REPORT DOCUMENTATION

HOSPITAL MANAGEMENT SYSTEM

1. Project Overview

The Hospital Management System (HMS) is designed to efficiently manage patient records, doctor details, appointments, and billing information using a relational database. This system demonstrates how to perform database creation, table design, data manipulation, stored procedures, and triggers using MySQL.

2. Objectives

- Design a database that maintains hospital records systematically
- Ensure data integrity using foreign keys and constraints
- Perform CRUD (Create, Read, Update, Delete) operations
- Use SQL features such as aggregate functions, subqueries, stored procedures, and triggers
- Provide examples of real-world database interactions for learning purposes

3. Database Design

Database Name: HospitalDB

3.1 Tables and Their Purpose

- Patients: Stores personal details of patients
- Doctors: Stores information about doctors and their specialization
- Appointments: Maintains appointment details between patients and doctors
- Bills: Manages patient billing information

3.2 Table Structures

➢ **Patients Table**

- patient_id (PK, Auto Increment)
- name (VARCHAR)
- age (INT)
- gender (VARCHAR)
- contact (VARCHAR)
- address (VARCHAR)

- ➢ **Doctors Table**
- doctor_id (PK, Auto Increment)
- name (VARCHAR)
- specialization (VARCHAR)
- contact (VARCHAR)
- ➢ **Appointments Table**
- appointment_id (PK, Auto Increment)
- patient_id (FK → Patients.patient_id)
- doctor_id (FK → Doctors.doctor_id)
- appointment_date (DATE)
- status (VARCHAR)
- ➢ **Bills Table**
- bill_id (PK, Auto Increment)
- patient_id (FK → Patients.patient_id)
- amount (DECIMAL)
- bill_date (DATE**)**

4. SQL Operations Implemented

- **Database and Table Creation**: Created HospitalDB database and designed normalized tables with primary keys and foreign keys
- **Data Insertion**: Added sample records for patients, doctors, appointments, and bills
- **Data Update**: Updated appointment status from "Scheduled" to "Completed"
- **Data Deletion**: Deleted a bill entry using a DELETE query
- **Data Retrieval**: Used WHERE condition, aggregate functions with GROUP BY, HAVING clause, LIKE operator, and subqueries

5. Advanced SQL GetPatient Features

- **Procedure:** Bills retrieves all bill details of a particular patient
- **Procedure Name:** GetPatientBills
- **Input:** pid (Patient ID)
- **Function:** Retrieves Stored all bill details of a particular patient
- **Example:** CALL GetPatientBills(1);
- **Trigger:** BeforeBillInsert ensures that bill amounts cannot be negative by setting them to zero if invalid data is inserted
- **Trigger Name:** BeforeBillInsert
- **Ensures:** Bill amounts cannot be negative

6. Project Outputs

- **Patients above age 26:** Returns patient details with age greater than 26
- **Query**: SELECT * FROM Patients WHERE age > 26;
- **Appointments per doctor:** Returns the number of appointments handled by each doctor
- **Query:** SELECT doctor_id, COUNT(*) AS total_appointments FROM Appointments GROUP BY doctor_id;
- **Patients with high bill amounts:** Lists patients with bills greater than ₹4000
- **Query:** SELECT name FROM Patients WHERE patient_id IN (SELECT patient_id FROM Bills WHERE amount > 4000);

7. Key Benefits of the System

- Simplifies hospital record management
- Ensures data integrity using constraints and triggers
- Reduces manual errors by automating repetitive checks
- Makes querying and report generation fast and efficient

8. Future Enhancements

- Add a login and authentication system for doctors and staff
- Implement views for simplified report generation
- Integrate with a front-end application (PHP, Java, or Python)
- Add inventory management for medicines and equipment
- Enable role-based permissions using SQL GRANT and REVOKE

CONCLUSION:

The Hospital Management System project has successfully demonstrated the design and implementation of a relational database for managing hospital records, doctor details, appointments, and billing information. The system provides a robust and efficient way to manage hospital data, reducing manual errors and improving querying and report generation. With future enhancements, the system will become an even more valuable tool for hospital management