VIETNAM NATIONAL UNIVERSITY, HO CHI MINH CITY
UNIVERSITY OF TECHNOLOGY
FACULTY OF COMPUTER SCIENCE AND ENGINEERING

DATABASE SYSTEMS (CO2013)

Assignment 1

Hospital System

Advisor: Pham Nguyen Hoang Nam

Students: Nguyen Duc Thanh - 1952983
Nguyen Le Phu Hoa - 1852391
Huynh Huu Khang - 1852441

HO CHI MINH CITY, NOVEMBER 2021

Contents

1 Overview

The targets of the course of Database Systems is to introduce the history and motivations for database systems, architecture and components of a database system, conceptual and logical data models such as relational data model and entity relationship model. Besides, it is also to discuss relational algebra, SQL, and database design methodologies as well as data storage, management, and security issues with database management systems to develop database applications effectively for information systems.
Moreover, the assignment provided in the course will help us design and running our own database based on the specified requirements, using a DBMS (Database Management System). We will choose MySQL as the DBMS to apply for this assignment.

2 Requirement description

2.1 Initial requirements

Reviewing medical examination/treatment activities at the hospital

Patient comes for medical examination/treatment at a **hospital ward** and is recorded with **patient demographics and health insurance information**. When examined/ treatment at a time during a **ward shift**, the patient is treated by a **main doctor**. A doctor may be in charge of zero or more patients. In the same shift medical examination/treatment with a doctor, **a nurse/taking-care staff** is assigned to assist the doctor. If the patient is recorded as requiring hospitalization, the patient becomes an **inpatient** stay in the respective medical ward. Otherwise, record as **outpatient** and cure at home according to the **prescription** and **the diet** given by the doctor. For outpatient, **follow-up time** is noted. For inpatients, **the time for the next visit** is noted. When an inpatient is admitted to the hospital, the patient's inpatient **medical record** is prepared with additional admission information about **the number of beds, the number of rooms, the doctor appointed to be admitted to the hospital, the patient being admitted diagnosis at admission, admission status, length of admission**. When internal patients hospital discharge, **discharge information recorded** in the inpatient **medical record** includes: **intended discharge, disease diagnosed at discharge, discharge status, time of discharge hospital, discharge notes**. When discharged from the hospital, the doctor can **prescribe medicine** and **diet** hospital discharge for patients to continue treatment and support recovery after discharge.

During the hospital stay, the doctor provides **medical examination/treatment** to the patient and recommends issue orders. The nurse carries out the doctor's orders to make sure the patient **takes the medicine** and/or **perform tests/photographs** as ordered. Nursing performs care take care of the patient according to the **doctor's instructions on nutrition**. When examining / treating each patient (inpatient or outpatient), the doctor will **perform the diagnosis** was made with some of the symptoms noted. Besides, the doctor can **assign** the patient to perform zero or more relevant **tests**. Each test performed by another physician on duty with one or more test scores. Each thread the number of tests may have **thresholds** of normal values or may have ranges of normal values regularly. The obtained **test values** are compared with the cutoff values to be notes as **"normal" or "abnormal"**. In addition, the doctor can **appoint** the patient do not or much **X-ray** involved. Each film capture result is processed by another doctor. The **imaging diagnoses** are recorded by the doctor along with the scan results film and send it back to the patient.

Based on your **symptoms, lab results, and/or radiographs**, your doctor the main medical examination/treatment will make a conclusion about the patient's disease and treatment for the disease core. For the treatment part, the doctor clearly specifies each **drug, dose, method of administration, and the duration of drug use is appropriate for the body, age, sex, and allergy characteristics of each patient**. For the nutrition part, the doctor clearly specifies **the nutritional regimen** related if necessary.

The application program developed from this topic is intended to be used by:
(i). Hospital management board
(ii). Doctor
(iii). Patient
The hospital management and the doctor are both hospital employees who are requesting develop this application program. The hospital management board oversees the activities medical examination/treatment and personnel coordination. Within the scope of the application program application, the hospital management view all information related to doctors, patients, and medical examination/treatment. The doctor updates the details of each examination/ treatment activity diseases under my charge and view details of each patient in each examination/ healing. Patients update their personal details such as personal information demographics and health insurance as prescribed by the hospital and view details of each activity medical examination/treatment performed during treatment.

The data requirements of each user group are as follows:
(i). Hospital management board
(i.1). View the list of doctors on duty in a shift in a day in a department.
(i.2). View a list of doctors on duty for a day in a department.
(i.3). View the list of doctors on duty at one shift in a day in all departments.
(i.4). View a list of doctors on duty for a day in all departments.
(i.5). View the total number of patients in a shift in a day in a ward.
(i.6). View the total number of inpatients admitted to a hospital in a single day in a ward.
(i.7). View the total number of outpatients in a shift in a day in a ward.
(i.9). View the total number of inpatients admitted to the hospital in one case in a day in all department.
(i.10). View the total number of outpatients in a case in a day in all departments.
(i.11). View the total number of tests done in a day in a ward.
(i.12). View the total number of tests done in a day in all departments.
(ii). Doctor
(ii.1). Update medical examination/treatment activities for outpatients in a shift.
(ii.2). Update medical examination/treatment activities for inpatients during a shift.
(ii.3). View the list of patients in a day that I have been in charge of.
(ii.4). View the medical diagnoses of a patient you have been in charge of.
(ii.5). View medications taken over the days by an inpatient that I was in charge.
(ii.6). View day-to-day testing orders for an inpatient that I was in charge.
(ii.7). View the day-to-day radiography orders of an inpatient that I was in charge.
(ii.8). View patients of the same disease you have diagnosed.
(ii.9). See patients with an "abnormal" note in their test results a disease that I have been diagnosed with.
(ii.10). View the discharged patients that you have been in charge of.
(ii.11). See inpatients whose intake of a drug decreases over the course of 3 times in a row using drugs that I was in charge of.

(ii.12). See patients with a "normal" note of a test number in most recent test results but noted "abnormal" for the same test tested in the previous test results that I was in charge of.
(iii). Patient
(iii.1). Update demographic information and health insurance information.
(iii.2). View a list of the medications you took during your last treatment.
(iii.3). See a list of medicines that you have taken in all of your treatments.
(iii.4). View test results from the last treatment.
(iii.5). See a list of all the tests you've done.
(iii.6). See a list of all tests with the "abnormal" note of a test metric experience that I did.
(iii.7). View the list of doctors who have treated the last treatment.
(iii.8). View a list of doctors who have treated in all treatments.
(iii.9). See the nutrition regimen if indicated during the last treatment.
(iii.10). See list of diets if indicated in all treatments that I treat.

3 Conceptual Design

3.1 Entities and Relationships

3.1.1 Entities

There are 8 entities (2 are inherited from the others) which is defined in the table below:

Entity Name	Description
Patient	Store information of every patient who checkup or have cure in the hospital. Patient can be either Inpatient or Outpatient.
Inpatient	Inherit from Patient. Have relationship with Nurse and Takingcare Staff to get medical order and have a good take care.
Outpatient	Inherit from Patient.
Doctor	Have all information of every patient's symptoms and allow to give medical order to Nurse and Takingcare Staff. Also Doctor can be able to assign other doctors to have X-ray record and symptom testing results.
X-ray Doctor	Assigned by Doctor. X-ray Doctor has X-ray record of the patients.
Testing Doctor	Assigned by Doctor. Testing Doctor has all the symptom testing result of the patients.
Nurse	Assigned by Doctor. Nurse do medical order given by Doctor and ensure all the processes for cure the patient.
Takingcare Staff	Assigned by Doctor. Takingcare Staff takes care of patients while they are in process of curing.

Figure 1: Entities of the Hospital database.

3.1.2 Relationships

There are 9 relationships which is defined in the table below:

Relationship Name	Participants and Constraints
Diagnose	Doctor and Patient. 1:N relationship. (1 Doctor to N Patients, 1 Patient to 1 Doctor).
Assign	Doctor and Takingcare Staff 1:N relationship. (1 Takingcare Staff to N Doctors, 1 Doctor to 1 Takingcare Staff)
Take care of	Nurse and Inpatient 1:N relationship. (1 Takingcare Staff to N Inpatients, 1 Inpatient to 1 Takingcare Staff)
Give medical instructions	Doctor and Nurse 1:N Relationship. (1 Nurse to N Doctor, 1 Doctor to 1 Nurse)
Take care of	Nurse and Inpatient 1:N relationship. (1 Nurse to N Inpatients, 1 Inpatient to 1 Nurse)
Assign X-ray	Doctor and X-ray Doctor 1:1 Relationship.
Assign testing	Doctor and Testing Doctor 1:1 Relationship.
Take a X-ray	X-ray Doctor and Inpatient. M:N relationship.
Test	Testing Doctor and Inpatient. M:N relationship.

Figure 2: Relationships of the Hospital database.

3.1.3 Weak Entities and Identifying Relationships

There are no weak entity included in this assignment.

3.1.4 Identifying Relationships

There are no identifying relationship included in this assignment.

3.1.5 Attributes

Entity/Relationship Name	Attribute	Description (if any)
Patient	**Identity Number**	Primary Key of Patient
	Health Insurance Number	
Inpatient	**Next re-examination time**	
	Medical record	Multivalue attribute: Hospitalized diagnosed disease, Bed number, Room number, Hospitalized status, Hospitalized time
	Discharge information	Multivalue attribute: Nutrition, Diagnosed disease, Appointed doctor, Time, Status, Notes
	Prescription	Multivalue attribute: Medicine type, Medicine usage, Usage time, Dosage
Outpatient	**Nutrition**	
	Prescription	
	Re-examination time	
Doctor	**Doctor ID**	Primary Key of Doctor
	First Name	
	Last Name	
X-ray Doctor	**Doctor ID**	Primary Key of X-ray Doctor
	First Name	
	Last Name	
Testing Doctor	**Doctor ID**	Primary Key of Testing Doctor
	First Name	
	Last Name	
Nurse	**Nurse ID**	Primary Key of Nurse
	First Name	
	Last Name	
Takingcare Staff	**Nurse ID**	Primary Key of Takingcare Staff
	First Name	
	Last Name	
Diagnose	**Symptom**	
Take a X-ray	**X-ray result**	
Test	**Result**	
	Index	Enum value: 'Normal' and 'Abnormal'

Figure 3: Attributes of the Hospital database.

3.1.6 Entity-Relationship Model

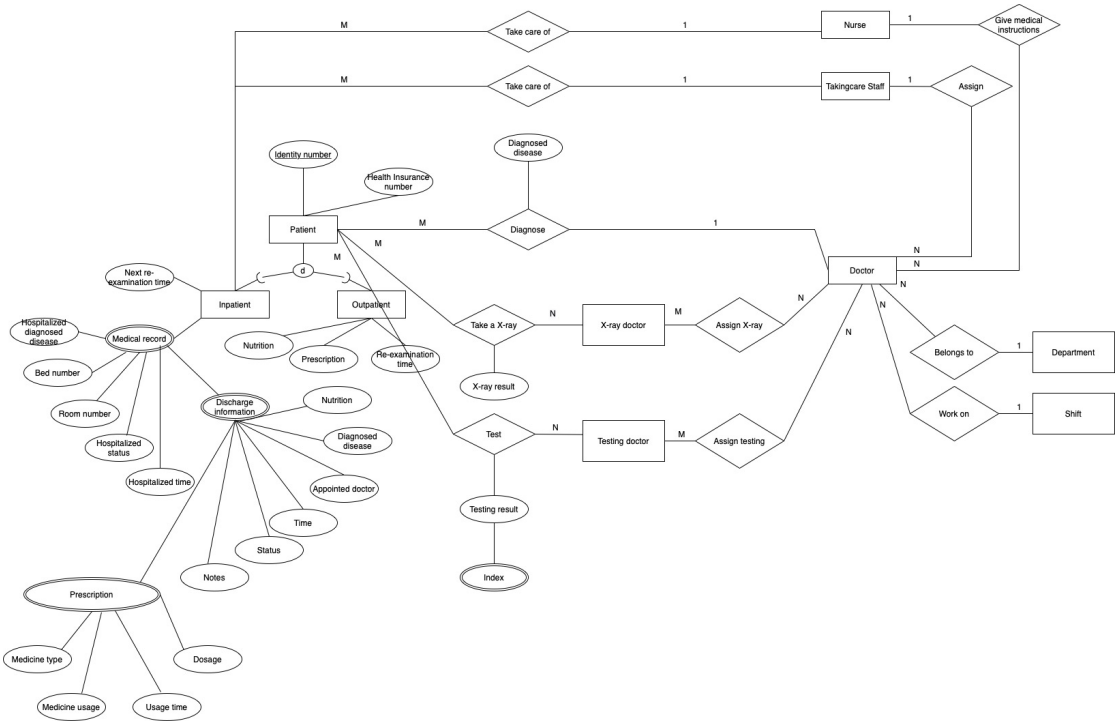

Figure 4: The ERD of the Hospital Database.

3.2 Logical Design

3.2.1 Mapping from Entity-Relationship Diagram to Relation Schema

The following modifications are made to map from ERD to Relation Schema (RS):
- These entities will become relations and keep all the entities that they have:
· Patient
· Doctor
· Testing doctor
· Nurse
· Takingcare staff
- Turn M:N relationships into relations. If these relationships has attribute, they will be the attribute in the relations:
· Take a X-ray
· Test
· Diagnose
- The 1:N relationships will become an attribute in a relation in the N side:
· Assign
· Take care of
· Give medical instructions
· Take care of

3.2.2 Result

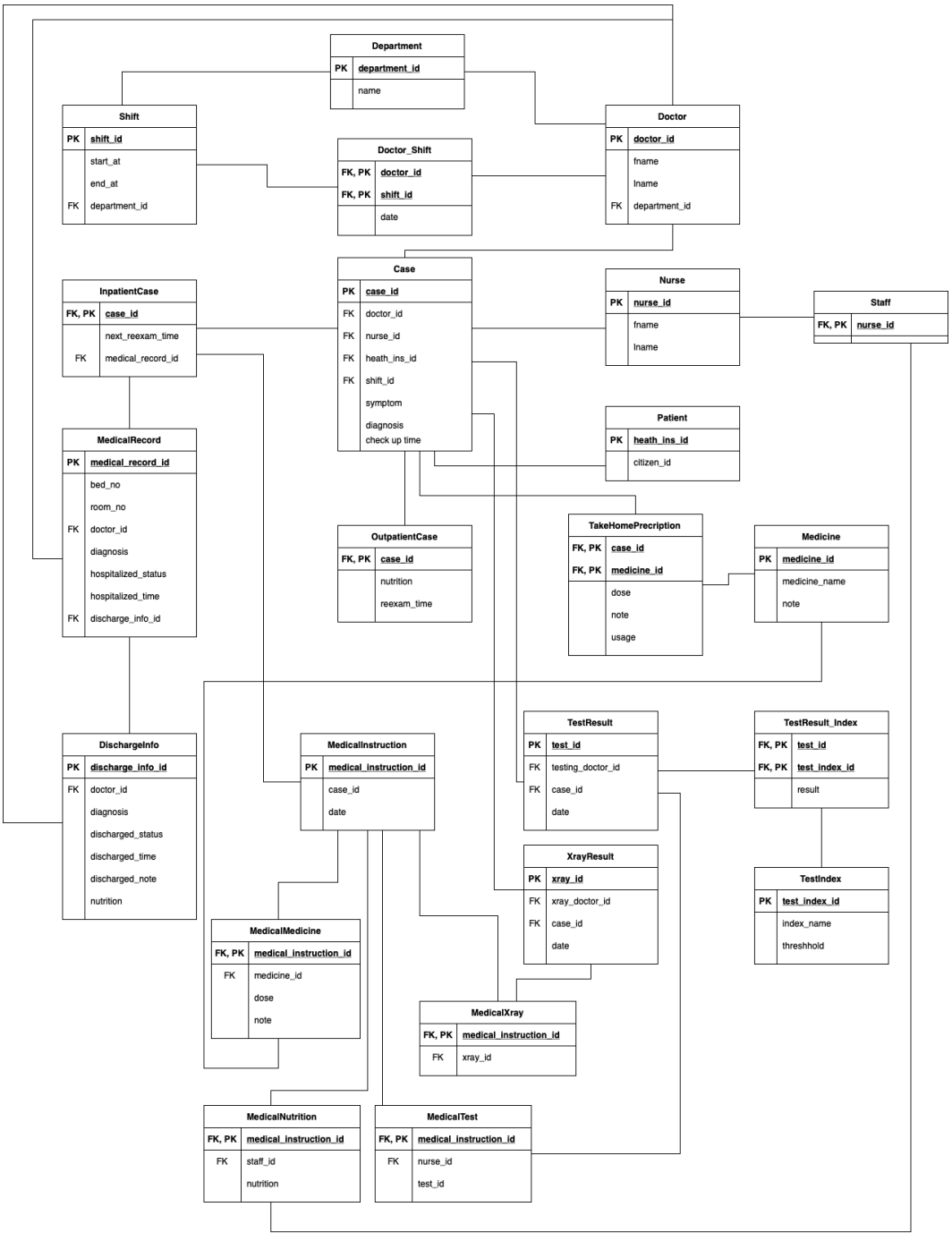

Figure 5: The Relation Schema of the Hospital Database

4 Tools for conceptual design

In this assignment, we will use MySQL Workbench as a tool to design our conceptual and databse.

4.1 What is MySQL Workbench?

MySQL Workbench is a Visual database designing and modeling access tool for MySQL server relational database. It facilitates creation of new physical data models and modification of existing MySQL databases with reverse/forward engineering and change management functions. The purpose of MySQL workbench is to provide the interface to work with databases more easily and in a more structured way.

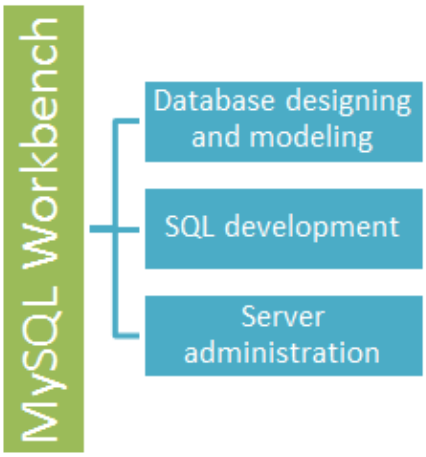

Figure 6: MySQL Workbench

4.2 Modeling and Design Tool

Models are at the core of most valid and high performance databases. MySQLworkbench has tools that allow developers and database administrators visually create physical database design models that can be easily translated into MySQL databases using forward engineering.
MySQL workbench supports creation of multiple models in the same environment. It supports all objects such as tables, views, stored procedures, triggers, etc. that make up a database.
MySQL workbench has a built in model validating utility that reports any issues that might be found to the data modeler. It also allows for different modeling notations and can be extended by using LUA a scripting language.

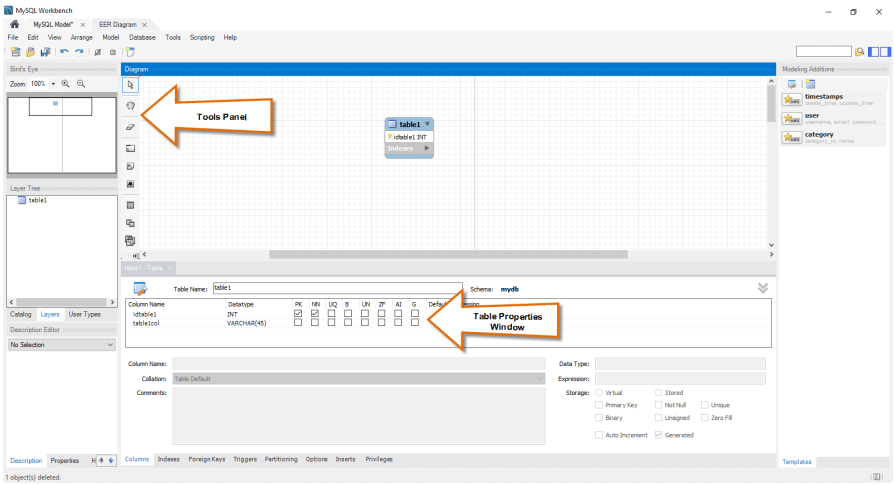

Figure 7: Modeling window for MySQL Workbench

4.3 SQL development tool

MySQLworkbench, has built in SQL visual editor.
The Visual SQL editor allows developers to build, edit and run queries against MySQL server databases. It has utilities for viewing data and exporting it.
Its syntax color highlighters help developers easily write and debug SQL statements.
Multiple queries can be run and results automatically displayed in different tabs. The queries are also saved in the history panel for later retrieval and running.

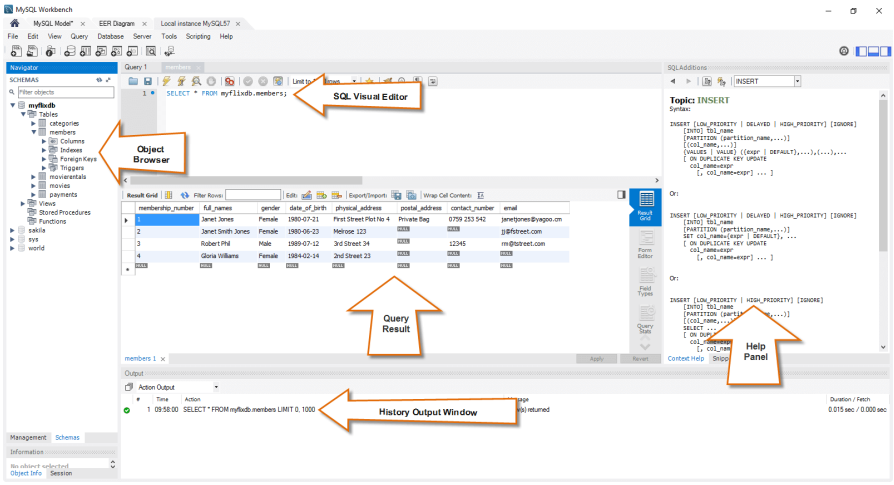

Figure 8: SQL development window for MySQL Workbench

5 Physical Design

5.1 Introduction to MySQL

MySQL is an open-source relational database management system (RDBMS). Its name is a combination of "My", the name of co-founder Michael Widenius's daughter, and "SQL", the abbreviation for Structured Query Language. A relational database organizes data into one or more data tables in which data types may be related to each other; these relations help structure the data. SQL is a language programmers use to create, modify and extract data from the relational database, as well as control user access to the database. In addition to relational databases and SQL, an RDBMS like MySQL works with an operating system to implement a relational database in a computer's storage system, manages users, allows for network access and facilitates testing database integrity and creation of backups.
MySQL is free and open-source software under the terms of the GNU General Public License, and is also available under a variety of proprietary licenses. MySQL was owned and sponsored by the Swedish company MySQL AB, which was bought by Sun Microsystems (now Oracle Corporation). In 2010, when Oracle acquired Sun, Widenius forked the open-source MySQL project to create MariaDB.
MySQL has stand-alone clients that allow users to interact directly with a MySQL database using SQL, but more often, MySQL is used with other programs to implement applications that need relational database capability. MySQL is a component of the LAMP web application software stack (and others), which is an acronym for Linux, Apache, MySQL, Perl/PHP/Python. MySQL is used by many database-driven web applications, including Drupal, Joomla, phpBB, and WordPress. MySQL is also used by many popular websites, including Facebook, Flickr, MediaWiki, Twitter, and YouTube.

5.2 SQL Commands for Data Requirements

The file Database_Creation.sql contains the source code of the database creation.

Create the database:

```
CREATE DATABASE hospital_system;
```

Access to the database:

```
USE hospital_system;
```

Next, we create the tables. Looking through the code to create the table, we will also find out the value type of each attribute.
Create the table patient:

```
CREATE TABLE Patient (
    health_ins_id BIGINT PRIMARY KEY,
    citizen_id BIGINT NOT NULL
);
```

Insert elements to the table patient:

```
  INSERT INTO patient VALUES (312482081,1223492347);
  INSERT INTO patient VALUES (312482321,4532492347);
  INSERT INTO patient VALUES (312482121,1398852347);
  INSERT INTO patient VALUES (314533081,9882492347);
```

Similar technique is apply to insertion of other tables.
Create the table doctor:

```
CREATE TABLE outpatients (
  id_card_no BIGINT PRIMARY KEY,
  reexam_time DATE,
  prescription VARCHAR(255) NOT NULL,
  nutrition VARCHAR(255),
  FOREIGN KEY (id_card_no) REFERENCES patients(id_card_no)
);
```

Create the table department:

```
CREATE TABLE Department (
    d_id BIGINT PRIMARY KEY,
    d_name VARCHAR(20) NOT NULL
);
```

Create the table shift:

```
CREATE TABLE Shift (
    shift_id INT PRIMARY KEY,
    start_at TIME NOT NULL,
    end_at TIME NOT NULL
);
```

Create the table doctor_shift:

```
CREATE TABLE Doctor_Shift (
    doctor_id BIGINT PRIMARY KEY,
    shift_id INT,
    work_date DATE,
    d_id BIGINT,
    FOREIGN KEY (doctor_id)
        REFERENCES Doctor (doctor_id),
    FOREIGN KEY (shift_id)
        REFERENCES Shift (shift_id),
    FOREIGN KEY (d_id)
        REFERENCES Department (d_id)
);
```

Create the table nurse:

```
CREATE TABLE Nurse (
    nurse_id BIGINT PRIMARY KEY,
    fname VARCHAR(20) NOT NULL,
    lname VARCHAR(20) NOT NULL
);
```

Create the table staff:

```
CREATE TABLE Staff (
    staff_id BIGINT PRIMARY KEY,
    FOREIGN KEY (staff_id)
        REFERENCES Nurse (nurse_id)
);
```

Create the table patient_case:

```
CREATE TABLE Patient_Case (
    case_id BIGINT PRIMARY KEY,
    doctor_id BIGINT,
    nurse_id BIGINT,
    health_ins_id BIGINT NOT NULL,
    checkup_time DATETIME,
    FOREIGN KEY (doctor_id)
        REFERENCES Doctor (doctor_id),
    FOREIGN KEY (nurse_id)
```

```
        REFERENCES Nurse (nurse_id),
    FOREIGN KEY (health_ins_id)
        REFERENCES Patient (health_ins_id),
    symptom VARCHAR(20) NOT NULL,
    diagnosis VARCHAR(20) NOT NULL
);
```

Create the table discharge_info:

```
CREATE TABLE Discharge_Info (
    discharge_info_id BIGINT PRIMARY KEY,
    doctor_id BIGINT,
    FOREIGN KEY (doctor_id)
        REFERENCES Doctor (doctor_id),
    diagnosis VARCHAR(20),
    discharged_status VARCHAR(20),
    discharged_time DATETIME NOT NULL,
    discharged_note VARCHAR(20),
    nutrition VARCHAR(20)
);
```

Create the table medical_record:

```
CREATE TABLE Medical_Record (
    medical_record_id BIGINT PRIMARY KEY,
    bed_no INT NOT NULL,
    room_no INT NOT NULL,
    doctor_id BIGINT,
    FOREIGN KEY (doctor_id)
        REFERENCES Doctor (doctor_id),
    diagnosis VARCHAR(20),
    hospitalized_status VARCHAR(20),
    hospitalized_time DATETIME NOT NULL,
    discharge_info_id BIGINT,
    FOREIGN KEY (discharge_info_id)
        REFERENCES Discharge_Info (discharge_info_id)
);
```

Create the table inpatient_case:

```
CREATE TABLE Inpatient_Case (
    case_id BIGINT PRIMARY KEY,
    FOREIGN KEY (case_id)
        REFERENCES Patient_Case (case_id),
    next_reexam_time DATE,
    medical_record_id BIGINT,
    FOREIGN KEY (medical_record_id)
        REFERENCES Medical_Record (medical_record_id)
);
```

Create the table outpatient_case:

```
CREATE TABLE Outpatient_Case (
    case_id BIGINT PRIMARY KEY,
    FOREIGN KEY (case_id)
        REFERENCES Patient_Case (case_id),
    nutrition VARCHAR(20),
    reexam_time DATETIME NOT NULL
);
```

Create the table medicine:

```
CREATE TABLE Medicine (
    medicine_id BIGINT PRIMARY KEY,
    medicine_name VARCHAR(20),
```

```
    note VARCHAR(20)
);
```

Create the table take_home_precription:

```
CREATE TABLE Take_Home_Precription (
    case_id BIGINT PRIMARY KEY,
    medicine_id BIGINT,
    FOREIGN KEY (case_id)
        REFERENCES Patient_Case (case_id),
    FOREIGN KEY (medicine_id)
        REFERENCES Medicine (medicine_id),
    dose INT,
    note VARCHAR(20),
    guide VARCHAR(20)
);
```

Create the table medical_instruction:

```
CREATE TABLE Medical_Instruction (
    medical_instruction_id BIGINT PRIMARY KEY,
    case_id BIGINT,
    FOREIGN KEY (case_id)
        REFERENCES Patient_Case (case_id),
  release_date DATETIME
);
```

Create the table medical_medicine:

```
CREATE TABLE Medical_Medicine (
    medical_instruction_id BIGINT PRIMARY KEY,
    medicine_id BIGINT,
    FOREIGN KEY (medical_instruction_id)
        REFERENCES Medical_Instruction (medical_instruction_id),
    FOREIGN KEY (medicine_id)
        REFERENCES Medicine (medicine_id),
    dose INT,
    note VARCHAR(20)
);
```

Create the table medical_nutrition:

```
CREATE TABLE Medical_Nutrition (
    medical_instruction_id BIGINT PRIMARY KEY,
    staff_id BIGINT,
    FOREIGN KEY (medical_instruction_id)
        REFERENCES Medical_Instruction (medical_instruction_id),
    FOREIGN KEY (staff_id)
        REFERENCES Staff (staff_id),
    nutrition VARCHAR(20)
);
```

Create the table test_result:

```
CREATE TABLE Test_Result (
    test_id BIGINT PRIMARY KEY,
    doctor_id BIGINT,
    case_id BIGINT,
    test_date DATETIME,
    FOREIGN KEY (doctor_id)
        REFERENCES Doctor (doctor_id),
    FOREIGN KEY (case_id)
        REFERENCES Patient_Case (case_id)
);
```

Create the table xray_result:

```
CREATE TABLE Xray_Result (
    xray_id BIGINT PRIMARY KEY,
    doctor_id BIGINT,
    case_id BIGINT,
    xray_date DATETIME,
    FOREIGN KEY (doctor_id)
        REFERENCES Doctor (doctor_id),
    FOREIGN KEY (case_id)
        REFERENCES Patient_Case (case_id)
);
```

Create the table test_index:

```
CREATE TABLE Test_Index (
    test_index_id BIGINT PRIMARY KEY,
    index_name VARCHAR(20),
    threshhold FLOAT
);
```

Create the table test_result_index:

```
CREATE TABLE Test_Result_Index (
    test_id BIGINT PRIMARY KEY,
    test_index_id BIGINT,
    FOREIGN KEY (test_id)
        REFERENCES Test_Result (test_id),
    FOREIGN KEY (test_index_id)
        REFERENCES Test_Index (test_index_id),
    result VARCHAR(20)
);
```

Create the table medical_xray:

```
CREATE TABLE Medical_Xray (
    medical_instruction_id BIGINT PRIMARY KEY,
    xray_id BIGINT,
    FOREIGN KEY (medical_instruction_id)
        REFERENCES Medical_Instruction (medical_instruction_id),
    FOREIGN KEY (xray_id)
        REFERENCES Xray_Result (xray_id)
);
```

Create the table medical_test:

```
CREATE TABLE Medical_Test (
    medical_instruction_id BIGINT PRIMARY KEY,
    nurse_id BIGINT,
    test_id BIGINT,
    FOREIGN KEY (medical_instruction_id)
        REFERENCES Medical_Instruction (medical_instruction_id),
    FOREIGN KEY (nurse_id)
        REFERENCES Nurse (nurse_id),
    FOREIGN KEY (test_id)
        REFERENCES Test_Result (test_id)
);
```