Project1 Design & Implement a Relational Database

- **Patient Management System**

Highlight nouns and verbs

1.Introduction

The **Patient Management System** is a critical **healthcare** application designed to assist patients in managing patient appointments, medical prescriptions, and the delivery of medications. This system is particularly valuable for streamlining the process of test scheduling, medicine tracking, and ensuring patients receive timely updates and treatments. By integrating patient data, medical test reports, and doctor prescriptions, the system enables efficient tracking of patient health progress and ensures proper medication adherence.

The system is designed to handle various use cases, such as appointment for tests for patients, sending prescribed medications to the patient's home address, and recording the medication cycle. The goal is to improve the overall healthcare management experience for both the patients and medical professionals involved.

2. Rules of the business

1.The **Patient Management APP** stores each user's most fundamental **personal information**. This data forms the base layer for managing patient records. The **Patient Management APP** has two primary functions: **Health Surveillance** and **Medicine Monitor**.

2. **The** **Medicine Monitor** **manages** the monitoring of prescribed **medications** for each patient, **recording** the **medication details**, **monitoring dates**, and **statuses** to ensure adherence to the treatment plan. And Multiple medications can be tracked under a **monitor Id(Patient Id)**, allowing the system to track various treatments and ensure compliance with the prescribed **dosages**.

3. **The Health Surveillance** record captures the patient's **symptoms** and **disease history**. Based on the **data** from **Health Surveillance**, **Health Surveillance** can make an appointment for one or more **Tests**.

4. **Tests** are conducted by **Labs**, and each test is associated with a **labId**. The **Lab Reports** are sent back to the system where doctors can **review** them to prescribe or adjust treatment. **Lab Reports** are **generated** based on the test results and contain important information such as **illnessName** and **memo** for the doctor's review.

5. **Doctors refer Prescriptions** to patients after **reviewing** the **Lab reports**. Each **Prescription** contains **information** such as the **date** and **dosage of Instructions**. The system **records** multiple **medications** per **prescription**, each with a unique **medicine**, **medicine name**, and **dosage frequency**.

6. **Delivery** ensures that prescribed medications are sent to the correct address for each patient, identified by a unique **delivery Id**. It tracks key details such as the **delivery date** and **address** to ensure timely delivery. The relationship between **Delivery** and **Medicine** is many-to-many, meaning each delivery can include multiple medications, and each medication can be part of multiple deliveries.

3. Assumption

1.PatientManagementAPP, MedicineMonitor, and HealthSurveillance

Relationship Type: Generalization because PatientManagementAPP is an application, while MedicineMonitor and HealthSurveillance are its two main functional modules.

2.Aggregation Relationship Between Lab and Test means that a lab is responsible for organizing and conducting multiple tests, but the tests are independent.

3. Aggregation Relationship Between Lab and LabReport means lab generates lab reports, which contain test results and analyses. LabReport exist independently within the system.

4. MedicineMonitor and Medicine: MedicineMonitor is responsible for recording the patient's medication, and one monitoring record can include multiple medicines.

5. HealthSurveillance and Test: A health surveillance may appointment multiple tests to evaluate the patient's health status.

6. Medicine and Delivery: There is a many-to-many relationship between medicine and delivery services, where one medicine can be delivered multiple times, and each delivery may include multiple medicines.

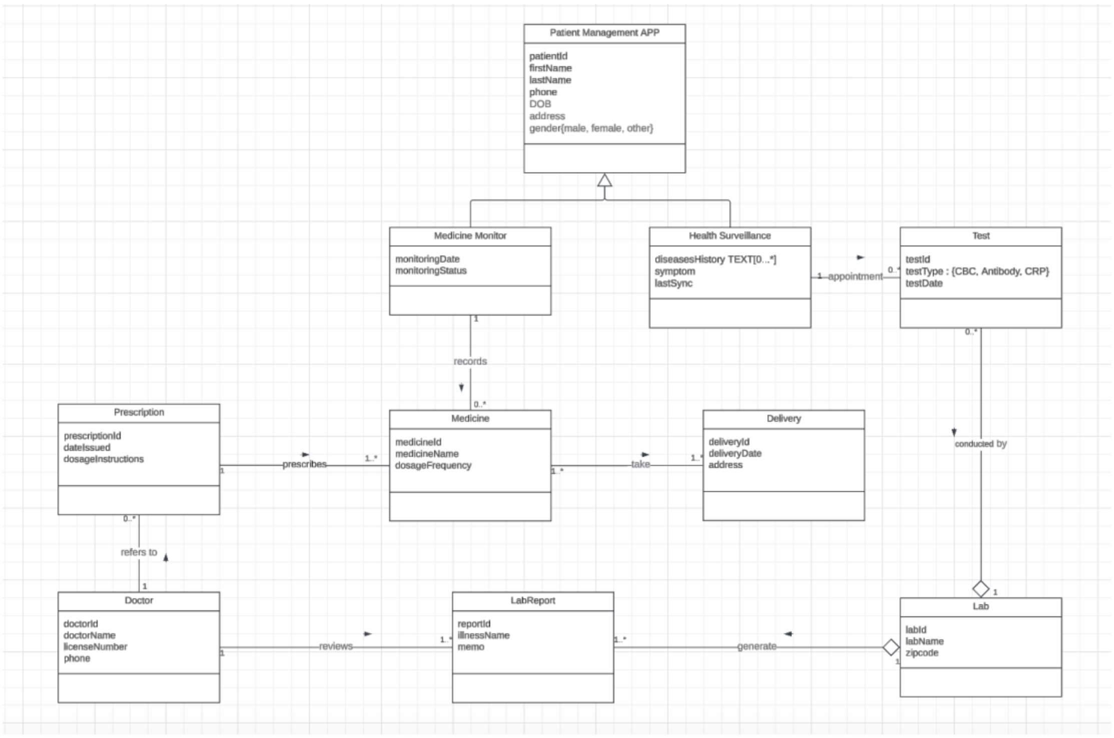
7.Prescription and Medicine: A prescription can include multiple medicines.

8.Prescription and Doctor: A doctor can issue multiple prescriptions.

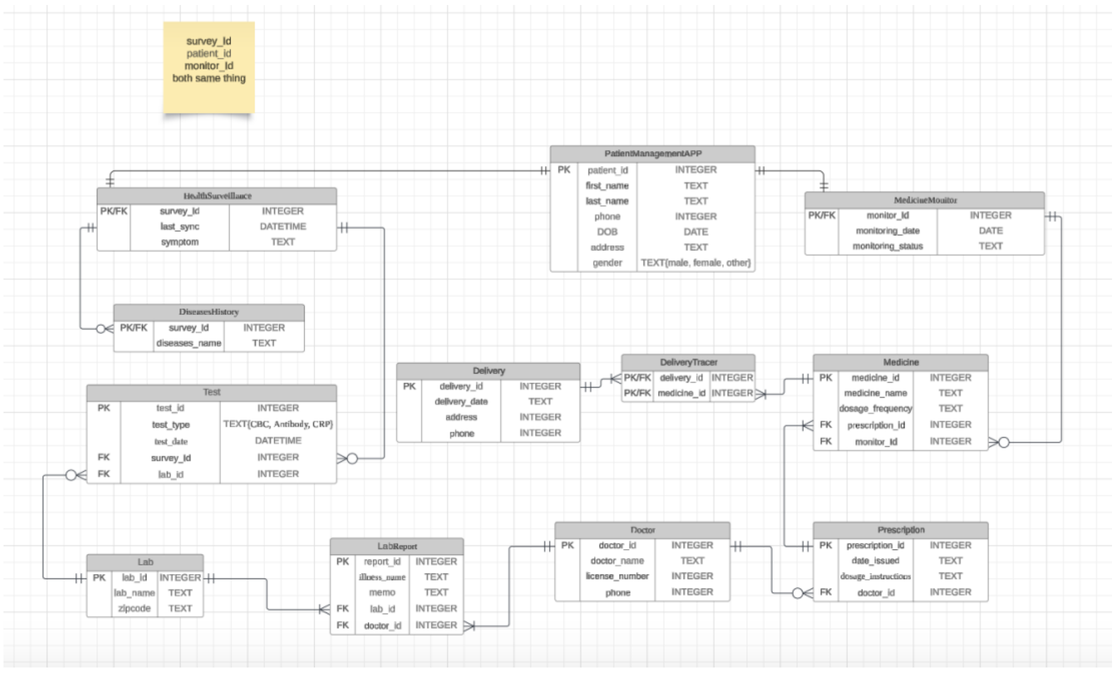
9. Doctor and LabReport: A doctor can view and evaluate multiple lab reports.

4.Conceptual model

LucidChart Tool



5. logical data model



6. Relational Schema Definitions

The logical schema shown above is resolved into the below relational schema. A relation R with attributes A1,A2,A3..An is shown as R(A1,A2,*A3*..An), where the primary key is underlined, and foreign keys are shown in *italics*.

PatientManagementAPP(patient_id, first_name, last_name, phone, DOB, address, gender)

HealthSurveillance(*survey_id*, last_sync, symptom)

DiseasesHistory(*survey_id*, diseases_name)

Test(test_id, test_type, test_date, *survey_id*, *lab_id*)

Lab(lab_id, lab_name, zipcode)

LabReport(report_id, illness_name, memo, *lab_id*, *doctor_id*)

Doctor(doctor_id, doctor_name, license_number, phone)

MedicineMonitor(*monitor_id*, monitoring_date, monitoring_status)

Medicine(medicine_id, medicine_name, dosage_frequency, *prescription_id*, *monitor_id)*

Prescription(prescription_id, date_issued, dosage_instructions, *doctor_id*)

Delivery(delivery_id, delivery_date, address, phone)

DeliveryTracer(*delivery_id*, *medicine_id*)

7. Proof using functional dependencies to show that schema is in BCNF

1.X → Y is a trivial functional dependency (i.e Y is a subset of X)

2.X is a super key for the schema

PatientManagementAPP

patient_id → {first_name, last_name, phone, DOB, address, gender}

HealthSurveillance

survey_id→{last_sync, symptom}

DiseasesHistory

survey_id→{diseases_name}

Test

test_id→{test_type, test_date, survey_id, lab_id}

Lab

lab_id→{lab_name, zipcode}

LabReport

report_id→{illness_name, memo, lab_id, doctor_id}

Doctor

doctor_id→ {doctor_name, license_number, phone}

MedicineMonitor

monitor_id→{monitoring_date, monitoring_status}

Medicine

medicine_id→ {medicine_name, dosage_frequency, prescription_id, monitor_id}

Prescription

prescription_id→{date_issued, dosage_instructions, doctor_id}

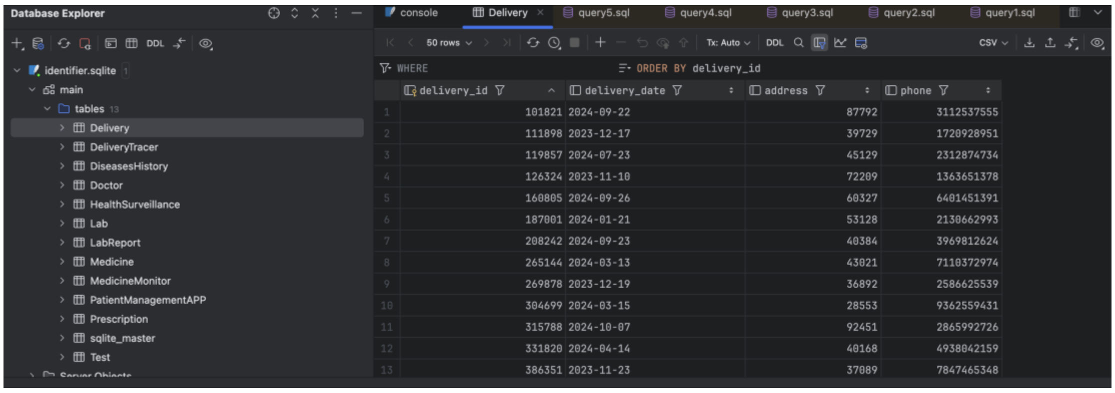
Delivery

delivery_id→ {delivery_date, address, phone}

DeliveryTracer

delivery_id, medicine_id form a composite primary key(trivial FD hence in BCNF)

8.Schema creation and Table definition in MySQL



```
WHERE    ORDER BY delivery_id
```

	delivery_id	delivery_date	address	phone
1	101821	2024-09-22	87792	3112537555
2	111898	2023-12-17	39729	1720928951
3	119857	2024-07-23	45129	2312874734
4	126324	2023-11-10	72209	1363651378
5	160805	2024-09-26	60327	6401451391
6	187001	2024-01-21	53128	2130662993
7	208242	2024-09-23	40384	3969812624
8	265144	2024-03-13	43021	7110372974
9	269878	2023-12-19	36892	2586625539
10	304699	2024-03-15	28553	9362559431
11	315788	2024-10-07	92451	2865992726
12	331820	2024-04-14	40168	4938042159
13	386351	2023-11-23	37089	7847465348

```
CREATE TABLE PatientManagementAPP (
    patient_id INTEGER PRIMARY KEY,
    first_name TEXT,
    last_name TEXT,
    phone INTEGER,
    DOB DATE,
    address TEXT,
    gender TEXT CHECK(gender IN ('male', 'female', 'other'))
);
CREATE TABLE HealthSurveillance (
    survey_id INTEGER PRIMARY KEY,
    last_sync DATETIME,
    symptom TEXT,
    FOREIGN KEY (survey_id) REFERENCES PatientManagementAPP(patient_id)
);
CREATE TABLE DiseasesHistory (
    survey_id INTEGER,
    diseases_name TEXT,
    PRIMARY KEY (survey_id),
    FOREIGN KEY (survey_id) REFERENCES HealthSurveillance(survey_id)
);
CREATE TABLE Test (
    test_id INTEGER PRIMARY KEY,
    test_type TEXT CHECK(test_type IN ('CBC', 'Antibody', 'CRP')),
    test_date DATETIME,
    survey_id INTEGER,
    lab_id INTEGER,
    FOREIGN KEY (survey_id) REFERENCES HealthSurveillance(survey_id),
    FOREIGN KEY (lab_id) REFERENCES Lab(lab_id)
);
CREATE TABLE Lab (
    lab_id INTEGER PRIMARY KEY,
    lab_name TEXT,
    zipcode TEXT
);
CREATE TABLE LabReport (
    report_id INTEGER PRIMARY KEY,
    illness_name TEXT,
    memo TEXT,
    lab_id INTEGER,
    doctor_id INTEGER,
    FOREIGN KEY (lab_id) REFERENCES Lab(lab_id),
    FOREIGN KEY (doctor_id) REFERENCES Doctor(doctor_id)
);
CREATE TABLE Doctor (
    doctor_id INTEGER PRIMARY KEY,
    doctor_name TEXT,
    license_number INTEGER,
    phone INTEGER
);
CREATE TABLE Prescription (
    prescription_id INTEGER PRIMARY KEY,
    date_issued TEXT,
    dosage_instructions TEXT,
    doctor_id INTEGER,
    FOREIGN KEY (doctor_id) REFERENCES Doctor(doctor_id)
);
CREATE TABLE Medicine (
    medicine_id INTEGER PRIMARY KEY,
    medicine_name TEXT,
    dosage_frequency TEXT,
    prescription_id INTEGER,
    monitor_id INTEGER,
    FOREIGN KEY (prescription_id) REFERENCES Prescription(prescription_id),
    FOREIGN KEY (monitor_id) REFERENCES MedicineMonitor(monitor_id)
);
```

```
CREATE TABLE MedicineMonitor (
    monitor_id INTEGER PRIMARY KEY,
    monitoring_date DATE,
    monitoring_status TEXT,
    FOREIGN KEY (monitor_id) REFERENCES PatientManagementAPP(patient_id)
);
CREATE TABLE Delivery (
    delivery_id INTEGER PRIMARY KEY,
    delivery_date TEXT,
    address INTEGER,
    phone INTEGER
);
CREATE TABLE DeliveryTracer (
    delivery_id INTEGER,
    medicine_id INTEGER,
    PRIMARY KEY (delivery_id, medicine_id),
    FOREIGN KEY (delivery_id) REFERENCES Delivery(delivery_id),
    FOREIGN KEY (medicine_id) REFERENCES Medicine(medicine_id)
);
```