed, and supported by MySQL AB, which is a Swedish company.

MySQL is becoming so popular because of many good reasons:

- MySQL is released under an open-source license. So you have nothing to pay to use it.
- MySQL is a very powerful program in its own right. It handles a large subset of the functionality of the most expensive and powerful database packages.
- MySQL uses a standard form of the well-known SQL data language.
- MySQL works on many operating systems and with many languages including PHP, PERL, C, C++, JAVA, etc.
- MySQL works very quickly and works well even with large data sets.
- MySQL is very friendly to PHP, the most appreciated language for web development.

- MySQL supports large databases, up to 50 million rows or more in a table.
- MySQL is customizable.

RESULT: The mysql database is studied.

Date of expt: 19/11/2020

EXPERIMENT NO: 2

DATA DEFINITION LANGUAGE (DDL) COMMANDS

AIM:

Consider the database for an organisation. Write the queries for the following

(i) create the database
(ii) select the current database
(iii) Create the following tables.

a) employee (emp_no,emp_name,DOB, address, doj, mobile_no, dept_no, salary).

b) department (dept_no, dept_name, location).

(iv) Include necessary constraints.
(v) List all the tables in the current database
(vi) Display the structure of the employee table
(vii) Add a new column Designation to the employee table.
(viii) Drop the column location from Dept table
(ix) Drop the tables
(x) Delete the database

OBJECTIVES: To understand the DDL commands

DATABASE QUERIES:

Before creating any tables, MySQL requires you to create a database by executing the CREATE DATABASE command.

Create a database

CREATE DATABASE <database name>;

Delete a database

DROP DATABASE <database name>;

Select the database

USE <database name>;

List all databases

SHOW databases;

Rename a database

ALTER DATABASE <database name> RENAME <new database name>

TABLE QUERIES:

To Create a table

CREATE TABLE (<fieldname>< fieldtype>(<fieldsize) , …) ;

List all tables in the current database

SHOW tables;

Show table format with column names and data types

DESCRIBE <table name>;

Modify the structure of table

ALTER TABLE <table name><alter specifications>;

ALTER TABLE < table name> DROP COLUMN <column name>;

ALTER TABLE <table name> ADD COLUMN<column name><datatype>(<Size>);

Delete the table

DROP TABLE <tablename>

CONSTRAINTS:

Primary key A PRIMARY KEY constraint for a table enforces the table to accept unique data for a specific column and this constraint create a unique index for accessing the table faster
UNIQUE The UNIQUE constraint in Mysql does not allow to insert a duplicate value in a column.

NOT NULL In Mysql NOT NULL constraint allows to specify that a column can not contain any NULL value.

FOREIGN KEY A FOREIGN KEY in mysql creates a link between two tables by one specific column of both table. The specified column in one table must be a PRIMARY KEY and referred by the column of another table known as FOREIGN KEY.

CHECK The CHECK constraint determines whether the value is valid or not from a logical expression.

DEFAULT While inserting data into a table, if no value is supplied to a column, then the column gets the value set as DEFAULT

PROCEDURE:

(i) CREATE DATABASE command
(ii) USE DATABASE command
(iii) CREATE TABLE command
(iv) PRIMARY KEY,NOT NULL etc
(v) SHOW TABLES command
(vi) DESCRIBE TABLE command
(vii) ALTER TABLE command
(viii) ALTER TABLE command
(ix) DROP TABLE command
(x) DROP DATABASE command

Code:

Code for (i)-(iv)

```
(i)create database organization;

(ii)use organization;

(iii)
   (a)  create table Department(dept_no int(5) primary key
        not null unique,dept_name varchar(30) not
        null,location varchar(30) not null);

   (b)  create table Employee(emp_no int(5) primary key
        not null unique,emp_name varchar(30) not null,DOB
        date not null,doj date not null,mobile_no
        varchar(10) not null,dept_no int(5),foreign
        key(dept_no) references department(dept_no),salary
        float(10) not null);
```

(v)show tables;

```
13 •    show tables;
```

Result Grid | Filter Rows: | Export: | Wrap

Tables_in_organization
department
employee

(vi)desc Employee;

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
emp_no	int(5)	NO	PRI	NULL	
emp_name	varchar(30)	NO		NULL	
DOB	date	NO		NULL	
doj	date	NO		NULL	
mobile_no	varchar(10)	NO		NULL	
dept_no	int(5)	YES	MUL	NULL	
salary	float	NO		NULL	
Designation	varchar(20)	YES		NULL	

```
(vii) alter table employee add column Designation
varchar(20);

(viii) alter table department drop column location;

(ix) drop table employee,department;

(x)drop database organization;
```

PROBLEMS

1) Consider the database for a college and design an ER diagram. Write the query for the following.

(i)Create the tables:

Student (sid, sname, sex, dob,dno)

Department (dno, dname)

Faculty (F_id, fname, designation, salary,dno)

Course (cid, cname, credits,dno)

Register (sid,cid,sem)

Teaching (f_id,cid,sem)

Hostel(hid,hname,seats,)

(ii) Include the necessary constraints NOT NULL, DEFAULT, CHECK, and PRIMARY KEY, UNIQUE.

(iii)Create a database college.

(iv)Use college as the current database

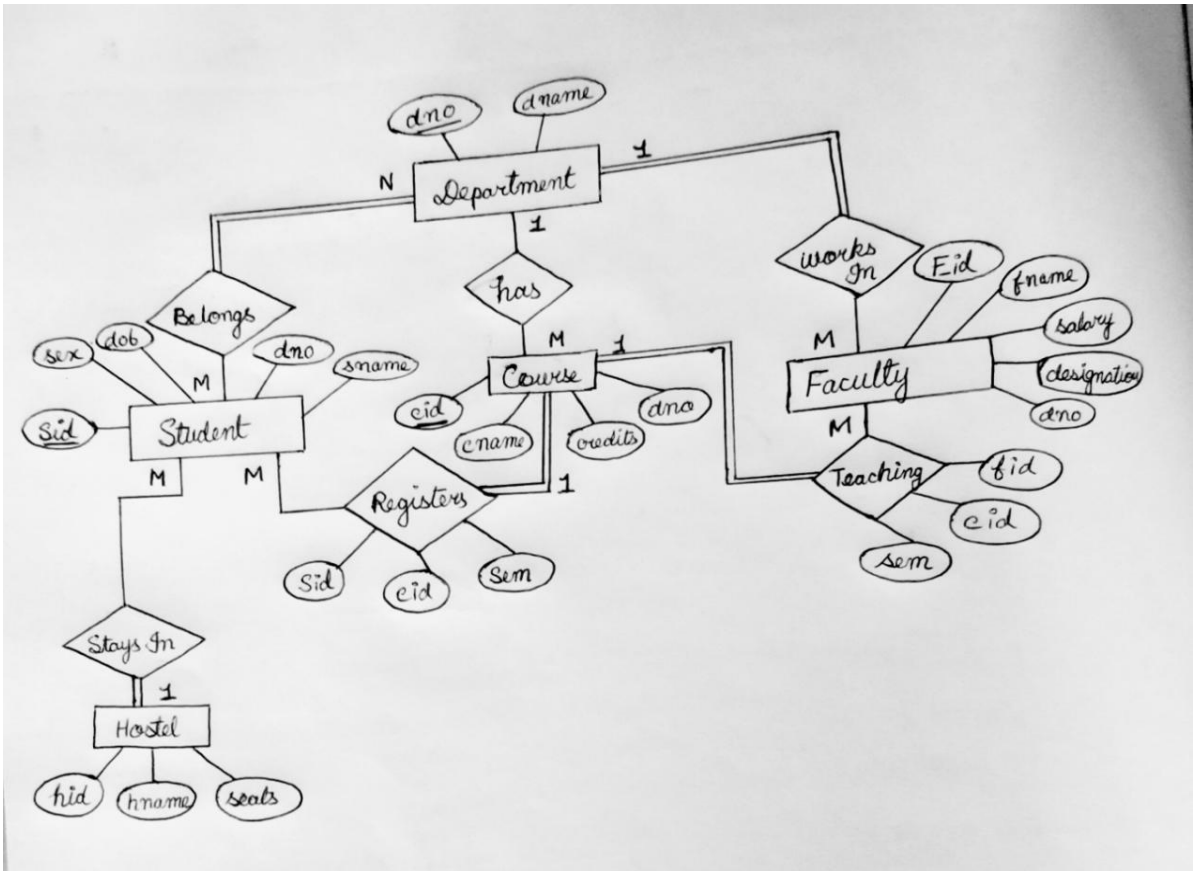
(v) Display all the tables in college database

(vi) Describe the structure of all tables

(vii) Modify the student table to add a new field 'grade'

Ans-

Er diagram



Answers of (i),(ii),(iii)

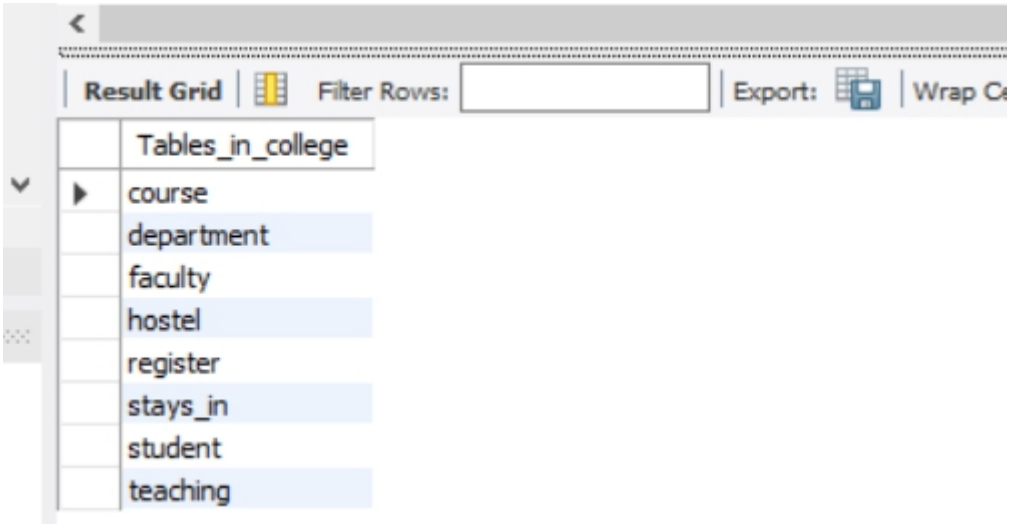
```
1
2     CREATE DATABASE College;
3     use college;
4
5     CREATE TABLE Department(dno int(10) NOT NULL primary key,
6     dname varchar(25) NOT NULL
7     );
8     CREATE TABLE Student(
9     sid integer(10) primary key NOT NULL unique,sname varchar(30) NOT NULL,sex char NOT NULL,dob date NOT NULL,
10    dno int(10) NOT NULL foreign key(dno) references department(dno)
11    );
12    CREATE TABLE Faculty(F_id int(10) Primary key unique,fname varchar(30),
13    designation varchar(20),salary int(10),dno int(10) NOT NULL,
14    FOREIGN KEY(dno) references department(dno)
15    );
16    CREATE TABLE Course(cid int(10) primary key,cname varchar(20),
17    credits int(3) NOT NULL,dno int(10) NOT NULL,
18    FOREIGN KEY(dno) references department(dno)
19    );
20    CREATE TABLE Register(sid int(10) primary key unique not null,
21    cid int(10) NOT NULL,FOREIGN KEY(cid) references Course(cid),sem int(2) not null
22    );
```

```
);

CREATE TABLE Teaching(f_id int(10) NOT NULL,
FOREIGN KEY(f_id) references Faculty(F_id),cid int(10) NOT NULL,
FOREIGN KEY(cid) references Course(cid),sem int(2) NOT NULL);
CREATE TABLE Hostel
(hid int(6) primary key,hname varchar(35) not null,seats int(20));
```

Answers of (iv),(v),(vi),(vii)

(iv)USE College;

(v)SHOW TABLES;

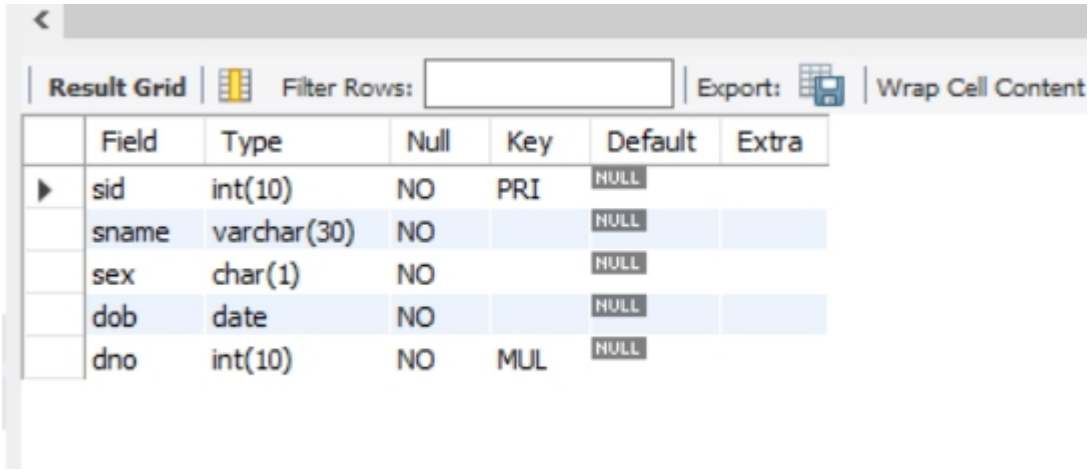


(vi)desc Student;
desc department;
desc Faculty;
desc Course;
desc Register;
desc Teaching;
desc Hostel;

(vii)ALTER TABLE student ADD COLUMN grade char not null;

OUTPUT:

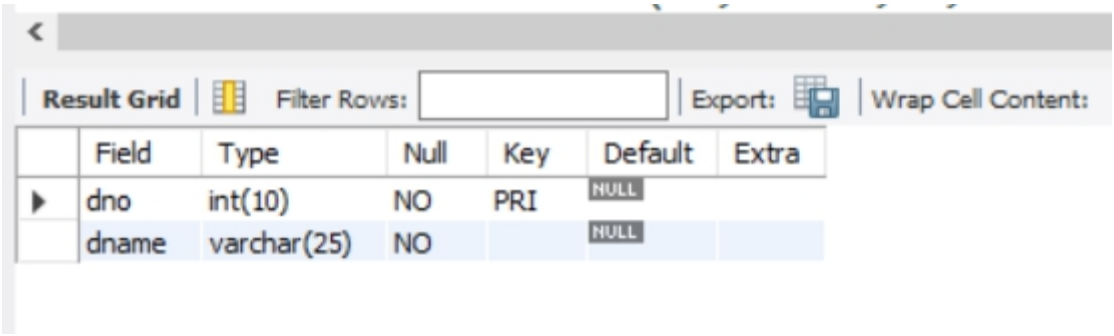
(vi)desc student



Result Grid | Filter Rows: | Export: | Wrap Cell Content

Field	Type	Null	Key	Default	Extra
sid	int(10)	NO	PRI	NULL	
sname	varchar(30)	NO		NULL	
sex	char(1)	NO		NULL	
dob	date	NO		NULL	
dno	int(10)	NO	MUL	NULL	

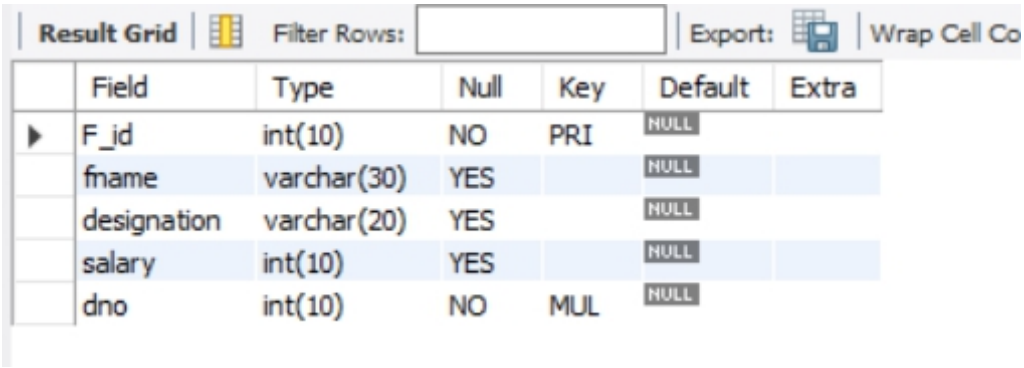
desc department;



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
dno	int(10)	NO	PRI	NULL	
dname	varchar(25)	NO		NULL	

desc faculty;



Result Grid | Filter Rows: | Export: | Wrap Cell Co

Field	Type	Null	Key	Default	Extra
F_id	int(10)	NO	PRI	NULL	
fname	varchar(30)	YES		NULL	
designation	varchar(20)	YES		NULL	
salary	int(10)	YES		NULL	
dno	int(10)	NO	MUL	NULL	

desc course;

Field	Type	Null	Key	Default	Extra
cid	int(10)	NO	PRI	NULL	
cname	varchar(20)	YES		NULL	
credits	int(3)	NO		NULL	
dno	int(10)	NO	MUL	NULL	

desc register;

Field	Type	Null	Key	Default	Extra
sid	int(10)	NO	MUL	NULL	
cid	int(10)	NO	MUL	NULL	
sem	int(2)	NO		NULL	

desc teaching;

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
f_id	int(10)	NO	MUL	NULL	
cid	int(10)	NO	MUL	NULL	
sem	int(2)	NO		NULL	

desc hostel;

Result Grid | Filter Rows: | Export: | Wrap Cell

Field	Type	Null	Key	Default	Extra
hid	int(6)	NO	PRI	NULL	
hname	varchar(35)	NO		NULL	
seats	int(20)	YES		NULL	

2)Consider the database for a banking enterprise. Write the queries for the below questions.

(i) Create the following tables

Table	Attributes
Customer	cid,cname,loc,sex,dob
Bank Brn	Bcode,bloc,bsate
Deposit	Sacno,dtype,ddate,damt
Loan	Lacno,ltype,ldate,lamt
Account in	Bcode,cid
Depositor	cid,dacno
Borrower	cid,lacno

(ii) Include necessary constraints.

(iii) Tables are created under the database 'bank'

(iv) Display all the tables in bank database

(v) Describe the structure of all tables

(vi) Delete tables

Answer:

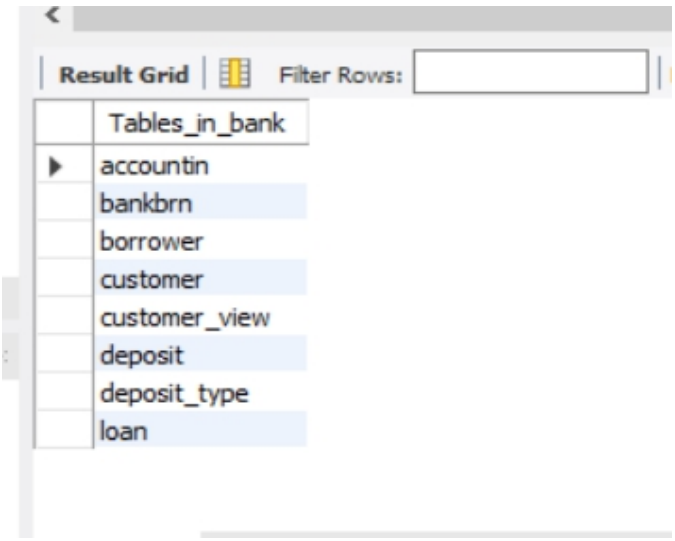
(i),(ii),(iii)

```
1  create database Bank;
2  use Bank;
3  create table Customer(cid int(5) primary key not null unique,cname varchar(40) not null,loc varchar(20),sex char not null,dob date not null)
4
5  create table BankBrn(Bcode int(4) primary key unique not null,bloc varchar(20) not null,bsate varchar(20));
6
7  create table Deposit(Dacno int(10) primary key unique not null,dtype varchar(20) not null,ddate date not null,damt int(6));
8
9  create table Loan(Lacno int(10),ltype varchar(20),ldate date,lamt int(6));
10 create table Accountin(Bcode int(4) NOT NULL,FOREIGN KEY(Bcode) references BankBrn(Bcode),cid int(5) not null);
11
12 create table Depositor(cid int(5) not null,FOREIGN KEY(cid) references Customer(cid),dacno int(10) not null);
13
14 create table borrower(cid int(5) not null,FOREIGN KEY(cid) references Customer(cid),lacno int(10) not null);
15
```

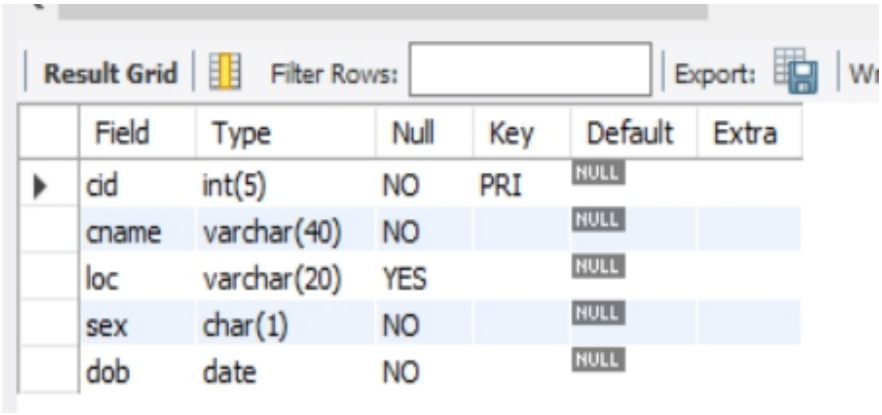
(iv) Code:

```
use Bank;
show tables;
```

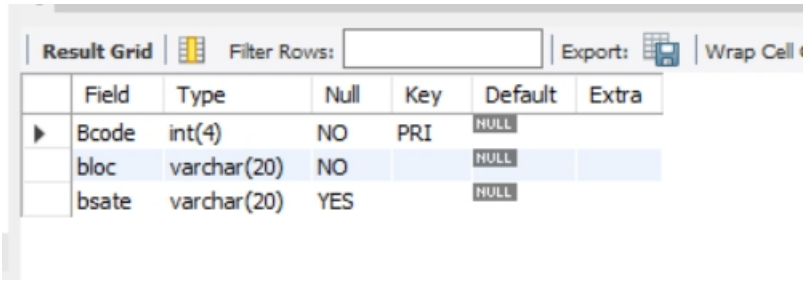
Output of (iv):



(v)desc customer;

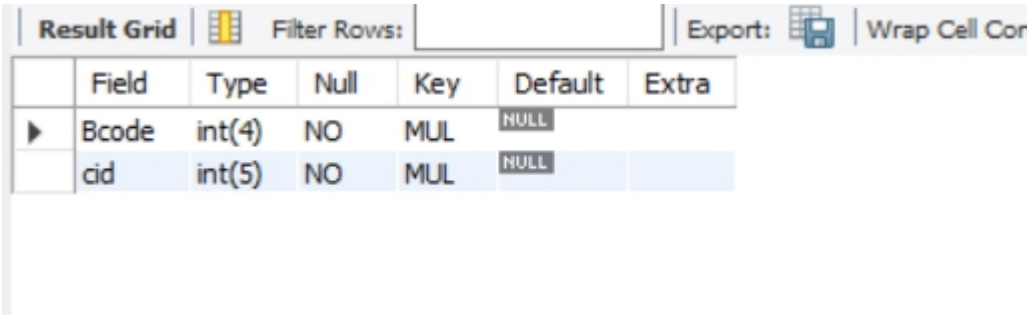


```
desc banknrn;
```



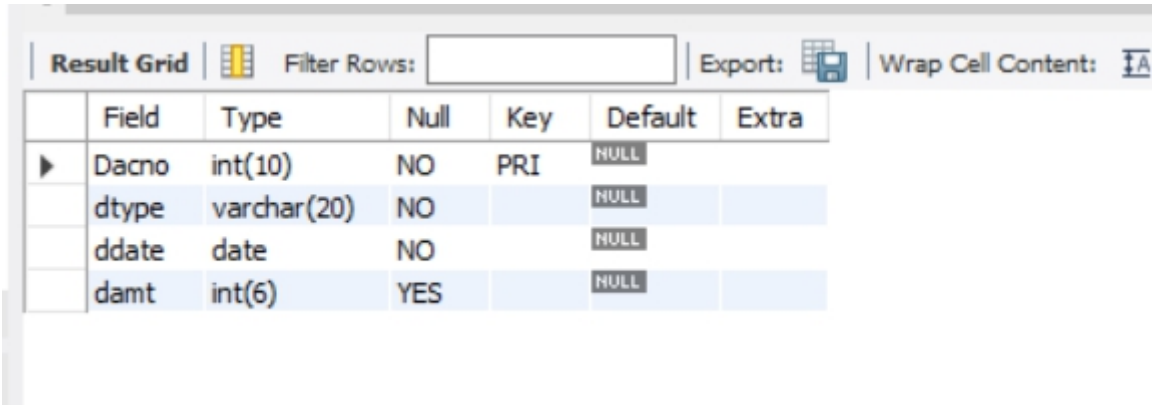
Field	Type	Null	Key	Default	Extra
Bcode	int(4)	NO	PRI	NULL	
bloc	varchar(20)	NO		NULL	
bsate	varchar(20)	YES		NULL	

```
desc accountin;
```



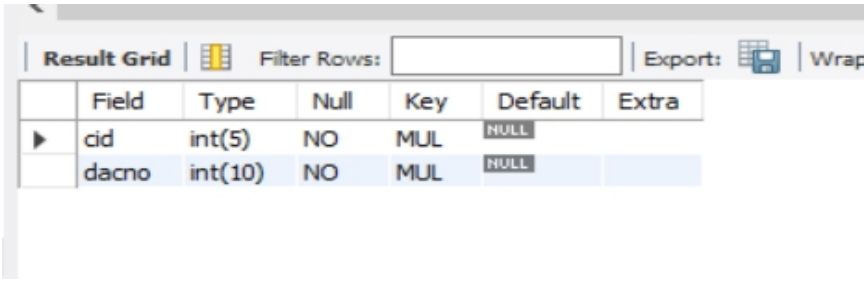
Field	Type	Null	Key	Default	Extra
Bcode	int(4)	NO	MUL	NULL	
cid	int(5)	NO	MUL	NULL	

desc Deposit;



Field	Type	Null	Key	Default	Extra
Dacno	int(10)	NO	PRI	NULL	
dtype	varchar(20)	NO		NULL	
ddate	date	NO		NULL	
damt	int(6)	YES		NULL	

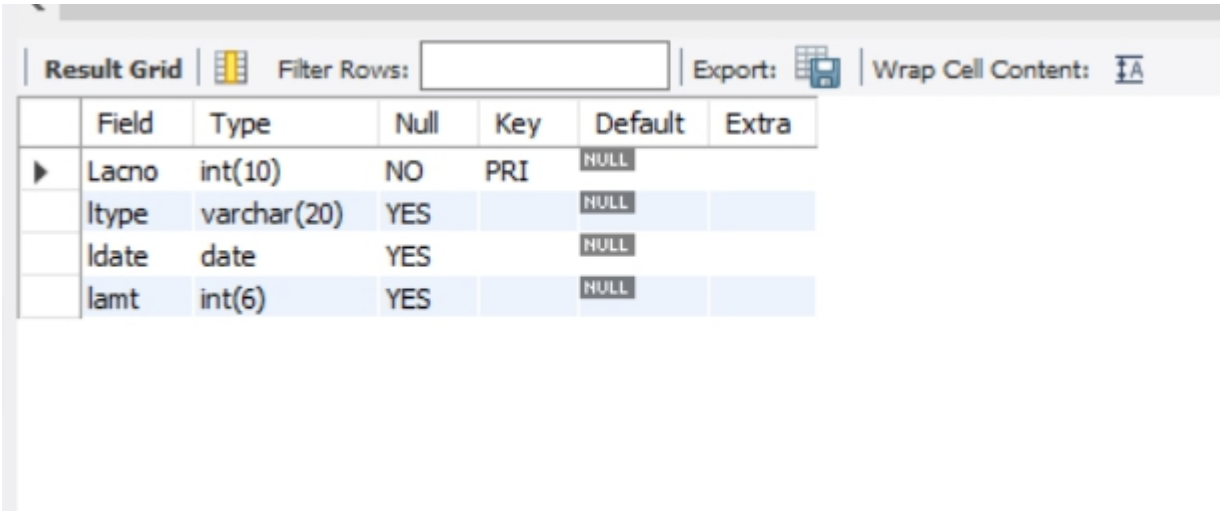
desc depositor;



Result Grid | Filter Rows: | Export: | Wrap

Field	Type	Null	Key	Default	Extra
cid	int(5)	NO	MUL	NULL	
dacno	int(10)	NO	MUL	NULL	

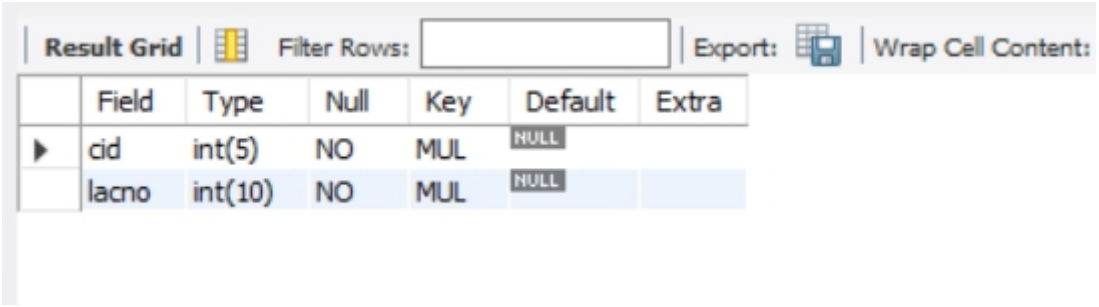
desc loan;



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
Lacno	int(10)	NO	PRI	NULL	
ltype	varchar(20)	YES		NULL	
ldate	date	YES		NULL	
lamt	int(6)	YES		NULL	

desc borrower



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
cid	int(5)	NO	MUL	NULL	
lacno	int(10)	NO	MUL	NULL	

(vi)

Drop table bankbrn;
Drop table customer;
Drop table accountin;
Drop table deposit;

Drop table depositor;
Drop table borrower;
Drop table loan;

Mac address-> 8C-16-45-E7-D6-DB

Date:26/11/20 **EXPERIMENT NO 3**

DATA MANIPULATION LANGUAGE(DML)

AIM:

Consider the database for an organization. Write the queries for the following

(i)Add 5 rows in the employee and dept tables
(ii) Display all the records from the above tables
(iii)Display the empno and name of all the employees from department no2.
(iv)Display empno,name,designation,dept no and salary in the descending order of salary.
(v)Display the empno and name of all employees whose salary is between 2000 and 5000.
(vi) Display all designations without duplicate values.
(vii)Display the dept name and total salary of employees of each department.
(viii)Change the salary of employees to 25000 whose designation is 'Typist'
(ix) Change the mobile no of employee named 'john'
(x) Delete all employees whose salaries are equal to Rs.7000
(xi) Select the department that has total salary paid for its employees more than 25000

OBJECTIVES:

To understand how to insert, update and delete data from within a table.
To learn how to retrieve data from a table using the SELECT statement.

THEORY

1. INSERT
   INSERT INTO *tablename* VALUES (*value1, value2, ..., valuen*).
2. UPDATE
   UPDATE <table> SET <field1> = <value1> AND <field2> = <value2> WHERE <conditions>
3. DELETE
   DELETE FROM <table> WHERE <condition>
4. SELECT
   a) Retrieve from all columns
      SELECT * FROM <table>
   b) Retrieve from selected columns
      SELECT <column 1>, <column 2> FROM <table>
   c) Retrieve unique values
      SELECT DISTINCT <column name> FROM <table>

d) Retrieve data satisfying a given condition

```
SELECT <columns> FROM <tables> WHERE <condition>
```

PROCEDURE :

(i) Use insert command

(ii) Use Select command

(iii) Use Select command with where condition

(iv) Use Select command with order by clause

(v) Use Select command with operators

(vi) Use Select command with DISTINCT keyword

(vii) Use Select command with group by clause

(viii) Use Update command

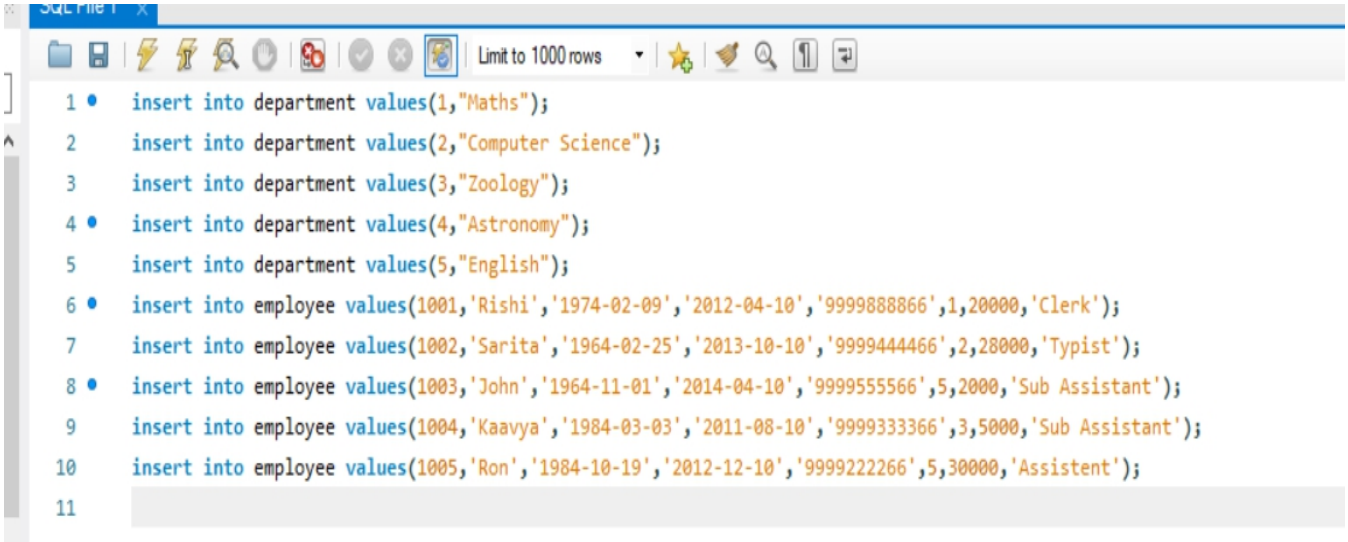
(ix) Use Update command

(x) Use Delete command

(xi) Use select command with group by and having clause

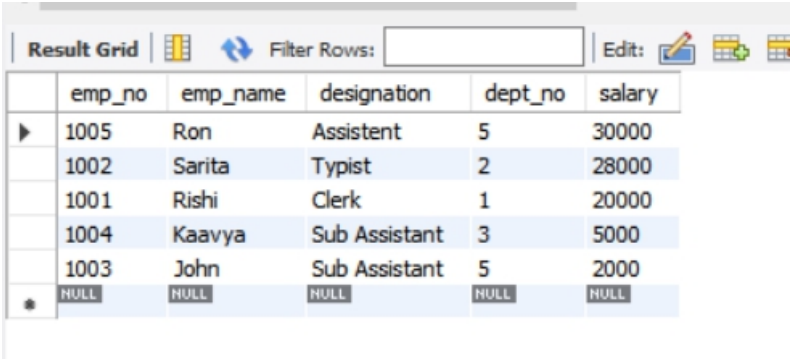
CODE :

(i)

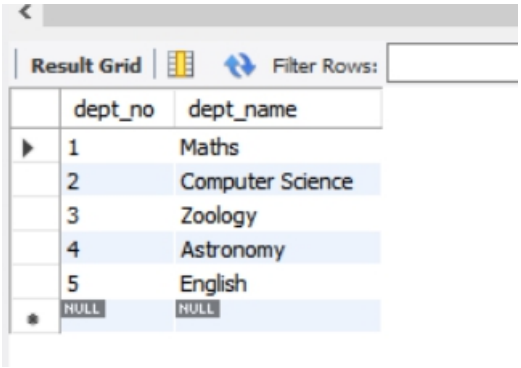


(ii) select * from employee;

select * from department;



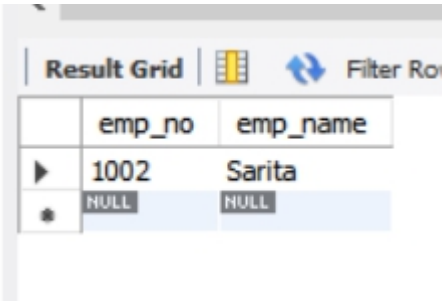
emp_no	emp_name	designation	dept_no	salary
1005	Ron	Assistent	5	30000
1002	Sarita	Typist	2	28000
1001	Rishi	Clerk	1	20000
1004	Kaavya	Sub Assistant	3	5000
1003	John	Sub Assistant	5	2000
NULL	NULL	NULL	NULL	NULL



dept_no	dept_name
1	Maths
2	Computer Science
3	Zoology
4	Astronomy
5	English
NULL	NULL

(iii)

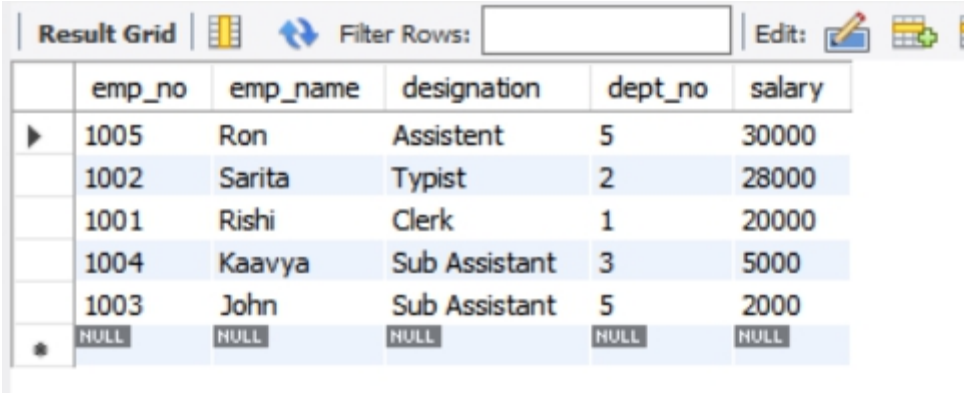
select emp_no,emp_name from Employee where dept_no=2;



emp_no	emp_name
1002	Sarita
NULL	NULL

(iv)

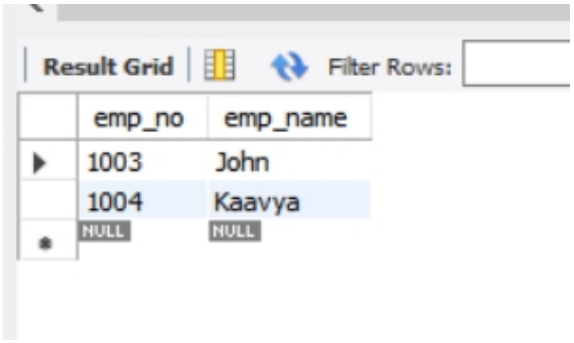
Select emp_no,emp_name,designation,dept_no,salary from Employee order by salary desc;



emp_no	emp_name	designation	dept_no	salary
1005	Ron	Assistent	5	30000
1002	Sarita	Typist	2	28000
1001	Rishi	Clerk	1	20000
1004	Kaavya	Sub Assistant	3	5000
1003	John	Sub Assistant	5	2000
NULL	NULL	NULL	NULL	NULL

(v)

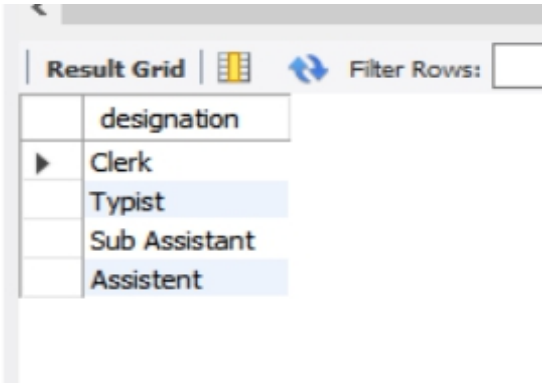
select emp_no,emp_name from Employee where salary between 2000 and 5000;



emp_no	emp_name
1003	John
1004	Kaavya
NULL	NULL

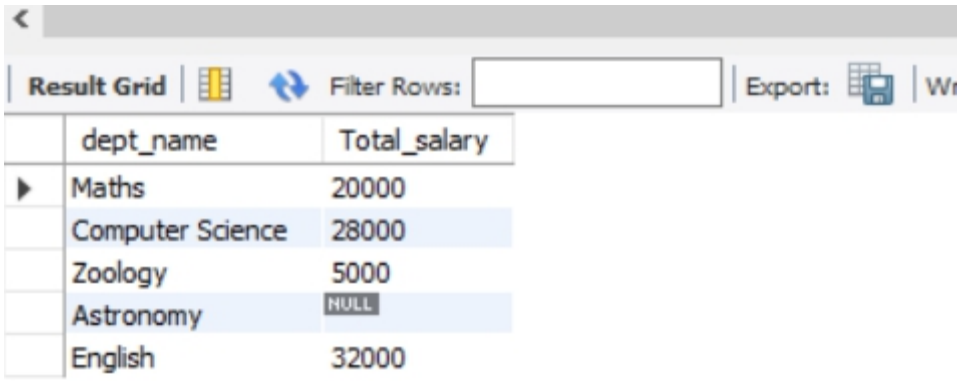
(vi)

select distinct designation from employee;



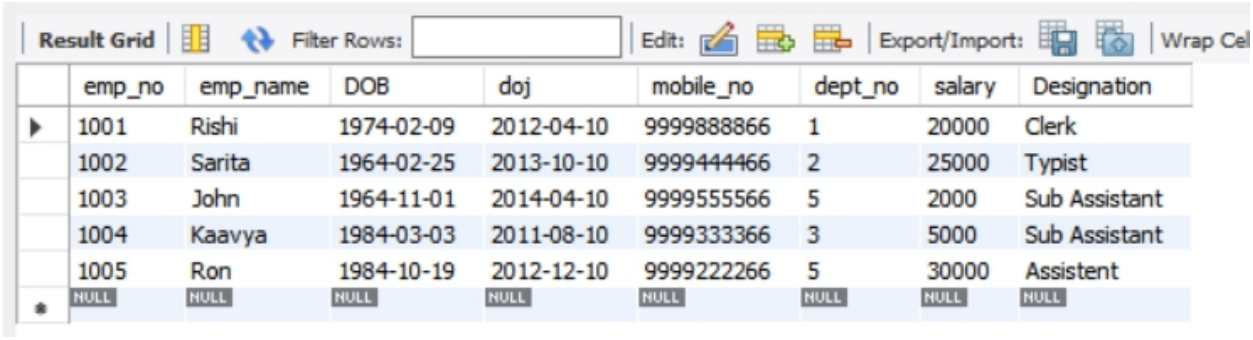
designation
Clerk
Typist
Sub Assistant
Assistent

(vii)select department.dept_name,sum(employee.salary) as 'Total_salary' from department left outer join employee on department.dept_no=employee.dept_no group by department.dept_no;



dept_name	Total_salary
Maths	20000
Computer Science	28000
Zoology	5000
Astronomy	NULL
English	32000

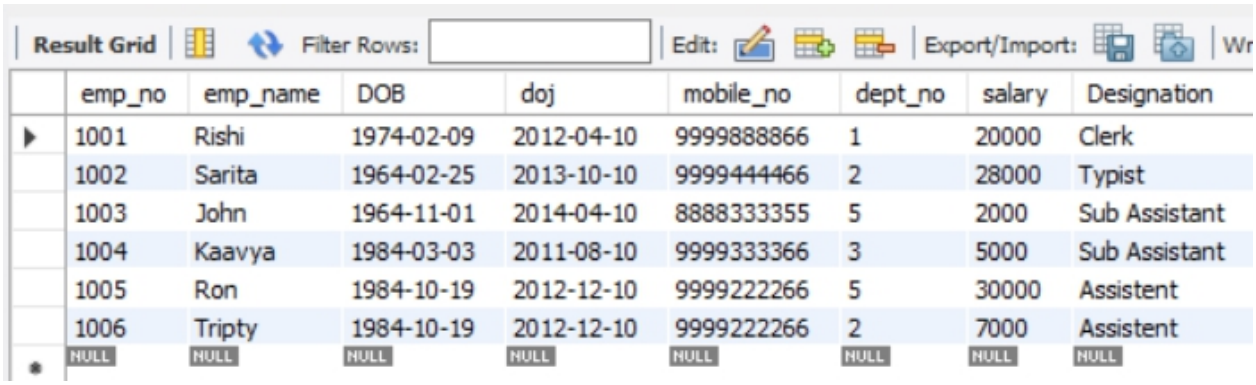
(viii)update employee set salary=25000 where employee.designation='Typist';



Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cel

emp_no	emp_name	DOB	doj	mobile_no	dept_no	salary	Designation
1001	Rishi	1974-02-09	2012-04-10	9999888866	1	20000	Clerk
1002	Sarita	1964-02-25	2013-10-10	9999444466	2	25000	Typist
1003	John	1964-11-01	2014-04-10	9999555566	5	2000	Sub Assistant
1004	Kaavya	1984-03-03	2011-08-10	9999333366	3	5000	Sub Assistant
1005	Ron	1984-10-19	2012-12-10	9999222266	5	30000	Assistent
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

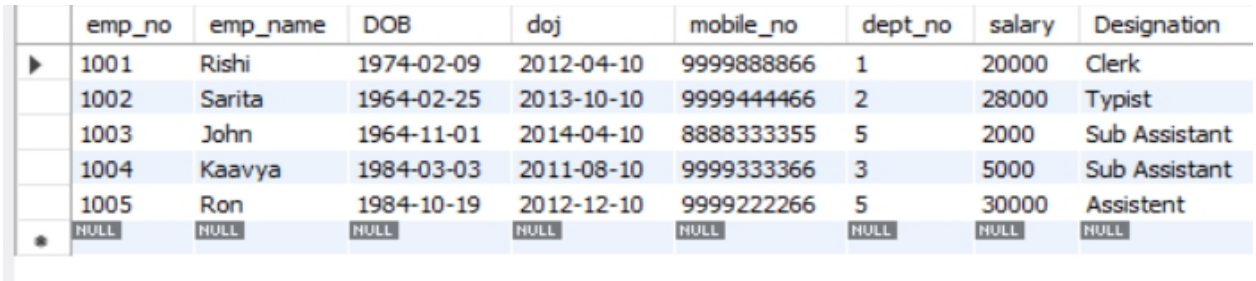
(ix) update employee set mobile_no="8888333355" where emp_name='John';



Result Grid | Filter Rows: | Edit: | Export/Import: | Wr

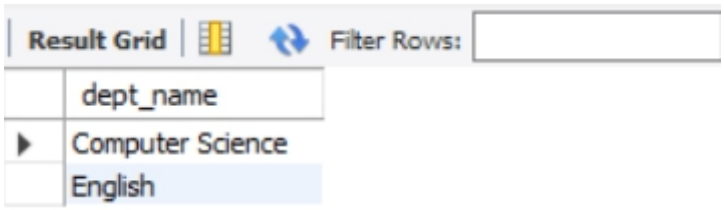
emp_no	emp_name	DOB	doj	mobile_no	dept_no	salary	Designation
1001	Rishi	1974-02-09	2012-04-10	9999888866	1	20000	Clerk
1002	Sarita	1964-02-25	2013-10-10	9999444466	2	28000	Typist
1003	John	1964-11-01	2014-04-10	8888333355	5	2000	Sub Assistant
1004	Kaavya	1984-03-03	2011-08-10	9999333366	3	5000	Sub Assistant
1005	Ron	1984-10-19	2012-12-10	9999222266	5	30000	Assistent
1006	Tripty	1984-10-19	2012-12-10	9999222266	2	7000	Assistent
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

(x)delete from employee where salary=7000;



emp_no	emp_name	DOB	doj	mobile_no	dept_no	salary	Designation
1001	Rishi	1974-02-09	2012-04-10	9999888866	1	20000	Clerk
1002	Sarita	1964-02-25	2013-10-10	9999444466	2	28000	Typist
1003	John	1964-11-01	2014-04-10	8888333355	5	2000	Sub Assistant
1004	Kaavya	1984-03-03	2011-08-10	9999333366	3	5000	Sub Assistant
1005	Ron	1984-10-19	2012-12-10	9999222266	5	30000	Assistent
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

(xi)select dept_name from department,employee where salary>25000;



RESULT:

The DML commands are executed successfully.

Problems

1. Consider the database for a college. Write the query for the following.

(i)Insert at least 5 tuples into each table.

(ii)List the details of students in the ascending order

of date of birth

(iii) Display the details of students from computer

department

(iv)List the faculties in the descending order of salary

(v)Display the total number of students in each department

(vi)Display the total number of faculties in each department with salary greater than 2500

Ans-

(i)

```
insert into department values(1,"Mathematics");
```

```
insert into department values(2,"Science");
insert into department values(3,"English");
insert into department values(4,"Biology");
insert into department values(5,"Zoology");
```

```
insert into student values(101,"Rishab","M","1969-12-28",1);
insert into student values(102,"Sonam","F","1972-11-21",2);
insert into student values(103,"Zoya","F","1975-02-18",2);
insert into student values(104,"Shraddha","F","1975-11-08",3);
insert into student values(105,"Abhinav","M","1964-02-17",5);
```

```
insert into faculty values(201,"Rajdeep","HOD",50000,1);
insert into faculty values(202,"Ramesh","Faculty Advisor",35000,2);
insert into faculty values(203,"Tara","Assistant professor",45000,2);
insert into faculty values(204,"Sourabh","Assistant
professor",12000,3);
insert into faculty values(205,"Abhi","Treasurer",37000,5);
```

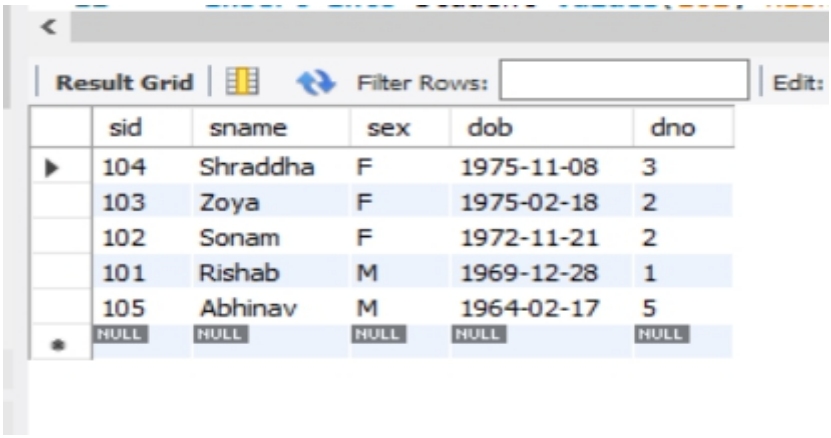
```
insert into course values(401,"Bsc",12,1);
insert into course values(402,"BCA",11,2);
insert into course values(403,"BSc",12,3);
insert into course values(404,"BTech",13,4);
insert into course values(405,"BTech",13,4);
```

```
insert into teaching values(201,401,2);
insert into teaching values(202,402,2);
insert into teaching values(203,404,3);
insert into teaching values(204,404,3);
insert into teaching values(205,405,5);
```

```
insert into register values(101,401,2);
insert into register values(102,402,2);
insert into register values(103,402,2);
insert into register values(104,403,2);
insert into register values(105,404,2);
```

```
insert into hostel values(301,"RCC",500);
insert into hostel values(302,"AT-8",200);
insert into hostel values(303,"RCC-5",300);
insert into hostel values(304,"RCC-2",400);
insert into hostel values(305,"AT-8",500);
```

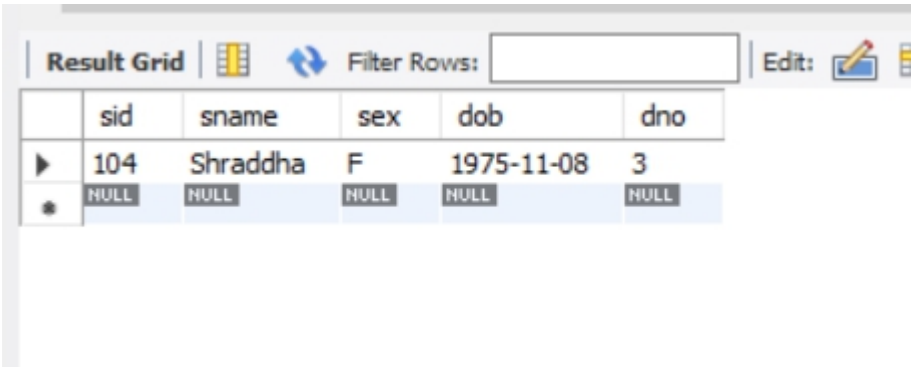
(ii)select * from student order by dob desc;



Result Grid | Filter Rows: | Edit:

sid	sname	sex	dob	dno
104	Shraddha	F	1975-11-08	3
103	Zoya	F	1975-02-18	2
102	Sonam	F	1972-11-21	2
101	Rishab	M	1969-12-28	1
105	Abhinav	M	1964-02-17	5
NULL	NULL	NULL	NULL	NULL

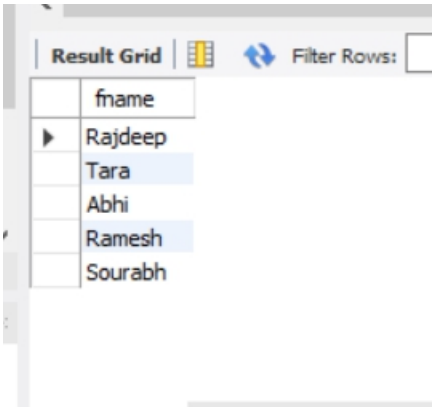
(iii)select * from student join department on department.dno=student.dno having dname="Computer";



Result Grid | Filter Rows: | Edit:

sid	sname	sex	dob	dno
104	Shraddha	F	1975-11-08	3
NULL	NULL	NULL	NULL	NULL

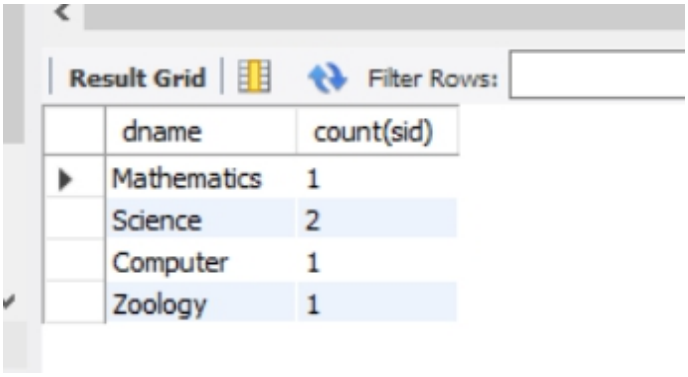
(iv)select fname from faculty order by salary desc;



Result Grid | Filter Rows:

fname
Rajdeep
Tara
Abhi
Ramesh
Sourabh

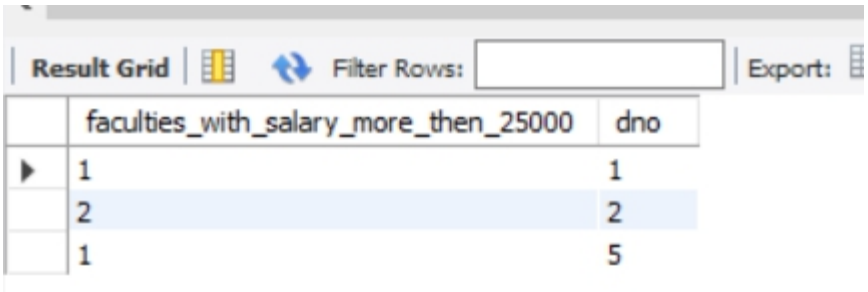
(v) select dname,count(sid) from student join department on student.dno=department.dno group by dname;



Result Grid | Filter Rows:

dname	count(sid)
Mathematics	1
Science	2
Computer	1
Zoology	1

(vi)select count(F_id) as faculties_with_salary_more_then_25000,dno from faculty where salary>25000 group by dno;



Result Grid | Filter Rows: | Export:

faculties_with_salary_more_then_25000	dno
1	1
2	2
1	5

2. Consider the database for a banking enterprise. Write the queries for the below questions.

(i)Insert at least 5 tuples in each table

(ii)Display the branch details

(iii)List the customers of ‘Mumbai’ city

(iv)List the male customers of ‘Kolkata’city

(v)List the state having more than one branch.

(vi)List the deposit schemes provided by the bank to

the customers

(vii) Delete the entire content of any table

Ans-
(i)

```
insert into customer values(101,'Rishi','Ring Road',"M",'1974-02-09');
insert into customer values(102,'Rumi','Park road',"F",'1964-02-09');
insert into customer values(103,'Shayan','Zoo Road',"M",'1972-12-09');
insert into customer values(104,'Tara','Park Road',"F",'1974-10-19');
insert into customer values(105,'Aisha','Ring Road',"F",'1973-02-09');
```

```
insert into bankbrn values(201,"Saharanpur","UP");
insert into bankbrn values(202,"Digboi","Assam");
insert into bankbrn values(203,"Guwahati","Assam");
insert into bankbrn values(204,"Tezpur","Assam");
insert into bankbrn values(205,"Manipur","Manipur");
```

```
insert into accountin values(201,101);
insert into accountin values(202,102);
insert into accountin values(203,103);
insert into accountin values(204,104);
insert into accountin values(205,105);
```

```
insert into deposit values(501,"Savings","2014-12-23",5000);
insert into deposit values(502,"Savings","2014-10-21",8000);
insert into deposit values(503,"Current","2014-09-18",7000);
insert into deposit values(504,"Savings","2014-12-26",2500);
insert into deposit values(505,"Current","2014-11-28",7600);
```

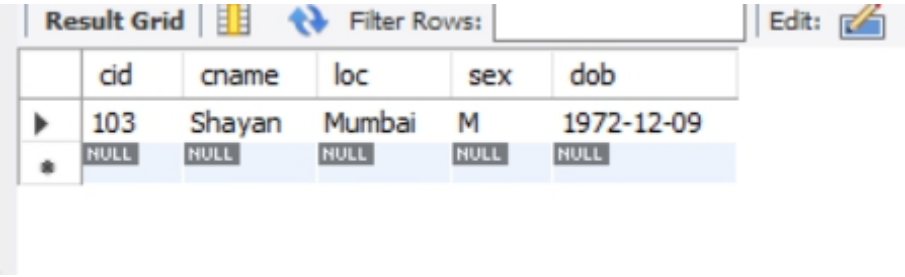
```
insert into borrower values(101,120048);
insert into borrower values(102,120049);
insert into borrower values(103,120042);
insert into borrower values(104,120043);
insert into borrower values(105,120044);
```

```
insert into loan values(12004489,"Car loan","2018-09-12",78000);
insert into loan values(12004478,"Home loan","2017-09-12",500000);
insert into loan values(12004789,"Car loan","2014-09-12",700000);
insert into loan values(12001489,"Educational loan","2018-10-12",75000);
insert into loan values(12004455,"Home loan","2016-09-11",78000);
```

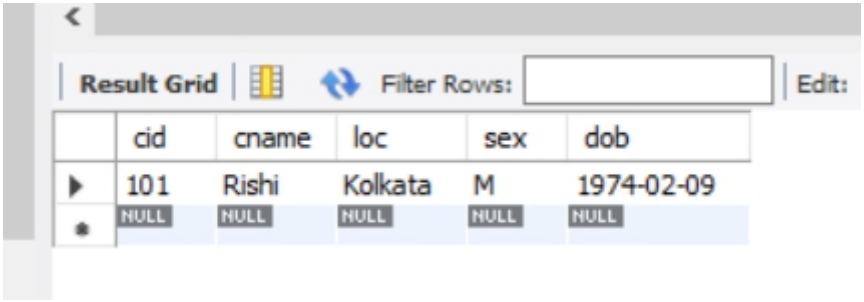
```
insert into depositor values(101,12004489);
```

```
insert into depositor values(102,12004389);
insert into depositor values(103,12004284);
insert into depositor values(104,12004136);
insert into depositor values(105,12004789);
```

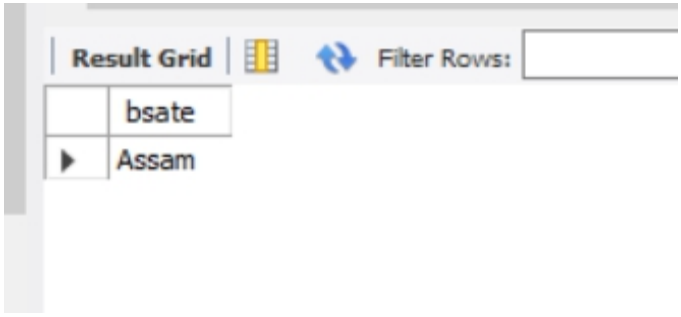
(ii)
select * from customer where loc="Mumbai";



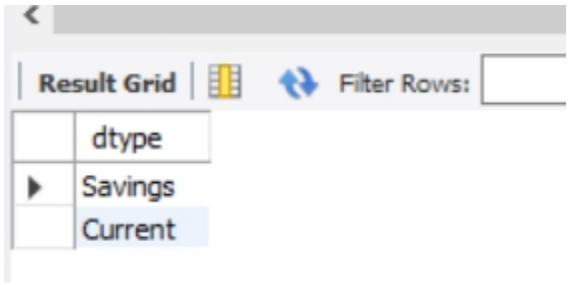
(iii)select * from customer where loc="Kolkata" and sex="M";



(iv)select bsate from bankbrn group by bsate having count(*)>1;



(v) select distinct dtype from deposit;



(vi) delete from loan;

Mac address-> 8C-16-45-E7-D6-DB

Date:2/12/20

EXPERIMENT NO:4
SUB QUERIES AND JOIN

AIM:
Consider the database for the organization and Write the queries for the following

(i) display the empno, name, and salaries for employees whose average salary is higher than the average salary of the organization
(ii)Display the details of employees whose salary is equal to the minimum salary of organisation.
(iii)Display all the employees whose designation is same as that of 'Arun'
(iv) display the empno and name of employees who earn more than any Employee in dept 1.
(v)Display the empno, name, departments that the departments are same in both the emp and dept
(vi) Display the employee details by implementing left inner join
(vii)Display employee details by implementing a right outer join

OBJECTIVES

To understand sub queries and join in Mysql.

THEORY

NESTED QUERIES:

A sub query is a query within a query. These sub queries can reside in the WHERE clause, the FROM clause, or the SELECT clause. The first query in the SQL statement is known as the outer query. The query inside the SQL statement is known as the inner query. The inner query is executed first. The output of an inner query is used as the input for the outer query. The entire SQL statement is sometimes referred to as a nested query.

JOINS:

MySQL JOINS are used to retrieve data from multiple tables. A MySQL JOIN is performed whenever two or more tables are joined in a SQL statement.

There are different types of MySQL joins:

MySQL INNER JOIN (or sometimes called simple join)
MySQL LEFT OUTER JOIN (or sometimes called LEFT JOIN)
MySQL RIGHT OUTER JOIN (or sometimes called RIGHT JOIN)

INNER JOIN (simple join)

MySQL INNER JOINS return all rows from multiple tables where the join condition is met.

Syntax

```
Select columns from table1
Inner join table2
On table1.column=table2.column;
```

LEFT OUTER JOIN

Another type of join is called a MySQL LEFT OUTER JOIN. This type of join returns all rows from the LEFT-hand table specified in the ON condition and only those rows from the other table where the joined fields are equal.

Syntax

```
Select columns from table
left join table2
On table1.column=table2.column;
```

RIGHT OUTER JOIN

Another type of join is called a MySQL RIGHT OUTER JOIN. This type of join returns all rows from the RIGHT-hand table specified in the ON condition and only those rows from the other table where the joined fields are equal.

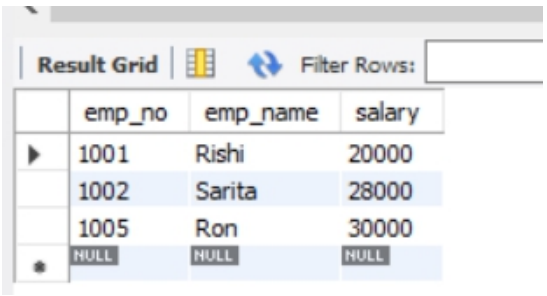
Syntax

```
Select columns from table 1
Right join table2
On table1.column=table2.column;
```

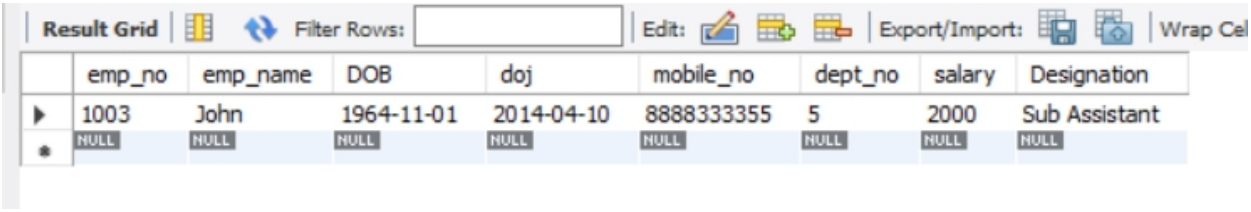
PROCEDURE: To be written by the student

Code and Output:

(i)SELECT emp_no,emp_name,salary from employee WHERE salary>(SELECT avg(salary)FROM employee);

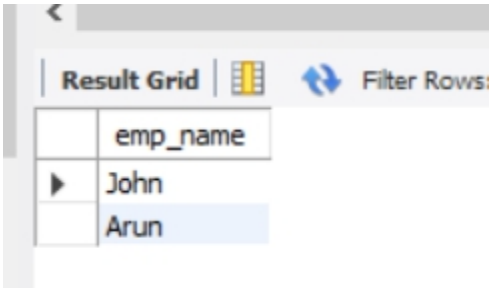


(ii)SELECT * from employee WHERE salary=(SELECT min(salary)FROM employee);



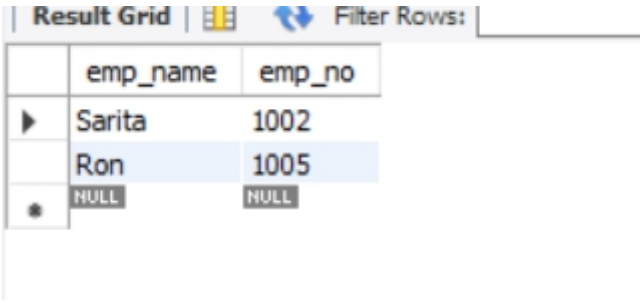
emp_no	emp_name	DOB	doj	mobile_no	dept_no	salary	Designation
1003	John	1964-11-01	2014-04-10	8888333355	5	2000	Sub Assistant
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

(iii)SELECT emp_name from employee WHERE designation=(SELECT designation FROM employee where emp_name="Arun");



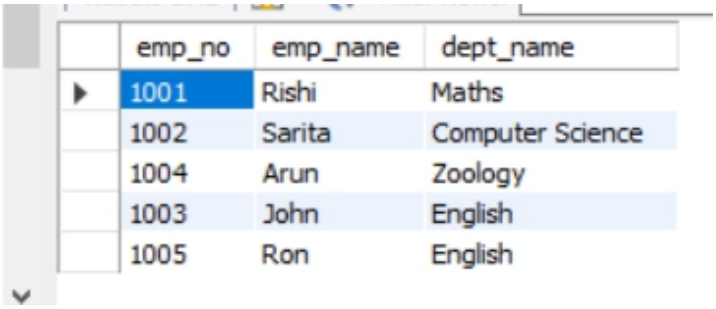
emp_name
John
Arun

(iv)SELECT emp_name ,emp_no from employee WHERE salary>(SELECT max(salary) FROM employee where dept_no=1);



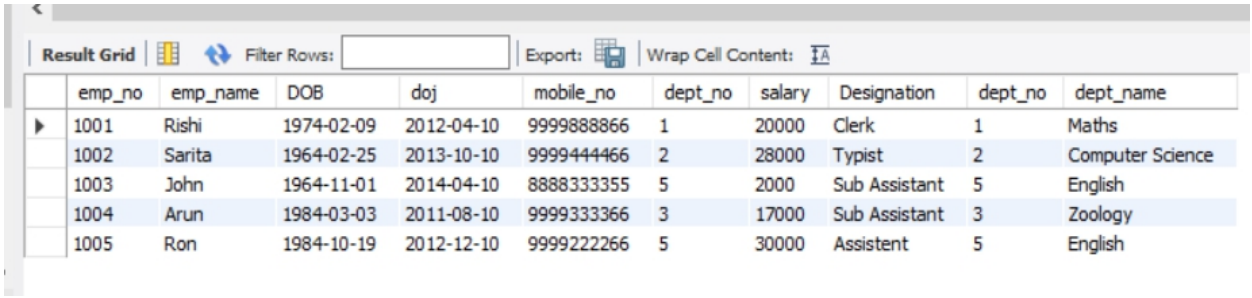
emp_name	emp_no
Sarita	1002
Ron	1005
NULL	NULL

(v)select emp_no,emp_name,dept_name from employee,department where employee.dept_no=department.dept_no;



emp_no	emp_name	dept_name
1001	Rishi	Maths
1002	Sarita	Computer Science
1004	Arun	Zoology
1003	John	English
1005	Ron	English

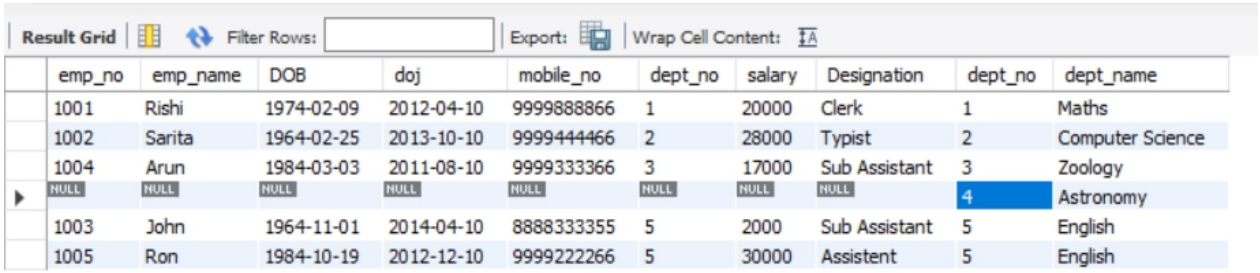
(vi) select * from employee left inner join department on employee.dept_no=department.dept_no;



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

emp_no	emp_name	DOB	doj	mobile_no	dept_no	salary	Designation	dept_no	dept_name
1001	Rishi	1974-02-09	2012-04-10	9999888866	1	20000	Clerk	1	Maths
1002	Sarita	1964-02-25	2013-10-10	9999444466	2	28000	Typist	2	Computer Science
1003	John	1964-11-01	2014-04-10	8888333355	5	2000	Sub Assistant	5	English
1004	Arun	1984-03-03	2011-08-10	9999333366	3	17000	Sub Assistant	3	Zoology
1005	Ron	1984-10-19	2012-12-10	9999222266	5	30000	Assistent	5	English

(vii) select * from employee right outer join department on employee.dept_no=department.dept_no;



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

emp_no	emp_name	DOB	doj	mobile_no	dept_no	salary	Designation	dept_no	dept_name
1001	Rishi	1974-02-09	2012-04-10	9999888866	1	20000	Clerk	1	Maths
1002	Sarita	1964-02-25	2013-10-10	9999444466	2	28000	Typist	2	Computer Science
1004	Arun	1984-03-03	2011-08-10	9999333366	3	17000	Sub Assistant	3	Zoology
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	4	Astronomy
1003	John	1964-11-01	2014-04-10	8888333355	5	2000	Sub Assistant	5	English
1005	Ron	1984-10-19	2012-12-10	9999222266	5	30000	Assistent	5	English

RESULT:

The nested queries and joins are executed successfully.

Programs

1. Consider the database for a banking enterprise. Write the queries for the below questions.

(i) List the deposit account number and amount in which the deposit scheme having maximum deposit is opened

(ii) List the account number and amount of that savings bank deposit scheme in which minimum amount is deposited.

(iii)List the customers having accounts in ‘Chennai’ branch

(iv)List the customers having more than one account

(v) List the customers having same name but different account numbers.

(vi)List the customer name that is having maximum deposit account in bank

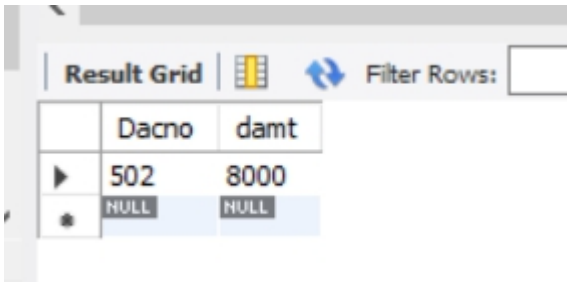
(vii)List the customer who has borrowed highest amount of home loan

(viii)Display the customer details by implementing left inner join.

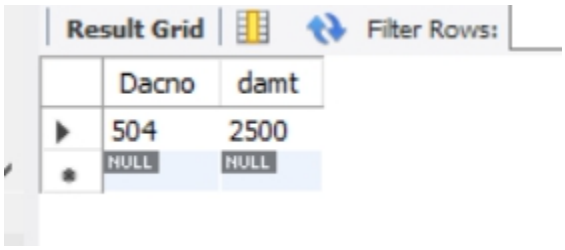
(ix)Display the customer details by implementing a right outer join.

Answers-

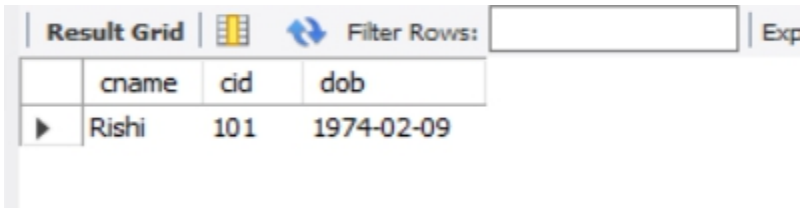
(i)select Dacno,damt from deposit where damt=(select max(damt) from deposit);



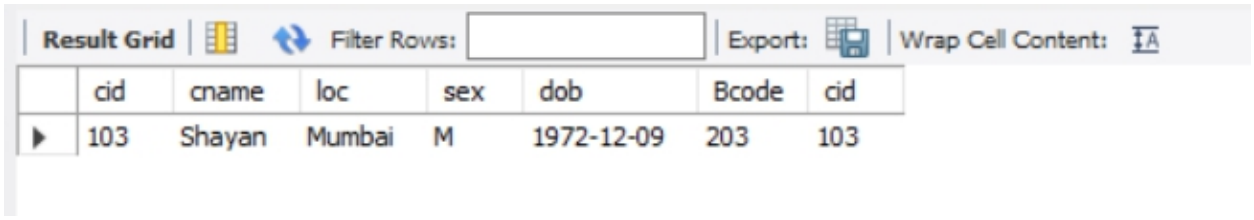
(ii)select Dacno,damt from deposit where damt=(select min(damt) from deposit where dtype=”savings”);



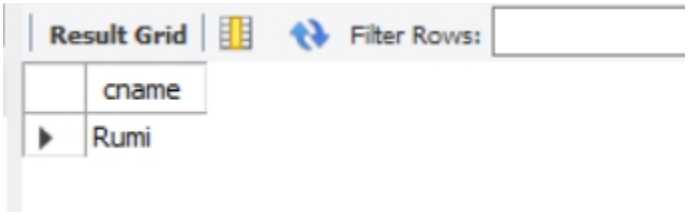
(iii)select cname,customer.cid,dob from customer inner join accountin on customer.cid=accountin.cid inner join bankbrn on accountin.Bcode=bankbrn.Bcode where bankbrn.bloc="Chennai";



(iv) select * from customer inner join accountin on customer.cid=accountin.cid group by accountin.cid having count(accountin.cid)>1;

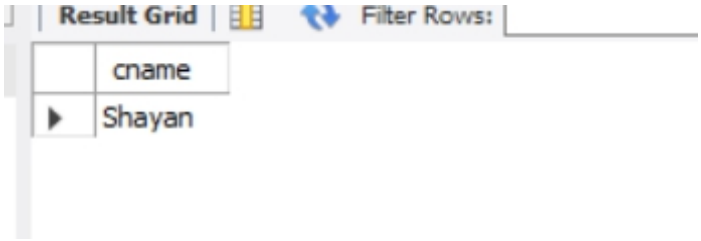


(vi)select cname from customer join depositor on customer.cid=depositor.cid join deposit on depositor.dacno=deposit.Dacno where deposit.damt=(select max(damt) from deposit);


Result Grid | Filter Rows:

cname
Rumi

(vii)select cname from customer join borrower on customer.cid=borrower.cid join loan on loan.Lacno=borrower.lacno where loan.lamt=(select max(lamt) from loan group by ltype having ltype="Home loan");

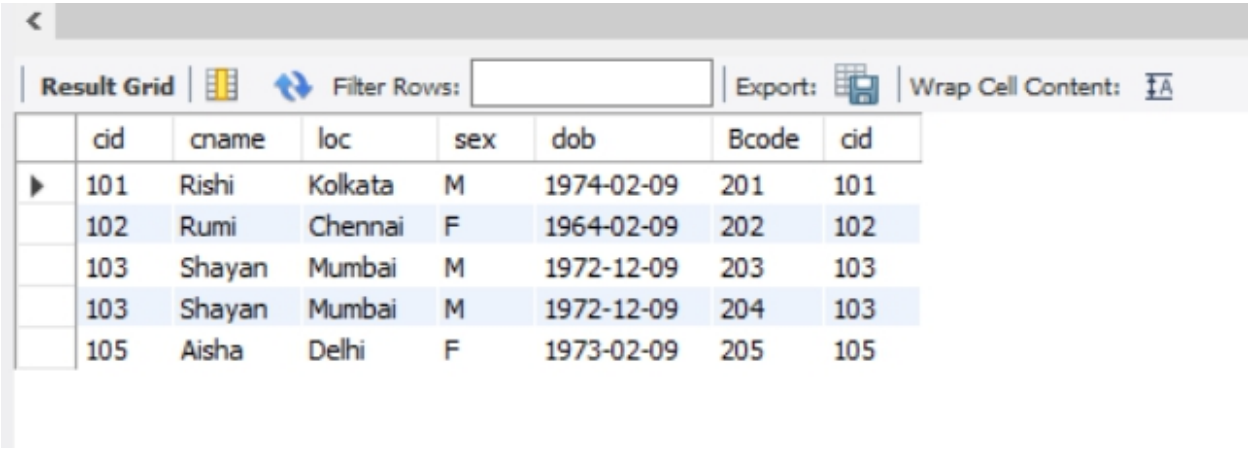

Result Grid | Filter Rows:

cname
Shayan

(viii)select * from customer left inner join accountin on accountin.cid=customer.cid order by customer.cid;

cid	cname	loc	sex	dob	Bcode
101	Rishi	Kolkata	M	1974-02-09	201
102	Rumi	Chennai	F	1964-02-09	202
103	Shayan	Mumbai	M	1972-12-09	203
103	Shayan	Mumbai	M	1972-12-09	204
104	Tara	Delhi	F	1974-10-19	NULL
105	Aisha	Delhi	F	1973-02-09	205

(ix)select * from customer right outer join accountin on accountin.cid=customer.cid order by customer.cid;



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

cid	cname	loc	sex	dob	Bcode	cid
101	Rishi	Kolkata	M	1974-02-09	201	101
102	Rumi	Chennai	F	1964-02-09	202	102
103	Shayan	Mumbai	M	1972-12-09	203	103
103	Shayan	Mumbai	M	1972-12-09	204	103
105	Aisha	Delhi	F	1973-02-09	205	105

2. Consider the database for a college. Write the queries for the below questions.

(i) List out the ID, Name and Date of Birth of students registered for a specific course.

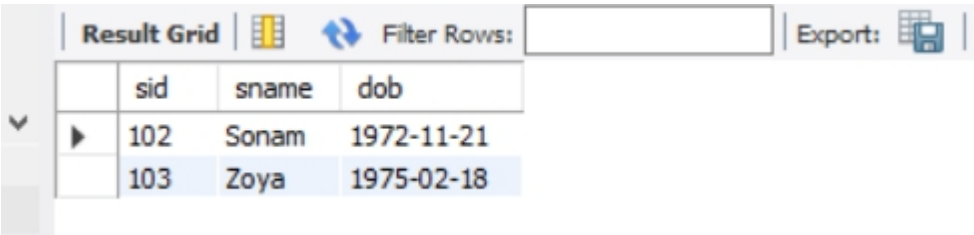
(ii) List out the ID, Name and Date of Birth of students registered for a specific course,staying in a specific Hostel.

(iii) List the names of faculties who teach for a specific course.

(iv)Display the student details by implementing left inner join
(v)Display the student details by implementing a right outer join

Answers:

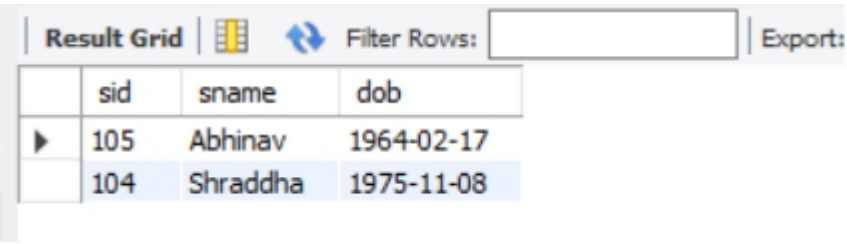
(i) select student.sid, student. sname, dob from student join register on register.sid=student.sid join course on course.cid=register.cid where course.cname="BTech";



Result Grid | Filter Rows: | Export:

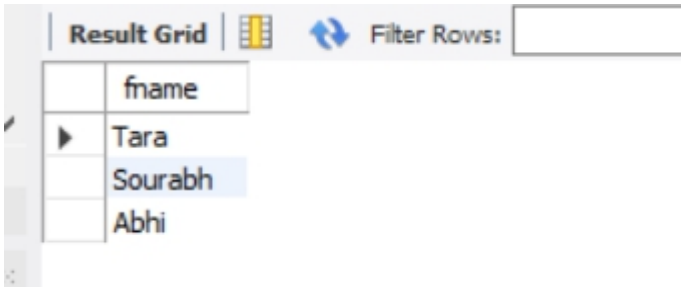
sid	sname	dob
102	Sonam	1972-11-21
103	Zoya	1975-02-18

(ii)select student.sid,student.sname,dob from student join register on register.sid=student.sid join course on course.cid=register.cid join stays_in on stays_in.sid=register.sid join hostel on stays_in.hid=hostel.hid where course.cname="BCA" and hostel.hname="AT-8";


Result Grid | Filter Rows: | Export:

sid	sname	dob
105	Abhinav	1964-02-17
104	Shraddha	1975-11-08

(iii)select fname from faculty join teaching on teaching.f_id=faculty.F_id join course on course.cid=teaching.cid where cname="BTech";


Result Grid | Filter Rows:

fname
Tara
Sourabh
Abhi

(iv) select * from student left inner join department on student.dno=department.dno left inner join stays_in on student.sid=stays_in.sid left inner join

register on student.sid=register.sid left inner join on course on register.cid=course.cid;

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

sid	sname	sex	dob	dno	hid	sid	hid	hname	seats
101	Rishab	M	1969-12-28	1	301	101	301	RCC	500
102	Sonam	F	1972-11-21	2	302	102	302	AT-8	200
103	Zoya	F	1975-02-18	2	302	103	302	AT-8	200
103	Zoya	F	1975-02-18	2	304	103	304	RCC-2	400

(v)select * from student right outer join
department on student.dno=department.dno
right outer join stays_in on
student.sid=stays_in.sid right outer join
register on student.sid=register.sid right outer join
course on register.cid=course.cid;

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

sid	sname	sex	dob	dno	dno	dname	hid	sid	sid	cid	sem	cid	cname	credits	dno
101	Rishab	M	1969-12-28	1	1	Mathematics	301	101	101	401	2	401	Bsc	12	1
102	Sonam	F	1972-11-21	2	2	Science	302	102	102	402	2	402	BCA	11	2
103	Zoya	F	1975-02-18	2	2	Science	302	103	103	402	2	402	BCA	11	2
103	Zoya	F	1975-02-18	2	2	Science	304	103	103	402	2	402	BCA	11	2
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	403	BSc	12	3
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	105	404	2	404	BTech	13	4
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	104	405	2	405	BTech	13	4

Mac address-> 8C-16-45-E7-D6-DB

Date: 9/12/20

EXPERIMENT NO 5

VIEWS

AIM:

Write the queries for the following
(i) Create a view emp from employee such that it contains only emp_no and emp_name and department.
(ii) Create a view dept from department with only dept_no and location.
(iii) Create a view that contains the details of employees who are managers only.
(iv) drop the views.

OBJECTIVES

To understand views in Mysql

THEORY

A view is the tailored presentation of data contained in one or more table and can also be said as restricted view to the data in the tables. A view is a "virtual table" or a "stored query" which takes the output of a query and treats it as a table. The table upon which a view is created is called as base table. A view is a logical table based on a table or another view. A view contains no data of its own but is like a window through which data from tables can be viewed or changed. The tables on which a view is based are called base tables. The view is stored as a SELECT statement in the data dictionary.

Advantages of a view:

a. Additional level of table security.

b. Hides data complexity.

c.Simplifies the usage by combining multiple tables into a single table

Creating and dropping view:

Syntax:

Create or replace view view_name AS SELECT column_name(s) FROM table_name WHERE condition;

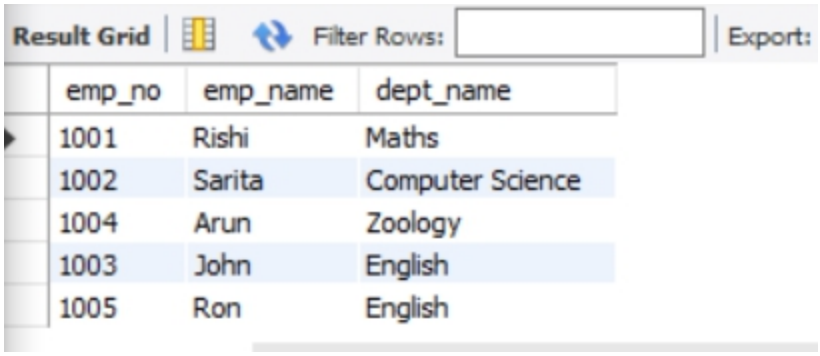
Drop view <view name>;

PROCEDURE

1) Create the employee table;
2) Create the view
3) display the content of view

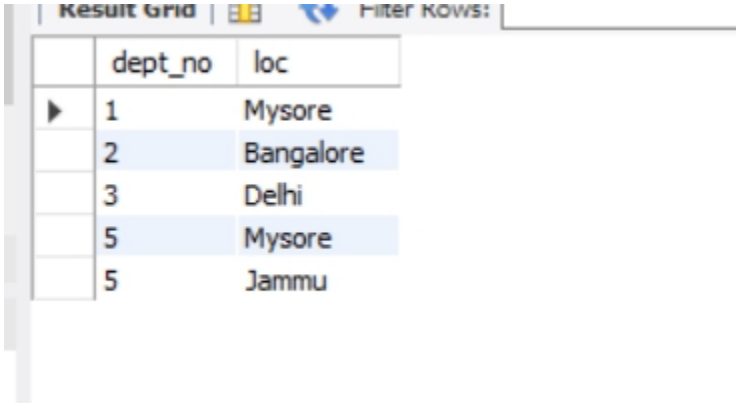
CODE FOR QUERIES:

(i)create view Emp_details as (select emp_no,emp_name,dept_name from employee,department where employee.dept_no=department.dept_no);
select * from Emp_details;


Result Grid | Filter Rows: | Export:

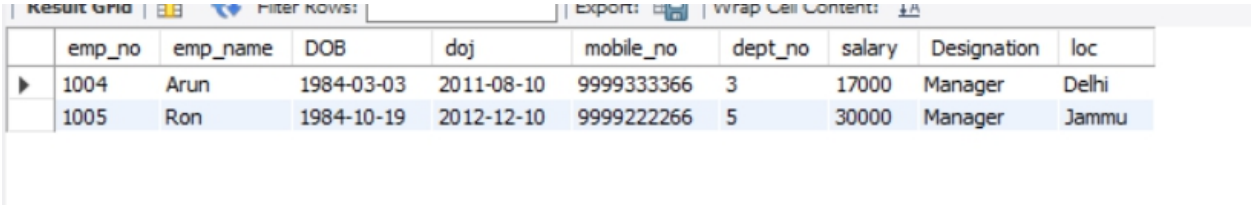
emp_no	emp_name	dept_name
1001	Rishi	Maths
1002	Sarita	Computer Science
1004	Arun	Zoology
1003	John	English
1005	Ron	English

(ii) create view Dept_View as (select department.dept_no,employee.loc from employee,department where employee.dept_no=department.dept_no);
select * from Dept_View;


Result Grid | Filter Rows:

dept_no	loc
1	Mysore
2	Bangalore
3	Delhi
5	Mysore
5	Jammu

(iii) create view Details as (select * from employee where designation="Manager");
select * from Details;



emp_no	emp_name	DOB	doj	mobile_no	dept_no	salary	Designation	loc
1004	Arun	1984-03-03	2011-08-10	9999333366	3	17000	Manager	Delhi
1005	Ron	1984-10-19	2012-12-10	9999222266	5	30000	Manager	Jammu

Result

Thus the views are created successfully

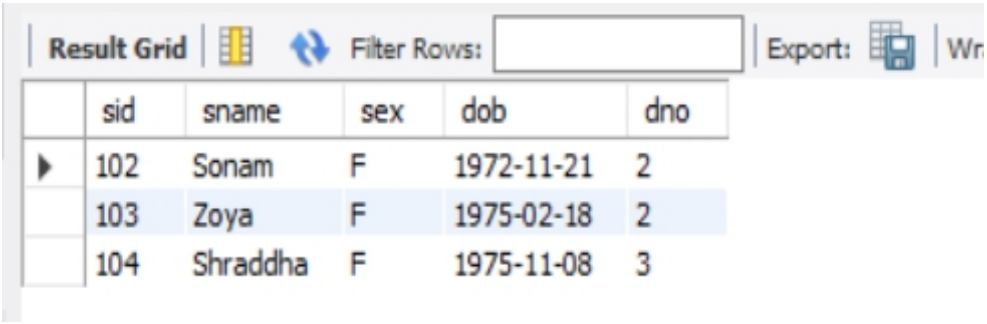
Problems

1. Create and drop views on student table

2. Create and drop views on customer and deposit table

1. Ans: -

```
create view Female_Student as (select * from student where sex="F");
select * from Female_Student;
drop view Female_Student;
```

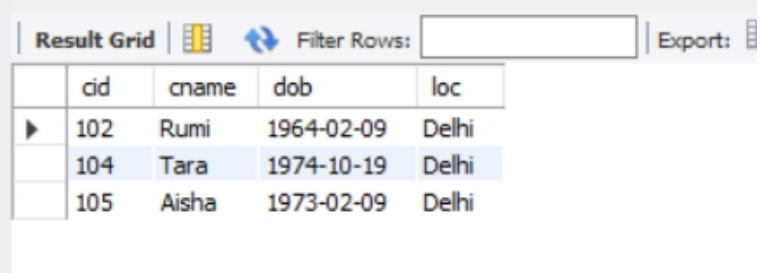


Result Grid | Filter Rows: | Export: | Wr

sid	sname	sex	dob	dno
102	Sonam	F	1972-11-21	2
103	Zoya	F	1975-02-18	2
104	Shraddha	F	1975-11-08	3

2. Ans:-

```
create view Customer_View as (select cid,cname,dob,loc from customer where loc="Delhi");

select * from Customer_View;

drop view Customer_View;
```



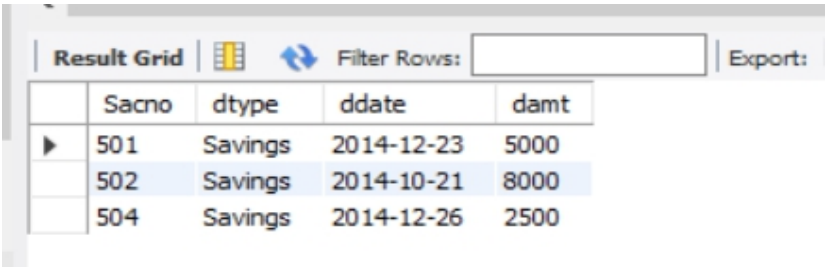
Result Grid | Filter Rows: | Export:

cid	cname	dob	loc
102	Rumi	1964-02-09	Delhi
104	Tara	1974-10-19	Delhi
105	Aisha	1973-02-09	Delhi

create view Deposit_type as (select * from deposit where dtype="Savings");

select * from Deposit_type;

drop view Deposit_type;



Result Grid | Filter Rows: | Export:

Sacno	dtype	ddate	damt
501	Savings	2014-12-23	5000
502	Savings	2014-10-21	8000
504	Savings	2014-12-26	2500

Mac address-> 8C-16-45-E7-D6-DB

Date:16/12/20 EXPERIMENT NO:6

PROCEDURE

AIM:

Write a procedure which increases the salary of an employee. It accepts an employee number and salary increase amount. It uses the employee number to find the current salary from the EMPLOYEE table and update the salary.

OBJECTIVES

To understand procedure in Mysql

THEORY

PROCEDURE:

In MySQL, a procedure is a stored program that you can pass parameters into. It does not return a value like a function does.

Syntax
Create procedure procedure name (parameter data type, parameter data type...)
Begin
Declaration section
Executable _section
End;

Procedure name
The name to assign to this procedure in MySQL.

Parameter

When creating a procedure, there are three types of parameters that can be declared:
1. IN - The parameter can be referenced by the procedure. The value of the parameter cannot be overwritten by the procedure.
2. OUT - The parameter cannot be referenced by the procedure, but the value of the parameter can be overwritten by the procedure.
3. IN OUT - The parameter can be referenced by the procedure and the value of the parameter can be overwritten by the procedure.

Declaration section
The place in the procedure where you declare local variables.

Executable section
The place in the procedure where you enter the code for the procedure.

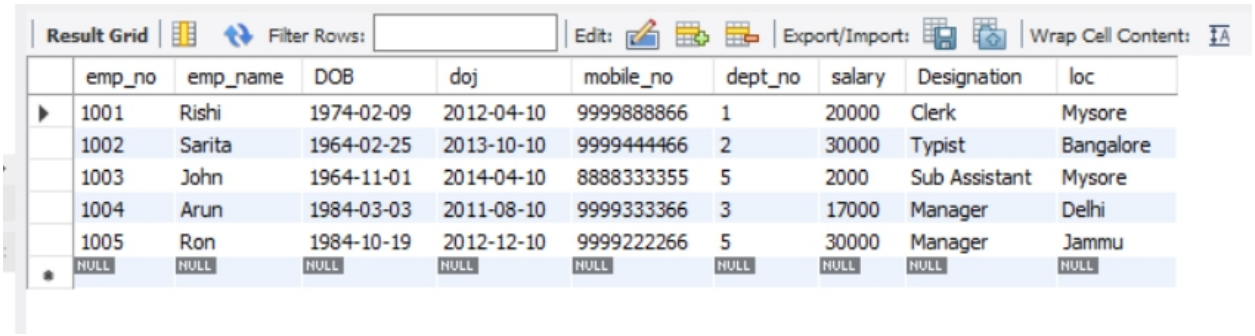
PROCEDURE:

i.)Write the procedure with empno and increment name as parameter
ii.)Use update command to increment salary

CODE

```
use organization;

CREATE PROCEDURE IncSalary(IN emp_num int,IN sal int)
BEGIN
        update employee set salary=salary+sal where emp_no=emp_num;
END;
call IncSalary(1002,2000);
```

OUTPUT:*Salary of emp_no 1002 changed to 30,000 from 28,000*



Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

emp_no	emp_name	DOB	doj	mobile_no	dept_no	salary	Designation	loc
1001	Rishi	1974-02-09	2012-04-10	9999888866	1	20000	Clerk	Mysore
1002	Sarita	1964-02-25	2013-10-10	9999444466	2	30000	Typist	Bangalore
1003	John	1964-11-01	2014-04-10	8888333355	5	2000	Sub Assistant	Mysore
1004	Arun	1984-03-03	2011-08-10	9999333366	3	17000	Manager	Delhi
1005	Ron	1984-10-19	2012-12-10	9999222266	5	30000	Manager	Jammu
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

RESULT:
The procedures are executed successfully.

Problems

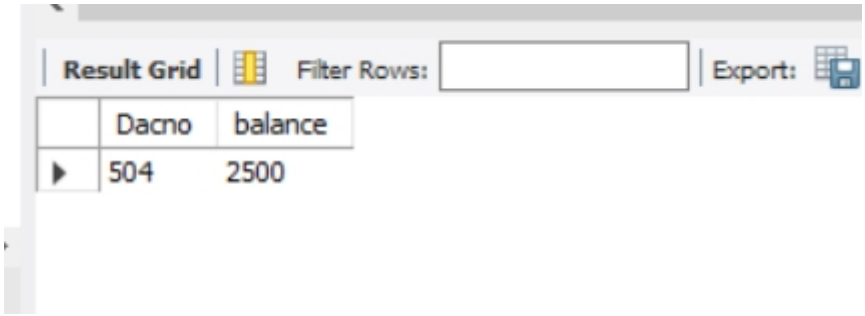
1. Write a procedure which accept the account number of a customer and retrieve the balance.
2. Write a procedure which accepts the student number and displays the department in which he belongs to.

Answers-

```
USE Bank;
1. DELIMITER //
CREATE PROCEDURE getbalance(IN acno int)
BEGIN
        select Dacno,damt as balance from deposit where Dacno=acno;
END //
```

DELIMITER ;
call getbalance(504);

OUTPUT:



2.

Ans-

use college;

DELIMITER //

CREATE PROCEDURE getdept(IN sno int)

BEGIN

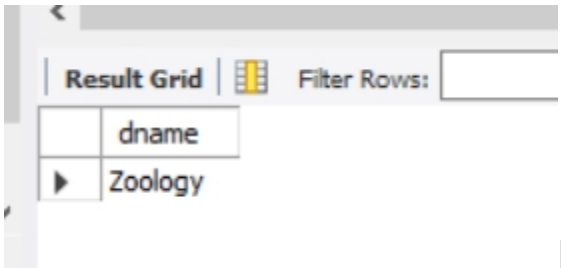
select department.dname from department join student on student.dno=department.dno where student.sid=sno;

END //

DELIMITER ;

call getdept(105);

OUTPUT



Mac address-> 8C-16-45-E7-D6-D

Date:4/2/20

EXPERIMENT NO:7
CURSORS

AIM:

Write a cursor to display the list of employees who are working as managers.

OBJECTIVES

To implement cursor

THEORY

A cursor is a SELECT statement that is defined within the *declaration* section of your stored program in MySQL.

1. Declare a cursor
Declare cursor name cursor for select statement;
2. Open the cursor.
Open cursor name;

3. Fetch cursor
The purpose of using a cursor, in most cases, is to retrieve the rows from your cursor so that some type of operation can be performed on the data. Fetch cursor name into variable list;

4. Close the cursor

Close cursor name;

PROCEDURE

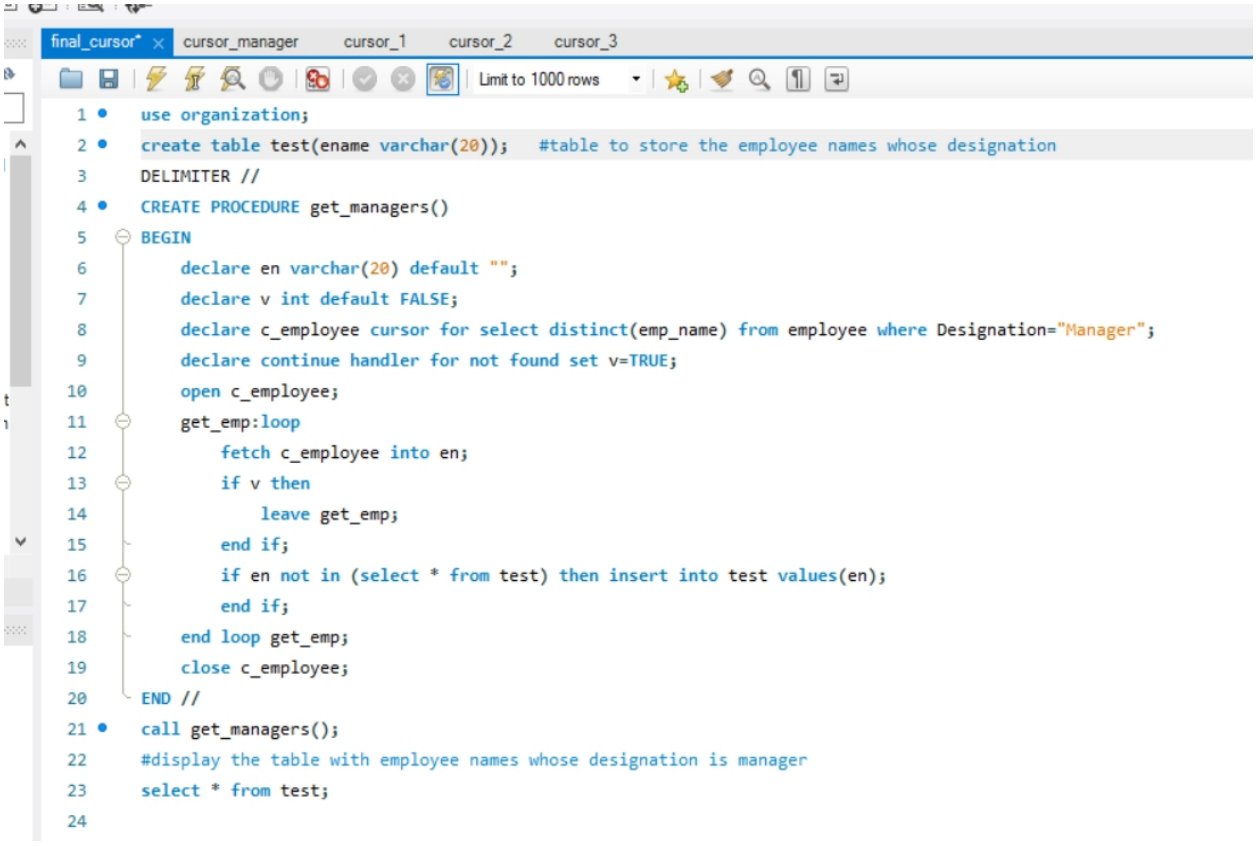
To be written by the student

Code:

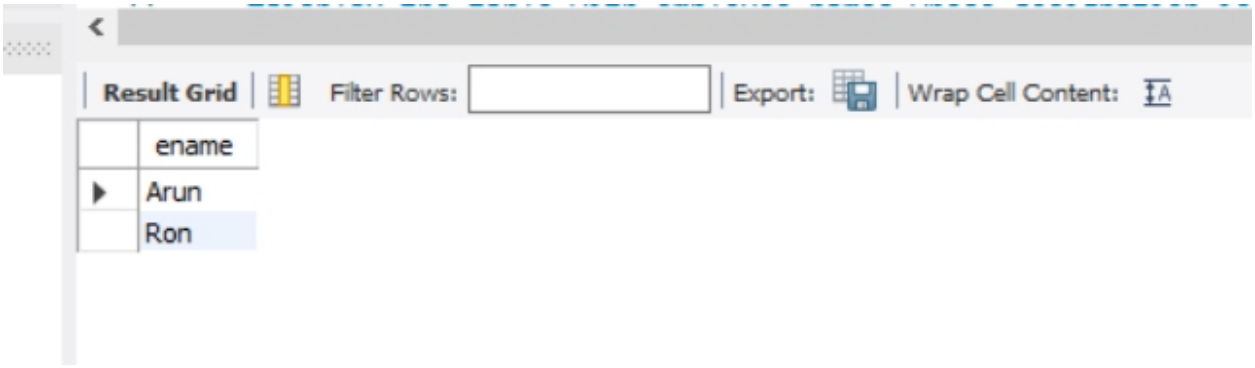
```
use organization;
create table test(ename varchar(20)); #table to store the employee names whose designation


DELIMITER //
CREATE PROCEDURE get_managers()
BEGIN
        declare en varchar(20) default "";
  declare v int default FALSE;
        declare c_employee cursor for select distinct(emp_name) from employee where
Designation="Manager";
  declare continue handler for not found set v=TRUE;
        open c_employee;
```

```
  get_emp:loop
                    fetch c_employee into en;
    if v then
                        leave get_emp;
                    end if;
    if en not in (select * from test) then insert into test values(en);
    end if;
        end loop get_emp;
        close c_employee;
END //
call get_managers();
#display the table with employee names whose designation is manager
select * from test;
```

Screenshot of code:



Output:



RESULT
The cursor is executed successfully.

PROGRAMS

1.Create a cursor to modify the salary of 'Professors' belonging to all departments by 150%.

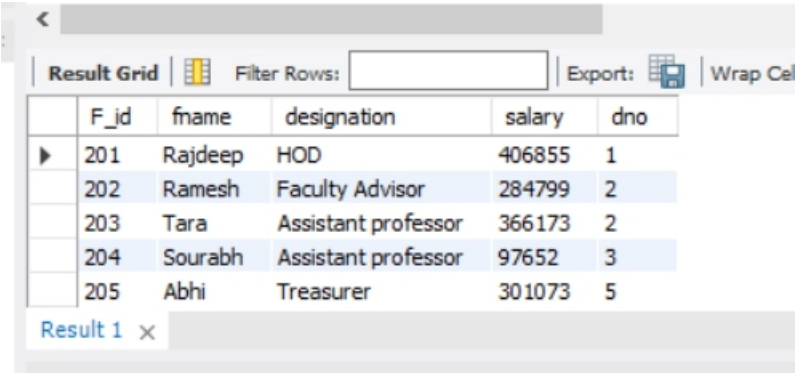
Ans-

1. Code:

```
 use organization;
drop table test;
drop procedure get_managers;
create table test(ename varchar(20));
select * from employee;
select * from faculty;
DELIMITER //
CREATE PROCEDURE get_s()
BEGIN
        declare en float;
   declare i float default 0.15;
   declare v int default FALSE;
```

```
        declare c_employee cursor for select salary from faculty where salary is not null;
    declare continue handler for not found set v=TRUE;
        open c_employee;
    get_emp:loop
                fetch c_employee into en;
      if v then
                        leave get_emp;
                end if;
      update faculty set salary=salary+i*salary;
        end loop get_emp;
        close c_employee;
END //
call get_s();
```

Output: Select * from faculty;



Result Grid | Filter Rows: | Export: | Wrap Cel

F_id	fname	designation	salary	dno
201	Rajdeep	HOD	406855	1
202	Ramesh	Faculty Advisor	284799	2
203	Tara	Assistant professor	366173	2
204	Sourabh	Assistant professor	97652	3
205	Abhi	Treasurer	301073	5

Result 1

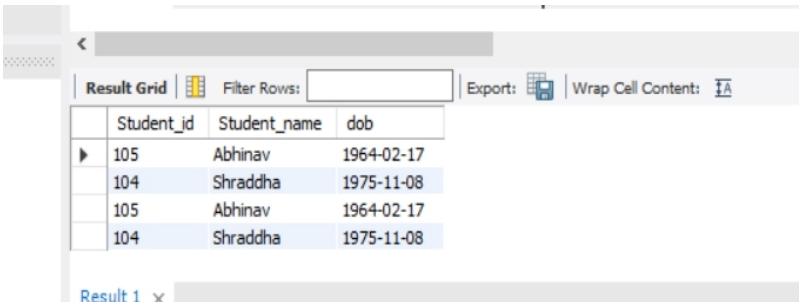
2.Consider the college database. Retrieve all students who have registered for a specific course and store their details into another table using cursors.

Ans: **Code:**

```
create table Btech_Student_details
(Student_id int,Student_name varchar(20),dob date);
select * from student;
DELIMITER //
```

```
CREATE PROCEDURE get_student()
BEGIN
        declare id int;
   declare names varchar(20) default "";
   declare v int default false;
   declare db date;
        declare c_stu cursor for select student.sid,sname,dob from student join register on student.sid=register.sid join course on register.cid=course.cid where cname="BTech";
   declare continue handler for not found set v=TRUE;
        open c_stu;
   get_stu:loop
                fetch c_stu into id,names,db;
      if v then
                        leave get_stu;
                end if;
      insert into Btech_Student_details value(id,names,db);
        end loop get_stu;
        close c_stu;
END //
call get_student();
select * from Btech_Student_details;
```

Output:



3.Consider the bank database. Retrieve all customers who have loan at a particular branch using cursor.
Ans-

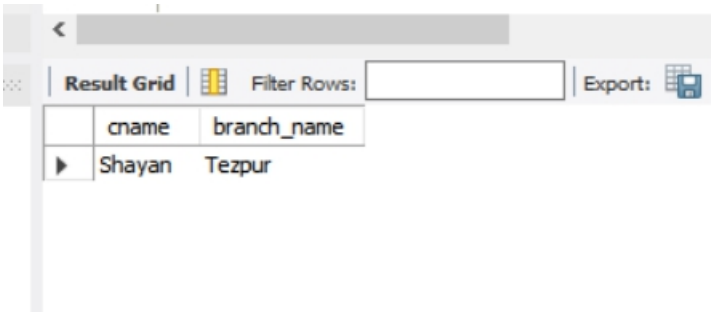
Code:

```
use bank;

create table brn_loan(cname varchar(20),branch_name varchar(20));

DELIMITER //

CREATE PROCEDURE get_customers()

BEGIN

        declare c varchar(20) default "";

  declare br varchar(20) default "";

  declare v int default false;

        declare c cursor for select cname,bankbrn.bloc from customer

  join accountin on customer.cid=accountin.cid

  join borrower on accountin.cid=borrower.cid

  join loan on borrower.lacno=loan.Lacno

  join bankbrn on bankbrn.bcode=accountin.bcode

        where bankbrn.Bcode=204;

  declare continue handler for not found set v=TRUE;

  open c;

  get_customers:loop

                fetch c into c,br;

    if v then

                        leave get_customers;

                end if;

    insert into brn_loan values(c,br);
```

```
        end loop get_customers;

        close c;

END //

call get_customers();

select * from brn_loan;
```

Output:

The customer anme and branch location of customers who have borrowed loan from branch with branch code 204 has been saved in a different table named brn_loan.



Date:18/2/20

EXPERIMENT NO:8
TRIGGER

AIM:

Write a Trigger for employee table it will store the updated salary into another table SALARY while updating salary.

OBJECTIVES

To understand triggers in Mysql

THEORY.

A trigger is a set of actions that are run automatically when a specified change operation (INSERT, UPDATE, or DELETE statement) is performed on a specified table. The syntax to create an AFTER UPDATE Trigger in MySQL is:

Create trigger trigger_nane

After update On tablename for each row

Begin Variable declarations

Trigger code

End;

PROCEDURE

(i) Create the employee table

(ii) Insert values

(iii) Write the after update trigger

(iv) Update the salary in employee table

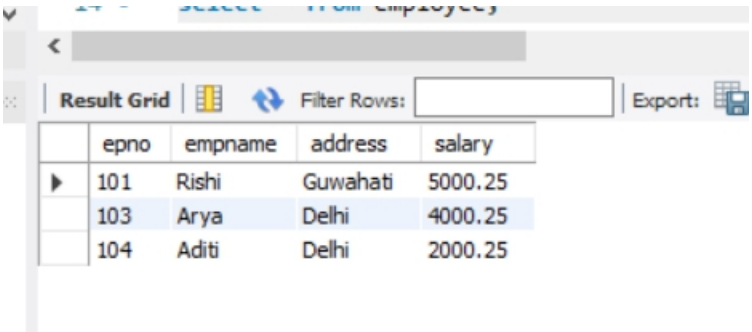
(v) Display the SALARY table

CODE

```
create database Trigger_db;
use Trigger_db;
desc employee;
create table Employee
(epno int,empname varchar(20),address varchar(30),salary float);
insert into employee values(101,"Rishi","Guwahati",5000.25);
insert into employee values(103,"Arya","Delhi",4000.25);
insert into employee values(104,"Aditi","Delhi",2000.25);
select * from employee;
create trigger update_salary before update on employee
for each row set new.salary=@sal+new.salary;
set @sal=0;
update employee set salary=5000.25 where epno=101;
select * from employee;
```

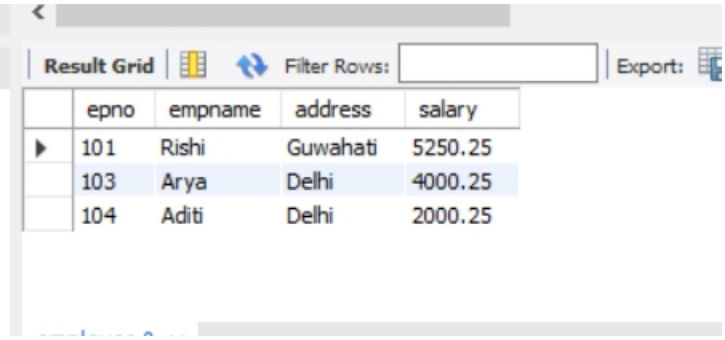
OUTPUT :

Employee table before any update:



epno	empname	address	salary
101	Rishi	Guwahati	5000.25
103	Arya	Delhi	4000.25
104	Aditi	Delhi	2000.25

Employee table after any update:



epno	empname	address	salary
101	Rishi	Guwahati	5250.25
103	Arya	Delhi	4000.25
104	Aditi	Delhi	2000.25

RESULT: The trigger procedure has been executed successfully.

Problems 1. Write an update trigger on Account table. The system should keep track of the records that are being updated.

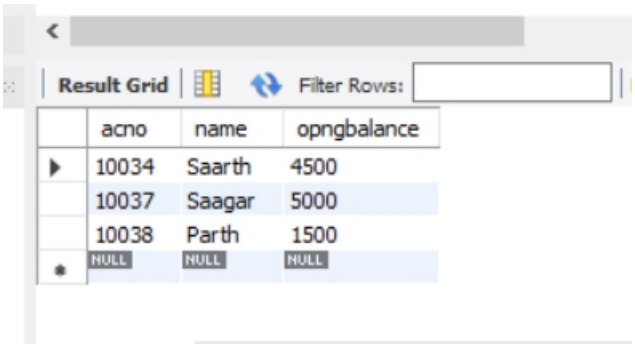
Ans-

Code:

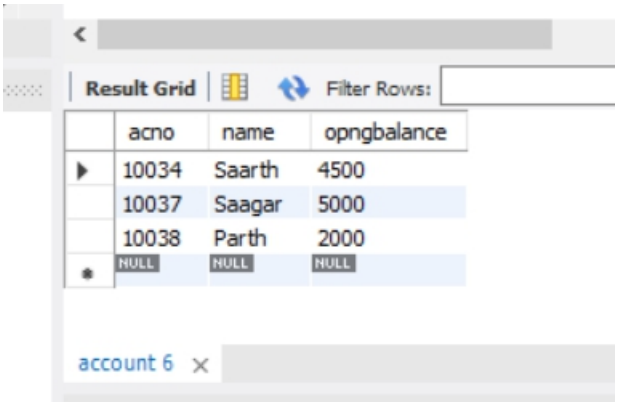
```
use bank;

create table Account(acno int primary key,name varchar(20),opngbalance int);

insert into account values(10034,"Saarth",4500);

insert into account values(10038,"Parth",1500);

insert into account values(10037,"Saagar",5000);

select * from account;

drop trigger update_opngbalance;

create trigger update_opngbalance before update on Account

for each row set new.opngbalance=@amt+new.opngbalance;

set @amt=1000;

update Account set opngbalance=500 where acno=10038;

select * from account;
```

Output:

Account table before any update:



Account table after any update:



Problems 2.Write a before delete trigger on Student table.

Ans-

Code:

use organization;

create table student(sid int,sname varchar(20),sex char);

insert into student values(101,"Rishab","M");

insert into student values(102,"Sonam","F");

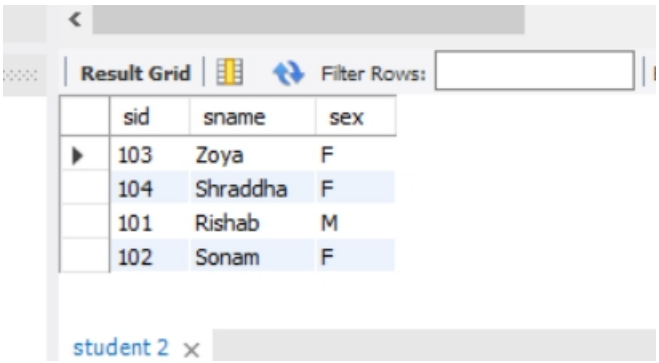
insert into student values(103,"Zoya","F");

insert into student values(104,"Shraddha","F");

```
select * from student;

drop trigger delete_row;

drop table deleted_rows;

create table deleted_rows(sid int,sname varchar(20),sex char);

DELIMITER //

create trigger delete_row before delete on student for each row

begin

        insert into deleted_rows values(old.sid,old.sname,old.sex);

END;//

DELIMITER ;

delete from student where sid=102;

select * from deleted_rows;
```

Output:

Student table before delete:



Student table after deleting a row:

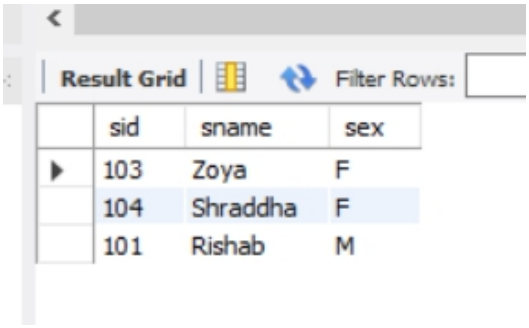
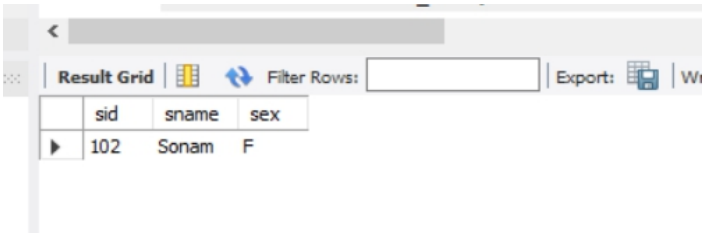


Table on which the deleted values are entered:



Date:25/2/20

Experiment No: 9

CONCEPTS OF NORMALIZATION

AIM:

Checking Normalization of a database table (First Normal form)

Problem Statement:

An exercise to check whether the given database table is normalized or not. If yes find out the status of normalization and reasoning.

Objective:

To study the concept of various levels of normalization and understand how to convert into normalized forms.

Requirements:

Mysql database software

Design/Theory

Create a database table in SQL with a few no of rows and columns. Analyze the table and determine to which normal form it belongs to according to the rules and regulations of each normal forms.

Procedure:

Consider a student table as given below.

Social Security Number	FirstName	LastName	Major
123-45-6789	Jack	Jones	Library and Information Science
222-33-4444	Lynn	Lee	Library and Information Science
987-65-4321	Mary	Ruiz	Pre-Medicin
123-54-3210	Lynn	Smith	Pre-Law

We can easily verify that this table satisfies the definition of 1NF: viz., it has no duplicated rows; each cell is single-valued (i.e., there are no repeating groups or arrays); and all the entries in a given column are of the same kind. In this table we can see that the key, SSN,functionally determines the other attributes; i.e.,FirstName, LastName, and Major.

Date:4/3/20

Experiment No: 10

AIM

Checking Normalization of a database table (Third normal form).

Problem Statement:

An excercise to check whether the given database table is normalized or not. If yes find out the status of normalization and reasoning.

Objective:

To study the concept of various levels of normalization and understand how to convert into normalized forms.

Requirements: Mysql database software

Design/Theory

Create a database table in SQL with a few no of rows and columns.Analyze the table and determine to which normal form it beongs to according to the rules and regulations of each normal forms.

Procedure:

Consider a book database table as given below.

Author Last Name	Author First Name	Book Title	Subject	Collection or Library	Building
Berdahl	Robert	The Politics of the Prussian Nobility	History	PCL General Stacks	Perry-Castañeda Library
Yudof	Mark	Child Abuse and Neglect	Legal Procedures	Law Library	Townes Hall
Harmon	Glynn	Human Memory and Knowledge	Cognitive Psychology	PCL General Stacks	Perry-Castañeda Library
Graves	Robert	The Golden Fleece	Greek Literature	Classics Library	Waggener Hall
Miksa	Francis	Charles Ammi Cutter	Library Biography	Library and Information Science Collection	Perry-Castañeda Library
Hunter	David	Music Publishing and Collecting	Music Literature	Fine Arts Library	Fine Arts Building
Graves	Robert	English and Scottish Ballads	Folksong	PCL General Stacks	Perry-Castañeda Library

By examining the table, we can infer that books dealing with history, cognitive psychology, and folksong are assigned to the PCL General Stacks collection; that books dealing with legal procedures are assigned to the Law Library; that books dealing with Greek literature are assigned to the Classics Library; that books dealing with library biography are assigned to the Library and Information Science Collection

(LISC);and that books dealing with music literature are assigned to the Fine Arts Library.
Moreover, we can infer that the PCL General Stacks collection and the LISC are both housed in the Perry-Castañeda Library (PCL) building; that the Classics Library is housed in Waggener Hall; and that the Law Library and Fine Arts Library are housed, respectively, in Townes Hall and the Fine Arts Building.

Thus we can see that a transitive dependency exists in the above table : any book that deals with history, cognitive psychology, or library biography will be physically housed in the PCL building (unless it is temporarily checked out to a borrower); any book dealing with legal procedures will be housed in Townes Hall; and so on. In short, if we know what subject a book deals with, we also know not only what library or collection it will be assigned to but also what building it is physically housed in.

A problem with transitive dependency is that, there is duplicated information: from three different rows we can see that the PCL General Stacks are in the PCL building. For another thing, we have possible deletion anomalies: if the Yudof book were lost and its row removed from table, we would lose the information that books on legal procedures are assigned to the Law Library and also the information the Law Library is in Townes Hall. As a third problem, we have possible insertion anomalies: if we wanted to add a chemistry book to the table, we would find that the above table nowhere contains the fact that the Chemistry Library is in Robert A.Welch Hall. As a fourth problem, we have the chance of making errors in updating: a careless data-entry clerk might add a book to the LISC but mistakenly enter Townes Hall in the building column.

To solve this problem decompose the above table into three different tables as

follows

Table A

Author Last Name	Author First Name	Book Title
Berdahl	Robert	The Politics of the Prussian Nobility
Yudof	Mark	Child Abuse and Neglect
Harmon	Glynn	Human Memory and Knowledge
Graves	Robert	The Golden Fleece

Miksa	Francis	Charles Ammi Cutter
Hunter	David	Music Publishing and Collecting
Graves	Robert	English and Scottish Ballads

Table B

Book Title	Subject
The Politics of the Prussian Nobility	History
Child Abuse and Neglect	Legal Procedures
Human Memory and Knowledge	Cognitive Psychology
The Golden Fleece	Greek Literature
Charles Ammi Cutter	Library Biography
Music Publishing and Collecting	Music Literature
English and Scottish Ballads	Folksong

Table C

Subject	Collection or Library
History	PCL General Stacks
Legal Procedures	Law Library
Cognitive Psychology	PCL General Stacks
Greek Literature	Classics Library
Library Biography	Library and Information Science Collection
Music Literature	Fine Arts Library
Folksong	PCL General Stacks

Table D

Collection or Library	Building
PCL General Stacks	Perry-Castañeda Library
Law Library	Townes Hall
Classics Library	Waggener Hall
Library and Information Science Collection	Perry-Castañeda Library
Fine Arts Library	Fine Arts Building