HOSPITAL MANAGEMENT SYSTEM

Department of Computer Science and Engineering

National Institute of Technology, Durgapur
Semester: 4th Semester
Academic Year: 2025

Submitted by:

Name	Roll Number
PIJUS MONDAL	23CS8046
ANJALI CHAUDHARY	23CS8047
ATMAJO BURMAN	23CS8048
NAYANDEEP DAS	23CS8049
SWATI SARDAR	23CS8050

Under the guidance of:

DR. PRASENJIT CHOUDHURY(Associate Professor, National Institute Of Technology, Durgapur)

SUBMISSION DATE : 2025-04-08

Table of Contents

ABSTRACT

The **Hospital Management System (HMS)** is a database project developed as part of the **DBMS Laboratory** course to simplify and organize essential hospital-related operations. Its primary goal is to store and manage important information such as patient records, doctor details, and appointment schedules in a structured and efficient manner.

The project includes three major components—**Patients**, **Doctors**, and **Appointments**—which are linked through relational keys. With this setup, the system can handle various tasks like viewing appointment history, identifying patients who visit frequently, updating appointment statuses, and more. It also supports advanced operations such as finding which doctors have the most consultations, checking appointments for the upcoming week, and identifying patients with serious medical conditions.

Key database principles such as **data normalization**, **entity relationships**, **SQL queries**, **stored procedures**, and **triggers** have been applied throughout the project. The design allows for future improvements, including additional modules for billing, emergency tracking, or inventory systems, making it scalable and practical for real-life use.

Overall, the HMS project showcases how databases can be used to support hospital management by improving record-keeping and enabling quick access to critical information.

PROBLEM STATEMENT

Develop a database for a Hospital Management System.

Description: The system should track patient details, doctors, and appointments.

Tables:

• Patients (PatientID, Name, Age, Contact, Disease)

• Doctors (DoctorID, Name, Specialization, Contact)

• Appointments (AppointmentID, PatientID, DoctorID, AppointmentDate, Status)

Queries:

• Retrieve all appointments for a specific doctor.

• Find patients who have visited multiple times.

• Update appointment status after completion.

• Get a list of doctors with the most appointments.

• Identify patients with critical conditions.

• Retrieve appointments scheduled for the next week.

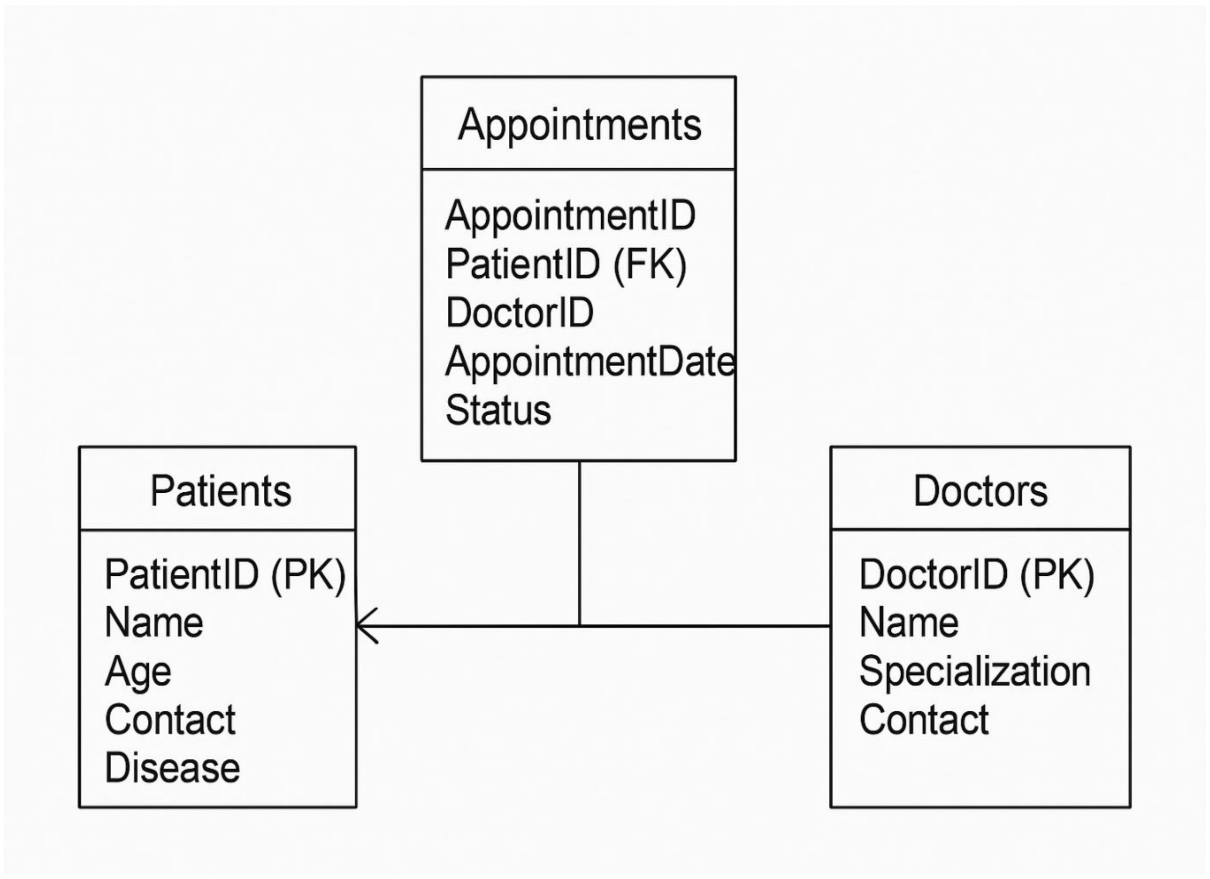
• Delete old patient records after five years.

• Calculate the total number of patients treated in a month.

• Find doctors available for emergency cases.

• Retrieve patients with the same disease.

ER DIAGRAM


Appointments
AppointmentID
PatientID (FK)
DoctorID
AppointmentDate
Status
Patients
PatientID (PK)
Name
Age
Contact
Disease
Doctors
DoctorID (PK)
Name
Specialization
Contact

SCHEMA DESIGN AND JUSTIFICATION

To build a reliable and manageable **Hospital Management System**, we designed a database schema that ensures efficient data handling and prevents inconsistency. The three main tables used are Patients, Doctors, and Appointments. Each of them is structured to support the system's objectives clearly and cleanly.

A. Normalization

We ensured that all tables follow **Third Normal Form (3NF)** principles:

- All fields contain simple, indivisible values.
- There's no repetition or duplication of data.
- Every non-key column depends only on the primary key of its table.

This setup helps avoid problems during data updates and keeps the database clean and consistent.

B. Separate Entities with Clear Primary Keys

Each table is focused on one main concept:

- Patients and Doctors are independent tables, each having its own unique identifier (PatientID, DoctorID).
- The Appointments table acts as a connection between patients and doctors. It logs each appointment using a unique AppointmentID and includes foreign keys to both the patient and the doctor involved.

C. Use of Foreign Keys for Data Integrity

The Appointments table includes:

- A PatientID that refers to a valid patient.
- A DoctorID that refers to a valid doctor.

This helps the database make sure that appointments are only created for real people and registered doctors, ensuring that no invalid data slips through.

D. Tracking and Management

We added fields like:

- Status in the Appointments table to show if the appointment is upcoming, completed, or canceled.
- AppointmentDate to help with time-based tracking and queries.

This allows the hospital to stay on top of daily operations and patient flow.

E. Room for Future Enhancements

The current schema is simple but powerful. It's easy to scale by adding:

- More information to patients (like address, gender, or insurance).
- New tables such as Billing, Rooms, or Departments.

Any future addition can be smoothly integrated without needing a full redesign.

F. Support for Useful Queries

Thanks to this structure, the system can quickly answer questions like:

- Which patients are visiting again?
- Which doctor has the most appointments?
- What appointments are scheduled for the coming week?
- Who are the critical condition patients?

This supports hospital staff in making better decisions quickly.

Final Thoughts

The schema we designed is solid and practical. It balances simplicity and functionality, making it ideal for hospital operations. With strong foundations like data normalization and foreign key constraints, this design ensures long-term reliability and easy maintenance.

SQL SCRIPTS

CREATE_TABLES.SQL

```
CREATE TABLE Patients (
   PatientID INT PRIMARY KEY,
   Name VARCHAR(100),
   Age INT,
   Contact VARCHAR(15),
   Disease VARCHAR(100)
);
```

```
CREATE TABLE Doctors (
    DoctorID INT PRIMARY KEY,
    Name VARCHAR(100),
    Specialization VARCHAR(100),
    Contact VARCHAR(15)
);

CREATE TABLE Appointments (
    AppointmentID INT PRIMARY KEY,
    PatientID INT,
    DoctorID INT,
    AppointmentDate DATE,
    Status VARCHAR(20),
    FOREIGN KEY (PatientID) REFERENCES Patients(PatientID),
    FOREIGN KEY (DoctorID) REFERENCES Doctors(DoctorID)
);
```

INSERT_DATA.SQL

```
INSERT INTO Patients (PatientID, Name, Age, Contact, Disease) VALUES
(1, 'Virat Kohli', 35, '9876543210', 'Flu'),
(2, 'Rohit Sharma', 36, '8765432109', 'Diabetes'),
(3, 'MS Dhoni', 42, '7654321098', 'Migraine'),
(4, 'KL Rahul', 31, '6543210987', 'Arthritis'),
(5, 'Hardik Pandya', 30, '5432109876', 'Asthma'),
(6, 'Shubman Gill', 25, '9321478065', 'Hypertension'),
(7, 'Ruturaj Gaikwad', 27, '9214578032', 'Allergy'),
(8, 'Sanju Samson', 30, '9891234567', 'Fever'),
(9, 'Ravindra Jadeja', 36, '9890011223', 'Thyroid'),
(10, 'Jasprit Bumrah', 31, '9765432101', 'Back Pain'),
(11, 'Yuzvendra Chahal', 33, '9123456789', 'Sinusitis'),
(12, 'Kuldeep Yadav', 29, '9012345678', 'Blood Pressure'),
(13, 'Suryakumar Yadav', 33, '9988776655', 'Flu'),
(14, 'Ishan Kishan', 26, '9876501234', 'Cold'),
(15,’Md. Shami’,39,’9564937999’,’Blood Pressure’);

INSERT INTO Doctors (DoctorID, Name, Specialization, Contact) VALUES
(101, 'Dr. Sharma', 'General Physician', '9998877665'),
(102, 'Dr. Verma', 'Cardiologist', '8887766554'),
(103, 'Dr. Rao', 'Neurologist', '7776655443'),
(104, 'Dr. Kapoor', 'Orthopedic', '6665544332'),
(105, 'Dr. Iyer', 'Pulmonologist', '9556677889'),
(106, 'Dr. Sen', 'Dermatologist', '9445566778'),
(107, 'Dr. Gupta', 'ENT Specialist', '9334455667'),
(108, 'Dr. Mehta', 'Gastroenterologist', '9223344556'),
(109, 'Dr. Das', 'Endocrinologist', '9112233445'),
(110, 'Dr. Abraham', 'Oncologist', '9001122334');
```

```
 INSERT INTO Appointments (AppointmentID, PatientID, DoctorID, AppointmentDate,
Status) VALUES
(1001, 1, 101, '2025-04-05', 'Completed'),
(1002, 2, 102, '2025-11-06', 'Scheduled'),
(1003, 3, 103, '2025-10-06', 'Cancelled'),
(1004, 4, 104, '2025-03-07', 'Scheduled'),
(1005, 5, 105, '2025-04-08', 'Critical'),
(1006, 6, 102, '2025-06-09', 'Scheduled'),
(1007, 7, 101, '2025-04-10', 'Emergency'),
(1008, 8, 105, '2025-10-11', 'Scheduled'),
(1009, 9, 106, '2025-11-12', 'Completed'),
(1010, 10, 107, '2025-10-12', 'Cancelled'),
(1011, 11, 108, '2025-04-13', 'Scheduled'),
(1012, 12, 109, '2025-11-14', 'Emergency'),
(1013, 13, 110, '2025-01-15', 'Scheduled'),
(1014, 14, 106, '2025-03-15', 'Critical'),
(1015, 15, 104, '2025-03-16', 'Completed'),
(1016, 3, 101, '2025-04-17', 'Scheduled'),
(1017, 5, 102, '2025-01-17', 'Critical'),
(1018, 7, 103, '2025-07-18', 'Emergency');
```

DELETE_REDUNDANT.SQL

```
DELETE FROM Appointments
WHERE Status='Cancelled';

DELETE FROM Patients
WHERE PatientID NOT IN (SELECT DISTINCT PatientID FROM Appointments);
```

STORED_PROCEDURE.SQL

```
#1. Get appointments by doctor name
DELIMITER $$
CREATE PROCEDURE GetAppointmentsByDoctor(IN docName VARCHAR(100))
BEGIN
   SELECT docName AS Doctor, AppointmentID, PatientID, AppointmentDate, Status
   FROM Appointments
   WHERE DoctorID IN (
      SELECT DoctorID FROM Doctors WHERE Name = docName
   );
END$$
DELIMITER ;

#2. Find patients who have visited multiple times
DELIMITER $$
```

```
CREATE PROCEDURE GetFrequentPatients()
BEGIN
   SELECT PatientID, COUNT(*) AS no_of_visits
   FROM Appointments
   GROUP BY PatientID
   HAVING no_of_visits > 1;
END$$
DELIMITER ;

#3. Update appointment status after completion.
DELIMITER $$
CREATE PROCEDURE UpdateAppointmentStatus(
   IN p_appointment_id INT,
   IN p_new_status VARCHAR(20)
)
BEGIN
   UPDATE Appointments
   SET Status = p_new_status
   WHERE AppointmentID = p_appointment_id;
END $$
DELIMITER ;

#4. Get a list of doctors with the most appointments.
DELIMITER $$
CREATE PROCEDURE GetDoctorsWithMaxAppointments()
BEGIN
   WITH T AS (
      SELECT DoctorID, COUNT(*) AS no_of_appointments
      FROM Appointments
      GROUP BY DoctorID
   )
   SELECT * FROM T
   WHERE no_of_appointments = (SELECT MAX(no_of_appointments) FROM T);
END$$
DELIMITER ;

#5. Identify patients with critical conditions.
DELIMITER $$
CREATE PROCEDURE GetCriticalPatients()
BEGIN
   WITH T AS (
      SELECT PatientID
      FROM Appointments
      WHERE Status = 'Critical'
   )
   SELECT DISTINCT P.PatientID, P.Name, P.Age, P.Contact, P.Disease
   FROM Patients P
```

```
    INNER JOIN T ON P.PatientID = T.PatientID;
END$$
DELIMITER ;

#6. Retrieve appointments scheduled for the next week.
DELIMITER $$
CREATE PROCEDURE GetNextWeekAppointments()
BEGIN
    SELECT A.AppointmentID, A.PatientID, P.Name AS PatientName,
           A.DoctorID, D.Name AS DoctorName, A.AppointmentDate, A.Status
    FROM Appointments A
    JOIN Patients P ON A.PatientID = P.PatientID
    JOIN Doctors D ON A.DoctorID = D.DoctorID
    WHERE A.AppointmentDate BETWEEN CURDATE() AND DATE_ADD(CURDATE(), INTERVAL
7 DAY)
          AND A.Status = 'Scheduled';
END $$
DELIMITER ;

#7. Delete old patient records after five years.
DELIMITER $$
CREATE PROCEDURE DeleteOldPatients()
BEGIN
    DELETE FROM Appointments
    WHERE PatientID IN (
        SELECT pid FROM (
            SELECT P.PatientID AS pid
            FROM Patients P
            LEFT JOIN Appointments A ON P.PatientID = A.PatientID
            GROUP BY P.PatientID
            HAVING MAX(IFNULL(A.AppointmentDate, '1900-01-01')) < DATE_SUB(CURDATE(),
INTERVAL 5 YEAR)
        ) AS temp
    );
DELETE FROM Patients
WHERE PatientID NOT IN (SELECT DISTINCT PatientID FROM Appointments);
END $$
DELIMITER ;


#8. Calculate the total number of patients treated in a month.
DELIMITER $$
CREATE PROCEDURE GetMonthlyAppointments(IN monthNum INT)
BEGIN
    SELECT monthNum AS Month, COUNT(*) AS no_of_patients
    FROM Appointments
    WHERE MONTH(AppointmentDate) = monthNum;
```

```
END$$
DELIMITER ;

#9. Find doctors available for emergency cases.
DELIMITER $$
CREATE PROCEDURE GetEmergencyDoctors()
BEGIN
  WITH T AS (
    SELECT DISTINCT DoctorID
    FROM Appointments
    WHERE Status = 'Emergency'
  )
  SELECT T.DoctorID, D.Name, D.Specialization, D.Contact
  FROM T
  INNER JOIN Doctors D ON T.DoctorID = D.DoctorID;
END$$
DELIMITER ;

#10. Retrieve patients with the same disease.
DELIMITER $$
CREATE PROCEDURE GetPatientsGroupedByDisease()
BEGIN
  SELECT Disease,
        GROUP_CONCAT(PatientID) AS PatientIDs
  FROM Patients
  GROUP BY Disease;
END$$
DELIMITER ;
```

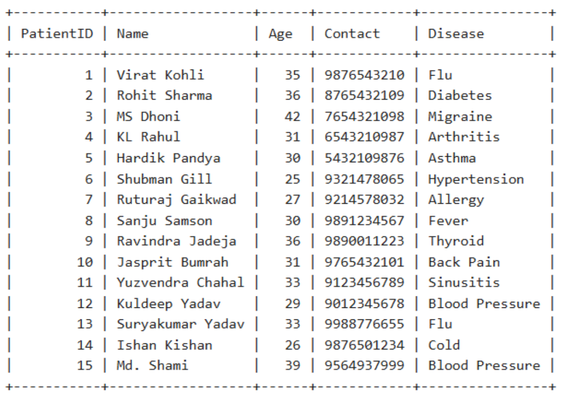
QUERY.SQL

```
CALL GetAppointmentsByDoctor('Dr. Iyer');
CALL GetFrequentPatients();
CALL UpdateAppointmentStatus(1002, 'Completed');
CALL GetDoctorsWithMaxAppointments();
CALL GetCriticalPatients();
CALL GetNextWeekAppointments();
CALL DeleteOldPatients();
CALL GetMonthlyAppointments(4); # April
CALL GetEmergencyDoctors();
CALL GetPatientsGroupedByDisease();
```

OUTPUTS

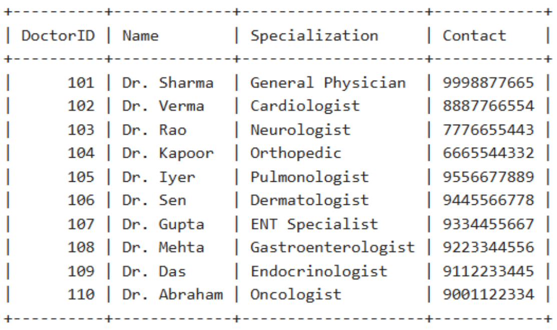
a) DATABASE SCHEMA:

Patients:



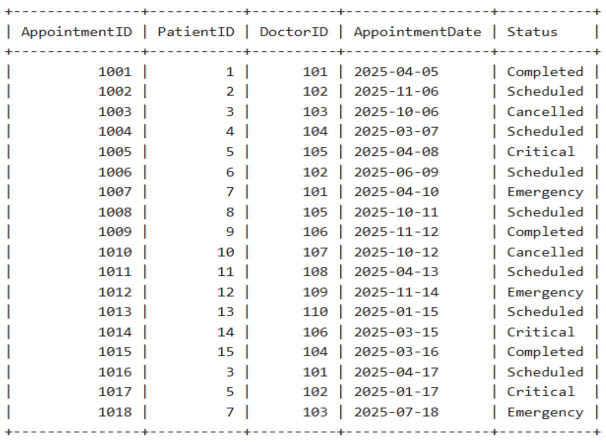
```
+-----------+------------------+-----+------------+----------------+
| PatientID | Name             | Age | Contact    | Disease        |
+-----------+------------------+-----+------------+----------------+
|         1 | Virat Kohli      |  35 | 9876543210 | Flu            |
|         2 | Rohit Sharma     |  36 | 8765432109 | Diabetes       |
|         3 | MS Dhoni         |  42 | 7654321098 | Migraine       |
|         4 | KL Rahul         |  31 | 6543210987 | Arthritis      |
|         5 | Hardik Pandya    |  30 | 5432109876 | Asthma         |
|         6 | Shubman Gill     |  25 | 9321478065 | Hypertension   |
|         7 | Ruturaj Gaikwad  |  27 | 9214578032 | Allergy        |
|         8 | Sanju Samson     |  30 | 9891234567 | Fever          |
|         9 | Ravindra Jadeja  |  36 | 9890011223 | Thyroid        |
|        10 | Jasprit Bumrah   |  31 | 9765432101 | Back Pain      |
|        11 | Yuzvendra Chahal |  33 | 9123456789 | Sinusitis      |
|        12 | Kuldeep Yadav    |  29 | 9012345678 | Blood Pressure |
|        13 | Suryakumar Yadav |  33 | 9988776655 | Flu            |
|        14 | Ishan Kishan     |  26 | 9876501234 | Cold           |
|        15 | Md. Shami        |  39 | 9564937999 | Blood Pressure |
+-----------+------------------+-----+------------+----------------+
```

Doctors:



```
+----------+-------------+--------------------+------------+
| DoctorID | Name        | Specialization     | Contact    |
+----------+-------------+--------------------+------------+
|      101 | Dr. Sharma  | General Physician  | 9998877665 |
|      102 | Dr. Verma   | Cardiologist       | 8887766554 |
|      103 | Dr. Rao     | Neurologist        | 7776655443 |
|      104 | Dr. Kapoor  | Orthopedic         | 6665544332 |
|      105 | Dr. Iyer    | Pulmonologist      | 9556677889 |
|      106 | Dr. Sen     | Dermatologist      | 9445566778 |
|      107 | Dr. Gupta   | ENT Specialist     | 9334455667 |
|      108 | Dr. Mehta   | Gastroenterologist | 9223344556 |
|      109 | Dr. Das     | Endocrinologist    | 9112233445 |
|      110 | Dr. Abraham | Oncologist         | 9001122334 |
+----------+-------------+--------------------+------------+
```

Appointments:



```
+---------------+-----------+----------+-----------------+-----------+
| AppointmentID | PatientID | DoctorID | AppointmentDate | Status    |
+---------------+-----------+----------+-----------------+-----------+
|          1001 |         1 |      101 | 2025-04-05      | Completed |
|          1002 |         2 |      102 | 2025-11-06      | Scheduled |
|          1003 |         3 |      103 | 2025-10-06      | Cancelled |
|          1004 |         4 |      104 | 2025-03-07      | Scheduled |
|          1005 |         5 |      105 | 2025-04-08      | Critical  |
|          1006 |         6 |      102 | 2025-06-09      | Scheduled |
|          1007 |         7 |      101 | 2025-04-10      | Emergency |
|          1008 |         8 |      105 | 2025-10-11      | Scheduled |
|          1009 |         9 |      106 | 2025-11-12      | Completed |
|          1010 |        10 |      107 | 2025-10-12      | Cancelled |
|          1011 |        11 |      108 | 2025-04-13      | Scheduled |
|          1012 |        12 |      109 | 2025-11-14      | Emergency |
|          1013 |        13 |      110 | 2025-01-15      | Scheduled |
|          1014 |        14 |      106 | 2025-03-15      | Critical  |
|          1015 |        15 |      104 | 2025-03-16      | Completed |
|          1016 |         3 |      101 | 2025-04-17      | Scheduled |
|          1017 |         5 |      102 | 2025-01-17      | Critical  |
|          1018 |         7 |      103 | 2025-07-18      | Emergency |
+---------------+-----------+----------+-----------------+-----------+
```

b) AFTER REMOVING REDUNDANT DATA:

Patients:

```
+-----------+-------------------+------+------------+----------------+
| PatientID | Name              | Age  | Contact    | Disease        |
+-----------+-------------------+------+------------+----------------+
|         1 | Virat Kohli       |   35 | 9876543210 | Flu            |
|         2 | Rohit Sharma      |   36 | 8765432109 | Diabetes       |
|         3 | MS Dhoni          |   42 | 7654321098 | Migraine       |
|         4 | KL Rahul          |   31 | 6543210987 | Arthritis      |
|         5 | Hardik Pandya     |   30 | 5432109876 | Asthma         |
|         6 | Shubman Gill      |   25 | 9321478065 | Hypertension   |
|         7 | Ruturaj Gaikwad   |   27 | 9214578032 | Allergy        |
|         8 | Sanju Samson      |   30 | 9891234567 | Fever          |
|         9 | Ravindra Jadeja   |   36 | 9890011223 | Thyroid        |
|        11 | Yuzvendra Chahal  |   33 | 9123456789 | Sinusitis      |
|        12 | Kuldeep Yadav     |   29 | 9012345678 | Blood Pressure |
|        13 | Suryakumar Yadav  |   33 | 9988776655 | Flu            |
|        14 | Ishan Kishan      |   26 | 9876501234 | Cold           |
|        15 | Md. Shami         |   39 | 9564937999 | Blood Pressure |
+-----------+-------------------+------+------------+----------------+
```

Appointments:

```
+---------------+-----------+----------+-----------------+-----------+
| AppointmentID | PatientID | DoctorID | AppointmentDate | Status    |
+---------------+-----------+----------+-----------------+-----------+
|          1001 |         1 |      101 | 2025-04-05      | Completed |
|          1002 |         2 |      102 | 2025-11-06      | Scheduled |
|          1004 |         4 |      104 | 2025-03-07      | Scheduled |
|          1005 |         5 |      105 | 2025-04-08      | Critical  |
|          1006 |         6 |      102 | 2025-06-09      | Scheduled |
|          1007 |         7 |      101 | 2025-04-10      | Emergency |
|          1008 |         8 |      105 | 2025-10-11      | Scheduled |
|          1009 |         9 |      106 | 2025-11-12      | Completed |
|          1011 |        11 |      108 | 2025-04-13      | Scheduled |
|          1012 |        12 |      109 | 2025-11-14      | Emergency |
|          1013 |        13 |      110 | 2025-01-15      | Scheduled |
|          1014 |        14 |      106 | 2025-03-15      | Critical  |
|          1015 |        15 |      104 | 2025-03-16      | Completed |
|          1016 |         3 |      101 | 2025-04-17      | Scheduled |
|          1017 |         5 |      102 | 2025-01-17      | Critical  |
|          1018 |         7 |      103 | 2025-07-18      | Emergency |
+---------------+-----------+----------+-----------------+-----------+
```

Get appointments by doctor name:

```
CALL GetAppointmentsByDoctor('Dr. Iyer');
+----------+---------------+-----------+-----------------+-----------+
| Doctor   | AppointmentID | PatientID | AppointmentDate | Status    |
+----------+---------------+-----------+-----------------+-----------+
| Dr. Iyer |          1005 |         5 | 2025-04-08      | Critical  |
| Dr. Iyer |          1008 |         8 | 2025-10-11      | Scheduled |
+----------+---------------+-----------+-----------------+-----------+
```

Find patients who have visited multiple times:

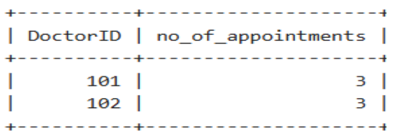
```
CALL GetFrequentPatients();
+-----------+--------------+
| PatientID | no_of_visits |
+-----------+--------------+
|         5 |            2 |
|         7 |            2 |
+-----------+--------------+
```

Update appointment status after completion.

```
CALL UpdateAppointmentStatus(1002, 'Completed');
|          1002 |         2 |      102 | 2025-11-06      | Completed |
```

Get a list of doctors with the most appointments.

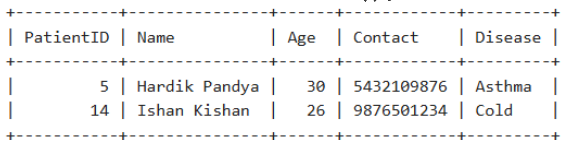
```
CALL GetDoctorsWithMaxAppointments();
```



```
+----------+---------------------+
| DoctorID | no_of_appointments |
+----------+---------------------+
|      101 |                   3 |
|      102 |                   3 |
+----------+---------------------+
```

Identify patients with critical conditions.

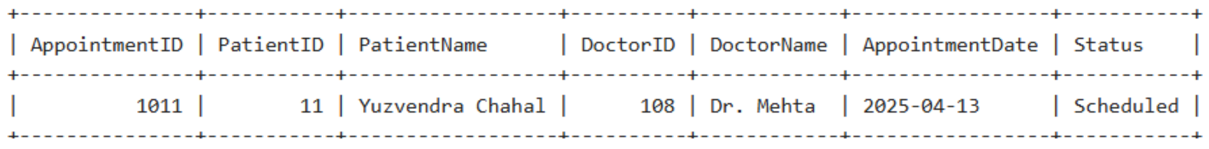
```
CALL GetCriticalPatients();
```



```
+-----------+--------------+-----+------------+---------+
| PatientID | Name         | Age | Contact    | Disease |
+-----------+--------------+-----+------------+---------+
|         5 | Hardik Pandya |  30 | 5432109876 | Asthma  |
|        14 | Ishan Kishan |  26 | 9876501234 | Cold    |
+-----------+--------------+-----+------------+---------+
```

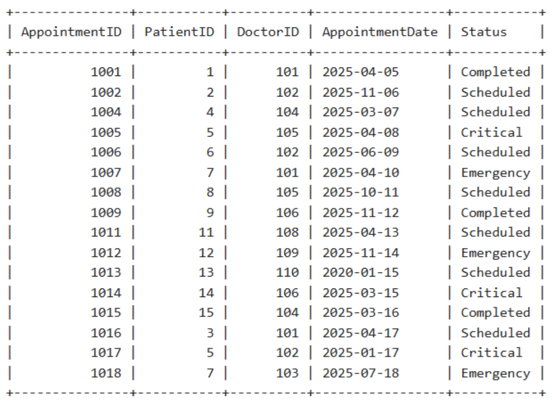
Retrieve appointments scheduled for the next week.

```
CALL GetNextWeekAppointments(); #Current Date: 2025-04-06
```

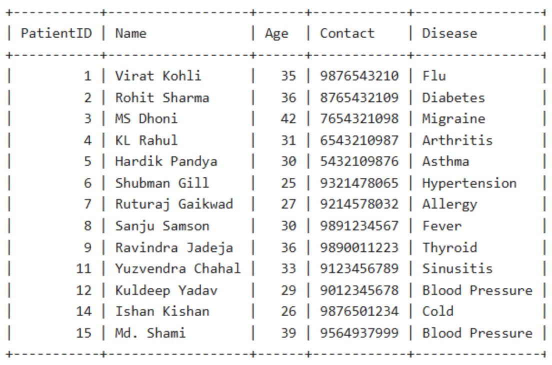


```
+---------------+-----------+------------------+----------+------------+-----------------+-----------+
| AppointmentID | PatientID | PatientName      | DoctorID | DoctorName | AppointmentDate | Status    |
+---------------+-----------+------------------+----------+------------+-----------------+-----------+
|          1011 |        11 | Yuzvendra Chahal |      108 | Dr. Mehta  | 2025-04-13      | Scheduled |
+---------------+-----------+------------------+----------+------------+-----------------+-----------+
```

```
Delete old patient records after five years.
CALL DeleteOldPatients();
Example:(Not the original Schema, only to demonstrate this procedure)
```



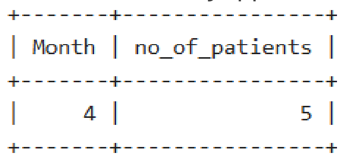
```
+---------------+-----------+----------+-----------------+-----------+
| AppointmentID | PatientID | DoctorID | AppointmentDate | Status    |
+---------------+-----------+----------+-----------------+-----------+
|          1001 |         1 |      101 | 2025-04-05      | Completed |
|          1002 |         2 |      102 | 2025-11-06      | Scheduled |
|          1004 |         4 |      104 | 2025-03-07      | Scheduled |
|          1005 |         5 |      105 | 2025-04-08      | Critical  |
|          1006 |         6 |      102 | 2025-06-09      | Scheduled |
|          1007 |         7 |      101 | 2025-04-10      | Emergency |
|          1008 |         8 |      105 | 2025-10-11      | Scheduled |
|          1009 |         9 |      106 | 2025-11-12      | Completed |
|          1011 |        11 |      108 | 2025-04-13      | Scheduled |
|          1012 |        12 |      109 | 2025-11-14      | Emergency |
|          1013 |        13 |      110 | 2020-01-15      | Scheduled |
|          1014 |        14 |      106 | 2025-03-15      | Critical  |
|          1015 |        15 |      104 | 2025-03-16      | Completed |
|          1016 |         3 |      101 | 2025-04-17      | Scheduled |
|          1017 |         5 |      102 | 2025-01-17      | Critical  |
|          1018 |         7 |      103 | 2025-07-18      | Emergency |
+---------------+-----------+----------+-----------------+-----------+
```



```
+-----------+------------------+-----+------------+----------------+
| PatientID | Name             | Age | Contact    | Disease        |
+-----------+------------------+-----+------------+----------------+
|         1 | Virat Kohli      |  35 | 9876543210 | Flu            |
|         2 | Rohit Sharma     |  36 | 8765432109 | Diabetes       |
|         3 | MS Dhoni         |  42 | 7654321098 | Migraine       |
|         4 | KL Rahul         |  31 | 6543210987 | Arthritis      |
|         5 | Hardik Pandya    |  30 | 5432109876 | Asthma         |
|         6 | Shubman Gill     |  25 | 9321478065 | Hypertension   |
|         7 | Ruturaj Gaikwad  |  27 | 9214578032 | Allergy        |
|         8 | Sanju Samson     |  30 | 9891234567 | Fever          |
|         9 | Ravindra Jadeja  |  36 | 9890011223 | Thyroid        |
|        11 | Yuzvendra Chahal |  33 | 9123456789 | Sinusitis      |
|        12 | Kuldeep Yadav    |  29 | 9012345678 | Blood Pressure |
|        14 | Ishan Kishan     |  26 | 9876501234 | Cold           |
|        15 | Md. Shami        |  39 | 9564937999 | Blood Pressure |
+-----------+------------------+-----+------------+----------------+
```

Note: Patient with PatientID=15 Has Been Dropped(Right)

```
Calculate the total number of patients treated in a month.
CALL GetMonthlyAppointments(4); # April
```



```
+-------+---------------+
| Month | no_of_patients |
+-------+---------------+
|     4 |             5 |
+-------+---------------+
```

```
Find doctors available for emergency cases.
CALL GetEmergencyDoctors();
```

```
+----------+------------+------------------+------------+
| DoctorID | Name       | Specialization   | Contact    |
+----------+------------+------------------+------------+
|      101 | Dr. Sharma | General Physician | 9998877665 |
|      103 | Dr. Rao    | Neurologist      | 7776655443 |
|      109 | Dr. Das    | Endocrinologist  | 9112233445 |
+----------+------------+------------------+------------+
```

Retrieve patients with the same disease.

```
CALL GetPatientsGroupedByDisease();
```

```
+----------------+------------+
| Disease        | PatientIDs |
+----------------+------------+
| Allergy        | 7          |
| Arthritis      | 4          |
| Asthma         | 5          |
| Blood Pressure | 12,15      |
| Cold           | 14         |
| Diabetes       | 2          |
| Fever          | 8          |
| Flu            | 1,13       |
| Hypertension   | 6          |
| Migraine       | 3          |
| Sinusitis      | 11         |
| Thyroid        | 9          |
+----------------+------------+
```

CONCLUSION

The Hospital Management System project was developed with the aim of organizing and simplifying the handling of crucial hospital-related data such as patient details, doctor records, and appointment scheduling. Through the use of a structured relational database, this system helps reduce manual errors, improves the speed of data access, and ensures that information remains accurate and consistent.

By applying principles like entity separation, normalization, and referential integrity, the database design ensures that data duplication is minimized while maintaining flexibility for future growth. The use of SQL queries, constraints, and procedures demonstrates a practical approach to real-world problems in hospital administration.

Overall, this project not only showcases our understanding of database concepts taught during the DBMS Laboratory course, but also reflects how digital systems can enhance healthcare efficiency. With further development, this system could incorporate modules for billing, inventory, staff management, and more, offering a complete solution to modern hospital data challenges.

REFERENCES

1. ***Korth, H. F., Silberschatz, A., & Sudarshan, S.*** (2010). *Database System Concepts* (6th ed.). McGraw-Hill.
   This book provided comprehensive insights into database design principles, normalization, and the fundamentals of relational database management systems.
2. ***OpenAI ChatGPT***. (2025). ChatGPT [Large language model]. OpenAI. Retrieved from https://openai.com/chatgpt
   ChatGPT was used to generate ideas, provide explanations, and assist in refining the project report and SQL scripts.