BMS Institute of Technology

Yelahanka, Bangalore-560 064

Data Base Management System (15CSL58)

Laboratory Manual

For

5th Semester B.E

Information Science & Engineering

Department of Information Science & Engineering

BMS Institute of Technology

Yelahanka, Bangalore-560 064

BMS Institute of Technology

Yelahanka, Bangalore-560 064

Data Base Management System(15CSL58)

Laboratory Manual

For
5th Semester B.E
Information Science & Engineering

Prepared by **Reviewed by**
HOD/ISE

Department of Information Science & Engineering
BMS Institute of Technology
Yelahanka, Bangalore-560 064

CONTENTS

	average. **2. Find the name and numbers of all salesman who had more than one customer.** **3. List all the salesman and indicate those who have and don't have customers in** **their cities (Use UNION operation.)** **4. Create a view that finds the salesman who has the customer with the highest** **order of a day.** **5. Demonstrate the DELETE operation by removing salesman with id 1000. All** **his orders must also be deleted.**	
3.	**Program 3:** **Consider the schema for Movie Database:** **ACTOR(Act_id, Act_Name, Act_Gender)** **DIRECTOR(Dir_id, Dir_Name, Dir_Phone)** **MOVIES(Mov_id, Mov_Title, Mov_Year, Mov_Lang, Dir_id)** **MOVIE_CAST(Act_id, Mov_id, Role)** **RATING(Mov_id, Rev_Stars)** **Write SQL queries to** **1. List the titles of all movies directed by 'Hitchcock'.** **2. Find the movie names where one or more actors acted in two or more movies.** **3. List all actors who acted in a movie before 2000 and also in a movie after** **2015 (use JOIN operation).** **4. Find the title of movies and number of stars for each movie that has at least** **One rating and find the highest number of stars that movie received. Sort the** **result by movie title.** **5. Update rating of all movies directed by 'Steven Spielberg' to 5.**	**7-8**
4.	**Program 4:** **Consider the schema for College Database:** **STUDENT(USN, SName, Address, Phone, Gender)** **SEMSEC(SSID, Sem, Sec)** **CLASS(USN, SSID)**	**9-10**

	SUBJECT(Subcode, Title, Sem, Credits) **IAMARKS(USN, Subcode, SSID, Test1, Test2, Test3, FinalIA)** **Write SQL queries to** **1. List all the student details studying in fourth semester 'C' section.** **2. Compute the total number of male and female students in each semester and** **In each section.** **3. Create a view of Test1 marks of student USN '1BI15CS101' in all subjects.** **4. Calculate the FinalIA (average of best two test marks) and update the** **corresponding table for all students.** **5. Categorize students based on the following criterion:** **If FinalIA = 17 to 20 then CAT = 'Outstanding'** **If FinalIA = 12 to 16 then CAT = 'Average'** **If FinalIA< 12 then CAT = 'Weak'** **Give these details only for 8th semester A, B, and C section students.**	
5.	**Program 5:** **Consider the schema for Company Database:** **EMPLOYEE(SSN, Name, Address, Sex, Salary, SuperSSN, DNo)** **DEPARTMENT(DNo, DName, MgrSSN, MgrStartDate)** **DLOCATION(DNo,DLoc)** **PROJECT(PNo, PName, PLocation, DNo)** **WORKS_ON(SSN, PNo, Hours)** **Write SQL queries to** **1. Make a list of all project numbers for projects that involve an employee** **whose last name is 'Scott', either as a worker or as a manager of the** **department that controls the project.** **2. Show the resulting salaries if every employee working on the 'IoT' project** **Is given a 10 percent raise.** **3. Find the sum of the salaries of all employees of the 'Accounts' Department** **As well as the maximum salary, the minimum salary, and**	**11-12**

	the average **salary in this department** **4. Retrieve the name of each employee who works on all the projects** **controlledby department number 5 (use NOT EXISTS operator).** **5. For each department that has more than five employees, retrieve the** **department number and the number of its employees who are making more** **than Rs. 6,00,000.** **b) Write a Perl program to accept UNIX command from a HTML form and to display the output of the command executed.**	

INTRODUCTION

What is a Database?

To find out what database is, we have to start from data, which is the basic building block of anyDBMS.

Data: Facts, figures, statistics etc. having no particular meaning (e.g. 1, ABC, 19 etc).

Record: Collection of related data items, e.g. in the above example the three data items had no meaning.
But if we organize them in the following way, then they collectively represent meaningful information.

Roll	Name	Age
1	ABC	19

Table or Relation: Collection of related records.

Roll	Name	Age
1	ABC	19
2	DEF	22
3	XYZ	28

The columns of this relation are called Fields, Attributes or Domains. The rows are called Tuples or Records.

Database: Collection of related relations. Consider the following collection of tables:

T1	Roll	Name	Age
	1	ABC	19
	2	DEF	22
	3	XYZ	28

T2	Roll	Address
	1	KOL
	2	DEL
	3	MUM

DATABASE APPLICATION LABORATORY
GSSSIETW 2 DEPT OF ISE

T3	Roll Year
1	I
2	II
3	I

T4

Year	Hostel
I	H1
II	H2

We now have a collection of 4 tables. They can be called a "related collection" because we can clearly Find out that there are some common attributes existing in a selected pair of tables. Because of these Common attributes we may combine the data of two or more tables together to find out the complete Details of a student. Questions like "Which hostel does the youngest student live in?" can be answered Now, although *Age* and *Hostel* attributes are in different tables.

In a database, data is organized strictly in row and column format. The rows are called Tuple or Record. The data items within one row may belong to different data types. On the other hand, the columns are Often called Domain or Attribute. All the data items within a single attribute are of the same data type.

What is Management System?

\

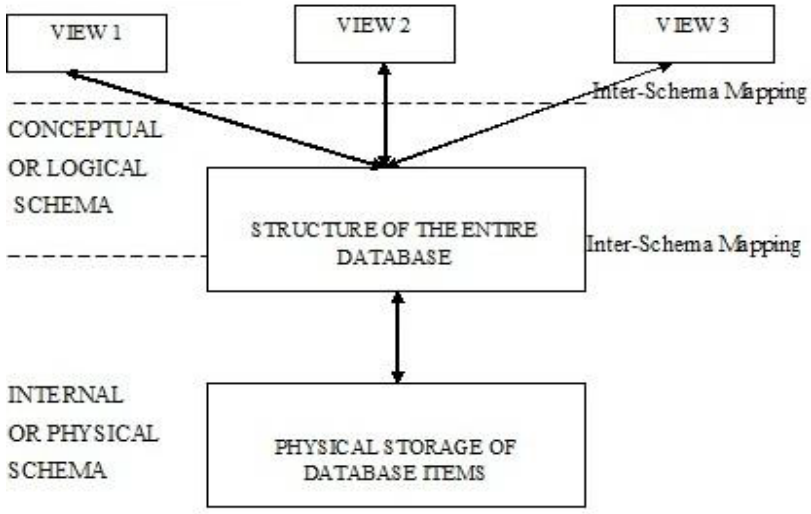
A management system is a set of rules and procedures which help us to create organize and manipulate the database. It also helps us to add, modify delete data items in the database. The management system can be either manual or computerized.

The management system is important because without the existence of some kind of rules and regulationsit is not possible to maintain the database. We have to select the particular attributes which should be

included in a particular table; the common attributes to create relationship between two tables; if a new record has to be inserted or deleted then which tables should have to be handled etc. These issues must beresolved by having some kind of rules to follow in order to maintain the integrity of the database.

Three Views of Data

If database is viewed from different angles produces difference sights. Likewise, the database that we have created already can have different aspects to reveal if seen from different levels of abstraction. The Term Abstraction is very important here. Generally it means the amount of detail you want to hide. Any Entity can be seen from different perspectives and levels of complexity to make it a reveal its current Amount of abstraction. Let us illustrate by a simple example. A computer reveals the minimum of its internal details, when seen from outside. We do not know what parts it is built with. This is the highest level of abstraction, meaning very few details are visible. If we open the computer case and look inside at the hard disc, motherboard, CD drive, CPU and RAM, we are in middle level of abstraction. If we move on to open the hard disc and examine its tracks, sectors and read-write heads, we are at the lowest level of abstraction, where no details are invisible. In the same manner, the database can also be viewed from different levels of abstraction to reveal different levels of details. From a bottom-up manner, we may find that there are three levels of abstraction or views in the database.



The word schema means arrangement – how we want to arrange things that we have to store. The Diagram above shows the three different schemas used in DBMS, seen from different levels of Abstraction.

The lowest level, called the Internal or Physical schema, deals with the description of how raw data Items (like 1, ABC, KOL, H2 etc.) are stored in the physical storage (Hard Disc, CD, Tape Drive etc.). It Also describes the data type of these data items, the size of the items in the storage media, the location (Physical address) of the items in the storage device and so on. This schema is useful for database Application developers and database administrator.

The middle level is known as the Conceptual or Logical Schema, and deals with the structure of the Entire database. Please note that at this level we are not interested with the raw data items anymore, we

Are interested with the structure of the database. This means we want to know the information about the attributes of each table, the common attributes in different tables that help them to be combined, what kind of data can be input into these attributes, and so on. Conceptual or Logical schema is very useful for database administrators whose responsibility is to maintain the entire database.

The highest level of abstraction is the External or View Schema. This is targeted for the end users. Now, An end user does not need to know everything about the structure of the entire database, rather than the Amount of details he/she needs to work with. We may not want the end user to become confused with Astounding amount of details by allowing him/her to have a look at the entire database, or we may also Not allow this for the purpose of security, where sensitive information must remain hidden from unwanted persons. The database administrator may want to create custom made tables, keeping in mind the specific kind of need for each user. These tables are also known as virtual tables, because they have no separate physical existence. They are crated dynamically for the users at runtime. Say for example, in our sample database we have created earlier, we have a special officer whose responsibility is to keep in touch with the parents of any under aged student living in the hostels. That officer does not need to know every detail except the ***Roll, Name, Address*** and ***Age*****.** The database administrator may create a virtual table with only these four attributes, only for the use of this officer.

Data Independence

This brings us to our next topic: data independence. It is the property of the database which tries to ensure that if we make any change in any level of schema of the database, the schema immediately above it would require minimal or no need of change.

What does this mean? We know that in a building, each floor stands on the floor below it. If we change the design of any one floor, e.g. extending the width of a room by demolishing the western wall of that room, it is likely that the design in the above floors will have to be changed also. As a result, one change needed in one particular floor would mean continuing to change the design of each floor until we reach the top floor, with an increase in the time, cost and labour. Would not life be easy if the change could be contained in one floor only? Data independence is the answer for this. It removes the need for additional amount of work needed in adopting the single change into all the levels above.

Data independence can be classified into the following two types:

1. Physical Data Independence: This means that for any change made in the physical schema, the need to change the logical schema is minimal. This is practically easier to achieve.

Let us explain with an example.

Say, you have bought an Audio CD of a recently released film and one of your friends has bought an Audio Cassette of the same film. If we consider the physical schema, they are entirely different. The first is digital recording on an optical media, where random access is possible. The second one is magnetic recording on a magnetic media, strictly sequential access. However, how this change is reflected in the logical schema is very interesting. For music tracks, the logical schema for both the CD and the Cassette is the title card imprinted on their back. We have information like Track no, Name of the Song, Name of

the Artist and Duration of the Track, things which are identical for both the CD and the Cassette. We can clearly say that we have achieved the physical data independence here.

2. Logical Data Independence: This means that for any change made in the logical schema, the need to change the external schema is minimal. As we shall see, this is a little difficult to achieve. Let us explain with an example.

3. Suppose the CD you have bought contains 6 songs, and some of your friends are interested in copying some of those songs (which they like in the film) into their favourite collection. One friend wants the songs 1, 2, 4, 5, 6, another wants 1, 3, 4, 5 and another wants 1, 2, 3, 6. Each of these collections can be compared to a view schema for that friend. Now by some mistake, a scratch has appeared in the CD and you cannot extract the song 3. Obviously, you will have to ask the friends who have song 3 in their proposed collection to alter their view by deleting song 3 from their proposed collection as well.

Database Administrator

The Database Administrator, better known as DBA, is the person (or a group of persons) responsible for the well being of the database management system. S/he has the flowing functions and responsibilities regarding database management:

1. Definition of the schema, the architecture of the three levels of the data abstraction, data independence.

2. Modification of the defined schema as and when required.

3. Definition of the storage structure i.e. and access method of the data stored i.e. sequential, indexed or direct.

4. Creating new used-id, password etc, and also creating the access permissions that each user can or cannot enjoy. DBA is responsible to create user roles, which are collection of the permissions (like read, write etc.) granted and restricted for a class of users. S/he can also grant additional permissions to and/or revoke existing permissions from a user if need be.

5. Defining the integrity constraints for the database to ensure that the data entered conform to some rules, Thereby increasing the reliability of data.

6. Creating a security mechanism to prevent unauthorized access, accidental or intentional handling of data that can cause security threat.

7. Creating backup and recovery policy. This is essential because in case of a failure the database must be able to revive itself to its complete functionality with no loss of data, as if the failure has never occurred. It is essential to keep regular backup of the data so that if the system fails then all data up to the point of failure will be available from a stable storage. Only those amount of data gathered during the failure would have to be fed to the database to recover it to a healthy status.

Advantages and Disadvantages of Database Management System

We must evaluate whether there is any gain in using a DBMS over a situation where we do not use it. Let us summarize the advantages.

1. Reduction of Redundancy: This is perhaps the most significant advantage of using DBMS. Redundancy is the problem of storing the same data item in more one place. Redundancy creates several problems like requiring extra storage space, entering same data more than once during data insertion, and deleting data from more than one place during deletion. Anomalies may occur in the database if insertion, deletion etc are not done properly

2. Sharing of Data: In a paper-based record keeping, data cannot be shared among many users. But in computerized DBMS, many users can share the same database if they are connected via a network.

3. Data Integrity: We can maintain data integrity by specifying integrity constrains, which are rules and restrictions about what kind of data may be entered or manipulated within the database. This increases the reliability of the database as it can be guaranteed that no wrong data can exist within the database at any point of time.

4. Data security: We can restrict certain people from accessing the database or allow them to see certain portion of the database while blocking sensitive information. This is not possible very easily in a paper based record keeping.

However, there could be a few disadvantages of using DBMS. They can be as following:

1. As DBMS needs computers, we have to invest a good amount in acquiring the hardware, software, installation facilities and training of users.

2. We have to keep regular backups because a failure can occur any time. Taking backup is a lengthy process and the computer system cannot perform any other job at this time.

3. While data security system is a boon for using DBMS, it must be very robust. If someone can bypass the security system then the database would become open to any kind of mishandling

LAB EXPERIMENTS

PART A: SQL PROGRAMMING

A. Consider the following schema for a Library Database:
BOOK (*Book_id, Title, Publisher_Name, Pub_Year*)
BOOK_AUTHORS (Book_id, Author_*Name*)
PUBLISHER (*Name, Address, Phone*)
BOOK_COPIES (*Book_id, Branch_id, No-of_Copies*)
BOOK_LENDING (*Book_id, Branch_id, Card_No, Date_Out, Due_Date*)
LIBRARY_BRANCH (*Branch_id, Branch_Name, Address*)
Write SQL queries to

1. Retrieve details of all books in the library – id, title, name of publisher, authors, number of copies in each branch, etc.

2. Get the particulars of borrowers who have borrowed more than 3 books, but from Jan 2017 to Jun 2017

3. Delete a book in BOOK table. Update the contents of other tables to reflect this data manipulation operation.

4. Partition the BOOK table based on year of publication. Demonstrate its working with a simple query.

5. Create a view of all books and its number of copies that are currently available in the Library.

Create database tables;

```
create table publisher(
        pname varchar(10) primary key,
        addr varchar(10),
        phno integer(10) );

create table book(
        bookid integer(4) primary key,
        title varchar(10),
        pyear integer(4),
        pname varchar(10) references publisher(pname) on delete cascade);

create table book_author(
        bookid integer(4) references book(bookid) on delete cascade,
        aname varchar(10),
        primary key(bookid,aname) );

create table library(branchid integer(5) primary key,
        bname varchar(10),
         addr varchar(10) );
```

```
create table book_copies(
        bookid integer(4) references book(bookid) on delete cascade,
        branchid integer(5) references library(branchid) on delete cascade,
        noc integer(3),
        primary key(bookid,branchid));

create table book_lending(
        bookid integer(4) references book(bookid) on delete cascade,
        branchid integer(5) references library(branchid) on delete cascade,
        cardno integer(6) not null,
        duedate date,
        date_out date,
        primary key(bookid,branchid,cardno));
```

Insertion of Values to Tables ;

```
insert into publisher values( 'bloomsbury','london',2076315600);
insert into publisher values( 'alen-unwin','uk',84250100);
insert into publisher values( 'marvel','NY',2125764000);
insert into publisher values( 'varun','Uganda',5656653662);

insert into book values(0007,'HP-TOOP',2007,'bloomsbury');
insert into book values(1004,'HOBBIT',1937,'alen-unwin');
insert into book values(0958,'FF:BOD',2007,'marvel');
insert into book values(3243,'13RW',2010,'varun');

insert into book_author values(0007,'JK.ROWLING');
insert into book_author values(1004,'JR.TOLKIEN');
insert into book_author values(0958,'M.FARMER');
insert into book_author values(3243,'RS.VARUN');

insert into library values(77777,'AVALAHALLI','BANGALORE');
insert into library values(24564,'GURGAUM','NEWDELHI');
insert into library values(53656,'KORMANGALA','BANGALORE');
insert into library values(53565,'AFJDAJ','BANGALORE');

insert into book_copies values(0007,77777,5);
insert into book_copies values(0007,24564,5);
insert into book_copies values(0958,24564,10);
insert into book_copies values(3243,53656,7);
insert into book_copies values(1004,53565,1);

insert into book_lending values(1004,53565,1,'10-oct-17','20-sep-17');
insert into book_lending values(0007,77777,43,'20-oct-17','21-sep-17');
insert into book_lending values(0007,24564,443,'11-jan-18','23-sep-17');
insert into book_lending values(3243,53656,45,'21-feb-18','24-sep-17');
insert into book_lending values(0958,24564,4565,'23-dec-17','25-sep-17');
insert into book_lending values(0958,24564,1,'29-dec-17','29-sep-17');
insert into book_lending values(0007,77777,1,'31-dec-17','30-sep-17');
```

Queries:

1. select book.bookid,title,pname,aname,noc,branchid from book,book_copies,book_author where book.bookid=book_author.bookid and book.bookid=book_copies.bookid order by branchid;

BOOKID	TITLE	PNAME	ANAME	NOC	BRANCHID
7	HP-TOOP	bloomsbury	JK.ROWLING	5	24564
958	FF:BOD	marvel	M.FARMER	10	24564
1004	HOBBIT	alen-unwin	JR.TOLKIEN	1	53565
3243	13RW	varun	RS.VARUN	7	53656
7	HP-TOOP	bloomsbury	J K.ROWLING	5	77777

```
mysql> insert into book_lending values(1004,53565,1,'10-oct-17','20-sep-17');
Query OK, 1 row affected, 2 warnings (0.04 sec)

mysql> insert into book_lending values(0007,77777,43,'20-oct-17','21-sep-17');
Query OK, 1 row affected, 2 warnings (0.03 sec)

mysql> insert into book_lending values(0007,24564,443,'11-jan-18','23-sep-17');
Query OK, 1 row affected, 2 warnings (0.03 sec)

mysql> insert into book_lending values(3243,53656,45,'21-feb-18','24-sep-17');
Query OK, 1 row affected, 2 warnings (0.03 sec)

mysql> insert into book_lending values(0958,24564,4565,'23-dec-17','25-sep-17');
Query OK, 1 row affected, 2 warnings (0.03 sec)

mysql> insert into book_lending values(0958,24564,1,'29-dec-17','29-sep-17');
Query OK, 1 row affected, 2 warnings (0.03 sec)

mysql> insert into book_lending values(0007,77777,1,'31-dec-17','30-sep-17');
Query OK, 1 row affected, 2 warnings (0.04 sec)

mysql> select book.bookid,title,pname,aname,noc,branchid from book,book_copies,book_author where book.bookid=book_author.bookid
 and book.bookid=book_copies.bookid order by branchid;
+--------+--------+------------+------------+------+-----------+
| bookid | title  | pname      | aname      | noc  | branchid  |
+--------+--------+------------+------------+------+-----------+
|      7 | dbms   | bloomsbury | jk.rowling |    5 |     24564 |
|    958 | ff:bod | marvel     | m.farmer   |   10 |     24564 |
|   1004 | hobbit | alen-unwin | jr.tolkien |    1 |     53565 |
|   3243 | 13rw   | varun      | rs.varun   |    7 |     53656 |
|      7 | dbms   | bloomsbury | jk.rowling |    5 |     77777 |
+--------+--------+------------+------------+------+-----------+
5 rows in set (0.00 sec)
```

2. select cardno,branchid,bookid from book_lending
where cardno = any (select cardno from book_lending where date_out between '19-sep-17' and '30-sep-17' having count(cardno)>2 group by(cardno));

CARDNO	BRANCHID	BOOKID
1	53565	1004
1	24564	958
1	77777	7

3.delete book where bookid=0007;

BOOKID	BRANCHID	CARDNO	DUEDATE	DATE_OUT
1004	53565	1	10-OCT-17	20-SEP-17
3243	53656	45	21-FEB-18	02-OCT-17
958	24564	4565	23-DEC-17	25-SEP-17
958	24564	1	29-DEC-17	29-SEP-17

```
mysql> select * from book order by(pyear);
+--------+--------+-------+------------+
| bookid | title  | pyear | pname      |
+--------+--------+-------+------------+
|   1004 | hobbit |  1937 | alen-unwin |
|      7 | dbms   |  2007 | bloomsbury |
|    958 | ff:bod |  2007 | marvel     |
|   3243 | 13rw   |  2010 | varun      |
+--------+--------+-------+------------+
4 rows in set (0.00 sec)

mysql> create view dbs as select bookid,title from book;
Query OK, 0 rows affected (0.03 sec)

mysql> select * from dbs;
+--------+--------+
| bookid | title  |
+--------+--------+
|      7 | dbms   |
|    958 | ff:bod |
|   1004 | hobbit |
|   3243 | 13rw   |
+--------+--------+
4 rows in set (0.00 sec)

mysql> delete book where bookid=0007;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for th
e right syntax to use near 'where bookid=0007' at line 1
mysql> delete FROM book where bookid=0007;
Query OK, 1 row affected (0.04 sec)

mysql> delete FROM book where bookid=0007;
Query OK, 0 rows affected (0.00 sec)

mysql> select * from book;
+--------+--------+-------+------------+
| bookid | title  | pyear | pname      |
+--------+--------+-------+------------+
|    958 | ff:bod |  2007 | marvel     |
|   1004 | hobbit |  1937 | alen-unwin |
|   3243 | 13rw   |  2010 | varun      |
+--------+--------+-------+------------+
3 rows in set (0.00 sec)

mysql>
```

4.select * from book order by(pyear);

BOOKID	TITLE	PYEAR	PNAME
1004	HOBBIT	1937	alen-unw
7	HP-TOOP	2007	bloomsbu
958	FF:BOD	2007	marvel
3243	13RW	2010	varun

5.create view dbs as select bookid,title from book;
select * from dbs;

BOOKID	TITLE
7	HP-TOOP
1004	HOBBIT
958	FF:BOD
3243	13RW

```
mysql> delete book where bookid=1004;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for th
e right syntax to use near 'where bookid=1004' at line 1
mysql> select * from book order by(pyear);
+--------+--------+-------+------------+
| bookid | title  | pyear | pname      |
+--------+--------+-------+------------+
|   1004 | hobbit |  1937 | alen-unwin |
|      7 | dbms   |  2007 | bloomsbury |
|    958 | ff:bod |  2007 | marvel     |
|   3243 | 13rw   |  2010 | varun      |
+--------+--------+-------+------------+
4 rows in set (0.00 sec)

mysql> create view dbs as select bookid,title from book;
Query OK, 0 rows affected (0.03 sec)

mysql> select * from dbs;
+--------+--------+
| bookid | title  |
+--------+--------+
|      7 | dbms   |
|    958 | ff:bod |
|   1004 | hobbit |
|   3243 | 13rw   |
+--------+--------+
4 rows in set (0.00 sec)

mysql>
```

B. Consider the following schema for Order Database:
SALESMAN (*Salesman_id, Name, City, Commission*)
CUSTOMER (*Customer_id, Cust_Name, City, Grade, Salesman_id*)
ORDERS (*Ord_No, Purchase_Amt, Ord_Date, Customer_id, Salesman_id*)
Write SQL queries to

1. Count the customers with grades above Bangalore's average.

2. Find the name and numbers of all salesmen who had more than one customer.

3. List all salesmen and indicate those who have and don't have customers in their cities (Use UNION operation.)

4. Create a view that finds the salesman who has the customer with the highest order of a day.

5. Demonstrate the DELETE operation by removing salesman with id 1000. All his orders must also be deleted.

Table Creation;

```
create table salesman(
        sid integer(4) primary key,
        sname varchar(10) not null,
        city varchar(10) not null,
        commission integer(10,2) not null);

create table customer(
        cid integer(5) primary key,
        cname varchar(10) not null,
        city varchar(10) not null,
        grade integer(3,1) not null check (grade<=10),
        sid integer(4) references salesman(sid) on delete cascade);

create table orders(
        ordno integer (3) primary key,
        pamt integer (10,2) not null,
        orddate date not null,
        sid integer(4) references salesman(sid) on delete cascade,
        cid integer(5) references customer(cid) on delete cascade);
```

Insertion of Values to Tables;

```
insert into salesman values(1000,'mike','bangalore',5000);
insert into salesman values(1001,'micheal','mangalore',500);
insert into salesman values(1002,'bevan','bangalore',1000);
insert into salesman values(1003,'clarke','pune',10000.50);
insert into salesman values(1004,'virat','mumbai',6500);

insert into customer values(25251,'varun','bellary',8.5,1003);
insert into customer values(25255,'pavan','shimoga',4.5,1000);
```

```
insert into customer values(25265,'sripad','bellary',7,1003);
insert into customer values(25278,'manish','jaipur',6,1004);
insert into customer values(25281,'kariyamma','chennai',2.4,1002);
insert into orders values(101,80000,'05-sep-17',1003,25265);
insert into orders values(102,45000,'05-oct-17',1000,25255);
insert into orders values(103,57000,'25-sep-17',1003,25251);
insert into orders values(104,10000,'29-nov-17',1004,25278);
insert into orders values(105,100000,'03-jan-17',1002,25281);


select sid from customer group by(sid) having count(sid)>=2;
update salesman set city='bangalore' where sname='clarke';
update customer set city='bangalore' where city='bellary';
update customer set grade=10 where cname='pavan';
update orders set pamt=1000000 where ordno=104;
```

Queries:

1.Count the customers with grades above Bangalore's average.

A; select count(cid) from customer where grade>=(select avg(grade) from customer where city='bangalore');

```
Count(cid)
---------
    2
```

2. Find the name and numbers of all salesmen who had more than one customer.

A; select sname,sid from salesman where sid=any(select sid from customer group by(sid) having count(sid)>=2);

```
 Sname     sid
---------  ---------
 Clarke    1003
```

```
+-------+
1 row in set (0.00 sec)

mysql> update salesman set city='bangalore' where sname='clarke';
Query OK, 1 row affected (0.03 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> update customer set city='bangalore' where city='bellary';
Query OK, 2 rows affected (0.04 sec)
Rows matched: 2  Changed: 2  Warnings: 0

mysql> update customer set grade=10 where cname='pavan';
Query OK, 1 row affected (0.03 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> update orders set pamt=1000000 where ordno=104;
Query OK, 1 row affected (0.04 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select count(cid) from customer where grade>=(select avg(grade) from customer where city='bangalore');
+------------+
| count(cid) |
+------------+
|          2 |
+------------+
1 row in set (0.00 sec)

mysql> select sname,sid from salesman where sid=any(
    -> select sid from customer group by(sid) having count(sid)>=2
    -> );
+--------+------+
| sname  | sid  |
+--------+------+
| clarke | 1003 |
+--------+------+
1 row in set (0.00 sec)

mysql> select sname from salesman,customer,'different' as city from salesman, customer where salesman.sid=customer.sid an
d salesman.city !=customer.city union
    -> select sname from salesman,customer,'same' as city from salesman,customer where salesman.sid=customer.sid and sale
sman.city =customer.city;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version
for the right syntax to use near ''different' as city from salesman, customer where salesman.sid=customer.sid and ' at li
ne 1
```

3. List all salesmen and indicate those who have and don't have customers in their cities (Use UNION operation.)

A; select s.sid,s.sname,s.city from salesman s,customer c where s.city=c.city union select s.sid, s.sname,'no match' from salesman s where not s.city=any(select city from customer);

Sid	Sname	city
1000	mike	bangalore
1002	bevan	bangalore
1003	clarke	bangalore
1001	micheal	no match
1004	virat	no match

```
mysql> create view abcde as select salesman.sid,sname from salesman,orders where orders.pamt=(select max(pamt) from order
s) and orders.sid=salesman.sid;
Query OK, 0 rows affected (0.05 sec)

mysql> select s.sid,s.sname,s.city from salesman s,customer c where s.city=c.city union select s.sid,s
    -> .sname,'no match' from salesman s where not s.city=any(select city from customer);
+------+---------+-----------+
| sid  | sname   | city      |
+------+---------+-----------+
| 1000 | mike    | bangalore |
| 1002 | bevan   | bangalore |
| 1003 | clarke  | bangalore |
| 1001 | micheal | no match  |
| 1004 | virat   | no match  |
+------+---------+-----------+
5 rows in set (0.00 sec)
```

4. Create a view that finds the salesman who has the customer with the highest order of a day.
A; Create view abcde as select salesman.sid,sname from salesman,orders where orders.pamt=(select max(pamt) from orders) and orders.sid=salesman.sid;

Ord_date	Sid	Sname
05-sep-17	1000	mike
05-oct-17	1001	micheal
25-sep-17	1001	micheal
29-nov-1	1002	bevan
03-jan-17	1001	micheal

5. Demonstrate the DELETE operation by removing salesman with id 1000. All his orders must also be deleted

A; delete from salesman where sid=1000;

1 row affected;

Sql> select * from salemman;

```
mysql> delete from  salesman where sid=1000;
Query OK, 1 row affected (0.03 sec)

mysql> select * from salesman;
+------+---------+-----------+------------+
| sid  | sname   | city      | commission |
+------+---------+-----------+------------+
| 1001 | micheal | mangalore |        500 |
| 1002 | bevan   | bangalore |       1000 |
| 1003 | clarke  | bangalore |      10001 |
| 1004 | virat   | mumbai    |       6500 |
+------+---------+-----------+------------+
4 rows in set (0.00 sec)
```

C. Consider the schema for Movie Database:
ACTOR (*Act_id, Act_Name, Act_Gender*)
DIRECTOR (*Dir_id, Dir_Name, Dir_Phone*)
MOVIES (*Mov_id, Mov_Title, Mov_Year, Mov_Lang, Dir_id*)
MOVIE_CAST (*Act_id, Mov_id, Role*)
RATING (*Mov_id, Rev_Stars*)
Write SQL queries to
1. List the titles of all movies directed by 'Hitchcock'.

2. Find the movie names where one or more actors acted in two or more movies.

3. List all actors who acted in a movie before 2000 and also in a movie after

2015 (use JOIN operation).
4. Find the title of movies and number of stars for each movie that has at least one rating and find the highest number of stars that movie received. Sort the result by movie title.

5. Update rating of all movies directed by 'Steven Spielberg' to 5.

Table Creation;

```
create table actor(
        aid integer(4) primary key,
        aname varchar(10),
        agender varchar(1));

create table director(
        did integer(5) primary key,
        dname varchar(10),
        phno varchar(10));

create table movies(
        mid integer(6) primary key,
        title varchar(10),
        moyear varchar(4),
        lang varchar(10),
        did integer(5) references director(did) on delete cascade);

create table moviecast(
        aid integer(4) references actor(aid) on delete cascade,
        mid integer(6) references movies(mid) on delete cascade,
        role varchar(10),
        primary key(aid,mid));

create table rating(
        mid integer (6) references movies(mid) on delete cascade,
        rev integer (2) check (rev<6),
        primary key(mid,rev));
```

Insertion of Values to Tables;

```
insert into actor values(1001,'preetham','M');
insert into actor values(1002,'jason','M');
insert into actor values(1003,'tom','M');
insert into actor values(1004,'jenna','f');
insert into actor values(1005,'anna','f');

insert into director values(10001,'hitchcock','9889898989');
insert into director values(10002,'steven','988989845');
insert into director values(10003,'varun','9889898798');
insert into director values(10004,'nolan','9889898111');
insert into director values(10005,'richbitch','9889898000');

insert into movies values(100001,'Psycho','1960','English',10001);
insert into movies values(100002,'Don','2011','Hindi',10003);
insert into movies values(100003,'super','2007','kannada',10003);
insert into movies values(100004,'vertigo','1958','English',10001);
insert into movies values(100005,'dunkirk','2016','English',10004);
insert into movies values(100006,'jaws','1975','English',10002);

insert into cast values(1001,100003,'Pscho');
insert into cast values(1004,100005,'Lead role');
insert into cast values(1003,100002,'villian');
insert into cast values(1002,100001,'Lead role');
insert into cast values(1005,100004,'side actor');
insert into cast values(1001,100006,'lead actor');

insert into rating values(100001,3);
insert into rating values(100002,4);
insert into rating values(100003,5);
insert into rating values(100004,1);
insert into rating values(100005,3);
insert into rating values(100006,2.5);
insert into rating values(100006,4);
```

Queries:

1. List the titles of all movies directed by 'Hitchcock'.

A; select * from movies,director where director.did=movies.did and dname='hitchcock';

MID	TITLE	MOYE	LANG	DID	DID	DNAME	PHNO
100001	Psycho	1960	English	10001	10001	hitchcock	9889898989
100004	vertigo	1958	English	10001	10001	hitchcock	9889898989

2. Find the movie names where one or more actors acted in two or more movies.

A; select title from movies where mid= any
(select mid from cast where aid=any
(select aid from cast having count(aid)>1 group by(aid)));

TITLE
-----
super
jaws

3. List all actors who acted in a movie before 2000 and also in a movie after 2015 (use JOIN operation).

A; select aname from actor,movies,cast where actor.aid=cast.aid and movies.mid=cast.mid and moyear <'2000' intersect
select aname from actor,movies,cast where actor.aid=cast.aid and movies.mid=cast.mid and moyear >'2015';

ANAME
----
preetham

4. Find the title of movies and number of stars for each movie that has at least one rating and find the highest number of stars that movie received. Sort the result by movie title.

A; select title,max(rev),count(rev)from movies, rating where movies.mid=rating.mid group by movies.title order by movies.title;

OR

A; select title,max(rev),count(rev)from movies, rating where movies.mid=rating.mid group by movies.title having max(rev)>0 order by movies.title

5. Update rating of all movies directed by 'Steven Spielberg' to 5

A; update rating set rev=5 where mid= any(select mid from movies,director where movies.did=director.did and dname='steven');

Sql> select * from rating;

```
mysql> insert into rating values(100002,4);
Query OK, 1 row affected (0.03 sec)

mysql> insert into rating values(100003,5);
Query OK, 1 row affected (0.03 sec)

mysql> insert into rating values(100004,1);
Query OK, 1 row affected (0.03 sec)

mysql> insert into rating values(100005,3);
Query OK, 1 row affected (0.03 sec)

mysql> insert into rating values(100006,2.5);
Query OK, 1 row affected (0.03 sec)

mysql> insert into rating values(100006,4);
Query OK, 1 row affected (0.03 sec)

mysql> desc actor;
+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| aid     | int(4)      | NO   | PRI | NULL    |       |
| aname   | varchar(10) | YES  |     | NULL    |       |
| agender | varchar(1)  | YES  |     | NULL    |       |
+---------+-------------+------+-----+---------+-------+
3 rows in set (0.00 sec)

mysql> select * from movies,director where director.did=movies.did and dname='hitchcock';
+--------+---------+--------+---------+-------+-------+-----------+------------+
| mid    | title   | moyear | lang    | did   | did   | dname     | phno       |
+--------+---------+--------+---------+-------+-------+-----------+------------+
| 100001 | Psycho  | 1960   | English | 10001 | 10001 | hitchcock | 9889898989 |
| 100004 | vertigo | 1958   | English | 10001 | 10001 | hitchcock | 9889898989 |
+--------+---------+--------+---------+-------+-------+-----------+------------+
2 rows in set (0.00 sec)
```

D. Consider the schema for College Database:
STUDENT (*USN, SName, Address, Phone, Gender*)
SEMSEC (*SSID, Sem, Sec*)
CLASS (*USN, SSID*)
SUBJECT (*Subcode, Title, Sem, Credits*)
IAMARKS (*USN, Subcode, SSID, Test1, Test2, Test3, FinalIA*)
Write SQL queries to

1. List all the student details studying in fourth semester 'C' section.

2. Compute the total number of male and female students in each semester and in each section.

3. Create a view of Test1 marks of student USN '1BI15CS101' in all subjects.

4. Calculate the FinalIA (average of best two test marks) and update the corresponding table for all students.

5. Categorize students based on the following criterion:

If FinalIA = 17 to 20 then CAT = 'Outstanding'
If FinalIA = 12 to 16 then CAT = 'Average'
If FinalIA< 12 then CAT = 'Weak'
Give these details only for 8th semester A, B, and C section students.
Solution:

Table Creation ;

```
create table student(usn varchar(10) primary key,
sname varchar(10) not null,
address varchar(10) not null,
phone integer(10),
gender varchar(6) not null);

create table semsec(ssid varchar(10) primary key,
sem integer(1) not null,
sec varchar(1) not null);

create table class(usn varchar(10),
ssid varchar(5),
primary key(usn,ssid),
foreign key (usn) references student (usn),
foreign key (ssid) references semsec (ssid));

create table subject(subcode varchar(6) primary key,
title varchar(10) not null,
sem integer(1) not null,
credits integer(2));

create table iamarks(usn varchar(10),
subcode varchar(6),
ssid varchar(5),
foreign key (usn) references student (usn),
```

```
foreign key (ssid) references semsec (ssid),
foreign key (subcode) references subject (subcode),
primary key(usn,subcode,ssid),
test1 integer(2),
test2 integer(2),
test3 integer(2),
finalia integer(2));
```

Insertion of values to tables;

```
insert into student values ('1by16cs417','vinod','abc street',7894563,'male');
insert into student values ('1by15cs02','def','abc street',7894563,'male');
insert into student values('1by15cs075','abcde','abcstreet',7894563,'female');

insert into semsec values('5a',5,'a');
insert into semsec values('5b',5,'b');
insert into semsec values('3a',3,'a');

insert into class values('1by16cs417','5a');
insert into class values('1by15cs02','5b');
insert into class values('1by15cs075','3a');

insert into subject values('15cs52','java',5,4);
insert into subject values('15cs51','cn',5,4);
insert into subject values('15cs53','dbms',5,4);

insert into iamarks values('1by16cs417','15cs52','5a',12,0,15,0);
insert into iamarks values('1by15cs02','15cs51','5b',13,14,0,0);
insert into iamarks values('1by15cs075','15cs53','3a',15,15,15,0);
```

Queries:

1. **List all the student details studying in fourth semester 'C' section.**
   A; select * from student s,semsec ss,class c where s.usn=c.usn and c.ssid=ss.ssid and sem=5 and sec='b';

USN	SNAME	ADDRESS	PHONE	GENDER	SSID	SEM	SEC	USN	SSID
1by15cs02	def	abc street	7894563	male	5b	5	b	1by15cs02	5b

2. Compute the total number of male and female students in each semester and in each section.

A; select count(s.gender) from student s,semsec ss,class c where s.usn=c.usn and c.ssid=ss.ssid group by sem,sec,gender;

COUNT(S.GENDER)

1
1
1

```
mysql>
mysql> insert into iamarks values('1by15cs075','15cs53','3a',15,15,15,0);
Query OK, 1 row affected (0.03 sec)

mysql> select * from student s,semsec ss,class c where s.usn=c.usn and
    -> c.ssid=ss.ssid and sem=5 and sec='b';
+-----------+-------+-----------+---------+--------+------+-----+-----+-----------+------+
| usn       | sname | address   | phone   | gender | ssid | sem | sec | usn       | ssid |
+-----------+-------+-----------+---------+--------+------+-----+-----+-----------+------+
| 1by15cs02 | def   | abc street | 7894563 | male   | 5b   |   5 | b   | 1by15cs02 | 5b   |
+-----------+-------+-----------+---------+--------+------+-----+-----+-----------+------+
1 row in set (0.00 sec)

mysql> Compute the total number of male and female students in each semester and in each section.
    -> select count(s.gender) from student s,semsec ss,class c where s.usn=c.usn and
    -> c.ssid=ss.ssid group by sem,sec,gender;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server versio
n for the right syntax to use near 'Compute the total number of male and female students in each semester and in eac' a
t line 1
mysql> Compute the total integer of male and female students in each semester and in each section. select count(s.gende
r) from student s,semsec ss,class c where s.usn=c.usn and c.ssid=ss.ssid group by sem,sec,gender;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server versio
n for the right syntax to use near 'Compute the total integer of male and female students in each semester and in ea' a
t line 1
mysql> select count(s.gender) from student s,semsec ss,class c where s.usn=c.usn and
    -> c.ssid=ss.ssid group by sem,sec,gender;
+----------------+
| count(s.gender) |
+----------------+
|              1 |
|              1 |
|              1 |
+----------------+
3 rows in set (0.00 sec)

mysql> create view test1 as
    -> select test1,subcode from iamarks where usn='1by15cs02';
Query OK, 0 rows affected (0.03 sec)
```

3. **Create a view of Test1 marks of student USN '1BI15CS101' in all subjects.**
   A; create view test1 as
   select test1,subcode from iamarks where usn='1by15cs02';
   View created.

   SQL> select * from test1;

TEST1	SUBCOD
13	15cs51

```
mysql> select count(s.gender) from student s,semsec ss,class c where s.usn=c.usn and
    -> c.ssid=ss.ssid group by sem,sec,gender;
+----------------+
| count(s.gender) |
+----------------+
|              1 |
|              1 |
|              1 |
+----------------+
3 rows in set (0.00 sec)

mysql> create view test1 as
    -> select test1,subcode from iamarks where usn='1by15cs02';
Query OK, 0 rows affected (0.03 sec)

mysql> select * from test1;
+-------+---------+
| test1 | subcode |
+-------+---------+
|    13 | 15cs51  |
+-------+---------+
```

4. **Calculate the FinalIA (average of best two test marks) and update the corresponding table for all students.**
   A; create or replace procedure avgmarks is
   cursor c_iamarks is
   select greatest(test1,test2) as a,greatest(test1,test3) as b,greatest(test2,test3) as c
   from iamarks
   where finalia=0 for update;

   c_a number;
   c_b number;
   c_c number;
   c_sum number;

```
c_avg number;

begin
open c_iamarks;
loop
fetch c_iamarks into c_a,c_b,c_c;
exit when c_iamarks%notfound;
dbms_output.put_line(c_a||''||c_b||''||c_c);
if c_a!=c_b then
c_sum:=c_a+c_b;
else
c_sum:=c_a+c_c;
end if;
c_avg:=c_sum/2;
dbms_output.put_line('sum is'||c_sum);
dbms_output.put_line('average is'||c_avg);
update iamarks set finalia=c_avg
where current of c_iamarks;
end loop;
close c_iamarks;
end;

/
To run the procedure
begin
  avgmarks();
 end;
   /
```

SQL> select * from iamarks;

USN	SUBCOD	SSID	TEST1	TEST2	TEST3	FINALIA
1by16cs417	15cs52	5a	12	0	15	0
1by15cs02	15cs51	5b	13	14	0	0
1by15cs075	15cs53	3a	15	15	15	0

```
n for the right syntax to use near 'c_avg:=c_sum/2' at line 1
    '> create  or replace procedure avgmarks is
        cursor c_iamarks is
        select greatest(test1,test2)as a, greatest(test1,test3)as b,greatest(test2,test3)as c from iamarks
        where finalia=0 for update;
        c_a number;
        c_b number;
        c_c number;
        c_sum number;
        c_avg number;
        begin
        open c_iamarks;
        loop
        fetch c_iamarks into c_a,c_b,c_c;
        exit when c_iamarks%notfound;
        dbms_output.put_line(c_a||''||c_b||''||c_c);
        if c_a!=c_b then
        c_sum:=c_a+c_b;
        else
        c_sum:=c_a+c_c;
        end if;
        c_avg:=c_sum/2;
        dbms_output.put_line('sum is||c_sum);
        dbms_output.put_line('average is'||c_avg);
        update iamarks set finalia=c_avg
        where current of c_iamarks;
        end loop;
        close c_iamarks;
        end;
        select * from iamarks;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server versio
n for the right syntax to use near 'dbms_output.put_line('sum is||c_sum);                                        ' a
t line 1
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server versio
n for the right syntax to use near 'dbms_output.put_line('average is'||c_avg)' at line 1
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server versio
n for the right syntax to use near 'of c_iamarks' at line 1
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server versio
n for the right syntax to use near 'end loop' at line 1
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server versio
n for the right syntax to use near 'close c_iamarks' at line 1
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server versio
n for the right syntax to use near 'end' at line 1
+------------+---------+------+-------+-------+-------+---------+
| usn        | subcode | ssid | test1 | test2 | test3 | finalia |
+------------+---------+------+-------+-------+-------+---------+
| 1by15cs02  | 15cs51  | 5b   |    13 |    14 |     0 |       0 |
| 1by15cs075 | 15cs53  | 3a   |    15 |    15 |    15 |       0 |
| 1by16cs417 | 15cs52  | 5a   |    12 |     0 |    15 |       0 |
+------------+---------+------+-------+-------+-------+---------+
3 rows in set (0.00 sec)
```

5. Categorize students based on the following criterion:
If FinalIA = 17 to 20 then CAT = 'Outstanding'
If FinalIA = 12 to 16 then CAT = 'Average'
If FinalIA< 12 then CAT = 'Weak'
Give these details only for 8th semester A, B, and C section students.

```
SQL> select student.usn,student.sname,student.address,student.gender,
  2  (case
  3  when iamarks.finalia  between 17 and 20 then 'outstanding'
  4  when iamarks.finalia  between 12 and 16 then 'average'
  5  else 'weak'
```

```
  6  end) as catagory
  7  from student,semsec,iamarks,subject where student.usn=iamarks.usn and
semsec.ssid=iamarks.ssid and subject.subcode=iamarks.subcode and subject.sem=8;

no rows selected   //no entries for 8th semester

SQL> select student.usn,student.sname,student.address,student.gender,
  (case
   when iamarks.finalia  between 17 and 20 then 'outstanding'
   when iamarks.finalia  between 12 and 16 then 'average'
   else 'weak'
   end) as catagory
    from student,semsec,iamarks,subject where student.usn=iamarks.usn and
semsec.ssid=iamarks.ssid and subject.subcode=iamarks.subcode and subject.sem=5;

USN          SNAME      ADDRESS    GENDER   CATAGORY
----------   ----------  ----------  ------   -----------
1by16cs417   vinod      abc street      male      weak
1by15cs02    def        abc street      male      weak
1by15cs075   abcde      abcstreet       female    weak
```

```
mysql> select student.usn,student.sname,student.address,student.gender,
      (case
  when iamarks.finalia  between 17 and 20 then 'outstanding'
     when iamarks.finalia  between 12 and 16 then 'average'
      else 'weak'
      end) as catagory
       from student,semsec,iamarks,subject where student.usn=iamarks.usn and semsec.ssid=iamarks.ssid and subject.subco
de=iamarks.subcode and subject.sem=8;
Empty set (0.00 sec)

mysql> select student.usn,student.sname,student.address,student.gender,
       (case
         when iamarks.finalia  between 17 and 20 then 'outstanding'
          when iamarks.finalia  between 12 and 16 then 'average'
          else 'weak'
          end) as catagory
          from student,semsec,iamarks,subject where student.usn=iamarks.usn and semsec.ssid=iamarks.ssid and subject.su
bcode=iamarks.subcode and subject.sem=5;
+------------+-------+-----------+--------+----------+
| usn        | sname | address   | gender | catagory |
+------------+-------+-----------+--------+----------+
| 1by15cs02  | def   | abc street | male   | weak     |
| 1by15cs075 | abcde | abcstreet  | female | weak     |
| 1by16cs417 | vinod | abc street | male   | weak     |
+------------+-------+-----------+--------+----------+
3 rows in set (0.00 sec)
```

E. Consider the schema for Company Database:
EMPLOYEE (*SSN, Name, Address, Sex, Salary, SuperSSN, DNo*)
DEPARTMENT (*DNo, DName, MgrSSN, MgrStartDate*)
DLOCATION (*DNo,DLoc*)
PROJECT (*PNo, PName, PLocation, DNo*)
WORKS_ON (*SSN, PNo, Hours*)
Write SQL queries to
1. Make a list of all project numbers for projects that involve an employee whose last name is 'Scott', either as a worker or as a manager of the department that controls the project.

2. Show the resulting salaries if every employee working on the 'IoT' project is given a 10 percent raise.

3. Find the sum of the salaries of all employees of the 'Accounts' department, as well as the maximum salary, the minimum salary, and the average salary in this department

4. Retrieve the name of each employee who works on all the projects controlled by department number 5 (use NOT EXISTS operator). For each department that has more than five employees, retrieve the department number and the number of its employees who are making more than Rs. 6,00,000.

Table Creation;

```
create table department (dno varchar(20) primary key,
dname varchar(20),
mgrstartdate date);
```

note: we must alter department table to add foreign key mgrssn**;**
```
alter table department add mgrssn references employee(ssn)
```

```
create table employee (ssn varchar(10) primary key,
fname varchar(10),
lname varchar(10),
addr varchar(30),
sex char (1),
salary integer,(10,2)
superssn varchar(10) references employee (ssn),
dno varchar(10) references department (dno));
```

```
alter table department add mgrssn varchar(10) references employee(ssn);
```

```
create table dlocation (dloc varchar(20),
dno integer(10) references department (dno),
primary key (dno, dloc));
```

```
create table project (pno integer primary key,
pname varchar(20),
ploc varchar(20),
```

```
dno varchar(10) references department (dno));
create table workson (noh integer (2),
ssn varchar(10) references employee (ssn),
pno integer(4) references project(pno),
primary key (ssn, pno));
```

Insertion of values to tables ;

```
insert into employee values (4000,'kumar','bangalore','male',80000,'null',101);
insert into employee values (4010,'scott','chennai','male',75000,'null',102);

insert into department values (101,'marketing','10-jan-01',4000);
insert into department values (102,'accounts','21-mar-2007',4010);

insert into delocation values(101,'banagalore');
insert into delocation values (102,'bangalore');

insert into project values (900,'iot','bangalore'101);
insert into project values (901,'datascience','chennai'101);
insert into project values (902,'machinelearning','bangalore'102);

insert into workson values (4000,900,10);
insert into workson values (4010,902,5);
insert into workson values (4001,900,0);
insert into workson values (4002,900,9);
insert into workson values (4011,902,8);
insert into workson values (4012,902,5);
```

Note: update entries of employee table to fill missing fields SUPERSSN and DNO

- ➔ Update employee set dno=101 where ssn=4000;
- ➔ Update employee set dno=102 where ssn=4010;

Queries:

1. Make a list of all project numbers for projects that involve an employee whose last name is 'Scott', either as a worker or as a manager of the department that controls the project.

```
A; select distinct pno from project, department, employee
where employee.dno= department.dno
and department.mgrssn= employee.ssn
and name='scott';
```

pno

900
901
902

```
mysql> desc department;
+--------------+-------------+------+-----+---------+-------+
| Field        | Type        | Null | Key | Default | Extra |
+--------------+-------------+------+-----+---------+-------+
| dno          | int(3)      | NO   | PRI | NULL    |       |
| dname        | varchar(10) | YES  |     | NULL    |       |
| mgrstartdate | date        | YES  |     | NULL    |       |
| mgrssn       | int(4)      | YES  |     | NULL    |       |
+--------------+-------------+------+-----+---------+-------+
4 rows in set (0.00 sec)

mysql> select distinct pno from project,department,employee where employee.dno=department.dno and department.mgrssn=empl
oyee.employee.ssn and name='scott';
ERROR 1054 (42S22): Unknown column 'employee.employee.ssn' in 'where clause'
mysql> select distinct pno from project,department,employee where employee.dno=department.dno and department.mgrssn=empl
oyee.ssn and name='scott';
Empty set (0.00 sec)

mysql> delete from table department;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version
 for the right syntax to use near 'table department' at line 1
mysql> delete from  department;
Query OK, 0 rows affected (0.00 sec)

mysql> insert into department values(101,'marketing','10-jan-2000',4000);
Query OK, 1 row affected, 1 warning (0.03 sec)

mysql> insert into department values(102,'accounts','21-mar-2007',4010);
Query OK, 1 row affected, 1 warning (0.03 sec)

mysql> select distinct pno from project,department,employee where employee.dno=department.dno and department.mgrssn=empl
oyee.ssn and name='scott';
+-----+
| pno |
+-----+
| 900 |
| 901 |
| 902 |
+-----+
3 rows in set (0.00 sec)
```

2. Show the resulting salaries if every employee working on the 'IoT' project is given a 10 percent raise.

A; select name, salary*1.1 as inc_sal
from employee where ssn in(select ssn from works-on where pno in(select pno from project where pname='iot'));

name	inc_sal
kumar	880000
raj	750000

3. Find the sum of the salaries of all employees of the 'Accounts' department, as well as the maximum salary, the minimum salary, and the average salary in this department

A; select sum (salary), max (salary), min (salary), avg (salary)
from employee where dno =(select dno from department where dname='ece');

sum(salary)	max(salary)	min(salary)	avg(salary)
2100000	75000	650000	700000

```
Query OK, 1 row affected (0.03 sec)

mysql> insert into workson values(4002,900,9);
Query OK, 1 row affected (0.03 sec)

mysql> insert into workson values(4011,900,8);
Query OK, 1 row affected (0.03 sec)

mysql> insert into workson values(4012,900,5);
Query OK, 1 row affected (0.03 sec)

mysql> select distinct pno from project,department,employee where employee.dno=department.dno and department.mgrssn=empl
oyee.employee.ssn and name='scott';
ERROR 1054 (42S22): Unknown column 'department.mgrssn' in 'where clause'
mysql> select distinct pno from project,department,employee where employee.dno=department.dno and department.mgrssn=empl
oyee.employee.ssn and name='scott';
ERROR 1054 (42S22): Unknown column 'department.mgrssn' in 'where clause'
mysql> SELECT E.FNAME, E.LNAME, 1.1*E.SALARY AS INCR_SAL
    -> FROM EMPLOYEE E, WORKS_ON W, PROJECT P
    -> WHERE E.SSN=W.SSN
    -> AND W.PNO=P.PNO
    -> AND P.PNAME='IOT';
ERROR 1146 (42S02): Table 'company.EMPLOYEE' doesn't exist
mysql> select name,salary*1.1 as inc_sal from employee where ssn in(select ssn from workson where pno in(select pno from
 project where pname='iot'));
+-------+---------+
| name  | inc_sal |
+-------+---------+
| kumar | 88000.0 |
+-------+---------+
1 row in set (0.00 sec)

mysql> select sum(salary),max(salary),min(salary),avg(salary)from employee where dno in(select dno from department where
 dname='accounts');
+-------------+-------------+-------------+-------------+
| sum(salary) | max(salary) | min(salary) | avg(salary) |
+-------------+-------------+-------------+-------------+
|        NULL |        NULL |        NULL |        NULL |
+-------------+-------------+-------------+-------------+
1 row in set (0.00 sec)
```

4. Retrieve the name of each employee who works on all the projects Controlled by department number 5 (use NOT EXISTS operator).

A; select fname from employee where not exists(select pno from project where dno=’d3’ minus select pno from workson where employee.ssn=workson.ssn);

no rows selected;

5. For each department that has more than five employees, retrieve the department number and the number of its employees who are making more than Rs. 6, 00,000.

A; select dno, count (*) from employee where dno in(select dno from employee having count(*)>5 group by((dno)) and salary>1000 group by(dno);

No rows selected;

Viva Questions

1. What is SQL?

Structured Query Language

2. What is database?

A database is a logically coherent collection of data with some inherent meaning, representing some aspect of real world and which is designed, built and populated with data for a specific purpose.

3. What is DBMS?

It is a collection of programs that enables user to create and maintain a database. In other words it is general-purpose software that provides the users with the processes of defining, constructing and manipulating the database for various applications.

4. What is a Database system?

The database and DBMS software together is called as Database system.

5. Advantages of DBMS?

- Redundancy is controlled.
- Unauthorized access is restricted.
- Providing multiple user interfaces.
- Enforcing integrity constraints.
- Providing backup and recovery.

6. Disadvantage in File Processing System?

- Data redundancy & inconsistency.
- Difficult in accessing data.
- Data isolation.
- Data integrity.
- Concurrent access is not possible.
- Security Problems.

7. Describe the three levels of data abstraction?

There are three levels of abstraction:

- Physical level: The lowest level of abstraction describes how data are stored.
- Logical level*:* The next higher level of abstraction, describes what data are stored in database and what relationship among those data.
- View level: The highest level of abstraction describes only part of entire database.

8. Define the "integrity rules"

There are two Integrity rules.

- Entity Integrity: States that “Primary key cannot have NULL value”
- Referential Integrity:States that “Foreign Key can be either a NULL value or should be Primary Key value of other relation.

9. What is extension and intension?

Extension - It is the number of tuples present in a table at any instance. This is time dependent.
Intension -It is a constant value that gives the name, structure of table and the constraints laid on it.

10. What is Data Independence?

Data independence means that "the application is independent of the storage structure and access strategy of data". In other words, The ability to modify the schema definition in one level should not affect the schema definition in the next higher level.
Two types of Data Independence:

- Physical Data Independence: Modification in physical level should not affect the logical level.
- Logical Data Independence: Modification in logical level should affect the view level.

NOTE: Logical Data Independence is more difficult to achieve

11. What is a view? How it is related to data independence?

A view may be thought of as a virtual table, that is, a table that does not really exist in its own right but is instead derived from one or more underlying base table. In other words, there is no stored file that direct represents the view instead a definition of view is stored in data dictionary.
Growth and restructuring of base tables is not reflected in views. Thus the view can insulate users from the effects of restructuring and growth in the database. Hence accounts for logical data independence.

A collection of conceptual tools for describing data, data relationships data semantics and constraints

13. What is E-R model?

This data model is based on real world that consists of basic objects called entities and of relationship among these objects. Entities are described in a database by a set of attributes.

14. What is Object Oriented model?

This model is based on collection of objects. An object contains values stored in instance variables within the object. An object also contains bodies of code that operate on the object. These bodies of code are called methods. Objects that contain same types of values and the same methods are grouped together into classes.

15. What is an Entity?

It is an 'object' in the real world with an independent existence.

16. What is an Entity type?

It is a collection (set) of entities that have same attributes.

17. What is an Entity set?

It is a collection of all entities of particular entity type in the database.

18. What is an Extension of entity type?

The collections of entities of a particular entity type are grouped together into an entity set.

19. What is an attribute?

It is a particular property, which describes the entity.

20. What is a Relation Schema and a Relation?

A relation Schema denoted by R(A1, A2, …, An) is made up of the relation name R and the list of attributes A_i that it contains. A relation is defined as a set of tuples. Let r be the relation which contains set tuples (t1, t2, t3, ...,tn). Each tuple is an ordered list of n-values t=(v1,v2, ..., vn).

21. What is degree of a Relation?

It is the number of attribute of its relation schema.

22. What is Relationship?

It is an association among two or more entities.

23. What is Relationship set?

24. *What is Relationship type?*

Relationship type defines a set of associations or a relationship set among a given set of entity types.

25. What is degree of Relationship type?

It is the number of entity type participating.

26. What is DDL (Data Definition Language)?

A data base schema is specified by a set of definitions expressed by a special language called DDL.

27. What is VDL (View Definition Language)?

It specifies user views and their mappings to the conceptual schema.

28. What is SDL (Storage Definition Language)?

This language is to specify the internal schema. This language may specify the mapping between two schemas.

29. What is Data Storage - Definition Language?

The storage structures and access methods used by database system are specified by a set of definition in a special type of DDL called data storage- definition language.

30. What is DML (Data Manipulation Language)?

This language that enable user to access or manipulate data as organized by appropriate data model.

- Procedural DML or Low level: DML requires a user to specify what data are needed and how to get those data.
- Non-Procedural DML or High level: DML requires a user to specify what data are needed without specifying how to get those data.

31. What is DML Compiler?

It translates DML statements in a query language into low-level instruction that the query evaluation engine can understand.

32. What is Relational Algebra?

It is a procedural query language. It consists of a set of operations that take one or two relations as input and produce a new relation.