



Course Project - Phase I

Team-51

1. INTRODUCTION TO MINI-WORLD : HOSPITAL DATABASE


Our Mini-World is a virtual model of a hospital, showcasing various aspects of healthcare and available services. It includes administration, doctors, healthcare workers, and additional services like pharmaceuticals. The system tracks patient admissions, doctor assignments, medical treatments, and resource management, facilitating smooth hospital operations. This database is designed to improve healthcare efficiency by managing essential services and streamlining the flow of information across departments.

2. PURPOSE OF THE DATABASE

- a. The purpose of the Hospital Database is to ensure **efficient management and organization** of critical healthcare services.
- b. **Manage patient records** including past medical conditions, treatments, and prescriptions.
- c. **Track hospital resources** like doctors, medical staff, medical supplies, finances and equipment.
- d. **Streamline scheduling** for appointments, surgeries, and diagnostics.
- e. **Handle administrative tasks** such as staff management, billing, and insurance.
- f. Facilitate **emergency services** like ambulances and real-time patient care tracking.
- g. **Manage employee information**, including healthcare workers and administrative staff, tracking their roles, schedules, and performance.

3. USERS OF THE DATABASE

Our database can be used by any personnel with authorized access and who want to gain knowledge about any particular records of the hospital.

- 
- a. **Doctors and Nurses** : They are the people closely associated with patients who have access to patient cases and operate on them accordingly.
 - b. **Patients** : Through a possible portal, to view their medicinal records, case reports, Meeting Briefings, appointments, and billing details.
 - c. **Administrative Staff** : It includes users such as hospital administrators, office managers and receptionists who can use our database to manage appointments, patient records, and general hospital operations.
 - d. **Pharmacists** : We will provide them with limited data of a patient regarding the prescription of medicines so that they can provide patients with medicines according to their prescriptions and manage medication inventory and dispensing records.
 - e. **Medical Students and Teachers** : For learning purposes, we will provide selective access to the medical cases of the patients to the medical students and teachers.
 - f. **IT and Database Administrators** : For managing, maintaining, and securing the database system.
 - g. **Hospital Management** : We will give access to the database to the Quality Assurance and Research teams so that they can keep track of the hospital needs and medical supplies, conduct statistical analysis of patient status and develop strategic plans for improvement.

4. DATABASE REQUIREMENTS

- a. **Assumptions:**
 - i. To a case, only one doctor is assigned to a case. A further requirement of another doctor can be considered a fresh entry.
 - ii. Cases that require authorizations from police, and government legal authorities are not considered.
 - iii. For each department there is at least one doctor.
 - iv. Each doctor works in exactly one department.
 - v. The last Id in the `past_medical_cases` will be the current case for any patient.

b. Strong Entity Types:

i. Doctor:

Name of Attribute	Data Type	Constraints
Doctor_Id	Integer	Primary Key
Name	varchar(255)	NOT NULL
Date_of_Birth	Date	NOT NULL
Paycheck	Integer	NOT NULL
Date_of_Joining	varchar(255)	NOT NULL
Age	varchar(255)	NOT NULL
Sex	Character	NOT NULL
Department_Id	varchar(255)	Foreign Key
Supervisor_Id	Integer	-
Previous_Degrees	varchar(255)	NOT NULL
Address	varchar(255)	NOT NULL
Phone_no	Integer	NOT NULL, 10 Digits

1. Subclass 1: Specialist (additional attributes of doctor)

Name of Attribute	Data Type	Constraints
Specialization_Area	varchar(255)	NOT NULL
Is_Surgeon	Boolean	NOT NULL

- **Name** is a COMPOSITE attribute with firstname and lastname. **Address** is COMPOSITE attribute as its composed of several components like (Apt#, House#, Street, City, State, ZipCode, Country).
- **Previous_Degrees** is a Composite Multivalued Attribute denoted by {Previous_Degrees(College,Year,Degree,Field)}

- **Age** is a DERIVED attribute from Date_of_Birth.
- Specialization is Cardiologist,Neurologist,Oncologist,General Practitioner, etc.

ii. Patient:

Name of Attribute	Data Type	Constraints
Patient_ Id	varchar(255)	Primary Key
Name	varchar(255)	NOT NULL
Sex	Character	NOT NULL
Blood_group	varchar(255)	NOT NULL
Phone_no	Integer	NOT NULL, 10 Digits
Past_Medical_cases	Integer	-
Insurance_Id	varchar(255)	NOT NULL
Age	Integer	NOT NULL
Diabetic	Boolean	NOT NULL
Blood_pressure	Boolean	NOT NULL

- **Name** is a COMPOSITE attribute with firstname and lastname.
- **Past_Medical_Cases** is a MULTIVALUED attribute.

iii. Inventory

Name of Attribute	Data Type	Constraints
Item_Name	varchar(255)	PRIMARY KEY
Supplier_Phone_No	Integer	NOT NULL, 10 Digits
Available_Stock	Integer	NOT NULL
Min_Stock_Level	Integer	NOT NULL

Name of Attribute	Data Type	Constraints
Item_Name	varchar(255)	PRIMARY KEY
Supplier_Phone_No	Integer	NOT NULL, 10 Digits
Purchase_Date	Date	NOT NULL
Expiry_Date	Date	NOT NULL
Cost_Per_Unit	Integer	NOT NULL

- If the item is below min_stock_level, then we have to order that item

iv. Department

Name of Attribute	Data Type	Constraints
Department_Id	Integer	Primary Key
Department_Name	varchar(255)	Primary Key
Department_Head_Id	Integer	Foreign Key
Floor_No	Integer	NOT NULL
Block_Name	varchar(255)	NOT NULL

v. Nurse

Name of Attribute	Data Type	Constraints
Nurse_Id	Integer	PRIMARY KEY
Name	varchar(255)	NOT NULL
Sex	Character	NOT NULL
Date_of_Birth	Date	NOT NULL
Previous_Degrees	varchar(255)	NOT NULL
Paycheck	Integer	NOT NULL
Department_Id	Integer	FOREIGN KEY

- **Name** is a COMPOSITE attribute with firstname and lastname.
- **Previous_Degrees** is a Composite Multivalued Attribute denoted by {Previous_Degrees(College,Year,Degree,Field)}

vi. Medical_Record

Name of Attribute	Data Type	Constraints
Case_Id	Integer	PRIMARY KEY
Case_start_date	DATE	NOT NULL
Department_Id	varchar(255)	FOREIGN KEY
Doctor_Id	Integer	FOREIGN KEY
Recent_visit	DATE	NOT NULL
Next_Appointment	DATE	NOT NULL
Total_visits	Integer	NOT NULL
Patient_Id	Integer	FOREIGN KEY
Status	varchar(255)	NOT NULL

vii. Accountant

Name of Attribute	Data Type	Constraints
Accountant_Id	Integer	PRIMARY KEY
Name	varchar(255)	NOT NULL
Sex	Character	NOT NULL
Date_of_Birth	DATE	NOT NULL
Paycheck	Integer	NOT NULL

viii. Receptionist

Name of Attribute	Data Type	Constraints
Receptionist_Id	Integer	PRIMARY KEY
Name	varchar(255)	NOT NULL
Sex	Char	NOT NULL
Date_of_Birth	DATE	NOT NULL
Paycheck	Integer	NOT NULL

- **Name** in the above examples are Composite Attributes.

ix. General_Staff

Name of Attribute	Data Type	Constraints
Staff_Id	Integer	PRIMARY KEY
Name	varchar(255)	NOT NULL
Sex	Char	NOT NULL
Date_of_Birth	DATE	NOT NULL
Hourly_wage	Integer	NOT NULL
Shift	Integer	NOT NULL
Weekly_wage	Integer	NOT NULL
Weekly_Hours	Integer	NOT NULL

- **Weekly_wages** is DERIVED Attribute from Hourly_wage and Weekly_Hours.

c. Weak Entity Types:

i. Invoice

Name of Attribute	Data Type	Constraints
Case_Id	Integer	PARTIAL KEY
Consultating_charges	Integer	NOT NULL
Operational_charges	Integer	-

Medicinal_charges	Integer	-
Transaction_Id	Integer	NOT NULL

- **Note:** Assuming there is ONLY ONE Invoice per Medical_Record

ii. Prescription:

Name of Attribute	Data Type	Constraints
Case_Id	Integer	FOREIGN KEY
Prescription_no	Integer	PARTIAL KEY
Medicines	varchar(255)	NOT NULL

- For a particular Case_Id, the Doctor prescribes many prescriptions whose Prescription_no starts from 1. So a prescription is uniquely identified by tuple of $\langle \text{Prescription_no}, \text{Case_Id} \rangle$. Hence, both together form a Partial Key. Medicines is a multivalued attribute.

d. Relationship Types:

i. Recursive Relationships:

1. Doctor Supervision

- A senior Doctor supervises a junior doctor
- Cardinality ratio (1:N)
 - For Junior doctor min,max : (0,1)
 - For Senior doctor min,max : (0, N)
- Attribute of this relationship
 - Performance Feedback:** Summary or notes on performance from the senior doctor.

ii. Binary relationships

1. Patient Medical History

- a. Participating entities are two strong entities: Patient and Medical_Record
- b. Cardinality ratio (N:1)
 - i. For Medical_Record min,max : (1,1)
 - ii. For Patient min,max : (1, N)

2. Departmental Medical Record

- a. Participating Entities: Medical_Record, Department
- b. Cardinality Ratio (N:1)
 - i. For Medical_Record min,max : (1,1)
 - ii. For Department min,max : (0,N)

3. Doctor **WORKS_IN** Department

- a. Participating entity types are two strong entities: Doctor and Department.
- b. Cardinality ratio (N:1)
 - i. For Doctor min, max (1,1)
 - ii. For Department (1, N)

4. Medical Record Billing

- a. Participating entity types are strong entity and weak entity respectively: Medical_Record and Invoice.
- b. Cardinality ratio (1:1)
 - iii. For Invoice min,max : (1,1)
 - iv. For Medical_Record min,max : (1,1)

5. Appointment Scheduling

- a. Participating entity types are two strong entities: Receptionist and Patient.

- b. Cardinality ratio (1:N)
 - i. For Receptionist min,max : (0, N)
 - ii. For Patient min,max : (1,1)

6. Doctor **IS_HEAD_OF** Department

- a. Participating entity types are two strong entities: Doctor and Department.
- b. Cardinality ratio (1:1)
 - i. For Doctor min,max : (0,1)
 - ii. For Department min,max : (1,1)

7. Prescription **ISSUED_IN** Medical_Record

- a. Participating entity types are strong entity and weak entity respectively: Medical_Record and Prescription.
- b. Cardinality ratio (N:1)
 - i. For Prescription min,max : (1,1)
 - ii. For Medical_Record min,max : (1,N)

8. Nurse **MONITORS** Patient

- a. Participating entity types are two strong entities: Nurse and Patient.
- b. Cardinality ratio (M:N)
 - i. For Nurse min,max : (1,N)
 - ii. For Patient min,max : (1,M)

iii. Ternary Relationships

1. Medication Issuance (A doctor will issue a prescription to a patient)

- a. Participating entity types are three entities: Doctor, Prescription and Patient.
- b. Cardinality Ratio: (1:N:1)
 - i. For Doctor min,max : (1,N)
 - ii. For Prescription min,max : (1,1)

- iii. For Patient min,max : (1,N)

2. Patient Billing

- a. Participating entity types are three entities: Invoice, Patient, and Accountant.
- b. Cardinality Ratio: (N:1:1)
 - i. For Invoice min,max : (1,1)
 - ii. For Patient min,max : (1,N)
 - iii. For Accountant min,max : (1,N)

iv. Quaternary Relationships

1. Surgery Assignment

- a. Participating Entity Types: Doctor, Patient, Nurse, Medical_Record
- b. This relationship represents a scenario where a doctor, assisted by a nurse, performs a surgery on a patient, with details recorded in the patient's medical record.
- c. Cardinality Ratio: (1:1:M:N)
 - i. For Doctor min,max : (1,N)
 - ii. For Patient min,max : (1,N)
 - iii. For Nurse min,max : (1,N)
 - iv. For Medical_Record min,max : (1,1)

5. FUNCTIONAL REQUIREMENTS

a. Modification

i. INSERT OPERATION

- INSERT INTO Department VALUES (5, "Radiology", 123, 2, "A");
 - **Key Constraint:** While inserting a department in the Department table, we will check the whole table if there exists any entry with Department_id=5 or Department_name="Radiology". If it exists then error will be triggered otherwise commit the insertion.

- **Entity Integrity:** We will also check whether the key values(Department_id and Department_name) are not NULL.
- **Referential Integrity:** We will check whether the Department_Head_id in the tuple is an actual ID of a Doctor in the Doctor table. If there doesn't exist any entry in the Doctor table with Doctor_id = Department_Head_id , then error will be triggered otherwise insertion will be committed.
- **Domain Constraint:** If any of the values in the tuple violates the domain of the respective attribute, then error will be triggered otherwise insertion will be committed.
- During an INSERT operation in the database, various integrity constraints are enforced to ensure data validity and consistency. These include **Entity Integrity**, which ensures that primary key attributes are not NULL; **Key Constraints**, which prevent the insertion of duplicate records based on key attributes; **Referential Integrity**, which verifies that foreign key references correspond to existing records in related tables; and **Domain Constraints**, which ensure that attribute values adhere to defined data types and acceptable ranges. If any of these constraints are violated, an error is triggered, and the insertion is aborted; otherwise, the new record is committed successfully.

ii. DELETE OPERATION

1. DELETE FROM General_Staff WHERE Staff_Id=13354
2. DELETE FROM Doctor WHERE Doctor_Id=34322
3. If DELETE operation violates referential integrity we will use the SET NULL option: set the foreign keys of the referencing tuples to NULL

iii. UPDATE OPERATION

1. UPDATE operation is similar to first deleting and then inserting in the same place. Some examples are given below :-
2. UPDATE Nurse SET Paycheck=20,000 WHERE Nurse_Id= 12345;

3. UPDATE Medical_Record SET Next_Appointment=12/01/2023 WHERE Case_Id= 10011;

b. Retrieval

i. SELECTION QUERY

1. Here we are listing all the tuples of Doctor who are working in a particular department.

```
SELECT * FROM Doctor WHERE Department_Id = 2
```

2. Here we are printing all the details of a particular patient including their cases, total visits, appointments etc.

```
SELECT * FROM Patient WHERE Patient_Id=45633.
```

ii. AGGREGATION QUERY

1. Performing Operations such as printing the average age of the employee (Doctor, Accountant, Receptionist, etc.) or the sum of the salary or hourly_wage of all the workers or maximum salary of a nurse.

```
SELECT AVG(Age) FROM Doctor
```

```
SELECT SUM(Hourly_wage) FROM Worker
```

```
SELECT MAX(Paycheck) FROM Nurse
```

iii. SEARCH QUERY

1. Here we are printing all the details of some Doctors whose name contains the substring "ra".

```
SELECT * FROM Doctor WHERE Name LIKE '%ra%'
```

iv. PROJECTION QUERY

1. Print all the details of the patient whose past medical records are greater than or equal to 5.

```
SELECT * FROM Patient WHERE Past_Medical_Cases >= 5
```

c. ANALYSIS

i. Report on Doctor Efficiency in Each Department:

1. **Query:** Find the number of patients treated by each doctor in each department, along with the average number of visits per patient.
2. **Purpose:** This report helps analyze doctor performance and patient load across

departments.

```
SELECT Doctor.Name, Department.Department_Name,  
COUNT(Patient.Patient_ID) AS Total_Patients, AVG(Medical_Record.Total_visits)  
AS Avg_Visits  
FROM Doctor  
JOIN Medical_Record ON Doctor.Doctor_ID = Medical_Record.Doctor_ID  
JOIN Patient ON Medical_Record.Patient_ID = Patient.Patient_ID  
JOIN Department ON Doctor.Department_ID = Department.Department_ID  
GROUP BY Doctor.Name, Department.Department_Name;
```

ii. **Report on Total Hospital Charges per Patient:**

1. **Query:** Calculate the total charges incurred by each patient, including consultation, operational, and medicinal charges.
2. **Purpose:** This report helps analyze patient billing and the financial performance of the hospital by understanding how charges accumulate per patient.

```
SELECT Patient.Name, SUM(Invoice.Consultating_charges+  
Invoice.Operational_charges + Invoice.Medicinal_Charges)  
FROM Patient  
JOIN Medical_Record ON Patient.Patient_ID =  
Medical_Record.Patient_ID  
JOIN Invoice ON Medical_Record.Case_ID = Invoice.Case_ID  
GROUP BY Patient.Name;
```



6. Summary

The Hospital Database Project is a system designed to streamline healthcare operations by managing patient records, doctor assignments, medical treatments, and hospital resources. It serves various users, including doctors, nurses, administrative staff, and patients, while ensuring data security and authorized access. The system tracks patient admissions, appointments, billing, and staff schedules. Key features include managing medical records, prescriptions, invoices, and hospital resources. The project includes functionality for modifying and retrieving data through various SQL operations and provides reports for analyzing doctor performance and hospital charges.