DBMS Lab #3: COMPANY DATABASE

A. Schema Diagram

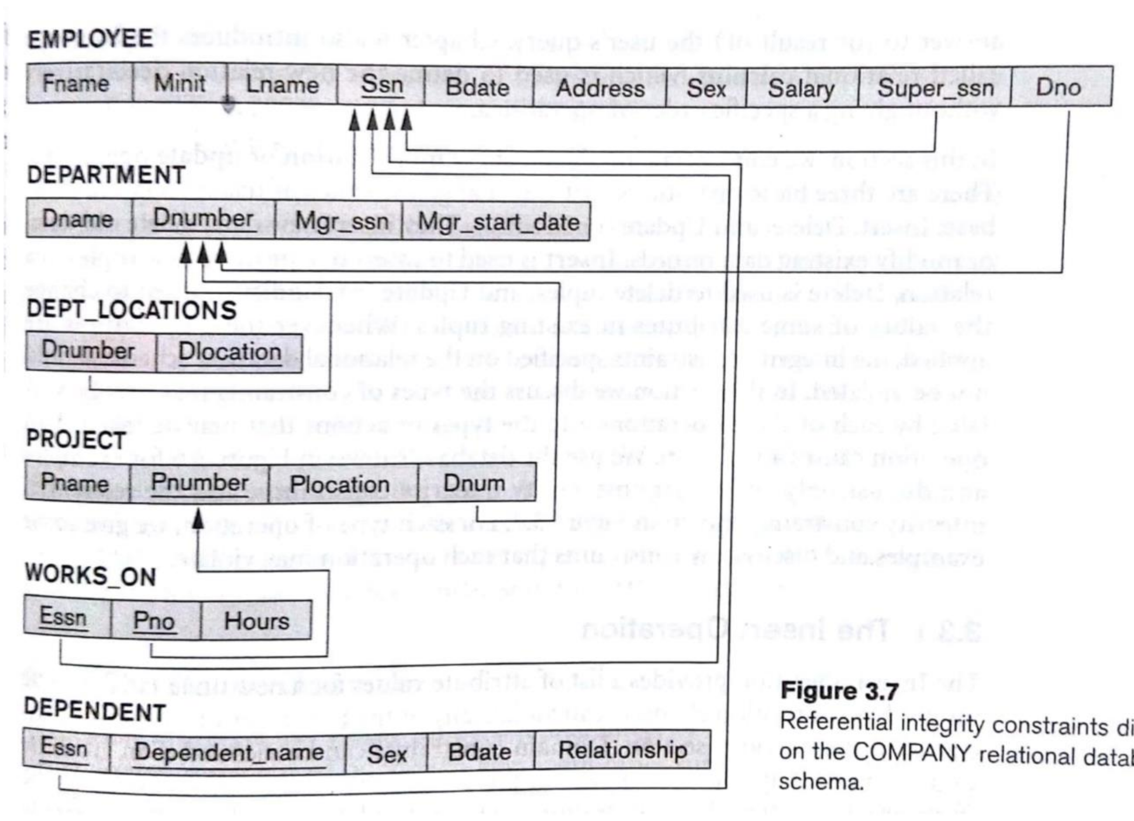


Figure 3.7
Referential integrity constraints displayed on the COMPANY relational database schema.

B. Create the tables with the following data types specified:

Define type Employee

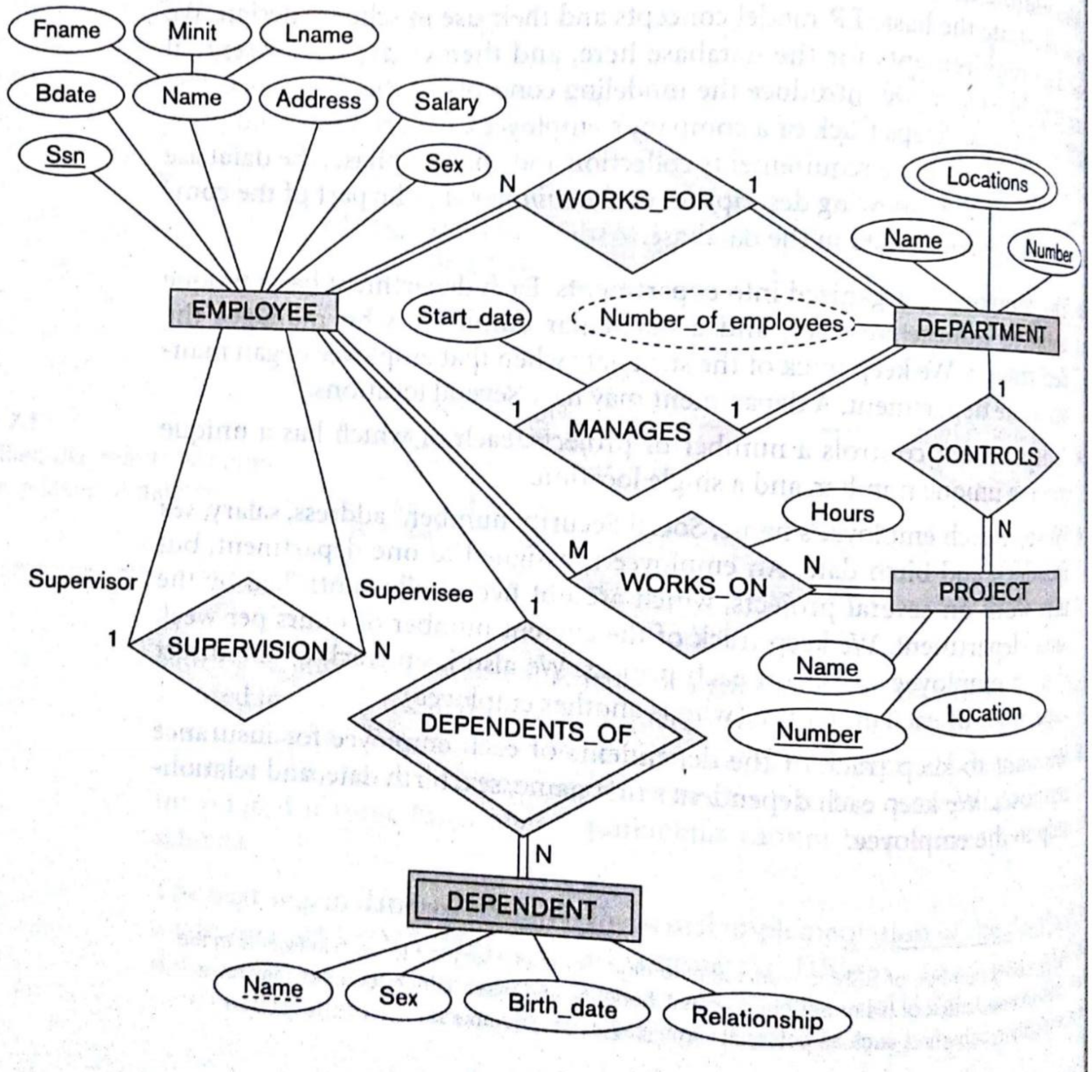
Tuple(

Fname: Varchar(10)
Minit: Varchar(10)
Lname: Varchar(10)
SSn: Integer(10)
Bdate:Date
Address:Varchar(30)
Sex:char(1)
Salary:dec(7,2)
Super_ssn:Integer(10)
Dno:Integer(3)
)

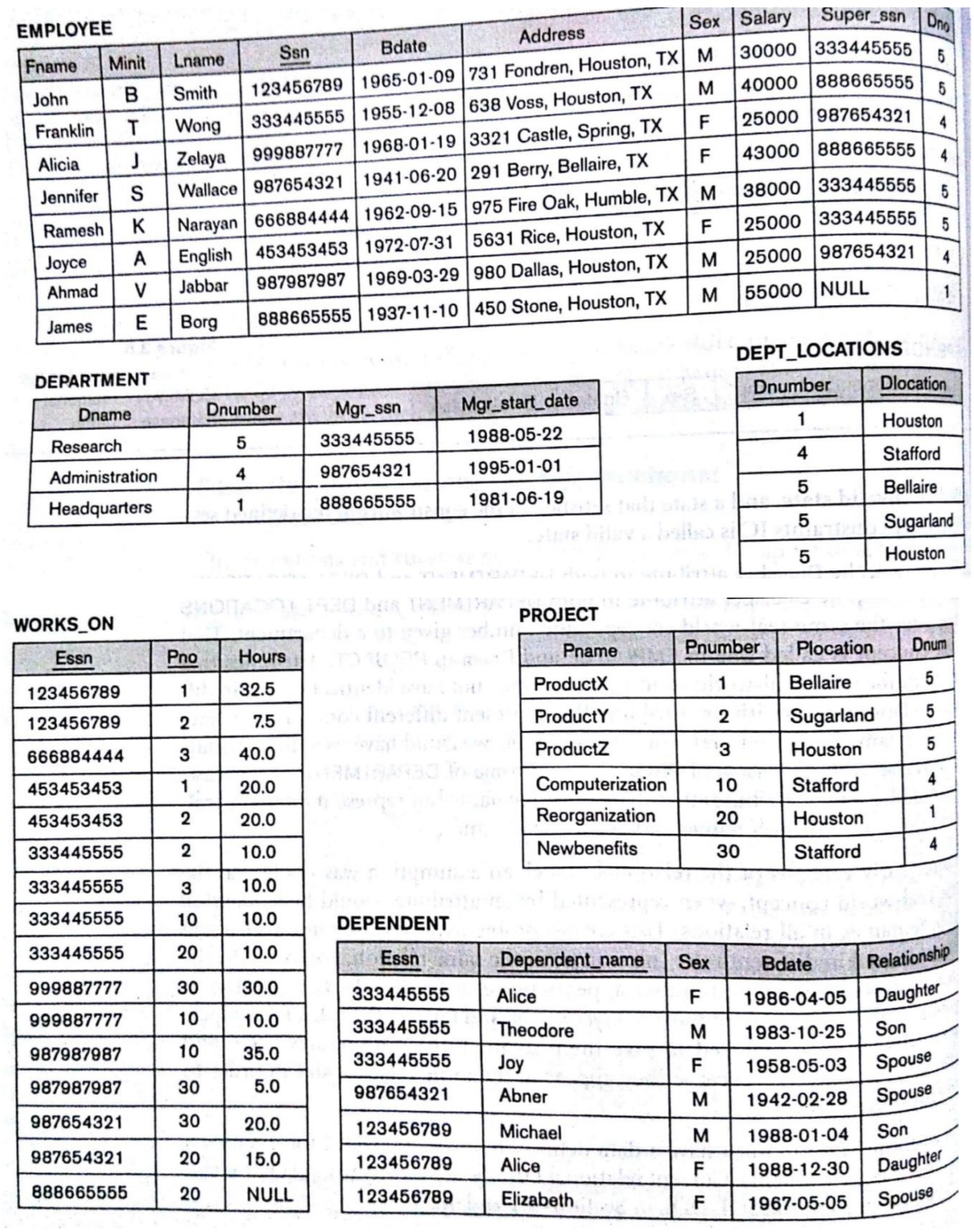
Define type Department
Tuple(
Dname:varchar(20)
Dnumber:Integer(3)
Mgr_ssn:Integer(10)
Mgr_start_date:date
)
Define type works_on
Tuple(
Essn:Integer(10)
Pno:Integer(3)
Hours:Dec(3,1)
)
Define type Dept_Locations
Tuple(
Dnumber:Integer(3)
Dlocation:Varchar(20)
)
Define type Project
Tule(
Pname:Varchar(20)
Pnumber:Integer(3)
Plocation:varchar(20)
Dnum:Integer(3)
)
Define type Dependent
Tuple(
Essn:Inetegr(10)
Dependent_name:Varchar(20)
Sex:char(1)
Bdate:date
Relationships:Varchar(10)
)

C. ER Diagram

Figure 7.2
An ER schema diagram for the COMPANY database. The diagrammatic notation is introduced gradually throughout this chapter and is summarized in Figure 7.14.



Enter Data for the Company relations:



EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	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

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
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	NULL

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
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	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

D. Solve the below Queries:

1. For each employee, retrieve the employee's name, and the name of his or her immediate supervisor.
   SELECT E.FNAME, E.LNAME, S.FNAME, S.LNAME FROM EMPLOYEE E S WHERE E.SUPERSSN=S.SSN
2. Show the effect of giving all employees who work on the 'ProductX' project a 10% raise.

   SELECT FNAME, LNAME, 1.1*SALARY

   FROM EMPLOYEE, WORKS_ON, PROJECT

   WHERE SSN=ESSN

   AND PNO=PNUMBER AND PNAME='ProductX'

3. For each project on which more than two employees work, retrieve the project number, project name, and the number of employees who work on that project.

   SELECT PNUMBER, PNAME, COUNT (*) FROM PROJECT, WORKS_ON WHERE PNUMBER=PNO
   GROUP BY PNUMBER, PNAME
   HAVING COUNT (*) > 2
4. Retrieve the names of all employees who have two or more dependents.
   SELECT LNAME, FNAME FROM EMPLOYEE
   WHERE (SELECT COUNT (*) FROM DEPENDENT
   WHERE SSN=ESSN) ≥ 2);
5. Make a list of all project numbers for projects that involve an employee whose last name is 'Smith' as a worker or as a manager of the department that controls the project.
   (SELECT PNAME FROM PROJECT, DEPARTMENT, EMPLOYEE WHERE DNUM=DNUMBER AND MGRSSN=SSN AND LNAME='Smith')
   UNION
   (SELECT PNAME FROM PROJECT, WORKS_ON, EMPLOYEE WHERE PNUMBER=PNO AND ESSN=SSN AND NAME='Smith')
6. Retrieve the name of each employee who has a dependent with the same first name as the employee.
   SELECT E.FNAME, E.LNAME FROM EMPLOYEE AS E WHERE E.SSN IN (SELECT ESSN FROM DEPENDENT WHERE ESSN=E.SSN AND E.FNAME=DEPENDENT_NAME)
7. Retrieve those employees who have no dependents.
   SELECT FNAME, LNAME FROM EMPLOYEE
   WHERE NOT EXISTS

   (SELECT * FROM DEPENDENT WHERE SSN=ESSN)
8. Produce summary report of each department whether it has employees or not.
   select d.dname,if(count(e.emp_id)=0,'No','Yes') as has_emp from employees e right outer join departments d on e.d_no=d.dno group by d.dname;
9. Procedure to find number of employees in each department along with their average salary.
   Create procedure dept_count(IN dno INT(3),OUT cnt int(3),OUT avg_sal INT(5))
   Begin
   Select count(*) into cnt from employees where d_no=dno;

```
Select avg(salary) into avg_sal from employees where d_no=dno;
Select cnt,avg_sal;
End $$
```

10. Procedure to find the number of employees under each manager. Display the employee count along with their respective managers.

```
Create procedure subordinate()
Begin
Declare sub_emp int(5);
Declare manager varchar(20);
Declare nomorerows int;
Declare sub_count cursor for
Select count(e.emp_id),m.ename from employees e, employees m where e.mgrid=m.eid
group by e.mgrid;
Declare continue handler for not found set nomorerows=0;
Open sub_count;
Looprows:loop
If nomorerows=0 then
Close sub_count;
Leave looprows;
End if;
Fetch sub_count into sub_emp, manager;
End loop;
End $$
```

11. Procedure to update salary:
    If salary < 30000 increase 15%
    If salary >30000 and less than 35000 increase 10%

```
delimiter $$
create procedure update_salary()
begin
declare sal int(5);
declare nomorerows int;
declare sal_cursor cursor for select salary from employees;
declare continue handler for not found set nomorerows=0;
open sal_cursor;
looprows:loop
if nomorerows=0 then
close sal_cursor;
leave looprows;
end if;
fetch sal_cursor into sal;
if sal<30000 then
update employees set salary=salary+(salary*0.15);
else
update employees set salary=salary+(salary*0.10);
```

```
end if;
end loop;
end $$
    delimiter ;
```

12. Procedure to find whether a number is even or odd.

```
delimiter $$
create procedure EVEN_ODD(IN NUM INT(5))
begin
if num%2=0 then
select 'num is even';
else
select 'num is odd';
end $$
delimiter ;
```

13. Procedure to concatenate two strings passed as argument and capitalize the first letter of both words.

```
delimiter $$
create procedure str_cat(IN STR1 VARCHAR(10),IN STR2 VARCHAR(10),OUT STR3
varchar(20))
begin
set
STR3=concat(concat(UCASE(LEFT(STR1,1)),LCASE(SUBSTRING(STR1,2))),concat(UCASE(LEFT(S
TR2,1)),LCASE(SUBSTRING(STR2,2))));
end $$
delimiter ;
```

14. Procedure to retrieve names of the employees above 25 years.

```
delimiter $$
create procedure find_age()
begin
declare v_age INT(5);
declare nomorerows INTEGER;
declare emp_age cursor for select YEAR(curdate())-year(dob) as age from employees where
(YEAR(curdate())-year(dob))>25;
declare continue handler for not found set nomorerows=0;
open emp_age;
looprows:loop
if nomorerows=0 then
close emp_age;
leave looprows;
end if;

fetch emp_age into v_age;
select v_age ;
end loop;
end $$
```

delimiter ;

15. Write a trigger to audit changes on emp table: record type of operation, time, old ename and deptid, new ename and deptid

```
//Trigger for insert

delimiter $$

create trigger emp_insert

after insert on employees

for each row

begin

insert into audit_emp values('insert',curdate(),Null,concat(NEW.emp_name,New.d_no));

end $$

delimiter ;

//trigger for update

delimiter $$

create trigger emp_update

after update on employees

for each row

begin

insert into audit_emp
values('update',curdate(),concat(OLD.emp_name,OLD.d_no),concat(NEW.emp_name,New.d_no));

end $$

delimiter ;

// Trigger for delete

delimiter $$

create trigger emp_delete

after delete on employees

for each row

begin

insert into audit_emp values('update',curdate(),concat(OLD.emp_name,OLD.d_no),Null);

end $$
```