Expt No 1

ER diagram and Relational Schema

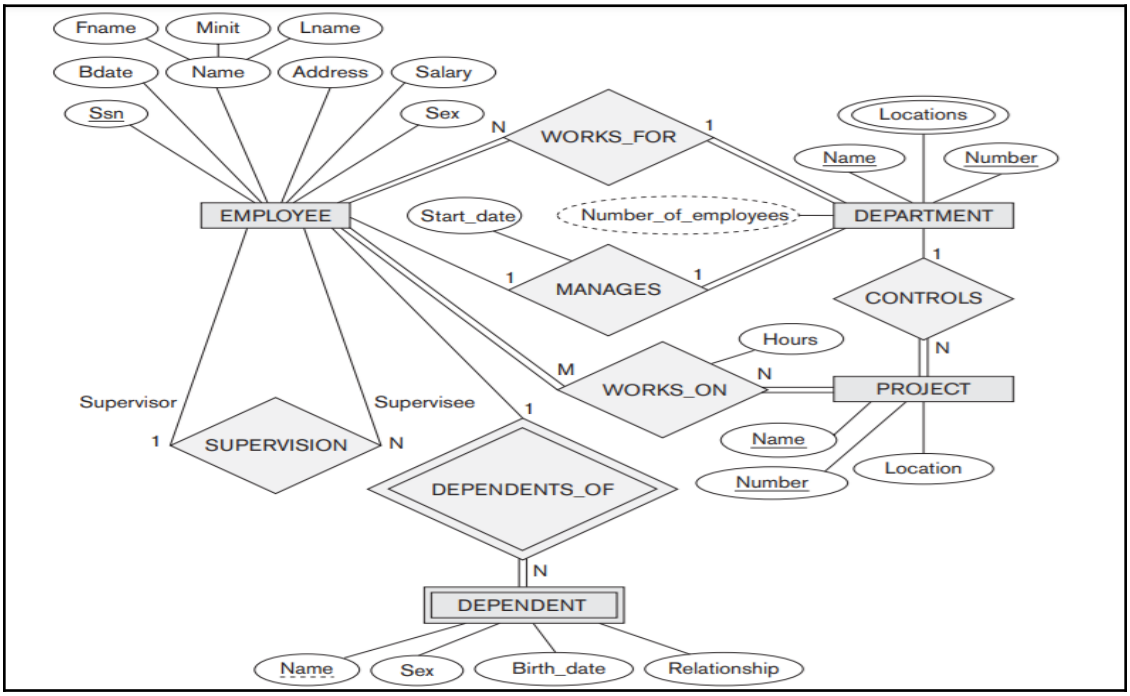
AIM:Design a database schema for an application with ER diagram from a problem description

Problem Description:

The COMPANY database keeps track of a company's employees, departments, and projects. Suppose that after the requirements collection and analysis phase, the database designers provide the following description of the miniworld—the part of the company that will be represented in the database.

- The company is organized into departments. Each department has a unique name, a unique number, and a particular employee who manages the department. We keep track of the start date when that employee began managing the department. A department may have several locations.
- A department controls a number of projects, each of which has a unique name, a unique number, and a single location.
- We store each employee's name, Social Security number,address, salary, sex (gender), and birth date. An employee is assigned to one department, but may work on several projects, which are not necessarily controlled by the same department. We keep track of the current number of hours per week that an employee works on each project. We also keep track of the direct supervisor of each employee (who is another employee).
- We want to keep track of the dependents of each employee for insurance purposes. We keep each dependent's first name, sex, birth date, and relationship to the employee.

ER Diagram:



Entities:

- An entity type DEPARTMENT with attributes Name, Number, Locations, Manager, and Manager_start_date. Locations is the only multivalued attribute. We can specify that both Name and Number are (separate) key attributes because each was specified to be unique.
- An entity type PROJECT with attributes Name, Number, Location, and Controlling_department. Both Name and Number are (separate) key attributes.
- An entity type EMPLOYEE with attributes Name, Ssn, Sex, Address, Salary, Birth_date, Department, and Supervisor. Both Name and Address may be composite attributes; however, this was not specified in the requirements.
- An entity type DEPENDENT with attributes Employee, Dependent_name, Sex, Birth_date, and Relationship (to the employee).

Relationship types:

- MANAGES, a 1:1 relationship type between EMPLOYEE and DEPARTMENT. EMPLOYEE participation is partial. DEPARTMENT participation is not clear from the requirements. We assume that a department must have a manager at all times, which implies total participation.The attribute Start_date is assigned to this relationship type.
- WORKS_FOR, a 1:N relationship type between DEPARTMENT and EMPLOYEE. Both participations are total.
- CONTROLS, a 1:N relationship type between DEPARTMENT and PROJECT. The participation of PROJECT is total, whereas that of DEPARTMENT is determined to be partial, assuming that some departments may control no projects.
- SUPERVISION, a 1:N relationship type between EMPLOYEE (in the supervisor role) and EMPLOYEE (in the supervisee role). Both participations are determined to be partial, assuming that not every employee is a supervisor and not every employee has a supervisor.
- WORKS_ON, determined to be an M:N relationship type with attribute Hours, after the users indicate that a project can have several employees working on it. Both participations are determined to be total. DEPENDENTS_OF, a 1:N relationship type between EMPLOYEE and DEPENDENT, which is also the identifying relationship for the weak entity type DEPENDENT. The participation of EMPLOYEE is partial, whereas that of DEPENDENT is total

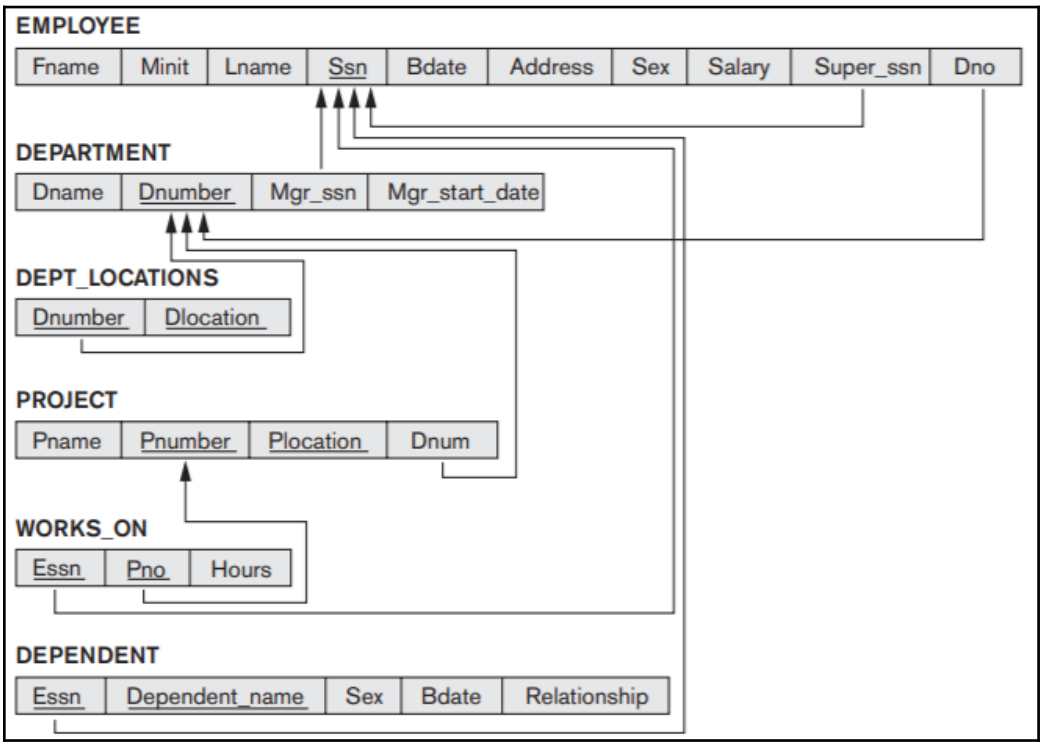
Summary :

Figure above displays the COMPANY ER database schema as an ER diagram. We now review the full ER diagram notation.

- Entity types such as EMPLOYEE, DEPARTMENT, and PROJECT are shown in rectangular boxes. Relationship types such as WORKS_FOR, MANAGES, CONTROLS, and WORKS_ON are shown in diamond-shaped boxes attached to the participating entity types with straight lines.
- Attributes are shown in ovals, and each attribute is attached by a straight line to its entity type or relationship type.
- Component attributes of a composite attribute are attached to the oval representing the composite attribute, as illustrated by the Name attribute of EMPLOYEE.
- Multivalued attributes are shown in double ovals, as illustrated by the Locations attribute of DEPARTMENT.
- Key attributes have their names underlined.

- Derived attributes are shown in dotted ovals, as illustrated by the Number_of_employees attribute of DEPARTMENT.
- Weak entity types are distinguished by being placed in double rectangles and by having their identifying relationship placed in double diamonds, as illustrated by the DEPENDENT entity type and the DEPENDENTS_OF identifying relationship type. The partial key of the weak entity type is underlined with a dotted line.
- The cardinality ratio of each binary relationship type is specified by attaching a 1, M, or N on each participating edge. The cardinality ratio of DEPARTMENT:EMPLOYEE in MANAGES is 1:1, whereas it is 1:N for DEPARTMENT: EMPLOYEE in WORKS_FOR, and M:N for WORKS_ON.
- The participation constraint is specified by a single line for partial participation and by double lines for total participation (existence dependency).
- In Figure we show the role names for the SUPERVISION relationship type because the same EMPLOYEE entity type plays two distinct roles in that relationship. Notice that the cardinality ratio is 1:N from supervisor to supervisee because each employee in the role of supervisee has at most one direct supervisor, whereas an employee in the role of supervisor can supervise zero or more employees

Relational Schema:



ER-to-Relational Schema Mapping:

Step 1: Mapping of Regular Entity Types

We create the relations EMPLOYEE, DEPARTMENT, and PROJECT in Figure to correspond to the regular entity types EMPLOYEE, DEPARTMENT, and PROJECT. We choose Ssn, Dnumber, and Pnumber as primary keys for the relations EMPLOYEE, DEPARTMENT, and PROJECT, respectively

Step 2: Mapping of Weak Entity Types

We create the relation DEPENDENT in this step to correspond to the weak entity type DEPENDENT. We include the primary key Ssn of the EMPLOYEE relation—which corresponds to the owner entity type—as a foreign key attribute of DEPENDENT; We rename it Essn. The primary key of the DEPENDENT relation is the combination {Essn, Dependent_name}, because Dependent_name is the partial key of DEPENDENT

Step 3: Mapping of Binary 1:1 Relationship Types.

We map the 1:1 relationship type MANAGES from Figure by choosing the participating entity type DEPARTMENT to serve in the role of S because its participation in the MANAGES relationship type is total (every department has a manager). We include the primary key of the EMPLOYEE relation as foreign key in the DEPARTMENT relation and rename it Mgr_ssn. We also include the simple attribute Start_date of the MANAGES relationship type in the DEPARTMENT relation and rename it Mgr_start_date

Step 4: Mapping of Binary 1:N Relationship Types

We now map the 1:N relationship types WORKS_FOR, CONTROLS, and SUPERVISION from Figure. For WORKS_FOR we include the primary key Dnumber of the DEPARTMENT relation as foreign key in the EMPLOYEE relation and call it Dno. For SUPERVISION we include the primary key of the EMPLOYEE relation as foreign key in the EMPLOYEE relation itself—because the relationship is recursive—and call it Super_ssn. The CONTROLS relationship is mapped to the foreign key attribute Dnum of PROJECT, which references the primary key Dnumber of the DEPARTMENT relation.

Step 5: Mapping of Binary M:N Relationship Types

We map the M:N relationship type WORKS_ON from the ER diagram by creating the relation WORKS_ON in relational schema. We include the primary keys of the PROJECT and EMPLOYEE relations as foreign keys in WORKS_ON and rename them Pno and Essn, respectively. We also include an attribute Hours in WORKS_ON to represent the Hours attribute of the relationship type. The primary key of the WORKS_ON relation is the combination of the foreign key attributes {Essn, Pno}.

Step 6: Mapping of Multivalued Attributes

We create a relation DEPT_LOCATIONS. The attribute Dlocation represents the multivalued attribute LOCATIONS of DEPARTMENT, while Dnumber—as foreign key—represents the primary key of the DEPARTMENT relation. The primary key of DEPT_LOCATIONS is the combination of {Dnumber, Dlocation}. A separate tuple will exist in DEPT_LOCATIONS for each location that a department has.

RESULT:

We successfully created an ER diagram from the problem description given. We were also able to extract the relational schema from the ER diagram.

Expt No 2

APPLICATION OF DDL COMMANDS USING UI AND SQL

AIM : Creation, modification, configuration, and deletion of databases using UI and SQL Commands.

QUERY

Create database students and execute various commands on it.

create database students;
show databases;
use students;

Create a table students with the fields student id, name, email and phone number.

create table Student(stud_id INT AUTO_INCREMENT PRIMARY KEY,stud_fname VARCHAR(20),stud_lname VARCHAR(20),stud_email VARCHAR(20),stud_ph VARCHAR(10));

Create a table subject to store the list of subjects.

create table Subject(sub_id INT AUTO_INCREMENT PRIMARY KEY,sub_name VARCHAR(20));

Create a table marks to store marks of students for various subjects.

create table Marks(sub_id INT,stud_id INT,marks INT, PRIMARY KEY(sub_id,stud_id));
show tables;

Display the details of the tables created.

desc Student;
desc Subject;
desc Marks;

Alter the tables to include foreign keys.

alter table Marks ADD FOREIGN KEY (stud_id) REFERENCES Student(stud_id);
alter table Marks ADD FOREIGN KEY (sub_id) REFERENCES Subject(sub_id);

Alter the tables to initialize auto increment value.

alter table Student AUTO_INCREMENT=100;
alter table Subject AUTO_INCREMENT=200;

Alter the tables to add new column.

alter table Student add address char(50);
describe Student;

Alter the tables to modify datatype of a column.

alter table Student modify column address varchar(50);
describe Student;

Alter the tables to rename column.

alter table Student rename column address to stud_address;
describe Student;

Alter the tables to delete column.

alter table Student drop column stud_address;
describe Student;

Truncate command

To test truncate command, we have to populate table.

insert into Student (stud_fname, stud_lname, stud_email, stud_ph) values ('Alice', 'Tom', ' alice@alice.com', '9889975555');

insert into Student (stud_fname, stud_lname, stud_email, stud_ph) values ('Bob', 'John', ' bob@bob.com', '9889975556');

*select * from Student;*

Test Truncate command
SET FOREIGN_KEY_CHECKS=0; # Used to disable referential integrity
truncate table Student;
*select * from Student; # Observe entire data is cleared from table.*

Drop the tables created.

drop table Student;
drop table Subject;
drop table Marks;
drop database students;

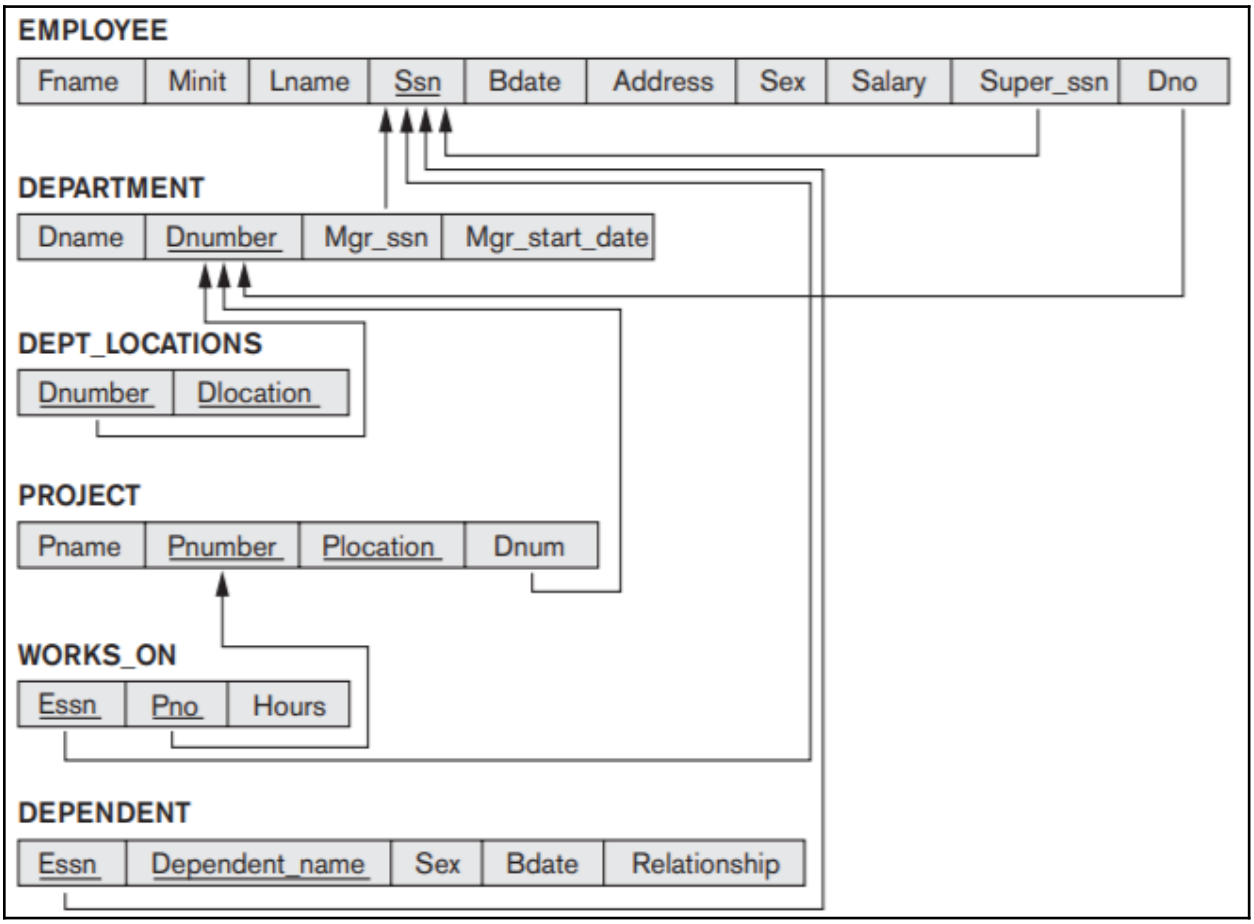
RESULT: Successfully executed the queries using MySQL Workbench.

Expt No 3

CREATION OF DATABASE SCHEMA AND EXTRACTION OF ER DIAGRAM

AIM: Creation of database schema - DDL (create tables, set constraints, enforce relationships, create indices, delete and modify tables). Export ER diagram from the database and verify relationships (with the ER diagram designed in step 1).

Create a database schema for the below diagram.



QUERY:

```
CREATE TABLE EMPLOYEE
( Fname          VARCHAR(10)   NOT NULL,
  Minit          CHAR,
  Lname          VARCHAR(20)      NOT NULL,
  Ssn            CHAR(9)          NOT NULL,
  Bdate          DATE,
```

```
  Address        VARCHAR(30),
  Sex            CHAR(1),
  Salary         DECIMAL(5),
  Super_ssn      CHAR(9),
  Dno            INT             NOT NULL,
PRIMARY KEY   (Ssn));

CREATE TABLE DEPARTMENT
( Dname          VARCHAR(15)      NOT NULL,
  Dnumber        INT              NOT NULL,
  Mgr_ssn        CHAR(9)          NOT NULL,
  Mgr_start_date  DATE,
PRIMARY KEY (Dnumber),
FOREIGN KEY (Mgr_ssn) REFERENCES EMPLOYEE(Ssn) );

CREATE TABLE DEPT_LOCATIONS
( Dnumber        INT              NOT NULL,
  Dlocation      VARCHAR(15)      NOT NULL,
PRIMARY KEY (Dnumber, Dlocation),
FOREIGN KEY (Dnumber) REFERENCES DEPARTMENT(Dnumber) );

CREATE TABLE PROJECT
( Pname          VARCHAR(15)      NOT NULL,
  Pnumber        INT              NOT NULL,
  Plocation      VARCHAR(15),
  Dnum           INT              NOT NULL,
PRIMARY KEY (Pnumber),
FOREIGN KEY (Dnum) REFERENCES DEPARTMENT(Dnumber) );

CREATE TABLE WORKS_ON
( Essn           CHAR(9)          NOT NULL,
  Pno            INT              NOT NULL,
  Hours          DECIMAL(3,1)     NOT NULL,
PRIMARY KEY (Essn, Pno),
FOREIGN KEY (Essn) REFERENCES EMPLOYEE(Ssn),
FOREIGN KEY (Pno) REFERENCES PROJECT(Pnumber) );

CREATE TABLE DEPENDENT
( Essn           CHAR(9)          NOT NULL,
  Dependent_name  VARCHAR(15)      NOT NULL,
  Sex            CHAR,
  Bdate          DATE,
  Relationship    VARCHAR(8),
PRIMARY KEY (Essn, Dependent_name),
FOREIGN KEY (Essn) REFERENCES EMPLOYEE(Ssn) );

ALTER TABLE DEPARTMENT
 ADD CONSTRAINT Dep_emp FOREIGN KEY (Mgr_ssn) REFERENCES
EMPLOYEE(Ssn);

ALTER TABLE EMPLOYEE
```

```
 ADD CONSTRAINT Emp_emp FOREIGN KEY (Super_ssn) REFERENCES
EMPLOYEE(Ssn);

ALTER TABLE EMPLOYEE
 ADD CONSTRAINT Emp_dno FOREIGN KEY  (Dno) REFERENCES
DEPARTMENT(Dnumber);

ALTER TABLE EMPLOYEE
 ADD CONSTRAINT Emp_super FOREIGN KEY  (Super_ssn) REFERENCES
EMPLOYEE(Ssn);
```

EER diagram: <SCREENSHOT>

Export ER Diagram from Workbench

RESULT:

We successfully created the Database Schema from the problem description given. We were also able to export the ER diagram from the database and relationships are verified.

Expt No 4

APPLICATION OF DML COMMANDS USING SQL

AIM :Insertion, updation, deletion, and selection of databases using SQL Commands

1. Insert data into the Employee schema created in Expt No 3.

```
SET FOREIGN_KEY_CHECKS=0;

INSERT INTO EMPLOYEE
VALUES      ('John','B','Smith',123456789,'1965-01-09','731 Fondren, Houston
TX','M',30000,333445555,5),
        ('Franklin','T','Wong',333445555,'1965-12-08','638 Voss, Houston
TX','M',40000,888665555,5),
        ('Alicia','J','Zelaya',999887777,'1968-01-19','3321 Castle, Spring
TX','F',25000,987654321,4),
        ('Jennifer','S','Wallace',987654321,'1941-06-20','291 Berry, Bellaire
TX','F',43000,888665555,4),
        ('Ramesh','K','Narayan',666884444,'1962-09-15','975 Fire Oak, Humble
TX','M',38000,333445555,5),
        ('Joyce','A','English',453453453,'1972-07-31','5631 Rice, Houston
TX','F',25000,333445555,5),
        ('Ahmad','V','Jabbar',987987987,'1969-03-29','980 Dallas, Houston
TX','M',25000,987654321,4),
```

```
        ('James','E','Borg',888665555,'1937-11-10','450 Stone, Houston
TX','M',55000,null,1);

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

INSERT INTO DEPT_LOCATIONS (Dnumber, Dlocation) VALUES (1,'Houston');
INSERT INTO DEPT_LOCATIONS (Dnumber, Dlocation) VALUES (4,'Stafford');
INSERT INTO DEPT_LOCATIONS (Dnumber, Dlocation) VALUES (5,'Bellaire');
INSERT INTO DEPT_LOCATIONS (Dnumber, Dlocation) VALUES (5,'Sugarland');
INSERT INTO DEPT_LOCATIONS (Dnumber, Dlocation) VALUES (5,'Houston');

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

INSERT INTO WORKS_ON
VALUES    (123456789,1,32.5),
       (123456789,2,7.5),
       (666884444,3,40.0),
       (453453453,1,20.0),
       (453453453,2,20.0),
       (333445555,2,10.0),
       (333445555,3,10.0),
       (333445555,10,10.0),
       (333445555,20,10.0),
       (999887777,30,30.0),
       (999887777,10,10.0),
       (987987987,10,35.0),
       (987987987,30,5.0),
       (987654321,30,20.0),
       (987654321,20,15.0),
       (888665555,20,16.0);

INSERT INTO DEPENDENT
VALUES     (333445555,'Alice','F','1986-04-04','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,'Alice','F','1988-12-30','Daughter'),
        (123456789,'Elizabeth','F','1967-05-05','Spouse');
```

2. UPDATE QUERY

- **Update Salary of all employee by 1000 $**

```
update EMPLOYEE Set Salary = Salary+1000;
```

- **Update Address of Ssn 666884444 to "100 Centre, Stafford TX 77477"**

```
update EMPLOYEE set Address = '100 Centre, Stafford TX 77477' where Ssn =
'666884444';
```

3. SELECT QUERIES:

- **Write a query to get the details of a Employee whose Ssn = 666884444.**

```
select *
from EMPLOYEE
where Ssn = '666884444';
```

- **Write a query to get the Address of Employee Ramesh Narayan**

```
select Address
from EMPLOYEE
where Fname = 'Ramesh' and Lname = 'Narayan';
```

- **Write a query to get the list of employees working in Department No = 5**

```
select Fname, Lname
from EMPLOYEE
where Dno = 5;
```

- **Write a query to get the list of Employees working in Research Department.**

```
select Fname, Lname
from EMPLOYEE, DEPARTMENT
where Dno = Dnumber and Dname = 'Research';
```

- **Write a query to get the Manager's Ssn of "Research" department.**

```
select Mgr_ssn
from DEPARTMENT
where Dname = 'Research';
```

- **Write a query to get the Manager's Name of "Research" department.**

select Fname, Lname
from EMPLOYEE, DEPARTMENT
where Dno = Dnumber and Dname = 'Research';

3. DELETE QUERIES:

- **Delete the details of Research department from DEPARTMENT tables**

  delete from DEPARTMENT where Dname = 'Research';

- **Delete the contents of DEPARTMENT Table**

  delete from DEPARTMENT;

4. VIEW

- **Create a view Emp(Ssn , Fname, Lname, Sex, Salary,Dno) from EMPLOYEE Table**

  CREATE VIEW Emp AS SELECT Ssn,Fname, Lname, Sex, Salary, Dno FROM EMPLOYEE;

- **Display the contents of View**

  *select * from Emp;*

RESULT: Successfully executed the queries using SQL DML Commands

Expt No 5

IMPLEMENTATION OF BUILT IN FUNCTIONS

AIM: Implementation of built in functions in RDBMS

A. Create a table store. Fields are order no, code, item, quantity, price, discount, mrp

QUERY

Create table store (order_no int primary key, code int, item char(15), quantity varchar(8), price int,
discount varchar(7), mrp int);
Insert into store values(‘1’, ‘1’, ‘soap’, ‘5’, ‘75’, ‘2%’, ‘72’,);
1 row created;
Insert into store values(‘2’, ‘2’, ‘chilly powder’, ‘2’, ‘24’, ‘3%’, ‘20’,);
1 row created;
Insert into store values(‘3’, ‘3’, ‘atta’, ‘2’, ‘70’, ‘3%’, ‘78’,);