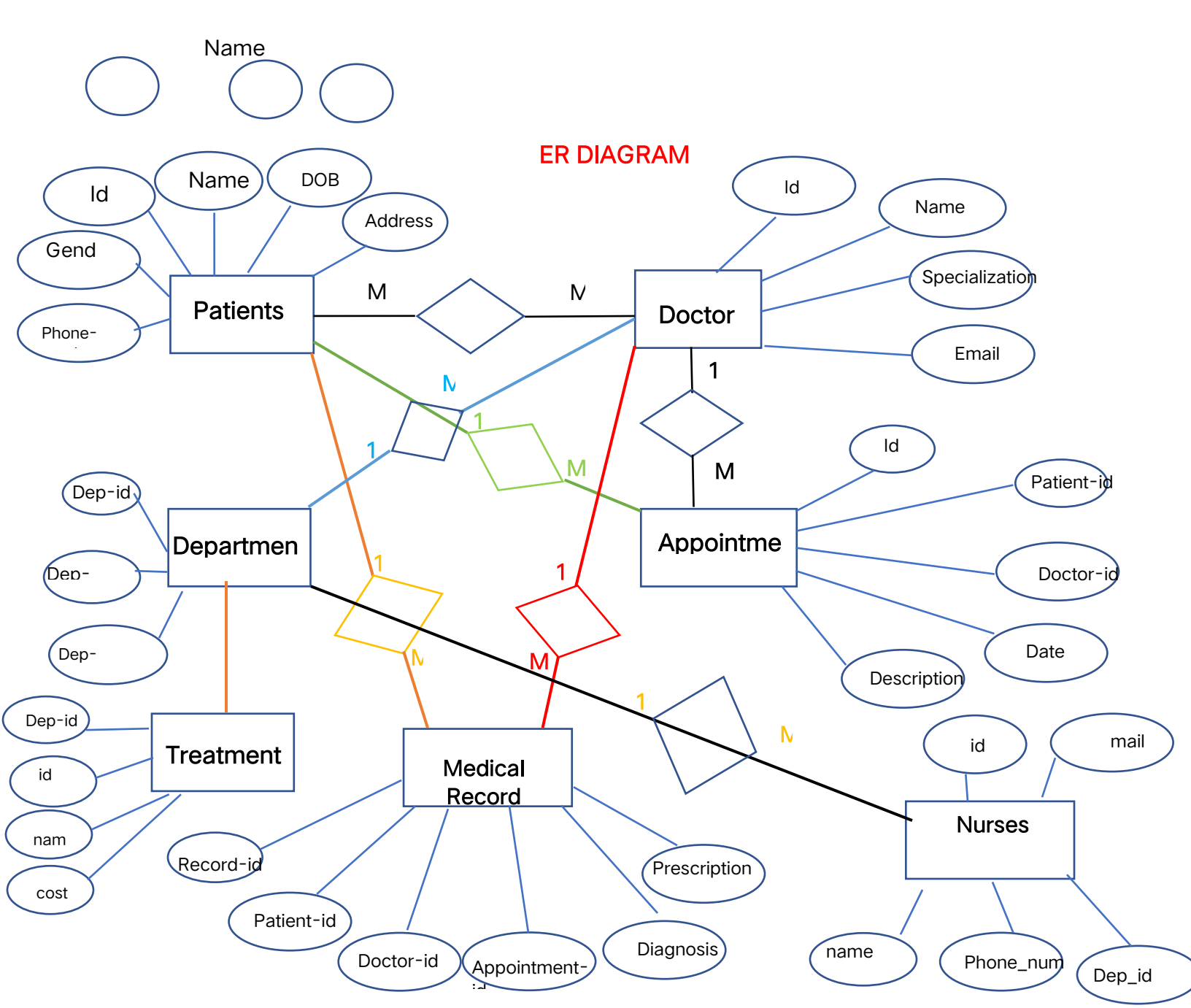
Name
ER DIAGRAM
Id
Name
DOB
Address
Gend
Phone-
Patients
M
M
Doctor
Id
Name
Specialization
Email
1
M
1
M
1
M
Id
Patient-id
Doctor-id
Date
Description
Appointme
Dep-id
Dep-
Dep-
Departmen
1
M
1
M
1
M
Dep-id
id
nam
cost
Treatment
Medical Record
Record-id
Patient-id
Doctor-id
Appointment-
Diagnosis
Prescription
id
mail
Nurses
name
Phone_num
Dep_id

Project

```sql
-- Table: patients
CREATE TABLE patients (
  patient_id INT PRIMARY KEY,
  patient_name VARCHAR(100),
  gender VARCHAR(10),
  date_of_birth DATE,
  phone_number VARCHAR(20),
  address VARCHAR(200)
);
-- Alter Table To Add Nurse Id
ALTER TABLE patients
ADD nurse_id INT,
FOREIGN KEY (nurse_id) REFERENCES nurses(nurse_id);

select * from patients
select * from nurses
UPDATE patients
SET nurse_id = 101
WHERE patient_id = 1
UPDATE patients
SET nurse_id = 102
WHERE patient_id = 2
UPDATE patients
SET nurse_id = 103
WHERE patient_id = 3
UPDATE patients
SET nurse_id = 104
WHERE patient_id = 4
UPDATE patients
SET nurse_id = 105
WHERE patient_id = 5
CREATE PROCEDURE InsertPatient(
   @patient_id INT,
   @patient_name VARCHAR(100),
   @gender VARCHAR(10),
   @date_of_birth DATE,
   @phone_number VARCHAR(20),
   @address VARCHAR(200)
)
AS
BEGIN
   INSERT INTO patients (patient_id, patient_name, gender, date_of_birth, phone_number,
address)
   VALUES (@patient_id, @patient_name, @gender, @date_of_birth, @phone_number, @address);
END;


--Insert Records In Patients Table
exec InsertPatient 1,'Ahsan','Male','2001-06-21','03034931341','City Kharian'
exec InsertPatient 2,'Nazim','Male','2003-01-01','03123456789','City Gulayana'
exec InsertPatient 3,'Molvi','Male','2004-05-19','03423456782','City Lala Musa'
```

```
exec InsertPatient 4,'Ali','Male','2005-04-30','03432562382','City Kotala'
exec InsertPatient 5,'Waseem','Male','2002-03-28','03056382456','City Jehlum'
select * from patients


-- Table: doctors
CREATE TABLE doctors (
  doctor_id INT PRIMARY KEY,
  doctor_name VARCHAR(100),
  specialization VARCHAR(100),
  phone_number VARCHAR(20),
  email VARCHAR(100)
);
-- Alter Table To Add Department Id as a Foreign Key
ALTER TABLE doctors
ADD department_id INT,
FOREIGN KEY (department_id) REFERENCES departments(department_id);

update doctors set department_id=1 where doctor_id=1

update doctors set department_id=2 where doctor_id=1

update doctors set department_id=3 where doctor_id=2

update doctors set department_id=4 where doctor_id=3
update doctors set department_id=4 where doctor_id=4
update doctors set department_id=5 where doctor_id=5
select * from doctors
select * from departments

-- Procedure: InsertDoctor
CREATE PROCEDURE InsertDoctor(
   @doctor_id INT,
   @doctor_name VARCHAR(100),
   @specialization VARCHAR(100),
   @phone_number VARCHAR(20),
   @email VARCHAR(100)
)
AS
BEGIN
   INSERT INTO doctors (doctor_id, doctor_name, specialization, phone_number, email)
   VALUES (@doctor_id, @doctor_name, @specialization, @phone_number, @email);
END;

--Insert Records In Doctors Table
exec InsertDoctor 1,'Dr Ali','Brain','123','drali@gmail.com'
exec InsertDoctor 2,'Dr Aswand','Skin','456','drAswand@gmail.com'
exec InsertDoctor 3,'Dr Muneeb','Eyes','789','drMuneeb@gmail.com'
exec InsertDoctor 4,'Dr Hassan','Heart','000','drHassan@gmail.com'
exec InsertDoctor 5,'Dr Khizer','Ears','111','drKhizer@gmail.com'
select * from doctors


-- Table: appointments
CREATE TABLE appointments (
```

```
  appointment_id INT PRIMARY KEY,
  patient_id INT,
  doctor_id INT,
  appointment_date DATETIME,
  description VARCHAR(200),
  FOREIGN KEY (patient_id) REFERENCES patients(patient_id),
  FOREIGN KEY (doctor_id) REFERENCES doctors(doctor_id)
);

-- Procedure: InsertAppointment
CREATE PROCEDURE InsertAppointment(
    @appointment_id INT,
    @patient_id INT,
    @doctor_id INT,
    @appointment_date DATETIME,
    @description VARCHAR(200)
)
AS
BEGIN
    INSERT INTO appointments(appointment_id, patient_id, doctor_id, appointment_date,
description)
    VALUES (@appointment_id, @patient_id, @doctor_id, @appointment_date, @description);
END;
-- Insertion of data in appointment table
exec InsertAppointment 1,1,5,'2023-06-06','Regular CheckUp'
exec InsertAppointment 2,2,2,'2023-06-06','Regular CheckUp'
exec InsertAppointment 3,3,4,'2023-06-23','Regular CheckUp'
exec InsertAppointment 4,4,3,'2023-04-06','Regular CheckUp'
exec InsertAppointment 5,5,2,'2023-03-19','Regular CheckUp'
select * from appointments

select * from patients;
select * from doctors;

-- Table: medical_records
CREATE TABLE medical_records (
  record_id INT PRIMARY KEY,
  patient_id INT,
  doctor_id INT,
  appointment_id INT,
  diagnosis VARCHAR(200),
  prescription VARCHAR(200),
  FOREIGN KEY (patient_id) REFERENCES patients(patient_id),
  FOREIGN KEY (doctor_id) REFERENCES doctors(doctor_id),
  FOREIGN KEY (appointment_id) REFERENCES appointments(appointment_id)
);


-- Procedure: InsertMedicalRecord
CREATE PROCEDURE InsertMedicalRecord(
    @record_id INT,
    @patient_id INT,
    @doctor_id INT,
    @appointment_id INT,
    @diagnosis VARCHAR(200),
```

```
    @prescription VARCHAR(200)
)
AS
BEGIN
    INSERT INTO medical_records (record_id, patient_id, doctor_id, appointment_id, diagnosis,
prescription)
    VALUES (@record_id, @patient_id, @doctor_id, @appointment_id, @diagnosis, @prescription);
END;

--Insertion Of Data In Medical Record
exec InsertMedicalRecord 1,1,5,1,'Eear Testing','Not Req'
exec InsertMedicalRecord 2,2,2,2,'Testing','Not Req'
exec InsertMedicalRecord 3,3,4,3,'Testing','Not Req'
exec InsertMedicalRecord 4,4,3,4,'Testing','Not Req'
exec InsertMedicalRecord 5,5,2,5,'Testing','Not Req'
select * from medical_records

-- Table: departments
CREATE TABLE departments (
  department_id INT PRIMARY KEY,
  department_name VARCHAR(100),
  description VARCHAR(200)
);

-- Procedure: InsertDepartment
CREATE PROCEDURE InsertDepartment(
    @department_id INT,
    @department_name VARCHAR(100),
    @description VARCHAR(200)
)
AS
BEGIN
    INSERT INTO departments(department_id, department_name, description)
    VALUES (@department_id, @department_name, @description);
END


--Add record in departments table
EXEC InsertDepartment 1, 'Cardiology', 'Deals with heart-related diseases';
EXEC InsertDepartment 2, 'Orthopedics', 'Specializes in bone and joint issues';
EXEC InsertDepartment 3, 'Neurology', 'Focuses on nervous system disorders';
EXEC InsertDepartment 4, 'Gynecology', 'Deals with women''s reproductive health';
EXEC InsertDepartment 5, 'Pediatrics', 'Specializes in children''s healthcare';
EXEC InsertDepartment 6, 'Oncology', 'Focuses on cancer diagnosis and treatment';
EXEC InsertDepartment 7, 'ENT (Ear, Nose, and Throat)', 'Deals with disorders of the ear, nose,
and throat';
EXEC InsertDepartment 8, 'General Surgery', 'Covers a wide range of surgical procedures';
EXEC InsertDepartment 9, 'Internal Medicine', 'Focuses on adult diseases and general health';
EXEC InsertDepartment 10, 'Dermatology', 'Specializes in skin-related issues';


select * from departments

-- Table: nurses
CREATE TABLE nurses (
```

```sql
  nurse_id INT PRIMARY KEY,
  nurse_name VARCHAR(100),
  department_id INT,
  phone_number VARCHAR(20),
  email VARCHAR(100),
  FOREIGN KEY (department_id) REFERENCES departments(department_id)
);

-- Procedure: InsertNurse
CREATE PROCEDURE InsertNurse(
    @nurse_id INT,
    @nurse_name VARCHAR(100),
    @department_id INT,
    @phone_number VARCHAR(20),
    @email VARCHAR(100)
)
AS
BEGIN
    INSERT INTO nurses(nurse_id, nurse_name, department_id, phone_number, email)
    VALUES (@nurse_id, @nurse_name, @department_id, @phone_number, @email);
END;

--insert data into nurses
-- First nurse
EXEC InsertNurse 101, 'Ayesha Khan', 1, '0300-1234567', 'ayesha.khan@example.com';

-- Second nurse
EXEC InsertNurse 102, 'Mohammad Ahmed', 2, '0345-9876543',
'mohammad.ahmed@example.com';

-- Third nurse
EXEC InsertNurse 103, 'Fatima Siddiqui', 3, '0312-3456789', 'fatima.siddiqui@example.com';

-- Fourth nurse
EXEC InsertNurse 104, 'Ali Hassan', 4, '0321-8765432', 'ali.hassan@example.com';

-- Fifth nurse
EXEC InsertNurse 105, 'Zainab Malik', 5, '0333-4567890', 'zainab.malik@example.com';

-- Sixth nurse
EXEC InsertNurse 106, 'Omar Khan', 6, '0344-9876543', 'omar.khan@example.com';

-- Seventh nurse
EXEC InsertNurse 107, 'Sana Mahmood', 7, '0311-2345678', 'sana.mahmood@example.com';

-- Eighth nurse
EXEC InsertNurse 108, 'Usman Ali', 8, '0302-8765432', 'usman.ali@example.com';

-- Ninth nurse
EXEC InsertNurse 109, 'Rabia Khan', 9, '0322-1234567', 'rabia.khan@example.com';

-- Tenth nurse
EXEC InsertNurse 110, 'Bilal Ahmed', 10, '0331-8765432', 'bilal.ahmed@example.com';
```

```
select * from nurses
-- Table: rooms
CREATE TABLE rooms (
  room_id INT PRIMARY KEY,
  room_number VARCHAR(20),
  department_id INT,
  capacity INT,
  availability VARCHAR(10),
  FOREIGN KEY (department_id) REFERENCES departments(department_id)
);

-- Procedure: InsertRoom
CREATE PROCEDURE InsertRoom(
    @room_id INT,
    @room_number VARCHAR(20),
    @department_id INT,
    @capacity INT,
    @availability VARCHAR(10)
)
AS
BEGIN
    INSERT INTO rooms (room_id, room_number, department_id, capacity, availability)
    VALUES (@room_id, @room_number, @department_id, @capacity, @availability);
END;

exec InsertRoom 1,1,1,20,'available'
exec InsertRoom 2,2,2,20,'available'
exec InsertRoom 3,3,3,20,'available'
exec InsertRoom 4,4,4,20,'available'
exec InsertRoom 5,5,5,20,'available'
select * from rooms

-- Table: treatments
CREATE TABLE treatmentTable(
  treatment_id INT PRIMARY KEY,
  treatment_name VARCHAR(100),
  department_id INT,
  cost DECIMAL(10, 2),
  FOREIGN KEY (department_id) REFERENCES departments(department_id)
);


-- Procedure: InsertTreatment
CREATE PROCEDURE InsertTreatmentTableData(
    @treatment_id INT,
    @treatment_name VARCHAR(100),
    @department_id INT,
    @cost DECIMAL(10, 2)
)
AS
BEGIN
    INSERT INTO treatmentTable(treatment_id, treatment_name, department_id, cost)
    VALUES (@treatment_id, @treatment_name, @department_id, @cost);
END;
```

```
-- Insertion of data in the treatment table
exec InsertTreatmentTableData 1, 'Heart Checkup', 1, 150.00;
exec InsertTreatmentTableData 2, 'Orthopedic Consultation', 2, 120.00;
exec InsertTreatmentTableData 3, 'Eye Examination', 3, 80.00;
exec InsertTreatmentTableData 4, 'Gynecology Consultation', 4, 100.00;
exec InsertTreatmentTableData 5, 'Pediatric Checkup', 5, 90.00;
exec InsertTreatmentTableData 6, 'Oncology Screening', 6, 200.00;
exec InsertTreatmentTableData 7, 'ENT Examination', 7, 75.00;
exec InsertTreatmentTableData 8, 'Surgical Procedure', 8, 500.00;
exec InsertTreatmentTableData 9, 'Internal Medicine Consultation', 9, 110.00;
exec InsertTreatmentTableData 10, 'Dermatology Checkup', 10, 95.00;


select * from patients
select * from doctors
select * from departments
select * from appointments
select * from medical_records
select * from nurses
select * from rooms
select * from treatmentTable
```

Project Description:

The provided SQL script represents a simplified database schema for a medical management system. The schema includes tables for patients, doctors, appointments, medical records, departments, nurses, rooms, and treatments. Below is a brief description of each table and its purpose:

Patients: This table stores information about the patients, including patient_id, patient_name, gender, date_of_birth, phone_number, and address. It also has a nurse_id as a foreign key to associate each patient with their nurse.

Doctors: This table contains data about the doctors, such as doctor_id, doctor_name, specialization, phone_number, and email. Each doctor is uniquely identified by the doctor_id.

Appointments: This table maintains records of patient appointments with doctors. It includes appointment_id, patient_id, doctor_id, appointment_date, and description. The patient_id and doctor_id are foreign keys to link appointments to specific patients and doctors.

Medical_Records: This table keeps track of medical records, such as diagnosis

and prescription, for each appointment. It uses record_id as the primary key and references patient_id, doctor_id, and appointment_id from their respective tables.

Departments: This table stores details about medical departments, including department_id, department_name, and description.

Nurses: This table contains information about the nurses, such as nurse_id, nurse_name, department_id, phone_number, and email. The department_id is a foreign key linking each nurse to their department.

Rooms: This table represents the available rooms in the medical facility. It contains room_id, room_number, department_id, capacity, and availability. The department_id is a foreign key to associate each room with its respective department.

TreatmentTable: This table stores information about various medical treatments offered in different departments. It includes treatment_id, treatment_name, department_id, and cost. The department_id is a foreign key to link each treatment to a specific department.

Improvements:

Data Validation: Implement data validation checks in stored procedures to ensure that only valid and appropriate data is inserted into the tables.

Indexing: Consider adding appropriate indexes to columns frequently used in search or join operations to improve query performance.

Error Handling: Enhance the stored procedures with proper error handling to provide meaningful error messages to users in case of any issues.

Unique Constraints: Apply unique constraints to appropriate columns (e.g., email in doctors and nurses table) to prevent duplicate entries.

Views: Create views to simplify complex queries or to present a specific subset of data to users.

Security: Implement user access controls and permissions to restrict unauthorized access to sensitive data.

Relationships: Review and ensure that all necessary relationships and constraints are correctly defined between tables.

Data Backup and Recovery: Establish a regular backup plan to safeguard data and create a recovery strategy in case of data loss.

Reporting: Consider building reports or analytics based on the data to provide valuable insights to administrators and medical staff.

Remember that the improvements mentioned above may vary depending on the specific requirements and scale of the medical management system. Always consider best practices and the actual needs of the application when making enhancements.