# PRACTICAL FILE

# OF

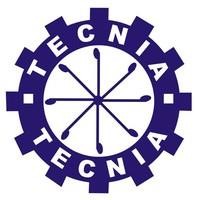
## “Practical-VI DBMS Lab”

**(BCA 176)**

submitted in the partial fulfillment of the requirement for the award of degree of

#### **BACHELOR**

**OF**

**COMPUTER APPLICATIONS**

**SUBMITTED BY:** **BHAWYA GARG**

**(02417002021)**

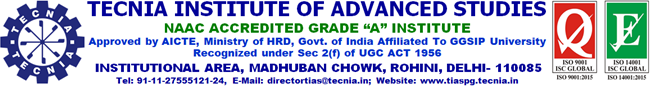
**Semester – 2nd**

**Shift – 1st**

**Batch: 2021-2024**

**SUBMITTED TO:**

**Ms.AMBOOJ YADAV**



**DEPARTMENT OF COMPUTER APPLICATIONS**

**PRACTICAL FILE**

**OF**

**WEB BASED PROGRAMMING**

**INDEX**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.NO.** | **TOPIC** | **DATE** | **REMARKS** |  |
| **1.** | Draw an E-R diagram from given entities and their attributes. |  |  |  |
| **2.** | Convertthe 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 a table using following integrity constraints:  Primary Key, Unique Key, Not NULL ,Check ,Default. |  |  |  |
| **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 a index on a coloumn of a table. |  |  |  |
| **13.** | Consider the Insurance company’s Database given below. The primary keys are |  |  |  |
| **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  (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 |  |  |  |
| **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:   1. Column followed by # is the primary key of the table. 2. Each client may be referred by another client. If so, the client number of the referring client is stored in referred\_by. 3. The total cost for a particular order line = no\_of\_items \* item\_cost.c.   Write queries for the following   1. Create all the above tables. 2. Insert at least five records. 3. Display all the rows and columns in the CLIENT table. Sort by client name in reverse alphabetical order. 4. 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. 5. Display all the client numbers in the ORDER table. Remove duplicates. 6. Display the order number and client number from the ORDER table. Output the result in the format. Client <clientno> ordered <orderno> 7. 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. 8. Display the client name and order date for all orders. 9. Repeat query (6) but also display all clients who have never ordered anything. 10. Display the client name and order date for all orders using the join keywords. 11. Display the client name and order date for all orders using the JOIN method. 12. 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. |  |  |  |

**Experiment 1**

**AIM:-** Draw an E-R diagram from given entities and their attributes.

**Experiment 2**

**Aim:-** Convertthe E-R diagram into a relational model with proper constraints.

**Experiment 3**

**AIM:- 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.**

**Procedure: -**

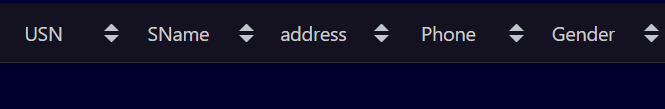
1. Create a database .
   1. **create DATABASE assignment;**
2. **Create**



Table and Insert five coloumn .

* 1. **CREATE TABLE student(USN INT, SName VARCHAR(14),**

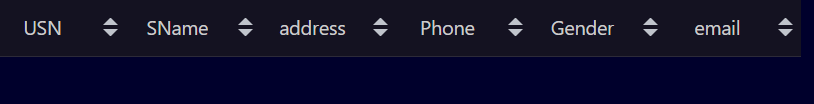
**address VARCHAR(40),Phone VARCHAR(12), Gender VARCHAR(6));**

1. To view the table:
   1. SELECT \* FROM *student;*
   2. ****
2. **ALTER**

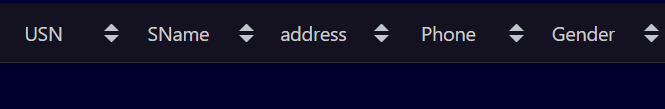
Change the size of a particular column.

1. **ALTER TABLE student modify Gender varchar(10);**

Add a new column to existing Table.

* 1. **ALTER Table student ADD email varchar(20);**
  2. ****

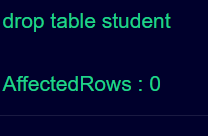
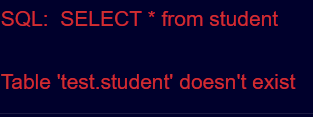
1. Delete a column from Existing Table;
   1. **alter Table student DROP column email;**

****

1. **DROP:**

Destroy TABLE Along with its data

* 1. **drop table student;**



**Experiment-4**

**AIM:-** **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**

**Procedure: -**

**INSERT**

1. **Create five table** 
   1. **STUDENT**

**CREATE TABLE student(Eno INT, firstName VARCHAR(14), lastname VARCHAR(14),address VARCHAR(40),city VARCHAR(10));**

* **INSERT INTO student VALUES**

**(0012021,'Kamar','Burari',755762302,'M');**

* **INSERT INTO student VALUES**

**(0022021,'Shruti','Rohini',905862302,'M');**

* **INSERT INTO student VALUES (0032021,'Muskan','Aadarsh\_Nagar',655561302,'M');**
* **INSERT INTO student VALUES (0042021,'Isha','Rohini',955961302,'M');**
* **INSERT INTO student VALUES (0052021,'Vanshu','Ghaziabad',855162302,'M');**
  1. **SEMSEC**

**CREATE table SEMSEC(**

**SSID INT,SEM INT, SEC VARCHAR(6));**

* **INSERT into SEMSEC VALUES(1,2,'A');**
* **INSERT into SEMSEC VALUES(2,2,'A');**
* **INSERT into SEMSEC VALUES(3,2,'B');**
* **INSERT into SEMSEC VALUES(4,3,'A');**
* **INSERT into SEMSEC VALUES(5,3,'A');**
  1. **CLASS**

**CREATE table CLASS(**

**USN INT,SSID INT);**

* **INSERT into CLASS VALUES(12021,1);**
* **INSERT into CLASS VALUES(22021,2);**
* **INSERT into CLASS VALUES(32021,3);**
* **INSERT into CLASS VALUES(42021,4);**
* **INSERT into CLASS VALUES(52021,5);**
  1. **SUBJECT**

**CREATE table SUBJECT(**

**SUBCODE INT,Title VARCHAR(10),SEM INT, Credits INT**

**);**

* **INSERT into subject VALUES(012,'DBMS',2,2);**
* **INSERT into SUBJECT VALUES(013,'WEBD',2,2);**
* **INSERT into SUBJECT VALUES(014,'DSA',2,2);**
* **INSERT into SUBJECT VALUES(015,'ES',2,1);**
* **INSERT into SUBJECT VALUES(016,'DM',2,2);**
  1. **IANMARKS**

**CREATE table IAMARKS(**

**USN INT,SUBCODE INT,SSID INT,**

**test1 INT, test2 INT, test3 INT,**

**FinalIA INT**

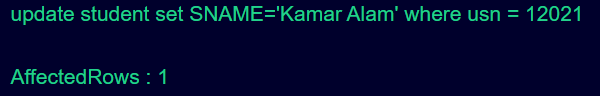
**);**

* **INSERT into IAMARKS VALUES(12021,012,1,87,92,90,288);**
* **INSERT into IAMARKS VALUES(22021,013,1,86,92,90,287);**
* **INSERT into IAMARKS VALUES(32021,014,1,84,96,90,278);**
* **INSERT into IAMARKS VALUES(42021,015,1,85,92,97,284);**
* **INSERT into IAMARKS VALUES(52021,016,1,84,91,94,281)**

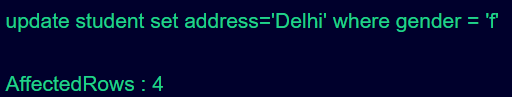
1. **UPDATE**

Modify data in single and multiple column in a table;

Single :

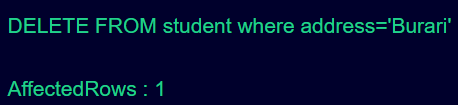
* update student set SNAME='Kamar Alam' where usn = 12021
* 

Multiple:

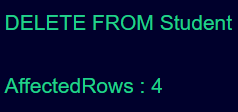
* update student set address='Delhi' where gender = 'f'
* 

1. **Delete**

Selective:

* DELETE FROM student where address='Burari';
* 

All record:

* DELETE FROM Student; .

**Experiment 5**

**AIM:-** **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.

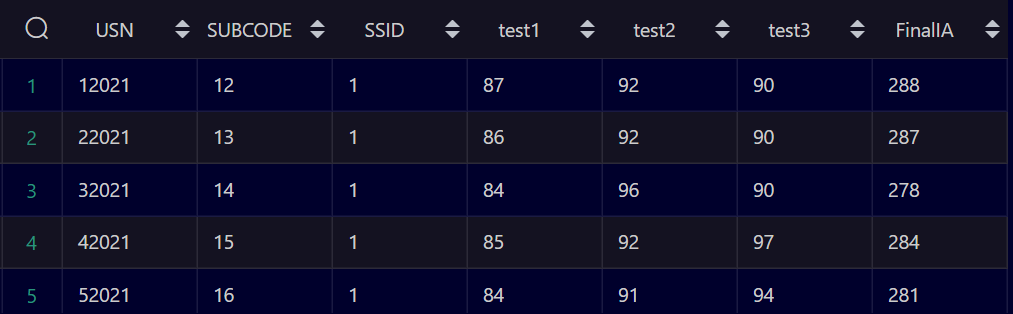
Sort the data in ascending and descending order in a table on the basis of one

column or more than one column.

Procedure:-

1. Retrieve the entire content of the table

**SELECT \* From iamarks;**

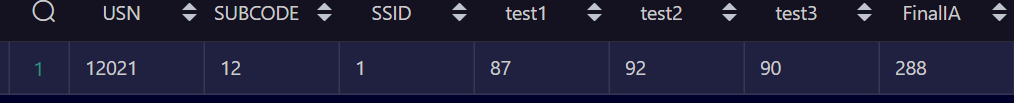
****

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

Retrieve contents from a table based on various operator i.e string operators,

* 1. **Logical operator**

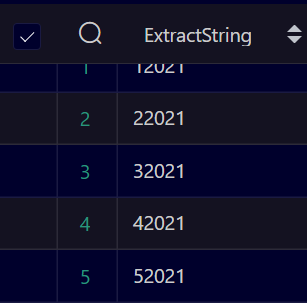
**SELECT \* from iamarks where test1=87 OR test2=90;**

****

* 1. String

**SELECT SUBSTRING(USN, 1, 5) AS ExtractString**

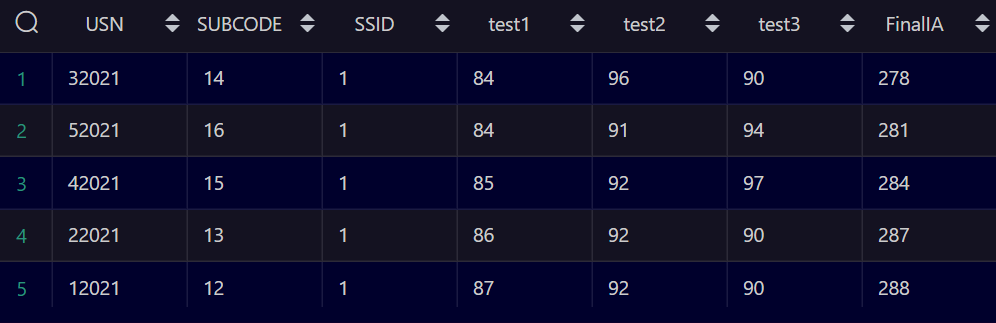
**FROM iamarks;**



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

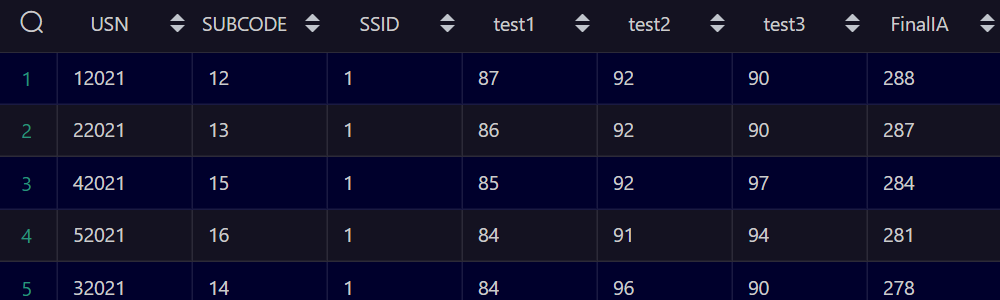
Ascending

**SELECT \* from iamarks order by finalia ;**

****

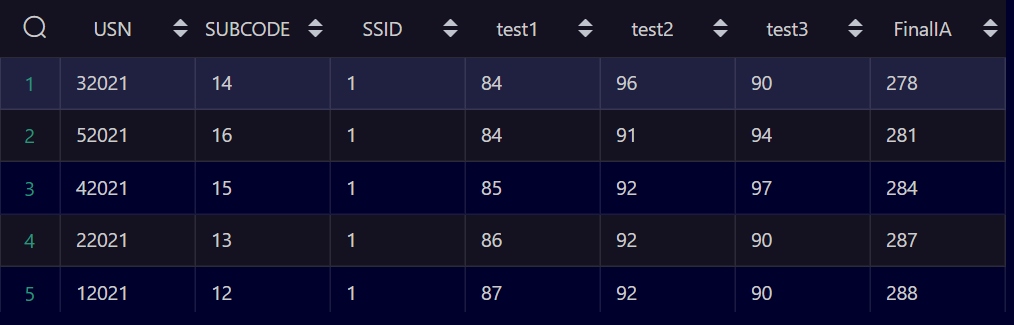
Descending

**Select \* From iamarks order by finalia DESC;**

****

**BY Using Multiple column**

**SELECT \* from iamarks order by finalia ,subcode ;**

****

**Experiment 6**

**Aim:-Create a table using following integrity constraints:**

**Primary Key, Unique Key, Not NULL ,Check ,Default.**

**Procedure:-**

**1.Primary key Constraints.**

* **Create a Table and Set (ID) as a Primary Key;**

CREATE TABLE ConstraintDemo1

(

ID INT PRIMARY KEY,

Name VARCHAR(50) NULL

);

****

* **Create a Table Using Unique key**

CREATE TABLE Persons (

ID int NOT NULL,

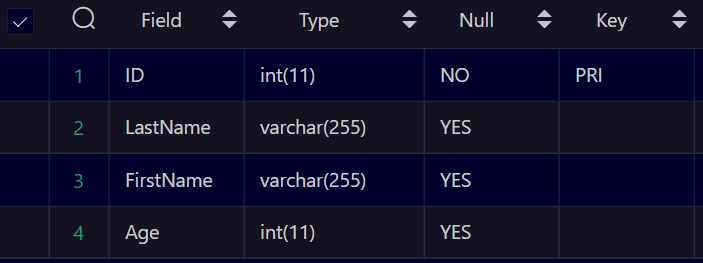
LastName varchar(255),

FirstName varchar(255),

Age int,

UNIQUE (ID)

);

****

* **Create a Table using Not NULL Constraints:**

CREATE TABLE ConstraintDemo1

(

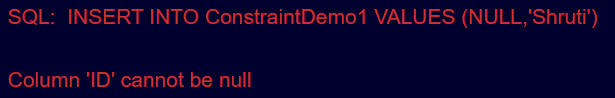
ID INT NOT NULL,

Name VARCHAR(50) NULL

);

**Now insert NULL value to confirm whether it’s working or not.**

INSERT INTO ConstraintDemo1 VALUES (NULL,'Shruti');



* **Create a table Using Check Constraints:**

CREATE TABLE check\_constrain

(

ID INT PRIMARY KEY,

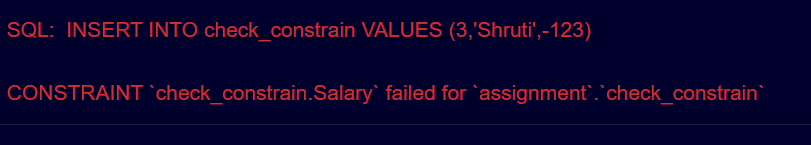
Name VARCHAR(50) NULL,

Salary INT CHECK (Salary>0)

);

* **Put a negative Value in Salary column to verify Check Condition**

**INSERT INTO check\_constrain VALUES (3,'Shruti',-123);**



* **Create a table using Default Constraints:**

CREATE TABLE default\_constrain

(

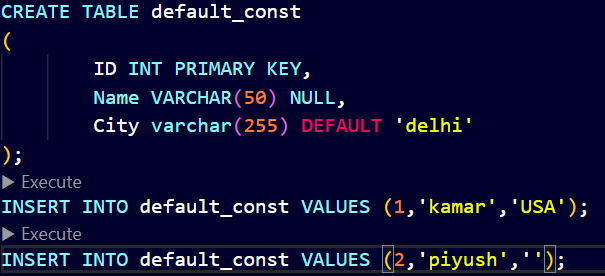
ID INT PRIMARY KEY,

Name VARCHAR(50) NULL,

EmployeeDate DATETIME NOT NULL DEFAULT CURRENT\_TIMESTAMP

)

* Now insert data and leave employee date as empty.



* **Default Value get Inserted Automatically i.e City (Delhi)**

****

* **Alter table add Foreign Key**
* ALTER TABLE iamarks add FOREIGN KEY (USN) REFERENCES class(USN);

**Experiment 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**

**Procedure:-**

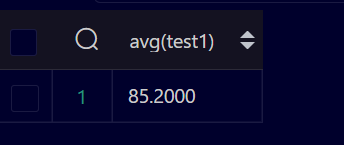
**SUM:**

* Select sum(test1) from iamarks;



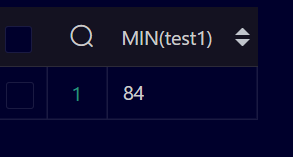
**AVERAGE:**

* Select avg(test1) from iamarks;

****

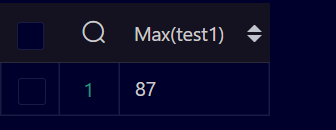
**MIN:**

* Select MIN(test1) from iamarks;



**MAX:**

* Select Max(test1) from iamarks;

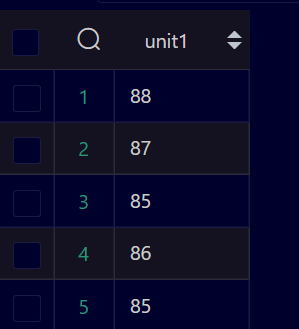


**Experiment 8**

**AIM: Retrieve data from a table using alias names .**

**Procedure:-**

SELECT test1 +1 as 'unit1' FROM iamarks

****

**Experiment 9**

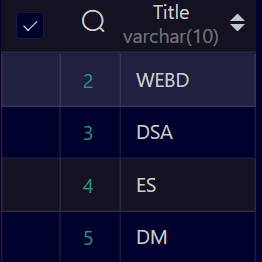
**Aim: Retrieve data of a table using nested queries.**

**Procedure:-**

SELECT (Title) FROM subject where SUBCODE>14 IN (

    SELECT test1 from iamarks WHERE usn>3000

        );

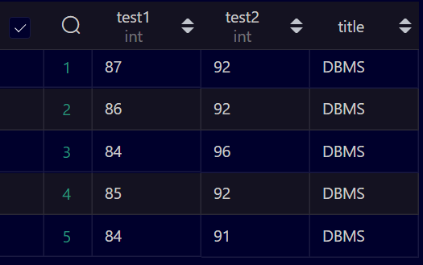


**Experiment 10**

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

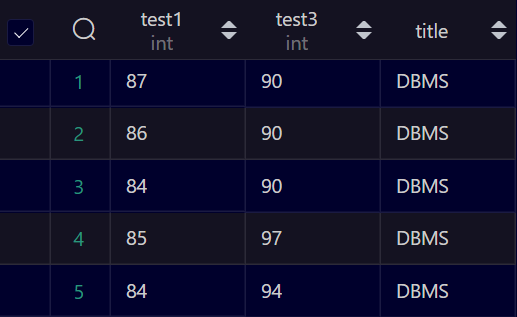
**Inner join:**

SELECT iamarks.test1,iamarks.test3,subject.title FROM iamarks inner JOIN subject ON iamarks.SUBCODE;



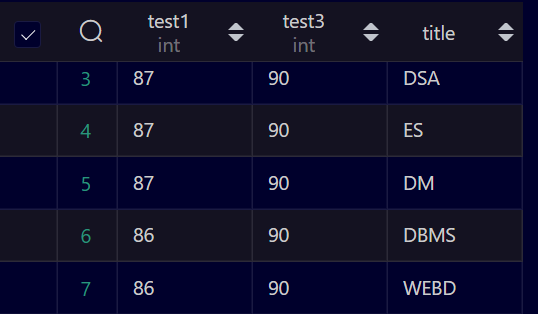
**Left outer join:**

SELECT iamarks.test1,iamarks.test3,subject.title FROM iamarks LEFT JOIN subject ON iamarks.SUBCODE;

****

**Right Outer joint:**

SELECT iamarks.test1,iamarks.test3,subject.title FROM iamarks RIGHT JOIN subject ON iamarks.SUBCODE;

****

**Full Outer joint:**

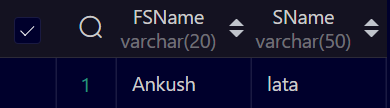
**Experiment-11**

AIM: Create view from one table and more than one table.

Procedure:

create view present as select FSName, SName from practice, demo where practice.SNo= demo.SNo;

create view past as select FSName from practice where SNo>=0;



**Experiment 12**

**AIM:-** Create a index on a coloumn of a table.

**Experiment 13**

**AIM:** 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.

create table person(driver\_id varchar(10),name varchar(10),address varchar(10),primary key(driver\_id));

 create table car(regno varchar(10),model varchar(10),year int,primary key(regno));

create table accident(report\_number int,accd\_date date,location varchar(10),primary key(report\_number));

 create table owns(driver\_id varchar(10),regno varchar(10),primary key(driver\_id,regno),foreign key(driver\_id) references person(driver\_id),foreign key(regno) references car(regno));

 create table participated(driver\_id varchar(10),regno varchar(10),report\_number int,damage\_amount int,primary key(driver\_id,regno,report\_number),foreign key(driver\_id) references person(driver\_id),foreign key(regno) references car(regno),foreign key(report\_number) references accident(report\_number));

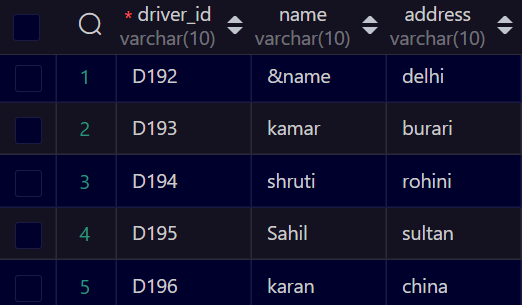
insert into person values('D192','&name','delhi');

insert into person values('D193','kamar','burari');

insert into person values('D194','shruti','rohini');

insert into person values('D195','Sahil','sultan');

insert into person values('D196','karan','china');

****

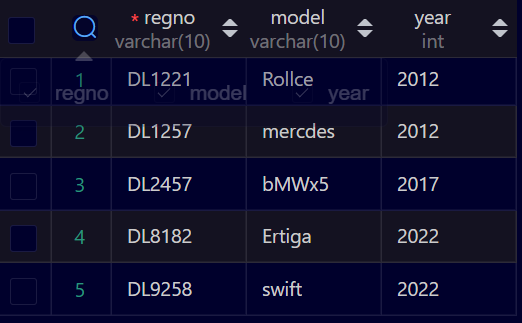
insert into  car values('DL8182','Ertiga',2022);

insert into  car values('DL9258','swift',2022);

insert into  car values('DL2457','bMWx5',2017);

insert into  car values('DL1257','mercdes',2012);

insert into  car values('DL1221','Rollce',2012);

****

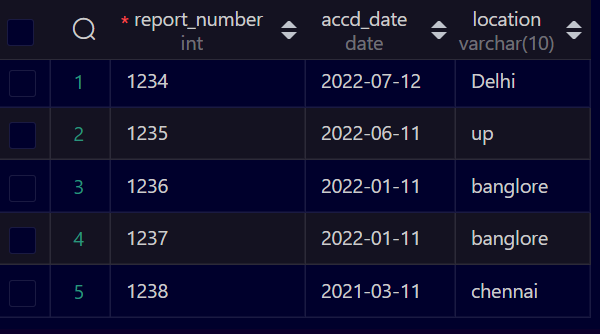
insert into accident values(1234,'2022-07-12','Delhi');

insert into accident values(1235,'2022-06-11','up');

insert into accident values(1236,'2022-01-11','banglore');

insert into accident values(1237,'2022-01-11','banglore');

insert into accident values(1238,'2021-03-11','chennai');

****

insert into owns values('D190','dl8213');

insert into owns values('D193','dl9113');

insert into owns values('D194','dl9116');

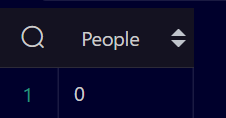
insert into owns values('D195','dl9616');

insert into owns values('D196','dl9617');

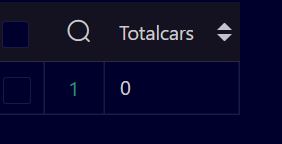


select count(distinct o.driver\_id) as People from owns o,participated p,accident a where a.accd\_date like

 '2008' and o.regno=p.regno and p.report\_number=a.report\_number;

****

select count(0) as Totalcars from car c,participated p where c.regno=p.regno and c.model='swift';



**Experiment-14**

**AIM**: 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

**Procedure:**

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

create table student(stud\_no int(5) primary key,stud\_name varchar(15));

create table membership(mem\_no int(5) primary key,stud\_no int(5) references student(stud\_no));

create table book\_(book\_no int(5) primary key,book\_name varchar(20),author varchar(2));

create table lss\_rec\_(iss\_no int primary key,iss\_date date,mem\_no int(5) references membership(mem\_no),book\_no int(5) references book(book\_no));

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

insert into student values (1001,"Kamar");

insert into student values (1002,"shruti");

insert into student values (1003,"muskan");

insert into student values (1004,"sahil");

insert into student values (1005,"Aditya");

insert into student values (1006,"isha");

insert into student values (1007,"gagan");

insert into student values (1008,"Nishtha");

insert into student values (1009,"niwanshu");

insert into student values (1010,"vanshu");

insert into student values (1011,"shubham");

* **Insert into membership table**

insert into membership values(101,1001);

insert into membership values(102,1002);

insert into membership values(103,1003);

insert into membership values(104,1004);

insert into membership values(105,1005);

insert into membership values(106,1006);

insert into membership values(107,1007);

insert into membership values(108,1008);

insert into membership values(109,1009);

insert into membership values(110,1010);

insert into membership values(111,1011);

* **Insert into book\_ table**

insert into book\_ values (1,"Harry Potter","JK");

insert into book\_ values (2,"Story of my life","Helen Keller");

insert into book\_ values (3,"Tell me a story","Ravinder Singh");

insert into book\_ values (4,"Can love happen","Ravinder Singh");

insert into book\_ values (5,"Your dreams are mine now","Ravinder Singh");

insert into book\_ values (6,"the perfect us","Durjoy Dutta");

insert into book\_ values (7,"Someone like u","Durjoy Dutta");

insert into book\_ values (8,"The open door","Helen Keller");

insert into book\_ values (9,"Five points someone","Chetan Bhagat");

insert into book\_ values (10,"One indian girl","Chetan Bhagat");

insert into book\_ values (11,"DBMS","Elmarsi & Navathe");

* **Insert into Iss\_rec\_**

insert into lss\_rec\_ values(01,"1998-05-08",104,10);

insert into lss\_rec\_ values(02,"1998-03-08",102,9);

insert into lss\_rec\_ values(03,"1998-04-08",103,8);

insert into lss\_rec\_ values(04,"1998-06-08",101,7);

insert into lss\_rec\_ values(05,"1998-07-08",105,6);

insert into lss\_rec\_ values(06,"1998-08-08",106,5);

insert into lss\_rec\_ values(07,"1997-01-09",107,4);

insert into lss\_rec\_ values(08,"1996-08-10",108,3);

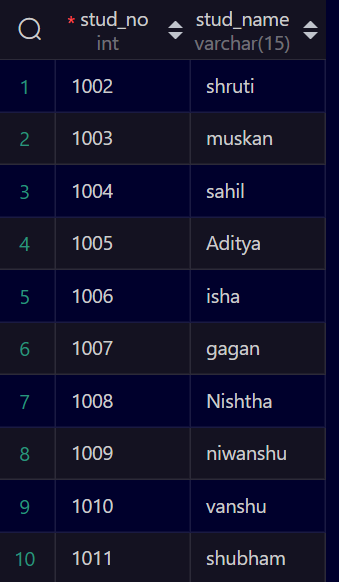
insert into lss\_rec\_ values(09,"1995-05-11",109,2);

insert into lss\_rec\_ values(010,"1995-05-12",110,1);

insert into lss\_rec\_ values(011,"1995-05-12",111,11);

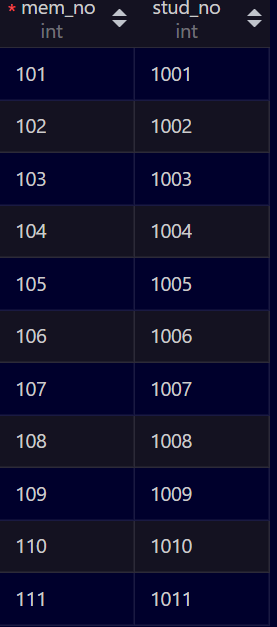
(iii)Display all records for all tables

* Retrieve records from student table.
* SELECT \* from student;

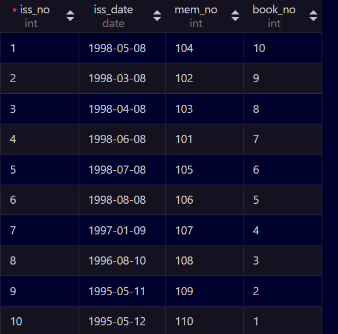


* Retrieve records from membership table.

SELECT \* from membership;



* Retrieve records from lss\_rec\_ table.
* SELECT \* from lss\_rec\_;

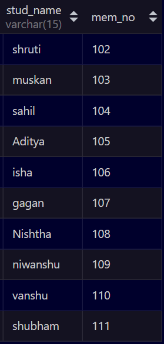


* Retrieve records from book\_ table.

SELECT \* from book\_;

****

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

select stud\_name, mem\_no from student s, membership m where m.stud\_no=s.stud\_****

(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 Nam

**Experiment-15**

**AIM**:- 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)

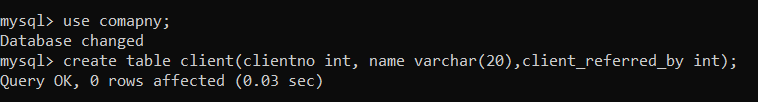
EMPLOYEE (empid#, emp\_type#, deptno, salary, firstname, lastname) Notes:

1. Column followed by # is the primary key of the table.
2. Each client may be referred by another client. If so, the client number of the referring client is stored in referred\_by.
3. The total cost for a particular order line = no\_of\_items \* item\_cost.c.

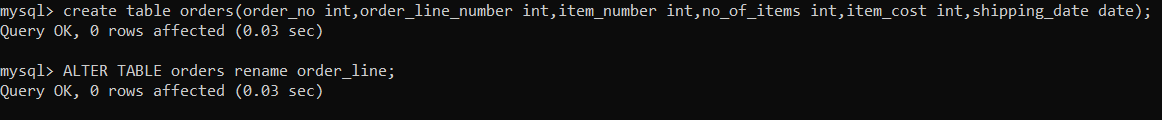
Write queries for the following

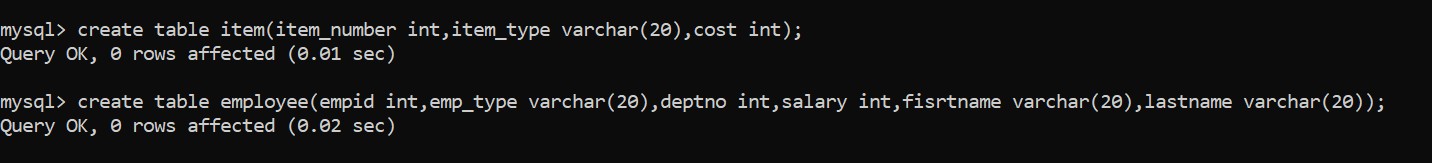
1. Create all the above tables.
2. Insert at least five records.
3. Display all the rows and columns in the CLIENT table. Sort by client name in reverse alphabetical order.
4. Display the item number and total cost for each order line (total cost = no of items X item cost). Name the calculated column TOTAL COST.
5. Display all the client numbers in the ORDER table. Remove duplicates.
6. Display the order number and client number from the ORDER table. Output the result in the format. Client <clientno> ordered <orderno>
7. 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.
8. Display the client name and order date for all orders.
9. Repeat query (6) but also display all clients who have never ordered anything.
10. Display the client name and order date for all orders using the join keywords.
11. Display the client name and order date for all orders using the JOIN method.
12. 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.

**(i)**

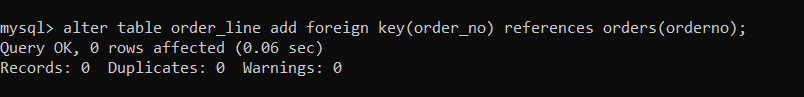
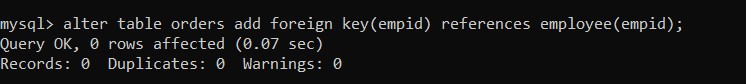
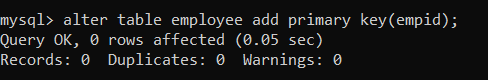
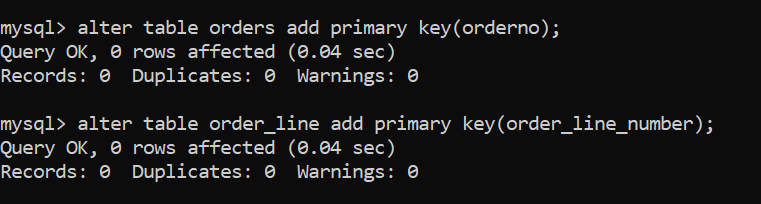






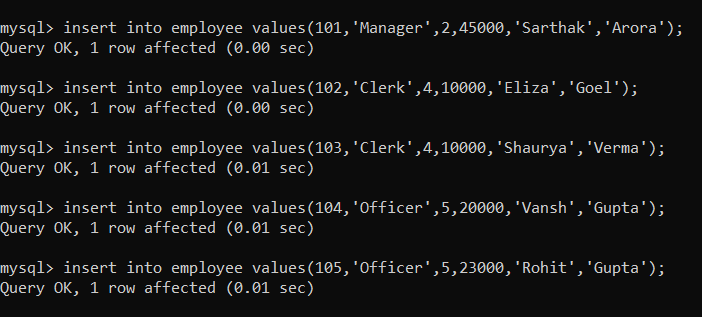
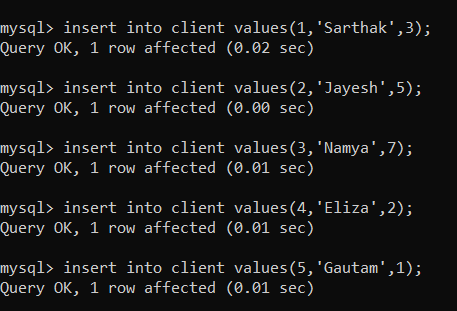


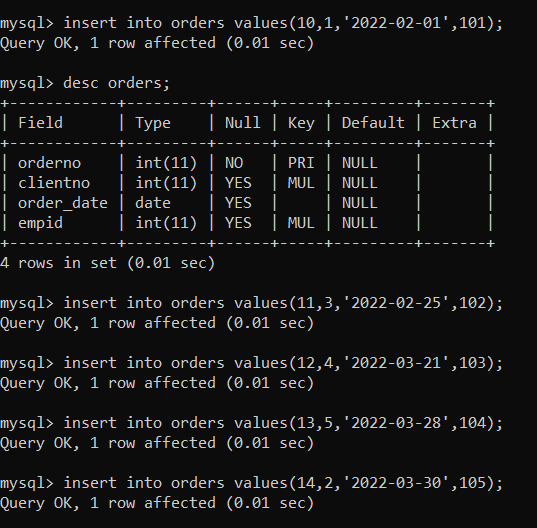
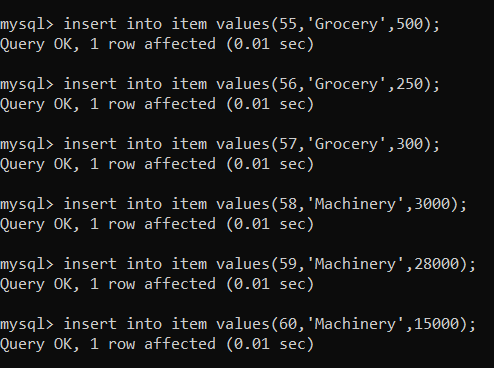
**Adding Keys**



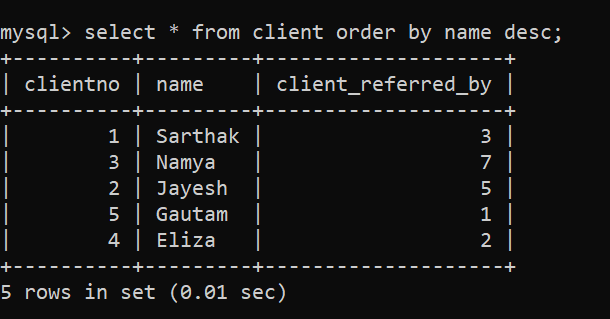
**(ii)**



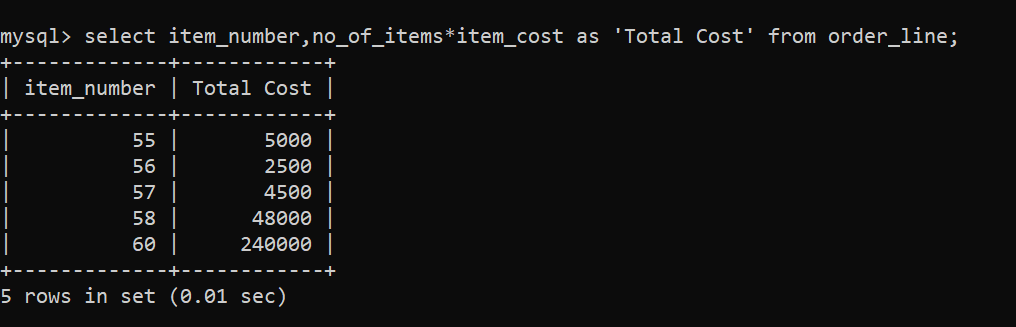


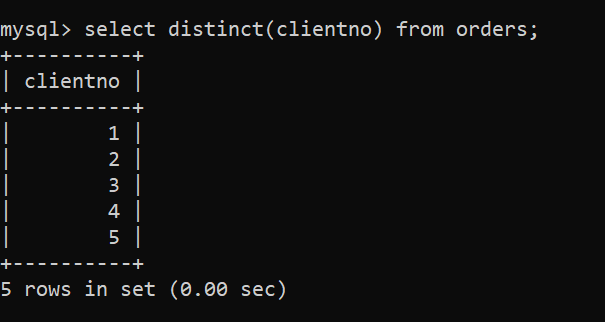
**(iii)**



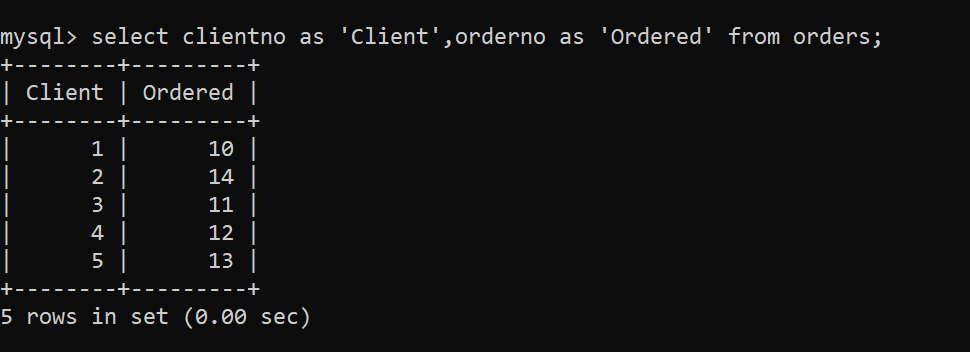
**(iv)**



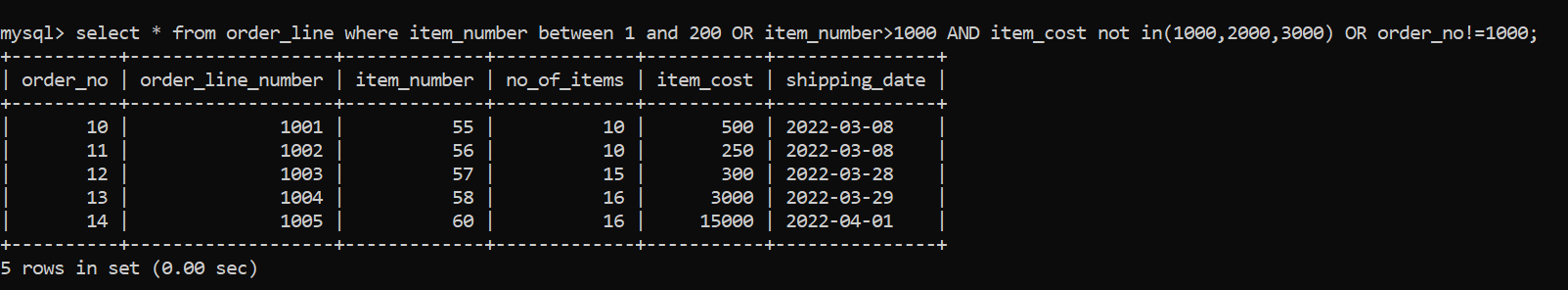
**(v)**



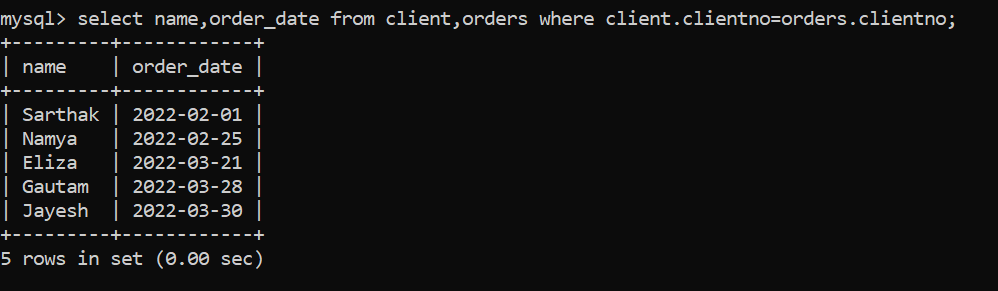
**(vi)**



**(vii)**



**(viii)**



**(ix)**