**HOSPITAL MANAGEMENT SYSTEM**

**UCS 310 Database Management System Project Report**

**END-Semester Evaluation**

**Submitted by:**

**(102117032) RIDHI THAKUR**

**(102117045) SHIYA MER**

**(102117052) HARNEET KAUR**

**(102117057) IRA GUPTA**

**BE Second Year, CSE**

**Submitted to:**

**ARCHANA SINGH**



**Computer Science and Engineering Department**

**TIET, Patiala**

**May 2023**

**INDEX**

|  |  |  |
| --- | --- | --- |
|  |  |  |
| S.NO | TOPIC PAGE NO. |  |
| 1 | PROJECT OBJECTIVE 3 |  |
|  |  |  |
| 2 | RELATIONAL SCHEMA 6 |  |
|  |  |  |
| 3 | ER DIAGRAM 6 |  |
|  |  |  |
| 4 | ER TO TABLE 7 |  |
|  |  |  |
| 5 | NORMALISATION 8 |  |
|  |  |  |
| 6 | FUNCTIONS ,CODES AND EXCEPTIONS 14 |  |
|  | * CREATE FUNCTION 14 |  |
|  | * INSERT FUNCTION 14 |  |
|  | * TRIGGERS 20 |  |
|  | * EXCEPTIONS 25 |  |
|  | * CURSORS 26 |  |
|  | * CONSTRAINTS 28 |  |
|  | * FUNCTIONS AND 29   PROCEDURES |  |
|  |  |  |
| 7 | QUERIES 30 |  |
|  |  |  |
| 8 | CONCLUSION 33 |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

1. **PROJECT OBJECTIVE :**

Our project Hospital Database Management system includes registration of patients, storing their disease details into the system. It will also contain doctor’s information and will digitalize the whole billing system. It has the facility to give a unique id for every patient and stores the details of every patient and staff automatically. It includes a search facility to know the current status of each room. User can search availability of a doctor and the details of a patient using the id. And the whole process conducted by Administrator.

Keywords- Hospital, Administrator, Patients, Doctor, Diseases, Staff, Treatments, Test, Lab reports, Schema.

1. BACKGROUND

A Hospital Database Management System (HDMS) is a computer or web based system that facilities managing the functioning of a hospital or any medical set up. This system will help in making the whole functioning paperless.

The hospital database includes all the necessary patient data. The disease history, test results, prescribed treatment can be accessed by doctors without much delay in order to make an accurate diagnosis and monitor the patient's health. It enables lower risks of mistakes.

A hospital is a place where Patients come up for general diseases. Hospitals Provide facilities like:

• Consultation by Doctors on Diseases.

• Diagnosis for diseases.

• Providing treatment facility.

• Facility for admitting Patients (providing beds, nursing, medicines etc.)

• Immunization for Patients/Children.

Various operational works that are done in a Hospital are:

• Recording information about the Patients that come.

• Generating bills.

• Recording information related to diagnosis given to Patients.

• Keeping record of the Immunization provided to Children/Patients.

• Keeping information about various diseases and medicines available to cure them.

These are the various jobs that need to be done in a Hospital by the operational staff and Doctors. All these works are done on papers.

The work is done as follows:

• Information about Patients is done by just writing the Patients name, age, and gender. Whenever the Patient comes up his information is stored freshly.

• Bills are generated by recording price for each facility provided to Patient on a separate sheet and at last they all are summed up.

• Diagnosis information to patients is generally recorded on the document, which contains Patients information. It is destroyed after some time period to decrease the paper load in the office.

• Immunization records of the children are maintained in pre-formatted sheets, which are kept in a file.

• Information about various diseases is not kept as any document. Doctors themselves do this job by remembering various medicines

All this work is done manually by the receptionist and other operational staff and lot of papers are needed to be handled and taken care of. Doctors have to remember various medicines available for diagnosis and sometimes miss better alternatives as they can’t remember them at that time.

2. IDEAS

The hospital database includes all the necessary patient data. The disease history, lab reports, prescribed treatment can be accessed by doctors without much delay in order to make an accurate diagnosis and monitor the patient's health. It enables lower risks of mistakes.

• The project maintains two levels of users:

• Administrator.

• User Level-Data Entry Operator.

• Now, I discuss the main facilities in this project are:

• Maintaining records of indoor/outdoor patients.

• Maintaining patient’s test and examinations details.

• Providing different test facilities to a Doctor for doctor for diagnosis of a patients.

• Maintaining patient’s prescription, medicine and

diet advice details.

• Providing billing details for indoor/outdoor patients.

• Results of tests, prescription, precautions and diet advice will be automatically updated in the database.

• In this project collection of data in form different pathology labs.

• Related test reports, patient’s details report, billing

reports can be generated as per user requirements.

• User or administrator can search a patient’s record

by his id.

• Hospital Database Management System Design:

The Hospital database management system design is a database design use for managing hospital functions and events. It enables the admin to register a patient for the hospital, stores their disease details into the database. Any of the staff members, doctor & admin is able to add, view, edit, update or delete data.

• Purpose of Hospital Database Management System: The purpose of the Hospital Management System database Design is to make a secure and easy

way of storing information of the patient, doctors,

inpatient, outpatient, Rooms, and Bill payment.

• Features of the Hospital Database Management System: There are seven (8) common features of Hospital Management System Database Design such as

Managing Administrator, Doctors, laboratory,

Inpatient, Outpatient, Rooms, and Hospital Bills information.

]

**Data Query Language (DQL)**

SELECT-Used to retrieve certain records from one or more tables.

**Data Manipulation Language (DML)**

INSERT - Used to create a record UPDATE - Used to change certain records. DELETE - Used to delete certain records.

**Data Definition Language (DDL)**

CREATE - Used to create a new table, a view of a table, or other object in database.

ALTER - Used to modify an existing database object, such as a table.

DROP - Used to delete an entire table, a view of a table or other object in the database.

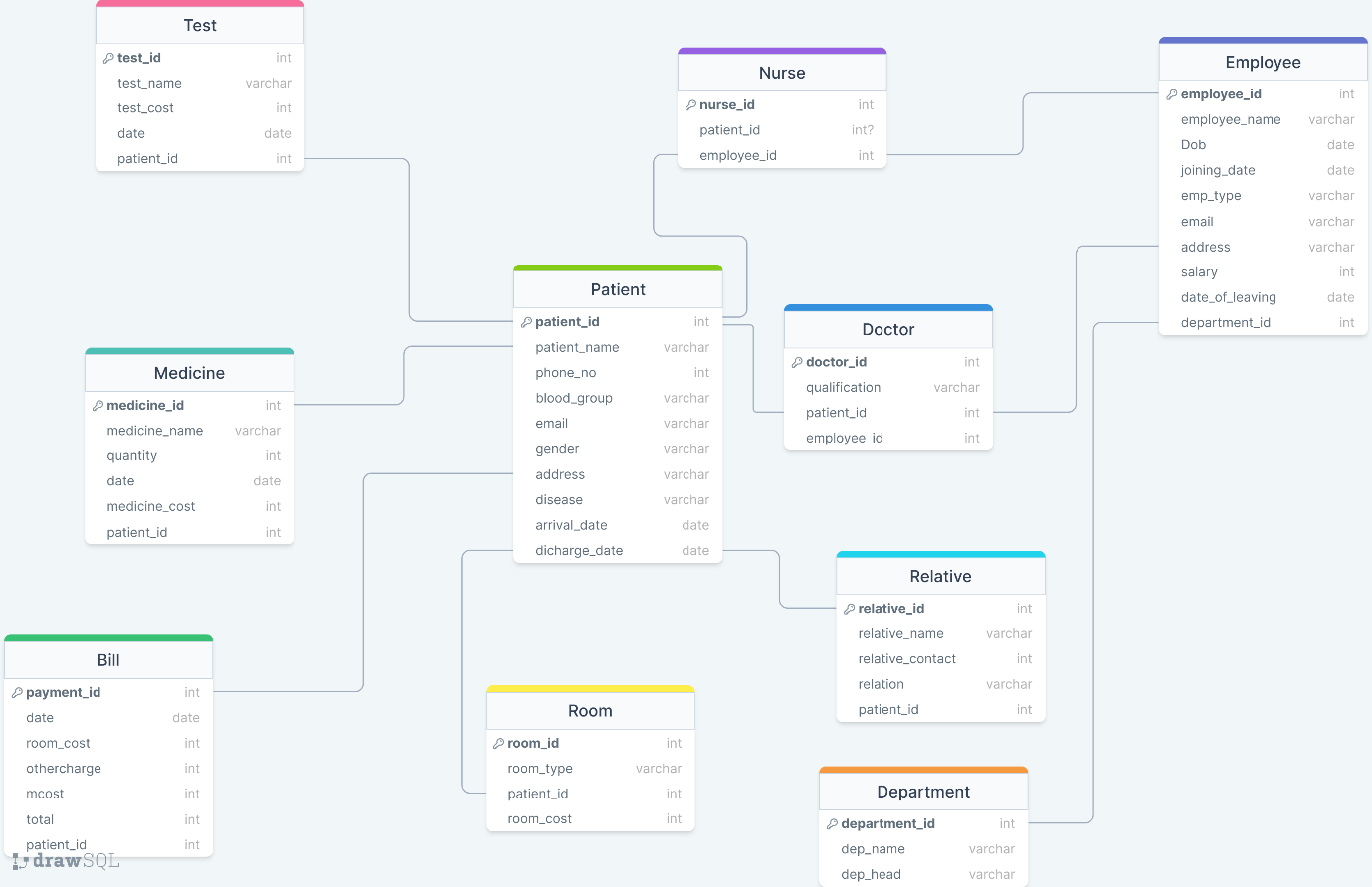
**SOFTWARE REQUIREMENTS**

|  |  |
| --- | --- |
| Operating System | : 64bit WINDOWS Operating System, X64-based processor |
| Database | : MYSQL |

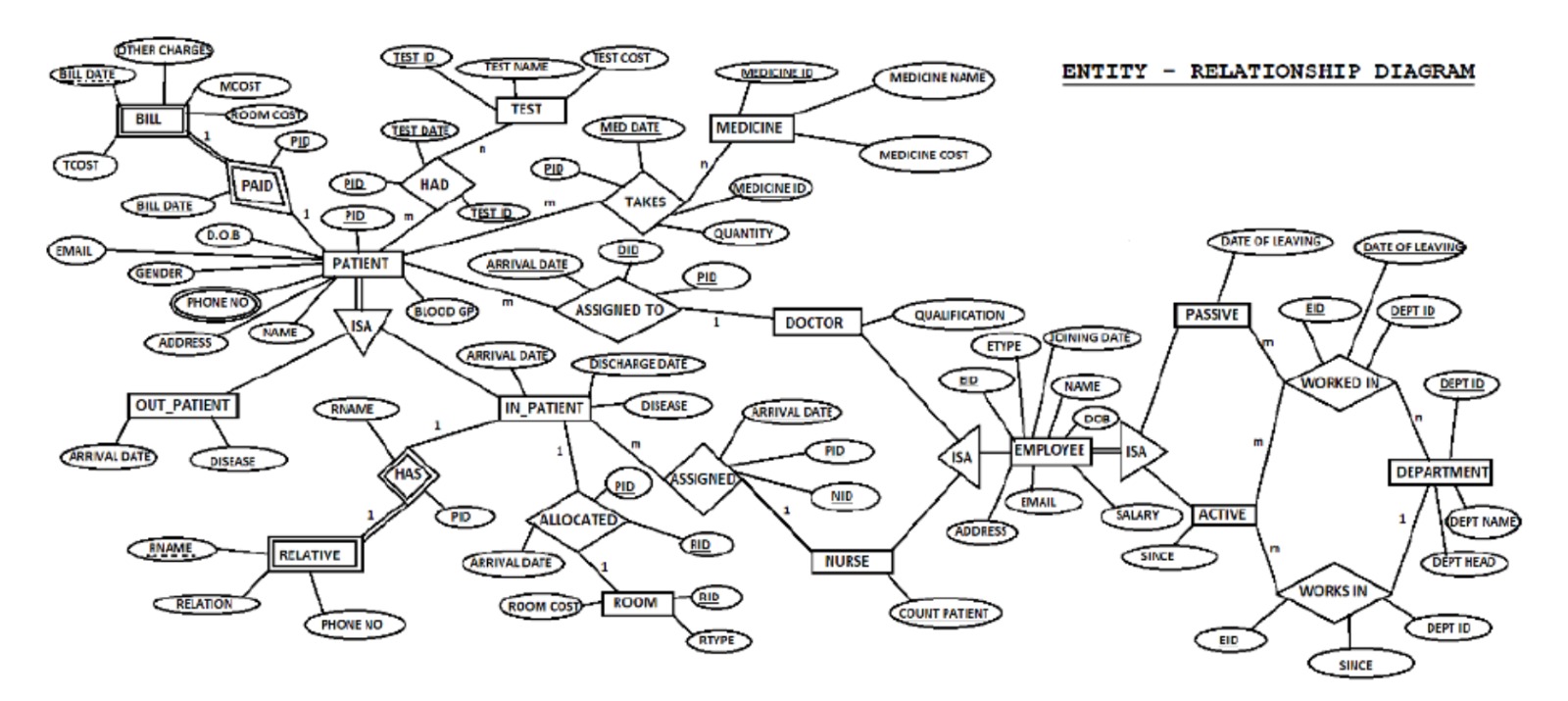
**HARDWARE REQUIREMENTS**

|  |  |
| --- | --- |
| Processor | : Intel Celeron CPU N3060 @1.60GHz or Above |
| RAM | : 4.00 GB or Above |
| Hard Disk | : 1 TB |
| Compact Disk | : CD-ROM, CD-R, CD-RW |
| Input device | : Keyboard |

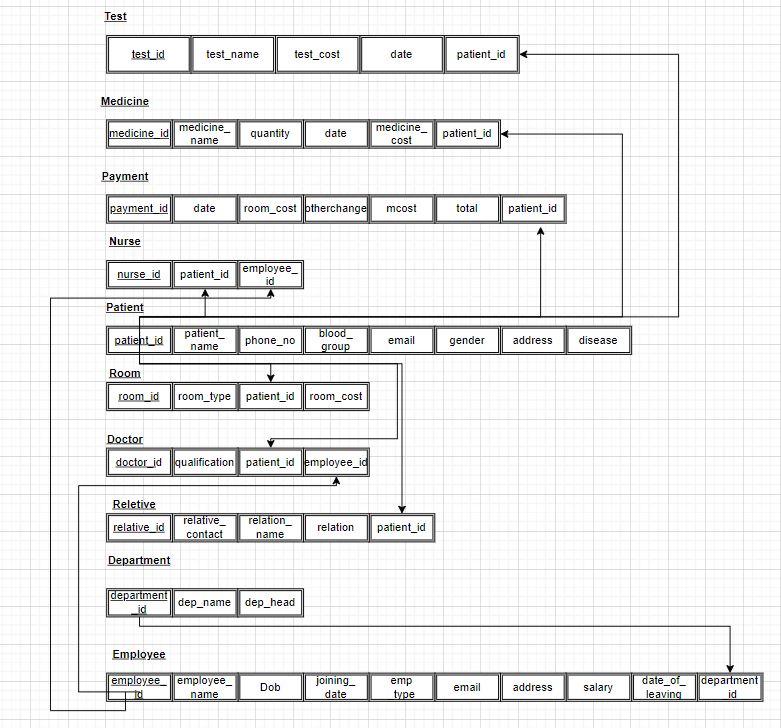
**2.RELATIONAL SCHEMA :**



**3.ER DIAGRAM :**

****

**4. ER TO TABLE**

****

**5.NORMALIZATION**

**5.1. First Normal Form(1NF)**

- All tables have a primary key

- No repeating groups or arrays

- Each column value is atomic

All tables in the given schema are already in 1NF.

**5.2. Second Normal Form (2NF)**

- Be in first normal form (1NF)

- No partial dependencies

The "employee" table has partial dependencies since some of its columns are dependent on only a part of the primary key (employee\_id). To remove partial dependencies, we can split the table into two tables:

employee\_details

- employee\_id (primary key)

- employee\_name

- dob

- joining\_date

- emp\_type

- email

- address

employee\_salary

- employee\_id (primary key and foreign key referencing employee\_details table)

- salary

- date\_of\_leaving

- department\_id (foreign key)

**5.3. Third Normal Form(3NF)**

- Be in second normal form (2NF)

- No transitive dependencies

The "bill" table has a transitive dependency since mcost column is dependent on medicine\_cost and room\_cost columns. To remove the transitive dependency, we can split the table into two tables:

bill\_details

- payment\_id (primary key)

- bill\_date

- room\_cost

- other\_charges

- patient\_id (foreign key)

medicine\_details

- medicine\_id (primary key)

- medicine\_name

- quantity

- med\_date

- medicine\_cost

- patient\_id (foreign key referencing bill\_details table)

bill\_total

- payment\_id (primary key and foreign key referencing bill\_details table)

- mcost

- Total

The resulting normalized schema would be:

department

- department\_id (primary key)

- dep\_name

- dep\_head

employee\_details

- employee\_id (primary key)

- employee\_name

- dob

- joining\_date

- emp\_type

- email

- address

employee\_salary

- employee\_id (primary key and foreign key referencing employee\_details table)

- salary

- date\_of\_leaving

- department\_id (foreign key)

patient

- patient\_id (primary key)

- patient\_name

- phone\_no

- blood\_group

- email

- gender

- address

- disease

- arrival\_date

- discharge\_date

medicine\_details

- medicine\_id (primary key)

- medicine\_name

- quantity

- med\_date

- medicine\_cost

- patient\_id (foreign key referencing bill\_details table)

bill\_details

- payment\_id (primary key)

- bill\_date

- room\_cost

- other\_charges

- patient\_id (foreign key)

bill\_total

- payment\_id (primary key and foreign key referencing bill\_details table)

- mcost

- Total

doctor

- doctor\_id (primary key)

- qualification

- patient\_id (foreign key referencing patient table)

- employee\_id (foreign key referencing employee\_details table)

nurse

- nurse\_id (primary key)

- patient\_id (foreign key referencing patient table)

- employee\_id (foreign key referencing employee\_details table)

relative

- relative\_id (primary key)

- relative\_name

- relative\_contact

- relation

- patient\_id (foreign key referencing patient table)

room

- room\_id (primary key)

- room\_type

- patient\_id (foreign key referencing patient table)

- room\_cost

test

- test\_id (primary key)

- test\_name

- test\_cost

- date1

- patient\_id (foreign key referencing patient table)

Note that this is just one possible normalization for the given schema and there could be other valid normalizations depending on the specific requirements and use cases of the hospital management system.

**5.4. Boyce – Coded Normal Form(BCNF)**

To achieve Boyce-Codd Normal Form (BCNF), we need to ensure that for every functional dependency X → Y in a table, X should be a superkey of that table.

1. department table (already in BCNF)

- department\_id (PK)

- dep\_name

- dep\_head

2. employee table (in BCNF)

- employee\_id (PK)

- employee\_name

- dob

- joining\_date

- emp\_type

- email

- address

- salary

- date\_of\_leaving

- department\_id (FK)

3. patient table (in BCNF)

- patient\_id (PK)

- patient\_name

- phone\_no

- blood\_group

- email

- gender

- address

- disease

- arrival\_date

- discharge\_date

4. medicine table (in BCNF)

- medicine\_id (PK)

- medicine\_name

- quantity

- med\_date

- medicine\_cost

- patient\_id (FK)

5. bill table (in BCNF)

- payment\_id (PK)

- bill\_date

- room\_cost

- other\_charges

- mcost

- Total

- patient\_id (FK)

6. doctor table (in BCNF)

- doctor\_id (PK)

- qualification

- patient\_id (FK)

- employee\_id (FK)

7. nurse table (in BCNF)

- nurse\_id (PK)

- patient\_id (FK)

- employee\_id (FK)

8. relative table (in BCNF)

- relative\_id (PK)

- relative\_name

- relative\_contact

- relation

- patient\_id (FK)

9. room table (in BCNF)

- room\_id (PK)

- room\_type

- patient\_id (FK)

- room\_cost

10. test table (in BCNF)

- test\_id (PK)

- test\_name

- test\_cost

- date1

- patient\_id (FK)

Note: All the tables are already in BCNF, which means that each table's functional dependencies are well-formed and satisfy the requirement of the BCNF.

1. **SCREENSHOTS WITH OUTPUTS**
2. **CREATION AND INSERTION OF TABLES**

**CREATION AND INSERTION OF DEPARTMENT**

create table department(

department\_id int primary key,

dep\_name varchar(256) not null,

dep\_head varchar(256) not null

);

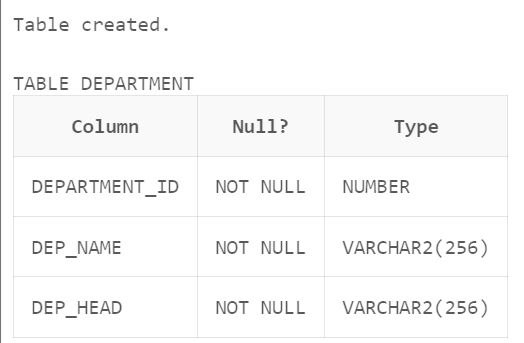
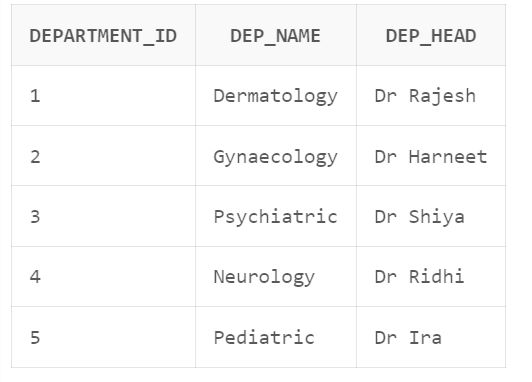
insert into department values(1,'Dermatology','Dr Rajesh');

insert into department values(2,'Gynaecology','Dr Harneet');

insert into department values(3,'Psychiatric','Dr Shiya');

insert into department values(4,'Neurology','Dr Ridhi');

insert into department values(5,'Pediatric','Dr Ira');



**CREATION AND INSERTION OF EMPLOYEE :**

create table employee(

employee\_id int primary key,

employee\_name varchar(256) not null,

dob date not null,

joining\_date date not null,

emp\_type varchar(256) not null,

email varchar(256) not null,

address varchar(256) not null,

salary int not null,

date\_of\_leaving date,

department\_id int,

foreign key (department\_id) references department(department\_id)

);

insert into employee values(101,'Dr Rajesh','10-November-1965','12-November-1987','Doctor','drrajesh@gmail.com','Patiala',150000,NULL,1);

insert into employee values(102,'Sunita','10-May-1985','18-November-2013','Nurse','sunita@gmail.com','Rajpura',28000,NULL,1);

insert into employee values(103,'Dr Harneet','02-May-1965','10-December-1989','Doctor','harneetkaur@gmail.com','Sangrur',200000,NULL,2);

insert into employee values(104,'Geeta','10-December-1987','12-June-2014','Nurse','geeta@gmail.com','Patiala',27000,NULL,2);

insert into employee values(105,'Dr Shiya','20-March-1969','12-November-1987','Doctor','shiyamer@gmail.com','Patiala',250000,NULL,3);

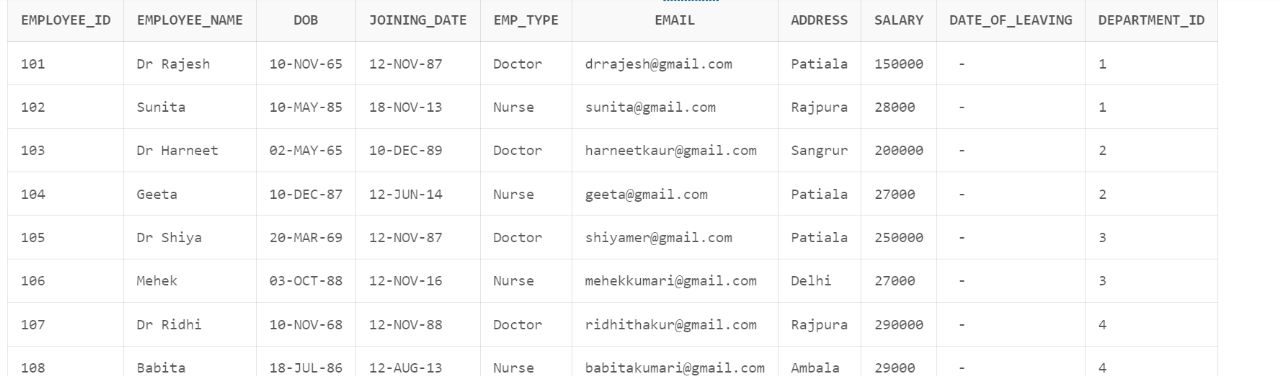
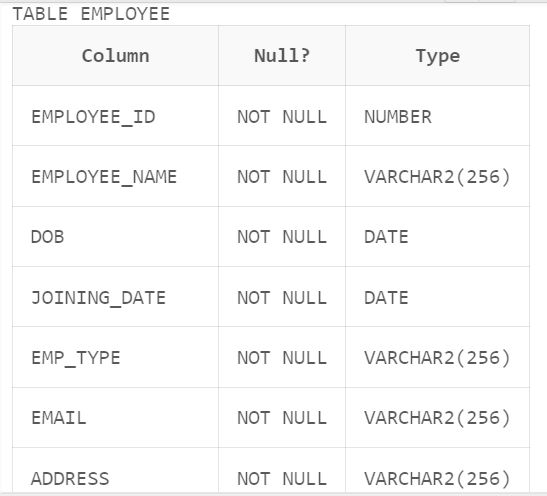
insert into employee values(106,'Mehek','3-October-1988','12-November-2016','Nurse','mehekkumari@gmail.com','Delhi',27000,NULL,3);

insert into employee values(107,'Dr Ridhi','10-November-1968','12-November-1988','Doctor','ridhithakur@gmail.com','Rajpura',290000,NULL,4);

insert into employee values(108,'Babita','18-July-1986','12-August-2013','Nurse','babitakumari@gmail.com','Ambala',29000,NULL,4);

insert into employee values(109,'Dr Ira','10-February-1963','12-November-1984','Doctor','iragupta@gmail.com','Sunam',270000,NULL,5);

insert into employee values(110,'Swati','10-June-1984','12-july-2017','Nurse','swatisingla@gmail.com','Sangrur',25000,NULL,5);



**CREATION AND INSERTION OF PATIENT TABLE :**

create table patient(

patient\_id int primary key,

patient\_name varchar(256) not null,

phone\_no varchar(256) not null,

blood\_group varchar(256) not null,

email varchar(256) not null,

gender varchar(256) not null,

address varchar(256) not null,

disease varchar(256) not null,

arrival\_date date not null,

discharge\_date date not null

);

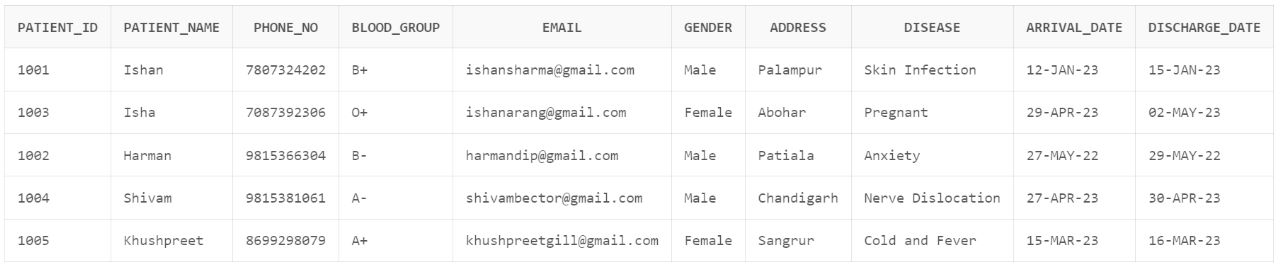
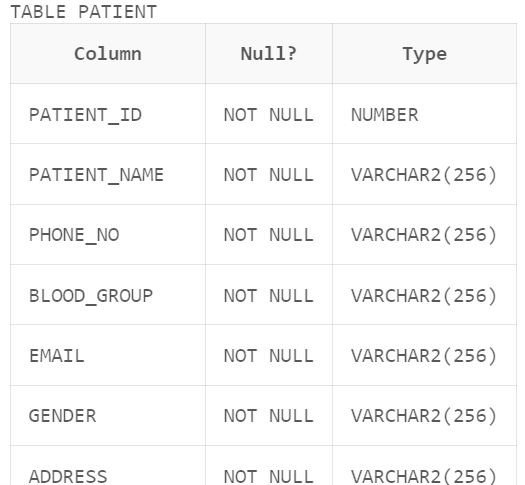
insert into patient values(1001,'Ishan','7807324202','B+','ishansharma@gmail.com','Male','Palampur','Skin Infection','12-January-2023','15-January-2023');

insert into patient values(1003,'Isha','7087392306','O+','ishanarang@gmail.com','Female','Abohar','Pregnant','29-April-2023','02-May-2023');

insert into patient values(1002,'Harman','9815366304','B-','harmandip@gmail.com','Male','Patiala','Anxiety','27-May-2022','29-May-2022');

insert into patient values(1004,'Shivam','9815381061','A-','shivambector@gmail.com','Male','Chandigarh','Nerve Dislocation','27-April-2023','30-April-2023');

insert into patient values(1005,'Khushpreet','8699298079','A+','khushpreetgill@gmail.com','Female','Sangrur','Cold and Fever','15-March-2023','16-March-2023');



**CREATION AND INSERTION OF MEDICINE TABLE :**

create table medicine(

medicine\_id int primary key,

medicine\_name varchar(256) not null,

quantity int not null,

med\_date date not null,

medicine\_cost int not null,

patient\_id int,

foreign key(patient\_id) references patient(patient\_id)

);,'JG123','AM1994','H123','Theodore','Chasseriau','France');

INSERT INTO ARTIST VALUES('ART10','NG123','PM1988','G123','Leonardo','da Vinci','Italy');

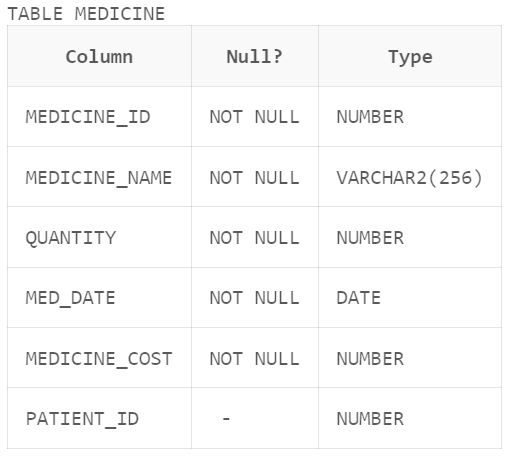
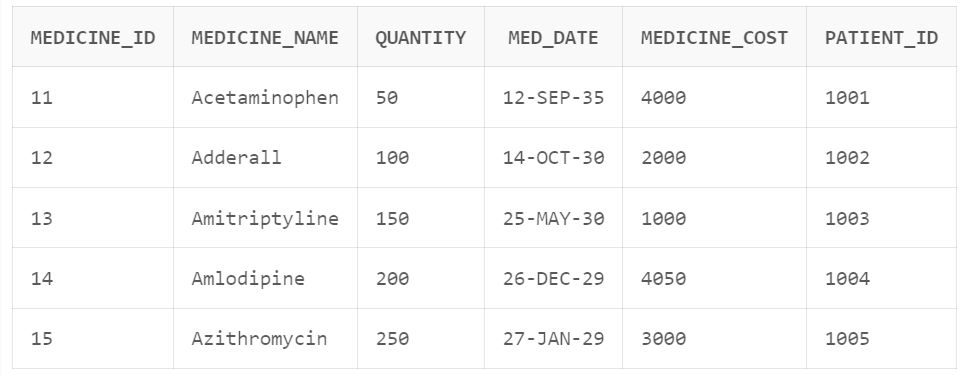
insert into medicine values(11,'Acetaminophen',50,'12-september-2035',4000,1001);

insert into medicine values(12,'Adderall',100,'14-october-2030',2000,1002);

insert into medicine values(13,'Amitriptyline',150,'25-May-2030',1000,1003);

insert into medicine values(14,'Amlodipine',200,'26-December-2029',4050,1004);

insert into medicine values(15,'Azithromycin',250,'27-January-2029',3000,1005);



**CREATION AND INSERTION OF BILL TABLE**

create table bill(

payment\_id int primary key,

bill\_date date not null,

room\_cost int not null,

other\_charges number(3) default null,

mcost int not null,

Total int not null,

patient\_id int,

foreign key(patient\_id) references patient(patient\_id)

);

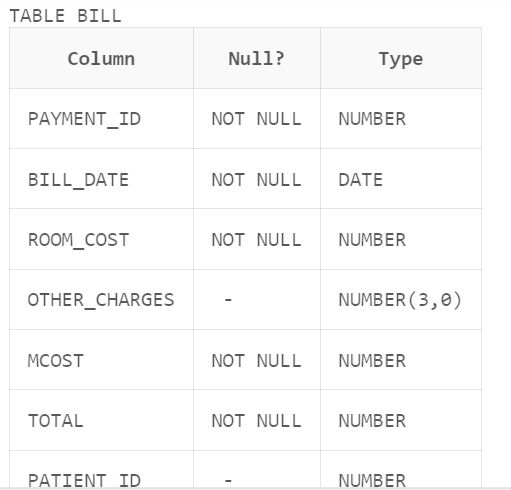
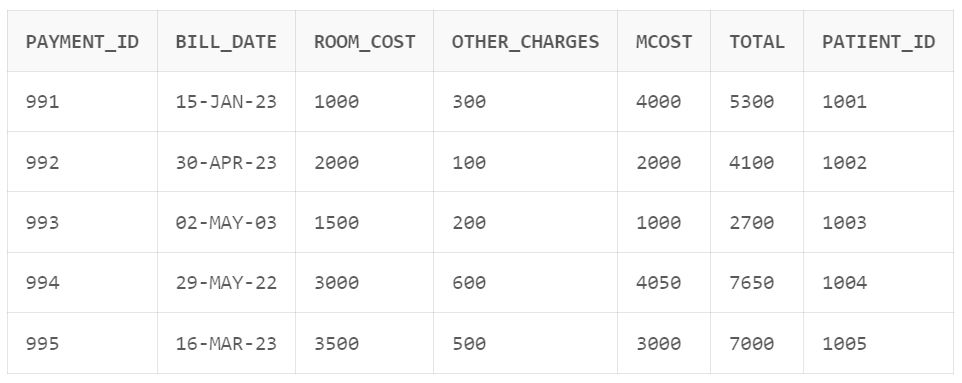
insert into bill values(991,'15-January-2023',1000,300,4000,5300,1001);

insert into bill values(992,'30-April-2023',2000,100,2000,4100,1002);

insert into bill values(993,'02-May-2003',1500,200,1000,2700,1003);

insert into bill values(994,'29-May-2022',3000,600,4050,7650,1004);

insert into bill values(995,'16-March-2023',3500,500,3000,7000,1005);



**CREATION AND INSERTION OF DOCTOR TABLE :**

create table doctor(

doctor\_id int primary key,

qualification varchar(50) not null,

patient\_id int,

foreign key(patient\_id) references patient(patient\_id),

employee\_id int,

foreign key(employee\_id) references employee(employee\_id)

);

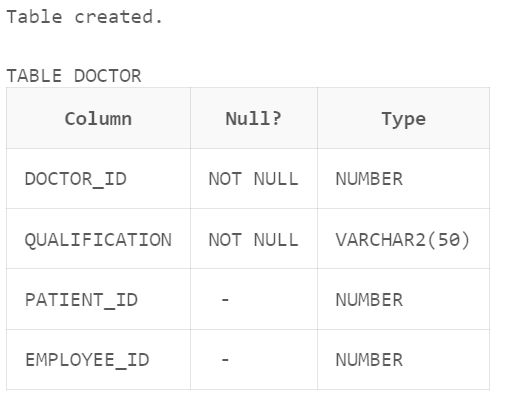
insert into doctor values(111,'MBBS MD Dermatology',1001,101);

insert into doctor values(112,'MBBS MD Gynaecology',1002,102);

insert into doctor values(113,'MBBS MD Psychology',1003,103);

insert into doctor values(114,'MBBS MD Neurology',1004,104);

insert into doctor values(115,'MBBS MD Paediatric',1005,105);



**CREATION AND INSERTION OF NURSE TABLE :**

create table nurse(

nurse\_id int Primary key,

patient\_id int,

foreign key(patient\_id) references patient(patient\_id),

employee\_id int,

foreign key(employee\_id) references employee(employee\_id)

);

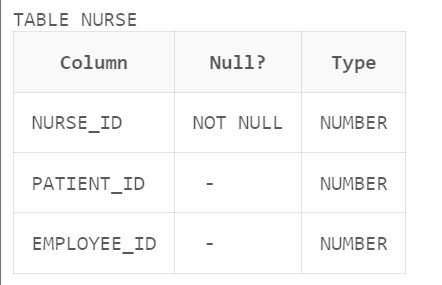
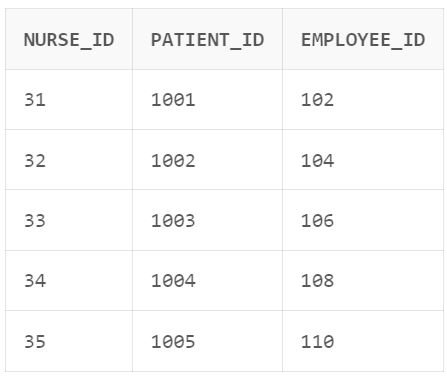
insert into nurse values(31,1001,102);

insert into nurse values(32,1002,104);

insert into nurse values(33,1003,106);

insert into nurse values(34,1004,108);

insert into nurse values(35,1005,110);



**CREATION AND INSERTION OF RELETIVE TABLE :**

create table reletive(

reletive\_id int Primary key,

reletive\_name varchar(50) Not null,

reletive\_contact varchar(50) not null, relation varchar(50) not null,

patient\_id int,

foreign key(patient\_id) references patient(patient\_id)

);

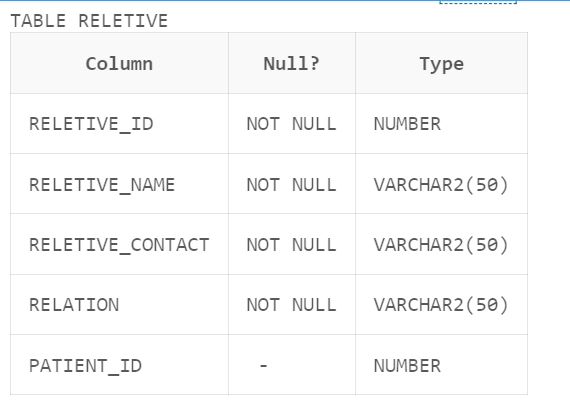
insert into reletive values(21,'Manvir','9856232652','Friend',1001);

insert into reletive values(23,'Swastik','9812332652','Husband',1002);

insert into reletive values(24,'Gunavri','9812342652','Wife',1003);

insert into reletive values(22,'Shivangi','9853232652','Sister',1004);

insert into reletive values(25,'Jovan','98568692652','Brother',1005);



**CREATION AND INSERTION OF ROOM TABLE :**

create table room(

room\_id int primary key,

room\_type varchar(30) not null,

patient\_id int,

foreign key(patient\_id) references patient(patient\_id),

room\_cost int not null

);

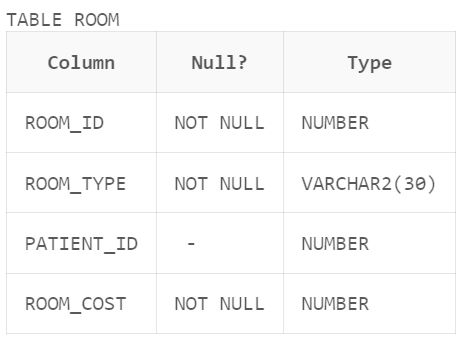
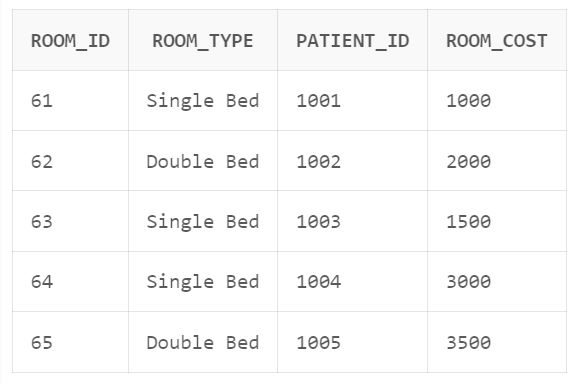
insert into room values(61,'Single Bed',1001,1000);

insert into room values(62,'Double Bed',1002,2000);

insert into room values(63,'Single Bed',1003,1500);

insert into room values(64,'Single Bed',1004,3000);

insert into room values(65,'Double Bed',1005,3500);



**CREATION AND INSERTION OF TEST TABLE :**

create table test(

test\_id int Primary key,

test\_name Varchar(60) Not null,

test\_cost int Not null,

date1 date not null,

patient\_id int,

foreign key(patient\_id) references patient(patient\_id)

);

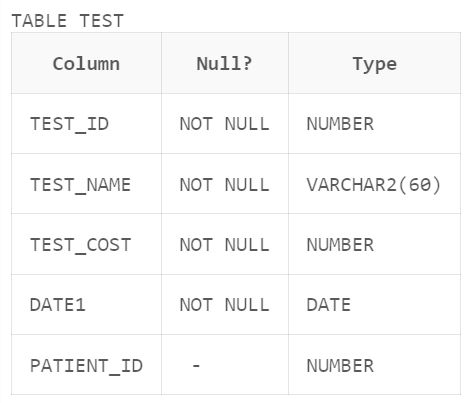
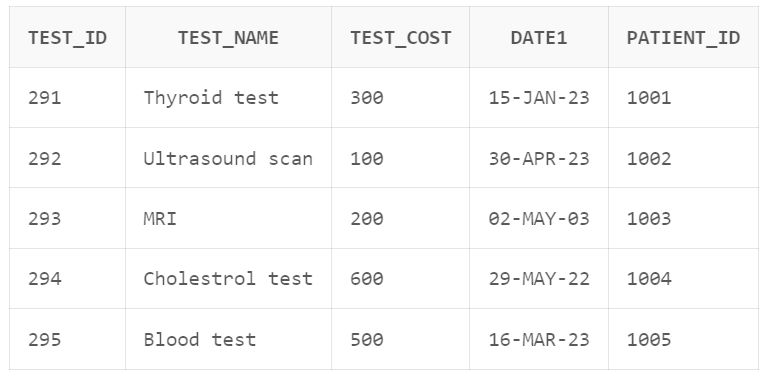
insert into test values(291,'Thyroid test',300,'15-January-2023',1001);

insert into test values(292,'Ultrasound scan',100,'30-April-2023',1002);

insert into test values(293,'MRI',200,'02-May-2003',1003);

insert into test values(294,'Cholestrol test',600,'29-May-2022',1004);

insert into test values(295,'Blood test',500,'16-March-2023',1005);



1. **TRIGGERS :**

**In DEPARTMENT :**

CREATE OR REPLACE TRIGGER trg\_department

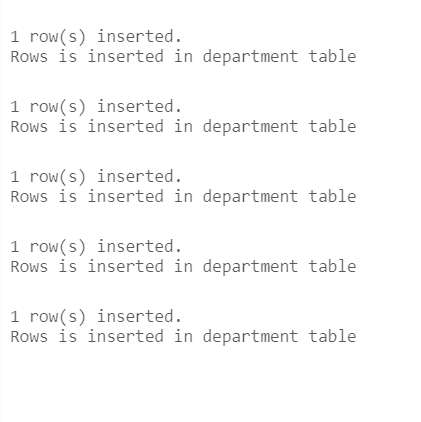
AFTER INSERT ON department

FOR EACH ROW

BEGIN

dbms\_output.put\_line('Rows is inserted in department table');

END;



**In employee :**

CREATE OR REPLACE TRIGGER trg\_emp

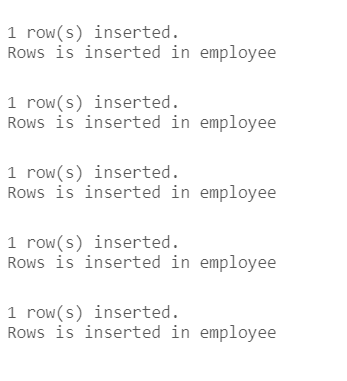
AFTER INSERT ON employee

FOR EACH ROW

BEGIN

dbms\_output.put\_line('Rows is inserted in employee');

END;



**In Patient :**

CREATE OR REPLACE TRIGGER trg\_patient

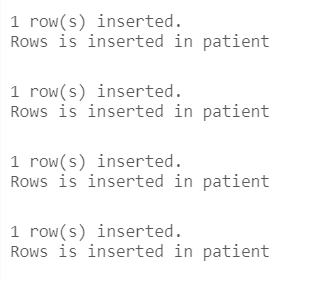
AFTER INSERT ON patient

FOR EACH ROW

BEGIN

dbms\_output.put\_line('Rows is inserted patient');

END;



**In Medicine :**

CREATE OR REPLACE TRIGGER trg\_medicine

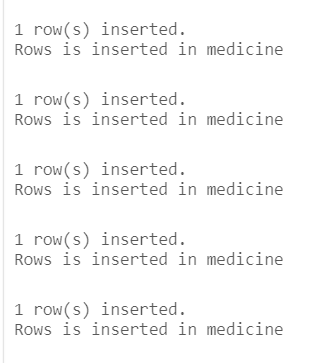
AFTER INSERT ON medicine

FOR EACH ROW

BEGIN

dbms\_output.put\_line('Rows is inserted in medicine');

END;



**In Bill :**

CREATE OR REPLACE TRIGGER trg\_bill

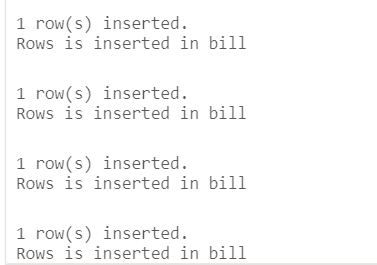
AFTER INSERT ON bill

FOR EACH ROW

BEGIN

dbms\_output.put\_line('Rows is inserted in bill');

END;



**In doctor :**

CREATE OR REPLACE TRIGGER trg\_doctor

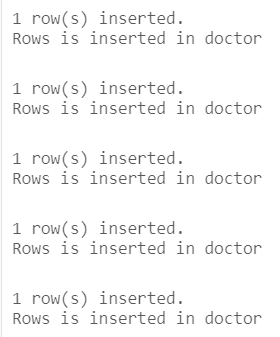
AFTER INSERT ON doctor

FOR EACH ROW

BEGIN

dbms\_output.put\_line('Rows is inserted in doctor');

END;



**In Nurse :**

CREATE OR REPLACE TRIGGER trg\_nurse

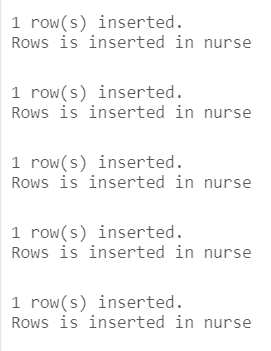
AFTER INSERT ON nurse

FOR EACH ROW

BEGIN

dbms\_output.put\_line('Rows is inserted in nurse');

END;



**In Relatives :**

CREATE OR REPLACE TRIGGER trg\_reletive

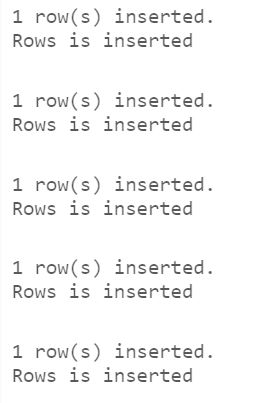
AFTER INSERT ON reletive

FOR EACH ROW

BEGIN

dbms\_output.put\_line('Rows is inserted');

END;



**In Room :**

CREATE OR REPLACE TRIGGER trg\_room

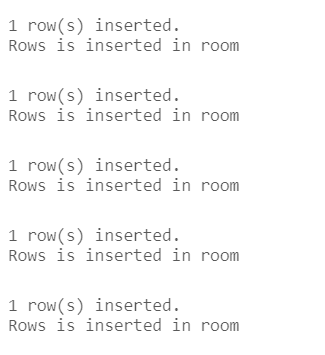
AFTER INSERT ON room

FOR EACH ROW

BEGIN

dbms\_output.put\_line('Rows is inserted in room');

END;



**In Test :**

CREATE OR REPLACE TRIGGER trg\_test

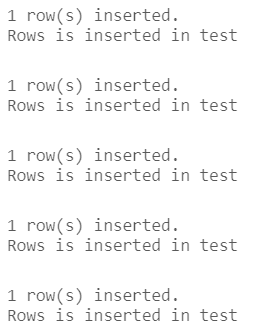
AFTER INSERT ON test

FOR EACH ROW

BEGIN

dbms\_output.put\_line('Rows is inserted in test');

END;



**C ) Exceptions :**

Exception handling when uniqueness property of primary key is violated:

BEGIN

INSERT INTO department(department\_id, dep\_name, dep\_head)

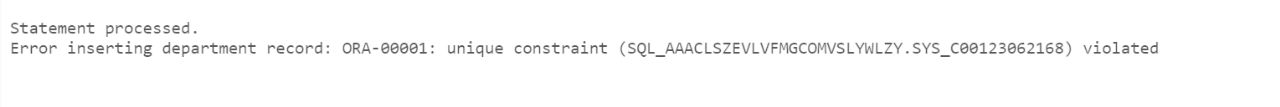
VALUES(1, 'Marketing', 'John Smith');

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Error inserting department record: ' || SQLERRM);

END;

****

5.3.2 Exception handling when no data is found:

BEGIN

UPDATE employee

SET salary = 50000

WHERE employee\_id = 16;

EXCEPTION

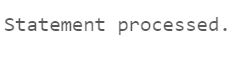
WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Employee record not found');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Error updating employee record: ' || SQLERRM);

END;

****

5.3.3 Exception handling when no data is found or any other exception:

BEGIN

DELETE FROM patient

WHERE patient\_id = 1;

EXCEPTION

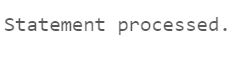
WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Patient record not found');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Error deleting patient record: ' || SQLERRM);

END;

****

**D) Cursors :**

Cursor to show the details of all the employees:

DECLARE

CURSOR C1 IS SELECT \* FROM employee order by salary desc ;

rec employee%rowtype;

begin

FOR rec in C1 loop

dbms\_output.put\_line(rec.employee\_id||' '||rec.employee\_name||' '||rec.emp\_type||' '||rec.email||' '||rec.salary);

end loop;

END;



Cursor to show the details of all the employees who are doctors:

DECLARE

CURSOR C1 IS SELECT \* FROM employee where emp\_type='Doctor';

rec employee%rowtype;

begin

FOR rec in C1 loop

dbms\_output.put\_line(rec.employee\_id||' '||rec.employee\_name||' '||rec.emp\_type||' '||rec.email||' '||rec.salary);

end loop;

END;



Cursor to show the details of all the employees who are nurses:

DECLARE

CURSOR C1 IS SELECT \* FROM employee where emp\_type='Nurse';

rec employee%rowtype;

begin

FOR rec in C1 loop

dbms\_output.put\_line(rec.employee\_id||' '||rec.employee\_name||' '||rec.emp\_type||' '||rec.email||' '||rec.salary);

end loop;

END;



Cursor to show which employee works in which department:

DECLARE

CURSOR emp\_dept\_cur IS

SELECT e.employee\_name, d.dep\_name

FROM employee e

JOIN department d ON e.department\_id = d.department\_id;

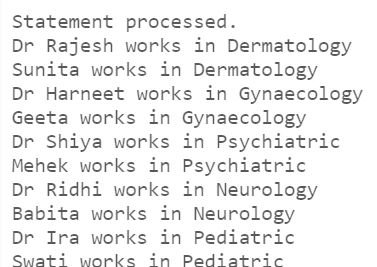
BEGIN

FOR emp\_dept\_rec IN emp\_dept\_cur LOOP

DBMS\_OUTPUT.PUT\_LINE(emp\_dept\_rec.employee\_name || ' works in ' || emp\_dept\_rec.dep\_name);

END LOOP;

END;



**E ) Constraints :**

ALTER TABLE employee ADD CONSTRAINT chk\_salary CHECK (salary > 0);

ALTER TABLE patient ADD CONSTRAINT chk\_blood\_group CHECK (

blood\_group IN ('A+', 'A-', 'B+', 'B-', 'O+', 'O-', 'AB+', 'AB-')

);

ALTER TABLE medicine ADD CONSTRAINT chk\_medicine\_cost CHECK (medicine\_cost >= 0);

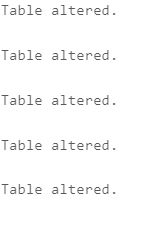
ALTER TABLE room ADD CONSTRAINT chk\_room\_cost CHECK (room\_cost > 0);

ALTER TABLE bill ADD CONSTRAINT chk\_total CHECK (Total > 0);

ALTER TABLE employee ADD CONSTRAINT chk\_joining\_date CHECK (

date\_of\_leaving IS NULL OR joining\_date < date\_of\_leaving

);



select \* from user\_constraints where table\_name='EMPLOYEE';

select \* from user\_constraints where table\_name='DEPARTMENT';

select \* from user\_constraints where table\_name='PATIENT';

select \* from user\_constraints where table\_name='MEDICINE';

select \* from user\_constraints where table\_name='BILL';

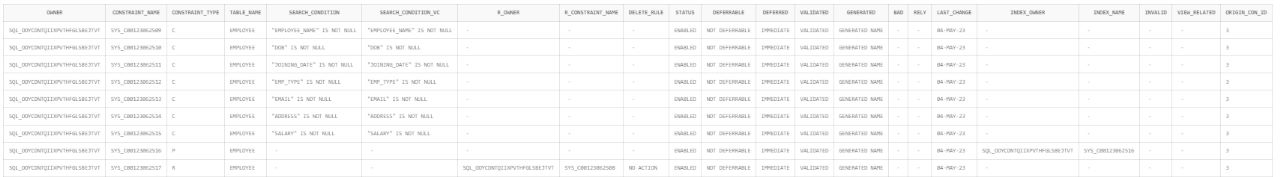
select \* from user\_constraints where table\_name='DOCTOR';

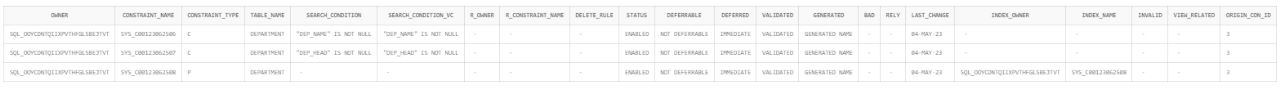
select \* from user\_constraints where table\_name='NURSE';

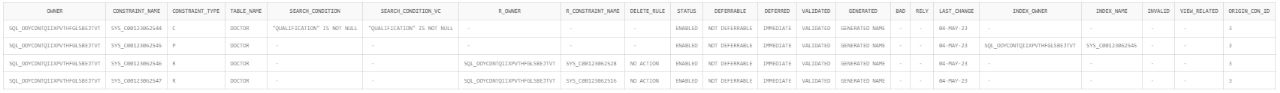
select \* from user\_constraints where table\_name='RELETIVE';

select \* from user\_constraints where table\_name='ROOM';

select \* from user\_constraints where table\_name='TEST';







**F ) Functions and Procedures:**

Procedure to insert a new department record into the department table:

CREATE OR REPLACE PROCEDURE insert\_department(department\_id IN INT, dep\_name IN VARCHAR2, dep\_head IN VARCHAR2) AS

BEGIN

INSERT INTO department VALUES (department\_id, dep\_name, dep\_head);

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Department record inserted successfully!');

EXCEPTION

WHEN OTHERS THEN

ROLLBACK;

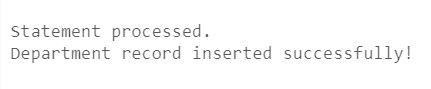
DBMS\_OUTPUT.PUT\_LINE('Error inserting department record: ' || SQLERRM);

END;

begin

insert\_department(6,'Opthalamology','Shivani');

end;



Function to calculate the total salary paid to employees in a particular department:

CREATE OR REPLACE FUNCTION get\_department\_salary(department\_id IN INT) RETURN INT AS

total\_salary INT;

BEGIN

SELECT SUM(salary) INTO total\_salary

FROM employee

WHERE department\_id = department\_id;

RETURN total\_salary;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('No employees found in department');

RETURN 0;

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Error calculating department salary: ' || SQLERRM);

RETURN 0;

END;

declare

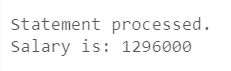
sal number;

begin

sal:=get\_department\_salary(6);

dbms\_output.put\_line('Salary is: '||sal);

end;



**7. QUERIES**

List all the patients who are currently admitted to the hospital:

SELECT \* FROM patient WHERE discharge\_date IS NULL;



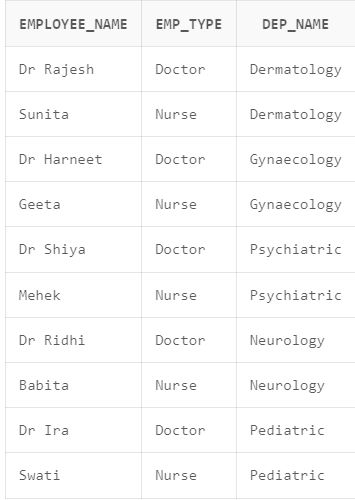
List all the doctors and nurses along with the department they belong to:

SELECT e.employee\_name, e.emp\_type, d.dep\_name

FROM employee e

JOIN department d ON e.department\_id = d.department\_id

WHERE e.emp\_type IN ('Doctor', 'Nurse');



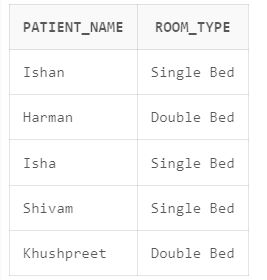
List all the patients along with the room they are currently occupying:

SELECT p.patient\_name, r.room\_type

FROM patient p

JOIN room r ON p.patient\_id = r.patient\_id

WHERE r.patient\_id IS NOT NULL;

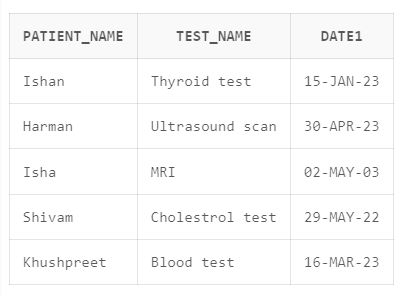


List all the patients along with the tests they have undergone:

SELECT p.patient\_name, t.test\_name, t.date1

FROM patient p

JOIN test t ON p.patient\_id = t.patient\_id;

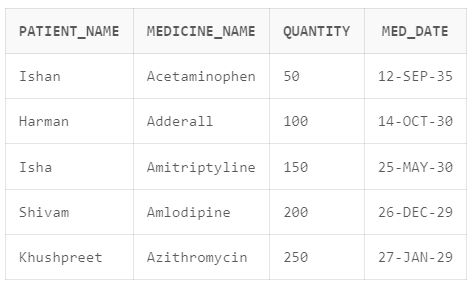


List all the patients along with the medicines they have been prescribed:

SELECT p.patient\_name, m.medicine\_name, m.quantity, m.med\_date

FROM patient p

JOIN medicine m ON p.patient\_id = m.patient\_id;

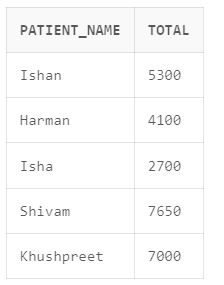


List all the patients along with their total bill amount:

SELECT p.patient\_name, b.Total

FROM patient p

JOIN bill b ON p.patient\_id = b.patient\_id;



List all the relatives of a particular patient:

SELECT \* FROM reletive WHERE patient\_id = 1001;



**8.CONCLUSION**

A hospital management system (HMS) is a software application designed to streamline the administration and management of healthcare facilities such as hospitals, clinics, and medical centers. The system can automate various administrative and operational tasks such as patient registration, appointment scheduling, billing and payment processing, patient record management, inventory management, and staff management.  
  
In conclusion, a hospital management system is an essential tool for modern healthcare facilities. It can help hospitals and clinics to improve their overall efficiency, reduce administrative costs, enhance patient care and satisfaction, and increase profitability. With the integration of advanced technologies such as artificial intelligence and machine learning, hospital management systems can provide even more sophisticated solutions for healthcare providers. As healthcare continues to evolve, hospital management systems will play a crucial role in shaping the future of healthcare delivery.