HOSPITAL

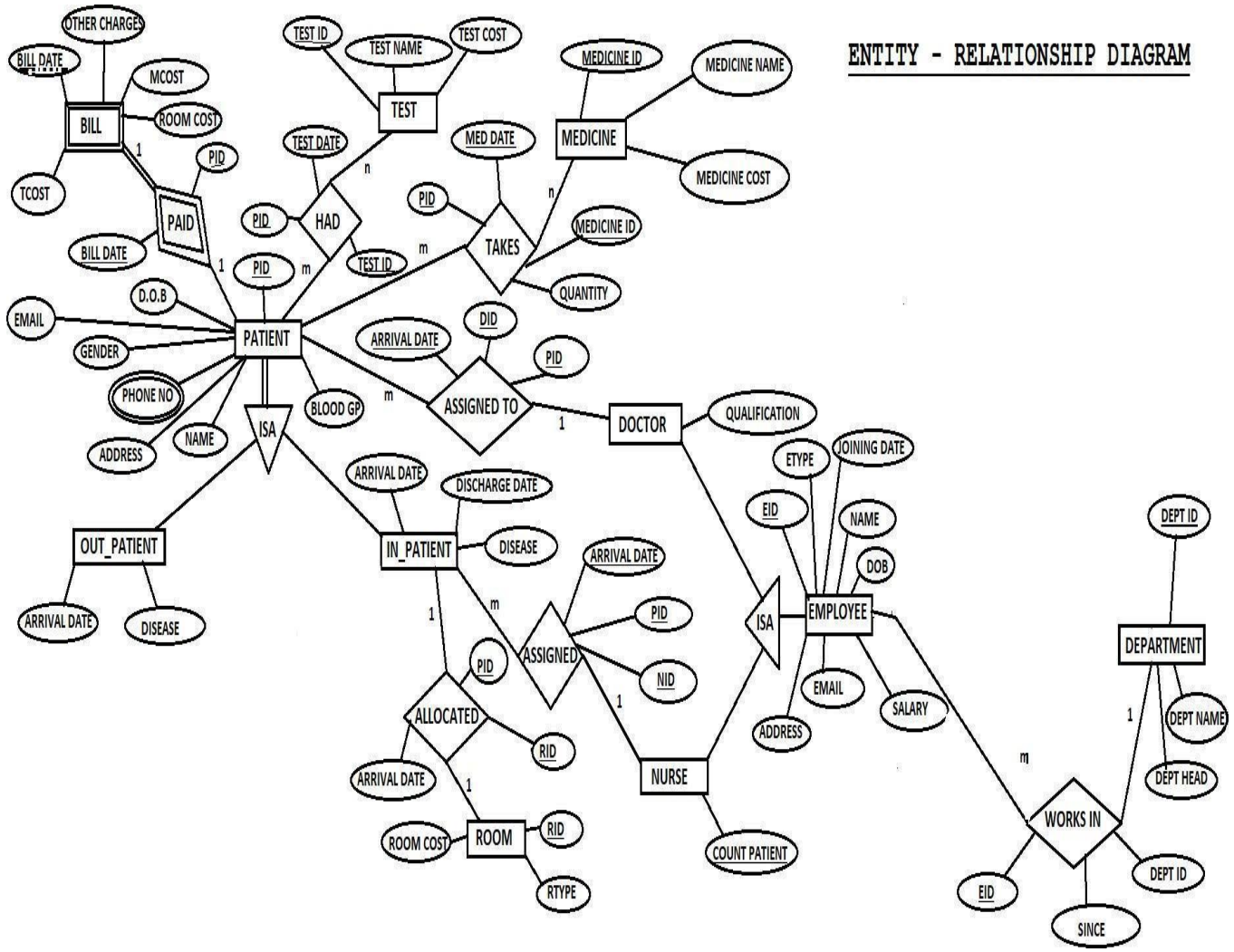
MANAGEMENT

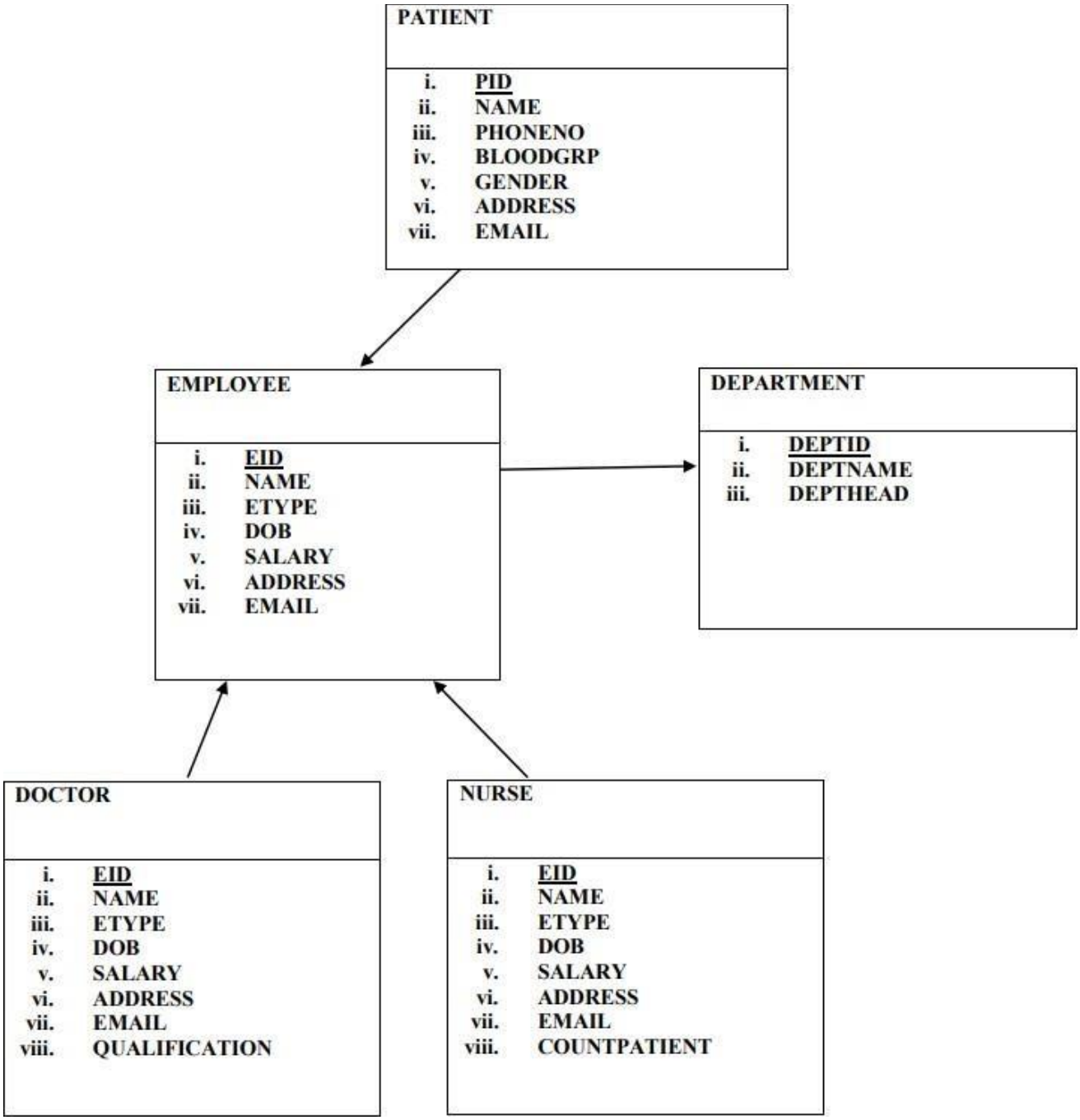
SYSTEM

BY-

KARTIKAY TANDON

ER- DIAGRAM


ENTITY - RELATIONSHIP DIAGRAM
OTHER CHARGES
BILL DATE
MCOST
BILL
ROOM COST
TCOST
PID
PAID
BILL DATE
TEST ID
TEST NAME
TEST COST
TEST
TEST DATE
HAD
PID
TEST ID
MED DATE
MEDICINE
MEDICINE ID
MEDICINE NAME
MEDICINE COST
TAKES
PID
MEDICINE ID
QUANTITY
D.O.B
EMAIL
GENDER
PHONE NO
ADDRESS
NAME
PATIENT
BLOOD GP
ISA
OUT_PATIENT
ARRIVAL DATE
DISEASE
ARRIVAL DATE
DISCHARGE DATE
IN_PATIENT
DISEASE
ARRIVAL DATE
ASSIGNED
PID
NID
PID
ALLOCATED
RID
ARRIVAL DATE
ROOM
RID
ROOM COST
RTYPE
ARRIVAL DATE
DID
PID
ASSIGNED TO
DOCTOR
QUALIFICATION
NURSE
COUNT PATIENT
ISA
EMPLOYEE
ETYPE
JOINING DATE
EID
NAME
DOB
EMAIL
SALARY
ADDRESS
DEPARTMENT
DEPT ID
DEPT NAME
DEPT HEAD
WORKS IN
EID
SINCE
DEPT ID


PATIENT
i. PID
ii. NAME
iii. PHONENO
iv. BLOODGRP
v. GENDER
vi. ADDRESS
vii. EMAIL
EMPLOYEE
i. EID
ii. NAME
iii. ETYPE
iv. DOB
v. SALARY
vi. ADDRESS
vii. EMAIL
DEPARTMENT
i. DEPTID
ii. DEPTNAME
iii. DEPTHEAD
DOCTOR
i. EID
ii. NAME
iii. ETYPE
iv. DOB
v. SALARY
vi. ADDRESS
vii. EMAIL
viii. QUALIFICATION
NURSE
i. EID
ii. NAME
iii. ETYPE
iv. DOB
v. SALARY
vi. ADDRESS
vii. EMAIL
viii. COUNTPATIENT

AIM:-

Our project is based on Hospital Management System. In our project we have tried to make it easy for a patient to book an appointment of a doctor and doctor will tell him disease. Final bill for a patient will also be updated.

Description:

In this project, we have focused on 3 main tables namely

Employee->Doctor , Nurse, Patient,Department. In this project, we have used technologies like SQL and PL/SQL for various operations that can beperformed in our database.

Normalization Process:

1NF- First Normal Form

If a relation contains a composite or multi-valued attribute, it violates the first normal form, or the relationship is in the first normal form if it does not contain any composite or multi-valued attribute. A relation is in its first normal form if every attribute in that relation is singled valued attribute. A table is in 1 NF iff:

1. There are only Single Valued Attributes.
2. Attribute Domain does not change.
3. There is a unique name for every Attribute/Column.
4. The order in which data is stored does not matter.

Patient Table

---

pid – pid column satisfies all the above conditions.
Patient name –Name column satisfies all the above conditions.
blood grp– blood grp column satisfies all the above conditions.

Phone No – Here phone number is a multivalued column. To get our table in a 1NF form we need to make it a single-valued column.

Gender-gender column satisfies all the above conditions.

Address-address column satisfies all the above conditions.

Email-email column satisfies all the above condition

PATIENT

PID	NAME	BLOODGRP	PHONENO	GENDER	ADDRESS	EMAIL

PID	NAME	BLOODGRP	PHONENO1	PHONENO2	GENDER	ADDRESS	EMAIL

EMPLOYEE Table

Eid – eid column satisfies all the above conditions.

etype – etype column satisfies all the above conditions.

Name – name column satisfies all the above conditions.

Dob – dob column satisfies all the above conditions.

salary-salary column satisfies all the above conditions.

Address-address column satisfies all the above conditions.

Email-email column satisfies all the above conditions.

All the attributes satisfy the above 4 conditions. Our EMPLOYEE table is alreadyin First Normal Form

.

EMPLOYEE

EID	NAME	ETYPE	DOB	GENDER	SALARY	EMAIL

DOCTOR Table

All the attributes satisfy the above 4 conditions. Our Complaint table is already in First Normal Form.

DOCTOR

EID	NAME	ETYPE	DOB	GENDER	SALARY	QUALIFICATION	EMAIL

NURSE Table

All the attributes satisfy the above 4 conditions. Our Complaint table is already in First Normal Form.

NURSE

EID	NAME	ETYPE	DOB	GENDER	SALARY	COUNTPATIENT	EMAIL

DEPARTMENT

All the attributes satisfy the above 4 conditions. Our Complaint table is already in First Normal Form.

DEPTID	DEPTNAME	DEPTHEAD

Now we have our database schema normalized to First Normal Form

2NF- Second Normal Form

To be in the second normal form, a relation must be in the first normal form and the relation must not contain any partial dependency. A relation is in 2NF if it has No Partial Dependency, i.e., no non-prime attribute (attributes that are not part of any candidate key) is dependent on any proper subset of any candidate key of the table.

Patient Table

PATIENT

PID	NAME

PID	PHONE_NO_1	PHONE_NO_2

PID	ADDRESS	GENDER	EMAIL

EmployeeTable

Employee

EID	NAME	ETYPE	DOB	GENDER	SALARY	EMAIL

DOCTOR Table

DOCTOR

EID	NAME	ETYPE	DOB	GENDER	SALARY	QUALIFICATION	EMAIL

NURSE Table

NURSE

EID	NAME	ETYPE	DOB	GENDER	SALARY	COUNTPATIENT	EMAIL

DEPARTMENT Table

DEPARTMENT

DEPTID	DEPTNAME	DEPTHEAD

3NF- Third Normal Form

A relation that is in First and Second Normal Form and in which no non-primary key attribute is transitively dependent on the primary key, then it is in Third Normal Form (3NF). If A->B and B->C are two FDs then A->C is called transitive dependency.

Patient

PID	NAME

PID	PHONE_NO_1	PHONE_NO_2

PID	ADDRESS	GENDER	EMAIL

EmployeeTable

Employee

EID	NAME	ETYPE	DOB	GENDER	SALARY	EMAIL

DOCTOR Table

DOCTOR

EID	NAME	ETYPE	DOB	GENDER	SALARY	QUALIFICATION	EMAIL

NURSE Table

NURSE

EID	NAME	ETYPE	DOB	GENDER	SALARY	COUNTPATIENT	EMAIL

DEPARTMENT Table

DEPARTMENT

DEPTID	DEPTNAME	DEPTHEAD

PLSQL COMMANDS TO CREATE TABLE AND TRIGGERS

```
CREATE TABLE patient(
   pid int,
   fname varchar(20) not null,
   lname varchar(20),
   gender varchar(6) not null,
   dob date not null,
   blood_group varchar(3),
   doc_id int,
   HNo varchar(10),
   street varchar(20),
   city varchar(16),
   state varchar(20),
   email varchar(30),
   Primary Key(pid));


CREATE TABLE Employee(
   empid int,
   fname varchar(20) not null,
    mname varchar(20),
   lname varchar(20),
   gender varchar(6) not null,
   emptype varchar(20) not null,
   Hno varchar(10),
   street varchar(20),
   city varchar(20), state
   varchar(20),
   date_of_joining date,
   email varchar(30),
   deptid int,
   since date,
date_of_birth date,
   PRIMARY key(empid));

CREATE TABLE department(
   deptid int,
   dname varchar(20) not null,
   dept_headid int(10),
   PRIMARY key(deptid));


CREATE table salary(
   etype varchar(20),
   salary float(20,2),
   PRIMARY key(etype));
```

```
CREATE TABLE nurse_assigned(
   nid int,
   countpatient int,
   PRIMARY KEY(nid),
    FOREIGN KEY(nid) REFERENCES employee(empid));

CREATE TABLE out_patient(
   pid int,
   arrival_date date,
   disease varchar(40),
   PRIMARY key(pid,arrival_date),
   FOREIGN KEY(pid) REFERENCES patient(pid));

CREATE TABLE room(
   rid int,
   roomtype varchar(20),
   PRIMARY key(rid));


   CREATE TABLE in_patient(
   pid int,
   nid int,
   rid int,
   arrival_date date,
      discharge_date date,
      disease varchar(40),
      primary key(pid,arrival_date),
      FOREIGN key(pid) REFERENCES patient(pid),
      FOREIGN KEY(nid) REFERENCES employee(empid),
      FOREIGN key(rid) REFERENCES room(rid));


CREATE TABLE room_cost(
   roomtype varchar(20),
   rcost int,
   PRIMARY KEY(rtype));

CREATE TABLE test(
   tid int,
   tname varchar(20),
   tcost float(10,2),
   primary KEY(tid));

CREATE TABLE hadtest(
   pid int,
   tid int,
   testdate date,
   PRIMARY KEY(pid,tid,testdate),
   FOREIGN KEY(pid) REFERENCES patient(pid),
   FOREIGN key(tid) REFERENCES test(tid));

CREATE TABLE medicine(
    mid int,
    mname varchar(40) not null,
   mcost  float(20,2),
   PRIMARY key(mid));
```

```
CREATE TABLE doctor(
 doc_id int,
 qualification varchar(20),
 PRIMARY KEY(doc_id),
 FOREIGN    KEY(doc_id) REFERENCES employee(empid) ON DELETE CASCADE);


  CREATE TABLE had_medicine(
   mid int,
   med_date date,
   qty int,
   PRIMARY KEY(pid,mid,med_date),
   FOREIGN KEY(pid) REFERENCES patient(pid),
   FOREIGN KEY(mid) REFERENCES medicine(mid));


  CREATE TABLE pt_phone(
   pid int,
   phoneno varchar(10),
   PRIMARY KEY(pid,phoneno),
   FOREIGN KEY(pid) REFERENCES patient(pid));


  CREATE TABLE emp_phone(
   empid int,
   phoneno varchar(10),
   PRIMARY KEY(empid,phoneno));


CREATE TABLE bill(
   pid int,
    mcost  float(20,2),
   tcost float(20,2),
   roomcharges   float(20,2),
   othercharges   float(20,2),
   billdate date,
   PRIMARY KEY(pid,billdate));


 CREATE TABLE  "DOCTORS"
 (            "DOC_ID" NUMBER(5,0) NOT NULL ENABLE,
 "DOC_NAME" VARCHAR2(200) NOT NULL ENABLE,
 "HIRE_DATE" DATE,
 "SALARY" NUMBER(12,2),
 "NATIONALITY" VARCHAR2(200),
 "DOB" DATE,
 "DEPT_ID" NUMBER(2,0) NOT NULL ENABLE,
 "P_COUNT" NUMBER(5,0),
  CONSTRAINT "DOCTORS_PK" PRIMARY KEY ("DOC_ID") ENABLE
  )

DELIMITER //
CREATE TRIGGER transfer_to_passive BEFORE DELETE ON employee
FOR EACH ROW
BEGIN
     INSERT into employee_inactive
      SELECT
empid,fname,mname,lname,gender,emptype,hno,street,city,state,date_of_joining,CURRENT_DATE,email FROM
employee WHERE employee.empid=OLD.empid;
END;//
```

```
DELIMITER //
CREATE TRIGGER on_insertemployee_update_dept
AFTER INSERT ON employee
for EACH ROW
BEGIN
SET @ab=(SELECT dept_headid FROM department WHERE department.deptid=NEW.deptid);
UPDATE department SET dept_headid=CASE
WHEN (@ab is NULL AND department.deptid=NEW.deptid)
THEN new.empid
else dept_headid
END;
END;//
DELIMITER ;


CREATE TABLE prev_department(
   empid int,
   deptid int,
   date_of_joining date,
   date_of_leaving date,
   PRIMARY KEY(empid,deptid,date_of_leaving));


DELIMITER //
CREATE TRIGGER transfer_to_prev_department AFTER UPDATE ON employee FOR
EACH ROW
BEGIN
    IF NEW.deptid<>OLD.deptid THEN
    INSERT into prev_department values(OLD.empid,OLD.deptid,OLD.since,CURRENT_TIMESTAMP); SET
    @ab=(SELECT dept_headid FROM department WHERE department.deptid=NEW.deptid); UPDATE
    department SET dept_headid=CASE
    WHEN (@ab is NULL)
    THEN new.empid
    else dept_headid
          END;
    SET @bb=(SELECT dept_headid FROM department WHERE department.deptid=OLD.deptid);

    IF OLD.empid=@bb THEN
    SET @gb=(SELECT empid FROM employee WHERE deptid=OLD.deptid);
    SET @mn=(SELECT MIN(since) FROM employee WHERE empid=@gb);
    SET @mm=(SELECT MIN(empid) FROM employee WHERE employee.since=@mn);
    UPDATE department SET dept_headid=@mm WHERE department.deptid=OLD. deptid;
    END IF;
    END IF;
END;//
DELIMITER ;

DELIMITER //
CREATE TRIGGER employee_on_delete AFTER DELETE ON employee FOR
EACH ROW
BEGIN
    INSERT into prev_department values(OLD.empid,OLD.deptid,OLD.since,CURRENT_TIMESTAMP); SET

    @bb=(SELECT dept_headid FROM department WHERE department.deptid=OLD.deptid);

    IF OLD.empid=@bb THEN
    SET @gb=(SELECT empid FROM employee WHERE deptid=OLD.deptid);
    SET @mn=(SELECT MIN(since) FROM employee WHERE empid=@gb);
```

```
SET @mm=(SELECT MIN(empid) FROM employee WHERE employee.since=@mn);
UPDATE department SET dept_headid=@mm WHERE department.deptid=OLD.deptid; END
IF;

END;//
DELIMITER ;
```

PLSQL COMMANDS FOR PROCEDURES AND INSERTING VALUES IN TABLE

```
CREATE OR REPLACE PROCEDURE
INSERTDOCTOR( D_NAME IN DOCTORS.DOC_NAME%TYPE, D_HIREDATE IN
DOCTORS.HIRE_DATE%TYPE, D_SALARY IN DOCTORS.SALARY%TYPE, D_NATIONALITY
IN DOCTORS.NATIONALITY%TYPE, D_DOB IN DOCTORS.DOB%TYPE, D_DEPT_ID IN
DOCTORS.DEPT_ID%TYPE)
IS
BEGIN
INSERT INTO DOCTORS(DOC_NAME, HIRE_DATE, SALARY, NATIONALITY, DOB,
DEPT_ID) VALUES(D_NAME, D_HIREDATE, D_SALARY, D_NATIONALITY, D_DOB,
D_DEPT_ID);
END;

CREATE OR REPLACE PROCEDURE INSERTPATIENT( P_NAME IN
PATIENTS.PAT_NAME%TYPE, P_GENDER IN PATIENTS.GENDER%TYPE, P_AGE IN
PATIENTS.AGE%TYPE, P_DIAGNOSIS IN PATIENTS.DIAGNOSIS%TYPE, P_DOCID IN
PATIENTS.DOC_ID%TYPE, P_ROOM IN PATIENTS.ROOM_NO%TYPE, P_BILL IN
PATIENTS.BILL%TYPE)
IS
BEGIN
INSERT INTO PATIENTS( PAT_NAME, GENDER, AGE, DIAGNOSIS, DOC_ID, ROOM_NO, BILL,
ADMITTED) VALUES (P_NAME, P_GENDER, P_AGE, P_DIAGNOSIS, P_DOCID, P_ROOM,
P_BILL, TO_DATE(SYSDATE, 'DD-MM-YYYY')); END;

CREATE OR REPLACE PROCEDURE UPDATEPATIENT( P_NAME IN
PATIENTS.PAT_NAME%TYPE, P_GENDER IN PATIENTS.GENDER%TYPE, P_AGE IN
PATIENTS.AGE%TYPE, P_DIAGNOSIS IN PATIENTS.DIAGNOSIS%TYPE, P_DOCID IN
PATIENTS.DOC_ID%TYPE, P_ROOM IN PATIENTS.ROOM_NO%TYPE, P_BILL IN
PATIENTS.BILL%TYPE, P_ADMITTED IN PATIENTS.ADMITTED%TYPE, P_INPUT_ID IN
PATIENTS.PAT_ID%TYPE)

CREATE OR REPLACE PROCEDURE BILL(PATIENT_ID IN PATIENTS.PAT_ID%TYPE) IS
DAYS NUMBER(8);
BEGIN
DAYS:=CALCULATEDAYS(PATIENT_ID);
UPDATE PATIENTS SET BILL = BILL+(3000*DAYS) WHERE PAT_ID = PATIENT_ID;
DBMS_OUTPUT.PUT_LINE('FINALIZED BILL');
EXCEPTION
WHEN NO_DATA_FOUND THEN
DBMS_OUTPUT.PUT_LINE('USER INVALID');
END;

INSERT test (1,"X-RAY",100); INSERT
test (2,"BLOOD TEST",300); INSERT
test (3,"URINE TEST",100); INSERT
test (4,"MRI SCAN",1200); INSERT test
(5,"ENDOSCOPY",3000);
```

INSERT test (6,"BIOPSY",3500);
INSERT test (7,"ULTRASOUND",900);
INSERT test (8,"CT-SCAN",1100);
INSERT test (9,"CBC",350);
INSERT test (10,"FLU TEST",1500);

TID	TNAME	TCOST
1	X-RAY	100
2	BLOOD TEST	300
3	URINE TEST	100
4	MRI SCAN	1200
5	ENDOSCOPY	3000
6	BIOPSY	3500
7	ULTRASOUND	900
8	CT-SCAN	1100
9	CBC	350
10	FLU TEST	1500

INSERT salary ("DOCTOR",70000);
INSERT salary ("NURSE",25000);
INSERT salary ("RECEPTIONIST",20000);
INSERT salary ("ACCOUNTANT",15000);
INSERT salary ("CLEANER",14000);
INSERT salary ("SECURITY",12000);
INSERT salary ("AMBULANCE DRIVER",13000);

ETYPE	SALARY
DOCTOR	70000
NURSE	25000
RECEPTIONIST	20000
ACCOUNTANT	15000
CLEANER	14000
SECURITY	12000
AMBULANCE DRIVER	13000

```
INSERT medicine (1,"CROCINE",10);
INSERT medicine (2,"ASPIRIN",8); INSERT
medicine (3,"NAMOSLATE",8); INSERT
medicine (4,"GLUCOSE",200); INSERT
medicine (5,"TARIVID",80); INSERT
medicine (6,"CANESTEN",12); INSERT
medicine (7,"DICLOFENAC",18); INSERT
medicine (8,"ANTACIDS",8); INSERT
medicine (9,"VERMOX",40); INSERT
medicine (10,"OVEX",25); INSERT medicine
(11,"OMEE",35); INSERT medicine
(12,"AVIL",4);
INSERT medicine (13,"HIDRASEC",50);
INSERT medicine (14,"UTINOR",80);
INSERT medicine (15,"PARIET",8); INSERT
medicine (16,"CIPROXIN",6); INSERT
medicine (17,"CYPROSTAT",12); INSERT
medicine (18,"ANDROCUR",80); INSERT
medicine (19,"DESTOLIT",82); INSERT
medicine (20,"URSOFALK",15); INSERT
medicine (21,"ORS",7);
INSERT medicine (22,"URSOGAL",20);
INSERT medicine (23,"OMNI GEL",30);
INSERT medicine (24,"DETTOL",45);
INSERT medicine (25,"BETADINE",8);
INSERT medicine (26,"LIVER-52",100);
INSERT medicine (27,"METHYLPHENIDATE",12);
INSERT medicine (28,"BETA-BLOCKER",90);
INSERT medicine (29,"BENZODIAZEPINES",120);
INSERT medicine (30,"Z-DRUG",150);
INSERT medicine (31,"ANTIPSYCHOTIC",200);
INSERT medicine (32,"SSRI-ANTIDEPRESSANT",250);
INSERT medicine (33,"MAOI-DRUG",140);
INSERT medicine (34,"BICASUL",1);
INSERT medicine (35,"NASAL DECONGESTANTS",20);
INSERT medicine (36,"EXPECTORANTS",10);
INSERT medicine (37,"COUGH SUPRESSANTS",60);
INSERT medicine (38,"ANTI HISTAMINES",40);
INSERT medicine (39,"ACETAMINOPHEN",60);
INSERT medicine (40,"HPV VACCINE",140); INSERT
medicine (41,"SYRINGE",3);
INSERT medicine (42,"INJECTION",10);
INSERT medicine (43,"MORFIN",5);
INSERT medicine (44,"ORLISTAT",10);
INSERT medicine (45,"ZALASTA",85);
INSERT medicine (46,"ZANTAC",84);
INSERT medicine (47,"ZEFFIX",82);
INSERT medicine (48,"ZINNAT",100);
INSERT medicine (49,"ZOFRAN",80);
INSERT medicine (50,"ZOCOR",18);
```

MID	MNAME	MCOST
1	CROCINE	10
2	ASPIRIN	8
3	NAMOSLATE	8
4	GLUCOSE	200
5	TARIVID	80
6	CANESTEN	12
7	DICLOFENAC	18
8	ANTACIDS	8
9	VERMOX	40
10	OVEX	25
11	OMEE	35
12	AVIL	4
13	HIDRASEC	50
14	UTINOR	80
15	PARIET	8
16	CIPROXIN	6
17	CYPROSTAT	12
18	ANDROCUR	80
19	DESTOLIT	82
20	URSOFALK	15
21	ORS	7
22	URSOGAL	20
23	OMNI GEL	30
24	DETTOL	45
25	BETADINE	8
26	LIVER-52	100
27	METHYLPHENIDATE	12
28	BETA-BLOCKER	90
29	BENZODIAZEPINES	120
30	Z-DRUG	150
31	ANTIPSYCHOTIC	200
32	SSRI-ANTIDEPRESSANT	250
33	MAOI-DRUG	140
34	BICASUL	1
35	NASAL DECONGESTANTS	20
36	EXPECTORANTS	10
37	COUGH SUPRESSANTS	60
38	ANTI HISTAMINES	40
39	ACETAMINOPHEN	60
40	HPV VACCINE	140
41	SYRINGE	3
42	INJECTION	10
43	MORFIN	5

44	ORLISTAT	10
45	ZALASTA	85
46	ZANTAC	84
47	ZEFFIX	82
48	ZINNAT	100
49	ZOFRAN	80
50	ZOCOR	18

```
DELIMITER //
CREATE TRIGGER update_nurse_assigned BEFORE INSERT ON employee
FOR EACH ROW
BEGIN
IF NEW.emptype="NURSE" THEN
INSERT INTO nurse_assigned VALUES(New.empid,0);
end if;
end; //

DELIMITER ;
```

```
DELIMITER //
CREATE TRIGGER decrease_on_discharge BEFORE INSERT ON bill FOR
EACH ROW
BEGIN
UPDATE room set isfree=0 WHERE rid=(SELECT rid FROM in_patient WHERE in_patient.pid=NEW.pid AND
discharge_date is null);
```

UPDATE nurse_assigned SET count patient=countpatient-1 WHERE nid=(SELECT nid FROM in_patient WHERE END; // DELIMITER ;

insert department (1,"ALLERGY",NULL);
insert department (2,"INTENSIVE CARE",NULL);
insert department (3,"ANESTHESIOLOGY",NULL);
insert department (4,"CARDIOLOGY",NULL);
insert department (5,"ENT",NULL);
insert department (6,"NEUROLOGY",NULL);
insert department (7,"ORTHOPEDICS",NULL);
insert department (8,"PATHOLOGY",NULL);
insert department (9,"RADIOLOGY",NULL);
insert department (10,"SURGERY",NULL); insert department (11,"NURSE",NULL);
insert department (12,"ACCOUNTS",NULL);
insert department (13,"SECURITY",NULL);
insert department (14,"CLEANER",NULL);

DEPTID	DNAME	DEPT_HEADID
1	ALLERGY	-
2	INTENSIVE CARE	-
3	ANESTHESIOLOGY	-
4	CARDIOLOGY	-
5	ENT	-
6	NEUROLOGY	-
7	ORTHOPEDICS	-
8	PATHOLOGY	-
9	RADIOLOGY	-

7	ORTHOPEDICS	-
8	PATHOLOGY	-
9	RADIOLOGY	-
10	SURGERY	-
11	NURSE	-
12	ACCOUNTS	-
13	SECURITY	-
14	CLEANER	-

```
ALTER TABLE employee ADD FOREIGN KEY(deptid) REFERENCES department(deptid) ON UPDATE CASCADE;

ALTER TABLE employee ADD FOREIGN KEY(emptype) REFERENCES salary(etype) ON DELETE RESTRICT;
DELIMITER //
CREATE TRIGGER delete_null_disease_from_outpatient AFTER INSERT ON bill
FOR EACH ROW
BEGIN
#SET @bb=SELECT pid,arrival_date FROM out_patient WHERE out_patient.pid=NEW.pid AND arrival_date is
NULL;
DELETE FROM out_patient WHERE disease is NULL;
END; //
DELIMITER ;
```