**Hospital DataBase Management System**

**Problem Statement :**

Design a database management system for a hospital that keeps track of the following information:

* Patients: patient information including personal details
* Appointments: appointment information, including the date, time, and doctor assigned.
* Doctors: doctor information, including personal details, specializations, and schedule.
* Nurses: nurse information, including personal details and schedule.
* Rooms: room information, including room number, room type, and availability.
* Prescriptions: prescription information, including the medication prescribed and dosage.

The system should also be able to handle the following operations:

* Schedule appointments and assign doctors to patients.
* Assign patients to rooms and update room availability.
* Prescribe medications and calculate the total cost of prescriptions.

The relationships between the entities include:

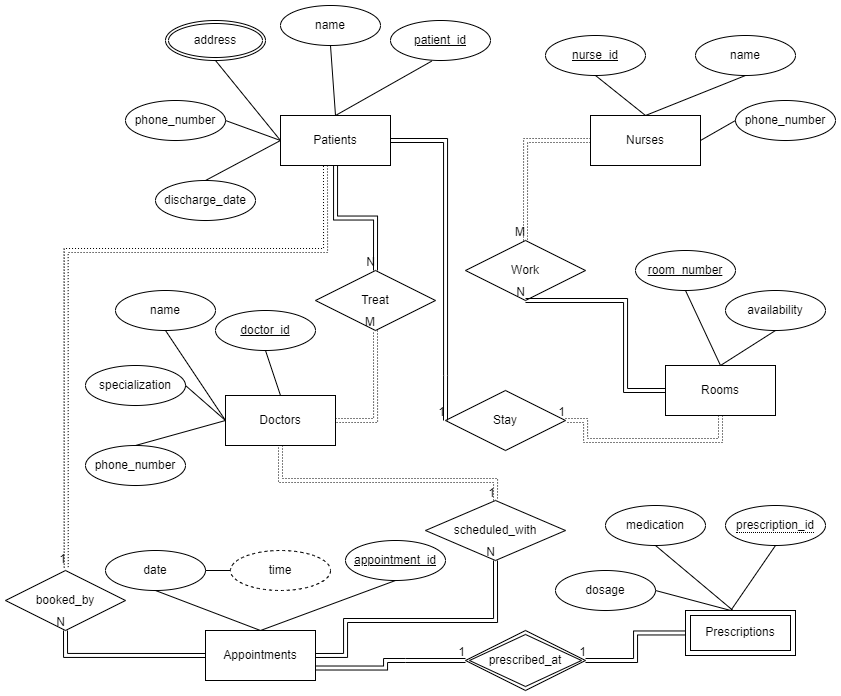
* Patients can have multiple Appointments, and an Appointment can have one Patient.
* Appointments can have one Doctor, and a Doctor can have multiple Appointments.
* Patients can be assigned to one Room, and a Room can have one Patient.
* Patients can have multiple Prescriptions, and a Prescription can be given to multiple Patients.
* Prescriptions can have one Doctor, and a Doctor can have multiple Prescriptions.
* Nurses can be assigned to multiple Rooms, and a Room can have multiple Nurses..
* Appointment can be assigned to one Prescriptions, and a Prescription can have one Appointment.

**Solution :**

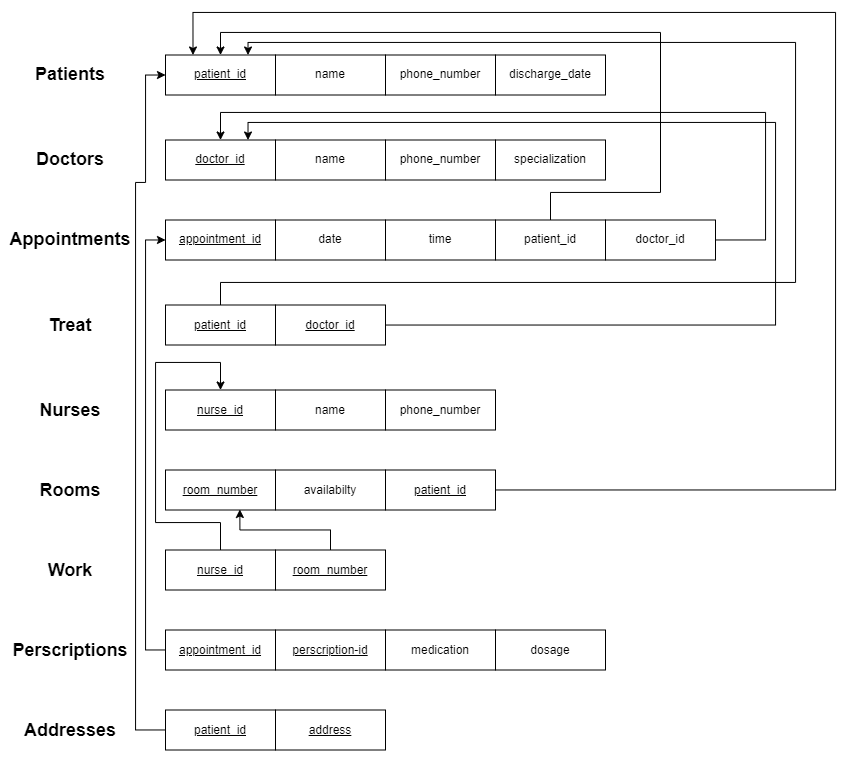
*Explanation:*

* The ER diagram includes entities for Patients, Appointments, Doctors, Nurses, Rooms and Prescriptions
* The Patients entity has attributes such as Patient ID, Name, Address, Phone Number and Discharge Date
* The Appointment entity has attributes such as Appointment ID, Date, Time, Doctor ID, and Patient ID.
* The Doctors entity has attributes such as Doctor ID, Name, Specialization, Phone Number and Schedule.
* The Nurses entity has attributes such as Nurse ID, Name, Phone Number and Schedule.
* The Rooms entity has attributes such as Room Number and Availability.
* The Prescriptions entity has attributes such as Prescription ID, Medication, Dosage and Appointment ID.

*ER Diagram:*



*Mapping:*



*Creating the Tables:*

| **CREATE** **TABLE** Patients (  patient\_id INT PRIMARY **KEY**,  **name** VARCHAR(25) **NOT** **NULL**,  phone\_number VARCHAR(10) **NOT** **NULL**,  discharge\_date DATE, );   **CREATE** **TABLE** Doctors (  doctor\_id INT PRIMARY **KEY**,  **name** VARCHAR(25) **NOT** **NULL**,  specialization VARCHAR(25) **NOT** **NULL**,  phone\_number VARCHAR(10) **NOT** **NULL**, );  **CREATE** **TABLE** Appointments (  appointment\_id INT PRIMARY **KEY**,  date DATE **NOT** **NULL**,  **time** **TIME** **AS** (**TIME**(date)) PERSISTED,  doctor\_id INT **NOT** **NULL**,  patient\_id INT **NOT** **NULL**,  FOREIGN **KEY** (doctor\_id) **REFERENCES** Doctors(doctor\_id),  FOREIGN **KEY** (patient\_id) **REFERENCES** Patients(patient\_id) );  **CREATE** **TABLE** Treat (  doctor\_id INT **NOT** **NULL**,  patient\_id INT **NOT** **NULL**,  FOREIGN **KEY** (doctor\_id) **REFERENCES** Doctors(doctor\_id),  FOREIGN **KEY** (patient\_id) **REFERENCES** Patients(patient\_id),  PRIMARY **KEY** (doctor\_id, patient\_id) );  **CREATE** **TABLE** Rooms (  room\_number INT PRIMARY **KEY**,  **availability** ENUM(**'Yes'**, **'No'**) **NOT** **NULL**,  patient\_id INT,  FOREIGN **KEY** (patient\_id) **REFERENCES** Patients(patient\_id) );  **CREATE** **TABLE** Nurses (  nurse\_id INT PRIMARY **KEY**,  **name** VARCHAR(25) **NOT** **NULL**,  phone\_number VARCHAR(10) **NOT** **NULL**, );   **CREATE** **TABLE** Work (  nurse\_id INT **NOT** **NULL**,  patient\_id INT **NOT** **NULL**,  FOREIGN **KEY** (nurse\_id) **REFERENCES** Nurses(nurse\_id),  FOREIGN **KEY** (room\_number) **REFERENCES** Rooms(room\_number),  PRIMARY **KEY** (nurse\_id, room\_number) );  **CREATE** **TABLE** Prescriptions (  appointment\_id INT,  prescription\_id INT,  medication VARCHAR(25) **NOT** **NULL**,  dosage VARCHAR(25) **NOT** **NULL**,  PRIMARY **KEY** ( prescription\_id, appointment\_id),  FOREIGN **KEY** (appointment\_id) **REFERENCES** Appointments(appointment\_id) );  **CREATE** **TABLE** Addresses (  patient\_id INT **NOT** **NULL**,  address VARCHAR(25),  PRIMARY **KEY** (patient\_id, address),  FOREIGN **KEY** (patient\_id) **REFERENCES** Patients(patient\_id) ) |
| --- |

*Inserting into Tables:*

| **INSERT** **INTO** Patients **VALUES**(1, **'John Doe'**, **'555-555-5555'**, **'2022-01-01'**); **INSERT** **INTO** Patients **VALUES**(2, **'Jane Smith'**, **'555-555-5556'**, **'2022-02-01'**); **INSERT** **INTO** Patients **VALUES**(3, **'Bob Johnson'**, **'555-555-5557'**, **'2022-03-01'**);  **INSERT** **INTO** Doctors **VALUES**(1, **'Dr. Smith'**, **'Surgeon'**, **'555-555-5558'**); **INSERT** **INTO** Doctors **VALUES**(2, **'Dr. Johnson'**, **'Pediatrician'**, **'555-555-5559'**); **INSERT** **INTO** Doctors **VALUES**(3, **'Dr. Williams'**, **'Cardiologist'**, **'555-555-5560'**);  **INSERT** **INTO** Appointments **VALUES**(1, **'2022-01-01'**, **'09:00:00'**, 1, 1); **INSERT** **INTO** Appointments **VALUES**(2, **'2022-01-02'**, **'10:00:00'**, 2, 2); **INSERT** **INTO** Appointments **VALUES**(3, **'2022-01-03'**, **'11:00:00'**, 3, 3);  **INSERT** **INTO** Treat **VALUES**(1, 1); **INSERT** **INTO** Treat **VALUES**(2, 2); **INSERT** **INTO** Treat **VALUES**(3, 3);  **INSERT** **INTO** Nurses **VALUES**(1, **'Nurse Jane'**, **'555-555-5561'**); **INSERT** **INTO** Nurses **VALUES**(2, **'Nurse Bob'**, **'555-555-5562'**); **INSERT** **INTO** Nurses **VALUES**(3, **'Nurse Mary'**, **'555-555-5563'**);  **INSERT** **INTO** Rooms **VALUES**(101, **'Yes'**, 1); **INSERT** **INTO** Rooms **VALUES**(102, **'No'**, **NULL**); **INSERT** **INTO** Rooms **VALUES**(103, **'Yes'**, 3);  **INSERT** **INTO** Work **VALUES**(1, 101); **INSERT** **INTO** Work **VALUES**(2, 102); **INSERT** **INTO** Work **VALUES**(3, 103);  **INSERT** **INTO** Prescriptions **VALUES**(1, 1, **'Ibuprofen'**, **'2 tablets 3 times a day'**); **INSERT** **INTO** Prescriptions **VALUES**(2, 2, **'Amoxicillin'**, **'1 tablet 2 times a day'**); **INSERT** **INTO** Prescriptions **VALUES**(3, 3, **'Metformin'**, **'1 tablet 3 times a day'**);  **INSERT** **INTO** Addresses **VALUES**(1, **'123 Main St.'**); **INSERT** **INTO** Addresses **VALUES**(2, **'456 Park Ave.'**); **INSERT** **INTO** Addresses **VALUES**(3, **'789 Elm St.'**); **INSERT** **INTO** Addresses **VALUES**(1, **'435 Main St'**); **INSERT** **INTO** Addresses **VALUES**(2, **'526 Park Ave'**); **INSERT** **INTO** Addresses **VALUES**(3, **'459 Elm St'**); |
| --- |

*Queries:*

1. Retrieve the names of all patients who have an appointment scheduled with a doctor who specializes in "Cardiology" on the date "2022-05-15":

| **SELECT** Patients.name **FROM** Patients **JOIN** Appointments **ON** Patients.patient\_id = Appointments.patient\_id **JOIN** Doctors **ON** Appointments.doctor\_id = Doctors.doctor\_id **WHERE** Appointments.date = **'2022-05-15'** **AND** Doctors.specialization =  **'Cardiology'**; |
| --- |

1. Retrieve the phone numbers of all patients who are currently hospitalized:

| **SELECT** Patients.phone\_number **FROM** Patients **JOIN** Rooms **ON** Patients.patient\_id = Rooms.patient\_id **WHERE** Rooms.availability = **'No'**; |
| --- |

1. Retrieve the names of all patients who are currently hospitalized and are being treated by a nurse with ID = 3:

| **SELECT** Patients.name **FROM** Patients **JOIN** Rooms **ON** Patients.patient\_id = Rooms.patient\_id **JOIN** Work **ON** Rooms.room\_number = Work.room\_number **WHERE** Rooms.availability = **'No'** **AND** Work.nurse\_id = 3; |
| --- |

1. What is the medication prescribed for patients with ID 123 in their last appointment?

| **SELECT** Prescriptions.medication **FROM** Prescriptions **JOIN** Appointments **ON** Prescriptions.appointment\_id =  Appointments.appointment\_id **WHERE** Appointments.patient\_id = 123 **ORDER** **BY** Appointments.date **DESC** **LIMIT** 1; |
| --- |

1. Retrieve the average salary of all nurses who work in room number 101:

| **SELECT** **AVG**(salary)  **FROM** Nurses **JOIN** Work **ON** Nurses.nurse\_id = Work.nurse\_id **JOIN** Rooms **ON** Work.room\_number = Rooms.room\_number **WHERE** Rooms.room\_number = 101; |
| --- |

1. Retrieve the name, phone number, and specialization of all doctors who treated more than 2 patients in the month of January:

| **SELECT** name, phone\_number, specialization  **FROM** Doctors **JOIN** Treat **ON** Doctors.doctor\_id = Treat.doctor\_id **JOIN** Appointments **ON** Treat.patient\_id = Appointments.patient\_id **WHERE** **MONTH**(date) = 1 **GROUP** **BY** Doctors.doctor\_id **HAVING** **COUNT**(**DISTINCT** Treat.patient\_id) > 2; |
| --- |

1. Write a query that returns the name and specialization of all doctors who have treated patients who have been discharged and have a phone number that contains the string "555".

| **SELECT** Rooms.room\_type, **COUNT**(**DISTINCT** Patients.patient\_id) total\_patients,  **AVG**(**DATEDIFF**(discharge\_date, admit\_date)) avg\_length\_of\_stay **FROM** Patients **INNER** **JOIN** Rooms **ON** Patients.patient\_id = Rooms.patient\_id **WHERE** discharge\_date **IS** **NOT** **NULL** **GROUP** **BY** Rooms.room\_type; |
| --- |

1. What are the unique addresses of patients who have had a prescription for a specific medication?

| **SELECT** **DISTINCT** address **FROM** Addresses **WHERE** patient\_id **IN** (**SELECT** patient\_id **FROM** Prescriptions **WHERE** medication = **'DOLO'**); |
| --- |

1. Write a query that gets the earliest and latest appointment dates, as well as the total patient count for appointments where the patient is currently not discharged and for appointments where the patient is discharged.

| **SELECT** **MIN**(date) **as** earliest\_appointment, **MAX**(date) **as** latest\_appointment, **SUM**(patient\_count) **as** total\_patient\_count **FROM**  (**SELECT** date, **COUNT**(patient\_id) **as** patient\_count  **FROM** Appointments  **WHERE** **EXISTS** (**SELECT** 1 **FROM** Patients **WHERE** Patients.patient\_id = Appointments.patient\_id **AND** discharge\_date **IS** **NULL**)  **GROUP** **BY** date  **UNION**  **SELECT** date, **COUNT**(patient\_id) **as** patient\_count **FROM** Appointments  **WHERE** **NOT** **EXISTS** (**SELECT** 1 **FROM** Patients **WHERE** Patients.patient\_id = Appointments.patient\_id **AND** discharge\_date **IS** **NULL**)  **GROUP** **BY** date) **as** appts; |
| --- |

*Stored Procedures:*

1. A stored procedure to retrieve all the appointments for a specific patient:

| **CREATE** **PROCEDURE** get\_patient\_appointments (**IN** patient\_id INT) **IS**  X Appointments%rowtype;  **CURSOR** C **IS**  **SELECT** a.appointment\_id, a.date, a.time  **FROM** Appointments a  **JOIN** Treat t **ON** a.patient\_id = t.patient\_id  **JOIN** Doctors d **ON** t.doctor\_id = d.doctor\_id  **WHERE** a.patient\_id = patient\_id; **BEGIN**  **FOR** X **IN** C **LOOP**  **SYS.DBMS\_OUTPUT.PUT\_LINE**(X.appointment\_id || ‘ ‘ || X.date || ‘ ‘ || X.time);  **END LOOP**;  **END**;  **/** |
| --- |

1. Procedure to update the phone number of a patient and all related doctors and nurses:

| **CREATE** **PROCEDURE** update\_patient\_phone\_number (**IN** patient\_id INT, **IN** phone\_number VARCHAR(10)) **IS**  X Doctors%rowtype;  Y Nurses%rowtype; **DECLARE** doctor\_id INT; **DECLARE** nurse\_id INT; **CURSOR** cur1 **IS**  **SELECT** doctor\_id  **FROM** Treat  **WHERE** patient\_id = patient\_id; **CURSOR** cur2 **IS**  **SELECT** nurse\_id  **FROM** Work **WHERE**  patient\_id = patient\_id;  **UPDATE** Patients **SET** phone\_number = phone\_number **WHERE** patient\_id = patient\_id;  **BEGIN**  **FOR** X **IN** cur1 **LOOP**  **UPDATE** Doctors **SET** phone\_number = phone\_number **WHERE** X.doctor\_id = doctor\_id;  **END** **LOOP**;  **FOR** Y **IN** cur2 **LOOP**  **UPDATE** Nurses **SET** phone\_number = phone\_number **WHERE** Y.nurse\_id = nurse\_id;  **END** **LOOP**; **END**;  **/** |
| --- |

1. Procedure to check if a patient has a room assigned, and if not, assign them one:

| **CREATE** **PROCEDURE** assign\_room(**IN** patient\_id INT) **IS**  X Rooms%rowtype; **DECLARE** room\_number INT; **DECLARE** cur\_patient\_id INT; **DECLARE** done INT **DEFAULT** **FALSE**; **CURSOR** cur **IS**  **SELECT** patient\_id **FROM** Rooms **WHERE** **availability** = **'Yes'** **ORDER** **BY** room\_number **LIMIT** 1; **BEGIN**  OPEN cur;  FETCH cur INTO room\_number;  IF done THEN  **SYS.DBMS\_OUTPUT.PUT\_LINE'Sorry, all rooms are currently occupied.'**);  ELSE  **UPDATE** Rooms **SET** patient\_id = patient\_id, **availability** = **'No'** **WHERE** room\_number = room\_number;  **SYS.DBMS\_OUTPUT.PUT\_LINE**(**'Room number '**, room\_number, **' has been assigned to patient ID '**, patient\_id);  **END** **IF**;  CLOSE cur; **END**;  **/** |
| --- |

*Triggers:*

1. Trigger to update the availability of a room when a patient is discharged:

| **CREATE** **TRIGGER** update\_room\_availability **AFTER** **UPDATE** **ON** Patients **FOR** **EACH** **ROW** **BEGIN** **DECLARE** room\_number INT; **DECLARE** availability VARCHAR(25);  IF OLD.discharge\_date IS **NOT NULL AND** NEW.discharge\_date IS **NULL THEN**  **SET** room\_number = (**SELECT** room\_number **FROM** Rooms **WHERE** patient\_id = OLD.patient\_id);  **SET** availability = **'No'**;  **UPDATE** Rooms **SET** availability = availability **WHERE** room\_number = room\_number; **END** **IF**; **END**;  **/** |
| --- |

1. Trigger to prevent patients from being discharged if they have an upcoming appointment:

| **CREATE** **TRIGGER** prevent\_discharge **BEFORE** **UPDATE** **ON** Patients **FOR** **EACH** **ROW** **BEGIN** **DECLARE** patient\_id INT; **DECLARE** discharge\_date DATE;  **SET** patient\_id = NEW.patient\_id; **SET** discharge\_date = NEW.discharge\_date;  IF discharge\_date IS NOT NULL THEN  IF EXISTS (**SELECT** appointment\_id **FROM** Appointments **WHERE** patient\_id = patient\_id **AND** date > discharge\_date) **THEN**   **RAISE\_APPLICATION\_ERROR**(**-20009**,**'Patient has an upcoming appointment and cannot be discharged.')**;  **END** **IF**; **END** **IF**; **END**;  **/** |
| --- |

1. A trigger to prevent a doctor from being deleted if they have ongoing appointments:

| **CREATE** **TRIGGER** prevent\_doctor\_deletion **BEFORE** **DELETE** **ON** Doctors **FOR** **EACH** **ROW** **BEGIN**  **DECLARE** appointment\_count INT;  **SELECT** **COUNT**(\*) **INTO** appointment\_count  **FROM** Appointments  **WHERE** doctor\_id = OLD.doctor\_id;  IF appointment\_count > 0 THEN   **RAISE\_APPLICATION\_ERROR**(**-20009**,**'Cannot delete a doctor with ongoing appointments')**;  **END** **IF**; **END**;  **/** |
| --- |