Hospital Management System

Course: Database Management System

Document Name: Phase 2 Submission

Prepared by: Meet Saraswat

Contents

1. Executive Summary

While there have been successful and comprehensive healthcare systems to improve patient outcomes, a better HMS(Hospital Management System) can always enhance efficiency. This project is designed to gather examples of past attempts, develop a high-level representation of the extent of work, and provide an in-depth analysis of data utilized in the hospital management system. This representation will include various data models which take into account representations of the data needed for appointment scheduling, hospital operations, and tracking finances. In this report, we start with the logic design and modeling of our dataset. First, we designed our ER/EER diagram, along with all underlying assumptions, which gives a visual presentation of the entities and relationships between them. In the next section, we focus on the relational schema, where the logical diagram is transformed into a database diagram incorporating more detailed information. And then we normalize the tables to ensure all of them conform to 3NF. Lastly, we conclude the design with a short summary.

2. Problem Description

The conventional Hospital Management System has been facing challenges, information is difficult to retrieve, for instance, patients' Identification numbers are unique throughout the system for each patient, and traditionally, this process is done manually. Therefore, errors occur in transaction processes, it takes time and effort to handle and secure patient information, and diagnosis data. This data model is made to improve the hospital management system, to register and maintain patient information for staff to access and update the information when needed. Simultaneous updates and changes are made and stored to the databases by administrators or receptionists.

3. Conceptual Design

Here is the EER diagram generated based on our project description and real-life experiences.

3.1. EER diagram with all assumptions

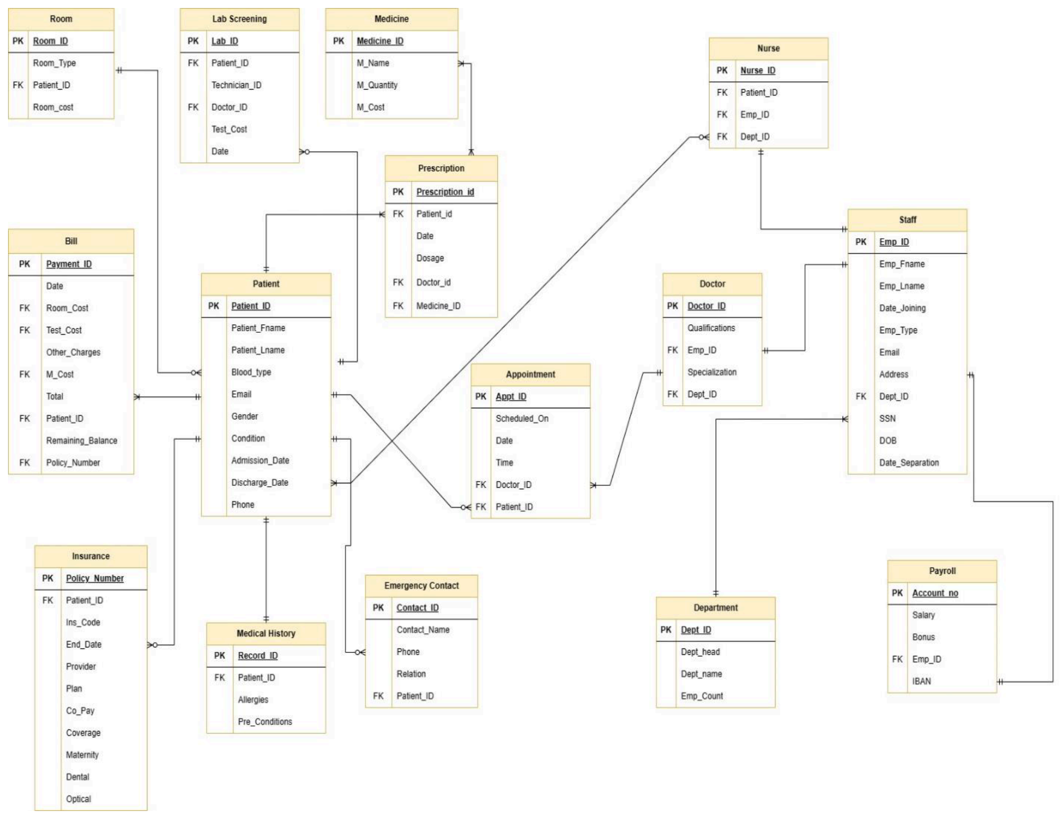
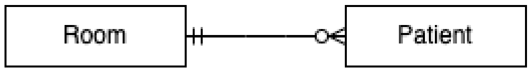
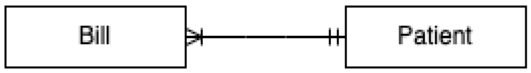
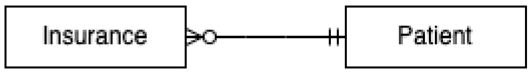
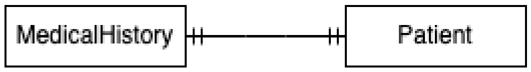
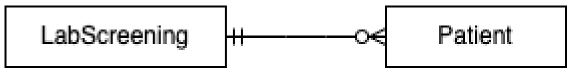
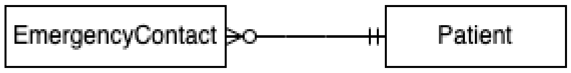
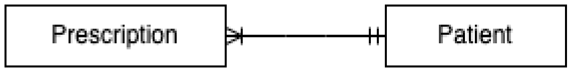
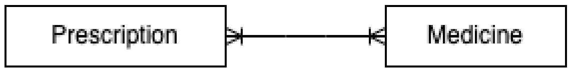
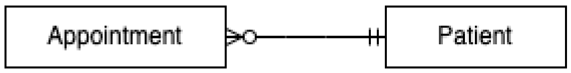
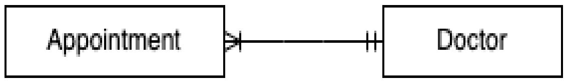
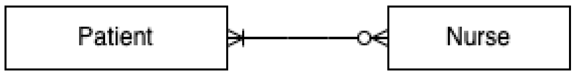
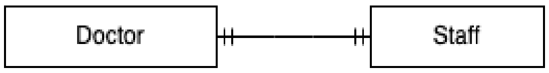
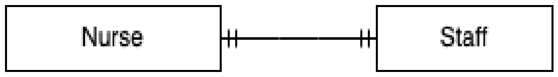
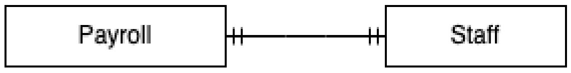


Figure 1. EER Design for Hospital Management System Database

3.2. Crow Foot Notation for Relationship

This section examines the min-max notations used in the EER diagram to portray the relationships between entities. Crow Foot notations are elaborated upon in Table 1 to facilitate comprehension and interpretation of the relationships that exist in the relational database.

Expression	Description
	A patient must be assigned to one and only one room. A room may have zero or many patients.
	A patient may have one to many bills. A bill must be charged to one and only one patient.
	A patient may have zero to many insurance policies. An insurance policy must be associated with one and only patient.
	A patient needs to have a medical history record with the hospital. A medical history record must be related to a patient.

	A patient may have zero to many lab screening procedures. A record of a lab screening procedure is associated with one and only one patient.
	A patient may have zero to many emergency contacts. An emergency contact should be linked to one and only one patient.
	A patient may have one to many prescriptions. A prescription must be associated with one and only one patient.
	A prescription may contain one to many medicines. A medicine may be prescribed in one or many prescriptions.
	A patient may have zero to many appointments. An appointment must be linked to one and only one patient.
	A doctor may have one to many appointments. An appointment must be scheduled with one and only one doctor.
	A nurse may be assigned to one or many patients. A patient may have zero to many nurses assigned to them.
	A doctor must have one and only one record under staff. A staff record can only be associated with one and only one doctor.
	A nurse must have one and only one record under staff. A staff record can only be associated with one and only one nurse.
	A payroll must have one and only one record under staff. A staff record can only be associated with one and only one payroll record.

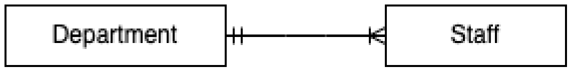
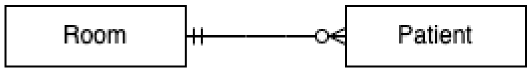
	A staff must work for one and only one department. A department may have one to many staff members.
	A patient must be assigned to one and only one room. A room may have zero or many patients.

Table 1. Explanation for Crow Foot Notation

4. Relational Schema

4.1. Relational Schema

The diagram highlights the relationships between each table by showing how each foreign key is connected to the primary key of the parent table. It is essentially a blueprint to show how information is correlated and retrieved in the database. For example, putting the Emp_ID in the Nurse and Doctor table allows their respective information to be retrieved when Emp_ID is called in a query.

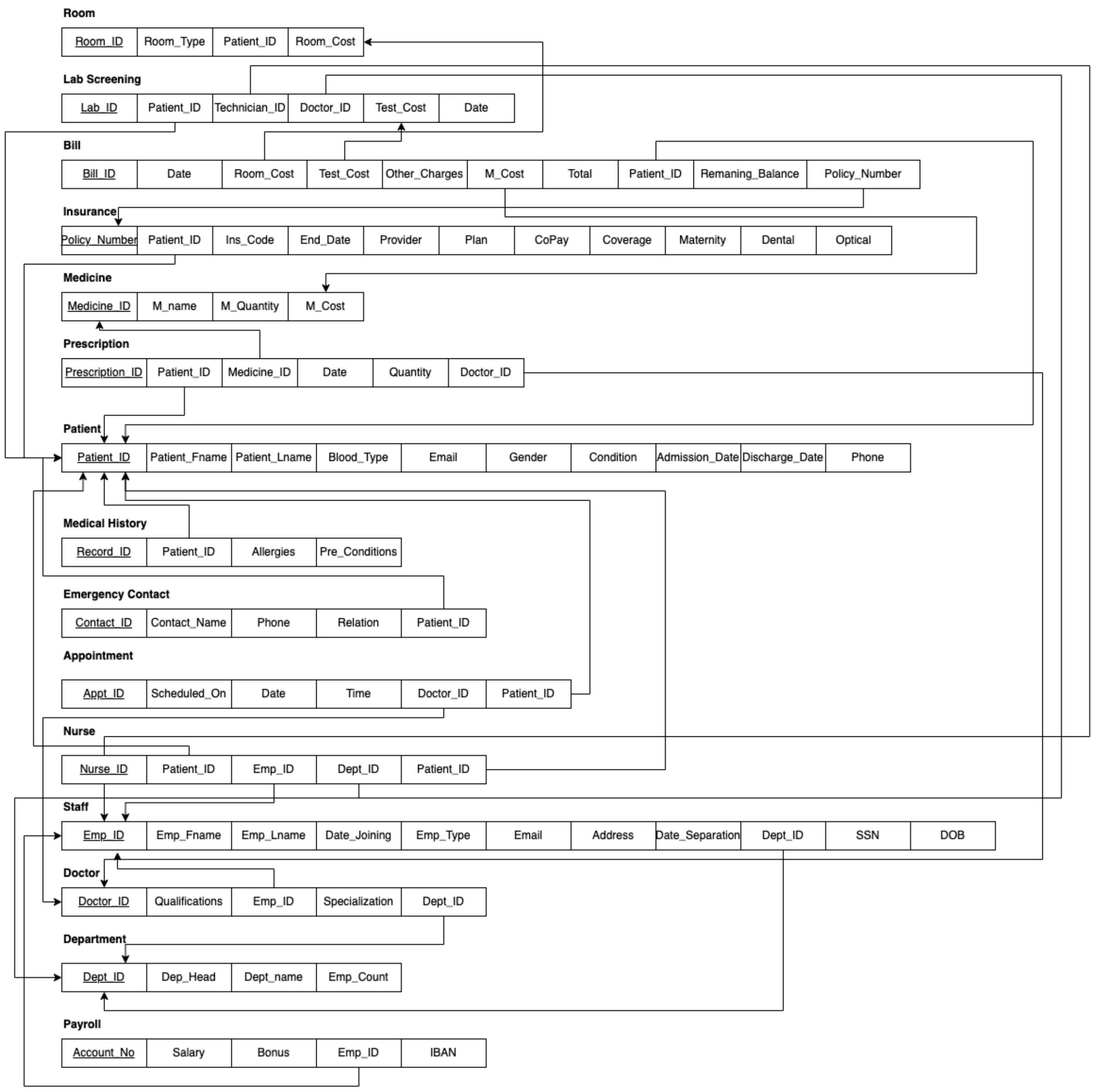

Room
Room_ID | Room_Type | Patient_ID | Room_Cost
Lab Screening
Lab_ID | Patient_ID | Technician_ID | Doctor_ID | Test_Cost | Date
Bill
Bill_ID | Date | Room_Cost | Test_Cost | Other_Charges | M_Cost | Total | Patient_ID | Remaning_Balance | Policy_Number
Insurance
Policy_Number | Patient_ID | Ins_Code | End_Date | Provider | Plan | CoPay | Coverage | Maternity | Dental | Optical
Medicine
Medicine_ID | M_name | M_Quantity | M_Cost
Prescription
Prescription_ID | Patient_ID | Medicine_ID | Date | Quantity | Doctor_ID
Patient
Patient_ID | Patient_Fname | Patient_Lname | Blood_Type | Email | Gender | Condition | Admission_Date | Discharge_Date | Phone
Medical History
Record_ID | Patient_ID | Allergies | Pre_Conditions
Emergency Contact
Contact_ID | Contact_Name | Phone | Relation | Patient_ID
Appointment
Appt_ID | Scheduled_On | Date | Time | Doctor_ID | Patient_ID
Nurse
Nurse_ID | Patient_ID | Emp_ID | Dept_ID | Patient_ID
Staff
Emp_ID | Emp_Fname | Emp_Lname | Date_Joining | Emp_Type | Email | Address | Date_Separation | Dept_ID | SSN | DOB
Doctor
Doctor_ID | Qualifications | Emp_ID | Specialization | Dept_ID
Department
Dept_ID | Dep_Head | Dept_name | Emp_Count
Payroll
Account_No | Salary | Bonus | Emp_ID | IBAN

Figure 2. Relational Schema for Hospital Management System Database

4.2. Data Format for Every Relation

Relation Names	Attributes	Data Type
Room	Room_ID	INT
	Room_Type	VARCHAR(50)
	Patient_ID	INT
	Room Cost	Decimal(10,2)

```
CREATE TABLE  Room (

 Room_ID INT NOT NULL,

 Room_Type VARCHAR(50) NOT NULL,

 Patient_ID  INT  NOT NULL,     _

 Room_Cost  DECIMAL(10,2),

PRIMARY KEY (Room_ID),

FOREIGN KEY (Patient_ID) REFERENCES Patient (Patient_ID)

 );
```

Relation Names	Attributes	Data Type
Lab Screening	Lab_ID	INT
	Patient_ID	INT
	Technician_ID	INT
	Doctor_ID	INT
	Test_Cost	Decimal(10,2)
	Date	DATE

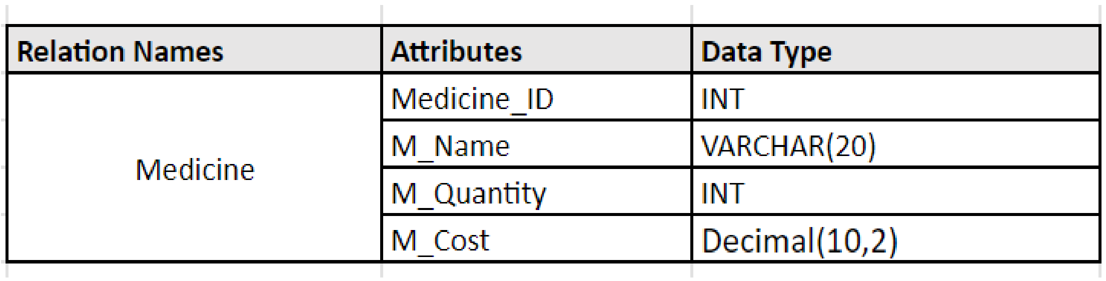
```
 CREATE TABLE  Lab_Screening (

Lab_ID INT NOT NULL,

Patient_ID  INT  NOT NULL,

Technician_ID  INT  NOT NULL,
    Doctor_ID  INT NOT NULL,

     Test_Cost  DECIMAL(10,2),

     Date  DATE  NOT NULL,

     PRIMARY KEY (Lab_ID),

    FOREIGN KEY (Patient_ID) REFERENCES Patient (Patient_ID),

    FOREIGN KEY (Doctor_ID) REFERENCES Doctor (Doctor_ID)

);
```

Relation Names	Attributes	Data Type
Bill	Bill_ID	INT
	Date	DATE
	Room_Cost	Decimal(10,2)
	Test_Cost	Decimal(10,2)
	Other_Charges	Decimal(10,2)
	M_Cost	Decimal(10,2)
	Total	Decimal(10,2)
	Patient_ID	INT
	Remaining_Balance	Decimal(10,2)
	Policy_Number	VARCHAR(20)

```
CREATE TABLE Bill (

    Bill_ID INT NOT NULL,

    Date  DATE,

   Room_Cost Decimal(10,2),

    Test_Cost  DECIMAL(10,2),

    Other_Charges  DECIMAL(10,2),

    M_Cost DECIMAL(10,2),

    Total  DECIMAL(10,2),
    Patient_ID INT NOT NULL,

    Remaining_Balance DECIMAL(10,2),

    Policy_Number VARCHAR(20) NOT NULL

    PRIMARY KEY (Payment_ID),

    FOREIGN KEY (Room_Cost) REFERENCES Room  (Room_Cost),

    FOREIGN KEY (Test_Cost) REFERENCES Lab_Screening (Test_Cost),

    FOREIGN KEY (M_Cost) REFERENCES Medicine (M_Cost),

    FOREIGN KEY (Patient_ID) REFERENCES Patient (Patient_ID),

    FOREIGN KEY (Policy_Number) REFERENCES Insurance (Policy_Number)

);
```

Relation Names	Attributes	Data Type
Insurance	Policy_Number	VARCHAR(20)
	Patient_ID	INT
	Ins_Code	VARCHAR(20)
	End_Date	VARCHAR(10)
	Provider	VARCHAR(20)
	Plan	VARCHAR(20)
	Co_Pay	Decimal(10,2)
	Coverage	VARCHAR(20)
	Maternity	BOOLEAN
	Dental	BOOLEAN
	Optical	BOOLEAN

```
CREATE TABLE Insurance (

    Policy_Number VARCHAR(20) NOT NULL,

    Patient_ID INT NOT NULL,

    Ins_Code VARCHAR(20) NOT NULL,

    End_Date VARCHAR(10),
    Provider VARCHAR(20),

    Plan VARCHAR(20),

    Co_Pay  DECIMAL(10,2),

    Coverage VARCHAR(20),

    Maternity  BOOLEAN,

   Dental  BOOLEAN,

   Optical BOOLEAN,

 PRIMARY  KEY (Policy_Number),

 FOREIGN KEY (Patient_ID) REFERENCES Patient (Patient_ID)

);
```



Relation Names	Attributes	Data Type
Medicine	Medicine_ID	INT
	M_Name	VARCHAR(20)
	M_Quantity	INT
	M_Cost	Decimal(10,2)

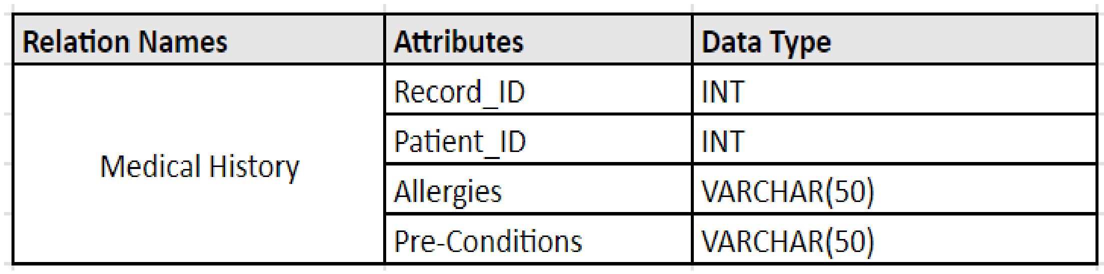
```
CREATE TABLE Medicine (

    Medicine_ID INT  NOT NULL,

    M_Name VARCHAR(20) NOT NULL,

   M_Quantity INT NOT NULL,

    M_Cost  Decimal(10,2),

  PRIMARY KEY (Medicine_ID)

   );
```

Relation Names	Attributes	Data Type
Prescription	Prescription_ID	INT
	Patient_ID	INT
	Medicine_ID	INT
	Date	DATE
	Dosage	INT
	Doctor_ID	INT

```
CREATE TABLE Prescription (

   Prescription_ID INT  NOT NULL,

   Patient_ID  INT  NOT NULL,

    Medicine_ID  INT  NOT NULL,

   Date  DATE,

   Dosage  INT,

   Doctor_ID INT NOT NULL,

   PRIMARY KEY (Prescription_ID),

   FOREIGN KEY (Patient_ID) REFERENCES Patient (Patient_ID),

   FOREIGN KEY (Doctor_ID) REFERENCES Doctor (Doctor_ID),

  FOREIGN KEY (Medicine_ID) REFERENCES Medicine (Medicine_ID)

);
```

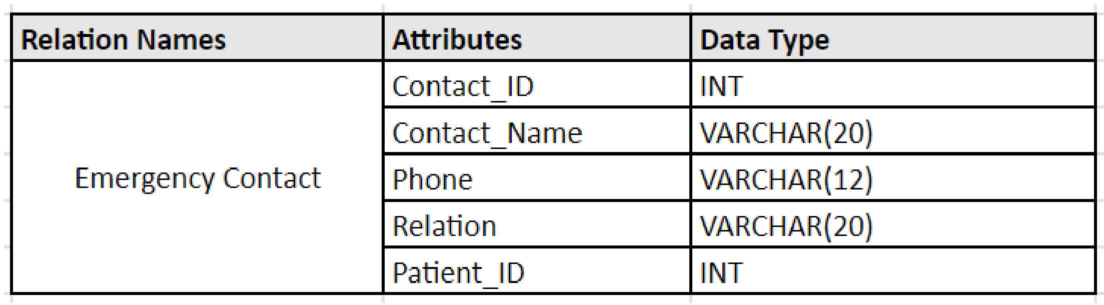
Relation Names	Attributes	Data Type
Patient	Patient_ID	INT
	Patient_FName	VARCHAR(20)
	Patient_LName	VARCHAR(20)
	Phone	VARCHAR(12)
	Blood_Type	VARCHAR(5)
	Email	VARCHAR(50)
	Gender	VARCHAR(10)
	Condition	VARCHAR(30)
	Admission_Date	DATE
	Discharge_Date	DATE

```
CREATE TABLE Patient (

    Patient_ID INT NOT NULL,

   Patient_FName VARCHAR(20) NOT NULL,

    Patient_LName VARCHAR(20) NOT NULL,

    Phone VARCHAR(12) NOT NULL,

    Blood_Type  VARCHAR(5) NOT NULL,

    Email VARCHAR(50),

    Gender  VARCHAR(10),

    Condition VARCHAR(30),

    Admission_Date DATE,

    Discharge_Date DATE,

   PRIMARY KEY (Patient_ID)

);
```



Relation Names	Attributes	Data Type
Medical History	Record_ID	INT
	Patient_ID	INT
	Allergies	VARCHAR(50)
	Pre-Conditions	VARCHAR(50)

```
CREATE TABLE Medical_History (
    Record_ID  INT NOT NULL,
   Patient_ID  INT NOT NULL,
   Allergies VARCHAR(50),
   Pre_Conditions VARCHAR(50),
  PRIMARY KEY (Record_ID),
 FOREIGN KEY (Patient_ID) REFERENCES Patient (Patient_ID)
);
```



Relation Names	Attributes	Data Type
Emergency Contact	Contact_ID	INT
	Contact_Name	VARCHAR(20)
	Phone	VARCHAR(12)
	Relation	VARCHAR(20)
	Patient_ID	INT

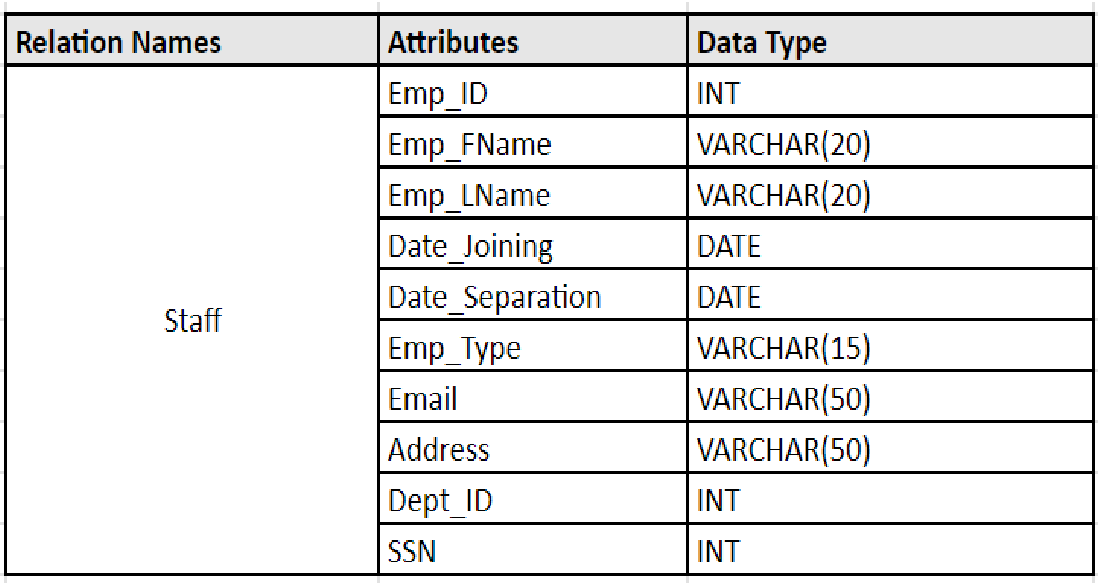
```
CREATE TABLE Emergency_Contact(
    Contact_ID INT  NOT NULL,
   Contact_Name VARCHAR(20) NOT NULL,
    Phone VARCHAR(12) NOT NULL,
   Relation VARCHAR(20) NOT NULL,
  Patient_ID  INT NOT NULL,
 PRIMARY KEY (Contact_ID),
 FOREIGN KEY (Patient_ID) REFERENCES Patient (Patient_ID)
);
```

Relation Names	Attributes	Data Type
Appointment	Appt_ID	INT
	Scheduled_On	DATETIME
	Date	DATE
	Time	TIME
	Doctor_ID	INT
	Patient_ID	INT

```
CREATE TABLE Appointment (
   Appt_ID INT  NOT NULL,
   Scheduled_On  DATETIME NOT NULL,
   Date  DATE,
   Time TIME,
   Doctor_ID INT NOT NULL,
   Patient_ID  INT NOT NULL,
  PRIMARY KEY (Appt_ID),
 FOREIGN KEY (Doctor_ID) REFERENCES Doctor (Doctor_ID),
  FOREIGN KEY (Patient_ID) REFERENCES Patient (Patient_ID)
   );
```

Relation Names	Attributes	Data Type
Nurse	Nurse_ID	INT
	Patient_ID	INT
	Emp_ID	INT
	Dept_ID	INT

```
CREATE TABLE Nurse (

    Nurse_ID INT  NOT NULL,

    Patient_ID  INT  NOT NULL,

    Emp_ID  INT NOT NULL,

    Dept_ID INT NOT NULL,

   PRIMARY KEY(Nurse_ID),

  FOREIGN KEY (Patient_ID) REFERENCES Patient (Patient_ID),

  FOREIGN KEY (Emp_ID) REFERENCES Staff  (Emp_ID),

FOREIGN KEY (Dept_ID) REFERENCES Department (Dept_ID)

);
```



Relation Names	Attributes	Data Type
Staff	Emp_ID	INT
	Emp_FName	VARCHAR(20)
	Emp_LName	VARCHAR(20)
	Date_Joining	DATE
	Date_Separation	DATE
	Emp_Type	VARCHAR(15)
	Email	VARCHAR(50)
	Address	VARCHAR(50)
	Dept_ID	INT
	SSN	INT

```
CREATE TABLE Staff (

   Emp_ID INT  NOT NULL,

   Emp_FName  VARCHAR(20) NOT NULL,

   Emp_LName  VARCHAR(20) NOT NULL,

   Date_Joining  DATE,

   Date_Seperation DATE,

   Emp_Type VARCHAR(15) NOT NULL,

   Email  VARCHAR(50),

   Address  VARCHAR(50) NOT NULL,

   Dept_ID  INT NOT NULL,

   SSN  INT NOT NULL,

PRIMARY KEY (Emp_ID),

FOREIGN KEY (Dept_ID) REFERENCES Department  (Dept_ID)

);
```

Relation Names	Attributes	Data Type
Doctor	Doctor_ID	INT
	Qualifications	VARCHAR(15)
	Emp_ID	INT
	Specialization	VARCHAR(20)
	Dept_ID	INT

```
CREATE TABLE Doctor (

    Doctor_ID INT NOT NULL,

    Qualifications VARCHAR(15) NOT NULL,

   Emp_ID INT NOT NULL,

  Specialization VARCHAR(20) NOT NULL,

  Dept_ID INT NOT NULL,

 PRIMARY KEY (Doctor_ID),

 FOREIGN KEY (Emp_ID) REFERENCES Staff (Emp_ID),

FOREIGN KEY (Dept_ID) REFERENCES Department (Dept_ID)

);
```

Relation Names	Attributes	Data Type
Department	Dept_ID	INT
	Dept_Head	VARCHAR(20)
	Dept_Name	VARCHAR(15)
	Emp_Count	INT

```
CREATE TABLE Department (
  Dept_ID INT NOT NULL,

  Dept_Head VARCHAR(20) NOT NULL,

  Dept_Name VARCHAR(15) NOT NULL,

 Emp_Count INT,

 PRIMARY  KEY (Dept_ID)

);
```

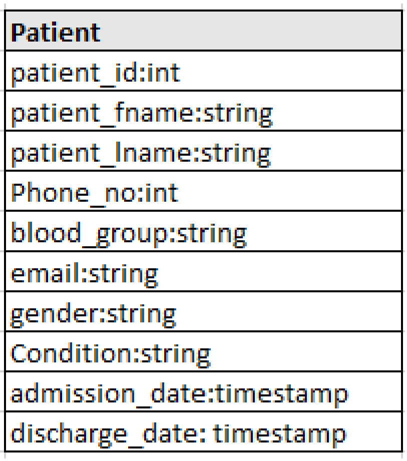
Relation Names	Attributes	Data Type
Payroll	Account.No	VARCHAR(25)
	Salary	Decimal(10,2)
	Bonus	Decimal(10,2)
	Emp_ID	INT
	IBAN	VARCHAR(25)

```
CREATE TABLE Payroll (

  Account_No VARCHAR(25) NOT NULL,

  Salary DECIMAL(10,2) NOT NULL,

  Bonus DECIMAL(10,2),

  Emp_ID INT NOT NULL,

  IBAN VARCHAR(25),

  PRIMARY KEY (Account_No),

  FOREIGN KEY (Emp_ID) REFERENCES Staff (Emp_ID)

 );
```

5. Normalization

In this part, we apply the principles of normalization to ensure all the tables conform to 3NF.

I. Patient Table



Patient
patient_id:int
patient_fname:string
patient_lname:string
Phone_no:int
blood_group:string
email:string
gender:string
Condition:string
admission_date:timestamp
discharge_date: timestamp

1NF Compliance: The table is in the First Normal Form (1NF) since it contains only atomic (indivisible) values in each column, and there are no repeating groups.

2NF Compliance: It is also in the Second Normal Form (2NF) because it has a primary key, "patient_id," which uniquely identifies each row, and all non-key attributes are fully functionally dependent on the primary key.

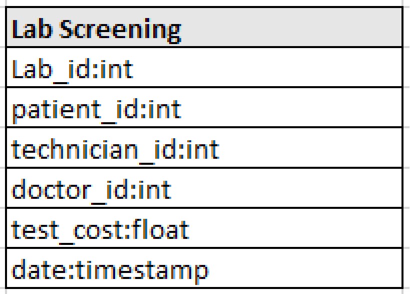
3NF Compliance: The table is in the Third Normal Form (3NF) because it has no transitive dependencies. All non-key attributes are directly dependent on the primary key "patient_id."

Functional Dependencies:

patient_id → patient_fname, patient_lname, Phone_no, blood_group, email, gender, condition, admission_date, discharge_date

This set of functional dependencies implies that each patient's name, phone number, blood group, email, gender, condition, admission date, and discharge date are directly determined by their unique patient ID. As a result, the "Patient" table is in 3NF.

II. Lab Screening Table



Lab Screening
Lab_id:int
patient_id:int
technician_id:int
doctor_id:int
test_cost:float
date:timestamp

1NF Compliance: The table is in the First Normal Form (1NF) as it contains atomic values in each column, and there are no repeating groups.

2NF Compliance: It is in the Second Normal Form (2NF) because it has a composite primary key consisting of "Lab_id" and "patient_id," which uniquely identifies each row, and all non-key attributes are fully functionally dependent on this composite key.

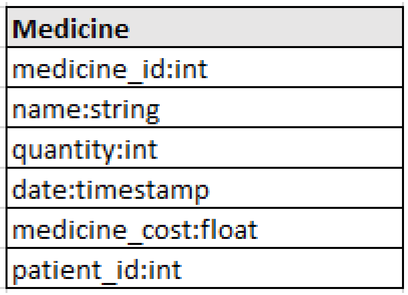
3NF Compliance: The table is also in the Third Normal Form (3NF) because it has no transitive dependencies. All non-key attributes are directly dependent on the composite primary key, "Lab_id" and "patient_id."

Functional Dependencies:

(Lab_id, patient_id) → technician_id, doctor_id, test_cost, date

This set of functional dependencies indicates that for each specific laboratory screening instance, which is identified by the combination of "Lab_id" and "patient_id," the attributes "technician_id," "doctor_id," "test_cost," and "date" are directly determined. This structure promotes data integrity and efficiency in managing lab screening records. As a result, the table is in 3NF.

III. Medicine Table



Medicine
medicine_id:int
name:string
quantity:int
date:timestamp
medicine_cost:float
patient_id:int

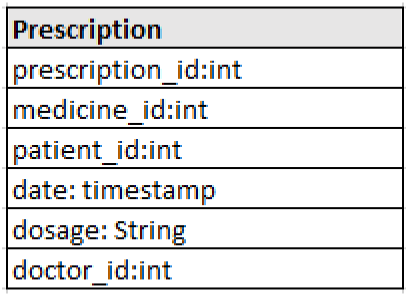
1NF Compliance: All attributes must be atomic, which means that they cannot be further divided. In this table, the attributes appear to be atomic, so it satisfies 1NF.

2NF Compliance: For a table to be in 2NF, it must first be in 1NF. Then, it must have no partial dependencies. A partial dependency occurs when an attribute depends on only a part of the candidate key.

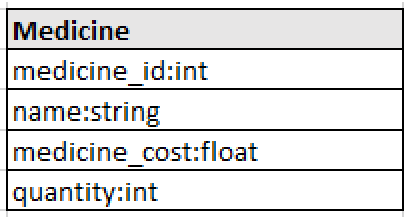
Assuming that medicine_id is the candidate key, the Patient_id attribute appears to depend on the candidate key (medicine_id), indicating that it associates a patient with a specific medicine. This might be a partial dependency issue.

The presence of Patient_id in the "Medicine" table suggests that it might have a functional dependency on the patient for whom the medicine is prescribed. This indicates a potential partial dependency, which would mean that the table is not in 2NF.

To bring the table to 2NF and resolve this issue, we created a separate table for the prescription or association between patients and medicines.



Prescription
prescription_id:int
medicine_id:int
patient_id:int
date: timestamp
dosage: String
doctor_id:int



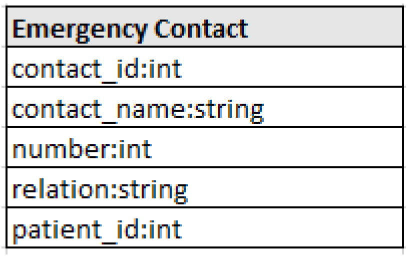
Medicine
medicine_id:int
name:string
medicine_cost:float
quantity:int

Functional Dependencies:

prescription_id -> medicine_id, patient_id, date, dosage, doctor_id
medicine_id -> name, medicine_cost, quantity
medicine_id -> name, medicine_cost, quantity

3NF Compliance: Both tables are in the Third Normal Form (3NF) because they meet the requirements of 1NF (atomic attributes), 2NF (no partial dependencies), and 3NF (no transitive dependencies). In these tables, attributes depend directly on the candidate keys, and there are no indirect dependencies.

IV. Emergency Contact Table



Emergency Contact
contact_id:int
contact_name:string
number:int
relation:string
patient_id:int

1NF Compliance: The table is in the First Normal Form (1NF) as it contains atomic values in each column, and there are no repeating groups.

2NF Compliance: It is also in the Second Normal Form (2NF) because it has a primary key, "contact_id," which uniquely identifies each row, and all non-key attributes are fully functionally dependent on the primary key.

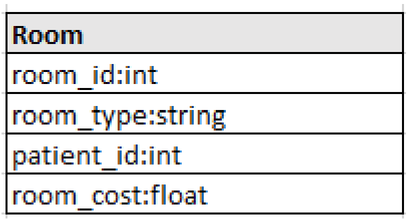
3NF Compliance: The table satisfies the Third Normal Form (3NF) because it has no transitive dependencies. All non-key attributes are directly dependent on the primary key, "contact_id," and "patient_id."

Functional Dependencies:

contact_id → contact_name, number, relation, patient_id

This set of functional dependencies indicates that for each unique emergency contact, identified by "contact_id," the attributes "contact_name," "number," "relation," and "patient_id" are directly determined.

V. Room Table



Room
room_id:int
room_type:string
patient_id:int
room_cost:float

1NF Compliance: The table is in the First Normal Form (1NF) because it contains atomic values in each column, and there are no repeating groups.

2NF Compliance: It is also in the Second Normal Form (2NF) since it has a primary key, "room_id," which uniquely identifies each row, and all non-key attributes are fully functionally dependent on the primary key.

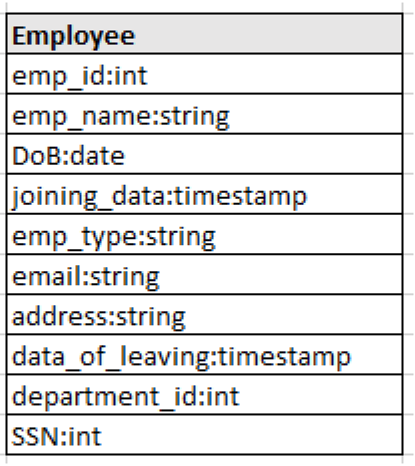
3NF Compliance: The table satisfies the Third Normal Form (3NF) because it does not contain any transitive dependencies. All non-key attributes are directly dependent on the primary key, "room_id."

Functional Dependencies:

room_id → room_type, patient_id, room_cost

This set of functional dependencies indicates that for each individual room, identified by "room_id," the attributes "room_type," "patient_id," and "room_cost" are directly determined. There are no transitive dependencies, ensuring that the "Room" table is well-structured and adheres to 3NF principles.

VI. Employee Table



Employee
emp_id:int
emp_name:string
DoB:date
joining_data:timestamp
emp_type:string
email:string
address:string
data_of_leaving:timestamp
department_id:int
SSN:int

1NF Compliance: The table is in the First Normal Form (1NF) because it contains atomic values in each column, and there are no repeating groups.

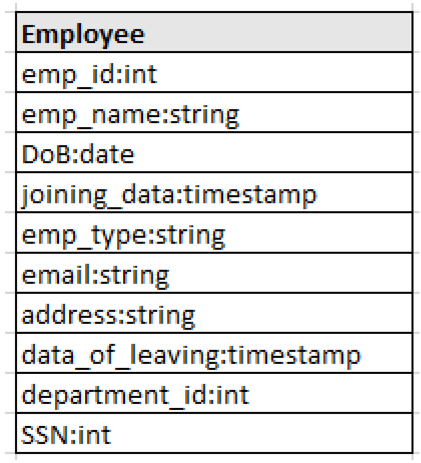
2NF Compliance: It is in the Second Normal Form (2NF) since it has a primary key, "emp_id," which uniquely identifies each row, and all non-key attributes are fully functionally dependent on the primary key.

3NF Compliance: The "Employee" table as provided does not fully satisfy the Third Normal Form (3NF) due to a transitive dependency between the "department_id" and "department_name" attributes. The "department_name" depends on "department_id," which itself depends on the "emp_id," violating the 3NF principles.

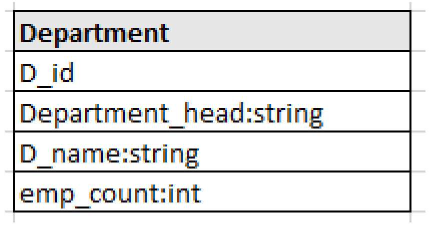
Functional Dependencies:

emp_id → emp_name, DoB, joining_date, emp_type, email, address, data_of_leaving, department_id, SSN
department_id → department_name

To bring the table into 3NF, we separated the "department_id" and "department_name" into a separate table that links department details to employees



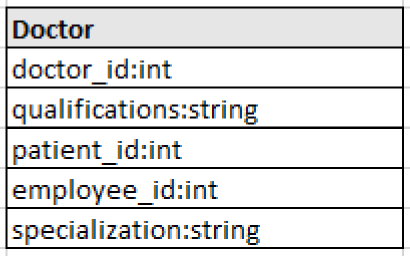
Employee
emp_id:int
emp_name:string
DoB:date
joining_data:timestamp
emp_type:string
email:string
address:string
data_of_leaving:timestamp
department_id:int
SSN:int



Department
D_id
Department_head:string
D_name:string
emp_count:int

With this there are no transitive or partial dependencies in any of the 2 table, So they conform to the 3NF normal form.

VII. Doctor Table



Doctor
doctor_id:int
qualifications:string
patient_id:int
employee_id:int
specialization:string

1NF Compliance: Each column in the table contains atomic (indivisible) values, meeting the requirements of 1NF.

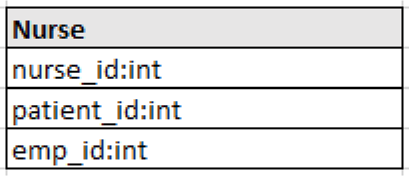
2NF Compliance: The table goes beyond 1NF by having a primary key, "doctor_id," which uniquely identifies each row. All non-key attributes are fully functionally dependent on the primary key, adhering to 2NF principles.

Functional Dependencies:

doctor_id → qualifications, patient_id, specialization, employee_id

3NF Compliance: The table satisfies the Third Normal Form (3NF) because it does not contain any transitive dependencies. All non-key attributes are directly dependent on the primary key, "Doctor_id."

VIII. Nurse Table



Nurse
nurse_id:int
patient_id:int
emp_id:int

1NF Compliance: The table is in the First Normal Form (1NF) as it contains atomic values in each column, and there are no repeating groups.

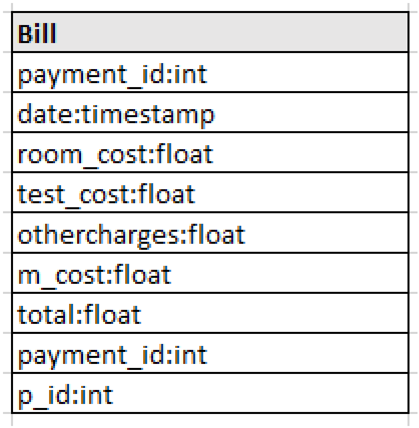
2NF Compliance: It is also in the Second Normal Form (2NF) because it has a primary key, "nurse_id," which uniquely identifies each row, and all non-key attributes are fully functionally dependent on the primary key.

3NF Compliance: The table satisfies the Third Normal Form (3NF) because it does not contain any transitive dependencies. All non-key attributes are directly dependent on the primary key, "nurse_id."

Functional Dependencies:
nurse_id → patient_id, emp_id

This set of functional dependencies indicates that for each individual nurse, identified by "nurse_id," the attributes "patient_id" and "emp_id" are directly determined. There are no transitive dependencies, ensuring that the "Nurse" table is well-structured and adheres to 3NF principles.

IX. Bill Table



Bill
payment_id:int
date:timestamp
room_cost:float
test_cost:float
othercharges:float
m_cost:float
total:float
payment_id:int
p_id:int

1NF Compliance: The table is in the First Normal Form (1NF) because it contains atomic values in each column, and there are no repeating groups.

2NF Compliance: It is also in the Second Normal Form (2NF) since it has a primary key, "payment_id," which uniquely identifies each row, and all non-key attributes are fully functionally dependent on the primary key.

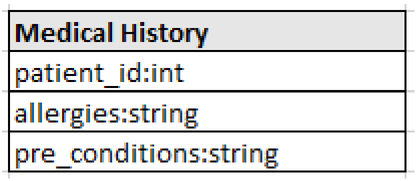
3NF Compliance: The table satisfies the Third Normal Form (3NF) because it does not contain any transitive dependencies. All non-key attributes are directly dependent on the primary key, "payment_id" and "p_id."

Functional Dependencies:

payment_id, p_id → date, room_cost, test_cost, othercharges, m_cost, total

This set of functional dependencies indicates that for each payment and associated patient (identified by "payment_id" and "p_id"), the attributes "date," "room_cost," "test_cost," "othercharges," "m_cost," and "total" are directly determined. There are no transitive dependencies, ensuring that the "Bill" table is well-structured and adheres to 3NF principles.

X. Medical History Table



Medical History
patient_id:int
allergies:string
pre_conditions:string

1NF Compliance: The table is in the First Normal Form (1NF) because it contains atomic values in each column, and there are no repeating groups.

2NF Compliance: It is also in the Second Normal Form (2NF) since it has a primary key, "patient_id," which uniquely identifies each row, and all non-key attributes are fully functionally dependent on the primary key.

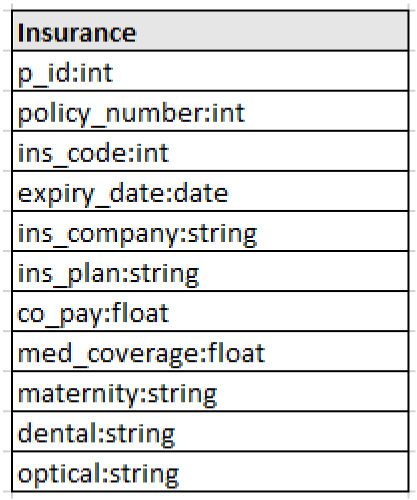
3NF Compliance: The table satisfies the Third Normal Form (3NF) because it does not contain any transitive dependencies. All non-key attributes are directly dependent on the primary key, "patient_id."

Functional Dependencies:

patient_id → allergies, pre_conditions

This set of functional dependencies indicates that for each patient, identified by "patient_id," the attributes "allergies" and "pre_conditions" are directly determined. There are no transitive dependencies, ensuring that the "Medical History" table is well-structured and adheres to 3NF principles.

XI. Insurance Table



Insurance
p_id:int
policy_number:int
ins_code:int
expiry_date:date
ins_company:string
ins_plan:string
co_pay:float
med_coverage:float
maternity:string
dental:string
optical:string

1NF Compliance: The table is in the First Normal Form (1NF) because it contains atomic values in each column, and there are no repeating groups.

2NF Compliance: It is also in the Second Normal Form (2NF) since it has a primary key, "p_id," which uniquely identifies each row, and all non-key attributes are fully functionally dependent on the primary key.

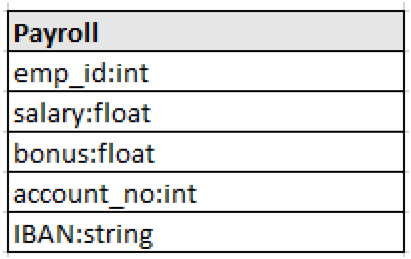
3NF Compliance: The table satisfies the Third Normal Form (3NF) because it does not contain any transitive dependencies. All non-key attributes are directly dependent on the primary key, "p_id."

Functional Dependencies:

p_id → policy_number, ins_code, expiry_date, ins_company, ins_plan, co_pay, med_coverage, maternity, dental, optical

This set of functional dependencies indicates that for each patient and their insurance information, identified by "p_id," the attributes like "policy_number," "ins_code," "expiry_date," "ins_company," "ins_plan," "co_pay," "med_coverage," "maternity," "dental," and "optical" are directly determined. There are no transitive dependencies, ensuring that the "Insurance" table is well-structured and adheres to 3NF principles.

XII. Payroll Table



Payroll
emp_id:int
salary:float
bonus:float
account_no:int
IBAN:string

1NF Compliance: The table is in the First Normal Form (1NF) because it contains atomic values in each column, and there are no repeating groups.

2NF Compliance: It is also in the Second Normal Form (2NF) since it has a primary key, "emp_id," which uniquely identifies each row, and all non-key attributes are fully functionally dependent on the primary key.

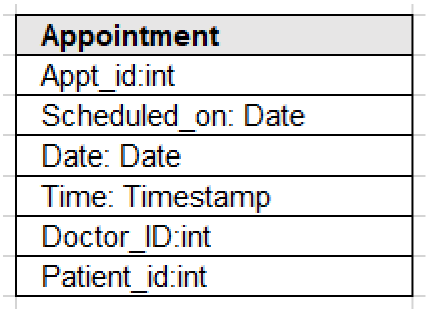
3NF Compliance: The table satisfies the Third Normal Form (3NF) because it does not contain any transitive dependencies. All non-key attributes are directly dependent on the primary key, "emp_id."

Functional Dependencies:

emp_id → salary, bonus, account_no, IBAN

This set of functional dependencies indicates that for each employee and their payroll information, identified by "emp_id," the attributes such as "salary," "bonus," "account_no," and "IBAN" are directly determined. There are no transitive dependencies, ensuring the "Payroll" table is well-structured and adheres to 3NF principles.

XIII. Appointment Table



Appointment
Appt_id:int
Scheduled_on: Date
Date: Date
Time: Timestamp
Doctor_ID:int
Patient_id:int

1NF Compliance: The table is in the First Normal Form (1NF) because it contains atomic values in each column, and there are no repeating groups.

2NF Compliance: It is also in the Second Normal Form (2NF) since it has a primary key, "Appt_id," which uniquely identifies each row, and all non-key attributes are fully functionally dependent on the primary key.

3NF Compliance: The table satisfies the Third Normal Form (3NF) because it does not contain any transitive dependencies. All non-key attributes are directly dependent on the primary keys, "Appt_id," "Doctor_ID," and "Patient_id."

Functional Dependencies:

Appt_id → Scheduled_on, Date, Time, Doctor_ID, Patient_id
Doctor_ID → Doctor_name, Doctor_specialization
Patient_id → Patient_name, Patient_date_of_birth

This set of functional dependencies indicates that for each appointment, identified by "Appt_id," the attributes "Scheduled_on," "Date," "Time," "Doctor_ID," and "Patient_id" are directly determined. There are no transitive dependencies, ensuring that the "Appointment" table is well-structured and adheres to 3NF principles.

6. Conclusion

In conclusion, this data model was made to help hospital staff maintain information and improve access, making the retrieval process easier. The Hospital database management must be improved or upgraded to meet any situation. In this report, we discuss and design the relational schema of the Hospital Management System Database. Our EER diagram and its associated relational schema show the conceptual and logical designs of the system. We also defined data types, assumptions and constraints for each attribute in the relations. The next step is to implement this database and change the design accordingly. The developed system and its evaluation should be carried out to improve the database system and management processes in hospitals. It is capable of storing a variety and large volume of databases. More so, the software has been designed to include program modules to handle the Medical Centre information such as patients' data, supply management, patients bill etc. Thus, this software contains the database files of patients, doctors, nurses and departments of a hospital and should provide the necessary information which will be compatible, accurate, flexible, secured and efficient for the desired purpose it is to serve.