UTM
UNIVERSITI TEKNOLOGI MALAYSIA

Malaysia-Japan International Institute of Technology (MJIIT)

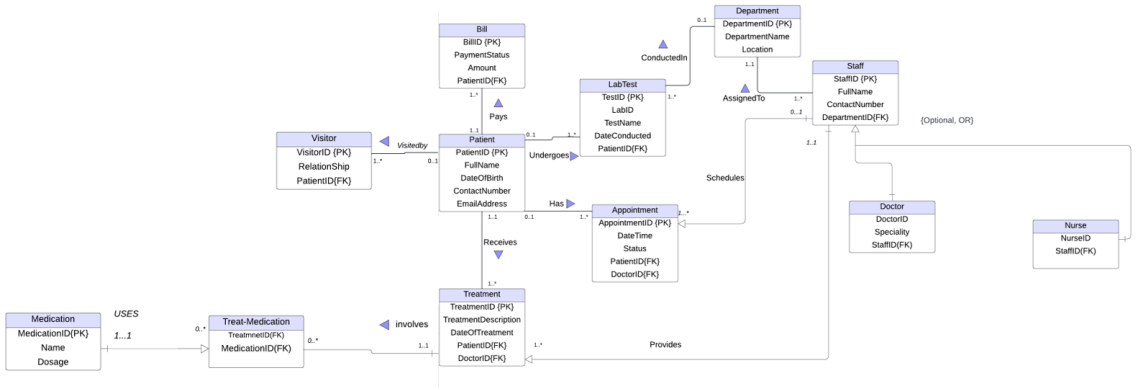
SECD2523-DATABASE
20242025 – SEMESTER 1
PHASE 4

Logical Design

FACULTY OF MJIIT

NAME	MATRIC ID
Liu Ruoyang	A23MJ4022
Pranto Anik Islam	A23MJ4024
Kahlan Sultan	A23MJ4021
Bu Guoshun	A23MJ4019
Saumik Hasan	A23MJ3009

- **Logical ERD :**



- **Relations schema for the above Logical ERD:**

1.Patient(PatientID (PK), FullName, DateOfBirth, ContactNumber, EmailAddress)
2.Visitor(Visitor (PK), Relationship, PatientID (FK))
3.Bill(BillID (PK), Amount, PaymentStatus, PatientID (FK))
4.LabTest(TestID (PK), TestName, DateConducted, LabID, PatientID (FK))
5.Department(DepartmentID (PK), DepartmentName, Location)
6.Staff(StaffID (PK), FullName, ContactNumber, DepartmentID (FK))
7.Doctor(DoctorID (PK), Specialty, StaffID (FK))
8.Nurse(NurseID (PK), StaffID (FK))
9.Appointment(AppointmentID (PK), DateTime, Status, PatientID (FK), DoctorID (FK))

10.Medication(MedicationID (PK), Name, Dosage)
11.Treatment(TreatmentID (PK), TreatmentDescription, DateOfTreatment, PatientID (FK), DoctorID (FK))
12.Treat-Medication (TreatmentID, MedicationID)

- **Normalization up till BCNF :**

1. **Patient**(PatientID (PK), FullName, DateOfBirth, ContactNumber, EmailAddress)

    - **First Normal Form**(removing repeating group)
      The relation is in 1NF
    - **Second Normal Form**(removing partial dependency)

The relation is in 2NF

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

- Potential dependency: ContactNumber → FullName, DateOfBirth, EmailAddress
    - A contact number might uniquely identify a patient, indicating a transitive dependency.

Decompose into 3NF:

To remove the transitive dependency, we split the table into:

1. **Patient :**
    - Attributes: PatientID (PK), Fullname, DateOfBirth , EmailAddress .
2. **Patient**:
    - Attributes: PatientID (PK,FK), ContactNumber .

Boyce-Codd Normal Form (BCNF)

Checking BCNF for Each Table:

- Both table satisfies BCNF.

2. Visitor(Visitor (PK), Relationship, PatientID (FK))
Functional Dependencies (FDs):
VisitorID → Relationship, PatientID

- **First Normal Form**(removing repeating group)
  The relation is in 1NF
- **Second Normal Form**(removing partial dependency)
  The relation is in 2NF
- **Third Normal Form**(removing transitive dependency)

Third Normal Form (3NF)

- To remove potential transitive dependencies we can make VisitorID and PatientID as composite Primary Key.

  Now the Schema is Visitor((VisitorID,PatientID)(PK),Relationship)

  - **1NF:** Satisfied because all attributes are atomic.

  - **2NF:** Satisfied because there are no partial dependencies.
  - **BCNF:** Also satisfied because every functional dependency has a superkey as its determinant.

3. Bill(BillID (PK), Amount, PaymentStatus, PatientID (FK))
Functional Dependencies (FDs):

BillID→Amount,PaymentStatus,PatientID
(The BillID determines all other attributes.)

- **First Normal Form**(removing repeating group)
  The relation is in 1NF
- **Second Normal Form**(removing partial dependency)
  The relation is in 2NF
- **Third Normal Form**(removing transitive dependency)
- **BCNF :**
  This table already satisfies BCNF

4.**LabTest**(TestID (PK), TestName, DateConducted, LabID, PatientID (FK))
Functional Dependencies (FDs):

TestID→TestName,DateConducted,LabID,PatientID
(The TestID uniquely determines all other attributes.)

- **First Normal Form**(removing repeating group)

The relation is in 1NF

- **Second Normal Form**(removing partial dependency)

The relation is in 2NF

- **Third Normal Form**(removing transitive dependency)

1. Satisfy 2NF.
2. Remove transitive dependencies (non-prime attributes should not depend on other non-prime attributes).

Functional Dependencies:

1. TestID → TestName, DateConducted, LabID, PatientID (Primary key determines all attributes).
2. Possible dependency: LabID → TestName (If a specific lab always conducts specific tests).

To remove the transitive dependency (LabID → TestName), we decompose the table.

Decomposition into 3NF:

LabTest:

Attributes:

- TestID (PK, shared across subclasses).
- TestName (common to both tables).

LabTestInfo:

- TestID .
- LabID to .

LabTest Instances:

DateConducted, LabID, and PatientID .

Here LabID and PatientID together made composite Primary Key but as LabID or PatientID alone cannot determined DateCounducted so we are free form partial dependencies as well as 2NF.

Here we cannot find any partial dependency or transitive dependencies so we are free from 3NF.

BCNF: All the tables satisfies BCNF.

5. Department(DepartmentID (PK), DepartmentName, Location)
Functional Dependencies (FDs):
DepartmentID→DepartmentName,Location
(The DepartmentID uniquely determines all other attributes.)
First Normal Form(removing repeating group)
The relation is in 1NF

- **Second Normal Form**(removing partial dependency)
  The relation is in 2NF
- **Third Normal Form**(removing transitive dependency)

Criteria for 3NF:

1. Satisfy 2NF.
2. Remove transitive dependencies (non-prime attributes should not depend on other non-prime attributes).

- **BCNF:** This table already satisfies BCNF .

6. **Staff**(StaffID (PK), FullName, ContactNumber, DepartmentID (FK))
Functional Dependencies (FDs):
StaffID→FullName,ContactNumber,DepartmentID

- **First Normal Form**(removing repeating group)
  The relation is in 1NF
  - **Second Normal Form**(removing partial dependency)

  The relation is in 2NF
  - **Third Normal Form**(removing transitive dependency)

1. Satisfy 2NF.
2. Remove transitive dependencies (non-prime attributes should not depend on other non-prime attributes).

BCNF: This table already satisfies BCNF

7.Doctor(DoctorID (PK), Specialty, StaffID (FK))
Functional Dependencies (FDs):
DoctorID→Specialty,StaffID
(The DoctorID uniquely determines both the doctor's Specialty and StaffID.)

- **First Normal Form**(removing repeating group)
  The relation is in 1NF
- **Second Normal Form**(removing partial dependency)
  The relation is in 2NF
- **Third Normal Form**(removing transitive dependency)

1. Satisfy 2NF.

2. Remove transitive dependencies (non-prime attributes should not depend on other non-prime attributes).We removed StaffID from Doctor table to remove potential transitive dependencies and anomaly. We added Name attribute to get doctors name. As there is no transitive dependency the table satisfies 3NF.

Functional Dependencies:

1. DoctorID → Specialty,Name

    - The primary key (DoctorID) determines all other attributes.

There are no transitive dependencies so the table satisfies 3NF.

- **BCNF: This table is already in BCNF .**

8.Nurse(NurseID (PK), StaffID (FK))
Functional Dependencies (FDs):

1. NurseID→StaffID(FK)
   (The NurseID uniquely determines the associated StaffID.)

    - **First Normal Form**(removing repeating group)
      The relation is in 1NF
    - **Second Normal Form**(removing partial dependency)
      The relation is in 2NF
    - **Third Normal Form**(removing transitive dependency)

1. Satisfy 2NF.
2. Remove transitive dependencies (non-prime attributes should not depend on other non-prime attributes).

    - We romevode StaffID form Nurse table to remove potential dependencies and added Name for more details about Nurse .NurseID is the primary key and determines all other attributes.

Since the only determinant is the primary key (NurseID), the table satisfies 3NF.

- **BCNF:** This table satisfies BCNF

9.Appointment(AppointmentID (PK), DateTime, Status, PatientID (FK), DoctorID (FK))
Functional Dependencies (FDs):

AppointmentID→DateTime,Status,PatientID,DoctorID
AppointmentID uniquely determines all other attributes in the relation.

- **First Normal Form**(removing repeating group)
  The relation is in 1NF
- **Second Normal Form**(removing partial dependency)
  The relation is in 2NF
- **Third Normal Form**(removing transitive dependency)

1. Satisfy 2NF.
2. Remove transitive dependencies (non-prime attributes should not depend on other non-prime attributes).

Functional Dependencies:

1. AppointmentID → DateTime, Status, PatientID, DoctorID

    - The primary key (AppointmentID) determines all other attributes.

There are no transitive dependencies because DateTime, Status, PatientID, and DoctorID do not depend on one another.
Thus, the table satisfies 3NF.

- **BCNF:**

1. Satisfy 3NF.
2. For every functional dependency, the determinant must be a superkey.

Functional Dependencies:

- AppointmentID → DateTime, Status, PatientID, DoctorID

- AppointmentID is the primary key and determines all other attributes.

Since the only determinant is the primary key (AppointmentID), the table satisfies BCNF.

10.Medication(MedicationID (PK), Name, Dosage)
Functional Dependencies (FDs):
MedicationID→Name,Dosage
(The MedicationID uniquely determines both Name and Dosage.)

- **First Normal Form**(removing repeating group)

The relation is in 1NF

- **Second Normal Form**(removing partial dependency)
  The relation is in 2NF
- **Third Normal Form**(removing transitive dependency)

1. Satisfy 2NF.
2. Remove transitive dependencies (non-prime attributes should not depend on other non-prime attributes).

Functional Dependencies:

1. MedicationID → Name, Dosage

- The primary key (MedicationID) determines all other attributes.

- **BCNF:** This table satisfies BCNF.

11.Treatment(TreatmentID (PK), TreatmentDescription, DateOfTreatment, PatientID (FK), DoctorID (FK))

Functional Dependencies (FDs):
TreatmentID→TreatmentDescription,DateOfTreatment,PatientID,DoctorID
(The TreatmentID uniquely determines all other attributes.)

- **First Normal Form**(removing repeating group)
  The relation is in 1NF
- **Second Normal Form**(removing partial dependency)
  The relation is in 2NF
- **Third Normal Form**(removing transitive dependency)

Criteria for 3NF:

1. Satisfy 2NF.
2. Remove transitive dependencies (non-prime attributes should not depend on other non-prime attributes).

Functional Dependencies:

1. TreatmentID → TreatmentDescription, DateOfTreatment, PatientID, DoctorID
   - The primary key (TreatmentID) determines all other attributes.

There are no transitive dependencies
because TreatmentDescription, DateOfTreatment, PatientID, and DoctorID do not depend on one another or any other non-prime attribute.
Thus, the table satisfies 3NF.

BCNF:

1. Satisfy 3NF.
2. For every functional dependency, the determinant must be a superkey.

Functional Dependencies:

1. TreatmentID → TreatmentDescription, DateOfTreatment, PatientID, DoctorID
    - TreatmentID is the primary key and determines all other attributes.

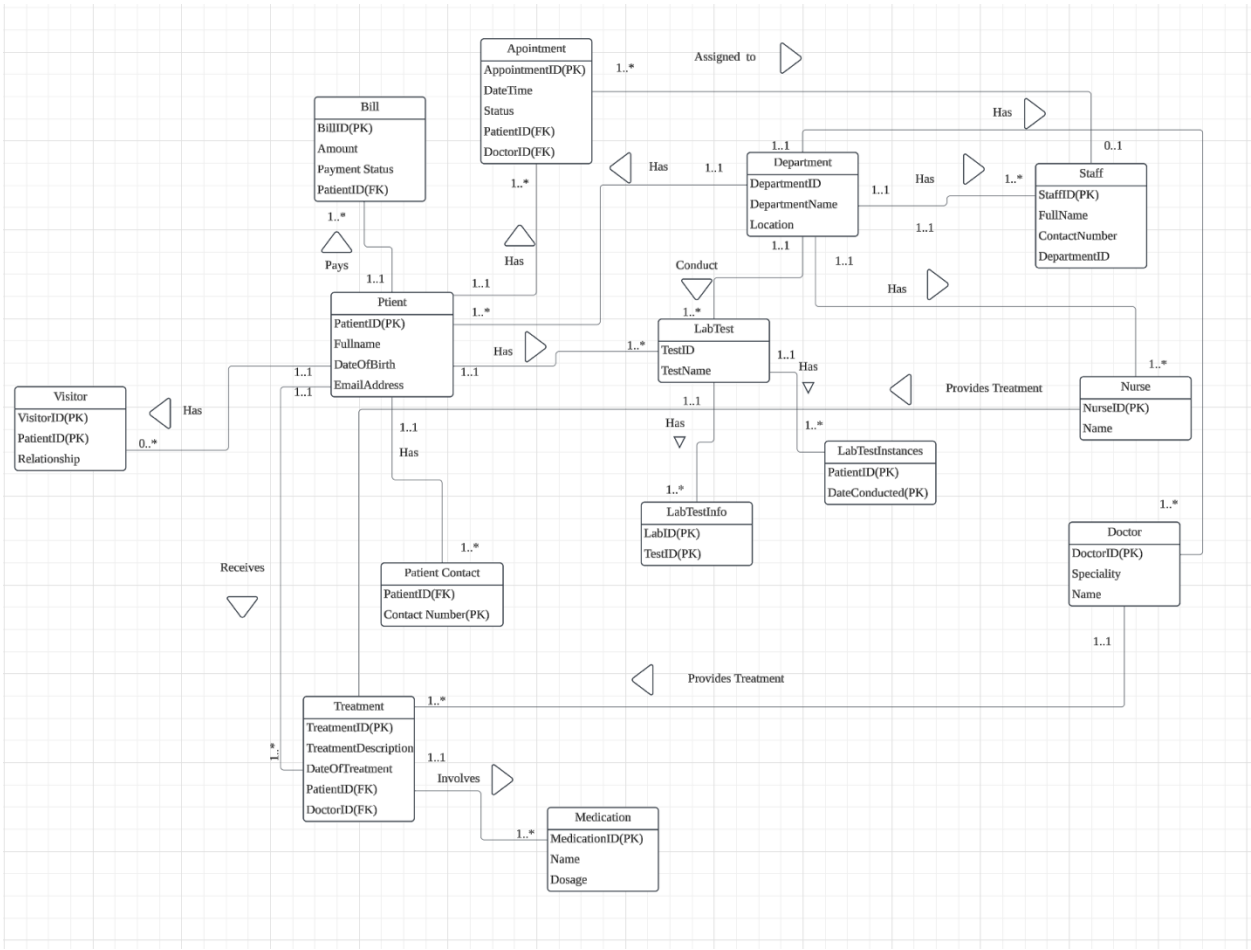
Since the only determinant is the primary key (TreatmentID), the table satisfies BCNF.

12.Treat-Medication (TreatmentID, MedicationID)
Functional Dependencies (FDs):
TreatmentID,MedicationID→(No additional attributes)
(Both attributes together form the composite primary key; no additional attributes are part of this schema.)

- **First Normal Form**(removing repeating group)
  The relation is in 1NF
- **Second Normal Form**(removing partial dependency)
  The relation is in 2NF
- **Third Normal Form**(removing transitive dependency)
  The relation is in 3NF

- **BCNF (** removing non-candidate key determinant)
  The relation is already in BCNF

Final Logical ERD:



Updated Data Dictionary for the Final Logical ERD:

Entity Name	Attribute Name	Data Type	Description	Key Type	Constrain
Patient	PatientID	INT	Unique Identifier for a Patient	Primary Key	Unique and Not null
	FullName	VARCHAR	Full name of the Patient		
	DateOfBirth	DATE	Patient's Date of Birth		
	EmailAddress	VARCHAR	Email Address of the Patient		
Patient Contact	PatientID	INT	Identifier of the Patient	Foreign Key	Not Null
	ContactNumber	VARCHAR	Patient's Contact Number	Primary Key	Unique and Not Null
Visitor	VisitorID	INT	Unique Identifier for the visitor	Primary Key	Unique and Not Null
	PatientID	INT	Identifier for the related Patient	Primary Key, Foreign Key	Not Null
	Relationship	VARCHAR	Relationship to Patient		
Bill	BillID		Unique identifier for Bill	Primary Key	Unique and Not Null

	Amount	DECIMAL	Total Bill Amount		
	PaymentStatus	VARCHAR	Status of Payment(Paid/ Unpaid)		
	PatientID	INT	Identifier for the related Patient	Foreign Key	
LabTest	TestID	INT	Unique Identifier for the Test.	Primary Key	Unique and Not Null
	TestName	VARCHAR	Name of the test		
LabTestInfo	LabID	INT	Unique Identifier for the Lab	Primary Key	Unique and Not Null
	TestID	INT	Identifier for the Related Test	Foreign Key	
LabTestInstances	TestID	INT	Identifier for the related Test	Primary Key	Unique and Not NULL
	PatientID	INT	Identifier for the related Patient	Primary Key	
	DateConducted	DATE	Date of the Lab Test when it was conducted.		
Department	DepartmentID	INT	Unique identifier for the Department.	Primary Key	Unique and Not Null
	DepartmentName	VARCHAR			
	Location	VARCHAR			
Staff	StaffID	INT	Unique Identifier for the staff member	Primary Key	Unique and Not Null
	FullName	VARCHAR	Name of the Staff member		
	ContactNumber	VARCHAR	Contact Number of the Staff member		

	DepartmentID	INT	Identifier for the related Departmen	Foreign key	
Doctor	DoctorID	INT	Unique identifier for a doctor	Primary Key	Unique and Not Null
	Speciality	VARCHAR	Doctor's area of expertise		
	Name	VARCHAR			
Nurse	NurseID	INT	Unique Identifier for a nurse	Primary Key	Unique and Not Null
	Name	VARCHAR			
Appointment	AppoinmentID	INT	Unique identifier for the appointment	Primary Key	Unique and Not Null
	DateTime	DATETIME			
	Status	VARCHAR	Status of the appointment		
	PatientID	INT	Identifier for the related Patient	Foreign Key	
	DoctorID	INT	Identifier for the assigned doctor	Foreign Key	
Treatment	TreatmentID	INT	Unique identifier for a treatment	Primary Key	Unique and Not Null
	TreatmentDescription	VARCHAR			
	DateOfTreatment	DATE	Date of the treatment		
	PatientID	INT	Identifier for the treated patient	Foreign Key	
	DoctorID	INT	Identifier for the doctor providing treatment	Foreign Key	
Medication	MedicationID	INT	Unique Identifier for medication	Primary Key	Unique and Not Null
	Name	VARCHAR	Name of the medication		

	Dosage	VARCHAR	Recommended dosage for medication		

Validation of Logical ERD based on the system's transaction requirements:

Creating Database Hospital and all the Tables:

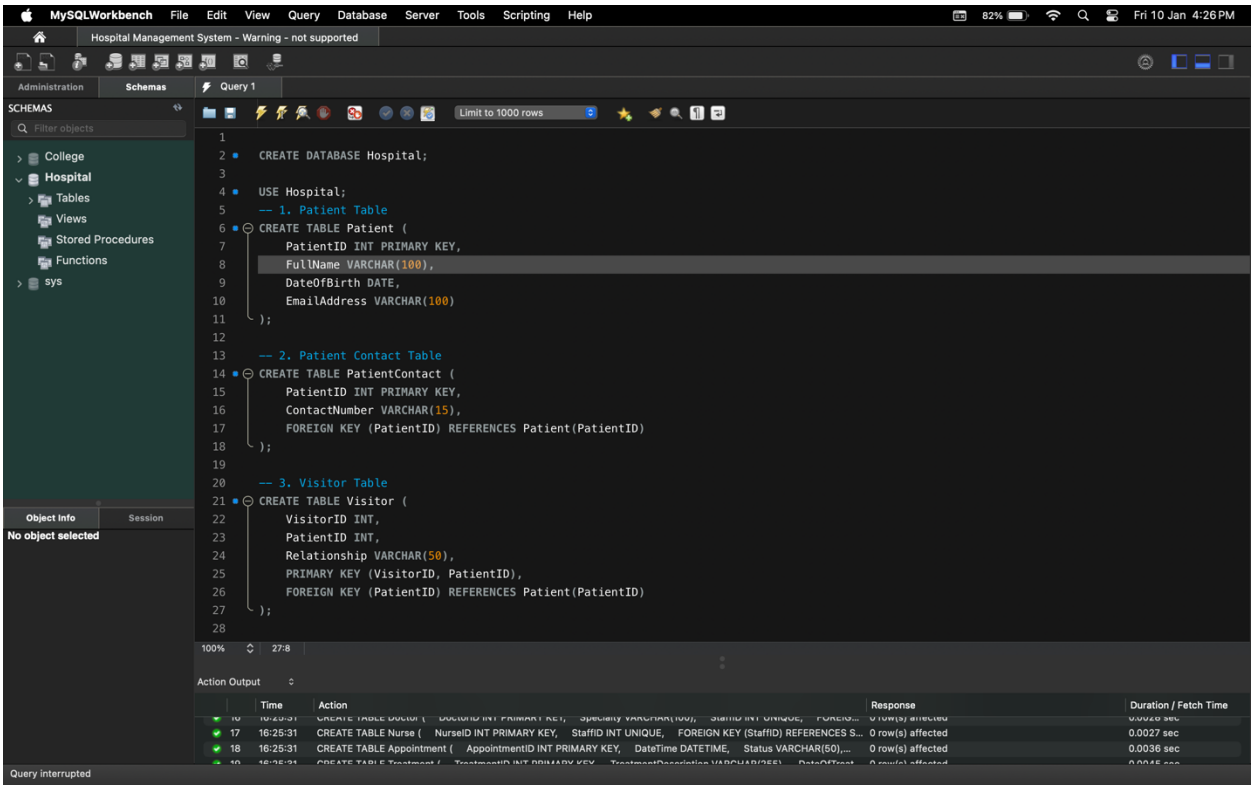
MYSQL Query:

```
CREATE DATABASE Hospital;
USE Hospital;
-- 1. Patient Table
CREATE TABLE Patient (
    PatientID INT PRIMARY KEY,
    FullName VARCHAR(100),
    DateOfBirth DATE,
    EmailAddress VARCHAR(100)
);

-- 2. Patient Contact Table
CREATE TABLE PatientContact (
    PatientID INT PRIMARY KEY,
    ContactNumber VARCHAR(15),
    FOREIGN KEY (PatientID) REFERENCES Patient(PatientID)
);

-- 3. Visitor Table
CREATE TABLE Visitor (
    VisitorID INT,
    PatientID INT,
    Relationship VARCHAR(50),
    PRIMARY KEY (VisitorID, PatientID),
    FOREIGN KEY (PatientID) REFERENCES Patient(PatientID)
);

-- 4. Bill Table
CREATE TABLE Bill (
    BillID INT PRIMARY KEY,
    Amount DECIMAL(10, 2),
    PaymentStatus VARCHAR(50),
    PatientID INT,
    FOREIGN KEY (PatientID) REFERENCES Patient(PatientID)
```

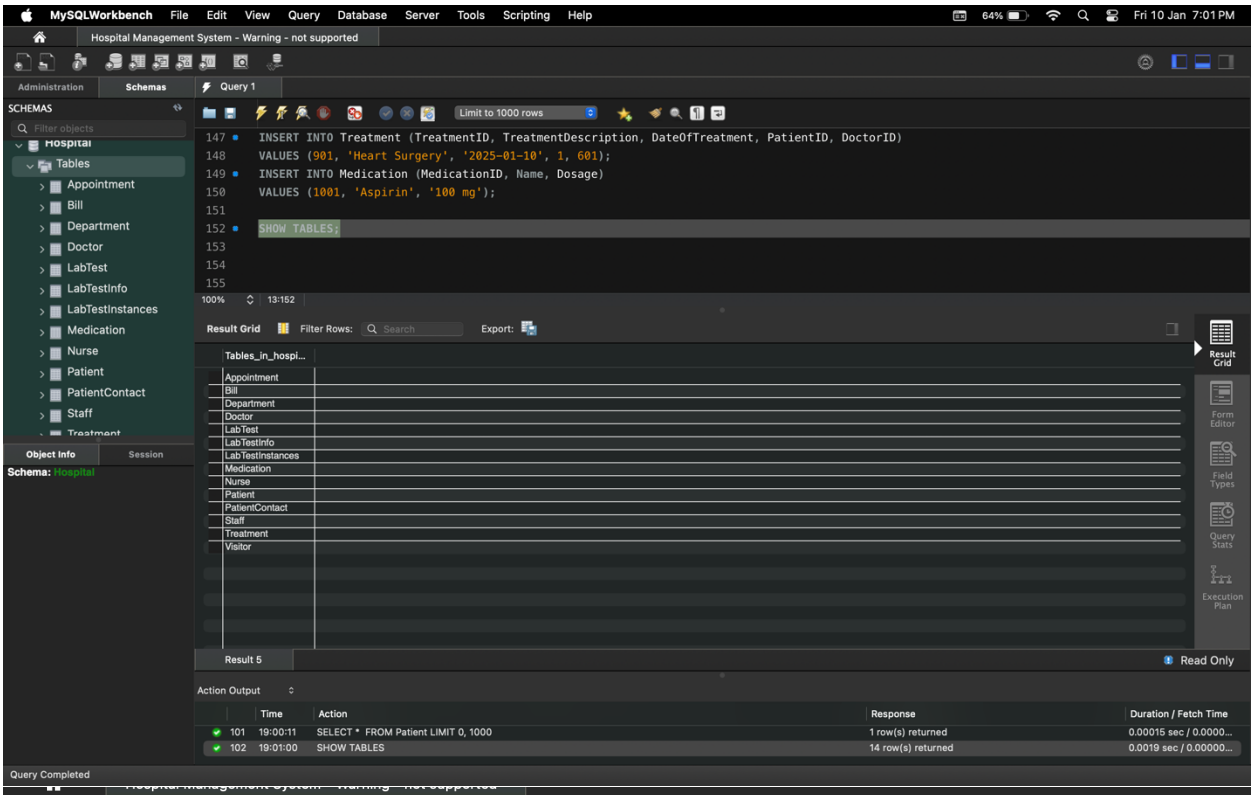
```
);

-- 5. LabTest Table
CREATE TABLE LabTest (
   TestID INT PRIMARY KEY,
   TestName VARCHAR(100)
);

-- 6. LabTestInfo Table
CREATE TABLE LabTestInfo (
   LabID INT PRIMARY KEY,
   TestID INT,
   FOREIGN KEY (TestID) REFERENCES LabTest(TestID)
);

-- 7. LabTestInstances Table
CREATE TABLE LabTestInstances (
   TestID INT,
   PatientID INT,
   DateConducted DATE,
   PRIMARY KEY (TestID, PatientID),
   FOREIGN KEY (TestID) REFERENCES LabTest(TestID),
   FOREIGN KEY (PatientID) REFERENCES Patient(PatientID)
);

-- 8. Department Table
CREATE TABLE Department (
   DepartmentID INT PRIMARY KEY,
   DepartmentName VARCHAR(100),
   Location VARCHAR(100)
);

-- 9. Staff Table
CREATE TABLE Staff (
   StaffID INT PRIMARY KEY,
   FullName VARCHAR(100),
   ContactNumber VARCHAR(15),
   DepartmentID INT,
   FOREIGN KEY (DepartmentID) REFERENCES Department(DepartmentID)
);

-- 10. Doctor Table
CREATE TABLE Doctor (
   DoctorID INT PRIMARY KEY,
   Specialty VARCHAR(100),
     Name VARCHAR
```

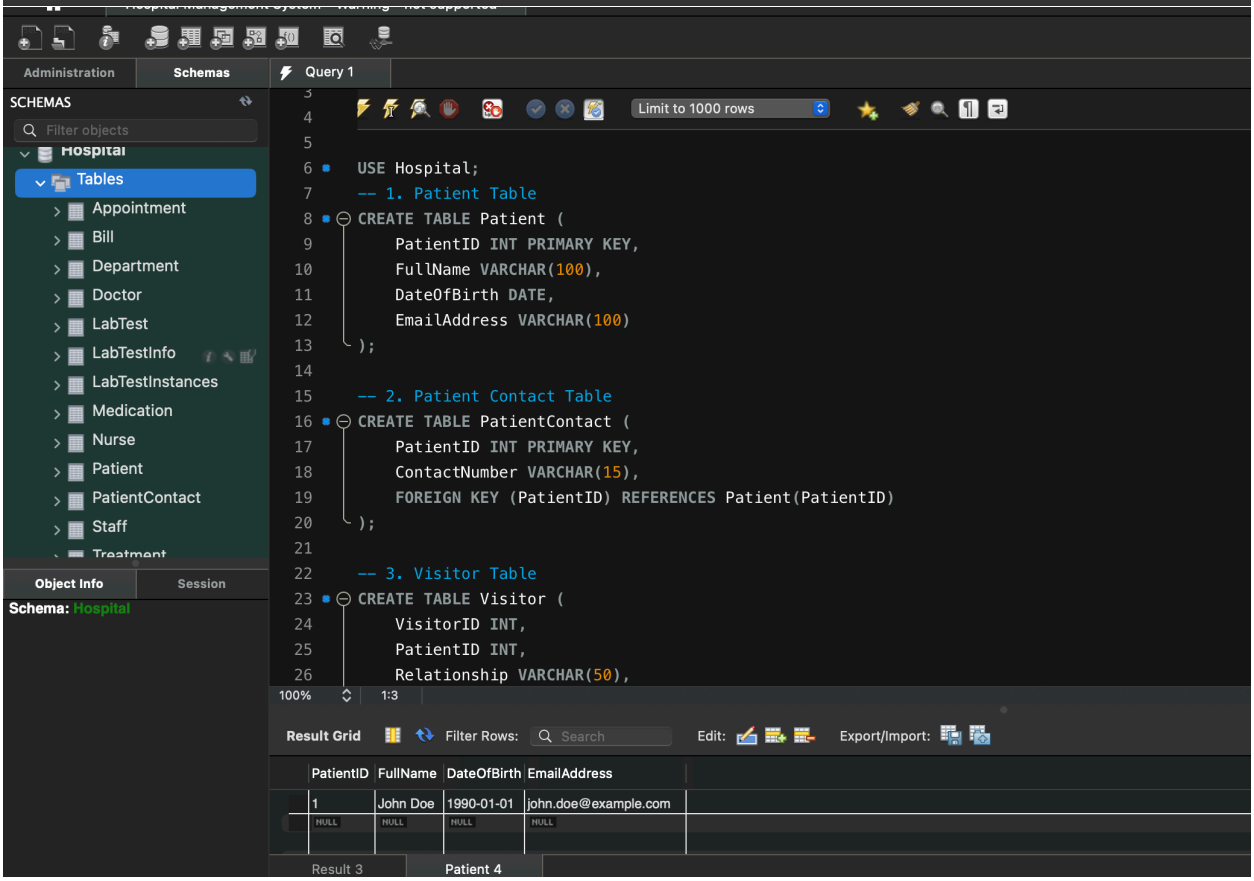
```
);

-- 11. Nurse Table
CREATE TABLE Nurse (
    NurseID INT PRIMARY KEY,
    Name VARCHAR
);

-- 12. Appointment Table
CREATE TABLE Appointment (
    AppointmentID INT PRIMARY KEY,
    DateTime DATETIME,
    Status VARCHAR(50),
    PatientID INT,
    DoctorID INT,
    FOREIGN KEY (PatientID) REFERENCES Patient(PatientID),
    FOREIGN KEY (DoctorID) REFERENCES Doctor(DoctorID)
);

-- 13. Treatment Table
CREATE TABLE Treatment (
    TreatmentID INT PRIMARY KEY,
    TreatmentDescription VARCHAR(255),
    DateOfTreatment DATE NOT NULL,
    PatientID INT,
    DoctorID INT,
    FOREIGN KEY (PatientID) REFERENCES Patient(PatientID),
    FOREIGN KEY (DoctorID) REFERENCES Doctor(DoctorID)
);

-- 14. Medication Table
CREATE TABLE Medication (
    MedicationID INT PRIMARY KEY,
    Name VARCHAR(100),
    Dosage VARCHAR(50)
);
```

Database and all the table created successfully

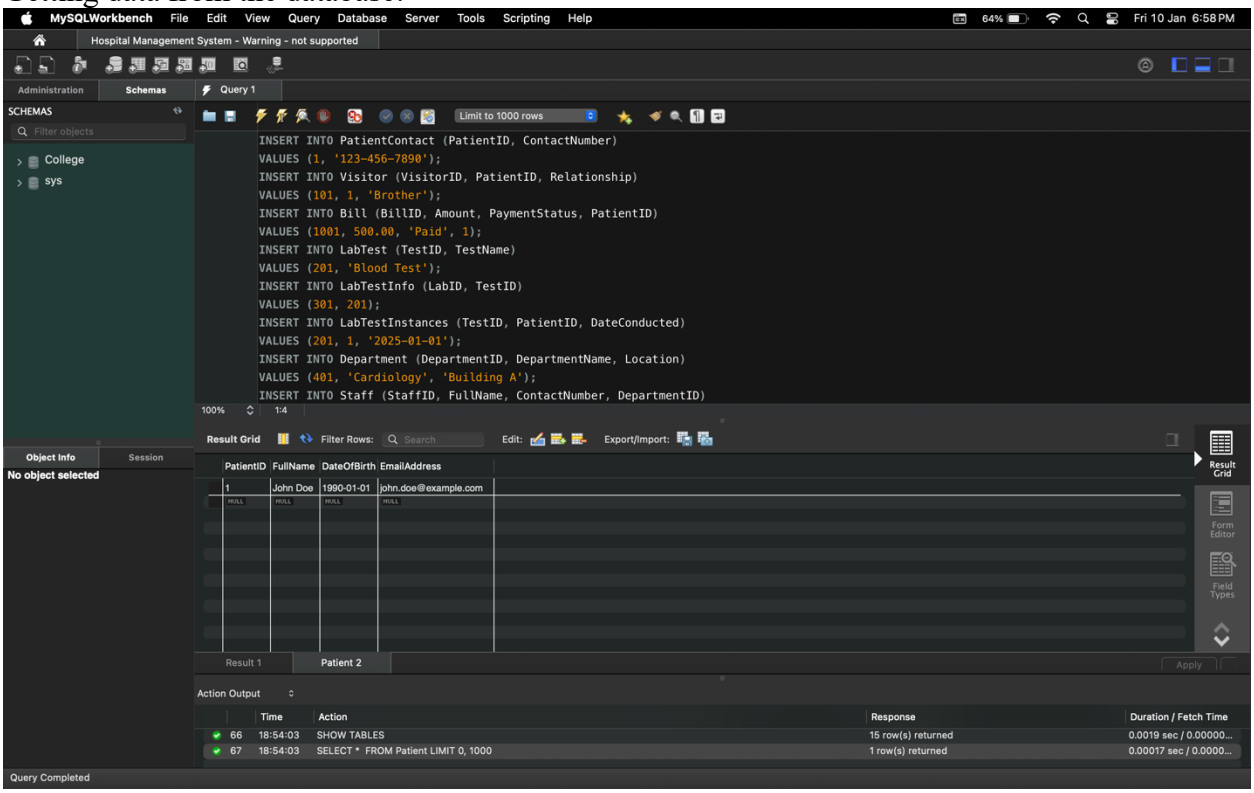

MySQLWorkbench File Edit View Query Database Server Tools Scripting Help
Fri 10 Jan 4:26 PM
Hospital Management System - Warning - not supported
Administration
Schemas
Query 1
SCHEMAS
Filter objects
College
Hospital
Tables
Views
Stored Procedures
Functions
sys
Limit to 1000 rows
1 CREATE DATABASE Hospital;
3 USE Hospital;
4 -- 1. Patient Table
5 CREATE TABLE Patient (
6 PatientID INT PRIMARY KEY,
7 FullName VARCHAR(100),
8 DateOfBirth DATE,
9 EmailAddress VARCHAR(100)
10);
11 -- 2. Patient Contact Table
12 CREATE TABLE PatientContact (
13 PatientID INT PRIMARY KEY,
14 ContactNumber VARCHAR(15),
15 FOREIGN KEY (PatientID) REFERENCES Patient(PatientID)
16);
17 -- 3. Visitor Table
18 CREATE TABLE Visitor (
19 VisitorID INT,
20 PatientID INT,
21 Relationship VARCHAR(50),
22 PRIMARY KEY (VisitorID, PatientID),
23 FOREIGN KEY (PatientID) REFERENCES Patient(PatientID)
24);
Object Info
Session
No object selected
100%
27:8
Action Output
Time
Action
Response
Duration / Fetch Time
17 16:25:31 CREATE TABLE Nurse (NurseID INT PRIMARY KEY, StaffID INT UNIQUE, FOREIGN KEY (StaffID) REFERENCES S... 0 row(s) affected 0.0027 sec
18 16:25:31 CREATE TABLE Appointment (AppointmentID INT PRIMARY KEY, DateTime DATETIME, Status VARCHAR(50),... 0 row(s) affected 0.0036 sec
Query interrupted

Show Tables:


MySQLWorkbench File Edit View Query Database Server Tools Scripting Help
Hospital Management System - Warning - not supported
Administration Schemas Query 1
SCHEMAS
Hospital
Tables
Appointment
Bill
Department
Doctor
LabTest
LabTestInfo
LabTestInstances
Medication
Nurse
Patient
PatientContact
Staff
Object Info Session
Schema: Hospital
147 INSERT INTO Treatment (TreatmentID, TreatmentDescription, DateOfTreatment, PatientID, DoctorID)
148 VALUES (901, 'Heart Surgery', '2025-01-10', 1, 601);
149 INSERT INTO Medication (MedicationID, Name, Dosage)
150 VALUES (1001, 'Aspirin', '100 mg');
151
152 SHOW TABLES;
153
154
155
Limit to 1000 rows
100% 13:152
Result Grid Filter Rows: Search Export:
Tables_in_hospi...
Appointment
Bill
Department
Doctor
LabTest
LabTestInfo
LabTestInstances
Medication
Nurse
Patient
PatientContact
Staff
Treatment
Visitor
Result 5
Read Only
Action Output
Time Action Response Duration / Fetch Time
101 19:00:11 SELECT * FROM Patient LIMIT 0, 1000 1 row(s) returned 0.00015 sec / 0.0000...
102 19:01:00 SHOW TABLES 14 row(s) returned 0.0019 sec / 0.00000...
Query Completed


Administration Schemas Query 1
SCHEMAS
Filter objects
Hospital
Tables
Appointment
Bill
Department
Doctor
LabTest
LabTestInfo
LabTestInstances
Medication
Nurse
Patient
PatientContact
Staff
Object Info Session
Schema: Hospital
Limit to 1000 rows
4
5
6 USE Hospital;
7 -- 1. Patient Table
8 CREATE TABLE Patient (
9 PatientID INT PRIMARY KEY,
10 FullName VARCHAR(100),
11 DateOfBirth DATE,
12 EmailAddress VARCHAR(100)
13);
14
15 -- 2. Patient Contact Table
16 CREATE TABLE PatientContact (
17 PatientID INT PRIMARY KEY,
18 ContactNumber VARCHAR(15),
19 FOREIGN KEY (PatientID) REFERENCES Patient(PatientID)
20);
21
22 -- 3. Visitor Table
23 CREATE TABLE Visitor (
24 VisitorID INT,
25 PatientID INT,
26 Relationship VARCHAR(50),
100% 1:3
Result Grid Filter Rows: Search Edit: Export/Import:
PatientID FullName DateOfBirth EmailAddress
1 John Doe 1990-01-01 john.doe@example.com
NULL NULL NULL NULL
Result 3 Patient 4

Getting data from the database:



As you can see all the table is created and our database connection was successful the validation is completed base on system's transaction requirement.