BIG DATA FOR MANAGERS AND ANALYTICS

ENTITY-RELATION DIAGRAM FOR HOSPITAL MANAGEMENT SYSTEM USING BOTTOM-UP APPROACH

SUBMITTED TO:

PROF. ASHOK HARNAL

SUBMITTED BY:

MEGHA GARG (045030)
SHOBHIT GUPTA (045041)
SANSKRITI BAHL (045050)

INTRODUCTION

This report presents a detailed analysis and development of an Entity-Relationship Diagram (ERD) for a comprehensive hospital management system. The ERD was created using a bottom-up approach, which involved a thorough examination of hospital operations, with a particular focus on patient registration processes and typical healthcare workflows.

In today's rapidly evolving healthcare landscape, efficient data management is crucial for providing high-quality patient care, streamlining administrative tasks, and ensuring regulatory compliance. A well-designed database system serves as the backbone of modern hospital operations, enabling seamless information flow across various departments and stakeholders.

Our approach to developing this ERD began with an in-depth analysis of patient registration forms. These forms serve as the primary point of data collection in most healthcare settings and provide valuable insights into the types of information essential

for patient management. By starting with these fundamental data points, we ensured that our database design would be grounded in practical, day-to-day hospital operations.

Furthermore, we considered the broader context of hospital functions, including appointment scheduling, doctor-patient interactions, treatment procedures, and prescription management. This holistic view allowed us to create an ERD that not only captures essential patient information but also facilitates efficient hospital operations across multiple domains. The resulting ERD represents a careful balance between comprehensiveness and simplicity. It aims to capture all critical aspects of hospital data management while maintaining a structure that is intuitive and easy to implement. The diagram includes key entities such as Patients, Doctors, Appointments, Treatments, and Prescriptions, along with their attributes and relationships.

Throughout this report, we will delve into the step-by-step process of creating this ERD, discuss the rationale behind our design choices, and explore the insights gained from the final diagram. We will also present the SQL schema derived from this ERD, demonstrating how the conceptual model translates into a practical database structure.

By providing a solid foundation for database design, this ERD aims to support the development of a robust hospital management system. Such a system can significantly enhance operational efficiency, improve patient care through better information management, and provide valuable data insights for strategic decision-making in healthcare settings.

As we progress through this report, we will see how each component of the ERD contributes to creating a comprehensive, scalable, and patient-centric database design that can adapt to the evolving needs of modern healthcare institutions.

OBJECTIVES

The primary objectives of this project were to:

- To create a comprehensive ERD for a hospital management system
- To identify key entities and their relationships within the hospital ecosystem
- To establish a solid foundation for database design and implementation
- To ensure a database structure supports efficient data management and retrieval

PROCEDURE FOR BOTTOM-UP APPROACH

The top-down approach in database system design begins with identifying the overall structure and high-level entities of the system, focusing on the big picture. Designers start by creating a conceptual model that outlines the primary entities and their relationships, often using tools like Entity-Relationship Diagrams (ERDs). This high-level design is then progressively broken down into more detailed components, such as specific attributes, tables, and relationships, ultimately leading to the physical design of the database. The approach emphasizes starting with a broad overview and systematically refining it into more detailed, implementable parts.

a) Data Collection: The first step in creating the ERD involved a thorough analysis of patient registration forms. These forms provided crucial insights into the types of information typically collected in a hospital setting. We carefully examined each field on these forms, considering their relevance and importance in the overall hospital management system. Additionally, we conducted a comprehensive review of typical hospital operations and workflows. This included studying appointment booking processes, treatment procedures, prescription handling, and doctor-patient interactions. By combining the insights from registration forms with an understanding of hospital

workflows, we ensured that our ERD would be grounded in practical, real-world requirements.

b) Entity Identification: Based on the collected data and operational insights, we identified five primary entities that form the core of our hospital management system: Patient, Doctor, Appointment, Treatment, and Prescription. Each entity represents a key component of hospital operations. For the Patient entity, we included attributes such as PatientID, PatientName, DOB, Gender, Address, and ContactNumber. The Doctor entity comprised DoctorID, DoctorName, Specialty, and ContactNumber. The Appointment entity included AppointmentID, AppointmentDate, AppointmentTime, and Reason. For Treatment, we defined TreatmentID, TreatmentName, and TreatmentCost. Lastly, the Prescription entity included attributes like PrescriptionID, MedicineName, Dosage, StartDate, and EndDate. These attributes were carefully selected to capture essential information while maintaining database efficiency.

c) Relationship Mapping: After identifying the entities, we proceeded to establish relationships between them. We determined that Patients can have multiple Appointments, each with one Doctor, represented by a many-to-one relationship. Prescriptions are linked to both Patients and Doctors, illustrating the prescribing process. We introduced a many-to-many relationship between Patients and Treatments through a junction table called PatientTreatment. For each relationship, we carefully considered the cardinality (one-to-one, one-to-many, or many-to-many) and participation constraints (total or partial participation). These considerations ensured that our ERD accurately reflected the real-world interactions between different entities in a hospital setting.

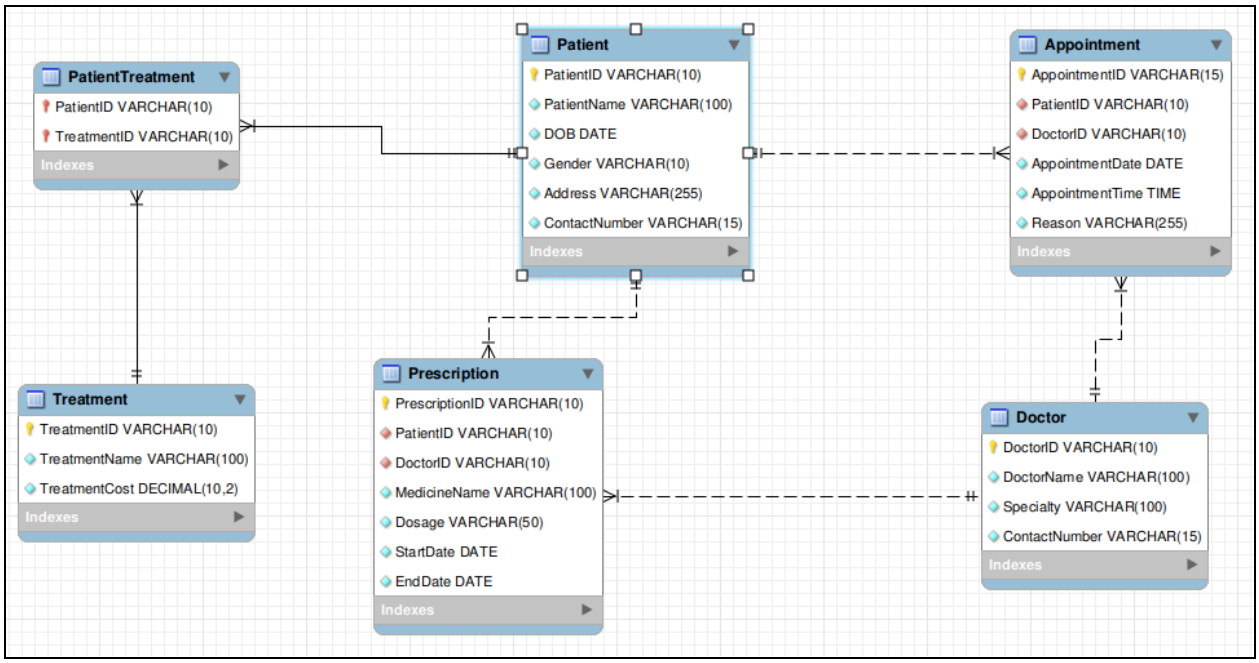
d) Primary and Foreign Key Assignment: To ensure data integrity and establish clear relationships between entities, we assigned primary keys to each entity. For instance, PatientID serves as the primary key for the Patient entity, while AppointmentID is the primary key for the Appointment entity. We then identified foreign keys to represent

relationships between entities. For example, the Appointment entity contains PatientID and DoctorID as foreign keys, linking it to the Patient and Doctor entities respectively. This step was crucial in maintaining referential integrity and enabling efficient data retrieval across the database.

e) ERD Creation (Conceptual Design): With entities, attributes, and relationships defined, we proceeded to draw the ERD using standard notation. We used rectangles to represent entities, ovals for attributes, and lines to show relationships between entities. Primary keys were underlined, and foreign keys were clearly indicated. We paid special attention to representing many-to-many relationships, such as the one between Patient and Treatment, by including the junction table PatientTreatment. After creating the initial diagram, we conducted a thorough verification process, checking each entity, attribute, and relationship to ensure they accurately represented the intended database structure and hospital operations.

f) SQL Schema Creation: The final step involved translating our ERD into SQL code for database creation. We wrote SQL statements to create each table, defining attributes with appropriate data types and constraints. Primary keys were established using the PRIMARY KEY constraint, and foreign keys were implemented using the FOREIGN KEY constraint, ensuring referential integrity. To test the schema and demonstrate its functionality, we added sample data points for each table. This included creating records for patients, doctors, appointments, treatments, and prescriptions, allowing us to verify that the database structure could effectively store and relate the required information.

ENTITY RELATION DIAGRAM



DATA ACCESS LANGUAGE

Data Access Language (DAL) is a set of commands or languages used to interact with databases, enabling users to retrieve, insert, update, and delete data. It serves as the interface for managing and accessing the data stored within a database system, ensuring that information can be efficiently manipulated and retrieved as needed.

1. **Patient Management User**

- **Entities**: Patient
- **Data Accessibility:** Full access to patient records, including personal details (PatientID, PatientName, DOB, Gender, Address, ContactNumber).
- Can view and update patient information as needed.
- **User:** patient_mgmt_user
- **Password:** patient_mgmt_*password*
- **Code:**

CREATE USER 'patient_mgmt_user'@'localhost' IDENTIFIED BY 'patient_mgmt_password';

GRANT SELECT, INSERT, UPDATE, DELETE ON HospitalDB.Patient TO 'patient_mgmt_user'@'localhost';

FLUSH PRIVILEGES;

```
mysql> SELECT * FROM Patient;
+-----------+---------------+------------+--------+----------------+----------------+
| PatientID | PatientName   | DOB        | Gender | Address        | ContactNumber  |
+-----------+---------------+------------+--------+----------------+----------------+
| P001      | John Doe      | 1985-05-12 | M      | 123 Elm St     | +91-9876543210 |
| P002      | Jane Smith    | 1990-07-23 | F      | 456 Oak Ave    | +91-9876543211 |
| P003      | Alice Johnson | 1975-02-15 | F      | 789 Pine Rd    | +91-9876543212 |
| P004      | Bob Brown     | 2000-10-10 | M      | 321 Maple Blvd | +91-9876543213 |
+-----------+---------------+------------+--------+----------------+----------------+
```

```
mysql> SELECT * FROM Patient WHERE PatientID = 'P001';
+-----------+-------------+------------+--------+------------+----------------+
| PatientID | PatientName | DOB        | Gender | Address    | ContactNumber  |
+-----------+-------------+------------+--------+------------+----------------+
| P001      | John Doe    | 1985-05-12 | M      | 123 Elm St | +91-9876543210 |
+-----------+-------------+------------+--------+------------+----------------+
```

```
mysql> UPDATE Patient SET      PatientName = 'John Wick',     DOB = '1985-01-01',     Gender = 'M',     Add
ress
Query OK, 1 row affected (0.15 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from Patient;
+-----------+--------------+------------+--------+---------------+----------------+
| PatientID | PatientName  | DOB        | Gender | Address       | ContactNumber  |
+-----------+--------------+------------+--------+---------------+----------------+
| P001      | John Wick    | 1985-01-01 | M      | New Address   | +91-9999999999 |
| P002      | Jane Smith   | 1990-07-23 | F      | 456 Oak Ave   | +91-9876543211 |
| P003      | Alice Johnson | 1975-02-15 | F      | 789 Pine Rd   | +91-9876543212 |
| P004      | Bob Brown    | 2000-10-10 | M      | 321 Maple Blvd | +91-9876543213 |
+-----------+--------------+------------+--------+---------------+----------------+
```

```
mysql> INSERT INTO Patient (PatientID, PatientName, DOB, Gender, Address, ContactNumber)
    -> VALUES ('P005', 'New Patient', '1995-09-15', 'F', 'New Address', '+91-9876543214');
Query OK, 1 row affected (0.13 sec)

mysql> select * from Patient;
+-----------+--------------+------------+--------+---------------+----------------+
| PatientID | PatientName  | DOB        | Gender | Address       | ContactNumber  |
+-----------+--------------+------------+--------+---------------+----------------+
| P001      | John Wick    | 1985-01-01 | M      | New Address   | +91-9999999999 |
| P002      | Jane Smith   | 1990-07-23 | F      | 456 Oak Ave   | +91-9876543211 |
| P003      | Alice Johnson | 1975-02-15 | F      | 789 Pine Rd   | +91-9876543212 |
| P004      | Bob Brown    | 2000-10-10 | M      | 321 Maple Blvd | +91-9876543213 |
| P005      | New Patient  | 1995-09-15 | F      | New Address   | +91-9876543214 |
+-----------+--------------+------------+--------+---------------+----------------+
```

```
mysql> DELETE FROM Patient WHERE PatientID = 'P005';
Query OK, 1 row affected (0.09 sec)

mysql> select * from Patient;
+-----------+--------------+------------+--------+---------------+----------------+
| PatientID | PatientName  | DOB        | Gender | Address       | ContactNumber  |
+-----------+--------------+------------+--------+---------------+----------------+
| P001      | John Wick    | 1985-01-01 | M      | New Address   | +91-9999999999 |
| P002      | Jane Smith   | 1990-07-23 | F      | 456 Oak Ave   | +91-9876543211 |
| P003      | Alice Johnson | 1975-02-15 | F      | 789 Pine Rd   | +91-9876543212 |
| P004      | Bob Brown    | 2000-10-10 | M      | 321 Maple Blvd | +91-9876543213 |
+-----------+--------------+------------+--------+---------------+----------------+
```

2. **Doctor/Staff Management User**

- **Entities:** Doctor
- **Data Accessibility:** Full access to doctor records, including details about their specialty and contact information.
- Responsible for managing doctor profiles and assigning them to patients.
- **User:** *doctor_mgmt_user*

- **Password:** *doctor_mgmt_user_password*
- **Code:**

CREATE USER 'doctor_mgmt_user'@'localhost' IDENTIFIED BY 'doctor_mgmt_user_password';

GRANT SELECT, INSERT, UPDATE, DELETE ON HospitalDB.Doctor TO 'doctor_mgmt_user'@'localhost';

FLUSH PRIVILEGES;

```
mysql> SELECT DoctorID, DoctorName, Specialty, ContactNumber
    -> FROM Doctor
    -> WHERE Specialty = 'Cardiology';
+----------+------------+------------+-----------------+
| DoctorID | DoctorName | Specialty  | ContactNumber   |
+----------+------------+------------+-----------------+
| D001     | Dr. Adams  | Cardiology | +91-9876543214 |
+----------+------------+------------+-----------------+
1 row in set (0.01 sec)
```

```
mysql> SELECT d.DoctorID, d.DoctorName, d.Specialty, COUNT(a.PatientID) AS NumberOfPatients
    -> FROM Doctor d
    -> LEFT JOIN Appointment a ON d.DoctorID = a.DoctorID
    -> GROUP BY d.DoctorID, d.DoctorName, d.Specialty;
+----------+------------+------------+-----------------+
| DoctorID | DoctorName | Specialty  | NumberOfPatients |
+----------+------------+------------+-----------------+
| D001     | Dr. Adams  | Cardiology |               1 |
| D002     | Dr. Brown  | Neurology  |               1 |
| D003     | Dr. Clark  | Orthopedics |              1 |
| D004     | Dr. Davis  | Pediatrics |               1 |
+----------+------------+------------+-----------------+
```

```
mysql> -- Update Doctor's Contact Information
mysql> UPDATE Doctor SET ContactNumber = '+91-9999999999' WHERE DoctorID = ('D002');
Query OK, 0 rows affected (0.00 sec)
Rows matched: 1  Changed: 0  Warnings: 0

mysql> select * from Doctor;
+----------+------------+-------------+----------------+
| DoctorID | DoctorName | Specialty   | ContactNumber  |
+----------+------------+-------------+----------------+
| D001     | Dr. Adams  | Cardiology  | +91-9876543214 |
| D002     | Dr. Brown  | Neurology   | +91-9999999999 |
| D003     | Dr. Clark  | Orthopedics | +91-9876543216 |
| D004     | Dr. Davis  | Pediatrics  | +91-9876543217 |
+----------+------------+-------------+----------------+
```

```
mysql> -- Insert a New Doctor Profile
mysql> INSERT INTO Doctor (DoctorID, DoctorName, Specialty, ContactNumber)
    -> VALUES ('D005', 'Dr. Evans', 'Dermatology', '+91-9876543218');
Query OK, 1 row affected (0.10 sec)

mysql> select * from Doctor;
+----------+------------+-------------+----------------+
| DoctorID | DoctorName | Specialty   | ContactNumber  |
+----------+------------+-------------+----------------+
| D001     | Dr. Adams  | Cardiology  | +91-9876543214 |
| D002     | Dr. Brown  | Neurology   | +91-9999999999 |
| D003     | Dr. Clark  | Orthopedics | +91-9876543216 |
| D004     | Dr. Davis  | Pediatrics  | +91-9876543217 |
| D005     | Dr. Evans  | Dermatology | +91-9876543218 |
+----------+------------+-------------+----------------+
5 rows in set (0.00 sec)
```

```
mysql> -- Delete a Doctor Profile if No Appointments Exist
mysql> DELETE FROM Doctor
    -> WHERE DoctorID = 'D005'
    -> AND NOT EXISTS (
    ->     SELECT 1
    ->     FROM Appointment
    ->     WHERE DoctorID = 'D005'
    -> );
Query OK, 1 row affected (0.19 sec)

mysql> select * from Doctor;
+----------+------------+-------------+-----------------+
| DoctorID | DoctorName | Specialty   | ContactNumber   |
+----------+------------+-------------+-----------------+
| D001     | Dr. Adams  | Cardiology  | +91-9876543214 |
| D002     | Dr. Brown  | Neurology   | +91-9999999999 |
| D003     | Dr. Clark  | Orthopedics | +91-9876543216 |
| D004     | Dr. Davis  | Pediatrics  | +91-9876543217 |
+----------+------------+-------------+-----------------+
```

3. **Appointments User**

- **Entities:** Appointment
- **Data Accessibility:** Full access to appointment records, including patient and doctor assignments, appointment dates, times, and reasons.
- Can schedule, reschedule, and cancel appointments.
- **User:** *appointments_user*
- **Password:** *appointments_user_password*
- **Code:**

  CREATE USER 'appointments_user'@'localhost' IDENTIFIED BY 'appointments_user_password';

  GRANT SELECT, INSERT, UPDATE, DELETE ON HospitalDB.Appointment TO 'appointments_user'@'localhost';

FLUSH PRIVILEGES;

```
mysql> -- Select All Appointments for a Specific Date
mysql> SELECT AppointmentID, PatientID, DoctorID, AppointmentTime, Reason
    -> FROM Appointment
    -> WHERE AppointmentDate = '2024-08-01';
+---------------+-----------+----------+-----------------+-----------------+
| AppointmentID | PatientID | DoctorID | AppointmentTime | Reason          |
+---------------+-----------+----------+-----------------+-----------------+
| JDDA001       | P001      | D001     | 09:00:00        | Routine Checkup |
+---------------+-----------+----------+-----------------+-----------------+
```

```
mysql> -- Insert a new Appointment
mysql> INSERT INTO Appointment (AppointmentID, PatientID, DoctorID, AppointmentDate, AppointmentTime, Reaso
n)
    -> VALUES ('BBDB005', 'P002', 'D001', '2024-08-06', '15:30', 'Follow-up Checkup');
Query OK, 1 row affected (0.28 sec)

mysql> select * from Appointment;
+---------------+-----------+----------+-----------------+-----------------+------------------+
| AppointmentID | PatientID | DoctorID | AppointmentDate | AppointmentTime | Reason           |
+---------------+-----------+----------+-----------------+-----------------+------------------+
| AJDC003       | P003      | D003     | 2024-08-03      | 11:00:00        | Knee Pain        |
| BBDB005       | P002      | D001     | 2024-08-06      | 15:30:00        | Follow-up Checkup |
| BBDD004       | P004      | D004     | 2024-08-04      | 13:00:00        | Child Checkup    |
| JDDA001       | P001      | D001     | 2024-08-01      | 09:00:00        | Routine Checkup  |
| JSDB002       | P002      | D002     | 2024-08-02      | 10:30:00        | Headache         |
+---------------+-----------+----------+-----------------+-----------------+------------------+
```

```
mysql> -- Select Upcoming Appointments with Patient and Doctor Details
mysql> SELECT a.AppointmentID, a.AppointmentDate, a.AppointmentTime, p.PatientName, d.DoctorName, a.Reason
    -> FROM Appointment a
    -> JOIN Patient p ON a.PatientID = p.PatientID
    -> JOIN Doctor d ON a.DoctorID = d.DoctorID
    -> WHERE a.AppointmentDate >= CURRENT_DATE
    -> ORDER BY a.AppointmentDate, a.AppointmentTime;
Empty set (0.18 sec)
```

```
mysql> -- Cancel an Appointment Using a Conditional Delete
mysql> DELETE FROM Appointment
    -> WHERE AppointmentID = 'BBDB005'
    -> AND AppointmentDate > CURRENT_DATE;
Query OK, 0 rows affected (0.00 sec)

mysql> select * from Appointment;
+---------------+-----------+----------+-----------------+-----------------+------------------+
| AppointmentID | PatientID | DoctorID | AppointmentDate | AppointmentTime | Reason           |
+---------------+-----------+----------+-----------------+-----------------+------------------+
| AJDC003       | P003      | D003     | 2024-08-03      | 11:00:00        | Knee Pain        |
| BBDB005       | P002      | D001     | 2024-08-06      | 15:30:00        | Follow-up Checkup |
| BBDD004       | P004      | D004     | 2024-08-04      | 13:00:00        | Child Checkup    |
| JDDA001       | P001      | D001     | 2024-08-01      | 09:00:00        | Routine Checkup  |
| JSDB002       | P002      | D002     | 2024-08-02      | 10:30:00        | Headache         |
+---------------+-----------+----------+-----------------+-----------------+------------------+
```

4. **Treatment Management User**

- **Entities:** Treatment and PatientTreatment
- **Data Accessibility:** Access to treatment records, including treatment types, costs, and patient-treatment assignments.
- Can manage the types of treatments offered and assign treatments to patients.
- **User:** *treatment_mgmt_user*
- **Password:** *treatment_mgmt_password*
- **Code:**

  CREATE USER 'treatment_mgmt_user'@'localhost' IDENTIFIED BY ' treatment_mgmt_password';

  GRANT SELECT, INSERT, UPDATE, DELETE ON HospitalDB.Treatment TO 'treatment_mgmt_user'@'localhost';

  GRANT SELECT, INSERT, UPDATE, DELETE ON HospitalDB.PatientTreatment TO 'treatment_mgmt_user'@'localhost';

  FLUSH PRIVILEGES;

```
mysql> -- Select All Treatment Types with Costs
mysql> SELECT TreatmentID, TreatmentName, TreatmentCost
    -> FROM Treatment;
+-------------+---------------+---------------+
| TreatmentID | TreatmentName | TreatmentCost |
+-------------+---------------+---------------+
| TREAT001    | Surgery       |       5000.00 |
| TREAT002    | Therapy       |       2000.00 |
| TREAT003    | Medication    |        150.00 |
| TREAT004    | X-Ray         |        300.00 |
+-------------+---------------+---------------+
```

```
mysql> -- Select Patients Assigned to a Specific Treatment
mysql> SELECT pt.PatientID, p.PatientName, t.TreatmentName, t.TreatmentCost
    -> FROM PatientTreatment pt
    -> JOIN Patient p ON pt.PatientID = p.PatientID
    -> JOIN Treatment t ON pt.TreatmentID = t.TreatmentID
    -> WHERE t.TreatmentID = 'TREAT002';
+-----------+-------------+---------------+---------------+
| PatientID | PatientName | TreatmentName | TreatmentCost |
+-----------+-------------+---------------+---------------+
| P002      | Jane Smith  | Therapy       |       2000.00 |
+-----------+-------------+---------------+---------------+
```

```
mysql> -- Update Treatment Cost Based on Treatment Name
mysql> UPDATE Treatment
    -> SET TreatmentCost = 5500.00
    -> WHERE TreatmentName = 'Surgery';
Query OK, 1 row affected (0.04 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from Treatment;
+-------------+---------------+---------------+
| TreatmentID | TreatmentName | TreatmentCost |
+-------------+---------------+---------------+
| TREAT001    | Surgery       |       5500.00 |
| TREAT002    | Therapy       |       2000.00 |
| TREAT003    | Medication    |        150.00 |
| TREAT004    | X-Ray         |        300.00 |
+-------------+---------------+---------------+
```

```
mysql> -- Insert a New Treatment Type
mysql> INSERT INTO Treatment (TreatmentID, TreatmentName, TreatmentCost)
    -> VALUES ('TREAT005', 'Physical Therapy', 2500.00);
Query OK, 1 row affected (0.01 sec)

mysql> select * from Treatment;
+-------------+------------------+---------------+
| TreatmentID | TreatmentName    | TreatmentCost |
+-------------+------------------+---------------+
| TREAT001    | Surgery          |       5500.00 |
| TREAT002    | Therapy          |       2000.00 |
| TREAT003    | Medication       |        150.00 |
| TREAT004    | X-Ray            |        300.00 |
| TREAT005    | Physical Therapy |       2500.00 |
+-------------+------------------+---------------+
```

```
mysql> -- Assign a Treatment to a Patient Using a Subquery
mysql> INSERT INTO PatientTreatment (PatientID, TreatmentID)
    -> SELECT 'P002', 'TREAT005'
    -> FROM Treatment
    -> WHERE TreatmentID = 'TREAT005';
Query OK, 1 row affected (0.02 sec)
Records: 1  Duplicates: 0  Warnings: 0

mysql> select * from PatientTreatment;
+-----------+-------------+
| PatientID | TreatmentID |
+-----------+-------------+
| P001      | TREAT001    |
| P002      | TREAT002    |
| P003      | TREAT003    |
| P004      | TREAT004    |
| P002      | TREAT005    |
+-----------+-------------+
```

```
mysql> -- Delete a Patient's Treatment Assignment
mysql> DELETE FROM PatientTreatment
    -> WHERE PatientID = 'P003'
    -> AND TreatmentID = 'TREAT003';
Query OK, 1 row affected (0.03 sec)

mysql> select * from PatientTreatment;
+-----------+-------------+
| PatientID | TreatmentID |
+-----------+-------------+
| P001      | TREAT001    |
| P002      | TREAT002    |
| P004      | TREAT004    |
| P002      | TREAT005    |
+-----------+-------------+
```

5. **Pharmacy Department User**

- **Entities:** Prescription
- **Data Accessibility:** Full access to prescription records, including patient and doctor assignments, medicine names, dosages, and prescription dates.
- Responsible for dispensing medications as per doctor prescriptions and managing medication inventory.
- **User:** *pharmacy_user*
- **Password:** *pharmacy_password*
- **Code:**

  CREATE USER 'pharmacy_user'@'localhost' IDENTIFIED BY 'pharmacy_password';

GRANT SELECT, INSERT, UPDATE, DELETE ON HospitalDB.Prescription TO 'pharmacy_user'@'localhost';

FLUSH PRIVILEGES;

```
mysql> -- Select All Prescriptions by a Specific Doctor
mysql> SELECT PrescriptionID, PatientID, MedicineName, Dosage, StartDate, EndDate
    -> FROM Prescription
    -> WHERE DoctorID = 'D001';
+----------------+-----------+--------------+--------+------------+------------+
| PrescriptionID | PatientID | MedicineName | Dosage | StartDate  | EndDate    |
+----------------+-----------+--------------+--------+------------+------------+
| PREP001        | P001      | Med1         | 10mg   | 2024-08-01 | 2024-08-10 |
+----------------+-----------+--------------+--------+------------+------------+
```

```
mysql> -- Select Current Prescriptions for a Specific Patient
mysql> SELECT PrescriptionID, MedicineName, Dosage, StartDate, EndDate
    -> FROM Prescription
    -> WHERE PatientID = 'P002'
    -> AND CURRENT_DATE BETWEEN StartDate AND EndDate;
Empty set (0.01 sec)
```

```
mysql> -- Update Dosage for a Specific Prescription
mysql> UPDATE Prescription
    -> SET Dosage = '15mg'
    -> WHERE PrescriptionID = 'PREP001';
Query OK, 1 row affected (0.02 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from Prescription;
+----------------+-----------+----------+--------------+--------+------------+------------+
| PrescriptionID | PatientID | DoctorID | MedicineName | Dosage | StartDate  | EndDate    |
+----------------+-----------+----------+--------------+--------+------------+------------+
| PREP001        | P001      | D001     | Med1         | 15mg   | 2024-08-01 | 2024-08-10 |
| PREP002        | P002      | D002     | Med2         | 5mg    | 2024-08-02 | 2024-08-12 |
| PREP003        | P003      | D003     | Med3         | 20mg   | 2024-08-03 | 2024-08-15 |
| PREP004        | P004      | D004     | Med4         | 15mg   | 2024-08-04 | 2024-08-20 |
+----------------+-----------+----------+--------------+--------+------------+------------+
```

```
mysql> -- Insert a New Prescription Record
mysql> INSERT INTO Prescription (PrescriptionID, PatientID, DoctorID, MedicineName, Dosage, StartDate, EndD
ate)
    -> VALUES ('PREP005', 'P003', 'D002', 'Med5', '10mg', '2024-08-05', '2024-08-15');
Query OK, 1 row affected (0.01 sec)

mysql> select * from Prescription;
+----------------+-----------+----------+--------------+--------+------------+------------+
| PrescriptionID | PatientID | DoctorID | MedicineName | Dosage | StartDate  | EndDate    |
+----------------+-----------+----------+--------------+--------+------------+------------+
| PREP001        | P001      | D001     | Med1         | 15mg   | 2024-08-01 | 2024-08-10 |
| PREP002        | P002      | D002     | Med2         | 5mg    | 2024-08-02 | 2024-08-12 |
| PREP003        | P003      | D003     | Med3         | 20mg   | 2024-08-03 | 2024-08-15 |
| PREP004        | P004      | D004     | Med4         | 15mg   | 2024-08-04 | 2024-08-20 |
| PREP005        | P003      | D002     | Med5         | 10mg   | 2024-08-05 | 2024-08-15 |
+----------------+-----------+----------+--------------+--------+------------+------------+
```

```
mysql> -- Delete Expired Prescriptions
mysql> DELETE FROM Prescription
    -> WHERE EndDate < CURRENT_DATE;
Query OK, 4 rows affected (0.02 sec)

mysql> select * from Prescription;
+----------------+-----------+----------+--------------+--------+------------+------------+
| PrescriptionID | PatientID | DoctorID | MedicineName | Dosage | StartDate  | EndDate    |
+----------------+-----------+----------+--------------+--------+------------+------------+
| PREP004        | P004      | D004     | Med4         | 15mg   | 2024-08-04 | 2024-08-20 |
+----------------+-----------+----------+--------------+--------+------------+------------+
```

```
mysql> -- Select Prescriptions and Group by Medicine Name
mysql> SELECT MedicineName, COUNT(PrescriptionID) AS NumberOfPrescriptions
    -> FROM Prescription
    -> GROUP BY MedicineName
    -> ORDER BY NumberOfPrescriptions DESC;
+--------------+------------------------+
| MedicineName | NumberOfPrescriptions |
+--------------+------------------------+
| Med4         |                      1 |
+--------------+------------------------+
```

INSIGHTS

1.1 Patient-Centric Design

Our ERD reflects a patient-centric approach to hospital management. The Patient entity serves as a central hub, directly connecting to the Appointment, Prescription, and PatientTreatment entities. This structure allows for comprehensive tracking of a patient's medical history, including their appointments, prescribed medications, and received

treatments. By centralizing patient information, the database facilitates quick access to critical data, enabling healthcare providers to make informed decisions and provide personalized care.

1.2 Flexible Appointment System

The Appointment entity in our ERD acts as a crucial link between patients and doctors. This design enables an efficient and flexible scheduling system. Each appointment record contains references to both the patient and the doctor involved, along with the date, time, and reason for the visit. This structure allows for easy querying of a doctor's schedule, tracking of patient visit history, and management of appointment slots. The flexibility built into this system can accommodate various appointment types and durations, adapting to the diverse needs of different medical departments.

1.3 Treatment and Prescription Management

Our ERD incorporates separate entities for Treatment and Prescription, allowing for detailed and organized medical record-keeping. The Treatment entity can store information about various medical procedures and their associated costs, while the Prescription entity maintains records of medications prescribed to patients. The inclusion of a PatientTreatment junction table enables a many-to-many relationship between patients and treatments. This design choice allows the system to track multiple treatments for each patient and multiple patients for each treatment type, providing a comprehensive view of the hospital's medical services and patient care history.

1.4 Scalable Structure

The ERD's design demonstrates considerable scalability, allowing for future growth and adaptation of the hospital management system. The clear separation of entities and their relationships provides a flexible foundation that can accommodate the addition of

new entities or attributes as the hospital's needs evolve. For instance, new specialties, departments, or types of medical equipment could be easily integrated into the existing structure without requiring a complete overhaul of the database design. This scalability ensures that the database can grow alongside the hospital, supporting long-term operational efficiency.

1.5 Data Integrity

A key strength of our ERD is its emphasis on maintaining data integrity throughout the database. The use of foreign keys in entities like Appointment, Prescription, and PatientTreatment ensures referential integrity across the database. This means that every reference to a patient, doctor, or treatment can be traced back to a valid record in the respective entity. Such a design prevents orphaned records and maintains the consistency and reliability of the data. This robust data integrity is crucial in a healthcare setting where accurate and reliable information is essential for patient care and operational efficiency.

CONCLUSION

The Entity-Relationship Diagram (ERD) developed for this hospital management system represents a significant milestone in our efforts to create a robust, efficient, and scalable database solution for healthcare institutions. By employing a bottom-up approach, starting from patient registration forms and expanding to encompass broader hospital operations, we have created a comprehensive yet flexible design that addresses the multifaceted needs of modern healthcare data management.

Our ERD successfully captures the core entities essential to hospital operations - Patients, Doctors, Appointments, Treatments, and Prescriptions - and establishes clear, meaningful relationships between them. This structure not only supports day-to-day

hospital functions but also lays the groundwork for advanced data analysis and reporting capabilities that can drive informed decision-making at all levels of hospital management.

One of the key strengths of this design is its patient-centric approach. By positioning the Patient entity at the center of our data model and connecting it to other crucial entities, we've ensured that comprehensive patient histories can be easily tracked and accessed. This approach aligns with the growing emphasis on personalized healthcare and can significantly enhance the quality of patient care.

The flexibility built into our ERD is another notable achievement. The design allows for easy addition of new entities or attributes as hospital needs evolve, ensuring that the database can grow and adapt alongside the institution it serves. This scalability is crucial in the rapidly changing landscape of healthcare technology and regulations.

Furthermore, the attention paid to data integrity through the careful assignment of primary and foreign keys ensures that the database will maintain consistent and reliable information. In a healthcare setting, where data accuracy can have life-altering implications, this focus on data integrity is paramount.

The translation of our ERD into SQL code, complete with sample data points, demonstrates the practical applicability of our design. This step bridges the gap between conceptual modeling and actual implementation, providing a clear path forward for database administrators and software developers. As we look to the future, this ERD serves as more than just a database design - it's a foundation for enhancing overall hospital operations. By enabling efficient data management and retrieval, it can contribute to reduced wait times, improved resource allocation, and enhanced patient satisfaction. The ability to easily access and analyze comprehensive patient data can also support research initiatives and contribute to advancements in medical care.

However, it's important to acknowledge that this ERD is a starting point, not an endpoint. As the database is implemented and used, ongoing evaluation and refinement

will be necessary. Feedback from healthcare professionals, administrators, and IT staff should be continuously incorporated to ensure the system remains aligned with the hospital's evolving needs. In conclusion, this ERD represents a thoughtful, comprehensive approach to hospital data management. It balances the complex requirements of healthcare operations with the need for a clear, implementable database structure. By providing a solid foundation for data organization and retrieval, this design has the potential to significantly improve hospital efficiency, enhance patient care, and support data-driven decision-making in healthcare settings.