Birla Institute of Technology and Science, Pilani

CS F212

Database Systems

HOSPITAL MANAGEMENT SYSTEM

DOCUMENTATION

Submitted to: Dr. Amit Dua & Parth Patel

Submitted by:

Rikhil Gupta (2021A7PS0533P)

Nachiketh S Shastry (2021A7PS2686P)

Introduction:

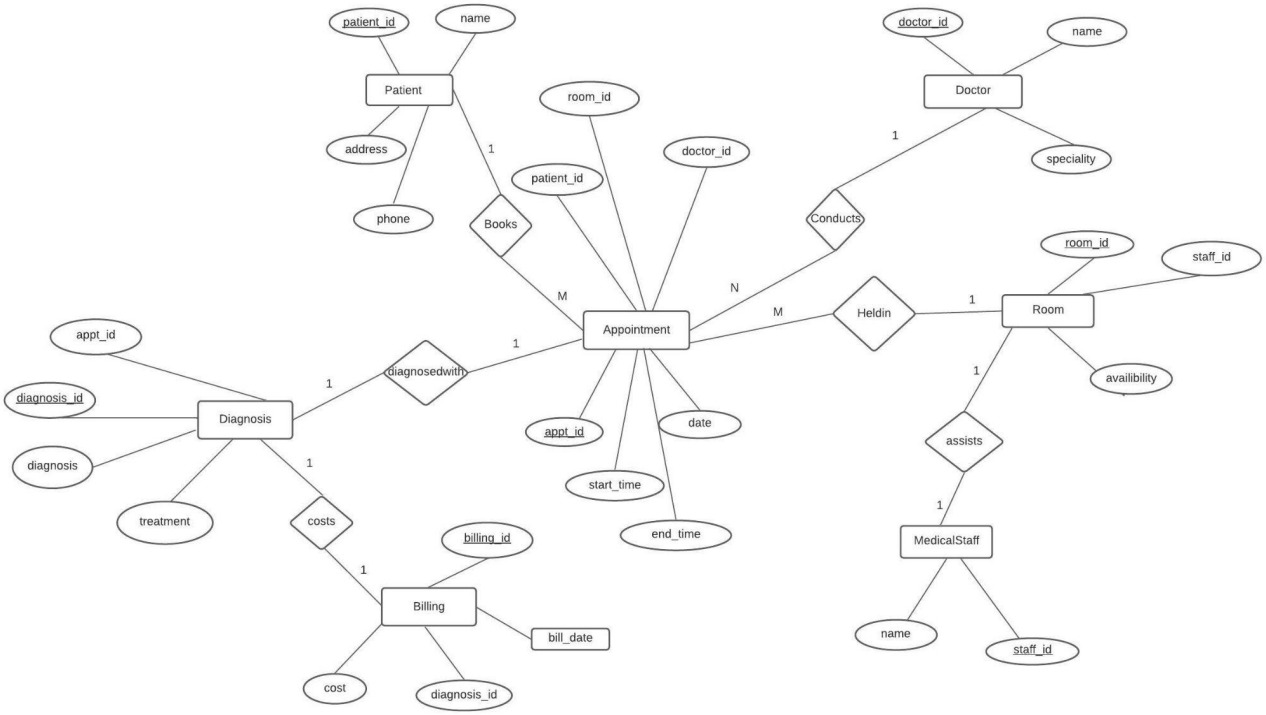
This is a robust database management project which records and facilitates regular day to day activities of a hospital. This schema has multiple tables, each responsible for storing a specific aspect of the functioning in the hospital. For instance, the patient table holds the details of patients etc. All these tables, along with their relations with one another and their attributes are described in a figure called Entity Relationship diagram, which is shown below.

As expected, we have recorded videos of us presenting the project, and the drive links for the video recordings are:

1. **Rikhil Gupta:**
   https://drive.google.com/file/d/1XiLbxuNFbR1GS8MQNVW2wlR30JnZjD22/view?usp=share_link

2. **Nachiketh S Shastry:**
   https://drive.google.com/file/d/1MGUdCe0O6bga_akpC4BhW2B2ghpsokdq/view?usp=share_link

ER Diagram:



All the Entities with their attributes and a brief description about each in the above ER diagram:

1. **Patient**: patient_id(primary key, int), name(varchar), address(varchar), phone(varchar)

This table holds the patient details.

2. **Doctor**: doctor_id(primary key, int), name(varchar), speciality(varchar)

This table holds the doctor details.

3. **MedicalStaff**: staff_id(primary key, int), name(varchar)

This table holds the details of the medical staff in the hospital.

4. **Room**: room_id(primary key, int), staff_id(Foreign key, int), availability(Bool)

This table holds details of the rooms available in the hospital.

5. **Diagnosis**: diagnosis_id(primary key, int), appt_id(Foreign key, int), patient_id(Foreign key, int), diagnosis(varchar), treatment(varchar)

This table holds details of diagnosis. A diagnosis is done and the disease as well as the treatment is recognized after an appointment and stored in this table.

6. **Billing**: billing_id(primary key, int), bill_date(DATE), patient_id(Foreign key, int), diagnosis_id(Foreign key, int), cost(DECIMAL(10,2))

This table holds the details of the cost of each diagnosis and its corresponding treatment. Patient_id tells which patient owes that money to the hospital.

7. **Appointment**: appt_id(primary key, int), patient_id(Foreign key, int), room_id(Foreign key, int), doctor_id(Foreign key, int), start_time(DATETIME), end_time(DATETIME), date(DATE)

This table stores the details of each appointment created. An appointment needs to have a patient, a specialized doctor, and a room with a medical staff available in it. An appointment runs from start_time till end_time during the date.

All the relationships in the ER diagram explained:

1. **Books**: 1:M relation which indicates that a patient can book multiple appointments, but each appointment can be booked by only one patient.

2. **Conducts**: 1:N relation which indicates that a doctor can conduct multiple appointments, but each appointment can be conducted by only one doctor.

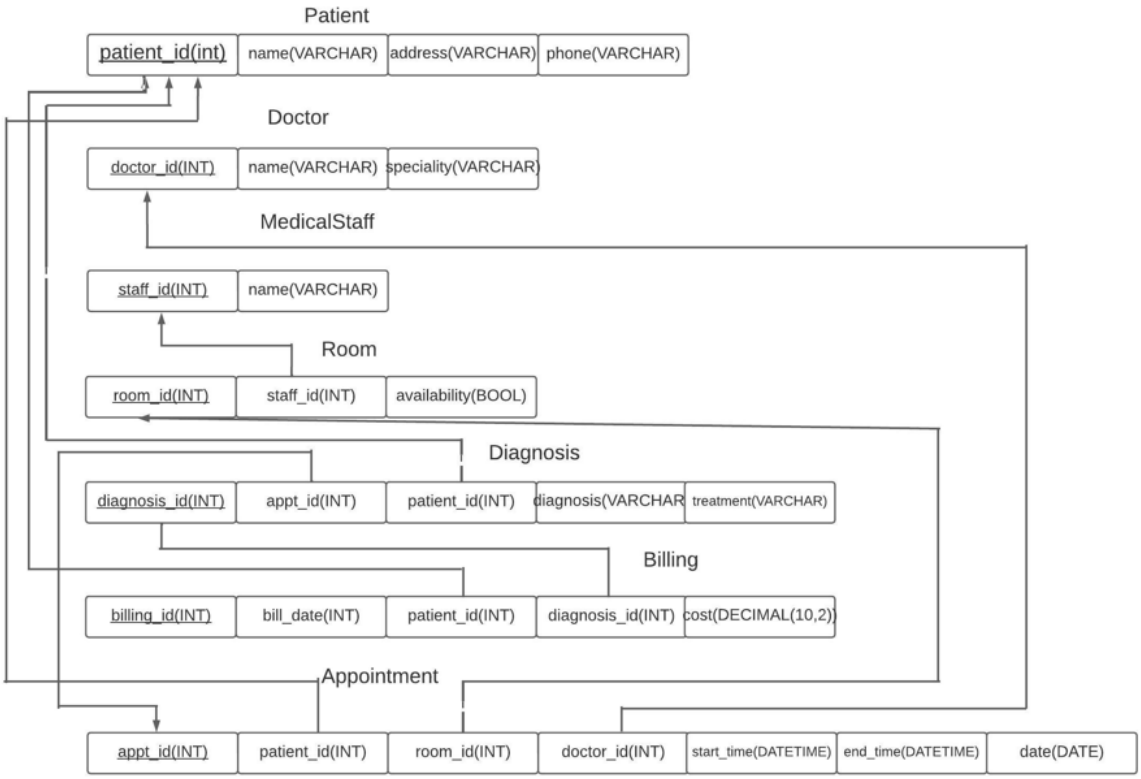
3. **Heldin**: 1:M relation which indicates that a room can hold multiple appointments at different intervals of time, but each appointment can be held in only one room.

4. **Assists**: 1:1 relation which indicates that one medical staff can be present in only one room, and each room can have only one medical staff which assists the doctor in the appointment.

5. **Diagnosedwith**: 1:1 relation which indicates that in each appointment a patient can be diagnosed with one disease and its corresponding treatment.

6. **Costs**: 1:1 relation which relates the cost of a particular diagnosis.

Relational Schema



Normalization:

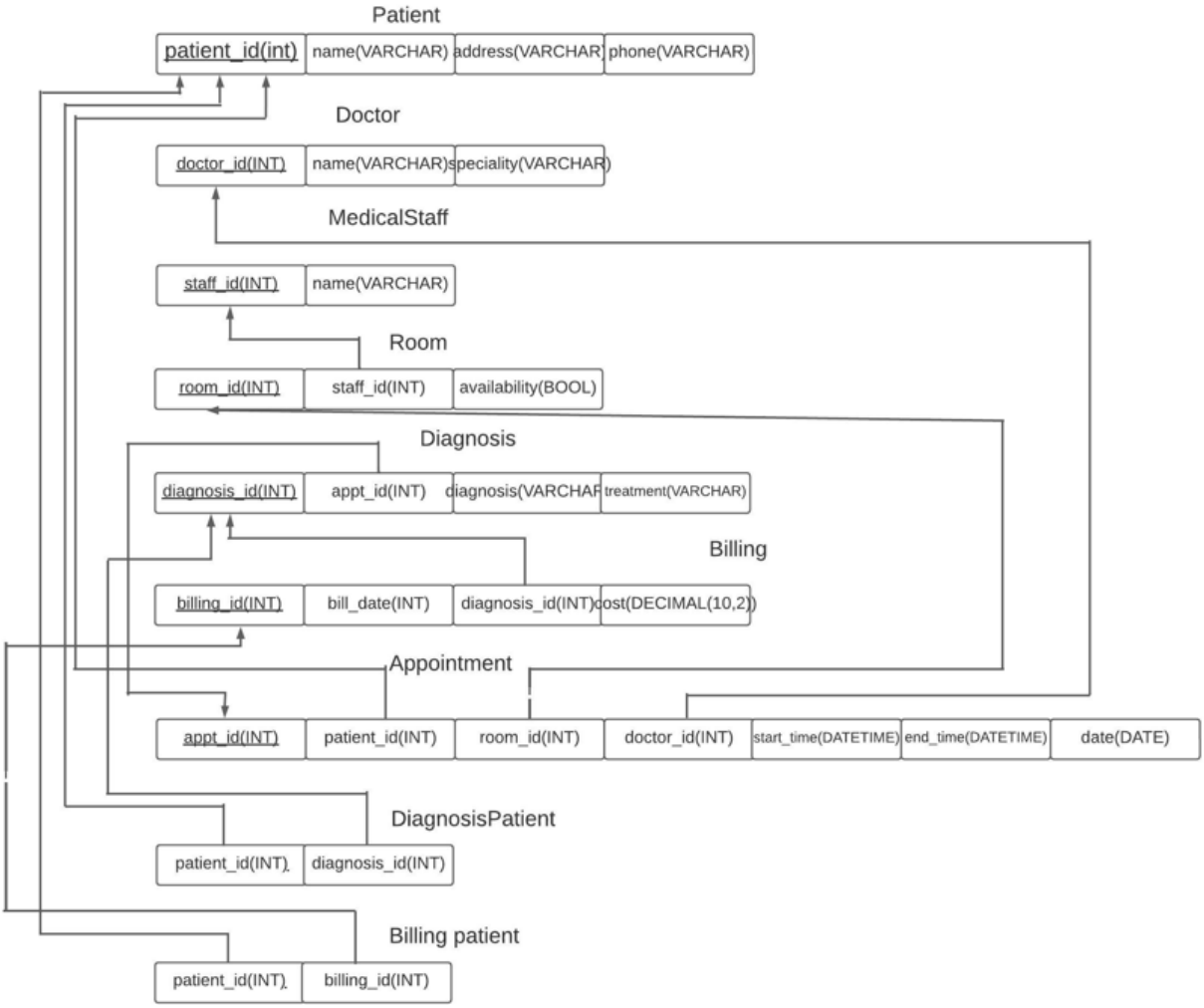
Patient, Doctor, Room, MedicalStaff, and Appointment are already in 3NF since they have no repeating groups or partial dependencies.

Diagnosis has a partial dependency on Patient since diagnosis and treatment are dependent on appt_id which in turn is dependent on patient_id. So we split it into Diagnosis and DiagnosisPatient tables to remove the partial dependency.

Billing has a partial dependency on Patient since cost and bill_date are dependent on diagnosis_id which in turn is dependent on patient_id. So we split it into Billing and BillingPatient tables to remove the partial dependency.

This schema should allow you to store and retrieve information about patients, doctors, appointments, diagnoses, and billing records in a healthcare system in a way that is free of data redundancy and anomalies.

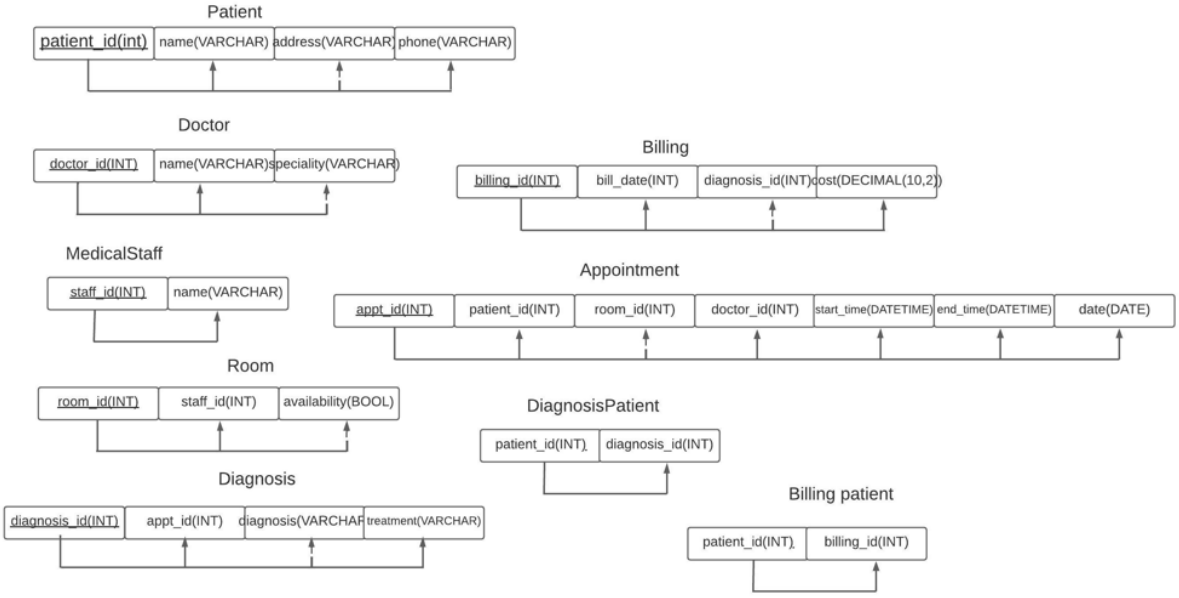
Normalized Schema



Tables:

1. Patient - patient_id is primary key, No foreign key
2. Doctor - doctor_id is primary key, No foreign key
3. MedicalStaff - staff_id is primary key, No foreign key
4. Room - room_id is primary key, staff_id is foreign key
5. Diagnosis - diagnosis_id is primary key, appt_id is foreign key
6. Billing - billing_id is primary key, diagnosis_id is foreign key
7. Appointment - appt_id is primary key; patient_id, room_id, doctor_id is foreign key
8. DiagnosisPatient - patient_id and diagnosis_id both are foreign keys
9. BillingPatient - patient_id and billing_id both are foreign keys

Functional Dependencies(3NF):



The following call functions are executed one by one and the output is shown after each call function in order

```
use  hospital_database;

DELIMITER //
//

CALL InsertPatient('Rikhil Gupta', 'Elec City Blr', '123-456');
//
CALL InsertDoctor('Nachiket Shastry', 'Cardiology');
//
CALL InsertMedicalStaff('Manas');
//
CALL UpdateRoomStatus(1, 1);

-- takes room_id as input and allots a medical staff to it
//
CALL AllotFirstAvailableStaffToRoom(2);
CALL AllotFirstAvailableStaffToRoom(1);
CALL AllotFirstAvailableStaffToRoom(4);
CALL AllotFirstAvailableStaffToRoom(5);
CALL AllotFirstAvailableStaffToRoom(6);
CALL AllotFirstAvailableStaffToRoom(3);
CALL AllotFirstAvailableStaffToRoom(7);
//
```

```
-- takes room_id as input and frees a staff from that room
//
CALL FreeStaffFromRoom(1);
//

-- takes patient_id ,specialisation required, date, start_time and end_time as input for booking ar
//
CALL makeappointment(1,'Cardiology','2023-08-01', '2023-08-01 11:00:00', '2023-08-01 12:00:00');
CALL makeappointment(3,'Neurology','2023-09-01', '2023-09-01 04:00:00', '2023-09-01 05:00:00');
CALL makeappointment(7,'Endocrinology','2023-10-01', '2023-10-01 07:00:00', '2023-10-01 08:00:00');
CALL makeappointment(3,'Oncology','2023-09-01', '2023-09-01 05:00:00', '2023-09-01 06:00:00');
//



-- takes apointment_id, new_date, new_start_time,new_end_time
//
CALL rescheduleAppointment(1, '2023-08-01', '2023-08-01 14:00:00', '2023-08-01 15:00:00');
//

-- takes appointment_id, diagnosis, treatment as inputs for that particular appointment and updates
//
```

```
CALL diagnose(1,'CANCER','Chemo');
CALL diagnose(3,'Malaria','BesRest');
CALL diagnose(2,'AIDS','NoCure');
CALL diagnose(2,'AIDS','NoCure');
//
CALL diagnose(4,'Fever','Dolo');

//

-- takes diagnosis_id and cost as input inserts billing
//
CALL insert_billing(1, 100);
CALL insert_billing(2, 25);
//

CREATE TRIGGER add_gst_to_billing
BEFORE INSERT ON Billing
FOR EACH ROW
BEGIN
  SET NEW.cost = NEW.cost * 1.18;
END;

//
CALL insert_billing(3, 50);
```

```
-- takes patient_id as input and generates that patients bill
//

CALL generate_bill(3);
//



-- takes name of doctor as input and displays his earnings
//

CALL total_earnings_doctor('Dr. Sanah Sheik');
//
CALL total_earnings_doctor('Dr. Gabriel Joe');
//
CALL total_earnings_doctor('Nachiket Shastry');
//



-- takes specialisation as input and displays it's earnings
//
CALL total_earnings_specialization('Cardiology');
//
CALL total_earnings_specialization('Endocrinology');
//
CALL total_earnings_specialization('Neurology');
//
```

```
103
104
105
106     -- takes name of doctor as input and displays the patients
107     //
108     CALL find_patients_for_doctor('Dr. Tarak');
109     //
110     CALL find_patients_for_doctor('Dr. Siddhardh');
111     //
112     CALL find_patients_for_doctor('Dr. Sanah Sheik');
113     //
114     CALL find_patients_for_doctor('Nachiket Shastry');
115     //
116
117
118     -- takes name of patient as input and displays the patient's previous medical records
119     //
120     CALL fetch_previous_diagnoses('Kshitish');
121     //
122     CALL fetch_previous_diagnoses('Nikesh');
123     //
124     CALL fetch_previous_diagnoses('Sanshrav');
125     //
126     CALL fetch_previous_diagnoses('Aryan');
127
128     //
129
130
```

Creation of the schema that includes all the tables starts from here;

```sql
create schema if not exists Hospital_Database;
use Hospital_Database;

CREATE TABLE Patient (
  patient_id INT AUTO_INCREMENT PRIMARY KEY,
  name VARCHAR(255),
  address VARCHAR(255),
  phone VARCHAR(20)
);

CREATE TABLE Doctor (
  doctor_id INT auto_increment PRIMARY KEY,
  name VARCHAR(255),
  speciality VARCHAR(255)
);

CREATE TABLE MedicalStaff (
  staff_id INT auto_increment PRIMARY KEY,
  name VARCHAR(255)
);

CREATE TABLE Room (
  room_id INT auto_increment PRIMARY KEY,
  staff_id INT,
  availability BOOLEAN,
  FOREIGN KEY (staff_id) REFERENCES MedicalStaff(staff_id)
);

```

```sql
CREATE TABLE Appointment (
  appt_id INT auto_increment PRIMARY KEY,
  patient_id INT,
  room_id INT,
  doctor_id INT,
  start_time DATETIME,
  end_time DATETIME,
  date DATE,
  FOREIGN KEY (patient_id) REFERENCES Patient(patient_id),
  FOREIGN KEY (room_id) REFERENCES Room(room_id),
  FOREIGN KEY (doctor_id) REFERENCES Doctor(doctor_id)
);

CREATE TABLE Diagnosis (
  diagnosis_id INT auto_increment PRIMARY KEY,
  appt_id INT,
  diagnosis VARCHAR(255),
  treatment VARCHAR(255),
  FOREIGN KEY (appt_id) REFERENCES Appointment(appt_id)
);

CREATE TABLE DiagnosisPatient (
  patient_id INT,
  diagnosis_id INT,
  PRIMARY KEY (patient_id, diagnosis_id),
  FOREIGN KEY (patient_id) REFERENCES Patient(patient_id),
  FOREIGN KEY (diagnosis_id) REFERENCES Diagnosis(diagnosis_id)
);
```

```sql
CREATE TABLE Billing (
  billing_id INT auto_increment PRIMARY KEY,
  diagnosis_id INT,
  cost DECIMAL(10, 2),
  bill_date DATE,
  FOREIGN KEY (diagnosis_id) REFERENCES Diagnosis(diagnosis_id)
);

CREATE TABLE BillingPatient (
  patient_id INT,
  billing_id INT,
  PRIMARY KEY (patient_id, billing_id),
  FOREIGN KEY (patient_id) REFERENCES Patient(patient_id),
  FOREIGN KEY (billing_id) REFERENCES Billing(billing_id)
);

```

```sql
INSERT INTO Patient (name, address, phone)
VALUES
  ('Kshitish', 'Ghaziabad', '1234512345'),
  ('Aryan', '6th Street Pune', '4567845678'),
  ('Nikesh', '1st Sector Faridabad', '9876598765'),
  ( 'Sanshrav', '2nd Sector Faridabad', '3467934679'),
  ( 'Harshit', '3rd Sector Faridabad', '56342184723'),
  ( 'Mayank', '5th Street Gurgaon', '34565902284'),
  ( 'Munish', 'Ranchi', '5482927282'),
  ('Rishabh', '6th Sector Faridabad', '987654321'),
  ('Samarth', '6th Street Gurgaon', '4682738723'),
  ('Chirag', 'Dhanbad', '6483927483');

```

```sql
INSERT INTO Doctor ( name, speciality)
VALUES
  ('Dr. Tarak', 'Cardiology'),
  ('Dr. Siddhardh', 'Oncology'),
  ('Dr. Sanah Sheik', 'Neurology'),
  ('Dr. Srikruti', 'Pediatrics'),
  ('Dr. Malhar Patel', 'Dermatology'),
  ('Dr. Gabriel Joe', 'Endocrinology'),
  ('Dr. Nishant DMello', 'Gastroenterology'),
  ('Dr. Nishant K', 'Hematology'),
  ('Dr. Stuti Agarwal', 'Infectious Disease'),
  ('Dr. Jeevitha Reddy', 'Internal Medicine');

```

```sql

INSERT INTO MedicalStaff (name)
VALUES
  ('Ravi'),
  ('Priya'),
  ('Shruti'),
  ('Akhil'),
  ('Aryan'),
  ('Amandeep'),
  ( 'Rijul'),
  ('Aadeesh'),
  ( 'Aakash'),
  ('Pranav');

```

```
116 •   INSERT INTO Room (staff_id, availability)
117     VALUES
118       (NULL, TRUE),
119       (NULL, TRUE),
120       (NULL, TRUE),
121       (NULL, TRUE),
122       (NULL, TRUE),
123       (NULL, TRUE),
124       (NULL, TRUE),
125       (NULL, TRUE),
126       (NULL, TRUE),
127       (NULL, TRUE);
128
```

patient_id	name	address	phone
1	Kshitish	Ghaziabad	1234512345
2	Aryan	6th Street Pune	4567845678
3	Nikesh	1st Sector Faridabad	9876598765
4	Sanshrav	2nd Sector Faridabad	3467934679
5	Harshit	3rd Sector Faridabad	56342184723
6	Mayank	5th Street Gurgaon	34565902284
7	Munish	Ranchi	5482927282
8	Rishabh	6th Sector Faridabad	987654321
9	Samarth	6th Street Gurgaon	4682738723
10	Chirag	Dhanbad	6483927483
NULL	NULL	NULL	NULL

doctor_id	name	speciality
1	Dr. Tarak	Cardiology
2	Dr. Siddhardh	Oncology
3	Dr. Sanah Sheik	Neurology
4	Dr. Srikruti	Pediatrics
5	Dr. Malhar Patel	Dermatology
6	Dr. Gabriel Joe	Endocrinology
7	Dr. Nishant DMello	Gastroenterology
8	Dr. Nishant K	Hematology
9	Dr. Stuti Agarwal	Infectious Disease
10	Dr. Jeevitha Reddy	Internal Medicine
NULL	NULL	NULL

staff_id	name
1	Ravi
2	Priya
3	Shruti
4	Akhil
5	Aryan
6	Amandeep
7	Rijul
8	Aadeesh
9	Aakash
10	Pranav
NULL	NULL

room_id	staff_id	availability
1	NULL	1
2	NULL	1
3	NULL	1
4	NULL	1
5	NULL	1
6	NULL	1
7	NULL	1
8	NULL	1
9	NULL	1
10	NULL	1
NULL	NULL	NULL

patient_id	name	address	phone
1	Kshitish	Ghaziabad	1234512345
2	Aryan	6th Street Pune	4567845678
3	Nikesh	1st Sector Faridabad	9876598765
4	Sanshrav	2nd Sector Faridabad	3467934679
5	Harshit	3rd Sector Faridabad	56342184723
6	Mayank	5th Street Gurgaon	34565902284
7	Munish	Ranchi	5482927282
8	Rishabh	6th Sector Faridabad	987654321
9	Samarth	6th Street Gurgaon	4682738723
10	Chirag	Dhanbad	6483927483
11	Rikhil Gupta	Elec City Blr	123-456
NULL	NULL	NULL	NULL

```
10  CREATE PROCEDURE InsertPatient(
11    IN patient_name VARCHAR(255),
12    IN patient_address VARCHAR(255),
13    IN patient_phone VARCHAR(20)
14  )
15  BEGIN
16    INSERT INTO Patient (name, address, phone)
17    VALUES (patient_name, patient_address, patient_phone);
18  END;
19  //
20
```

```
21
22  CREATE PROCEDURE InsertDoctor(
23      IN doctor_name VARCHAR(255),
24      IN doctor_specialty VARCHAR(255)
25  )
26  BEGIN
27      INSERT INTO Doctor (name, speciality)
28      VALUES (doctor_name, doctor_specialty);
29  END;
30  //
31
```

doctor_id	name	speciality
1	Dr. Tarak	Cardiology
2	Dr. Siddhardh	Oncology
3	Dr. Sanah Sheik	Neurology
4	Dr. Srikruti	Pediatrics
5	Dr. Malhar Patel	Dermatology
6	Dr. Gabriel Joe	Endocrinology
7	Dr. Nishant DMello	Gastroenterology
8	Dr. Nishant K	Hematology
9	Dr. Stuti Agarwal	Infectious Disease
10	Dr. Jeevitha Reddy	Internal Medicine
11	Nachiket Shastry	Cardiology
NULL	NULL	NULL

```
31
32  CREATE PROCEDURE InsertMedicalStaff(
33      IN staff_name VARCHAR(255)
34  )
35  BEGIN
36      INSERT INTO MedicalStaff (name)
37      VALUES (staff_name);
38  END;
39  //
40
```

staff_id	name
1	Ravi
2	Priya
3	Shruti
4	Akhil
5	Aryan
6	Amandeep
7	Rijul
8	Aadeesh
9	Aakash
10	Pranav
11	Manas
NULL	NULL

```
41
42  CREATE PROCEDURE UpdateRoomStatus(
43      IN r_id INT,
44      IN new_availability BOOLEAN
45  )
46  BEGIN
47      UPDATE Room
48      SET availability = new_availability
49      WHERE room_id = r_id;
50  END;
51  //
52
```

room_id	staff_id	availability
1	NULL	1
2	NULL	1
3	NULL	1
4	NULL	1
5	NULL	1
6	NULL	1
7	NULL	1
8	NULL	1
9	NULL	1
10	NULL	1
NULL	NULL	NULL

```
53
54  CREATE PROCEDURE AllotFirstAvailableStaffToRoom(
55      IN r_id INT
56  )
57  BEGIN
58      DECLARE available_staff_id INT;
59
60    SELECT staff_id into available_staff_id
61    FROM MedicalStaff
62    WHERE staff_id NOT IN (
63      SELECT staff_id FROM Room
64      WHERE staff_id IS NOT NULL
65    )
66    LIMIT 1;
67
68      UPDATE Room
69      SET staff_id = available_staff_id
70      WHERE room_id = r_id;
71  END;
72    //
```

room_id	staff_id	availability
1	2	1
2	1	1
3	6	1
4	3	1
5	4	1
6	5	1
7	7	1
8	NULL	1
9	NULL	1
10	NULL	1
NULL	NULL	NULL

```
73
74
75  CREATE PROCEDURE FreeStaffFromRoom(
76      IN r_id INT
77  )
78  BEGIN
79      UPDATE Room
80      SET staff_id = NULL
81      WHERE room_id = r_id;
82  END;
83      //
84
85
```

room_id	staff_id	availability
1	NULL	1
2	1	1
3	6	1
4	3	1
5	4	1
6	5	1
7	7	1
8	NULL	1
9	NULL	1
10	NULL	1
NULL	NULL	NULL

```
CREATE PROCEDURE makeAppointment (
  IN patient_id INT,
  IN specialization VARCHAR(255),
  IN appt_date DATE,
  IN appt_start_time DATETIME,
  IN appt_end_time DATETIME
)
BEGIN
  DECLARE doc_id INT;
  DECLARE r_id INT;
  DECLARE s_id INT;

  -- Get the doctor of the specified specialization
  SELECT doctor_id INTO doc_id
  FROM Doctor
  WHERE speciality = specialization
  AND doctor_id NOT IN (
    SELECT doctor_id FROM Appointment WHERE date = appt_date
    AND ((appt_start_time BETWEEN start_time AND end_time) OR (appt_end_time BETWEEN start_time AND end_time))
  )
  ORDER BY RAND()
  LIMIT 1;

  IF doc_id IS NULL THEN
    SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'No available doctor found for the specified specialization on the given date and time';
  END IF;

  -- Get an available room with staff
  SELECT room_id, staff_id INTO r_id, s_id
  FROM Room
```

```
  -- Get an available room with staff
  SELECT room_id, staff_id INTO r_id, s_id
  FROM Room
  WHERE availability = 1
  AND staff_id IS NOT NULL
  AND room_id NOT IN (
    SELECT room_id FROM Appointment WHERE date = appt_date
    AND ((appt_start_time BETWEEN start_time AND end_time) OR (appt_end_time BETWEEN start_time AND end_time))
  )
  ORDER BY RAND()
  LIMIT 1;

  IF r_id IS NULL THEN
    SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'No available room found for the given date and time';
  END IF;

  -- Insert the appointment
  INSERT INTO Appointment (patient_id, room_id, doctor_id, start_time, end_time, date) VALUES (patient_id, r_id, doc_id, appt_start_time

  -- Update the room availability
  UPDATE Room SET availability = 0 WHERE room_id = r_id;

  SELECT CONCAT('Appointment created with doctor ', (SELECT name FROM Doctor WHERE doctor_id = doc_id), ' in room ', r_id, ' with staff
END;

//
```

appt_id	patient_id	room_id	doctor_id	start_time	end_time	date
1	1	7	11	2023-08-01 11:00:00	2023-08-01 12:00:00	2023-08-01
2	3	4	3	2023-09-01 04:00:00	2023-09-01 05:00:00	2023-09-01
3	4	5	6	2023-10-01 07:00:00	2023-10-01 08:00:00	2023-10-01
4	1	3	1	2023-08-01 11:00:00	2023-08-01 12:00:00	2023-08-01
NULL	NULL	NULL	NULL	NULL	NULL	NULL

```
CREATE PROCEDURE rescheduleAppointment (
  IN apt_id INT,
  IN new_date DATE,
  IN new_start_time DATETIME,
  IN new_end_time DATETIME
)
BEGIN
  DECLARE r_id INT;
  DECLARE s_id INT;

  -- Check if the appointment has already been diagnosed
  IF EXISTS (SELECT * FROM Diagnosis WHERE appt_id = apt_id) THEN
    SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'The appointment has already been diagnosed and cannot be rescheduled';
  END IF;

  -- Check if the new date/time is available for the same room
  IF EXISTS (
    SELECT * FROM Appointment
    WHERE room_id = (SELECT room_id FROM Appointment WHERE appt_id = apt_id)
    AND date = new_date
    AND ((new_start_time BETWEEN start_time AND end_time) OR (new_end_time BETWEEN start_time AND end_time))
  ) THEN
    SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'The new date/time is not available for the same room';
  END IF;

  -- Get an available room with staff
  SELECT room_id, staff_id INTO r_id, s_id
  FROM Room
  WHERE availability = 1
  AND staff_id IS NOT NULL
  AND room_id NOT IN (
    SELECT room_id FROM Appointment WHERE date = new_date
    AND ((new_start_time BETWEEN start_time AND end_time) OR (new_end_time BETWEEN start_time AND end_time))
    AND appt_id != apt_id
  )
  ORDER BY RAND()
  LIMIT 1;

  IF r_id IS NULL THEN
    SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'No available room found for the given date and time';
  END IF;

  -- Update the appointment
  UPDATE Appointment SET start_time = new_start_time, end_time = new_end_time, date = new_date, room_id = r_id WHERE appt_id = apt_id;

  -- Update the room availability
  UPDATE Room SET availability = 0 WHERE room_id = r_id;
  UPDATE Room SET availability = 1 WHERE room_id = (SELECT room_id FROM Appointment WHERE appt_id = apt_id AND room_id != r_id);

  SELECT CONCAT('Appointment rescheduled to room ', r_id, ' with staff ', (SELECT name FROM MedicalStaff WHERE staff_id = s_id)) AS message;
END;
//
```

appt_id	patient_id	room_id	doctor_id	start_time	end_time	date
1	1	6	11	2023-08-01 14:00:00	2023-08-01 15:00:00	2023-08-01
2	3	4	3	2023-09-01 04:00:00	2023-09-01 05:00:00	2023-09-01
3	4	5	6	2023-10-01 07:00:00	2023-10-01 08:00:00	2023-10-01
4	1	3	1	2023-08-01 11:00:00	2023-08-01 12:00:00	2023-08-01
NULL	NULL	NULL	NULL	NULL	NULL	NULL

```
CREATE PROCEDURE diagnose (
  IN apt_id INT,
  IN diagnosis VARCHAR(255),
  IN treatment VARCHAR(255)
)
BEGIN
  DECLARE p_id INT;

  -- Check if patient has already been diagnosed in the same appointment
  SELECT patient_id INTO p_id
  FROM DiagnosisPatient
  WHERE diagnosis_id IN (
    SELECT diagnosis_id FROM Diagnosis WHERE appt_id = apt_id
  )
  AND patient_id IN (
    SELECT patient_id FROM Appointment WHERE appt_id = apt_id
  );

  IF p_id IS NOT NULL THEN
    SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Patient has already been diagnosed in this appointment';
  END IF;

  -- Insert the diagnosis
  INSERT INTO Diagnosis (appt_id, diagnosis, treatment) VALUES (apt_id, diagnosis, treatment);

  -- Update room availability
  UPDATE Room SET availability = 1 WHERE room_id = (SELECT room_id FROM Appointment WHERE appt_id = apt_id);
```

```

  -- Insert into DiagnosisPatient
  INSERT INTO DiagnosisPatient (patient_id, diagnosis_id)
  SELECT patient_id, LAST_INSERT_ID()
  FROM Appointment
  WHERE appt_id = apt_id;
END;
//
```

diagnosis_id	appt_id	diagnosis	treatment
1	1	CANCER	Chemo
2	3	Malaria	BesRest
3	2	AIDS	NoCure
NULL	NULL	NULL	NULL

```
CREATE PROCEDURE insert_billing(
IN p_diagnosis_id INT,
    IN p_cost DECIMAL(10,2)
)
BEGIN
    DECLARE v_appointment_date DATE;
    DECLARE v_patient_id INT;
    SELECT date, patient_id INTO v_appointment_date, v_patient_id FROM Appointment WHERE appt_id = (SELECT appt_id FROM Diagnosis WHERE diagnosis_id = p_diagnosis_id);
    INSERT INTO Billing(diagnosis_id, cost, bill_date) VALUES(p_diagnosis_id, p_cost, v_appointment_date);
    INSERT INTO BillingPatient(patient_id, billing_id) VALUES(v_patient_id, LAST_INSERT_ID());
END;
//
```

Result Grid | Filter Rows: Search | Edit:

billing_id	diagnosis_id	cost	bill_date
1	1	100.00	2023-08-01
2	2	25.00	2023-10-01
NULL	NULL	NULL	NULL

```
CREATE TRIGGER add_gst_to_billing
BEFORE INSERT ON Billing
FOR EACH ROW
BEGIN
  SET NEW.cost = NEW.cost * 1.18;
END;

//
```

billing_id	diagnosis_id	cost	bill_date
1	1	100.00	2023-08-01
2	2	25.00	2023-10-01
3	3	59.00	2023-09-01
NULL	NULL	NULL	NULL

```sql
CREATE PROCEDURE generate_bill(IN p_patient_id INT)
BEGIN
  DECLARE total_cost DECIMAL(10, 2) DEFAULT 0;
  DECLARE patient_name VARCHAR(255);
  DECLARE patient_address VARCHAR(255);
  DECLARE patient_phone VARCHAR(20);

  SELECT name, address, phone INTO patient_name, patient_address, patient_phone
  FROM Patient
  WHERE patient_id = p_patient_id;

  SELECT SUM(cost) INTO total_cost
  FROM Billing
  JOIN Diagnosis ON Billing.diagnosis_id = Diagnosis.diagnosis_id
  JOIN DiagnosisPatient ON Diagnosis.diagnosis_id = DiagnosisPatient.diagnosis_id
  WHERE DiagnosisPatient.patient_id = p_patient_id;

  SELECT patient_name AS 'Patient Name', patient_address AS 'Patient Address', patient_phone AS 'Patient Phone Number', CONCAT('Total cost: $', total_cost) AS 'Bill Summary';
END;
```

Patient Name	Patient Address	Patient Phone Numb...	Bill Summary
Nikesh	1st Sector Faridabad	9876598765	Total cost: $59.00

```sql
 CREATE VIEW all_doctors AS
SELECT doctor_id, name, speciality
FROM Doctor;
//

CREATE VIEW patient_medical_history AS
SELECT p.patient_id, p.name AS patient_name, p.address, p.phone, d.diagnosis, d.treatment, b.cost, b.bill_date
FROM Patient p
LEFT JOIN DiagnosisPatient dp ON p.patient_id = dp.patient_id
LEFT JOIN Diagnosis d ON dp.diagnosis_id = d.diagnosis_id
LEFT JOIN Billing b ON d.diagnosis_id = b.diagnosis_id;
//
```

doctor_id	name	speciality
1	Dr. Tarak	Cardiology
2	Dr. Siddhardh	Oncology
3	Dr. Sanah Sheik	Neurology
4	Dr. Srikruti	Pediatrics
5	Dr. Malhar Patel	Dermatology
6	Dr. Gabriel Joe	Endocrinology
7	Dr. Nishant DMello	Gastroenterology
8	Dr. Nishant K	Hematology
9	Dr. Stuti Agarwal	Infectious Disease
10	Dr. Jeevitha Reddy	Internal Medicine
11	Nachiket Shastry	Cardiology

patient_id	patient_name	address	phone	diagnosis	treatment	cost	bill_date
1	Kshitish	Ghaziabad	1234512345	CANCER	Chemo	100.00	2023-08-01
2	Aryan	6th Street Pune	4567845678	NULL	NULL	NULL	NULL
3	Nikesh	1st Sector Faridabad	9876598765	AIDS	NoCure	59.00	2023-09-01
4	Sanshrav	2nd Sector Faridabad	3467934679	Malaria	BesRest	25.00	2023-10-01
5	Harshit	3rd Sector Faridabad	56342184723	NULL	NULL	NULL	NULL
6	Mayank	5th Street Gurgaon	34565902284	NULL	NULL	NULL	NULL
7	Munish	Ranchi	5482927282	NULL	NULL	NULL	NULL
8	Rishabh	6th Sector Faridabad	987654321	NULL	NULL	NULL	NULL
9	Samarth	6th Street Gurgaon	4682738723	NULL	NULL	NULL	NULL
10	Chirag	Dhanbad	6483927483	NULL	NULL	NULL	NULL
11	Rikhil Gupta	Elec City Blr	123-456	NULL	NULL	NULL	NULL

```
278
279 ●   CREATE PROCEDURE total_earnings_doctor(IN doctor_name VARCHAR(255))
280     BEGIN
281       DECLARE doctor_id INT;
282       DECLARE total DECIMAL(10,2);
283
284       SELECT doctor_id INTO doctor_id FROM Doctor WHERE name = doctor_name;
285
286       SELECT SUM(b.cost) INTO total
287       FROM Billing b
288       INNER JOIN Diagnosis d ON b.diagnosis_id = d.diagnosis_id
289       INNER JOIN Appointment a ON d.appt_id = a.appt_id
290       WHERE a.doctor_id = doctor_id;
291
292       SELECT CONCAT(doctor_name, ' has earned ', total, ' in total.') AS result;
293     END;
294     //
295
```

result
Dr. Sanah Sheik has earned 59.00 in total.

```
296    //
297
298    CREATE PROCEDURE total_earnings_specialization(IN speciality_name VARCHAR(255))
299    BEGIN
300      DECLARE total DECIMAL(10,2);
301
302      SELECT SUM(b.cost) INTO total
303      FROM Billing b
304      INNER JOIN Diagnosis d ON b.diagnosis_id = d.diagnosis_id
305      INNER JOIN Appointment a ON d.appt_id = a.appt_id
306      INNER JOIN Doctor doc ON a.doctor_id = doc.doctor_id
307      WHERE doc.speciality = speciality_name;
308
309      SELECT CONCAT('The specialization ', speciality_name, ' has earned ', total, ' in total.') AS result;
310    END;
311    //
```

result
The specialization Cardiology has earned 100.00 in total.

```
314    CREATE PROCEDURE find_patients_for_doctor(IN doc_name VARCHAR(255))
315    BEGIN
316      SELECT DISTINCT p.patient_id, p.name
317      FROM Patient p
318      JOIN Appointment a ON a.patient_id = p.patient_id
319      JOIN Doctor d ON d.doctor_id = a.doctor_id
320      WHERE d.name = doc_name;
321    END;
322    //
```

patient_id	name
1	Kshitish

```
324
325    CREATE PROCEDURE fetch_previous_diagnoses(IN patient_name VARCHAR(255))
326    BEGIN
327      SELECT d.diagnosis, d.treatment, a.start_time, a.end_time, a.date, doc.name AS doctor_name
328      FROM Diagnosis d
329      JOIN Appointment a ON d.appt_id = a.appt_id
330      JOIN Doctor doc ON a.doctor_id = doc.doctor_id
331      JOIN Patient p ON a.patient_id = p.patient_id
332      WHERE p.name = patient_name;
333    END;
334    //
```

Result Grid Filter Rows: Search Export:

diagnosis	treatment	start_time	end_time	date	doctor_name
CANCER	Chemo	2023-08-01 14:00:00	2023-08-01 15:00:00	2023-08-01	Nachiket Shastry
Fever	Dolo	2023-08-01 11:00:00	2023-08-01 12:00:00	2023-08-01	Dr. Tarak