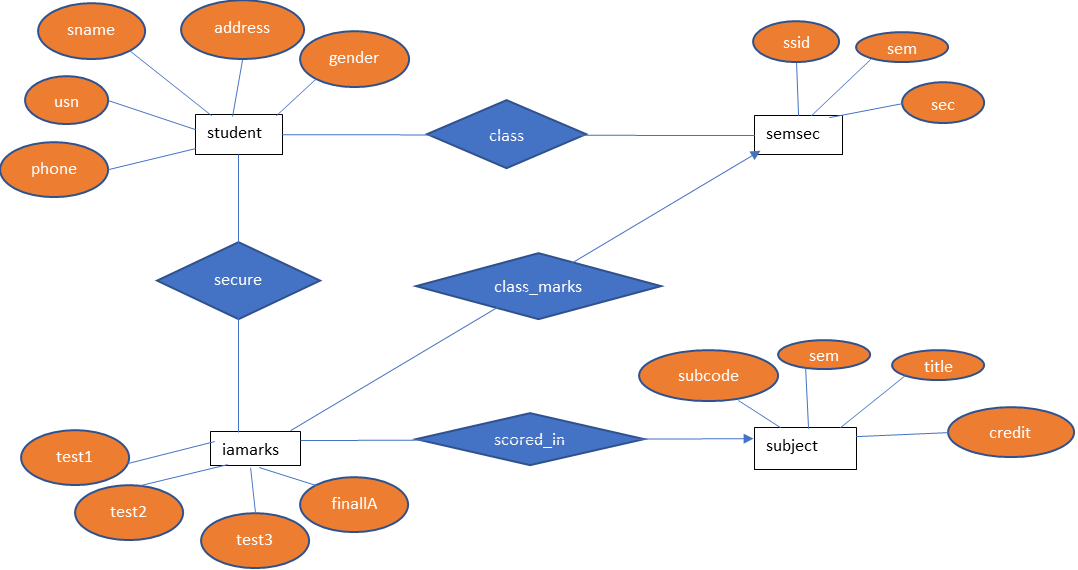
|  |  |
| --- | --- |
| 1 | Draw an E-R diagram from given entities and their attributes |
| 2 | Convert the E-R diagram into a Relational model with proper constraints. |
| 3 | Write queries to execute following DDL commands :  CREATE :Create the structure of a table with at least five columns  ALTER:Change the size of a particular column.  Add a new column to the existing table.  Remove a column from the table.  DROP: Destroy the table along with its data. |
| 4 | Write queries to execute following DML commands :  INSERT: Insert five records in each table.  UPDATE: Modify data in single and multiple columns in a table  DELETE: Delete selective and all records from a table |
| 5 | Write queries to execute following DML command :  SELECT: Retrieve the entire contents of the table.  Retrieve the selective contents (based on provided conditions) from a table.  Retrieve contents from a table based on various operators i.e. string operators, logical operators and conditional operators,Boolean operators.  Sort the data in ascending and descending order in a table on the basis of one column or more than one column. |

|  |  |
| --- | --- |
| 6 | Create table using following integrity constraints:  Primary Key  Unique Key  Not Null  Check  Default  Foreign Key |
| 7 | Write queries to execute following Aggregate functions  Sum,Avg,Count,Minimum and Maximum value of a numeric column of a table using aggregate function |
| 8 | Retrieve data from a table using alias names . |
| 9 | Retrieve data of a table using nested queries. |
| 10 | Retrieve data from more than one table using inner join, left outer, right outer and full outer joins |
| 11 | Create view from one table and more than one table. |
| 12 | Create index on a column of a table. |
| 13 | Consider the Insurance company’s Database given below. The primary keys are underlined and the data typesare specified.  PERSON(driver\_id# : string, name : string, address : string)  CAR(regno : string, model : string, year : int)  ACCIDENT(report\_number : int, acc\_date : date, location : string)  OWNS(driver\_id# : string, regno : string)  PARTICIPATED(driver\_id# : string, regno : string, report\_number : int, damage\_amount :number(10,2) )  (i) Create the above tables by properly specified the primary key and the foreign key  (ii) Enter at least five tuples for each relation  (iii) Demonstrate how you can  a. Update the damage amount for the car with a specific regno, the accident with report number 12 to 25000.  b. Add a new accident to the database.  (iv) Find the total number of people who owned cars that were involved in accident in2002.  (iv) Find the number of accident in which cars belonging to a specific models were involved |
| 14 | Consider the following schema of a library management system.Write the SQL queries for the questions given below;  **Student(Stud\_no : integer, Stud\_name: string)**  **Membership(Mem\_no: integer, Stud\_no: integer)**  **Book\_(book\_no: integer, book\_name:string, author: string)**  **lss\_rec\_(iss\_no:integer, iss\_date: date, Mem\_no: integer, book\_no: integer)**  (i) Create the tables with the appropriate integrity constraints  (ii) Insert around 10 records in each of the tables  (iii)Display all records for all tables  (iv) List all the student names with their membership numbers  (v) List all the issues for the current date with student and Book names |
| 15 | Use the relations below to write SQL queries to solve the business problems  specified.  CLIENT (clientno#,name, client\_referred\_by#)  ORDER (orderno#, clientno#, order\_date, empid#)  ORDER\_LINE (orderno#, order line number#, item\_number#, no\_of\_items,  item\_ cost,shipping\_date)  ITEM (item\_number#, item\_type, cost)  EMPLOYEE (empid#, emp\_type#, deptno, salary, firstname, lastname)  Notes:  a. Column followed by # is the primary key of the table.  b. Each client may be referred by another client. If so, the client number of  the referring client is stored in referred\_by.  c. The total cost for a particular order line = no\_of\_items \* item\_cost.c.  Write queries for the following  (i) Create all the above tables.  (ii) Insert at least five records.  (iii) Display all the rows and columns in the CLIENT table. Sort by client name in reverse alphabetical order.  (iv) Display the item number and total cost for each order line (total cost = no of items X item cost). Name the calculated column TOTAL COST.  (v) Display all the client numbers in the ORDER table. Remove duplicates.  (vi) Display the order number and client number from the ORDER table. Output the result in the format. Client <clientno> ordered <orderno>  (vii) Display full details from the ORDER\_LINE table where the item  number is (first condition) between 1 and 200 (no > or < operators) OR  the item number is greater than1000 AND (second condition) the item  cost is not in the list 1000, 2000, 3000 OR the order number is not  equal to 1000.  (viii) Display the client name and order date for all orders.  (ix) Repeat query (6) but also display all clients who have never ordered anything.  (x) Display the client name and order date for all orders using the join keywords.  (xi) Display the client name and order date for all orders using the JOIN method.  (xii) Display the client number, order date and shipping date for all orders  where the shipping date is between three and six months after the order date.  (xiii) Display the client number and name and the client number and name  of the person who referred that client.  (xiv) Display the client name in upper case only and in lower case only.  (xv) Display the second to fifth characters in each client name. |

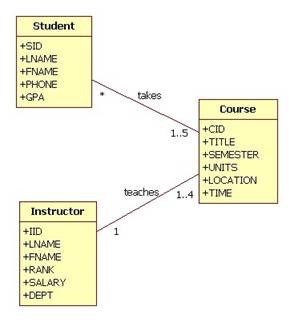
**Question 1 -:**

Draw an E-R diagram from given entities and their attributes



**Question 2 -:**

Convert the E-R diagram into a Relational model with proper constraints.



**question 3:-**

Write queries to execute following DDL commands :

* + - CREATE : Create the structure of a table with at least five columns .

CREATE DATABASE store;

USE store;

CREATE TABLE customers (

customer\_id int(11) NOT NULL AUTO\_INCREMENT,

first\_name varchar(50) NOT NULL,

last\_name` varchar(50) NOT NULL,

birth\_date date DEFAULT NULL,

phone varchar(50) DEFAULT NULL,

address varchar(50) NOT NULL,

city varchar(50) NOT NULL,

state char(2) NOT NULL,

points int(11));

INSERT INTO customers VALUES (1,'Babara','MacCaffrey','1986-03-28','781-932-9754','0 Sage Terrace','Waltham','MA',2273);

INSERT INTO customers VALUES (2,'Ines','Brushfield','1986-04-13','804-427-9456','14187 Commercial Trail','Hampton','VA',947);

INSERT INTO customers VALUES (3,'Freddi','Boagey','1985-02-07','719-724-7869','251 Springs Junction','Colorado Springs','CO',2967);

INSERT INTO customers VALUES (4,'Ambur','Roseburgh','1974-04-14','407-231-8017','30 Arapahoe Terrace','Orlando','FL',457);

INSERT INTO customers VALUES (5,'Clemmie','Betchley','1973-11-07',NULL,'5 Spohn Circle','Arlington','TX',3675);

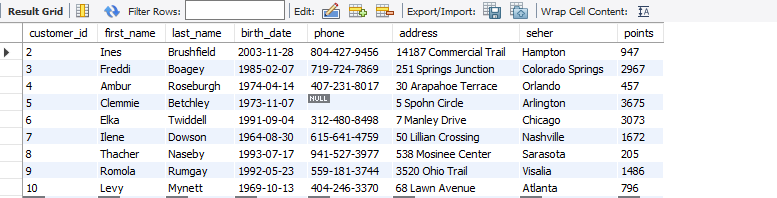
INSERT INTO customers VALUES (6,'Elka','Twiddell','1991-09-04','312-480-8498','7 Manley Drive','Chicago','IL',3073);

INSERT INTO customers VALUES (7,'Ilene','Dowson','1964-08-30','615-641-4759','50 Lillian Crossing','Nashville','TN',1672);

INSERT INTO customers VALUES (8,'Thacher','Naseby','1993-07-17','941-527-3977','538 Mosinee Center','Sarasota','FL',205);

INSERT INTO customers VALUES (10,'Levy','Mynett','1969-10-13','404-246-3370','68 Lawn Avenue','Atlanta','GA',796);

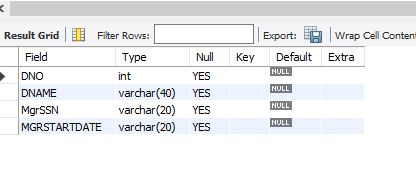
select \* from customers;



* ALTER: Change the size of a particular column.

alter TABLE department

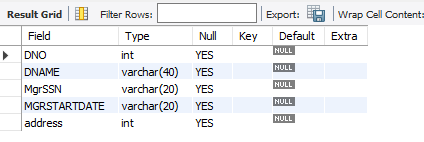
MODIFY COLUMN DNAME varchar(40);



Add a new column to the existing table.

ALTER TABLE department

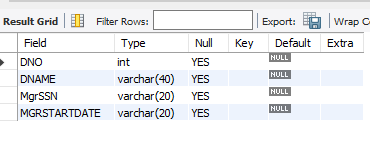
add COLUMN address int;



Remove a column from the table.

ALTER TABLE department

drop COLUMN address;



DROP: Destroy the table along with its data.

use collage;

drop table test;



**question 4:-**

Write queries to execute following DML commands :

INSERT: Insert five records in each table.

UPDATE: Modify data in single and multiple columns in a table

DELETE: Delete selective and all records from a table.

Insert five records in table.

Use company;

INSERT into DLOCATION VALUES(207,345);

INSERT into DLOCATION VALUES(208,123);

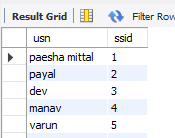
INSERT into DLOCATION VALUES(209,342);

INSERT into DLOCATION VALUES(210,978);

Modify data in single and multiple columns in a table

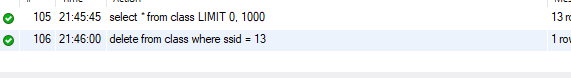
use collages;

update class set usn = "payal" where ssid = 2;



DELETE: Delete selective and all records from a table.

Use colleges;

delete from class where ssid = 13;  


**question 5:-**

Write queries to execute following DML command :

SELECT: Retrieve the entire contents of the table.

Retrieve the selective contents (based on provided conditions) from a table.

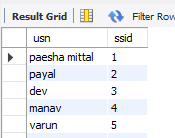
Retrieve contents from a table based on various operators

i.e. string operators, logical operators and conditional operators, Boolean operators.

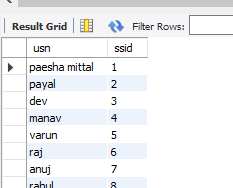
Sort the data in ascending and descending order in a table on the basis of one column or more than one column.

1. SELECT: Retrieve the entire contents of the table.

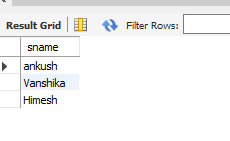
select \* from class;

\ 

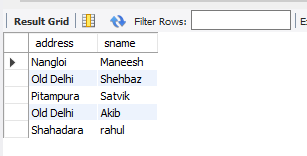
1. Retrieve the selective contents (based on provided conditions) from a table.
2. SELECT \* FROM students WHERE usn <=9 AND Gender = "M";



1. SELECT sname FROM students WHERE gender ="f" || address ="shastri nagar";

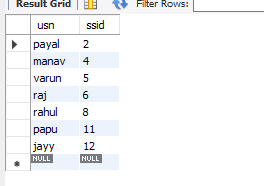


1. SELECT address, sname FROM students WHERE usn BETWEEN 5 AND 9;

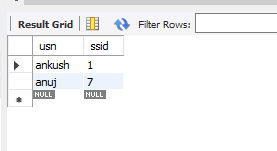


1. Retrieve contents from a table based on various operators

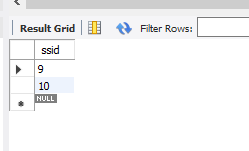
select \* from class where usn like '\_a%';



SELECT \* FROM class WHERE usn LIKE 'a\_%';

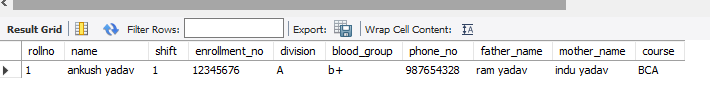


SELECT ssid FROM class WHERE USN LIKE 's\_%';

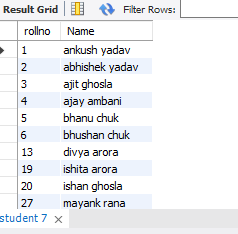


Logical Operators

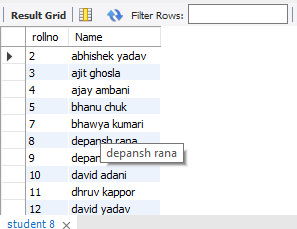
SELECT \* FROM student WHERE rollno <= 3 AND blood\_group = "b+";



SELECT rollno, Name FROM student WHERE course = "bca" || rollno <7;



SELECT rollno, Name FROM student WHERE not(rollno = 6 OR rollno = 1);

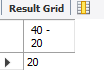


Arithmetic Operators ( + - \* / % )

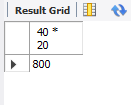
SELECT 40 + 20;



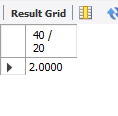
SELECT 40 - 20;



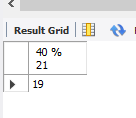
SELECT 40 \* 20;



SELECT 40 / 20;

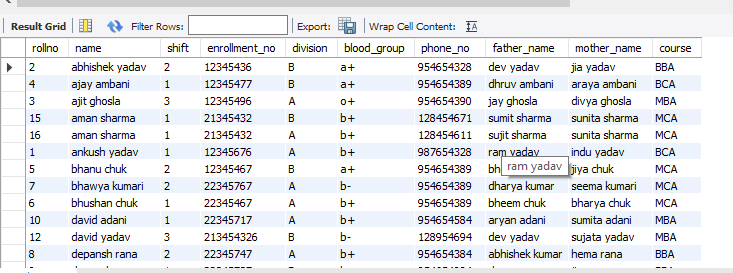


SELECT 40 % 21;



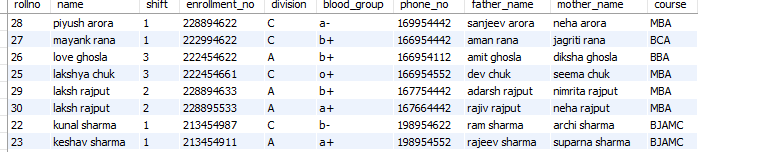
Sort Table In Ascending Order by name basis of one column or more than one column.

SELECT \*from best ORDER BY Name ,Program;



Sort Table In descending Order by name basis of one column or more than one column.

SELECT \*from student ORDER BY Name DESC;



**Question :-6**

Create table using following integrity constraints:

1. Primary Key
2. Unique Key
3. Not Null
4. Check
5. Foreign Key
6. Default

**1st table.**

**create table id( name varchar(20)not null unique,age int not null check(age >=18),rollno int unique not null,city varchar(20) not null default 'patna');**

**insert into ID values("sheela", 20,5,default);**

2nd table

CREATE Table Personal(

ID INT NOT NULL UNIQUE,

Code\_NO INT UNIQUE,

FNAME VARCHAR(60) NOT NULL,

Age INT NOT NULL check (Age >=18),

Phone\_no VARCHAR(10) UNIQUE,

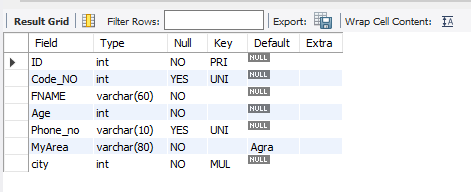
MyArea VARCHAR(80) NOT null DEFAULT 'Agra',

city INT NOT NULL,

PRIMARY KEY(ID),

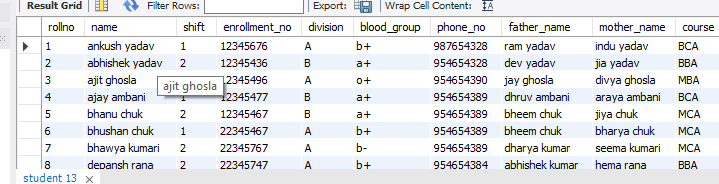
FOREIGN KEY (City)REFERENCES id(rollno)

);

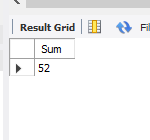


**Question :-7** Write queries to execute following Aggregate functions Sum, Avg, Count, Minimum and Maximum value of a numeric column of a table using aggregate function.

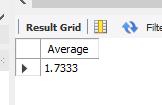
SELECT \*from student;



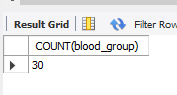
SELECT SUM(shift) As Sum FROM student;



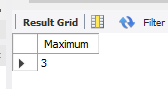
SELECT AVG(shift) As Average FROM student;



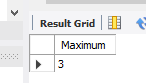
SELECT COUNT(blood\_group) FROM student;



SELECT MAX(shift) As Maximum FROM student;

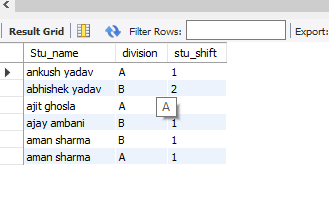


SELECT MIN(shift) As Minimum FROM student;

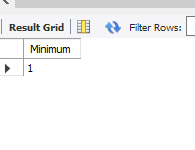


**Question** **:-8** Retrieve data from a table using alias names .

SELECT name as Stu\_name, division,shift as stu\_shift FROM student WHERE Name LIKE 'a%';



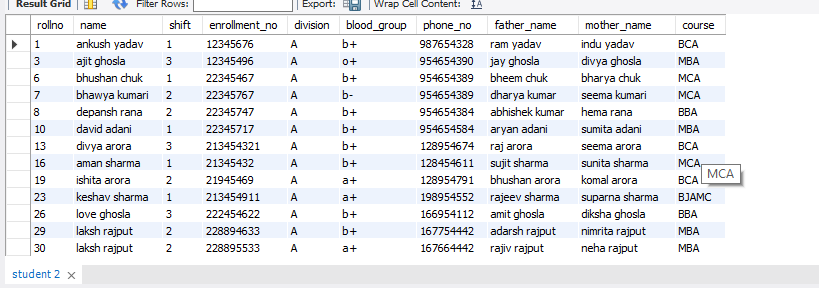
SELECT MIN(shift) As Minimum FROM student;



**Question :- 9**

Retrieve data of a table using nested queries.

SELECT \* FROM student WHERE division = all (SELECT division FROM student WHERE division = 'a');

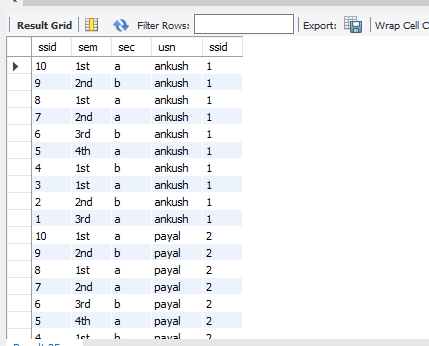


**Question :-10**

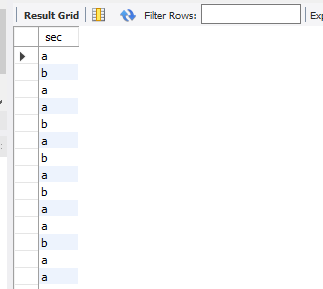
Retrieve data from more than one table using inner join, left outer, right outer and full outer joins.

1. **Inner join**

select \* from semsec inner join class;

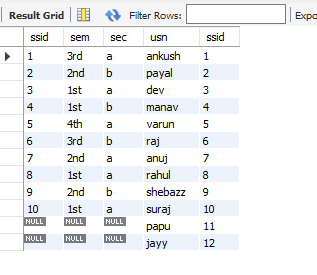


select sec from semsec inner join class;



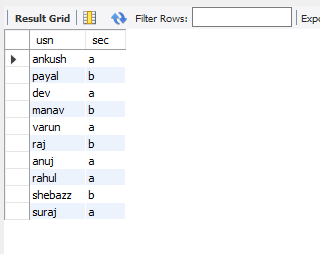
1. **Right outer join**

select \* from semsec right join class on semsec.ssid= class.ssid;

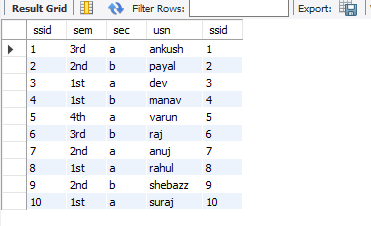


1. **Left outer join.**

select usn,sec from semsec left join class on semsec.ssid= class.ssid;

ssid

select \* from semsec left join class on semsec.ssid= class.ssid;



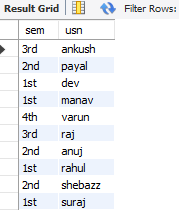
**Question 11 -:**

Create view from one table and more than one table.

1. From one table.

create view informatiom as select sem, usn from semsec, class where semsec.ssid= class.ssid;

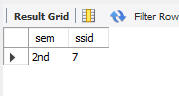
select \* from informatiom;



1. From more than one table.

create view showw as select sem,ssid from semsec where ssid = 7;

select \* from showw;

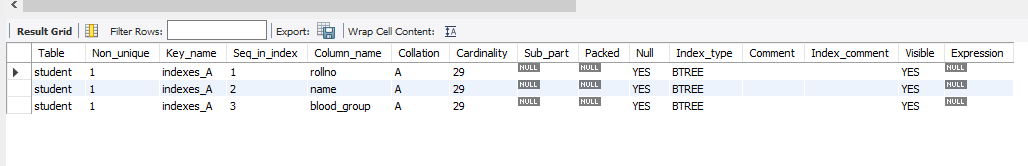


**Question 12 :**

Create index on a column of a table.

create index indexes\_A on student(rollno, name, blood\_group);

show index from student;



**Question : 13**

Consider the Insurance company’s Database given below.

The primary keys are underlined and the data typesare specified.

PERSON(driver\_id# : string, name : string, address : string)

CAR(regno : string, model : string, year : int)

ACCIDENT(report\_number : int, acc\_date : date, location : string)

OWNS(driver\_id# : string, regno : string)

PARTICIPATED(driver\_id# : string, regno : string, report\_number : int, damage\_amount :number(10,2) )

1. Create the above tables by properly specified the primary key and the foreign key

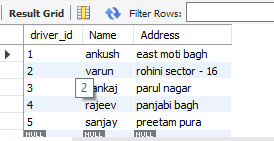
(ii) Enter at least five tuples for each relation

1. Demonstrate how you can a. Update the damage amount for the car with a specific regno, the accident with report number 12 to 25000. b. Add a new accident to the database.
2. Find the total number of people who owned cars that were involved in accident in2002.
3. Find the number of accident in which cars belonging to a specific models were involved

create database lic;

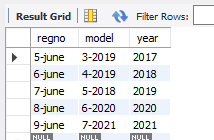
create table person(id varchar(20),name varchar(45),address varchar(45));

select \* from person;



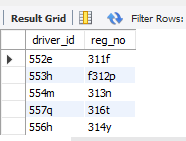
create table car(reg\_no varchar(20),model varchar(45),year int );

select \* from car;



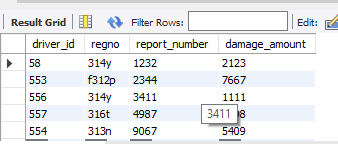
create table ownss(driver\_id varchar(20),reg\_no varchar(20) );

select \* from ownss;



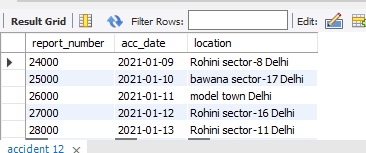
create table participated(driver\_id varchar(50),regno varchar(45),report\_number int,damage\_amount int );

select \* from participated;



create table accident(report\_no int,acc\_date int,location varchar(40) );

select \* from accident;



(i) Create the above tables by properly specified the primary key and the foreign key

ALTER TABLE  accident ADD FOREIGN KEY (report\_number) REFERENCES participated(report\_number);

(ii) (A) Update the damage amount for the car with a specific regno, the

accident with report number 12 to 25000.

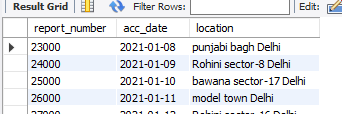
UPDATE participated SET damage\_amount=damage\_amount+1000

WHERE report\_number BETWEEN 12 AND 25000 AND regno ='314y';



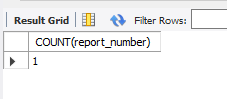
(ii) (B) Add a new accident to the database.

INSERT INTO accident VALUES(23000,'2021-01-8',"punjabi bagh Delhi");



(iii) Find the total number of people who owned cars that were involved in accident in2002.

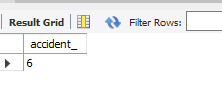
SELECT COUNT(report\_number) FROM accident WHERE acc\_date ='2002-01-10';

****

(iv) Find the number of accident in which cars belonging to a specific models were involved.

Select count(report\_number) as accident\_ from ACCIDENT where report\_number IN

(select report\_number from car where model='3-2019' );



**Question:- 14**

Consider the following schema of a library\_management system.Write the SQL

queries for the questions given below;

Student(Stud\_no : integer, Stud\_name: string)

Membership(Mem\_no: integer, Stud\_no: integer)

Book\_(book\_no: integer, book\_name:string, author: string)

lss\_rec\_(iss\_no:integer, iss\_date: date, Mem\_no: integer, book\_no: integer)

(i) Create the tables with the appropriate integrity constraints

(ii) Insert around 10 records in each of the tables

(iii)Display all records for all tables

(iv)List all the student names with their membership numbers

(v) List all the issues for the current date with student and Book names

(vi) List the details of students who borrowed book whose author is Elmarsi & Navathe

(vii) Give a count of how many books have been bought by each student

(viii) Give a list of books taken by student with stud\_no as 1005

(ix) Delete the List of books details which are issued as of today

(x) Create a view which lists out the iss\_no, iss \_date, stud\_name, book name

CREATE DATABASE library\_management;

use library\_management;

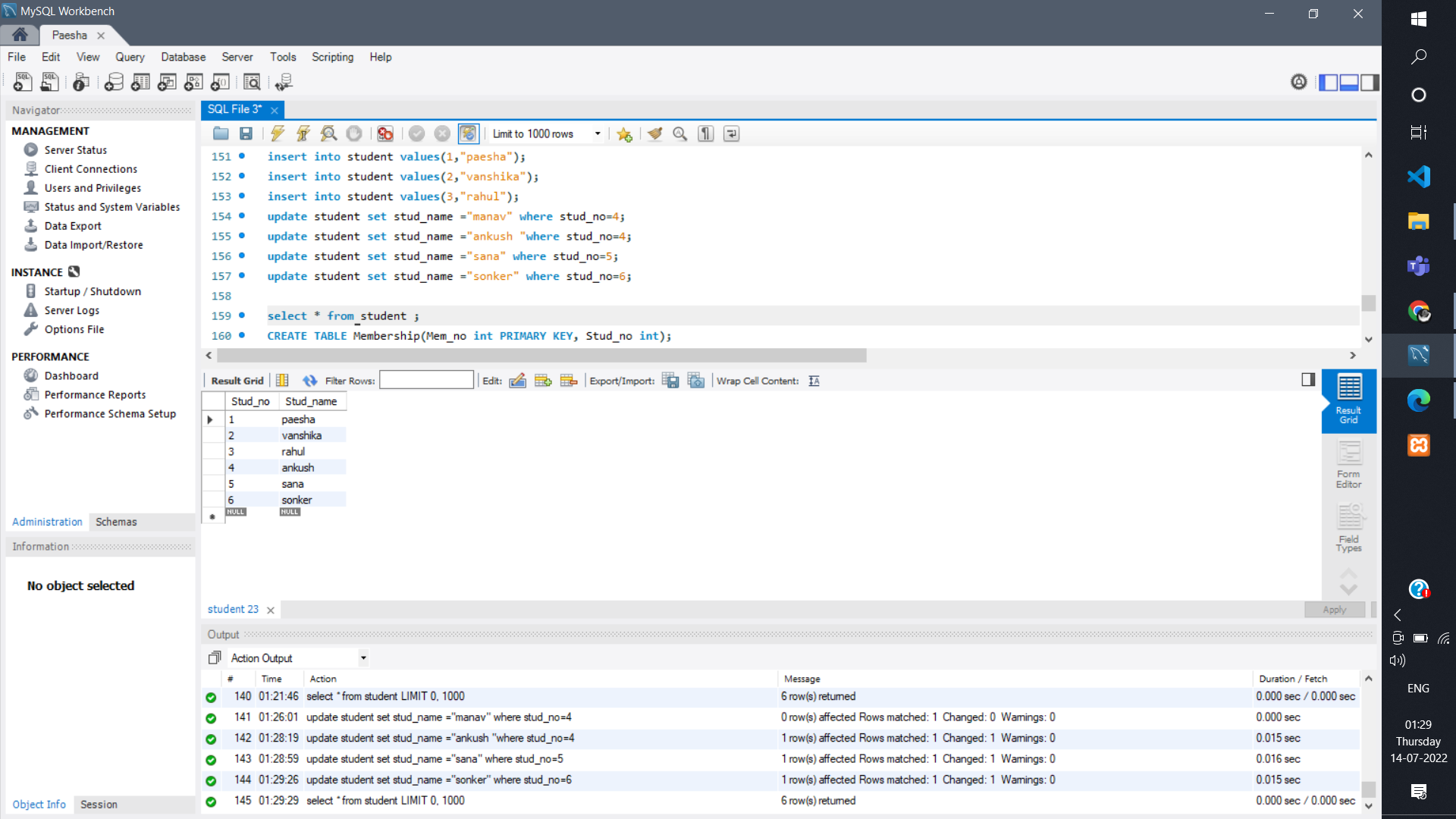
**(i) Create the tables with the appropriate integrity constraints**

**(ii) Insert around 10 records in each of the tables**

**(iii)Display all records for all tables**

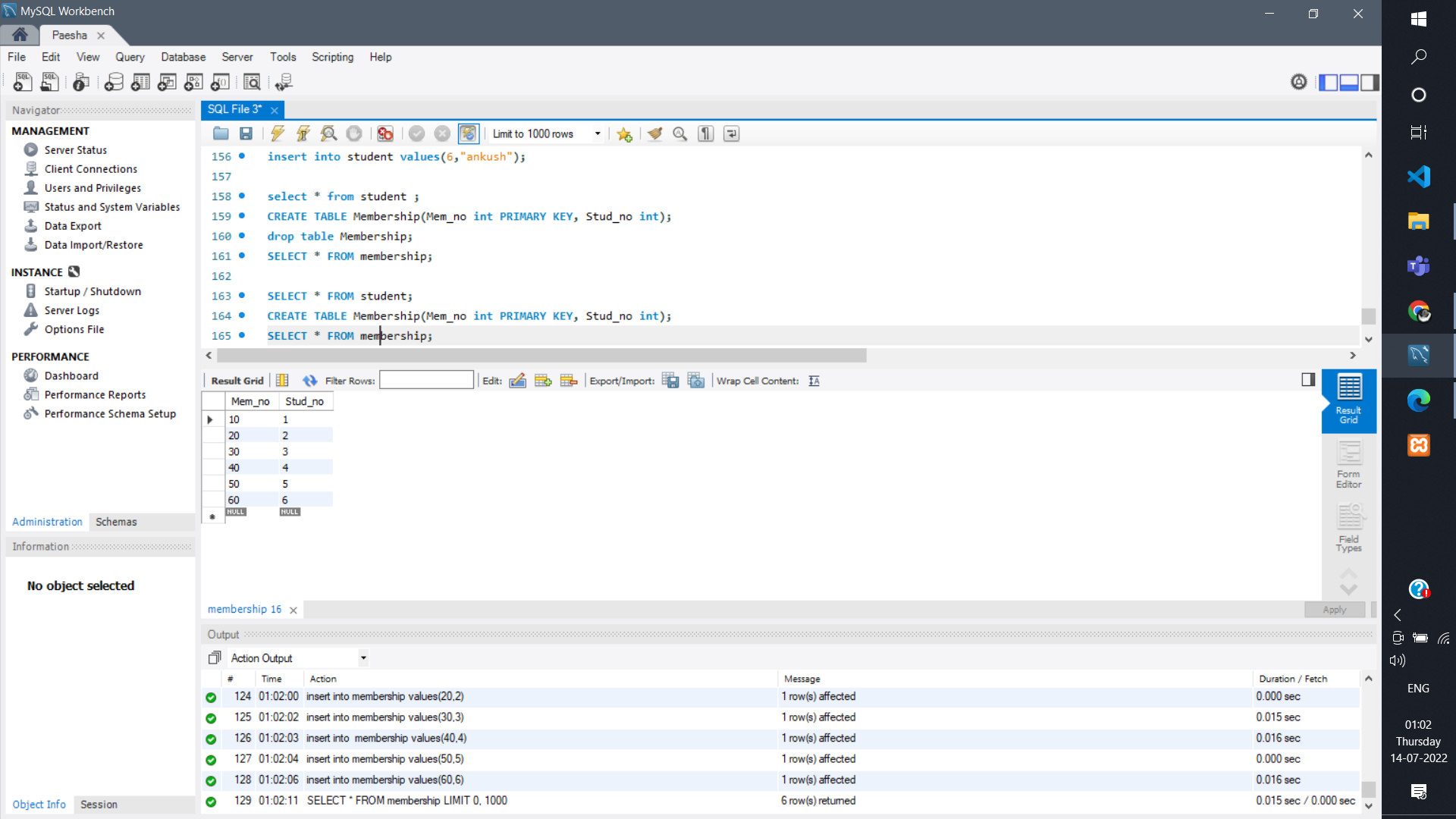
CREATE TABLE Student(Stud\_no int PRIMARY KEY, Stud\_name VARCHAR(60));

SELECT \* FROM student;



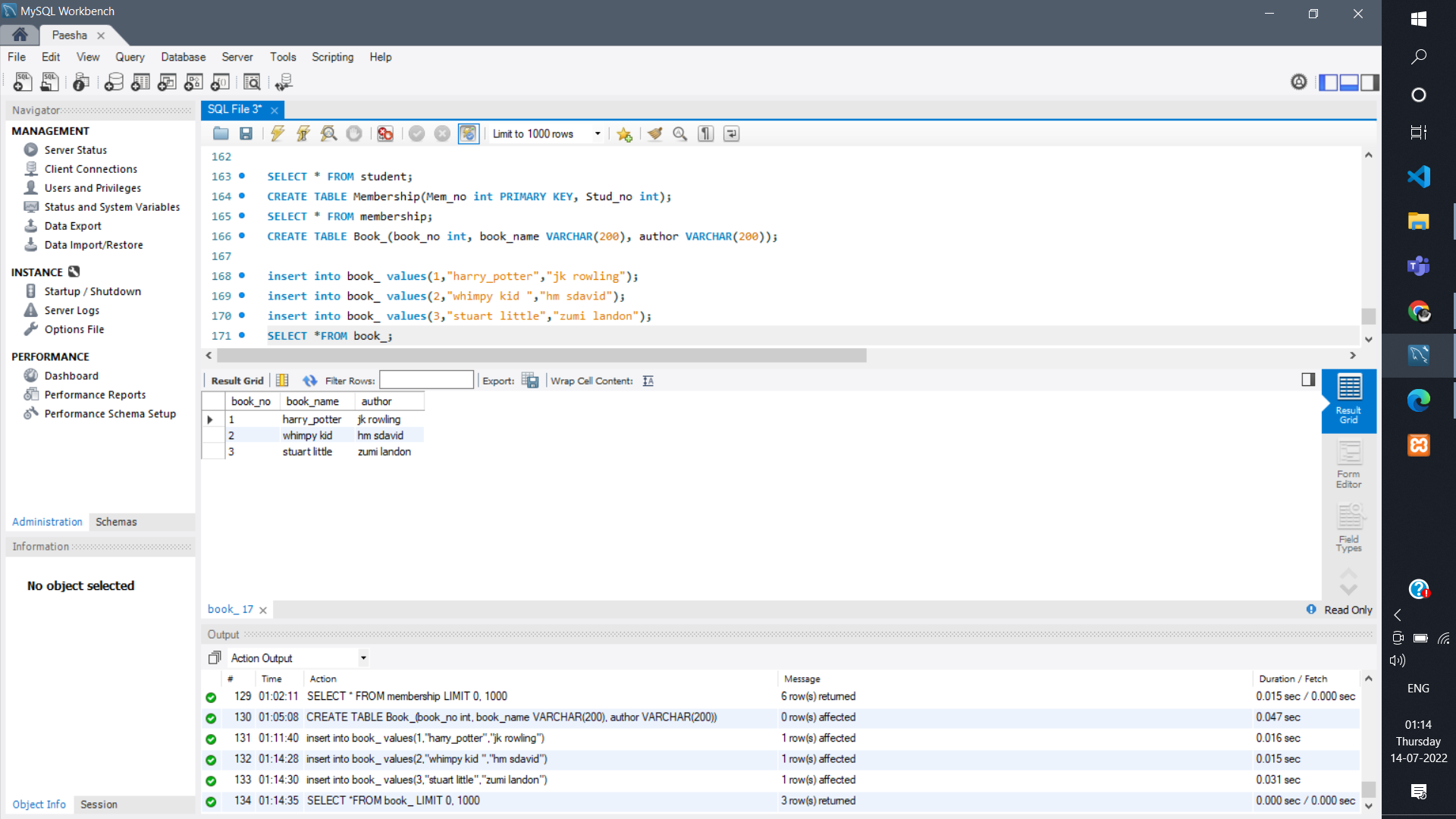
CREATE TABLE Membership(Mem\_no int PRIMARY KEY, Stud\_no int);

SELECT \* FROM membership;



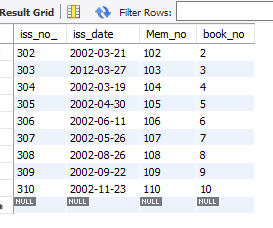
CREATE TABLE Book\_(book\_no int, book\_name VARCHAR(200), author VARCHAR(200));

SELECT \*FROM book\_;



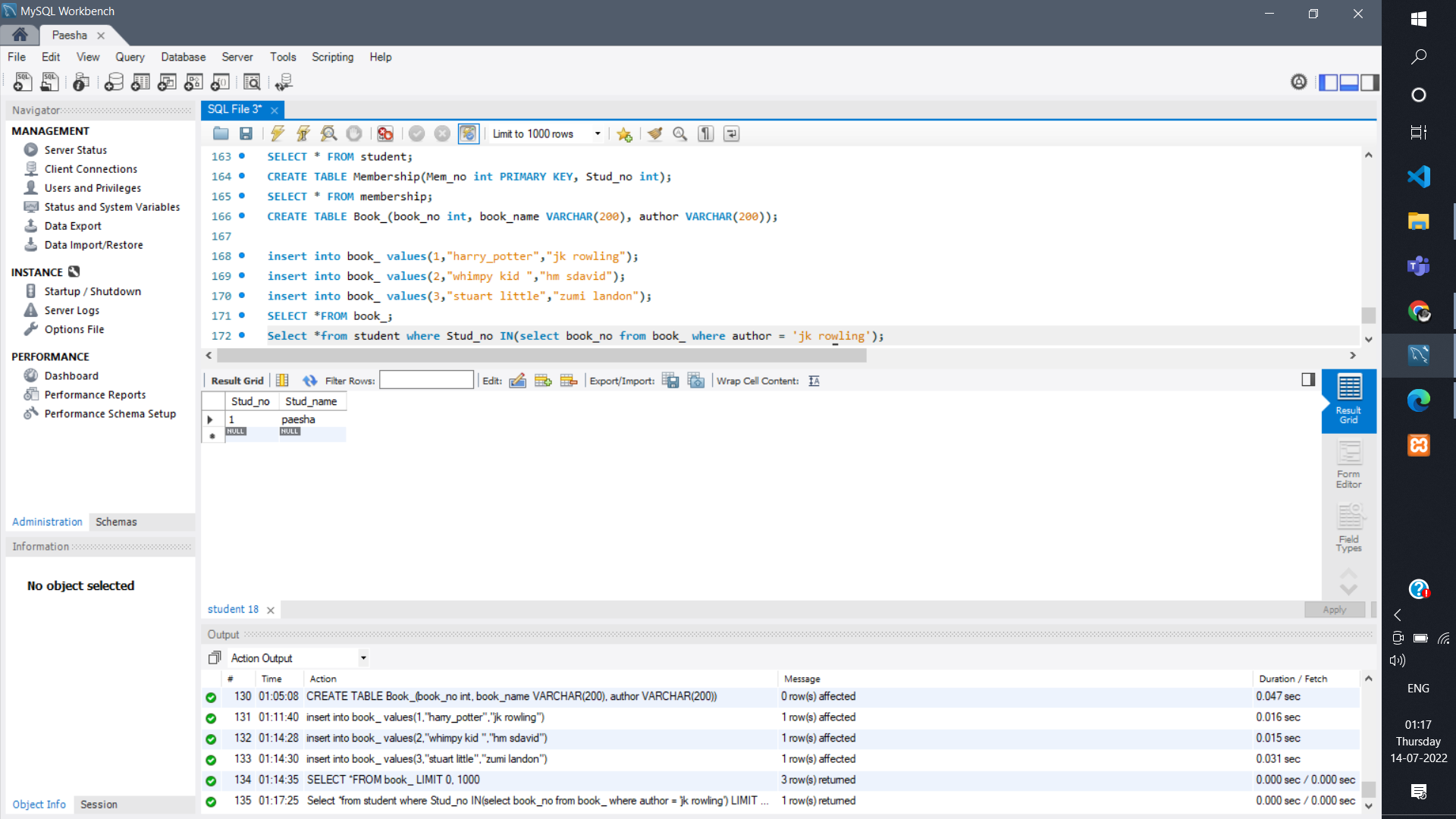
CREATE TABLE lss\_rec(iss\_no\_ int PRIMARY KEY, iss\_date date, Mem\_no int, book\_no int);

select \* from lss\_rec;



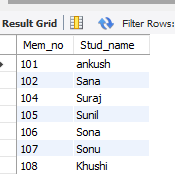
(vi)List the details of students who borrowed book whose author is Elmarsi & Navathe.

Select \*from student where Stud\_no IN(select book\_no from book\_ where author = 'Elmarsi & Navathe');



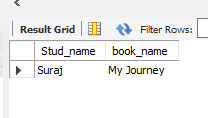
(iv)List all the student names with their membership numbers.

SELECT Mem\_no,Stud\_name FROM membership JOIN student WHERE membership.Stud\_no =student.Stud\_no;



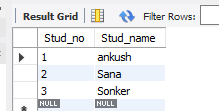
1. List all the issues for the current date with student and Book names

select Stud\_name, book\_name from student, book\_, lss\_rec ,membership where iss\_date= '2002-03-19' AND student.Stud\_no= membership.Stud\_no AND  lss\_rec.Mem\_no= membership.Mem\_no AND lss\_rec.book\_no= book\_.book\_no;



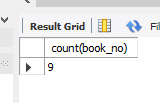
(vi)List the details of students who borrowed book whose author is zumi landon

Select \*from student where Stud\_no IN(select book\_no from book\_ where author = 'zumi landon’);



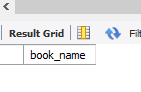
(vii) Give a count of how many books have been bought by each student

select count(book\_no) from lss\_rec;



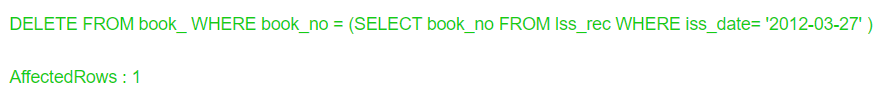
(viii) Give a list of books taken by student with stud\_no as 1005

select book\_name from book\_ where book\_no=(select book\_no from lss\_rec where Mem\_no=(select Mem\_no from membership where Stud\_no= 1005));



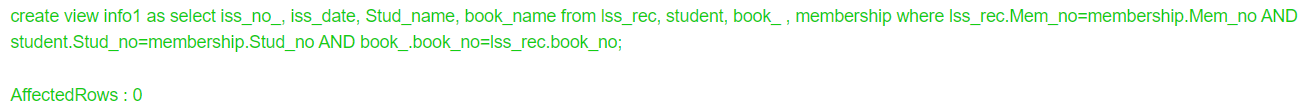
(ix) Delete the List of books details which are issued as of today

DELETE FROM book\_ WHERE book\_no = (SELECT book\_no FROM lss\_rec WHERE iss\_date= '2012-03-27' );

****

(x) Create a view which lists out the iss\_no, iss \_date, stud\_name, book name.

create view info1 as select iss\_no\_, iss\_date, Stud\_name, book\_name from lss\_rec, student, book\_ , membership where lss\_rec.Mem\_no=membership.Mem\_no AND student.Stud\_no=membership.Stud\_no AND book\_.book\_no=lss\_rec.book\_no;

****

**Question – : 15**

Use the relations below to write SQL queries to solve the business problems

specified.

CLIENT (clientno#,name, client\_referred\_by#)

ORDER (orderno#, clientno#, order\_date, empid#)

ORDER\_LINE (orderno#, order line number#, item\_number#, no\_of\_items,

item\_ cost,shipping\_date)

ITEM (item\_number#, item\_type, cost)

EMPLOYEE (empid#, emp\_type#, deptno, salary, firstname, lastname)

Notes:

a. Column followed by # is the primary key of the table.

b. Each client may be referred by another client. If so, the client number of

the referring client is stored in referred\_by.

c. The total cost for a particular order line = no\_of\_items \* item\_cost.c.

Write queries for the following

(i) Create all the above tables.

(ii) Insert at least five records.

(iii) Display all the rows and columns in the CLIENT table. Sort by client name in reverse alphabetical order.

(iv) Display the item number and total cost for each order line (total cost = no of items X item cost). Name the calculated column TOTAL COST.

(v) Display all the client numbers in the ORDER table. Remove duplicates.

(vi) Display the order number and client number from the ORDER table. Output the result in the format. Client <clientno> ordered <orderno>

(vii) Display full details from the ORDER\_LINE table where the item

number is (first condition) between 1 and 200 (no > or < operators) OR

the item number is greater than1000 AND (second condition) the item

cost is not in the list 1000, 2000, 3000 OR the order number is not

equal to 1000.

(viii) Display the client name and order date for all orders.

(ix) Repeat query (6) but also display all clients who have never ordered anything.

(x) Display the client name and order date for all orders using the join keywords.

(xi) Display the client name and order date for all orders using the JOIN method.

(xii) Display the client number, order date and shipping date for all orders

where the shipping date is between three and six months after the order date.

(xiii) Display the client number and name and the client number and name

of the person who referred that client.

(xiv) Display the client name in upper case only and in lower case only.

(xv) Display the second to fifth characters in each client name.

CREATE Table CLIENT (clientno int PRIMARY KEY,name VARCHAR(50) , client\_referred\_by int);

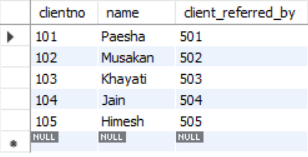
INSERT INTO client VALUES (101,"paesha ",501);

INSERT INTO client VALUES (102,"muskan ",502);

INSERT INTO client VALUES (103,"khayati",503);

INSERT INTO client VALUES (104,"jain",504);

INSERT INTO client VALUES (105,"himesh505);



CREATE Table ORDER\_(orderno int PRIMARY KEY, clientno int NOT NULL, order\_date DATE, empid INT NOT NULL);

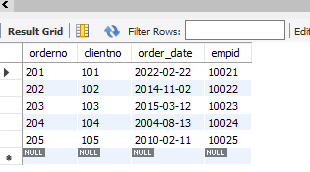
INSERT INTO order\_ VALUES (201,101,"2022-02-22",10021);

INSERT INTO order\_ VALUES (202,102,"2014-11-02",10022);

INSERT INTO order\_ VALUES (203,103,"2015-03-12",10023);

INSERT INTO order\_ VALUES (204,104,"2004-08-13",10024);

INSERT INTO order\_ VALUES (205,105,"2010-02-11",10025);



CREATE TABLE ORDER\_LINE (orderno int PRIMARY KEY, order\_line\_ int, item\_number int, no\_of\_items int, item\_cost INT ,shipping\_date DATE);

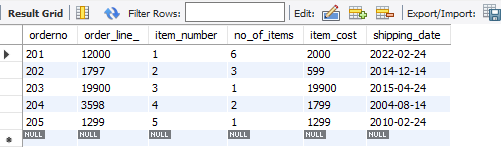
INSERT INTO order\_line VALUES (201,12000,001,6,2000,"2022-02-24");

INSERT INTO order\_line VALUES (202,1797,002,3,599,"2014-12-14");

INSERT INTO order\_line VALUES (203,19900,003,1,19900,"2015-04-24");

INSERT INTO order\_line VALUES (204,3598,004,2,1799,"2004-08-14");

INSERT INTO order\_line VALUES (205,1299,005,1,1299,"2010-02-24");



CREATE TABLE ITEM (item\_number int PRIMARY KEY, item\_type VARCHAR(50), cost INT);

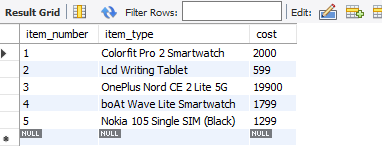
INSERT INTO item VALUES (001,"Colorfit Pro 2 Smartwatch",2000);

INSERT INTO item VALUES (002,"Lcd Writing Tablet",599);

INSERT INTO item VALUES (003,"OnePlus Nord CE 2 Lite 5G",19900);

INSERT INTO item VALUES (004,"boAt Wave Lite Smartwatch",1799);

INSERT INTO item VALUES (005,"Nokia 105 Single SIM (Black)",1299);



CREATE TABLE EMPLOYEE (empid int PRIMARY KEY, emp\_type VARCHAR(50), deptno int , salary BIGINT, firstname VARCHAR(50), lastname VARCHAR(50));

INSERT INTO EMPLOYEE VALUES (10021 ,"Full-time employee", 901, 120000, "paesha", "mittal");

INSERT INTO employee VALUES (10022,"Part-time employee",902,12000,"muskan","singh");

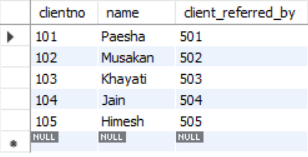
INSERT INTO employee VALUES (10025,"Part-time employee",905,143000,"khayati","luthra");

INSERT INTO employee VALUES (10024,"Part-time employee",904,124000,"sarthak","jainl");

INSERT INTO employee VALUES (10023,"Temporary employee",903,122000,"himesh","mandi");

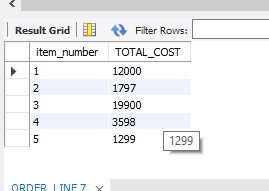
(iii) Display all the rows and columns in the CLIENT table. Sort by client name in reverse alphabetical order.

SELECT \* FROM client ORDER BY name DESC;



(iv) Display the item number and total cost for each order line (total cost = no of items X item cost). Name the calculated column TOTAL COST.

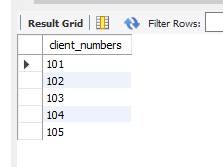
SELECT item\_number,order\_line\_ AS TOTAL\_COST FROM ORDER\_LINE;



3

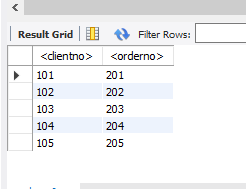
1. Display all the client numbers in the ORDER table. Remove duplicates.

SELECT DISTINCT clientno AS client\_numbers FROM client;



1. Display the order number and client number from the ORDER table. Output the result in the format. Client <clientno> ordered <orderno>

SELECT clientno AS "<clientno>",orderno AS "<orderno>" from order\_;



(vii) Display full details from the ORDER\_LINE table where the item

number is (first condition) between 1 and 200 (no > or < operators)

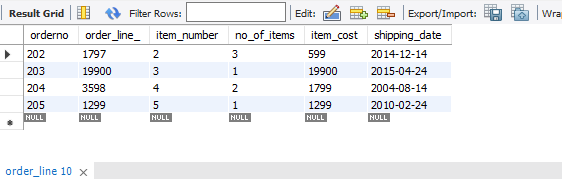
OR the item number is greater than1000 AND

(second condition) the item

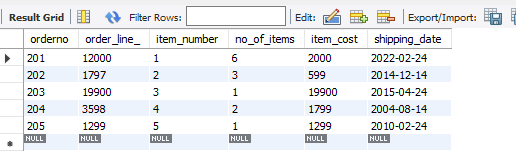
cost is not in the list 1000, 2000, 3000 OR the order number is not

equal to 1000.

SELECT \* FROM order\_line where item\_number BETWEEN 2 AND 5 OR item\_number>2;

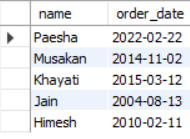


SELECT \* FROM order\_line where item\_cost NOT IN(1000,2000,3000) OR NOT(orderno=1000);



1. Display the client name and order date for all orders.

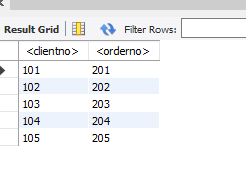
SELECT name,order\_date FROM client,order\_ WHERE client.clientno = order\_.clientno ;



(6) Display the order number and client number from the ORDER table. Output the result in the format. Client <clientno> ordered <orderno>

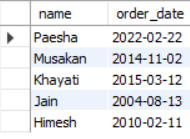
(ix) Repeat query (6) but also display all clients who have never ordered anything.

SELECT clientno AS "<clientno>",orderno AS "<orderno>" from order\_;



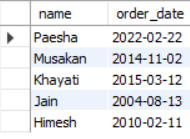
1. Display the client name and order date for all orders using the join keywords.

SELECT name,order\_date FROM client JOIN order\_ ON client.clientno = order\_.clientno ;



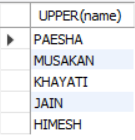
1. Display the client name and order date for all orders using the JOIN method.

SELECT name,order\_date FROM client INNER JOIN order\_ ON client.clientno = order\_.clientno ;

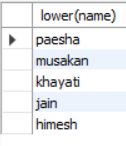


(xiv) Display the client name in upper case only and in lower case only.

SELECT UPPER(name) FROM client;



SELECT lower(name) FROM client;



(xv) Display the second to fifth characters in each client name.

SELECT SUBSTRING(name,2,5) FROM client;

