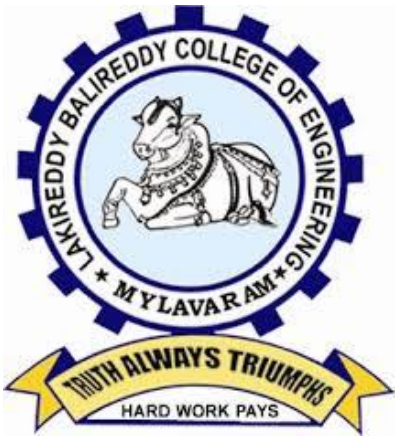
DATABASE MANAGEMENT SYSTEMS
LABARATORY MANUAL
(20CS56)

B.Tech. (II Year I Sem.)



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

LAKIREDDY BALI REDDY COLLEGE OF ENGINEERING

(AUTONOMOUS)
Accredited by NAAC & NBA (Under Tier - I) ISO 9001:2015 Certified Institution
Approved by AICTE, New Delhi. and Affiliated to JNTUK, Kakinada
L.B. REDDY NAGAR, MYLAVARAM, KRISHNA DIST., A.P.-521 230.

Vision of the Department

The Computer Science & Engineering aims at providing continuously stimulating educational environment to its students for attaining their professional goals and meet the global challenges.

Mission of the Department

- **DM1:** To develop a strong theoretical and practical background across the computer science discipline with an emphasis on problem solving.
- **DM2:** To inculcate professional behaviour with strong ethical values, leadership qualities, innovative thinking and analytical abilities into the student.
- **DM3:** Expose the students to cutting edge technologies which enhance their employability and knowledge.
- **DM4:** Facilitate the faculty to keep track of latest developments in their research areas and encourage the faculty to foster the healthy interaction with industry.

Program Educational Objectives (PEOs)

- **PEO1:** Pursue higher education, entrepreneurship and research to compete at global level.
- **PEO2:** Design and develop products innovatively in computer science and engineering and in other allied fields.
- **PEO3:** Function effectively as individuals and as members of a team in the conduct of interdisciplinary projects; and even at all the levels with ethics and necessary attitude.
- **PEO4:** Serve ever-changing needs of society with a pragmatic perception.

PROGRAMME OUTCOMES (POs):

PO 1	**Engineering knowledge**: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
PO 2	**Problem analysis**: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
PO 3	**Design/development of solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
PO 4	**Conduct investigations of complex problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
PO 5	**Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
PO 6	**The engineer and society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
PO 7	**Environment and sustainability**: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the

	knowledge of, and need for sustainable development.
PO 8	**Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
PO 9	**Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
PO 10	**Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
PO 11	**Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
PO 12	**Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change

PROGRAMME SPECIFIC OUTCOMES (PSOs):

PSO 1	The ability to apply Software Engineering practices and strategies in software project development using open-source programming environment for the success of organization.
PSO 2	The ability to design and develop computer programs in networking, web applications andIoT as per the society needs.
PSO 3	To inculcate an ability to analyze, design and implement database applications.

Pre-requisite : Programming language, Discrete Mathematical Structures, and Data Structures.

Course Educational Objective: The objective of this lab is to provide a strong formal foundation in database concepts, technology, and practice to the participants to groom them into well-informed database application developers.

Course Outcomes (CO): At the end of this course, the student will be able to:

CO1: Create & manipulate the relational database using SQL.(Apply- L3)
CO2: Implement Views, procedures, triggers, and cursors on relational database. (Apply- L3)
CO3: Create Unstructured Databases using MongoDB.(Apply- L3)
CO 4: Improve individual / teamwork skills, communication & report writing skills with ethical values.

Introduction: Language basics and example queries (one or two weeks).

1) Create a table STUDENT with appropriate data types and perform the following queries. Attributes are Roll number, student name, date of birth, branch and year of study.

a) Insert 5 to 10 rows in a table?
b) List all the students of all branches
c) List student names whose name starts with 's'.
d) List student names whose name contains 's as third literal.
e) List student names whose contains two 's' anywhere in the name
f) List students whose branch is NULL.
g) List students of CSE & ECE who born after 1980.
h) List all students in reverse order of their names.
i) Delete students of any branch whose name starts with 's'.
j) Update the branch of CSE students to ECE. k) Display student name padded with *'after the name of all the students.

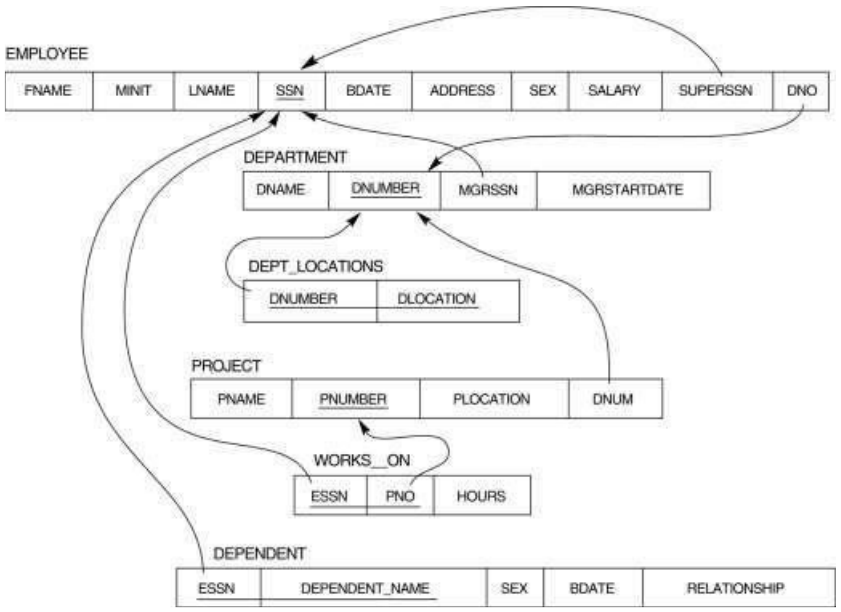
2) Create the following tables based on the above Schema Diagram with appropriate data Types and constraints and perform the following queries.

SAILORS (Saild, Salname, Rating, Age)
RESERVES (Sailid, boatid, Day)
BOATS (Boatid, Boat-name, Color)

a) Insert 5 to 10 rows in all tables?
b) Find the name of sailors who reserved boat number 3.
c) Find the name of sailors who reserved green boat.
d) Find the colors of boats reserved by Ramesh.
e) Find the names of sailors who have reserved at least one boat.
f) Find the allsailid of sailors who have a rating of 10 or have reserved boated 104.
g) Find the Sailid's of sailors with age over 20 who have not registered a red boat.
h) Find the names of sailors who have reserved a red or green boat.
i) Find sailors whose rating is better than some sailor called Salvador.
j) Find the names of sailors who are older than the oldest sailor with a rating of 10

3) Schema Diagram for the rest of the SQL and PLSQL Programs.
Create the following tables based on the above Schema Diagram with appropriate data types and constraints.

EMPLOYEE (Fname, Mname, Lname, SSN, Bdate, Address, Gender, Salary, SuperSSN,Dno)
DEPARTMENT (Dnumber, Dname, MgrSSN, Mgrstartdate)
DEPENDENT (ESSN, Dependent_Name, Gender, Bdate, Relationship)



a) Insert 5 to 10 rows into all the tables.
b) Display all employees' names along with their department names.
c) Display all employees' names along with their dependent details.
d) Display name and address of all employees who work for Research department.
e) List the names of all employees with two or more dependents.
f) List the names of employee who have no dependents.
g) List the names of employees who have at least one dependent.
h) List the names of the employees along with names of their supervisors using aliases.
i) Display name of the department and name of manager for all the departments.
j) Display the name of each employee who has a dependent with the same first name and gender as the employee.

4) Create the following tables based on the above Schema Diagram with appropriate data Types and constraints in addition to the tables in Experiment 2.

DEPT_LOCATIONS (Dnumber, Dloaction)
PROJECT (Pname, Pnumber, Plocation, Dnum)
WORKS_ON (ESSN, Pno, Hours)

a) Insert 5 to 10 rows into all the tables.
b) Find the names of the employees who work on all the projects controlled by the department Research.
c) List the project number, name and no. Of employees who work on that project for all the projects.
d) List the names of all the projects controlled by the departments department wise.
e) Retrieve the names of employees who work on all projects that John works on.
f) List the project numbers for projects that involve an employee either as worker or as a manager of the department that controls the project.
g) List the names of all employees in one department who work more than 10 hours on one specific project.
h) For each project, list the project name and total hours (by all employees) spent on that project.
i) Retrieve the names of all employees who work on every project.
j) Retrieve the names of all employees who do not work on any project.

5) Create a view that has project name, controlling department name, number of employees and total hours worked on the project for each project with more than one employee working on it.
a) List the projects that are controlled by one department from this view.
b) List the managers of the controlling departments for all the projects.
c) Demonstrate one update operation on this view.
d) List the Location of the controlling departments for all the projects.

e) Retrieve the data from the view.

6) Create a view emp from employee such that it contains only emp_noemp_name and department.
7) Create a view dept from department with only dept_no and location.
8) Create a view that contains the details of employees who are managers only.
9) Write a procedure to check whether the given number is Armstrong or not.
10) Write a procedure which accept the account number of a customer and retrieve the balance.
11) Write a procedure which accepts the student number and displays the department in which he belongs to.
12) Create a cursor to modify the salary of all employees belonging to 'Research' department by 150%.
13) Consider the college database. Retrieve all students who have registered for a specific course and store their details into another table using Cursors.
14) Write an update trigger on Account table. The system should keep track of the records that are being updated.
15) Create NoSQL database for a sample application and perform CURD operations.

Design Database for any one of the following Case Studies

Case Study1 : Hospital Management System
Case Study2 : Railway Reservation
Case Study3 : Painting Hire Business

1) Create a table STUDENT with appropriate data types and perform thefollowingqueries.

Attributes are Roll number, student name, date of birth, branch,andyearofstudy.

```
mysql>     create     table     student(Roll_number     int
primarykey,student_namevarchar(20),date_of_birthdate,branch
varchar(5),year_of_study int);
```

a) Insert 5 to10rowsinatable?

```
 mysql>insert into student( Roll_number,student_name,
        date_of_birth,branch,yer_of_stdy)values(222,"venkat","
        1991-09-26","cse",2020),
 (333,'siva','1990-04-10','AIDS',2021),
 (111,"srikanth","1990-03-16","cse",2020),
 (444,'Rajani','1980-05-12','IT',2010),
 (555,'Sindhu','1993-03-26','ECE',2017),
 (666,'Nayana','1995-05-05','AIML',2002);
```

```
 mysql>select *from student;
--+-+------+--- -+------+------+- -------------------- ------------------- --------------------
 |Roll_number|student_name|date_of_birth|branch|year_of_study|
--+-+------+--- -+------+------+- -------------------- ------------------- --------------------
 | 111|srikanth   |1990-03-16    |cse |    2020|
 | 222|venkat     |1991-09-26    | cse     |    2020|
 | 333|siva  |1990-04-10    |AIDS     |    2021|
 | 444|Rajani     |1980-05-12    | IT |2010     |
 | 555|Sindhu     |1993-03-26    |ECE |2017     |
 | 666|Nayana     |1995-05-05    |AIML     |2002     |
--+-+------+--- -+------+------+- -------------------- ------------------- --------------------
----------------------------------------------------------------------------------------------------
  ---
```

b)List all the students of all branches

```
 mysql>select student_name from student;
---++------------
 |student_name|
---++------------
 |srikanth  |
 |venkat    |
 | siva|
 |Rajani    |
 |Sindhu    |
 |Nayana    |
---++------------

----------------------------------------------------------------------------------------------------
-----
 6  rows inserted(0.00sec)
```

C)list all student names start with 's'

```
mysql>select student_name from student where student_name
       like's%';

-----+------------
|student_name|
+---+------------
|srikanth    |
| siva  |
|Sindhu |
+---+------------
  3  rowsinset(0.00sec)
```

---

d)List student names whose name contains 's' as the third literal

```
mysql>select * from student where student_name like 's%';
Emptyset(0.00sec)
```

---

e)liststudentnameswhosecontainstwo's'anywhere

```
mysql> select student_name from student where student_name
       like '%s%s%';
```

```
       Emptyset(0.00sec)
```

f)list of students whose branch is null

```
mysql>insert into student (Roll_number,student_name,
        date_of_birth,branch,year_of_study)values
       (777,'nandana','2003-04-28',null,2020);
mysql>select *from student;
--+--+------+--- --+------+------+- -------------------- ------------------- --------------------
  |Roll_number|student_name|date_of_birth|branch|year_of_study|
--+--+------+--- --+------+------+- -------------------- ------------------- --------------------
  | 111|srikanth   |1990-03-16    |cse |    2020|
  | 222|venkat     |1991-09-26    |cse |    2020|
  | 333|siva  |1990-04-10    |AIDS     |    2021|
  | 444|Rajani     |1980-05-12    | IT |    2010|
  | 555|Sindhu     |1993-03-26    |ECE |    2017|
  | 666|Nayana     |1995-05-05    |AIML     |    2002|
  | 777|nandana    |2003-04-28    |NULL     |    2020|
--+--+------+--- --+------+------+- -------------------- ------------------- --------------------
   7 rows in set (0.01 sec)

mysql>select *from student where branch isnull;
+-----+------+--- --+------+------+- -------------------- ------------------- --------------------
|Roll_number|student_name|date_of_birth|branch|year_of_study|
+-----+------+--- --+------+------+- -------------------- ------------------- --------------------
|   777  |nandana  |2003-04-28    | NULL    |    2020|
+-----+------+--- --+------+------+- -------------------- ------------------- --------------------
1 rowinset(0.00sec)
```

---

g)List students of CSE & ECE who born after 1980.

```
mysql>select * from student where branch in('cse','ECE')and
      date_of_birth>1980;
```

```
+ -----+------+--- -+------+------+- -------------------- ------------------- --------------------
|Roll_number|student_name|date_of_birth|branch|year_of_study|
+ -----+------+--- -+------+------+- -------------------- ------------------- --------------------
|    111|srikanth   |1990-03-16    |cse |    2020|
|    222|venkat     |1991-09-26    |cse |    2020|
|    555|Sindhu     |1993-03-26    |ECE |    2017|
+ -----+------+--- -+------+------+- -------------------- ------------------- --------------------
3rowsinset,1warning(0.00sec)
 ------------------------------------------------------------------------------------------------------
 ----------
```

h)List all students in reverse order of their names

```
mysql>select reverse(student_name)from student;
```

i) Delete students of any branch whose name starts with 's'.

```
mysql> delete from student where student_name like "s%";
       QueryOK,3rowsaffected(0.06sec)
mysql>select*from student;
+ -----+------+--- -+------+------+- -------------------- ------------------- --------------------
|Roll_number|student_name|date_of_birth|branch|year_of_study|
+ -----+------+--- -+------+------+- -------------------- ------------------- --------------------
|    222|venkat     |1991-09-26    |cse |    2020|
|    444|Rajani     |1980-05-12    | IT |    2010|
|    666|Nayana     |1995-05-05    |AIML     |    2002|
|    777|nandana    |2003-04-28    |NULL     |    2020|
+ -----+------+--- -+------+------+- -------------------- ------------------- --------------------
4 rowsinset(0.00sec)
```

To disable auto commit use

```
mysql>set autocommit=false;
QueryOK,0rowsaffected(0.03sec)
```

```
 ------------------------------------------------------------------------------------------------------
 --------------
```

i)update the branch of cse students to ece

```
mysql>update student set branch='ece' where branch='cse';
Query OK, 1 row affected (0.00 sec)Rows matched: 1
Changed: 1 Warnings: 0
mysql>select*from student;
+ -----+------+--- -+------+------+- -------------------- ------------------- --------------------
|Roll_number|student_name|date_of_birth|branch|year_of_study|
+ -----+------+--- -+------+------+- -------------------- ------------------- --------------------
|    222|venkat     |1991-09-26    |ece |    2020|
|    444|Rajani     |1980-05-12    |IT  |    2010|
|    666|Nayana     |1995-05-05    |AIML     |    2002|
|    777|nandana    |2003-04-28    |NULL     |    2020|
+ -----+------+--- -+------+------+- -------------------- ------------------- --------------------
4 rowsinset(0.00sec)
```

To create a savepoint we need to start the transtaction first.

```
mysql>starttransaction;
Query OK, 0 rows
affected
```

```
transaction-1
transaction-2
transaction-3
transaction-4
transaction-5
savepointA;

transaction-6
transaction-7
transaction-8
ifwerollbacktoA;
 ------------------------------------------------------------------------------------------------------
 --------------
thentransaction-6,7,8areremoved.
```

j)display student name padded with * after the name of all the students.

```
      mysql>selectRPAD(student_name,30,"*") as Name fromstudent;
  --+--+-----------------------------------
    | Name |
  --+--+-----------------------------------
    |venkat************************|
    |Rajani************************|
    |Nayana************************|
    |nandana***********************|
  --+--+-----------------------------------
    4 rowsinset(0.00sec)
```

2) Create the following tables with appropriate data types and constraints and perform the following queries.

SAILORS(Saild,Salname,Rating,Age)
RESERVES (Sailid, boatid, Day)
BOATS(Boatid,Boat-name,Color)

TABLE CREATION:-

```
       mysql>create table sailors(Sailid int primary key, Salname
       varchar(20),Ratingint,Ageint);

       mysql>create table boats (Boatid int primary key,Boat_name
       varchar(20),colorvarchar(10));

       mysql>create table reserves(Sailid int, Boatid int, day
       date, foreign key(Sailid)references sailors(Sailid),
 ------------------------------------------------------------------------------------------------------
 ---------
       foreign key(Boatid) references boats(Boatid));

mysql>show tables;
+ -----+----------------------
|Tables_in_20761A0589|
+ -----+----------------------
| boats   |
|reserves |
|sailors  |
+ -----+----------------------
```

```
3 rows in set (0.01 sec)
```

```
mysql>desc reserves;
+ -----+-----+-+----------+-------+- -----+- -----+------- ----------------
| Field   |Type|Null|Key|Default|Extra|
+ -----+-----+-+----------+-------+- -----+- -----+------- ----------------
| Sailid|int   |YES |MUL|NULL |     |
| Boatid|int   |YES |MUL|NULL |     |
| day     |date|YES |     |NULL     |     |
+ -----+-----+-+----------+-------+- -----+- -----+------- ----------------
rowsinset(0.00sec)

mysql>desc sailors;
+ -----+------+------+------+-- ------+------ --+----- ------------- ----------------
| Field   |Type     |Null|Key|Default|Extra|
+ -----+------+------+------+-- ------+------ --+----- ------------- ----------------
|Sailid   |int | NO |PRI|NULL |     |
|Salname|varchar(20)|YES |     | NULL    |     |
|Rating   |int |YES |     |NULL     |     |
| Age     |int |YES |     | NULL    |     |
+ -----+------+------+------+-- ------+------ --+----- ------------- ----------------
rowsinset(0.00sec)

mysql>desc boats;
+ -----+------+- ----+------+----- -+----------+-------- ---------------------- ----------------
| Field   |Type     |Null|Key|Default|Extra|
+ -----+------+- ----+------+----- -+----------+-------- ---------------------- ----------------
|Boatid   |int | NO |PRI|NULL |     |
|Boat_name|varchar(20)|YES    |     |NULL     |     |
| color   |varchar(10)|YES    |     |NULL     |     |
+ -----+------+- ----+------+----- -+----------+-------- ---------------------- ----------------
3 rows in set (0.00 sec)INSERTING DATA:-
```

```
   mysql> insert into sailors
   (Sailid,Salname,Rating,Age)values(22,'Dustin',7,45),(29,'Brutu
   s',1,33),(31,'Lubber',8,55),
   (32,'Andy',8,25),(58,'Rusty',10,35),(64,'Horatio',7,35),
          (71,'Zobra',10,16), (74,'Horatio',9,35),
   (85,'Art',3,25),(95,'Bob',3,63.5);

   mysql>insert into boats(Boatid,Boat_name,color)
   values(101,'Interlake','Blue'),(102,'Interlake','Red'),(103,'Cl
   ipper','Green'),(104,'Marine','Red');

          mysql>insert into  reserves(Sailid,Boatid,Day)
          values(22,101,'1998-10-10'),(22,102,'1998-08-
          13'),(22,103,'1984-05-24'), (22,104,'1990-06-
          13'),(31,102,'1997-02-13'),(31,103,'1998-06-
          11'),(31,104,'1998-12-11'),(64,101,'1998-05-
          09'),(64,102,'1998-07-09'),(74,103,'1998-07-09');
```

a)Find the name of sailors who reserved boat number 3.

```
mysql> select s.Salname from sailors s,reserves r where
s.Sailid=r.Sailidandr.Boatid=3;
mysql>show tables;
```

```
--------+---+---------------
  |Tables_in_20761A0589|
---++----------------------
  | boats    |
  |reserves  |
  |sailors   |
---++----------------------
```

```
mysql>select * from boats;
------+---- -+---------+-------- +--------------
     |Boatid|Boat_name |color|
------+---- -+---------+-------- +---------
     | 101    |Interlake|Blue     |
     | 102    |Interlake|Red |
     | 103    |Clipper  |Green|
     | 104    |Marine   |Red |
------+---- -+---------+-------- +---------
```

```
mysql>select *from reserves;
---------- ---------------- ----------------
  | Sailid|Boatid|day  |
---++----- -+---------+--- ----------------
  |22   |101 |1998-10-10|
  |22   |102 |1998-08-13|
  |22   |103 |1984-05-24|
  |22   |104 |1990-06-13|
  |31   |102 |1997-02-13|
  |31   |103 |1998-06-11|
  |31   |104 |1998-12-11|
  |64   |101 |1998-05-09|
  |64   |102 |1998-07-09|
  |74   |103 |1998-07-09|
---++----- -+---------+--- ----------------
```

```
   mysql>select*fromsailors;
-----+----- -+---------+---- --+------+ --------------
   |Sailid|Salname| Rating|Age   |
-----+----- -+---------+---- --+------+ --------------
   |22  |Dustin   |7   | 45 |
   |29  |Brutus   |1   | 33 |
   |31  |Lubber   |8   | 55 |
   |32  |Andy     |8   | 25 |
   |58  |Rusty    |10  | 35 |
   |64  |Horatio|7     | 35 |
   |71  |Zobra    |10  | 16 |
   |74  |Horatio|9     | 35 |
   |85  |Art  | 3  |25  |
```

```
|95  |Bob | 3  | 64 |
-----+----- -+-----------+----- ---+-------+ --------------
```

```
mysql> select s.Salname from sailors s, reserves r where
       s.Sailid =r.Sailidandr.Boatid=103;
--+--+------
 |Salname|
--+--+------
 |Dustin|
 |Lubber|
 |Horatio|
--+--+------
```

----

3 rowsinset(0.00sec)

b)Find the name of sailors who reserved green boat.

```
mysql>select s.Salname from sailors s, reserves r, boats b
      where s.Sailid=r.Sailid and r.Boatid=b.Boatid and
      b.color='Green';
--+--+------
 |Salname|
--+--+------
 |Dustin|
 |Lubber|
 |Horatio|
--+--+------
 rowsinset(0.00sec)
```

----

c)Find the color of boats reserved by Dustin

```
mysql> select color from boats inner join reserves on
       reserves.Boatid =boats.Boatid innerjoin sailors on
       sailors.Sailid=reserves.Sailid where Salname='Dustin';
+---+---
| color|
+---+---
| Blue  |
| Red   |
| Green|
| Red   |
+---+---
```

d) Find the names of the sailors who have reserved atleast one boat.

```
mysql>select s.Salname from sailors s,reserves r where
      s.Sailid=r.Sailid;
--+--+------
 |Salname|
--+--+------
 |Dustin|
 |Dustin|
 |Dustin|
```

```
|Dustin|
|Lubber|
|Lubber|
|Lubber|
|Horatio|
|Horatio|
|Horatio|
--+--+-----
10rowsinset(0.00 sec)
```

---

e) Find the all sailid of sailors who have a rating of 10 or have reserved boate 104.

```
mysql>select Salname from sailors innerjoin reserves on
       sailors.Sailid=reserves.Sailid where Rating=10 o
       rBoatid=104;
--+--+-----
|Salname|
--+--+-----
|Dustin |
|Lubber |
--+--+-----
```

```
2  rowsinset(0.00sec)
```

f) Find the Sailid's of sailors with age over 20 who have not registereda redboat.

```
mysql> select distinct s.Sailid from sailors s,boats
       b,reserves r where s.Sailid = r.Sailid and r.Boatid =
       b.Boatid and s.Age> 20 and b.color !='Red';
+ -----+-----
|Sailid|
+ -----+-----
|    22|
|    64|
|    31|
|    74|
+ -----+-----
```

g) Find the names of sailors who have reserved a red or green boat.

```
mysql> select s.Salname from sailors s, boats b, reserves r
       where s.Sailid = r.Sailid and r.Boatid = b.Boatid and
       (b.color = 'Red' orb.color='Green');
---++-----
  |Salname|
---++-----
  |Dustin    |
  |Lubber    |
  |Horatio|
  |Dustin    |
  |Lubber    |
  |Horatio|
  |Dustin    |
  |Lubber    |
```

```
   7  rowsinset(0.00sec)
---------------------------------------------------------------------------------------------------------
----------------
```

h) Find sailors whose rating is better than some sailor called Salvador.

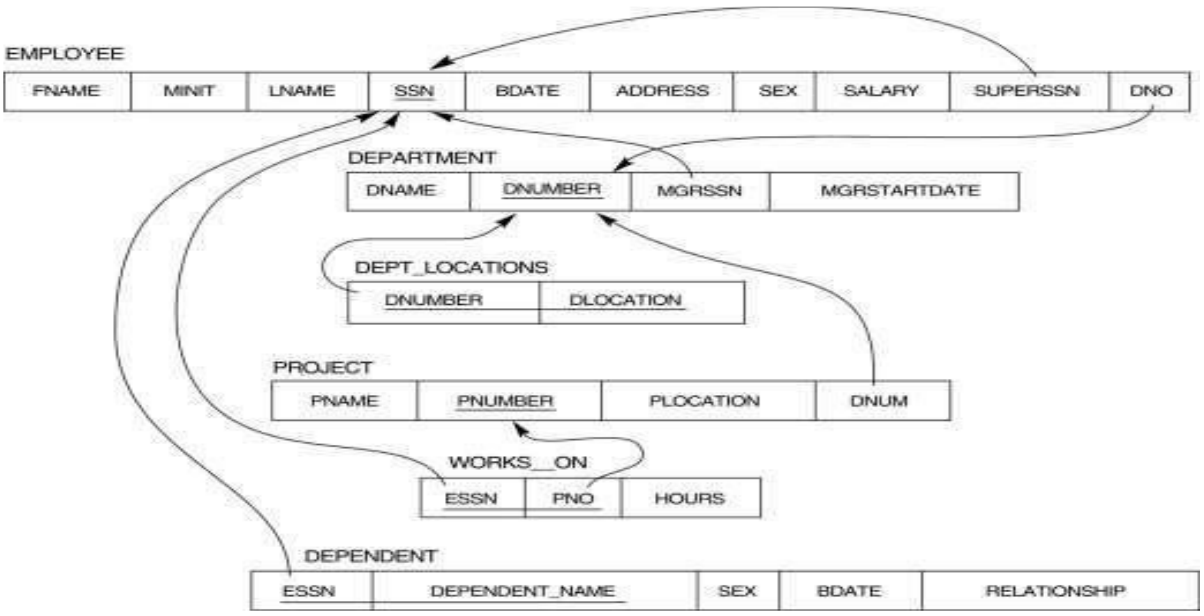
```
  mysql> select Salname from sailors where Rating > (select
         Rating from sailors where Salname='Dustin');
+---+------
|Salname|
+---+------
|Lubber |
| Andy  |
| Rusty |
| Zobra |
|Horatio|
+---+------

---------------------------------------------------------------------------------------------------------
----------------
            4  rowsinset(0.00sec)
```

i) Find the names of sailors who are older than the oldest sailor with aratingof10.

```
mysql>select Salname from sailors where Age>(select max(Age)from
       sailors where Rating=10);
+ -----+------
|Salname|
+ -----+------
|Dustin   |
|Lubber   |
| Bob     |
+ -----+------
```

3) Schema Diagram for the rest of the SQL and PLSQL Programs.Create the following tables based on the above Schema Diagram withappropriatedatatypesandconstraints.



EMPLOYEE (Fname, Mname, Lname, SSN, Bdate, Address, Gender, Salary,SuperSSN,Dno)
DEPARTMENT(Dnumber,Dname,MgrSSN,Mgrstartdate)
DEPENDENT (ESSN, Dependent_Name, Gender, Bdate, Relationship)

TABLECREATION:-

DEPARTMENT TABLE:-

```
CREATE TABLE DEPARTMENT (DNO VARCHAR(20) PRIMARY KEY, DNAME
VARCHAR(20),MGRSTARTDATE DATE);
```

EMPLOYEETABLE:

```
CREATE TABLE EMPLOYEE (FNAME VARCHAR(20), MNAME VARCHAR(20),
LNAME VARCHAR(20), SSN VARCHAR (20) PRIMARY KEY, DOB DATE,
ADDRESS VARCHAR(20),GENDER VARCHAR(10), SALARY INTEGER, SUPERSSN
VARCHAR(20)REFERENCES EMPLOYEE(SSN),DNO VARCHAR(20)REFERENCES
DEPARTMENT(DNO));
```

NOTE: Once DEPARTMENT and EMPLOYEE tables are created we must alter department table to add foreign constraint MGRSSN using sql command

```
ALTER TABLE DEPARTMENT ADD MGRSSN VARCHAR(20)REFERENCES
EMPLOYEE(SSN);
```

DEPENDENTTABLE:

```
CREATE TABLE DEPENDENT (ESSN VARCHAR(20) REFERENCES
EMPLOYEE(SSN),DEPENDENTNAME VARCHAR(20), GENDER VARCHAR(20), DOB
DATE, RELATIONSHIP VARCHAR(20));
```

DLOCATIONTABLE:

```
CREATE TABLE DLOCATION(DLOC VARCHAR(20),DNO VARCHAR(20)
REFERENCES DEPARTMENT(DNO),PRIMARY KEY(DNO,DLOC));
```

PROJECTTABLE:

```
CREATE TABLE PROJECT (PNAME VARCHAR(20),PNO INTEGER PRIMARY
KEY,PLOCATION VARCHAR(20),DNO VARCHAR(20)REFERENCES
DEPARTMENT(DNO));
```

WORKS_ONTABLE:

```
CREATE TABLE WORKS_ON(ESSN VARCHAR(20) REFERENCES
EMPLOYEE(SSN),PNO INTEGER REFERENCES PROJECT(PNO),PRIMARY
KEY(ESSN,PNO),HOURS INTEGER);
```

a)Insert 5 to 10 rows into all the tables.

INSERT DATA INTO EMPLOYEE:

```
    INSERT INTO EMPLOYEE (FNAME , MNAME , LNAME , SSN , DOB ,
    ADDRESS ,GENDER,SALARY , SUPERSSN , DNO )
    VALUES('John','B','Smith','123456789','1965-02-
    09','731Fondren','M',30000,'333445555',5),('Franklin','T','Wo
    ng','333445555','1955-12-
    08','638Voss','M',40000,'888665555',5),('Alicia','J','Zelaya'
    ,'999887777','1968-01-19','3321
    Castle','F',25000,'987654321',4),('Jennifer','S','Wallance','9
    87654321','1941-06-20','291Berry','F', 43000,'888665555',4),
```

```
('Ramesh','K','Narayana','666884444','1962-09-15','975
FireOak','M',38000,'333445555',5),('Joyce','A','English','453
453453','1972-07-31','5631Rice','F',25000,'333445555',5)
,('Ahmad','V','Jabbar','987987987','1969-03-22','980Dallas',
'M',25000,'987987987',4),('James','E','Brog','888665555','193
7-10-10','450Stone','M',55000,'NULL',1);
```

INSERTINTODEPARTMENT:

```
INSERT INTO DEPARTMENT (DNO, DNAME,MGRSTART DATE,MGRSSN)
VALUES('5','Research','1988-05-
22','333445555'),('4','Administration','1995-01-
01','987654321'),('1','Headquarters','1981-06-
19','888665555');
```

INSERTINTODEPENDENT:

```
INSERT INTO DEPENDENT (ESSN,DEPENDENTNAME,GENDER,DOB,
RELATIONSHIP)VALUES('333445555','Alice','F','1986-04-
05','Daughter'),('333445555','Theodore','M','1983-10-
25','Son'),('333445555','Joy','F','1958-05-
03','Spouse'),('987654321','Abner','M','1942-02-
28','Spouse'),('123456789','Michael','M','1988-01-
04','Son'),('123456789','Elizabeth','F','1967-05-05','Spouse');
```

INSERTINTODLOCATION:

```
INSERT INTO DLOCATION(DLOC,DNO)VALUES
('Houstan','1'),
('Stafford','4'),
('Bellaire','5'),
('Sugarland','5'),
('Houstan','5');
```

INSERTINTOPROJECT:

```
INSERT INTO PROJECT(PNAME,PNO,      PLOCATION,DNO)VALUES
('ProductX',1,'Bellaire','5'),
('ProductY',2,'Sugarland','5'),
('ProductZ',3,'Houstan','5'),
('Computerization',10,'Stafford','4'),
('Reorganization',20,'Houstan','1'),
('Newbenefits',30,'Stafford','4');
```

INSERTINTOWORKS_ON:

```
INSERT INTO WORKS_ON(ESSN,PNO,HOURS)
VALUES('123456789',1,32),('123456789',2,47),
('666884444',3,40),
('453453453',1,20),
('333445555',2,20),
('333445555',1,10),
('333445555',3,10),
('333445555',10,10),
('999887777',20,10),
('999887777',30,30),
('987987987',10,10),
('987987987',11,35),
```

```
('987654321',30,5),
('987654321',31,20),
('888665555',20,15);
```

b) Display all employees' names along with their department names.

```
mysql> CREATE VIEW RESULTB AS (SELECT FNAME,LNAME,DNAME FROM
       EMPLOYEE,DEPARTMENT WHERE EMPLOYEE.DNO=DEPARTMENT.DNO);
Query OK, 0 rows affected (0.06 sec)
mysql>SELECT * FROM RESULTB;
+ -----+------+ ---------+---------- ----------------------------------
| FNAME    |LNAME     |DNAME     |
+ -----+------+ ---------+---------- ----------------------------------
| John     |Smith     |Research  |
|Franklin|Wong  |Research  |
| Joyce    |English   |Research  |
|Ramesh    |Narayana|Research   |
| James    |Brog      |Headquarters   |
|Jennifer|Wallance|Administration|
| Ahmad    |Jabbar    |Administration|
|Alicia    |Zelaya    |Administration|
+ -----+------+ ---------+---------- ----------------------------------
8 rowsinset(0.02sec)
```

c) Display all employees' names along with their dependent details

```
mysql> CREATE VIEW RESULTC AS (SELECT FNAME,LNAME,DEPENDENTNAME
       FROM EMPLOYEE,DEPENDENT WHERE
       DEPENDENT.ESSN=EMPLOYEE.SSN);
Query OK, 0 rows affected (0.05 sec)
mysql> SELECT* FROM RESULTC;
+ -----+------+ ---------+---------- ----------------------------------
| FNAME    |LNAME     |DEPENDENTNAME|
+ -----+------+ ---------+---------- ----------------------------------
|Franklin|Wong  |Alice     |
|Franklin|Wong  |Theodore  |
|Franklin|Wong  |Joy  |
|Jennifer|Wallance|Abner  |
| John     |Smith     |Michael   |
| John     |Smith     |Elizabeth      |
+ -----+------+ ---------+---------- ----------------------------------
6 rowsinset(0.00sec)
```

d) Display name and address of all employees who work for Researchdepartment.

```
mysql> CREATE VIEW RESULTD AS (SELECT FNAME,LNAME,ADDRESS FROM
       EMPLOYEE,DEPARTMENT WHERE EMPLOYEE.DNO=DEPARTMENT.DNO
       ANDDNAME='RESEARCH');
```

```
Query OK, 0 rows affected (0.05 sec)mysql>SELECT*FROM RESULTD;
+ -----+------+ --------+---------- ------------------------------
| FNAME    |LNAME     |ADDRESS   |
+ -----+------+ --------+---------- ------------------------------
| John     |Smith     |731Fondren     |
|Franklin|Wong |638Voss   |
| Joyce    |English   |5631Rice |
|Ramesh    |Narayana|975FireOak|
+ -----+------+ --------+---------- ------------------------------
4 rowsinset(0.00sec)
 ------------------------------------------------------------------------------------------------------------
---
```

e)List the names of all employees with two or more dependents

```
mysql> CREATE VIEW RESULTE AS (SELECT FNAME,LNAME FROM EMPLOYEE
       WHERE(SELECTCOUNT(*)FROM DEPENDENT WHERE SSN=ESSN)>=2);
Query OK, 0 rows affected (0.06 sec)
mysql>SELECT * FROM RESULTE;
+ -----+------+ ---------------
| FNAME    |LNAME|
+ -----+------+ ---------------
| John     |Smith|
|Franklin|Wong |
+ -----+------+ ---------------
2 rowsinset(0.00sec)
 ------------------------------------------------------------------------------------------------------------------------------------------------------
---
```

f)List the names of employee who have no dependents.

```
mysql> CREATE VIEW RESULTF AS (SELECT FNAME,LNAME FROM EMPLOYEE
       WHERE NOTEXISTS(SELECT*FROM DEPENDENTWHERESSN=ESSN));
Query OK, 0 rows affected (0.06 sec)mysql>SELECT*FROM RESULTF;
+ -----+----- -+-----------------
| FNAME    |LNAME     |
+ -----+----- -+-----------------
| Joyce    |English   |
|Ramesh|Narayana|
| James    |Brog      |
| Ahmad    |Jabbar    |
|Alicia|Zelaya |
+ -----+----- -+-----------------
5 rowsinset(0.01sec)
 ------------------------------------------------------------------------------------------------------------
 --------
```

g)List the names of employees who have atleast one dependent.

```
mysql>CREATE VIEW RESULTG AS(SELECT FNAME,LNAME FROM EMPLOYEE
       WHERE EXISTS (SELECT * FROM DEPENDENT WHERE SSN=ESSN) AND
       EXISTS (SELECT * FROM DEPARTMENT WHERE SSN=MGRSSN));
Query OK, 0 rows affected (0.05 sec)
mysql>SELECT*FROM RESULTG;
```

```
+ -----+------+ ----------
| FNAME    |LNAME     |
+ -----+------+ ----------
|Franklin|Wong |
|Jennifer|Wallance|
+ -----+------+ ----------
2 rowsinset(0.00sec)
```

h)List the names of the employees along with names of their supervisorsusingaliases.

```
mysql> CREATE VIEW RESULTH AS (SELECT E1.FNAME,E1.LNAME,E2.FNAME
       AS SUPERVISOR FROM EMPLOYEEE1,EMPLOYEEE2 WHERE
       E2.SSN=E1.SUPERSSN);
Query OK, 0 rows affected (0.05 sec)
mysql>SELECT *FROM RESULTH;
+ -----+------+ ---------+---------- ----------
| FNAME    |LNAME     |SUPERVISOR|
+ -----+------+ ---------+---------- ----------
| John     |Smith     |Franklin |
|Franklin|Wong |James     |
| Joyce    |English   |Franklin |
|Ramesh    |Narayana|Franklin  |
|Jennifer|Wallance|James |
| Ahmad    |Jabbar    |Ahmad     |
|Alicia    |Zelaya    |Jennifer |
+ -----+------+ ---------+---------- ----------
7 rowsinset(0.00sec)
```

i)Display name of the department and name of manager for all thedepartments.

```
mysql> CREATE VIEW RESULTI AS (SELECT DNAME,FNAME FROM EMPLOYEE
       E,DEPARTMENT D WHERE E.SSN=D.MGRSSN);
Query OK, 0 rows affected (0.05 sec)
mysql>SELECT*FROM RESULTI;
+ -----+------+--------- ----------
| DNAME    |FNAME     |
+ -----+------+--------- ----------
|Headquarters  |James     |
|Administration|Jennifer|
|Research |Franklin|
+ -----+------+--------- ----------
rowsinset(0.00sec)
```

j)Display the name of each employee who has a dependent with the samefirstnameandgenderastheemployee.

```
mysql> CREATE VIEW RESULTJ AS (SELECT E.FNAME,E.LNAME FROM
EMPLOYEE AS EWHERE E.SSN IN (SELECT ESSN FROM EMPLOYEE,DEPENDENT
WHERE FNAME=DEPENDENTNAME AND EMPLOYEE.GENDER=DEPENDENT.GENDER));
```

```
Query OK, 0 rows affected (0.05 sec)
mysql>SELECT *FROM RESULTJ;
Emptyset(0.00sec)
```

4) Create the following tables based on the above Schema Diagram with appropriate data types and constraints in addition to the tables in Experiment

DEPT_LOCATIONS(Dnumber,Dloaction)
PROJECT (Pname, Pnumber, Plocation,Dnum)
WORKS_ON(ESSN,Pno,Hours)

TABLE CREATION :-

DLOCATIONTABLE:

```
CREATE TABLE DLOCATION(DLOC VARCHAR(20),DNO VARCHAR(20)
REFERENCESDEPARTMENT(DNO),PRIMARYKEY(DNO,DLOC));
```

PROJECTTABLE:

```
CREATE TABLE PROJECT (PNAME VARCHAR(20),PNO INTEGER PRIMARY
KEY,PLOCATIONVARCHAR(20),DNOVARCHAR(20)REFERENCESDEPARTMENT(DNO))
;
```

WORKS_ONTABLE:

```
CREATE TABLE WORKS_ON(ESSN VARCHAR(20) REFERENCES
EMPLOYEE(SSN),PNOINTEGERREFERENCESPROJECT(PNO),PRIMARYKEY(ESSN,PN
O),HOURSINTEGER);
```

a) Insert 5 to 10 rows into all the tables.

```
INSERT INTO DLOCATION(DLOC,DNO)VALUES ('Houstan','1'),
('Stafford','4'), ('Bellaire','5'), ('Sugarland','5'),
('Houstan','5');

INSERT INTO PROJECT(PNAME,PNO,PLOCATION,DNO)VALUES
('ProductX',1,'Bellaire','5'), ('ProductY',2,'Sugarland','5'),
('ProductZ',3,'Houstan','5'),
('Computerization',10,'Stafford','4'),
('Reorganization',20,'Houstan','1'),('Newbenefits',30,'Stafford',
'4');

INSERT INTO WORKS_ON(ESSN,PNO,HOURS) VALUES('123456789',1,32),
('123456789',2,47),('666884444',3,40),('453453453',1,20),
('333445555',2,20), ('333445555',1,10), ('333445555',3,10),
('333445555',10,10), ('999887777',20,10), ('999887777',30,30),
('987987987',10,10), ('987987987',11,35), ('987654321',30,5),
('987654321',31,20), ('888665555',20,15);
```

a) Find the names of the employees who work on all the projects controlled by the department Research.

```
mysql> CREATE VIEW RESULT4B AS (SELECT DISTINCT E.FNAME,E.LNAME
       FROMEMPLOYEE E,DEPARTMENT D, PROJECT P,WORKS_ON W WHERE
       D.DNAME='RESEARCH'ANDD.DNO=P.DNOANDW.ESSN=E.SSNANDP.PNO=W.
       PNO);
```

```
Query OK, 0 rows affected (0.05 sec)
mysql>SELECT *FROM RESULT4B;
+ -----+------+ ----------------------
| FNAME    |LNAME     |
+ -----+------+ ----------------------
| John     |Smith     |
|Franklin|Wong |
| Joyce    |English   |
|Ramesh    |Narayana|
+ -----+------+ ----------------------
rowsinset(0.00sec)
```

b)List the project number, name and no. Of employees who work on that project for all the projects.

```
mysql> CREATE VIEW RESULT4C AS (SELECT .PNO,P.PNAME,COUNT(W.ESSN)
       FROM PROJECT P,WORKS_ON W WHERE P.PNO=W.PNO GROUPBY
       P.PNO,P.PNAME);
Query OK, 0 rows affected (0.09 sec)
mysql>SELECT *FROM RESULT4C;
+ -----+- -----+------+---------- ----------------------------------
| PNO|PNAME     |COUNT(W.ESSN)|
+ -----+- -----+------+---------- ----------------------------------
|     1|ProductX      |     3|
|     2|ProductY      |     2|
|     3|ProductZ      |     2|
|     10|Computerization| 2|
|     20|Reorganization    |     2|
|     30|Newbenefits  |     2|
+ -----+- -----+------+---------- ----------------------------------

 ------------------------------------------------------------------------------------------------------
             5  rowsinset(0.01sec)
```

c)List the names of all the projects controlled by the departments department wise.

```
mysql> CREATE VIEW RESULT4D AS (SELECT P.PNAME,D.DNAME FROM
       PROJECT P,DEPARTMENT D WHERE P.DNO=D.DNO);
Query OK, 0 rows affected (0.06 sec)
mysql>SELECT*FROM RESULT4D;
+ -----+------+---------- ----------------------
| PNAME    |DNAME     |
+ -----+------+---------- ----------------------
|ProductX  |Research  |
|ProductY  |Research  |
|ProductZ  |Research  |
|Computerization|Administration|
|Reorganization     |Headquarters   |
|Newbenefits    |Administration|
+ -----+------+---------- ----------------------

             6  rowsinset(0.01sec)
```

d) Retrieve the names of employees who work on all projects that Johnworkson.

```
mysql> CREATE VIEW RESULT4E AS (SELECT DISTINCT E.FNAME,E.LNAME
       FROM EMPLOYEE E,WORKS_ON W WHERE E.SSN=W.ESSN AND W.PNO IN
       (SELECT W.PNO  FROM EMPLOYEE E,WORKS_ON W WHERE
       E.SSN=W.ESSN AND E.FNAME='JOHN'));
Query OK, 0 rows affected (0.10 sec)
mysql>SELECT *FROM RESULT4E;
+ -----+------+ ----------------------
| FNAME    |LNAME     |
+ -----+------+ ----------------------
| John     |Smith     |
|Franklin|Wong |
| Joyce    |English |
+ -----+------+ ----------------------

 ---------------------------------------------------------------------------------------------------------
      3  rowsinset(0.00sec)
---
```

e)List the project numbers for projects that involve an employee either as worker or as a manager of the department that controls the project.

```
mysql> CREATE VIEW RESULT4F AS (SELECT DISTINCT P.PNO FROM
       PROJECT P,DEPARTMENT D,EMPLOYEE E WHERE    D.MGRSSN=E.SSN
       AND D.DNO=P.DNO)UNION(SELECT DISTINCT P.PNO FROM EMPLOYEE
       E,PROJECT P,WORKS_ON W WHEREE.SSN=W.ESSNANDP.PNO=W.PNO);
Query OK, 0 rows affected (0.10 sec)
mysql>SELECT *FROM RESULT4F;
+ -----+·
|PNO|
+ -----+·
|    20 |
|    10 |
|    30 |
|    1|
|    2 |
|    3 |
+ -----+·
```

---

f)List the names of all employees in one department who work more than 10 hours on one specific project.

```
mysql> CREATE VIEW RESULT4G AS (SELECT E.FNAME,E.LNAME FROM
       EMPLOYEE E,PROJECT P,DEPARTMENT D,WORKS_ON W WHERE
       D.DNO=E.DNO ANDD.DNAME='RESEARCH' AND D.DNO=P.DNO AND
       P.PNAME='PRODUCTX' AND P.PNO=W.PNOANDW.ESSN=E.SSNAND
       W.HOURS>10);
Query OK, 0 rows affected (0.11 sec)
mysql>SELECT *FROM RESULT4G;
```

```
+ -----+---- --+---------
| FNAME|LNAME   |
+ -----+---- --+---------
| John     |Smith     |
|Joyce|English|
+ -----+---- --+---------
```


```
  2  rowsinset(0.00sec)
```

g)For each project, list the project name and total hours (by all employees)spent on that project.

```
mysql> CREATE VIEW RESULT4H AS (SELECT P.PNAME,SUM(W.HOURS) FROM
   PROJECT P,WORKS_ON W WHERE P.PNO=W.PNO GROUPBY P.PNAME,P.PNO);
Query OK, 0 rows affected (0.04 sec)mysql>SELECT*FROM RESULT4H;
+ -----+------+--------- -------------------
| PNAME    |SUM(W.HOURS)|
+ -----+------+--------- -------------------
|Computerization|    20 |
|Newbenefits    |     35|
|ProductX |     62 |
|ProductY |     67 |
|ProductZ |     50 |
|Reorganization      |     25 |
+ -----+------+--------- -------------------
```


```
            7  rowsinset(0.00sec)
```

h)Retrieve the names of all employees who workon every project.

```
mysql> CREATE VIEW RESULT4I AS (SELECT E.FNAME FROM EMPLOYEE E
       WHEREE.SSN IN (SELECT W.ESSN FROM WORKS_ON W WHERE
       W.PNO=ALL(SELECT PNO FROM PROJECT)));
Query OK, 0 rows affected (0.06 sec)

mysql>SELECT*FROM RESULT4I;
Emptyset(0.00sec)
```

---

i)Retrieve the names of all employees who do not work on any project.

```
mysql> CREATE VIEW RESULT4J AS (SELECT E.FNAME,E.LNAME FROM
       EMPLOYEE E WHERE E.SSN NOTIN(SELECT W.ESSN FROM WORKS_ON
       W));
Query OK, 0 rows affected (0.08 sec)
mysql>SELECT*FROM RESULT4J;
      Emptyset(0.01sec)
```

5) Create a view that has project name, controlling department name,number of employees and total hours worked on the project for each project with more than on employee working on it.

```
mysql> CREATE VIEW PROJECT_VIEW(PNAME,DNAME,NOOFEMP,NOOFHRS) AS
SELECT P.PNAME,D.DNAME,COUNT(W.ESSN),SUM(W.HOURS) FROM PROJECT
P,DEPARTMENT D,WORKS_ON W WHERE P.DNO=D.DNO AND P.PNO=W.PNO GROUP
BY W.PNO,P.PNAME,D.DNAME;
mysql>SELECT * FROM PROJECT_VIEW;
+ -----+------+------+-- ----+----------------- ------------ ----------------------
| PNAME    |DNAME     |NOOFEMP|NOOFHRS|
+ -----+------+------+-- ----+----------------- ------------ ----------------------
|ProductX |Research |     3 |  62|
|ProductY |Research |     2 |  67|
|ProductZ |Research |     2 |  50|
|Computerization|Administration|    2 |  20|
|Reorganization      |Headquarters  |     2 |  25|
|Newbenefits    |Administration|     2 |  35|
+ -----+------+------+-- ----+----------------- ------------ ----------------------
6 rowsinset(0.00sec)
```

---

a) List the projects that are controlled by one department from this view.

```
mysql> CREATE VIEW RESULT5A AS (SELECT PNAME FROM PROJECT_VIEW
       WHERE DNAME='RESEARCH');
Query OK, 0 rows affected (0.09 sec)
mysql>SELECT * FROM RESULT5A;
+ -----+--------
| PNAME    |
+ -----+--------
|ProductZ|
|ProductX|
|ProductY|
+ -----+--------
3 rowsinset(0.00sec)
```

---

b) List the managers of the controlling departments for all the projects.

```
mysql> CREATE VIEW RESULT5B AS (SELECT E.FNAME, E.LNAME, P.DNAME,
       P.PNAMEFROM EMPLOYEE E, PROJECT_VIEW P,DEPARTMENT D WHERE
       E.SSN=D.MGRSSN ANDD.DNAME=P.DNAME);
Query OK, 0 rows affected (0.09 sec)

mysql>SELECT *FROM RESULT5B;
```

```
+ -----+------+ --------+----------+---------------------------------- ----------------------------------
| FNAME    |LNAME     |DNAME     |PNAME     |
+ -----+------+ --------+----------+---------------------------------- ----------------------------------
| James    |Brog      |Headquarters  |Reorganization      |
|Jennifer|Wallance|Administration|Newbenefits     |
|Jennifer|Wallance|Administration|Computerization|
|Franklin|Wong  |Research  |ProductY  |
|Franklin|Wong  |Research  |ProductZ  |
|Franklin|Wong  |Research  |ProductX  |
+ -----+------+ --------+----------+---------------------------------- ----------------------------------
6 rowsinset(0.00sec)
 ------------------------------------------------------------------------------------------------------------------------------------------------
```

c) Demonstrateoneupdateoperationonthisview.

```
IF PARENT TABLE HAS ANY CONSTRAINTS VIEW TABLE IS NOT UPDATED;
 ------------------------------------------------------------------------------------------------------------
```

d) List the Location of the controlling departments for all the projects.

```
mysql> CREATE VIEW RESULT5D AS (SELECT P.PNAME,PV.DNAME,D.DLOC
       FROM PROJECT_VIEW PV,DLOCATION D,PROJECT P WHERE
       P.DNO=D.DNO ANDP.PNAME=PV.PNAME);
Query OK, 0 rows affected (0.09 sec)
mysql>SELECT * FROM RESULT5D;
+ -----+------+-------+-- ---------------------- ---------------
| PNAME    |DNAME     | DLOC     |
+ -----+------+-------+-- ---------------------- ---------------
|ProductX  |Research  |Bellaire  |
|ProductX  |Research  |Houstan   |
|ProductX  |Research  |Sugarland|
|ProductY  |Research  |Bellaire  |
|ProductY  |Research  |Houstan   |
|ProductY  |Research  |Sugarland|
|ProductZ  |Research  |Bellaire  |
|ProductZ  |Research  |Houstan   |
|ProductZ  |Research  |Sugarland|
|Computerization|Administration|Stafford      |
|Reorganization      |Headquarters  |Houstan   |
|Newbenefits    |Administration|Stafford  |
+ -----+------+-------+-- ---------------------- ---------------
```

e) Retrieve the data from the view.

```
mysql>SELECT*FROMPROJECT_VIEW;
+ -----+------+-------+-- ----+----------------- ------------- ------------------------
| PNAME    |DNAME     |NOOFEMP|NOOFHRS|
+ -----+------+-------+-- ----+----------------- ------------- ------------------------
|ProductX  |Research  |     3  |   62|
|ProductY  |Research  |     2  |   67|
|ProductZ  |Research  |     2  |   50|
|Computerization|Administration|    2  |   20|
|Reorganization      |Headquarters  |     2  |   25|
|Newbenefits    |Administration|     2  |   35|
+ -----+------+-------+-- ----+----------------- ------------- ------------------------
```

6) Create a view emp from employee such that it contains only emp_no emp_name and department.

```
mysql> CREATE VIEW EMP_VIEW AS (SELECT E.SSN,E.FNAME,
       E.LNAME,D.DNAME FROM EMPLOYEEE,DEPARTMENTD);
QueryOK,0rowsaffected(0.11sec)

mysql>SELECT*FROMEMP_VIEW;
+ -----+------+- ----+------+- -------------- ----------------------------------
| SSN      |FNAME     |LNAME     |DNAME     |
+ -------+---------+- ------+---------+- -------------------- ------------------------
  | 123456789 | John     | Smith    | Headquarters   |
  | 123456789 | John     | Smith    | Administration |
  | 123456789 | John     | Smith    | Research       |
  | 333445555 | Franklin | Wong     | Headquarters   |
  | 333445555 | Franklin | Wong     | Administration |
  | 333445555 | Franklin | Wong     | Research       |
  | 453453453 | Joyce    | English  | Headquarters   |
  | 453453453 | Joyce    | English  | Administration |
  | 453453453 | Joyce    | English  | Research       |
  | 666884444 | Ramesh   | Narayana | Headquarters   |
  | 666884444 | Ramesh   | Narayana | Administration |
  | 666884444 | Ramesh   | Narayana | Research       |
  | 888665555 | James    | Brog     | Headquarters   |
  | 888665555 | James    | Brog     | Administration |
  | 888665555 | James    | Brog     | Research       |
  | 987654321 | Jennifer | Wallance | Headquarters   |
  | 987654321 | Jennifer | Wallance | Administration |
  | 987654321 | Jennifer | Wallance | Research       |
  | 987987987 | Ahmad    | Jabbar   | Headquarters   |
  | 987987987 | Ahmad    | Jabbar   | Administration |
  | 987987987 | Ahmad    | Jabbar   | Research       |
  | 999887777 | Alicia   | Zelaya   | Headquarters   |
  | 999887777 | Alicia   | Zelaya   | Administration |
|999887777|Alicia|Zelaya|Research       |
+ -----+------+- ----+------+- -------------- ----------------------------------
24rowsinset(0.00sec)
 ------------------------------------------------------------------------------------------------------------
 ----------------------
```

7) Create a view dept from department with only dept_no and location.

```
mysql>CREATE VIEW DEPT_VIEW AS(SELECTDNO,DLOCFROMDLOCATION);
 QueryOK,0rowsaffected(0.09sec)

 mysql>SELECT*FROMDEPT_VIEW;
 + -----+- -----+---------
 | DNO|DLOC     |
 + -----+- -----+---------
 |1   |Houstan  |
 |4   |Stafford |
 |5   |Bellaire |
 |5   |Houstan  |
 |5   |Sugarland|
 + -----+- -----+---------
 5 rowsinset(0.00sec)
  ----------------------------------------------------------------------------------------------------
  ----------------------
```

8) Create a view that contains the details of employees who are managersonly.

```
 mysql> CREATE VIEW MANAGER AS (SELECT FNAME,LNAME FROM EMPLOYEE
        WHERESSN=SUPERSSN);
 QueryOK,0rowsaffected(0.05sec)

 mysql>SELECT *FROM MANAGER;
 + -----+---- --+--------
 | FNAME|LNAME   |
 + -----+---- --+--------
 | Ahmad|Jabbar|
 + -----+---- --+--------
 1 rowinset(0.00sec)
```

9) Write a procedure to check whether the given number is Armstrong or not.

```
mysql>delimiter//
mysql>create procedure arms(in n int)
->begin
->declaremint;
->declaresumint;
->declaretempint;
->declarelenint;
->settemp=n;
->setsum=0;
->setlen=char_length(n);
->whilen>0do
->setm=mod(n,10);
->setsum=sum+pow(m,len);
->setn=ndiv10;
->endwhile;
->selectif(sum=temp,'armstromg','notaarmstrong');
->end
-> //
```

```
mysql>delimiter ;
mysql>call arms(153);

+ -----+-----------------------------------------------------
|if(sum=temp,'armstromg','notaarmstrong')|
+ -----+-----------------------------------------------------
|armstromg     |
+ -----+-----------------------------------------------------
1 rowinset(0.00sec)
QueryOK,0rowsaffected(0.00sec)
 ----------------------------------------------------------------------------------------------------
```

10) Write a procedure which accept the account number of a customer and retrieve the balance.

```
mysql> create table customer(acc int,name varchar(20),bal
       int);QueryOK,0rowsaffected(0.49sec)

mysql>insert into customer(acc,name,bal) values
      (1,'sagar',1050),(2,'ram',150),(3,'bhim',100),
      (4,'srk',105),(5,'sir',175);
Query OK, 5 rows affected (0.07 sec)Records:5     Duplicates:0
     Warnings:0

mysql>select *from customer;
+ -----+-- -----+-------- +-----------
| acc     |name     |bal |
+ -----+-- -----+-------- +-----------
|    1|sagar|1050|
|    2|ram     |    150|
|    3|bhim    |    100|
|    4|srk     |    105|
|    5|sir     |    175|
+ -----+-- -----+-------- +-----------
5 rowsinset(0.00sec)

mysql>delimiter//
mysql>create procedure tab(in ac int)
->begin
->select bal from customer where acc=ac;
->end
->//
Query OK,0rowsaffected(0.00sec)

mysql>delimiter ;
mysql>call tab(1);
+ -----+--
| bal     |
+ -----+--
| 1050|
+ -----+--
1 rowinset(0.00sec)
```

11) Write a procedure which accepts the student number and displays the department in which he belongs to.

```
mysql>delimiter//
mysql>create procedure stud(in a int)
->begin
->select branch from student where dept_no=a;
->end
->//
QueryOK,0rowsaffected(0.00sec)

mysql>delimiter ;mysql>callstud(4);
+ -----+-----
|branch|
+ -----+-----
| AIDS    |
+ -----+-----
1 rowinset(0.00sec)
QueryOK,0rowsaffected(0.00sec)
```

12) Create a cursor to modify the salary of all employees belonging to'Research'departmentby150%.

```
mysql>delimiter//
->create procedure emp_sal_update(in dept varchar(20))
->begin
->declare flag int default0;
->declare s int default0;
->declare update_cur cursor for select SALARY from EMPLOYEE,
   DEPARTMENT where EMPLOYEE.DNO=DEPARTMENT.DNO and
   DEPARTMENT.DNAME=dept;
->declare continue handler for notfound set flag=1;
->open update_cur;
->getRec:LOOP
->fetch update_cur into s;
->if flag=1 then
->LEAVE getRec;
->endif;
->update EMPLOYEE,DEPARTMENT set SALARY=SALARY+(s*150/100) where
   EMPLOYEE.DNO=DEPARTMENT.DNO and DEPARTMENT.DNAME=dept;
->END LOOP getRec;
->close update_cur;
->end
->//
mysql>delimiter;
```

13) Consider the college database. Retrieve all students who haveregistered for a specific course and store their details into another table using Cursors.

```
mysql>create table student(sno int primary key,sname varchar(20),
      dob date,course varchar(5),year int);
QueryOK,0rowsaffected(0.00sec)

mysql>insert into student(sno,sname,dob,course,year) values
      (222,"venu","1991-09-26","cse",2020),(333,'siva','1990-04-
      10','AIDS',2021),(111,"sagar","1990-03-16","cse",2020),
      (444,'Ramu','1980-05-12','IT',2010),(555,'niteesh','1993-
      03-26', 'ECE',2017),(666,'joel','1995-05-05','AIML',2002);
Query OK, 6 rows affected (0.00 sec)Records:6     Duplicates:0
     Warnings: 0
mysql>select * from student;
+ -----+- -----+------+------+------+- ----------- --------
| sno|sname    |dob |course| year|
+ -----+- -----+------+------+------+- ----------- --------
| 111|sagar    |1990-03-16|cse            |2020|
| 222|venu     |1991-09-26|cse     |2020|
| 333|siva     |1990-04-10|AIDS    |2021|
| 444|Ramu     |1980-05-12|IT | 2010|
| 555|niteesh|1993-03-26|ECE  |2017|
| 666|joel     |1995-05-05|AIML    |2002|
+ -----+- -----+------+------+------+- ----------- --------
6 rowsinset(0.00sec)

mysql> create table temp_student(stdnoint, stdname varchar(20),
       stdcourse varchar(20));
QueryOK,0rowsaffected(0.00sec)

mysql>delimiter//
mysql>create procedure getStudents(in x varchar(10))
->begin
->declare flag int default0;
->declare stdno int;
->declare stdname varchar(20);
->declare stdcourse varchar(10);
->declare get_cur cursor for select sno,sname,course from student
  where course=x;
->declare continue handler for notfound set flag=1;
->open get_cur;
->getRec:LOOP
->fetch get_cur into stdno,stdname,stdcourse;
->if flag=1 then
->LEAVE getRec;
->endif;
->insert intot emp_student values(stdno,stdname,stdcourse);
->ENDLOOP getRec;
->close get_cur;
->end
->//
```

```
mysql>delimiter;
mysql>call getStudents("cse");
Query OK, 0 rows affected, 1 warning (0.00 sec)mysql>select*from
temp_student;
+ -----+---- --+------+- --------------
|stdno|stdname|stdcourse|
+ -----+---- --+------+- --------------
|    111|sagar |cse |
|    222|venu  | cse     |
+ -----+---- --+------+- --------------
2 rowsinset(0.00sec)
```

14) Write an update trigger on Account table. The system should keeptrackoftherecordsthatarebeingupdated.

```
mysql>delimiter//
->CREATE TRIGGER ACCUPDATE BEFORE UPDATE ON Account FOR EACHROW
->BEGIN
->DECLARE emsg varchar(250);
->SET emsg="NEW BALANCE CANNOT BE LESSTHAN OLD BALANCE";
->IF new.balance<old.balance THEN
->SIGNAL SQL STATE'45000'
->SET MESSAGE_TEXT=emsg;
->ENDIF;
->END
->//
mysql>delimiter ;
mysql>delimiter//

mysql>
```

15. Create NOSQL database for a Sample Application and Perform CURD operations.

FOR CREATING DATABASE:

How and when to create database

If there is no existing database, the following command is used to create a new database.

Syntax:

```
use  DATABASE_NAME
```

If the database already exists, it will return the existing database.

Let' take an example to demonstrate how a database is created in MongoDB. In the following example, we are going to create a database "database".

See this example

```
>use db
```

output:

Swithched to db database

To check the currently selected database, use the command db:

```
>db
```

output:
---
db

To check the database list, use the command show dbs:

```
>show dbs
```

output:
---
local 0.078GB

Here, your created database "db" is not present in the list, insert at least one document into it to display database:

```
>db.movie.insert({"name":"database"})
```

Here movie is collection, automatically created

output:

WriteResult({ "Inserted": 1})

```
>show dbs
```

output:
---
db   0.078GB
local 0.078GB

INSERT OPERATIONS:

The insert operation is one of the crucial operations in the database system. MongoDB supports the below mentioned three methods to insert document data in your database:

- insert()
- insertOne()
- insertMany()

INSERT METHOD:

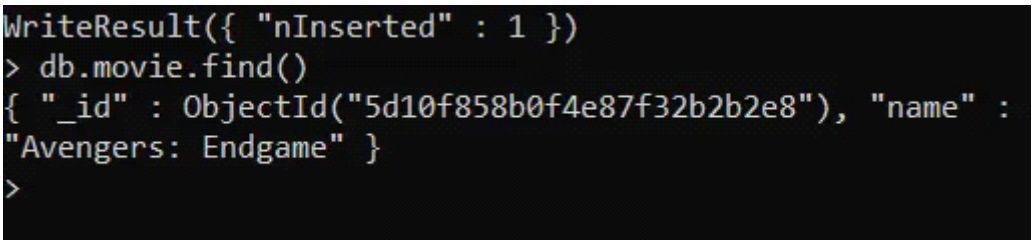
The insert() method is used to insert one or multiple documents in a collection. The collection name is associated with the insert() method and the parameters. The syntax to insert a single document is

In the above syntax, the document will consist of { name: "data_value" }. As it is a JSON document, these documents will consist of the data as name-value pairs, surrounded by curly braces, i.e. {}.

---->Syntax:

```
db.movie.insert({"name":"Avengers: Endgame"})
db.movie.find()
```

----->OUTPUT:



The **_id** which is provided by MongoDB is a 12-byte value of ObjectId which is prepared from the following values:

- a 4-byte value denoting the seconds as Unix epoch,
- a 3-byte device identifier value,
- a 2-byte processing id,
- a 3 byte counter which is a random value.

Create multiple documents using insert method():

It is also possible for you to insert multiple document values in a particular insert() method.

Let us take an **example** where you can insert multiple documents at a time:

```
db.movie.insert(
[
{ name: "Avengers: Infinity War" },
{ name: "Avengers: Endgame" }
]
)
```

It is to be noted that the documents are supplied in the form of an array. Document values are packed or enclosed in square brackets [] and separated by commas.

Executing the above statements will pop up with messages something like this:

OUTPUT:

```
> db.movie.insert([{name:"Avengers: Infinity War"}
,{name:"Avengers: Endgame"}])
BulkWriteResult({
        "writeErrors" : [ ],
        "writeConcernErrors" : [ ],
        "nInserted" : 2,
        "nUpserted" : 0,
        "nMatched" : 0,
        "nModified" : 0,
        "nRemoved" : 0,
        "upserted" : [ ]
})
>
```

EMBEDDED DOCUMENTS:

MongoDB also allow users to create document containing other documents, arrays of values, as well as arrays of documents.

EXAMPLE:

```
db.writer.insert({
writername: "Stan Lee",
  comics: [
{ comics: "DC Comics", year: 2004, name: "Superman" },
{ project: "DC Comics", year: 2001, level: "Batman" },
{ project: "Marvel Comics", year: 1968, level: "Captain America"
}
  ]
})
```

```
> db.writer.insert({
...   writername: "Stan Lee",
...   comics: [
...     { comics: "DC Comics", year: 2004, name: "Superman" },
...     { project: "DC Comics", year: 2001, level: "Batman" },
...     { project: "Marvel Comics", year: 1968, level: "Captain America" }
...   ]
... })
WriteResult({ "nInserted" : 1 })
> db.writer.find()
{ "_id" : ObjectId("5d1101a0b0f4e87f32b2b2f4"), "writername" : "Stan Lee", "comics" : [ {
 "comics" : "DC Comics", "year" : 2004, "name" : "Superman" }, { "project" : "DC Comics ,
 "year" : 2001, "level" : "Batman" }, { "project" : "Marvel Comics", "year" : 1968, "le e
l" : "Captain America" } ] }
>
```

OUTPUT:

THE INSERTONE() METHOD:

Another way to insert documents is by using the insertOne() method for a single document in a collection:

EXAMPLE:

```
db.movie.insertOne({ _id: 2, writername: "Stan Lee", name:
"Aquaman" })
```

In this case, you have a particular non-existent collection of data. In the case of the insert() method, a precise collection will get produced in case it does not exist previously.

Here you will observe that the output appeared to be different in format than that of insert() method:

```
> db.movie.insertOne({ _id: 2, writername: "Stan Lee", name: "Aquaman" })
{ "acknowledged" : true, "insertedId" : 2 }
>
```

OUTPUT:

INSERTMANY() METHOD:

As the name is explaining its working, is used for inserting multiple documents:

EXAMPLE:

```
db.developers.insertMany(
[
{ _id: 20, devname: "John Wick", tools: "Visual Studio", born:
1948 },
{ _id: 21, devname: "Ganesh Roy", tools: "Net Beans", born: 1945
},
{ _id: 22, devname: "Deeksha Raul", tools: "Unity 3D", born: 1954
}
]
)
```

```
> db.developers.insertMany(
... [
...   { _id: 20, devname: "John Wick", tools: "Visual Studio", born: 1948 },
...   { _id: 21, devname: "Ganesh Roy", tools: "Net Beans", born: 1945 },
...   { _id: 22, devname: "Deeksha Raul", tools: "Unity 3D", born: 1954 }
... ]
... )
{ "acknowledged" : true, "insertedIds" : [ 20, 21, 22 ] }
>
```

OUTPUT:

UPDATE OPERATION:

Use of update operation :

The update operation in a database is used to change or update values in a document. MongoDB makes use of the update() method for updating the documents within a MongoDB collection. For updating any specific documents, a new criterion can be incorporated with the update statement, which will only update the selected documents.

You have to put some specific condition in the form of the parameter to update the document in MongoDB. Here is a stepwise representation of how this can be performed:

- Make use of the update() method.
- Prefer the circumstance that you wish to implement for deciding which document needs an update in their database. Let us assume an example where you want to update your document which is having an id 4.
- Then make use of the set command for modifying the Field Name.
- Select which Field Name you wish for modifying and go into the new value consequently.

Syntax:

```
db.collection.update(

<query>,

<update>,

   {

upsert: <boolean>,

     multi: <boolean>,

writeConcern: <document>,

     collation: <document>,

arrayFilters: [ <filterdocument1>, ... ]

   }

)
```

Example :

```
     db.musicians.find({ _id: 4 }).pretty()
```

Output:

```
> db.musicians.find({ _id: 4 }).pretty()
{
        "_id" : 4,
        "name" : "Steven Morse",
        "instrument" : "Violin",
        "born" : 1954
}
>
```

So, now let us update the list of instrument played by this person, by making use of the $set operator for updating a single field.

Example:

```
db.musicians.update(

{_id: 4},

{

  $set: { instrument: ["Vocals", "Violin", "Octapad"] }

}

)
```

Output:

```
> db.musicians.update(
...
... {_id: 4},
... {
...   $set: { instrument: ["Vocals", "Violin", "Octapad"] }
... }
... )
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 0 })
> db.musicians.find({ _id: 4 }).pretty()
{
        "_id" : 4,
        "name" : "Steven Morse",
        "instrument" : [
                "Vocals",
                "Violin",
                "Octapad"
        ],
        "born" : 1954
}
>
```

Characteristics of update of MongoDB:

- In case your field does not subsist in the current document, the $set operator will insert a new field with the specified value, until and unless it violates the type constraint.
- MongoDB users can also make use of { multi: true } for updating multiple documents which will meet the query criteria.
- Making use of { upsert: true } for creating a new document is also possible as no document goes with the query.

Save() Method:

The save() method is used to replace a document with another document conceded in the form of a parameter.

In other words, it can be said that the save() is a blend of both update() as well as insert(). As the save() method is used, the document that exists will get updated. Otherwise, when it does not exist, it will create one. When an _id field is not specified, MongoDB automatically creates a document with this _id containing an ObjectId value (as conducted by the insert() method).

Example:

```
db.musicians.save({

  "_id": 4,

  "name": "Steven Morse",

  "instrument": "Violin",

  "born": 1954

})
```

Output:

```
> db.musicians.save({
...    "_id": 4,
...    "name": "Steven Morse",
...    "instrument": "Violin",
...    "born": 1954
... })
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 0 })
> db.musicians.find().pretty()
{
        "_id" : 4,
        "name" : "Steven Morse",
        "instrument" : "Violin",
        "born" : 1954
}
>
```

DELETING DOCUMENTS IN MONGODB:

MongoDB allows you to delete a document or documents collectively using its one of the three methods. Three methods provided by MongoDB for deleting documents are:

- db.collection.deleteOne()
- db.collection.remove()
- db.collection.deleteMany()

db.collection.deleteOne() Method:

This method is used to delete only a single document, even when more than one document matches with the criteria. Here is an example of using this db.collection.deleteOne() method for deleting the single document. To perform this process here, we have created a database and saved all the data separately.

Example:

```
db.programmers.find()
```

Output:

```
> db.programmers.find( )
{ "_id" : ObjectId("5d13002d3c59ac532372df63"), "name" : "James Gosling" }
{ "_id" : ObjectId("5d13002d3c59ac532372df64"), "name" : "Dennis Ritchie" }
{ "_id" : ObjectId("5d13002d3c59ac532372df65"), "name" : "Bjarne Stroustrup" }
>
```

Example:

```
db.programmers.deleteOne( { name: { $in: [ "Dennis Ritchie", "Bjarne Stroustrup"] } } )
```

Executing this statement, you will notice that, although two documents match the criteria, only one document gets deleted.

OUTPUT:

```
> db.programmers.deleteOne( { name: { $in: [ "Dennis Ritchie", "Bjarne Stroustrup"] } } )
{ "acknowledged" : true, "deletedCount" : 1 }
> db.programmers.find()
{ "_id" : ObjectId("5d13002d3c59ac532372df63"), "name" : "James Gosling" }
{ "_id" : ObjectId("5d13002d3c59ac532372df65"), "name" : "Bjarne Stroustrup" }
>
```

Delete all documents:

To delete all documents from the programmers' collection, you have to write the query like this:

Example:

```
db.programmers.remove( {} )
```

Output:

```
> db.programmers.remove( {} )
WriteResult({ "nRemoved" : 2 })
>
```

But, when the filtering is done for removing elements, the db.collection.remove() method will document which matches the specified criteria. Here, we delete all documents where the artist name is "James Gosling".

Example:

```
db.programmers.remove( { name: "James Gosling" } )
```

Output:

```
> db.programmers.remove( { name: "James Gosling" } )
WriteResult({ "nRemoved" : 1 })
>
```

db.collection.deleteMany() Method:

MongoDB allows you to delete multiple documents using the db.collection.deleteMany() method. This method deletes all your documents whichever match its criteria mentioned in the parameter. To check its implementation of db.collection.deleteMany() method, you can use the method the same way as done previously:

Example:

```
db.programmers.deleteMany( { name: { $in: [ "Dennis Ritchie", "Bjarne Stroustrup" ] } } )
```

Output:

```
> db.programmers.deleteMany( { name: { $in: [ "Dennis Ritchie", "Bjarne Stroustrup" ] } } )
{ "acknowledged" : true, "deletedCount" : 2 }
>
```

In this way, continuing from the previous example implementation, now all the records were also deleted, and the deleteMany() method was used this time. Now, if you try to find the documents using find() method, it won't show any result of your query or search.

QUERY DOCUMENTS IN MONGODB:

- **The find() method:** This method is used for querying data from a MongoDB collection.
  The basic syntax for using this method is:

Syntax:

```
db.collection_name.find()
```

Example:

```
db.writers.find()
```

Various other options can be used to make the query specific. It will be discussed below.

- **The pretty() method:** This method is used for giving a proper format to the output extracted by the query.
  The basic syntax for using this method is:

Syntax:

```
db.collection_name.find().pretty()
```

Example:

```
db.writers.find().pretty()
```

FILTERING CRITERIA OF MONGODB: It is also possible to filter your results by giving or adding some specific criteria in which you are interested to. For example, if you wish to see the Gaurav Mandes data, you can add a specific attribute to the find() to fetch the data of Gaurav Mandes from that particular database.

Example:

```
db.writers.find( { author: "Gaurav Mandes" } )
```

Output:

```
> db.writers.find( { author: "Gaurav Mandes" } )
{ "_id" : ObjectId("5d16f6a1e198c897a4105d0d"), "title" : "Networking", "descriptio
n" : "Networking Essentials", "author" : "Gaurav Mandes" }
{ "_id" : ObjectId("5d16f6a1e198c897a4105d0e"), "title" : "Game Engineering", "desc
ription" : "Game Engineering and Development", "author" : "Gaurav Mandes" }
>
```

MongoDB query which Specify "AND" Condition:

MongoDB also allows you in specifying data values of the documents holding two or more specified values to be fetched from the query. Here are two examples showing the use of specifying queries using AND.

Example:

```
db.writers.find( { tools: "Visual Studio", born: 1948} )
```

MongoDB Query Which Specify "OR" Condition:

MongoDB allows users to specify either one or multiple values to be true. According to this, till one of the conditions is true, the document data will get returned. Here is an example showing the use of OR condition:

Example:

```
db.musicians.find({$or: [ { instrument: "Drums" }, { born: 1945 } ] } )
```

Output:

```
> db.musicians.find({$or: [ { instrument: "Drums" }, { born: 1945 } ]})
{ "_id" : 2, "name" : "Ian Paice", "instrument" : "Drums", "born" : 1948 }
{ "_id" : 3, "name" : "Roger Glover", "instrument" : "Bass", "born" : 1945 }
{ "_id" : 7, "name" : "Jeff Burrows", "instrument" : "Drums", "born" : 1968 }
>
```

$in operator

The $in operator is another special operator used in queries for providing a list of values in the query. When your document holds any of those provided values, it gets returned. Here is an example:

Example:

```
db.musicians.find( { "instrument": { $in: [ "Keyboards", "Bass" ] } } )
```

Output:

```
> db.musicians.find( { "instrument": { $in: [ "Keyboards", "Bass" ] } } )
{ "_id" : 3, "name" : "Roger Glover", "instrument" : "Bass", "born" : 1945 }
{ "_id" : 5, "name" : "Don Airey", "instrument" : "Keyboards", "born" : 1948 }
>
```