**[Hospital Database Management System (DBMS)]**

Project submitted to the

         SRM University – AP, Andhra Pradesh

for the partial fulfilment of the requirements to award the degree of

**Bachelor of Technology**

      In

**Computer Science and Engineering**

**School of Engineering and Sciences**

  Submitted by: K. Hema



**SRM University–AP**

**NeeruKonda, Mangalagiri, Guntur**

**Andhra Pradesh – 522 240**

**[May 2024]**

**INDEX**

* Abstract, Aim, Entity, Attributes, Relations 3-5
* Relationships 6
* ER-Model Diagram 7
* Relational Diagram 8-9
* Normalization 10-15
* Entity records 16-18
* Queries 18-19
* Views 20-21

**HOSPITAL DATABASE MANAGEMENT SYSTEM**

**Abstract**

In modern healthcare settings, hospitals face the challenge of managing complex operations efficiently while delivering high-quality patient care. The Hospital Management System (HMS) is a sophisticated software solution that leverages a Database Management System (DBMS) to centralize and digitize various aspects of hospital operations. This abstract explores the integration of different modules within the HMS, each utilizing the DBMS to streamline functions such as patient management, appointment scheduling, inventory control, pharmacy management, laboratory information systems, and billing and invoicing. By facilitating seamless coordination between these modules, the HMS aims to revolutionize hospital operations, improve workflow efficiency, enhance patient care delivery, and ensure compliance with regulatory standards.

**Aim**

The aim of the Hospital Database Management System is to revolutionize hospital operations by harnessing modern technology to enhance efficiency, improve patient care, and ultimately enhance patient outcomes.

**Entity**

* It is a “thing” or “object” in the enterprise that is distinguishable from other objects

Described by a set of attributes

* List of Entities in the database:

**1. Patients:**

   - This entity stores information about the individuals who receive medical care at the hospital.

   - Each patient has a unique identifier called PatientID.

  - For each patient, we record details such as their Name, Age, Gender, Contact (phone number), and Problem (what health problem they have).

**2. Doctors:**

   - This entity represents the medical professionals who provide care at the hospital.

   - Each doctor has a unique identifier called DoctorID.

   - For each doctor, we record details such as their Name, Age, Specialization (the area of medicine they focus on), Salary, and Contact.

**3. Appointments:**

   - This entity keeps track of scheduled meetings between patients and doctors.

   - Each appointment is uniquely identified by an AppointmentID.

   - For each appointment, we record the PatientID (identifying the patient), DoctorID (identifying the doctor), Date, Time, and Status (whether it's Scheduled, Cancelled, or Completed).

**4. Medical Records:**

   - This entity stores detailed information about the medical history and treatment of patients.

   - Each medical record has a unique identifier called RecordID.

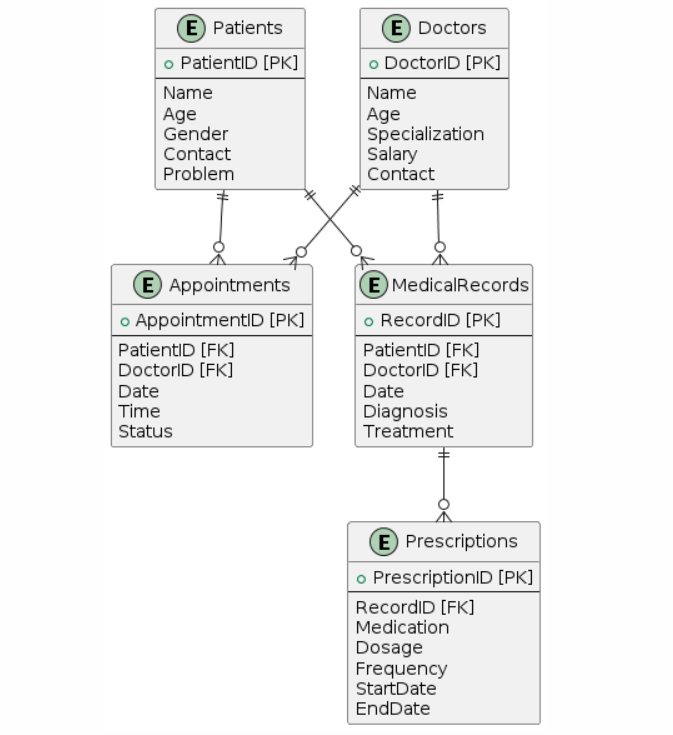
   - For each medical record, we record the PatientID (identifying the patient), DoctorID (identifying the doctor who provided the treatment), Date (when the record was created), Diagnosis (the medical condition or illness diagnosed), and Treatment (the course of action taken by the doctor).

**5. Prescriptions:**

- This entity records the medications prescribed to patients as part of their treatment.

- Each prescription has a unique identifier called PrescriptionID.

- For each prescription, we record the RecordID (identifying the medical record to which the prescription belongs), Medication (the name of the prescribed medicine), Dosage (the amount of medicine to be taken), Frequency (how often the medicine should be taken), Start Date, and End Date (the duration for which the medicine is prescribed).

****

**Attributes**

* They are characteristics of an entity, and has an oval symbol.
* There are different types of attributes:
* Key attribute: An attribute uniquely distinguishes the entity in an entity set.
* Simple attribute: An attribute that cannot be further subdivided into components.
* Composite attribute: An attribute that can be split into components.
* Single-valued attribute: The attribute which takes up only a single value for each entity instance.
* Multivalued attribute: The attribute which takes up more than a single value for each entity instance.
* Stored attribute: An attribute that stores the data which can be used to get the derived attribute.
* Derived attribute: An attribute that can be derived from other attributes.
* Attributes for each entity in the hospital database:

1. **Patient:** PatientId, Name, Age, Gender, Contact, Problem.
2. **Doctor**: DoctorID, Name, Age, Specialization, Salary, Contact.
3. **Appointments**: AppointmentId, PatientId, DoctorId, Date, Time, Status.
4. **Medical Records**: RecordId, PatientId, DoctorId, Date, Diagnosis, Treatment.
5. **Prescriptions**: PrescriptionId, RecordID, Medication, Dosage, Frequency, StartDate, EndDate.

**Relationships**

* A relationship is an association among several entities
* Relationships for each in hospital database

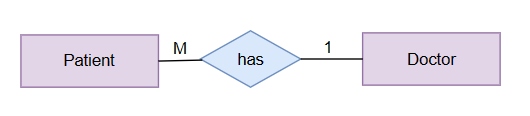
1. A patient can have medical records.
2. A patient can have a problem.
3. A doctor can have medical records.
4. A doctor can give appointments.
5. An appointment can have a medication.
6. A prescription can have a medication.

**Relations:**

1. **Patient has Doctor**

Relation: has

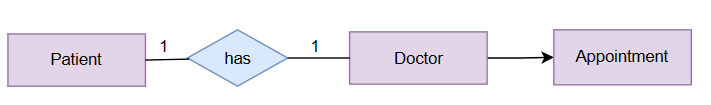
Cardinality: many to one



1. **Patient has Doctor Appointment**

Relation: has

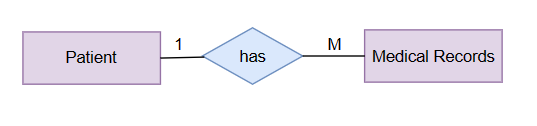
Cardinality: one to one



1. **Patient has Medical Records**

Relation: has

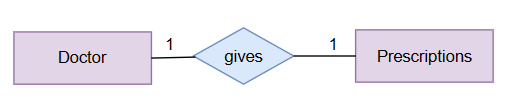
Cardinality: one to many



1. **Doctor gives Prescription**

Relation: gives

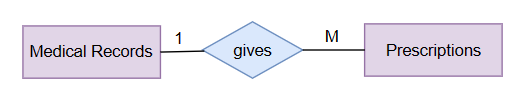
Cardinality: one to one

****

1. **Medical Records have Prescriptions**

Relation: have

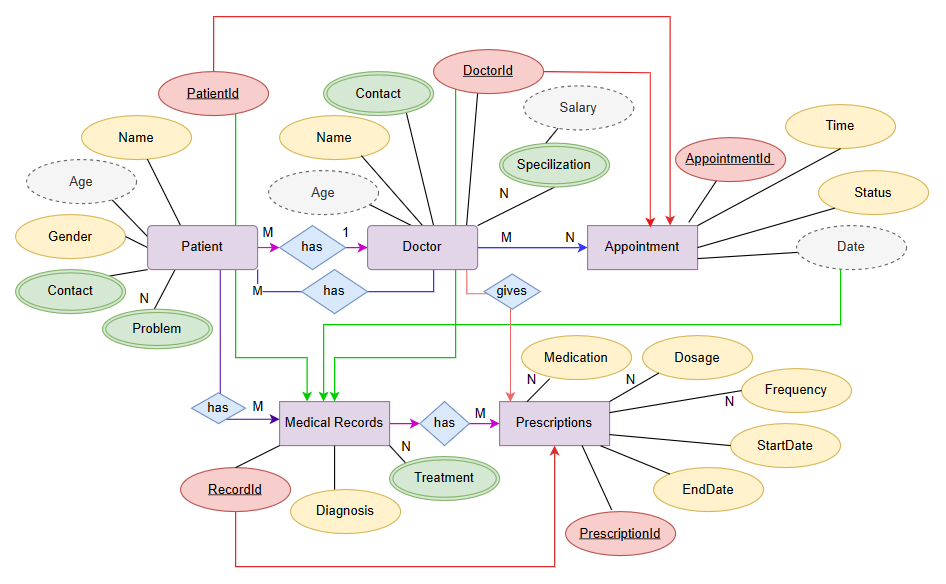
Cardinality: one to many

****

**E-R Model**

* ER model stands for an Entity-Relationship model.
* It is a high-level data model. This model is used to define the data elements and relationships for a specified system.
* It develops a conceptual design for the database. It also develops a very simple and easy-to-design view of data.

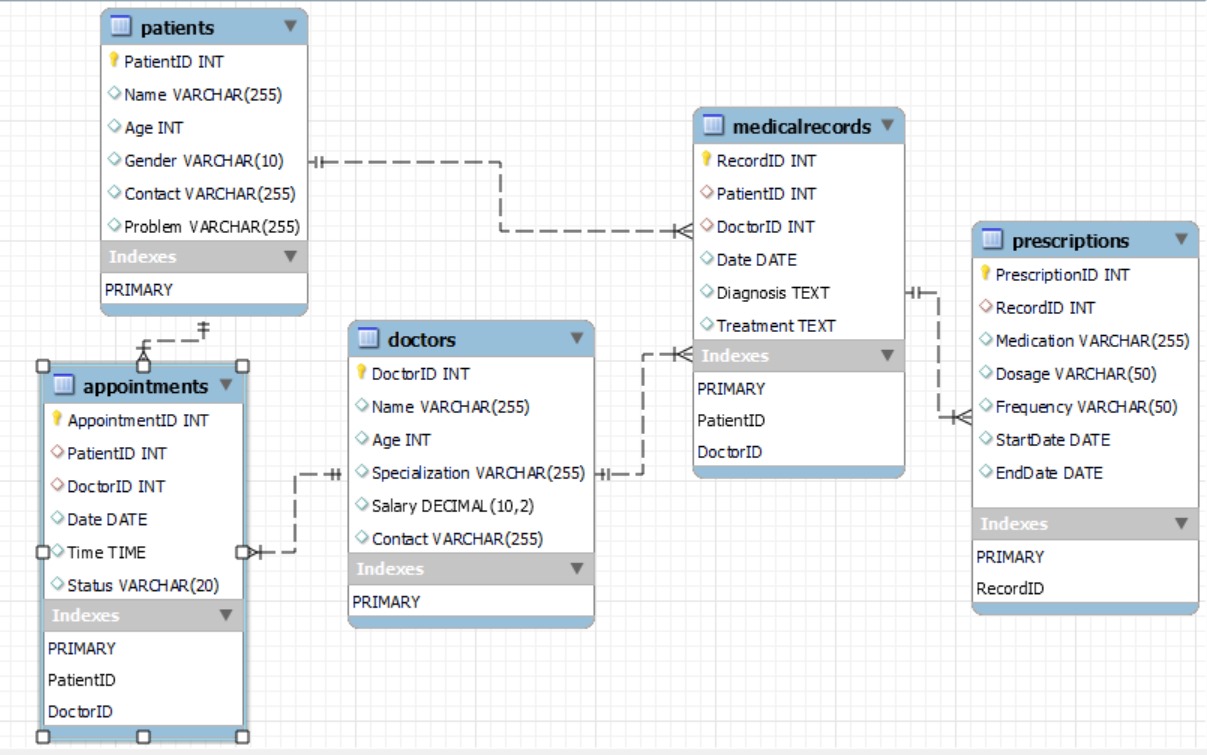
**ER Diagram**

****

**Relational Diagram**

Converting ER model to tables/relations, commonly used, flexible.

Each and every column header is called an attribute. The row header is called a tuple.

****

**Conversion of ER diagram into Tables**

**Creating Table Patients**

CREATE TABLE Patients (

    PatientID INT PRIMARY KEY AUTO\_INCREMENT,

    Name VARCHAR(255),

    Age INT,

    Gender VARCHAR(10),

    Contact VARCHAR(255),

    Problem VARCHAR(255)

);

**Creating Table  Doctors**

CREATE TABLE Doctors (

    DoctorID INT PRIMARY KEY AUTO\_INCREMENT,

    Name VARCHAR(255),

    Age INT,

    Specialization VARCHAR(255),

    Salary DECIMAL(10,2),

    Contact VARCHAR(255)

);

**Creating Table Appointments**

CREATE TABLE Appointments (

    AppointmentID INT PRIMARY KEY AUTO\_INCREMENT,

    PatientID INT,

    DoctorID INT,

    Date DATE,

    Time TIME,

    Status VARCHAR(20),

    FOREIGN KEY (PatientID) REFERENCES Patients(PatientID),

    FOREIGN KEY (DoctorID) REFERENCES Doctors(DoctorID)

);

**Creating Table  Medical Records**

CREATE TABLE MedicalRecords (

    RecordID INT PRIMARY KEY AUTO\_INCREMENT,

    PatientID INT,

    DoctorID INT,

    Date DATE,

    Diagnosis TEXT,

    Treatment TEXT,

    FOREIGN KEY (PatientID) REFERENCES Patients(PatientID),

    FOREIGN KEY (DoctorID) REFERENCES Doctors(DoctorID)

);

**Creating Table Prescriptions**

CREATE TABLE Prescriptions (

    PrescriptionID INT PRIMARY KEY AUTO\_INCREMENT,

    RecordID INT,

    Medication VARCHAR(255),

    Dosage VARCHAR(50),

    Frequency VARCHAR(50),

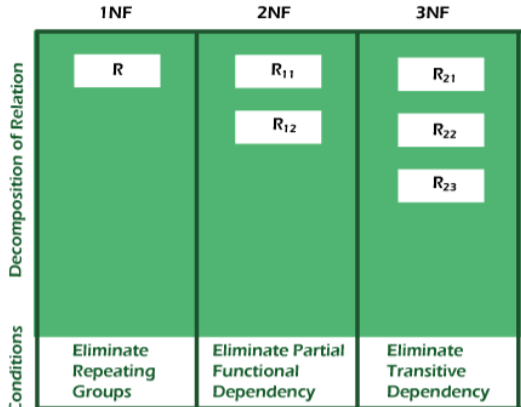
    StartDate DATE,

    EndDate DATE,

    FOREIGN KEY (RecordID) REFERENCES MedicalRecords(RecordID)

);

**Normalization**



Normalization is used to minimize the redundancy from a relation or set of relations.

**1. First Normal Form (1NF):**

A relation is said to be in its First Normal form if it has got no non-atomic attribute.

(Non-atomic attribute means the attribute which can’t be subdivided).

**2. Second Normal Form (2NF):**

A relation that is in 1NF is said to have a second normal form if it satisfies any

one of the following conditions.

a. The primary key contains only one attribute.

b. There exist no non-key attributes.

c. Every non-key attribute present in the relation should functionally depend

upon a full set of the primary key.

**3. Third Normal Form (3NF):**

The relation in 2Nf is said to be 3NF if there exists no transitive dependency of

any non-key attribute on the set of the primary key.

Normalization of Database:

1. **Patients Table (PatientId, Name, Age, Gender, Contact, Problem)**

1NF: Meets the 1NF because it has no non-atomic attribute.

2NF: It doesn’t meet the 2NF because the non-prime attribute is not fully functionally dependent on the entire primary key.

3NF: This is not in 3N due to the existence of the transitive dependency.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **PATIENT ID** | **NAME** | **AGE** | **GENDER** | **CONTACT** | **PROBLEM** |
| 1001 | Amulya | 35 | Female | 9876543210 | Sprained Ankle |
| 1002 | Rajesh | 28 | Male | 9876543211 | Spinal Cord Injury |
| 1003 | Ajay | 40 | Male | 9876543212 | Chest pain |
| 1004 | Priya | 10 | Female | 9876543213 | Fever, Cough |

**The decomposition of the Patient’s table into 1NF is shown below:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **PATIENT ID** | **NAME** | **AGE** | **GENDER** | **CONTACT** | **PROBLEM** |
| 1001 | Amulya | 35 | Female | 9876543210 | Sprained Ankle |
| 1002 | Rajesh | 28 | Male | 9876543211 | Spinal Cord Injury |
| 1003 | Ajay | 40 | Male | 9876543212 | Chest Pain |
| 1004 | Priya | 10 | Female | 9876543213 | Fever |
| 1004 | Priya | 10 | Female | 9876543213 | Cough |

Solution: Split the last column into 2 parts

1. **Doctor’s Table (DoctorId, Name, Age, Specialization, Salary, Contact)**

1NF: Meets the 1NF because it has no non-atomic attribute.

2NF: Meets the 2NF Rule-1 The primary key contains only one attribute.

3NF: This is not in 3N due to the existence of the transitive dependency.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **DOCTOR ID** | **NAME** | **AGE** | **SPECIALIZATION** | **SALARY** | **CONTACT** |
| 1 | Dr.Vikram Malhotra | 32 | Cardiology | 150000 | 6305094874 |
| 1 | Dr.Vikram Malhotra | 32 | Dermatology | 50000 | 6305094874 |
| 2 | Dr.Rahul Kapoor | 58 | Neurology | 150000 | 6304585693 |
| 3 | Dr.Sneha Joshi | 41 | Pediatrics | 10000 | 8074126426 |
| 4 | Dr.Pooja Verma | 29 | Orthopedics | 15000 | 9876543203 |

**To convert the given table into 2NF, we decompose it into two tables:**

Doctor Details:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **DOCTOR ID** | **NAME** | **AGE** | **CONTACT** | **SALARY** |
| 1 | Dr.Vikram Malhotra | 32 | 6305094874 | 150000 |
| 2 | Dr.Rahul Kapoor | 58 | 6304585693 | 150000 |
| 3 | Dr.Sneha Joshi | 41 | 8074126426 | 10000 |
| 4 | Dr.Pooja Verma | 29 | 9876543203 | 15000 |

Doctor Specialization:

|  |  |
| --- | --- |
| **Doctor ID** | **SPECIALIZATION** |
| 1 | Cardiology |
| 1 | Dermatology |
| 2 | Neurology |
| 3 | Paediatrics |
| 4 | Orthopaedics |

Doctor Details → Doctor ID, Name, Age, Contact, Salary.

Doctor Specialization → Doctor ID, Specialization.

Solution: Split the relation into two relations named Doctor Details and Doctor Specialization.

Doctor Details (Doctor ID (key), Name, Age, Contact, Salary).

Doctor Specialization (Doctor ID (key), Specialization).

1. **Appointment’s Table (Appointment ID, Patient ID, Doctor Id, Date, Time, Status)**

1NF: Meets the 1NF because it has no non-atomic attribute.

2NF: Meets the 2NF Rule-1 The primary key contains only one attribute.

3NF: This is not in 3N due to the removal of the transitive dependency

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **AppointmentID** | **Patient ID** | **Doctor ID** | **DATE** | **TIME** | **STATUS** |
| 1 | 1001 | 4 | 2024-04-15 | 10:15 | Completed |
| 2 | 1002 | 2 | 2024-04-16 | 16:30 | Scheduled |
| 3 | 1003 | 1 | 2024-04-17 | 11:00 | Scheduled |
| 4 | 1004 | 3 | 2024-04-18 | 14:45 | Cancelled |

**To convert the given table into 3NF, we decompose it into three tables:**

Appointment Table:

|  |  |  |  |
| --- | --- | --- | --- |
| **AppointmentID** | **Date** | **Time** | **Status** |
| 1 | 2024-04-15 | 10:15 | Completed |
| 2 | 2024-04-16 | 16:30 | scheduled |
| 3 | 2024-04-17 | 11:00 | scheduled |
| 4 | 2024-04-18 | 14:45 | canceled |

Patient Table:

|  |  |
| --- | --- |
| **Appointment ID** | **Patient ID** |
| 1 | 1001 |
| 2 | 1002 |
| 3 | 1003 |
| 4 | 1004 |

Doctor Table:

|  |  |
| --- | --- |
| **Appointment ID** | **Doctor ID** |
| 1 | 4 |
| 2 | 2 |
| 3 | 1 |
| 4 | 3 |

Appointment Table → Appointment ID, Date, Time, Status.

Patient Table → Appointment Id, Patient ID.

Doctor Table → Appointment Id, Doctor ID.

Solution: Split the relation into three relations named Appointment table, Patient Table and Doctor Table.

Appointment Table (Appointment ID (key), Date, Time, Status).

Patient Table (Appointment ID (key), Patient ID).

Doctor Table (Appointment ID (key), Doctor ID)

1. **Medical Records (RecordID, PatientID, DoctorID, Date, Diagnosis, Treatment)**

1NF: Meets the 1NF because it has no non-atomic attribute.

2NF: Meets the 2NF Rule-1 The primary key contains only one attribute.

3NF: This is not in 3N due to the existence of the transitive dependency

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **RecordID** | **PatientID** | **DoctorID** | **Date** | **Diagnosis** | **Treatment** |
| 1 | 1001 | 4 | 2024-04-15 | Sprained Ankle | Rest and Ice |
| 2 | 1002 | 2 | 2024-04-16 | Spinal Cord injury | Immobilisation |
| 3 | 1003 | 1 | 2024-04-17 | Chest Pain | Aspirin and evaluation |
| 4 | 1004 | 3 | 2024-04-18 | Fever | Paracetamol and rest |

**To convert the given table into 2NF, we decompose it into two tables:**

Appointments Table:

|  |  |  |  |
| --- | --- | --- | --- |
| **RecordID** | **PatientID** | **DoctorID** | **Date** |
| 1 | 1001 | 4 | 2024-04-15 |
| 2 | 1002 | 2 | 2024-04-16 |
| 3 | 1003 | 1 | 2024-04-17 |
| 4 | 1004 | 3 | 2024-04-18 |

Treatment Table:

|  |  |  |
| --- | --- | --- |
| **RecordID** | **Diagnosis** | **Treatment** |
| 1 | Sprained Ankle | Rest and Ice |
| 2 | Spinal Cord injury | Immobilisation |
| 3 | Chest Pain | Aspirin and evaluation |
| 4 | Fever | Paracetamol and rest |

Appointment Table → RecordID, PatientID, DoctorID, Date.

Treatment Table → RecordID, Diagnosis, Treatment.

Solution: Split the relation into two relations named Appointment Table and Treatment Table.

Appointment Table (RecordID (key), PatientID, DoctorID, Date).

Treatment Table (RecordID (key), Diagnosis, Treatment).

**5. Table Prescriptions**

1NF: Meets the 1NF because it has no non-atomic attribute.

2NF: Meets the 2NF Rule-1 The primary key contains only one attribute.

3NF: This is not in 3N due to the existence of the transitive dependency

Prescription Table:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **PrescriptionID** | **RecordID** | **Medication** | **Dosage** | **Frequency** | **StartDate** | **EndDate** |
| 1 | 1 | Ibuprofen | 400mg | Every6 hrs | 2024-04-15 | 2024-04-19 |
| 2 | 2 | Methylprednisolone | 40mg | Once daily | 2024-04-19 | 2024-04-21 |
| 3 | 3 | Nitroglycern | 0.4mg | As needed | 2024-04-20 | 2024-04-22 |
| 4 | 4 | Paracetamol,Aspirin | 500mg | Every 6hrs | 2024-04-18 | 2024-04-22 |

**The decomposition of the Patient’s table into 1NF is shown below:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **PrescriptionID** | **RecordID** | **Medication** | **Dosage** | **Frequency** | **StartDate** | **EndDate** |
| 1 | 1 | Ibuprofen | 400mg | Every 6 hrs | 2024-04-15 | 2024-04-19 |
| 2 | 2 | Methylprednisolone | 40mg | Once daily | 2024-04-19 | 2024-04-21 |
| 3 | 3 | Nitroglycerin | 0.4mg | As needed | 2024-04-20 | 2024-04-22 |
| 4 | 4 | Paracetamol | 500mg | Every 6hrs | 2024-04-18 | 2024-04-22 |
| 4 | 4 | Aspirin | 500mg | Every 6hrs | 2024-04-18 | 2024-04-22 |

Solution: Split the last column into 2 parts

**Creation of Data In The Tables [ENTITY RECORDS]**

Inserting the data into the tables:

**Patients Table:**

INSERT INTO Patients (PatientID**,** Name, Age, Gender, Contact, Problem) VALUES

('1001 ', 'Amulya', '28','Female', '9876543210', 'Sprained Ankle'),

('1002 ','Rajesh', '28', 'Male', '9876543211', 'Spinal cord injury'),

('1003 ','Ajay', '40', 'Male', '9876543212', 'Chest pain'),

('1004 ','Priya', '10', 'Female', '9876543213', 'Fever');

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **PatientID** | **Name** | **Age** | **Gender** | **Contact** | **Problem** |
| 1001 | Amulya | 35 | Female | 9876543210 | Sprained Ankle |
| 1002 | Rajesh | 28 | Male | 9876543211 | Spinal cord injury |
| 1003 | Ajay | 40 | Male | 9876543212 | Chest pain |
| 1004 | Priya | 10 | Female | 9876543213 | Fever |

**Doctors Table:**

INSERT INTO Doctors (DoctorID, Name, Age, Specialization, Salary, Contact) VALUES

('1', 'Dr. Vikram Malhotra', '32', 'Cardiology', '150000', '6305094874'),

('2', 'Dr. Rahul Kapoor', '58', 'Neurology', '150000', '6304585693'),

('3', 'Dr. Sneha Joshi', '41', 'Pediatrics', '10000', '8074126426'),

('4', 'Dr. Pooja Verma', '29', 'Orthopedics', '15000', '9876543203');

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **DoctorID** | **Name** | **Age** | **Specialization** | **Salary** | **Contact** |
| 1 | Dr. Vikram Malhotra | 32 | Cardiology | 150000 | 6305094874 |
| 2 | Dr. Rahul Kapoor | 58 | Neurology | 150000 | 6304585693 |
| 3 | Dr. Sneha Joshi | 41 | Pediatrics | 10000 | 8074126426 |
| 4 | Dr. Pooja Verma | 29 | Orthopedics | 15000 | 9876543203 |

**Appointments Table:**

INSERT INTO Appointments (AppointmentID, PatientID, DoctorID, Date, Time, Status) VALUES

('1', '1001', '4', '2024-04-15', '10:15', 'Completed'),

('2', '1002', '2', '2024-04-16', '16:30', 'Scheduled'),

('3', '1003', '1', '2024-04-17', '11:00', 'Scheduled'),

('4', '1004', '3', '2024-04-18', '14:45', 'Cancelled');

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **AppointmentID** | **PatientID** | **DoctorID** | **Date** | **Time** | **Status** |
| 1 | 1001 | 4 | 2024-04-15 | 10:15 | Completed |
| 2 | 1002 | 2 | 2024-04-16 | 16:30 | Scheduled |
| 3 | 1003 | 1 | 2024-04-17 | 11:00 | Scheduled |
| 4 | 1004 | 3 | 2024-04-18 | 14:45 | Cancelled |

**Medical Records Table:**

INSERT INTO MedicalRecords(RecordId, PatientID, DoctorID, Date, Diagnosis, Treatment) VALUES

('1', '1001', '4', '15-04-24', 'Sprained Ankle', 'Rest and Ice'),

('2', '1002', '2', '16-04-24', 'Spinal Cord injury', 'Immobilization'),

('3', '1003', '1', '17-04-24', 'Chest Pain', 'Aspirin and evaluation'),

('4', '1004', '3', '18-04-24', 'Fever', 'Paracetamol and rest');

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **RecordID** | **PatientID** | **DoctorID** | **Date** | **Diagnosis** | **Treatment** |
| 1 | 1001 | 4 | 2024-04-15 | Sprained Ankle | Rest and Ice |
| 2 | 1002 | 2 | 2024-04-16 | Spinal Cord injury | Immobilization |
| 3 | 1003 | 1 | 2024-04-17 | Chest Pain | Aspirin and evaluation |
| 4 | 1004 | 3 | 2024-04-18 | Fever | Paracetamol and rest |

**Prescriptions Table:**

INSERT INTO Prescriptions (PrescriptionId, RecordID, Medication, Dosage, Frequency, StartDate, EndDate) VALUES

('1', '1', 'Ibuprofen', '400 mg', 'Every 6 hrs', '15-04-24', '19-04-24'),

('2', '2', 'Methylprednisolone', '40 mg', 'Once daily', '19-04-24', '21-04-24'),

('3', '3', 'Nitroglycerin', '0.4 mg', 'As needed', '20-04-24', '22-04-24'),

('4', '4', 'Paracetamol', '500 mg', 'Every 6 hrs', '18-04-24', '22-04-24');

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **PrescriptionID** | **RecordID** | **Medication** | **Dosage** | **Frequency** | **StartDate** | **EndDate** |
| 1 | 1 | Ibuprofen | 400mg | Every 6 hrs | 2024-04-15 | 2024-04-19 |
| 2 | 2 | Methylpredni-solone | 40mg | Once daily | 2024-04-19 | 2024-04-21 |
| 3 | 3 | Nitroglycerin | 0.4mg | As needed | 2024-04-20 | 2024-04-22 |
| 4 | 4 | Paracetamol | 500mg | Every 6 hrs | 2024-04-18 | 2024-04-22 |

**Few sql queries on the created tables**

➳*Data Retrieval Queries:*

1. **Find all records in “Medical Records”**

**SQL Command:**

SELECT \* FROM Medical Records;

1. **Write a query to find Medication, Dosage and Frequency from “Prescription” table**

**SQL Command:**

SELECT \* Medication, Dosage, Frequency

FROM Prescriptions;

1. **Write a query to retrieve all “Appointments” where DoctorId is ‘4’**

**SQL Command:**

SELECT\*

FROM Appointments

WHERE DoctorId = 4;

➳*Data Manipulation Queries:*

1. **Find query that inserts multiple records into the "Patients" table**

**SQL Command:**

INSERT INTO Patients (PatientId, Name, Age, Gender, Contact, Problem) VALUES

(5, 'Raju', 30, 'M', '9876543342', 'Leukemia'),

(6, 'Latha', 53, 'F', '8769654341', 'Fracture');

1. **Write a query to update the salary of a doctor with DoctorID ‘1’ in the "Doctors" table**

**SQL Command:**

UPDATE Doctors

SET Salary = 180000

WHERE DoctorID = 1;

1. **Write a query to delete records from "Appointments" table where PatientID is '1002':**

**SQL Command:**

DELETE FROM Appointments

WHERE PatientID = '1002';

1. **Write a query to add a new column “Address” to the “Patients” table with data type of ‘VARCHAR (255)’**

**SQL Command:**

ALTER TABLE Patients

ADD Address VARCHAR (255);

1. **Write a query to modify the age of patient with PatientId ‘1001’**

**SQL Command:**

UPDATE Patients

SET Age = 35

WHERE PatientID = 1001;

➳ *Aggregate Functions:*

1. **Calculate the ‘average age’ of patients in the “Patients” table**

**SQL Command:**

SELECT AVG(Age) AS AverageAge

FROM Patients;

1. **Calculate the ‘total salary’ of all doctors in the “Doctor” table**

**SQL Command:**

SELECT SUM(Salary) AS TotalSalary

FROM Doctors;

**Creation of 5 views using the tables**

**1)** **Completed Patient Appointments View**

CREATE VIEW CompletedPatientAppointmentsView AS

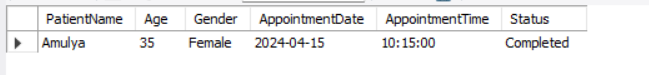
SELECT p.Name AS PatientName, p.Age, p.Gender, a.Date AS AppointmentDate, a.Time AS AppointmentTime, a.Status

FROM Patients p

INNER JOIN Appointments a ON p.PatientID = a.PatientID

WHERE a.Status = 'Completed';

**QUERY:** SELECT \* FROM CompletedPatientAppointmentsView;



**2) Patient Medical Records For Diagnosis View**

CREATE VIEW PatientMedicalRecordsForDiagnosisView AS

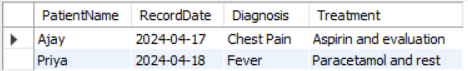
SELECT p.Name AS PatientName, m.Date AS RecordDate, m.Diagnosis, m.Treatment

FROM Patients p

INNER JOIN MedicalRecords m ON p.PatientID = m.PatientID

WHERE m.Diagnosis IN ('Fever', 'Chest Pain');

**QUERY:** SELECT\* FROM PatientMedicalRecordsForDiagnosisView;



**3)Patient Medical Records View**

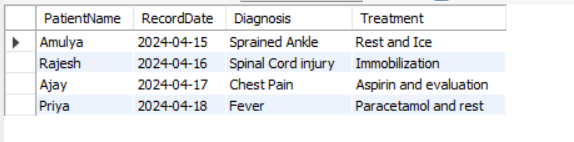
CREATE VIEW PatientMedicalRecordsView AS

SELECT p.Name AS PatientName, m.Date AS RecordDate, m.Diagnosis, m.Treatment

FROM Patients p

INNER JOIN MedicalRecords m ON p.PatientID = m.PatientID;

**QUERY:** SELECT \* FROM PatientMedicalRecordsView ;



**4)Doctor Patients View**

CREATE VIEW DoctorPatientsView AS

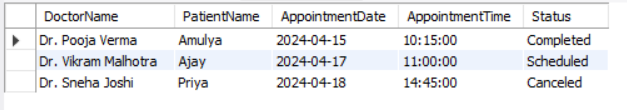
SELECT d.Name AS DoctorName, p.Name AS PatientName, a.Date AS AppointmentDate, a.Time AS AppointmentTime, a.Status

FROM Doctors d

INNER JOIN Appointments a ON d.DoctorID = a.DoctorID

INNER JOIN Patients p ON a.PatientID = p.PatientID;

**QUERY:** SELECT \* FROM DoctorPatientsView;



**5)Prescription Details View**

CREATE VIEW PrescriptionDetailsView AS

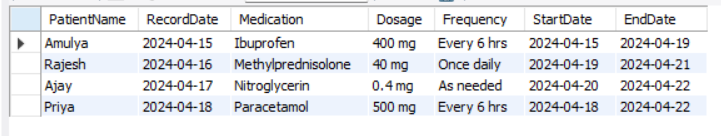
SELECT p.Name AS PatientName, m.Date AS RecordDate, pr.Medication, pr.Dosage, pr.Frequency, pr.StartDate, pr.EndDate

FROM Patients p

INNER JOIN MedicalRecords m ON p.PatientID = m.PatientID

INNER JOIN Prescriptions pr ON m.RecordID = pr.RecordID;

**QUERY:** SELECT \* FROM PrescriptionDetailsView;



**THANK YOU**

**SUBMITTED BY:**

|  |  |
| --- | --- |
| **NAME** | **REGISTRATION NUMBER** |
| Kanyadhara Bandhavi | AP22110010079 |
| Kakarlapudi Hema | AP22110010080 |
| Nagalla Swathi Chowdary | AP22110010084 |
| Kolasani Vaishnavi | AP22110010126 |