

COMSATS UNIVERSITY ISLAMABAD, ABBOTTABAD CAMPUS

DBMS Semester Project

Hospital Management System

*By:*

Hanzla Nouman FA21-BSE-015-4A

Laiba Binta Tahir FA21-BSE-019-4A

Muhammad Irfan FA21-BSE-027-4A

Arfah Ali FA21-BSE-080-4A

*Supervisor*

Dr. Rab Nawaz jadoon

Dated: 13 July 2023



EXECUTIVE SUMMARY

The collaboration and management of data in hospitals have been major challenges, leading to inefficiencies and compromised patient care. The existing manual systems result in discrepancies, delays, and hinder effective decision-making. To address these issues, we propose the development of a comprehensive Hospital Management System Database.

The Hospital Management System Database project aims to enhance collaboration, ensure data accuracy, and streamline hospital processes. By implementing a centralized database, stakeholders including doctors, nurses, administrators, and patients will have access to real-time information, improving operational efficiency and decision-making.

The system will include modules such as patient management, appointment scheduling, billing and payment, and reporting . These modules will enable healthcare professionals to manage patient records, schedule appointments, automate billing processes, and generate insightful reports.

By transitioning from manual record-keeping to a digital platform, the Hospital Management System Database will improve patient care, ensure data accuracy, and enhance overall efficiency. The system will facilitate seamless communication, resource allocation, and data-driven decision-making for administrators and healthcare professionals.

With the successful implementation of the Hospital Management System Database project, hospitals can expect improved operational efficiency, enhanced patient care, and secure data management. The proposed system aims to revolutionize hospital management practices, streamline processes, and empower healthcare professionals to provide quality care to patients.

Table Of Contents

[1 Introduction 7](#_Toc140218696)

[1.1 Relevance to Course Modules 7](#_Toc140218697)

[1.2 Project Background 7](#_Toc140218698)

[1.3 Literature Review 8](#_Toc140218699)

[2 Problem Definition 9](#_Toc140218700)

[2.1 Problem Statement 9](#_Toc140218701)

[2.2 Proposal 9](#_Toc140218702)

[3 ER Diagram 11](#_Toc140218703)

[3.1 Level 0 ERD Diagram 11](#_Toc140218704)

[3.2 Level 1 ERD Diagram 12](#_Toc140218707)

[3.3 Level 2 ERD Diagram 13](#_Toc140218710)

[3.4 Normalized ER Diagram 14](#_Toc140218713)

[4 Normalization 15](#_Toc140218716)

[5 Transition schema 16](#_Toc140218717)

[6. DDL statements: 19](#_Toc140218718)

List Of Figures

Figure 1 Level 0 ERD Diagram………………………………………………………………………………………………………………………………………11

Figure 2 Level 1 ERD Diagram………………………………………………………………………………………………………………………………………12

Figure 3 Level 2 ERD Diagram……………………………………………………………………………………………………………………………………….13

Figure 4 Normalized ERD Diagram……………………………………………………………………………………………………………………………….14

Glossary

|  |  |
| --- | --- |
| HMS | Hospital management system |
| ER Diagram | Entity Relationship diagram |
| DBMS | Database management system |
| DDL | Data definition language |

Hospital Management System

# Introduction

The Hospital Management System Database is a comprehensive project aimed at addressing the challenges faced in managing and organizing hospital data effectively. By implementing a centralized database, the system aims to enhance collaboration, ensure data accuracy, and streamline various hospital processes. This digital system will empower healthcare providers with efficient data management tools and enable improved patient care and administrative decision-making.

## Relevance to Course Modules

The project aligns with various concepts covered in the database system course. It encompasses data modeling techniques, including entity-relationship diagram (ERD), normalization, and the creation of a well-structured database schema. These concepts ensure efficient data organization, minimize redundancy, and maintain data integrity within the Hospital Management System Database.

## Project Background

Hospitals face significant challenges in managing and sharing data among different stakeholders, including doctors, nurses, administrators, and patients. The existing manual systems and fragmented data storage often result in inefficiencies, errors, and delays in accessing critical patient information. Furthermore, the lack of a centralized system hinders effective decision-making, resource allocation, and overall hospital management.

To overcome these challenges, the Hospital Management System Database is proposed. This system aims to create a unified platform for managing patient records, scheduling appointments, tracking medical inventory, and facilitating seamless communication between various departments. By adopting this digital solution, hospitals can enhance their operational efficiency, improve patient care, and ensure data accuracy.

## Literature Review

While there are several hospital management systems available, real-life implementations can vary based on the specific requirements and constraints of each healthcare institution. Extensive research has been conducted on the design and development of database systems for hospitals, focusing on data modeling, security, scalability, and interoperability. However, the implementation and customization of such systems to meet the unique needs of different hospitals remain crucial areas of exploration.

# Problem Definition

The challenges faced in hospital management include fragmented data storage, inefficient communication channels, and a lack of real-time access to critical information. These issues can lead to errors, delays, and compromised patient care. Additionally, the absence of an integrated system hampers effective decision-making and resource allocation.

## Problem Statement

To address these challenges, the proposed Hospital Management System Database aims to develop a centralized digital platform that facilitates seamless collaboration, data integrity, and efficient information exchange among various stakeholders. The system will provide healthcare professionals with quick and secure access to patient records, enable streamlined appointment scheduling, automate inventory management, and support decision-making through data analytics.

## Proposal

The proposed Hospital Management System Database will revolutionize hospital operations by transitioning from manual record-keeping to a digital platform. Key modules of the system will include:

a) Patient Management: Facilitate the storage, retrieval, and management of patient records, including medical history, diagnosis, treatment plans, and prescription details.

b) Appointment Scheduling: Enable patients to schedule appointments online, manage doctor availability, and send automated reminders to patients and healthcare providers.

c) Billing and Payment: Automate the billing process, generate invoices, and manage payments from patients, insurance providers, and other payers.

e) Reporting and Analytics: Provide data-driven insights, generate reports on key performance indicators, and support decision-making for administrators and healthcare professionals.

The software requirements for the project may include a database management system such as MySQL, programming languages like JavaScript, and web development frameworks.

By implementing the Hospital Management System Database, hospitals can significantly improve operational efficiency, enhance patient care, and ensure accurate and secure data management. The proposed system aims to revolutionize hospital management practices, streamline processes, and enable healthcare professionals to focus more on providing quality care to patients.

# ER Diagram

### 3.1 Level 0 ERD Diagram

### 3.1.2 Entities:

* Labs
* Patient
* Wards
* Doctor
* Reports
* MedicalHistory
* Bills
* Bed
* Treatment
* Pharmacy
* Schedule
* Prescription

### 3.1.3 Diagram:

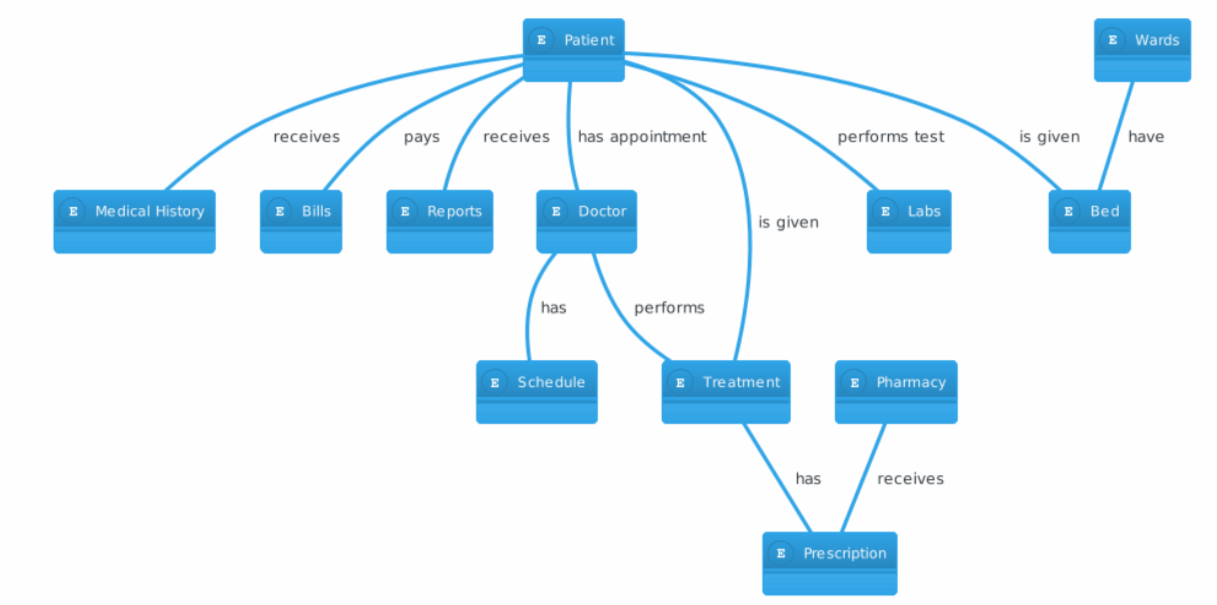


Figure 1 Level 0 ERD Diagram

### 3.2 Level 1 ERD Diagram

### 3.2.2 Entities:

* Labs
* Patient
* Wards
* Doctor
* Reports
* MedicalHistory
* Bills
* Bed
* Treatment
* Pharmacy
* Schedule
* Prescription

### 3.2.3 Diagram:

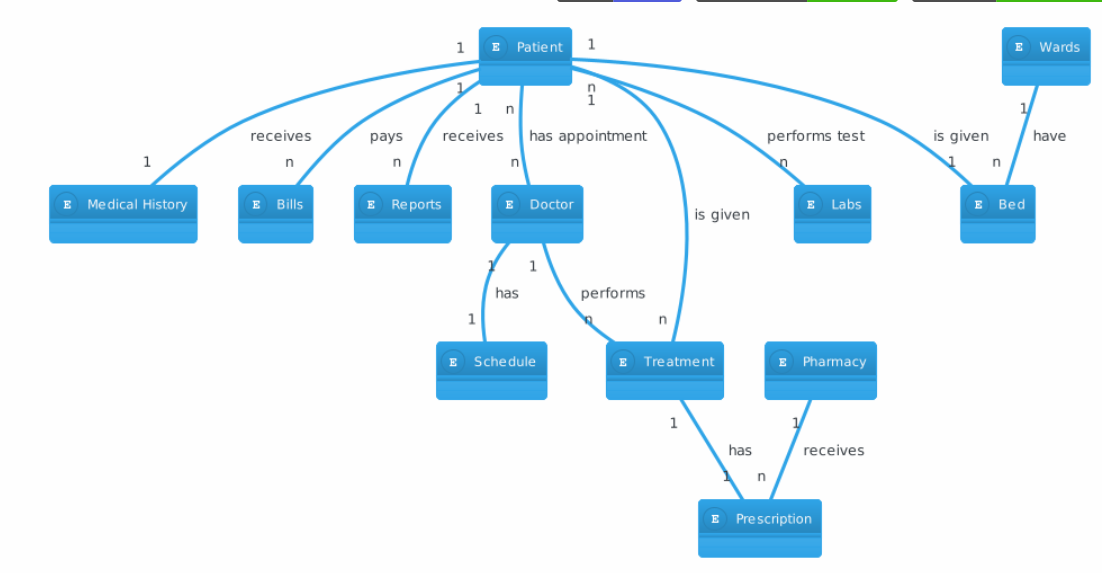


Figure 2 Level 1 ERD Diagram

### 3.3 Level 2 ERD Diagram

Resolved Many to Many Relations Of ERD

1. patient Doctor
2. Labs patient

### 3.3.2 Entities:

* Labs
* Patient
* Wards
* Doctor
* Reports
* MedicalHistory
* Bills
* Bed
* Treatment
* Pharmacy
* Schedule
* Prescription
* Appointments’
* Tests

### 3.3.3 Diagram:

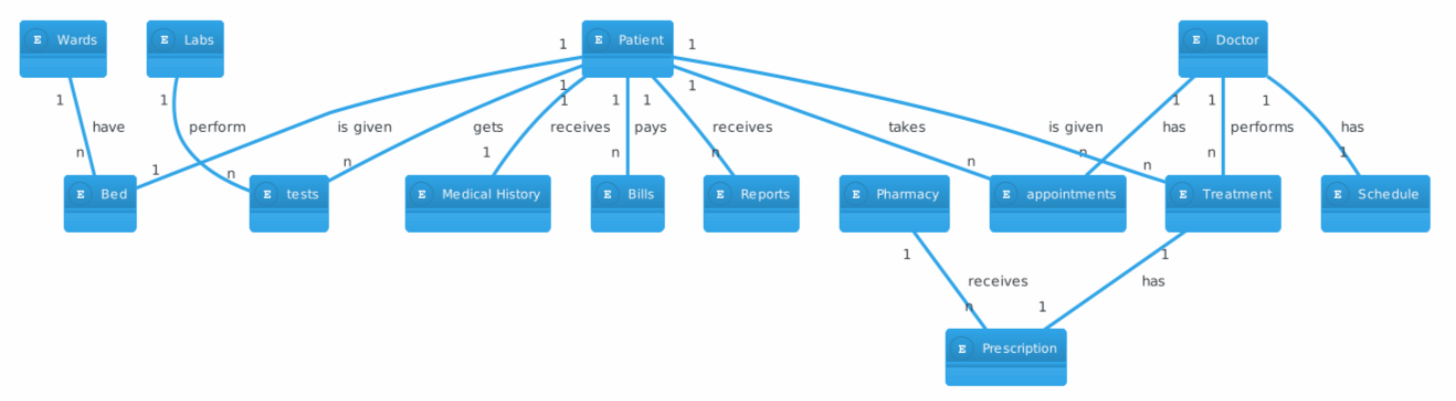


Figure 3 Level 2 ERD Diagram

### 3.4 Normalized ER Diagram

### 3.4.1 Description:

This is the final version of ER diagram.

### 3.4.2 Diagram:

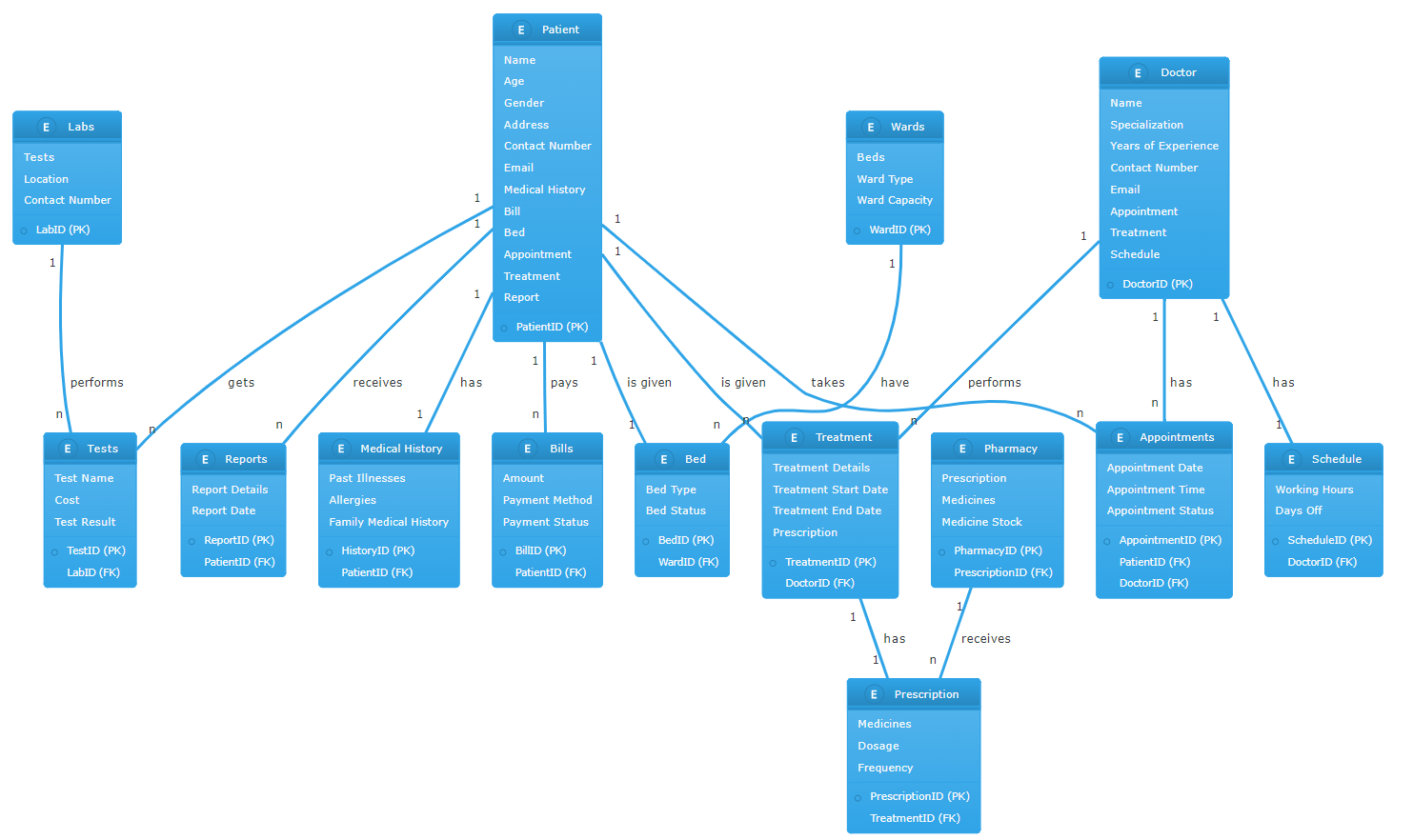
****

Figure 4 Normalized ER Diagram

# Normalization

1NF (First Normal Form):

A table is in 1NF if it satisfies the following conditions:

It contains only atomic (indivisible) values.

Each column contains only values that are of the same type.

Each column has a unique name.

The order in which data is stored does not matter.

In the context of these tables, all tables are in 1NF as they all have a primary key (Patient\_ID, Doctor\_ID, Treatment\_ID, etc.), all the values are atomic, each column contains values of the same type, and the order of data doesn't matter.

2NF (Second Normal Form):

A table is in 2NF if it is in 1NF and every non-key attribute is fully functionally dependent on the primary key. In other words, if for a table with composite primary keys, every non-key attribute must depend on the full set of primary key attributes.

In the context of these tables, all tables are in 2NF as well. This is because all the tables have a single column primary key and all other columns in each table are dependent on the primary key of that table.

3NF (Third Normal Form):

A table is in 3NF if it is in 2NF and there is no transitive dependency for non-prime attributes. This means that non-prime attributes (attributes that are not part of any candidate key) do not depend on other non-prime attributes.

In the context of these tables, all tables are in 3NF as well. This is because all non-prime attributes in each table are dependent only on the primary key of that table and not on any other non-prime attribute.

# Transition schema

|  |  |  |
| --- | --- | --- |
| **Labs** |  |  |
| **Field** | **Type** | **Extra** |
| LabID | int(11) | PK |
| Tests | varchar(255) |  |
| Location | varchar(255) |  |
| ContactNumber | varchar(15) |  |

|  |  |  |
| --- | --- | --- |
| **Tests** |  |  |
| **Field** | **Type** | **Extra** |
| TestID | int(11) | PK |
| TestName | varchar(255) |  |
| Cost | decimal(10,2) |  |
| TestResult | varchar(255) |  |
| LabID | int(11) | FK -> Labs |

|  |  |  |
| --- | --- | --- |
| **Reports** |  |  |
| **Field** | **Type** | **Extra** |
| ReportID | int(11) | PK |
| ReportDetails | text |  |
| ReportDate | date |  |
| PatientID | int(11) | FK -> Patient |

|  |  |  |
| --- | --- | --- |
| **MedicalHistory** |  |  |
| **Field** | **Type** | **Extra** |
| HistoryID | int(11) | PK |
| PastIllnesses | text |  |
| Allergies | text |  |
| FamilyMedicalHistory | text |  |
| PatientID | int(11) | FK -> Patient |

|  |  |  |
| --- | --- | --- |
| **Patient** |  |  |
| **Field** | **Type** | **Extra** |
| PatientID | int(11) | PK |
| Name | varchar(255) |  |
| Age | int(3) |  |
| Gender | varchar(10) |  |
| Address | varchar(255) |  |
| ContactNumber | varchar(15) |  |
| Email | varchar(255) |  |

|  |  |  |
| --- | --- | --- |
| **Bills** |  |  |
| **Field** | **Type** | **Extra** |
| BillID | int(11) | PK |
| Amount | decimal(10,2) |  |
| PaymentMethod | varchar(50) |  |
| PaymentStatus | varchar(50) |  |
| PatientID | int(11) | FK -> Patient |

|  |  |  |
| --- | --- | --- |
| **Bed** |  |  |
| **Field** | **Type** | **Extra** |
| BedID | int(11) | PK |
| BedType | varchar(50) |  |
| BedStatus | varchar(50) |  |
| WardID | int(11) | FK -> Wards |

|  |  |  |
| --- | --- | --- |
| **Treatment** |  |  |
| **Field** | **Type** | **Extra** |
| TreatmentID | int(11) | PK |
| TreatmentDetails | text |  |
| StartDate | date |  |
| EndDate | date |  |
| DoctorID | int(11) | FK -> Doctor |

|  |  |  |
| --- | --- | --- |
| **Pharmacy** |  |  |
| **Field** | **Type** | **Extra** |
| PharmacyID | int(11) | PK |
| Medicines | text |  |
| MedicineStock | int(11) |  |
| PrescriptionID | int(11) | FK -> Prescription |

|  |  |  |
| --- | --- | --- |
| **Wards** |  |  |
| **Field** | **Type** | **Extra** |
| WardID | int(11) | PK |
| WardType | varchar(50) |  |
| WardCapacity | int(11) |  |

|  |  |  |
| --- | --- | --- |
| **Doctor** |  |  |
| **Field** | **Type** | **Extra** |
| DoctorID | int(11) | PK |
| Name | varchar(255) |  |
| Specialization | varchar(100) |  |
| YearsOfExperience | int(3) |  |
| ContactNumber | varchar(15) |  |
| Email | varchar(255) |  |

|  |  |  |
| --- | --- | --- |
| **Appointments** |  |  |
| **Field** | **Type** | **Extra** |
| AppointmentID | int(11) | PK |
| AppointmentDate | date |  |
| AppointmentTime | time |  |
| AppointmentStatus | varchar(50) |  |
| PatientID | int(11) | FK -> Patient |
| DoctorID | int(11) | FK -> Doctor |

|  |  |  |
| --- | --- | --- |
| **Schedule** |  |  |
| **Field** | **Type** | **Extra** |
| ScheduleID | int(11) | PK |
| WorkingHours | varchar(50) |  |
| DaysOff | varchar(100) |  |
| DoctorID | int(11) | FK -> Doctor |

|  |  |  |
| --- | --- | --- |
| **Prescription** |  |  |
| **Field** | **Type** | **Extra** |
| PrescriptionID | int(11) | PK |
| Medicines | text |  |
| Dosage | varchar(100) |  |
| Frequency | varchar(50) |  |
| TreatmentID | int(11) | FK -> Treatment |

# DDL statements:

CREATE TABLE Labs (

LabID INT PRIMARY KEY,

Tests VARCHAR(255),

Location VARCHAR(255),

ContactNumber VARCHAR(15)

);

CREATE TABLE Tests (

TestID INT PRIMARY KEY,

TestName VARCHAR(255),

Cost DECIMAL(10,2),

TestResult VARCHAR(255),

LabID INT,

FOREIGN KEY (LabID) REFERENCES Labs(LabID)

);

CREATE TABLE Reports (

ReportID INT PRIMARY KEY,

ReportDetails TEXT,

ReportDate DATE,

PatientID INT

-- Foreign key PatientID is referenced after creating Patient table.

);

CREATE TABLE MedicalHistory (

HistoryID INT PRIMARY KEY,

PastIllnesses TEXT,

Allergies TEXT,

FamilyMedicalHistory TEXT,

PatientID INT

-- Foreign key PatientID is referenced after creating Patient table.

);

CREATE TABLE Patient (

PatientID INT PRIMARY KEY,

Name VARCHAR(255),

Age INT,

Gender VARCHAR(10),

Address VARCHAR(255),

ContactNumber VARCHAR(15),

Email VARCHAR(255)

);

-- Adding the foreign key references to the 'Reports' and 'MedicalHistory' tables

ALTER TABLE Reports

ADD FOREIGN KEY (PatientID) REFERENCES Patient(PatientID);

ALTER TABLE MedicalHistory

ADD FOREIGN KEY (PatientID) REFERENCES Patient(PatientID);

CREATE TABLE Bills (

BillID INT PRIMARY KEY,

Amount DECIMAL(10,2),

PaymentMethod VARCHAR(50),

PaymentStatus VARCHAR(50),

PatientID INT,

FOREIGN KEY (PatientID) REFERENCES Patient(PatientID)

);

CREATE TABLE Wards (

WardID INT PRIMARY KEY,

WardType VARCHAR(50),

WardCapacity INT

);

CREATE TABLE Bed (

BedID INT PRIMARY KEY,

BedType VARCHAR(50),

BedStatus VARCHAR(50),

WardID INT,

FOREIGN KEY (WardID) REFERENCES Wards(WardID)

);

CREATE TABLE Doctor (

DoctorID INT PRIMARY KEY,

Name VARCHAR(255),

Specialization VARCHAR(100),

YearsOfExperience INT,

ContactNumber VARCHAR(15),

Email VARCHAR(255)

);

CREATE TABLE Treatment (

TreatmentID INT PRIMARY KEY,

TreatmentDetails TEXT,

StartDate DATE,

EndDate DATE,

DoctorID INT,

FOREIGN KEY (DoctorID) REFERENCES Doctor(DoctorID)

);

CREATE TABLE Prescription (

PrescriptionID INT PRIMARY KEY,

Medicines TEXT,

Dosage VARCHAR(100),

Frequency VARCHAR(50),

TreatmentID INT,

FOREIGN KEY (TreatmentID) REFERENCES Treatment(TreatmentID)

);

CREATE TABLE Pharmacy (

PharmacyID INT PRIMARY KEY,

Medicines TEXT,

MedicineStock INT,

PrescriptionID INT,

FOREIGN KEY (PrescriptionID) REFERENCES Prescription(PrescriptionID)

);

CREATE TABLE Appointments (

AppointmentID INT PRIMARY KEY,

AppointmentDate DATE,

AppointmentTime TIME,

AppointmentStatus VARCHAR(50),

PatientID INT,

DoctorID INT,

FOREIGN KEY (PatientID) REFERENCES Patient(PatientID),

FOREIGN KEY (DoctorID) REFERENCES Doctor(DoctorID)

);

CREATE TABLE Schedule (

ScheduleID INT PRIMARY KEY,

WorkingHours VARCHAR(50),

DaysOff VARCHAR(100),

DoctorID INT,

FOREIGN KEY (DoctorID) REFERENCES Doctor(DoctorID)

);