University
Mohammed VI
Polytechnic

College of Computing

Normalization and SQL implementation

Data Management Course

UM6P College of Computing

Professor: Karima Echihabi **Program:** Computer Engineering
Session: Fall 2025

Team Information

Team Name	QueryMaster
Member 1	El Mehdi Regagui
Member 2	Yasser Jarboua
Member 3	Adam Ibourg-EL Idrissi
Member 4	Salma Mana
Member 5	Hiba Mhirit
Member 6	Sara Qiouame
Member 7	Douaae Mabrouk
Repository Link	`https://github.com/yasserJarboua/QueryMasters/`

1 Introduction

The Moroccan National Health Services (MNHS) requires a robust database to manage patients, staff, hospitals, departments, appointments, prescriptions, medications, insurance, billing, and emergencies. This deliverable will produce a fully normalized and implemented database solution for the MNHS by first rigorously validating the relational schema against Boyce-Codd Normal Form (BCNF) to ensure data integrity, and then practically realizing it through SQL Data Definition Language (DDL). The database will be populated and managed using Data Manipulation Language (DML), and its operational utility will be demonstrated through a comprehensive set of SQL queries, ranging from simple data retrieval to complex grouped aggregations for analytical reporting.

2 Requirements

This deliverable addresses the following tasks:

- Validate each relation against BCNF and verify lossless join and dependency preservation after decomposition.
- Implement the refined MNHS schema using SQL Data Definition Language (DDL).
- Populate and manage MNHS data using Data Manipulation Language (DML).
- Write and execute SQL queries ranging from simple selections to grouped aggregations and analytics.

3 Methodology

In this lab, we worked on the normalization of all relations in the MNHS project. We reviewed the main concepts related to functional dependencies, normal forms, and the need for decomposition, as well as the essential properties that must be verified, such as dependency preservation and lossless join.

The second part focused on SQL implementation. We worked with DDL (Data Definition Language), which refers to SQL commands used to define and modify the structure of the database, including the creation of tables, the alteration of columns, and the specification of constraints. We also worked with DML (Data Manipulation Language), which consists of commands used to manage and manipulate the data stored in the database, such as inserting, updating, deleting, and querying records. Finally, we applied our SQL knowledge by implementing various queries ranging from simple selections to grouped aggregations and analytical operations.

4 Implementation & Results

4.1 The final refined MNHS relational schema

1. Patient

Primary key: IID

Attributes: CIN, Name, Sex, Birth, BloodGroup, Phone

Functional Dependencies (FDs):

- IID → Name, Sex, Birth, BloodGroup, Phone, CIN
- CIN → IID, Name, Sex, Birth, BloodGroup, Phone

Explanation: CIN is a candidate key since it uniquely identifies each patient. IID is the primary key and determines all attributes.

BCNF Verification: Both determinants (IID, CIN) are superkeys. The relation is in BCNF.

Lossless Joins and Dependency Preservation: No decomposition. Trivially lossless and dependency preserving.

2. ContactLocation

Primary key: CLID

Attributes: Street, City, Province, PostalCode, Phone

FDs:

- CLID → Street, City, Province, PostalCode, Phone

BCNF Verification: CLID is a superkey. The relation is in BCNF.

Lossless & DP: Trivially satisfied.

3. Staff

Primary key: STAFF_ID

Attributes: Name, Status

FDs:

- STAFF_ID → Name, Status

ISA Subtype FDs:

- STAFF_ID → LicenseNumber, Specialty (Practitioner)
- STAFF_ID → Grade, Ward (Caregiving)
- STAFF_ID → Modality, Certifications (Technical)

BCNF Verification: STAFF_ID is the only determinant and is a superkey.

Lossless & DP: Trivially satisfied.

4. Hospital

Primary key: HID

Attributes: Name, City, Region

FDs:

- HID → Name, City, Region

BCNF Verification: HID is a superkey. Hospital is in BCNF.

Lossless & DP: Trivially satisfied.

5. Department

Primary key: DEP_ID
Attributes: Name, Specialty, HID
Inherited Dependencies:

- DEP_ID → Name, Specialty, HID
- HID → HospitalName, City, Region
- DEP_ID → Name, Specialty, HID, HospitalName, City, Region

BCNF Verification: Determinants (DEP_ID, HID) are superkeys in their respective tables.
Lossless & DP: Trivially satisfied.

7. ClinicalActivity

Primary key: CAID
Attributes: Title, Time, Date, IID, STAFF_ID, DEP_ID, ExpID
FDs inside ClinicalActivity:

- CAID → IID, STAFF_ID, DEP_ID, Date, Time

Foreign Key FDs (from referenced tables):

- IID → CIN, FullName, Birth, Sex, BloodGroup, Phone
- STAFF_ID → FullName, Status
- DEP_ID → Name, Specialty, HID

By composition:

$$CAID \rightarrow allPatient, Staff, Department, Hospitalattributes$$

BCNF Verification: Only CAID acts as a determinant; it is a superkey.
Lossless & DP: Joins are lossless due to foreign keys; dependencies preserved.

8. Expense

Primary key: ExID
Attributes: Total, InsID, CAID
FDs:

- ExID → Total, InsID, CAID
- CAID → ExID (one-to-one relationship)

BCNF Verification: Both ExID and CAID are superkeys.
Lossless & DP: Trivially satisfied.

9. Prescription

Primary key: PID
Attributes: DateIssued, CAID
FDs:

- PID → DateIssued, CAID
- CAID → Time, Date, IID, STAFF_ID, DEP_ID
- PID → DateIssued, Time, Date, IID, STAFF_ID, DEP_ID

BCNF Verification: PID is the only true determinant and is a superkey.
Lossless & DP: Trivially satisfied.

10. Appointment

Primary key: CAID
Attributes: Reason, Status, Time, Date, IID, STAFF_ID, DEP_ID
FDs:

- CAID → Reason, Status, Time, Date, IID, STAFF_ID, DEP_ID

BCNF Verification: CAID is a superkey.
Lossless & DP: Trivially satisfied.

11. Emergency

Primary key: CAID
Attributes: TriageLevel, Outcome, Time, Date, IID, STAFF_ID, DEP_ID
FDs:

- CAID → TriageLevel, Outcome, Time, Date, IID, STAFF_ID, DEP_ID

BCNF Verification: CAID is a superkey.
Lossless & DP: Trivially satisfied.

12. Medication

Primary key: DrugID
Attributes: Name, Form, Strength, Class, ActiveIngredient, Manufacturer
FDs:

- DrugID → Name, Form, Strength, Class, ActiveIngredient, Manufacturer

BCNF Verification: DrugID is a superkey.
Lossless & DP: Trivially satisfied.

13. Stock

Primary key: {HID, MID, StockTimestamp}
Attributes: UnitPrice, Qty, ReorderLevel
FDs:

- {HID, MID, StockTimestamp} → UnitPrice, Qty, ReorderLevel

BCNF Verification: Composite key is a superkey.
Lossless & DP: Trivially satisfied.

15. Patient–ContactLocation

Primary key: {IID, CLID}
FDs:

- {IID, CLID} → relationship attributes

BCNF Verification: Composite key is a superkey.

16. Patient–Insurance

Primary key: {IID, InsID}
FDs:

- {IID, InsID} → relationship attributes
- {IID, InsID} → Patient attributes, Insurance attributes

BCNF Verification: Determinant is a superkey.

17. Staff–Department

Primary key: {STAFF_ID, DEP_ID}
FDs:

- {STAFF_ID, DEP_ID} → relationship attributes

BCNF Verification: Composite key is a superkey.

5 SQL Implementation

```
CREATE DATABASE IF NOT EXISTS MNHS;
USE MNHS;

CREATE TABLE Patient (
    IID INT PRIMARY KEY,
    CIN VARCHAR(10) UNIQUE NOT NULL,
    FullName VARCHAR(100) NOT NULL,
    Birth DATE,
    Sex ENUM('M', 'F') NOT NULL,
    BloodGroup ENUM('A+', 'A-', 'B+', 'B-', 'O+', 'O-', 'AB+', 'AB-'),
    Phone VARCHAR(15)
);

CREATE TABLE Hospital (
    HID INT PRIMARY KEY,
    Name VARCHAR(100) NOT NULL,
    City VARCHAR(50) NOT NULL,
    Region VARCHAR(50)
);
```

```
CREATE TABLE Department (
    DEP_ID INT PRIMARY KEY,
    HID INT not null,
    Name VARCHAR(100) NOT NULL,
    Specialty VARCHAR(100),
    FOREIGN KEY (HID) REFERENCES Hospital(HID)
);

CREATE TABLE Staff (
    STAFF_ID INT PRIMARY KEY,
    FullName VARCHAR(100) NOT NULL,
    Status ENUM('Active', 'Retired') DEFAULT 'Active'
);

CREATE TABLE Practitioner (
    STAFF_ID INT PRIMARY KEY,
    LicenseNumber VARCHAR(50),
    Specialty VARCHAR(100),
    FOREIGN KEY (STAFF_ID) REFERENCES Staff(STAFF_ID)
);

CREATE TABLE Caregiving (
    STAFF_ID INT PRIMARY KEY,
    Grade VARCHAR(50),
    Ward VARCHAR(50),
    FOREIGN KEY (STAFF_ID) REFERENCES Staff(STAFF_ID)
);

CREATE TABLE Technical (
    STAFF_ID INT PRIMARY KEY,
    Certifications VARCHAR(200),
    Modality VARCHAR(100),
    FOREIGN KEY (STAFF_ID) REFERENCES Staff(STAFF_ID)
);

CREATE TABLE Work_in (
    STAFF_ID INT,
    Dep_ID INT,
    PRIMARY KEY (STAFF_ID, Dep_ID),
    FOREIGN KEY (STAFF_ID) REFERENCES Staff(STAFF_ID),
    FOREIGN KEY (Dep_ID) REFERENCES Department(DEP_ID)
);

CREATE TABLE ClinicalActivity (
    CAID INT PRIMARY KEY,
    IID INT NOT NULL,
    STAFF_ID INT NOT NULL,
    DEP_ID INT NOT NULL,
```

```
    Date DATE NOT NULL,
    Time TIME,
    FOREIGN KEY (IID) REFERENCES Patient(IID),
    FOREIGN KEY (STAFF_ID) REFERENCES Staff(STAFF_ID),
    FOREIGN KEY (DEP_ID) REFERENCES Department(DEP_ID)
);

CREATE TABLE Appointment (
    CAID INT PRIMARY KEY,
    Reason VARCHAR(100),
    Status ENUM('Scheduled', 'Completed', 'Cancelled') DEFAULT 'Scheduled',
    FOREIGN KEY (CAID) REFERENCES ClinicalActivity(CAID)
);

CREATE TABLE Emergency (
    CAID INT PRIMARY KEY,
    TriageLevel INT CHECK (TriageLevel BETWEEN 1 AND 5),
    Outcome ENUM('Discharged', 'Admitted', 'Transferred', 'Deceased'),
    FOREIGN KEY (CAID) REFERENCES ClinicalActivity(CAID)
);

CREATE TABLE Insurance (
    InsID INT PRIMARY KEY,
    Type ENUM('CNOPS', 'CNSS', 'RAMED', 'Private', 'None') NOT NULL
);

CREATE TABLE Covers (
    InsID INT,
    IID INT,
    PRIMARY KEY (InsID, IID),
    FOREIGN KEY (InsID) REFERENCES Insurance(InsID),
    FOREIGN KEY (IID) REFERENCES Patient(IID)
);

CREATE TABLE Expense (
    ExpID INT PRIMARY KEY,
    InsID INT not null,
    CAID INT UNIQUE NOT NULL,
    Total DECIMAL(10,2) NOT NULL CHECK (Total >= 0),
    FOREIGN KEY (InsID) REFERENCES Insurance(InsID),
    FOREIGN KEY (CAID) REFERENCES ClinicalActivity(CAID)
);

CREATE TABLE Medication (
    MID INT PRIMARY KEY,
    Name VARCHAR(100) NOT NULL,
    Form VARCHAR(50),
    Strength VARCHAR(50),
```

```
    ActiveIngredient VARCHAR(100),
    TherapeuticClass VARCHAR(100),
    Manufacturer VARCHAR(100)
);

CREATE TABLE Stock (
    HID INT,
    MID INT,
    StockTimestamp DATETIME DEFAULT CURRENT_TIMESTAMP,
    UnitPrice DECIMAL(10,2) CHECK (UnitPrice >= 0),
    Qty INT DEFAULT 0 CHECK (Qty >= 0),
    ReorderLevel INT DEFAULT 10 CHECK (ReorderLevel >= 0),
    PRIMARY KEY (HID, MID, StockTimestamp),
    FOREIGN KEY (HID) REFERENCES Hospital(HID),
    FOREIGN KEY (MID) REFERENCES Medication(MID)
);

CREATE TABLE Prescription (
    PID INT PRIMARY KEY,
    CAID INT UNIQUE NOT NULL,
    DateIssued DATE NOT NULL,
    FOREIGN KEY (CAID) REFERENCES ClinicalActivity(CAID)
);

CREATE TABLE Includes (
    PID INT,
    MID INT,
    Dosage VARCHAR(100),
    Duration VARCHAR(50),
    PRIMARY KEY (PID, MID),
    FOREIGN KEY (PID) REFERENCES Prescription(PID),
    FOREIGN KEY (MID) REFERENCES Medication(MID)
);

CREATE TABLE ContactLocation (
    CLID INT PRIMARY KEY,
    City VARCHAR(50),
    Province VARCHAR(50),
    Street VARCHAR(100),
    Number VARCHAR(10),
    PostalCode VARCHAR(10),
    Phone VARCHAR(15)
);

CREATE TABLE Have (
    IID INT,
    CLID INT,
    PRIMARY KEY (IID, CLID),
```

```
    FOREIGN KEY (IID) REFERENCES Patient(IID),
    FOREIGN KEY (CLID) REFERENCES ContactLocation(CLID)
);
INSERT INTO Patient (IID, CIN, FullName, Birth, Sex, BloodGroup, Phone) VALUES
(1, 'AB234567', 'Amina Benjelloun', '1985-03-15', 'F', 'O+', '0661234567'),
(2, 'CD789012', 'Youssef El Alami', '1978-11-22', 'M', 'A+', '0677890123'),
(3, 'EF345678', 'Fatima Idrissi', '1992-07-08', 'F', 'B+', '0612345678'),
(4, 'GH901234', 'Mohammed Tazi', '1965-01-30', 'M', 'AB+', '0698765432'),
(5, 'IJ567890', 'Khadija Bennani', '1988-09-12', 'F', 'O-', '0623456789');

INSERT INTO Hospital (HID, Name, City, Region) VALUES
(1, 'CHU Ibn Rochd', 'Casablanca', 'Casablanca-Settat'),
(2, 'Hôpital Cheikh Khalifa', 'Rabat', 'Rabat-Salé-Kénitra'),
(3, 'CHU Hassan II', 'Fès', 'Fès-Meknès'),
(4, 'Hôpital Avicenne', 'Marrakech', 'Marrakech-Safi'),
(5, 'CHU Mohammed VI', 'Tanger', 'Tanger-Tétouan-Al Hoceïma');

INSERT INTO Department (DEP_ID, HID, Name, Specialty) VALUES
(1, 1, 'Cardiologie', 'Maladies cardiovasculaires'),
(2, 1, 'Urgences', 'Médecine d\'urgence'),
(3, 2, 'Pédiatrie', 'Soins des enfants'),
(4, 3, 'Radiologie', 'Imagerie médicale'),
(5, 4, 'Chirurgie générale', 'Interventions chirurgicales');

INSERT INTO Staff (STAFF_ID, FullName, Status) VALUES
(1, 'Dr. Rachid Hamouda', 'Active'),
(2, 'Dr. Samira Chakir', 'Active'),
(3, 'Dr. Mehdi Alaoui', 'Active'),
(4, 'Dr. Hassan Zemmouri', 'Retired'),
(5, 'Dr. Leila Fassi', 'Active'),
(6, 'Nadia Lahlou', 'Active'),
(7, 'Laila Ouazzani', 'Active'),
(8, 'Zineb Yassin', 'Active'),
(9, 'Samira Benali', 'Active'),
(10, 'Aicha Bentayeb', 'Active'),
(11, 'Karim Bouzid', 'Active'),
(12, 'Houda Fassi', 'Active'),
(13, 'Omar Regragui', 'Active'),
(14, 'Salma Amor', 'Active'),
(15, 'Jamal Din Dibiaza', 'Active');

INSERT INTO Practitioner (STAFF_ID, LicenseNumber, Specialty) VALUES
(1, 'MED-2008-4521', 'Cardiologie'),
(2, 'MED-2012-7834', 'Pédiatrie'),
(3, 'MED-2015-9201', 'Radiologie'),
(4, 'MED-1995-1203', 'Chirurgie générale'),
(5, 'MED-2010-5678', 'Médecine d\'urgence');
```

```
INSERT INTO Caregiving (STAFF_ID, Grade, Ward) VALUES
(6, 'Aide-soignante', 'Urgences'),
(7, 'Infirmière diplômée', 'Cardiologie B'),
(8, 'Infirmière auxiliaire', 'Pédiatrie'),
(9, 'Infirmière chef', 'Chirurgie'),
(10, 'Aide-soignante', 'Radiologie');

INSERT INTO Technical (STAFF_ID, Certifications, Modality) VALUES
(11, 'Échographie cardiaque, ECG', 'Cardiologie'),
(12, 'Radiologie pédiatrique', 'Imagerie'),
(13, 'Manipulation d\'équipements', 'Laboratoire'),
(14, 'IRM, Scanner', 'Radiologie'),
(15, 'Analyses biologiques', 'Laboratoire');

INSERT INTO Work_in (STAFF_ID, Dep_ID) VALUES
(1, 1),
(2, 3),
(3, 4),
(4, 5),
(5, 2);

INSERT INTO ClinicalActivity (CAID, IID, STAFF_ID, DEP_ID, Date, Time) VALUES
(1, 1, 1, 1, '2025-10-18', '09:30:00'),
(2, 2, 2, 3, '2025-10-20', '14:00:00'),
(3, 3, 1, 1, '2025-11-12', '11:15:00'),
(4, 4, 4, 5, '2025-11-15', '10:30:00'),
(5, 1, 2, 3, '2025-12-05', '10:00:00'),
(6, 1, 5, 2, '2024-11-10', '22:15:00'),
(7, 3, 5, 2, '2024-11-05', '01:30:00'),
(8, 4, 5, 2, '2024-11-08', '18:45:00'),
(9, 2, 5, 2, '2024-10-30', '05:20:00'),
(10, 1, 5, 2, '2024-11-14', '12:00:00'),
(11, 2, 5, 2, '2024-11-01', '08:00:00'),
(12, 3, 5, 2, '2023-11-02', '09:00:00'),
(13, 4, 5, 2, '2024-11-03', '10:00:00'),
(14, 5, 5, 2, '2024-11-04', '11:00:00'),
(15, 1, 5, 2, '2025-11-06', '13:00:00'),
(16, 2, 5, 2, '2025-11-07', '14:00:00'),
(17, 3, 5, 2, '2024-11-09', '15:00:00'),
(18, 4, 5, 2, '2024-11-11', '16:00:00'),
(19, 5, 5, 2, '2024-11-12', '17:00:00'),
(20, 1, 5, 2, '2024-11-13', '18:00:00'),
(21, 2, 5, 2, '2023-11-15', '19:00:00'),
(22, 3, 5, 2, '2024-10-20', '20:00:00'),
(23, 4, 5, 2, '2024-10-21', '21:00:00'),
(24, 5, 5, 2, '2024-10-22', '22:00:00'),
(25, 1, 5, 2, '2024-10-23', '23:00:00');
```

```
INSERT INTO Appointment (CAID, Reason, Status) VALUES
(1, 'Consultation cardiaque', 'Scheduled'),
(2, 'Vaccination enfant', 'Scheduled'),
(3, 'Douleurs thoraciques', 'Scheduled'),
(4, 'Contrôle post-opératoire', 'Completed'),
(5, 'Contrôle pédiatrique', 'Cancelled');

INSERT INTO Emergency (CAID, TriageLevel, Outcome) VALUES
(6, 2, 'Admitted'),
(7, 1, 'Admitted'),
(8, 3, 'Admitted'),
(9, 4, 'Admitted'),
(10, 2, 'Admitted'),
(11, 3, 'Admitted'),
(12, 1, 'Admitted'),
(13, 2, 'Admitted'),
(14, 4, 'Admitted'),
(15, 3, 'Admitted'),
(16, 2, 'Admitted'),
(17, 1, 'Admitted'),
(18, 3, 'Admitted'),
(19, 2, 'Admitted'),
(20, 4, 'Admitted'),
(21, 3, 'Admitted'),
(22, 2, 'Discharged'),
(23, 1, 'Transferred'),
(24, 3, 'Discharged'),
(25, 2, 'Admitted');

INSERT INTO Insurance (InsID, Type) VALUES
(1, 'CNOPS'),
(2, 'CNSS'),
(3, 'RAMED'),
(4, 'Private'),
(5, 'None');

INSERT INTO Covers (InsID, IID) VALUES
(1, 1),
(2, 2),
(3, 3),
(4, 4),
(5, 5);

INSERT INTO Expense (ExpID, InsID, CAID, Total) VALUES
(1, 1, 1, 450.00),
(2, 2, 2, 200.00),
(3, 3, 3, 0.00),
(4, 4, 4, 1250.00),
```

```
(5, 2, 5, 180.00);

INSERT INTO Medication (MID, Name, Form, Strength, ActiveIngredient, TherapeuticClass
(1, 'Paracétamol', 'Comprimé', '500mg', 'Paracétamol', 'Analgésique', 'Sothema'),
(2, 'Amoxicilline', 'Gélule', '500mg', 'Amoxicilline', 'Antibiotique', 'Laprophan'),
(3, 'Aspirine', 'Comprimé', '100mg', 'Acide acétylsalicylique', 'Antiagrégant plaquet
(4, 'Oméprazole', 'Gélule', '20mg', 'Oméprazole', 'Inhibiteur de la pompe à protons',
(5, 'Ciprofloxacine', 'Comprimé', '500mg', 'Ciprofloxacine', 'Antibiotique', 'Galenic

INSERT INTO Stock (HID, MID, StockTimestamp, UnitPrice, Qty, ReorderLevel) VALUES
(1, 1, '2024-10-01 08:00:00', 2.50, 5000, 500),
(1, 2, '2024-10-01 08:00:00', 8.75, 1200, 200),
(1, 3, '2024-10-01 08:00:00', 250.00, 3500, 400),
(1, 4, '2024-10-01 08:00:00', 180.00, 800, 150),
(1, 5, '2024-10-01 08:00:00', 120.00, 50, 300),
(2, 1, '2024-10-02 09:30:00', 3.20, 3500, 400),
(2, 2, '2024-10-02 09:30:00', 15.50, 150, 200),
(2, 3, '2024-10-02 09:30:00', 300.00, 2000, 400),
(2, 4, '2024-10-02 09:30:00', 190.00, 900, 150),
(2, 5, '2024-10-02 09:30:00', 5.00, 100, 200),
(3, 1, '2024-10-03 10:15:00', 2.00, 4000, 500),
(3, 2, '2024-10-03 10:15:00', 9.00, 1100, 200),
(3, 3, '2024-10-03 10:15:00', 280.00, 3000, 400),
(3, 4, '2024-10-03 10:15:00', 12.00, 800, 150),
(3, 5, '2024-10-03 10:15:00', 125.00, 2000, 300),
(4, 1, '2024-10-04 11:45:00', 2.80, 4500, 500),
(4, 2, '2024-10-04 11:45:00', 10.00, 1300, 200),
(4, 3, '2024-10-04 11:45:00', 320.00, 3200, 400),
(4, 4, '2024-10-04 11:45:00', 185.00, 700, 150),
(4, 5, '2024-10-04 11:45:00', 130.00, 1800, 300),
(5, 1, '2024-10-05 07:30:00', 1.50, 5500, 500),
(5, 2, '2024-10-05 07:30:00', 7.50, 1400, 200),
(5, 3, '2024-10-05 07:30:00', 260.00, 3800, 400),
(5, 4, '2024-10-05 07:30:00', 175.00, 850, 150),
(5, 5, '2024-10-05 07:30:00', 0.00, 50, 300),
(1, 4, '2024-10-06 10:00:00', 12.00, 900, 150);

INSERT INTO Prescription (PID, CAID, DateIssued) VALUES
(1, 1, '2024-11-18'),
(2, 2, '2024-11-20'),
(3, 3, '2024-11-22'),
(4, 4, '2024-10-25'),
(5, 6, '2024-11-10');

INSERT INTO Includes (PID, MID, Dosage, Duration) VALUES
(1, 3, '1 comprimé par jour', '30 jours'),
(2, 1, '1 comprimé 3 fois/jour', '5 jours'),
(3, 5, '1 comprimé 2 fois/jour', '10 jours'),
```

```
(4, 2, '1 gélule 2 fois/jour', '7 jours'),
(5, 4, '1 gélule avant repas', '20 jours');

INSERT INTO ContactLocation (CLID, City, Province, Street, Number, PostalCode, Phone)
(1, 'Casablanca', 'Casablanca', 'Boulevard Zerktouni', '45', '20000', '0522234567'),
(2, 'Rabat', 'Rabat', 'Avenue Mohammed V', '12', '10000', '0537123456'),
(3, 'Fès', 'Fès', 'Rue Talaa Kebira', '78', '30000', '0535678901'),
(4, 'Marrakech', 'Marrakech', 'Avenue Hassan II', '23', '40000', '0524345678'),
(5, 'Tanger', 'Tanger', 'Boulevard Pasteur', '56', '90000', '0539456789');

INSERT INTO Have (IID, CLID) VALUES
(1, 1),
(2, 2),
(3, 3),
(4, 4),
(5, 5);

UPDATE Patient
SET Phone = '+212600112233'
WHERE IID = 201;

UPDATE Hospital
SET Region = 'Marrakech{Safi'
WHERE HID = 1;

DELETE FROM Appointment
WHERE Status = 'Cancelled';


-- DDL 3 (Alter a table to add an attribute)
ALTER TABLE Patient
ADD COLUMN Email VARCHAR(50);

-- Queries

-- 1. Select all patients ordered by last name.
SELECT *
FROM Patient
ORDER BY SUBSTRING_INDEX(FullName, ' ', -1);

-- 2. List distinct insurance types.
SELECT DISTINCT Type
FROM  Insurance ;

-- 3
SELECT DISTINCT s.STAFF_ID, s.FullName
FROM Staff s
JOIN Work_in w ON s.STAFF_ID = w.STAFF_ID
```

```
JOIN Department d ON w.Dep_ID = d.DEP_ID
JOIN Hospital h ON d.HID = h.HID
WHERE h.City = 'Rabat';

-- Query 4 --
select a.*
from Appointment as a
    join ClinicalActivity as CA on a.CAID = CA.CAID
where a.Status  = "Scheduled"
 and CA.date between current_date() and date_add(current_date(), interval 7 day);

-- 5
SELECT CA.DEP_ID, COUNT(*) AS numberOfAppointement
FROM Appointment A , ClinicalActivity CA
WHERE A.CAID = CA.CAID
GROUP BY CA.DEP_ID;

-- query 6
SELECT H.Name AS Hospital, AVG(S.UnitPrice) AS avg_unit_price
FROM Hospital H, Stock S
WHERE H.HID = S.HID
GROUP BY H.Name;

-- query 7
SELECT H.* FROM Hospital H Where H.HID IN
(SELECT D.HID FROM Department D  JOIN ClinicalActivity C ON D.DEP_ID=C.DEP_ID WHERE C
( SELECT E.CAID FROM Emergency WHERE Outcome= 'Admitted' )
GROUP BY D.HID HAVING COUNT(C.CAID) > 20 );

--  8.Find medications in the therapeutic class Antibiotic where the unit price is be
SELECT
    m.MID,
    m.Name,
    m.Form,
    m.Strength,
    m.ActiveIngredient,
    m.TherapeuticClass,
    m.Manufacturer
FROM Medication m
JOIN Stock s ON m.MID = s.MID
WHERE m.TherapeuticClass = 'Antibiotic'
  AND m.UnitPrice < 200;

-- 9. For each hospital list the top three most expensive medications.
SELECT H.Name AS HospitalName, M.Name AS MedicationName, S.UnitPrice
FROM Hospital H
JOIN Stock S ON H.HID = S.HID
JOIN Medication M ON M.MID = S.MID
```

```
WHERE S.UnitPrice >= (
    SELECT MIN(UnitPrice)
    FROM (
        SELECT UnitPrice
        FROM Stock S2
        WHERE S2.HID = H.HID
        ORDER BY UnitPrice DESC
        LIMIT 3
    ) AS top3
);

-- 10
SELECT
    d.DEP_ID,
    (SELECT COUNT(*) FROM Appointment a
     JOIN ClinicalActivity c ON a.cid = c.cid
     WHERE c.DEP_ID = d.DEP_ID AND a.status = 'Scheduled') AS Scheduled_Count,
    (SELECT COUNT(*) FROM Appointment a
     JOIN ClinicalActivity c ON a.cid = c.cid
     WHERE c.DEP_ID = d.DEP_ID AND a.status = 'Completed') AS Completed_Count,
    (SELECT COUNT(*) FROM Appointment a
     JOIN ClinicalActivity c ON a.cid = c.cid
     WHERE c.DEP_ID = d.DEP_ID AND a.status = 'Cancelled') AS Cancelled_Count
FROM Departement d;
-- Query 11 --
select p.*
from Patient as p
where
 not exists(
  select *
        from ClinicalActivity as CA
            join Appointment as A on A.CAID = CA.CAID
  where CA.IID = p.IID
     and A.Status = "Scheduled"
              and CA.date between current_date() and date_add(current_date(), interva
 );
-- 12
SELECT  CA.STAFF_ID, H.HID AS Hospital_Id,  COUNT(*) AS Staff_Appointments,
COUNT(*) * 100.0 / (
        SELECT COUNT(*)
        FROM ClinicalActivity CA2
        JOIN Appointment A2 ON A2.CAID = CA2.CAID
        JOIN Department D2 ON D2.DEP_ID = CA2.DEP_ID
        WHERE D2.HID = H.HID
    ) AS Percentage
FROM ClinicalActivity CA
JOIN Appointment A ON A.CAID = CA.CAID
JOIN Department D ON D.DEP_ID = CA.DEP_ID
```

```
JOIN Hospital H ON H.HID = D.HID
GROUP BY H.HID, CA.STAFF_ID;

-- query 13
SELECT M.Name AS Medication, H.Name AS Hospital
FROM Hospital H, Stock S, Medication M
WHERE H.HID = S.HID AND M.MID = S.MID AND S.Qty < S.OrderLevel;

-- query 14
SELECT H.*  FROM Hospital H
JOIN Stock S ON S.HID=H.HID
JOIN Medication M ON M.MID = S.MID  WHERE M.TherapeuticClass='antibiotic'
GROUP BY H.HID
HAVING COUNT(DISTINCT M.MID )=(SELECT COUNT(*) FROM  Medication M WHERE M.Therapeutic

-- 15. For each hospital and drug class return the average unit price and flag whethe
SELECT
    h.HID,
    m.TherapeuticClass,
    AVG(s.UnitPrice) AS AvgPriceHospital,
    CASE
        WHEN AVG(s.UnitPrice) > (
            SELECT AVG(s2.UnitPrice)
            FROM Stock s2
            JOIN Medication m2 ON s2.MID = m2.MID
            WHERE m2.TherapeuticClass = m.TherapeuticClass
        ) THEN 'Above city avg'
        ELSE 'Not above city avg'
    END AS PriceFlag
FROM Stock s
JOIN Hospital h ON s.HID = h.HID
JOIN Medication m ON s.MID = m.MID
GROUP BY h.HID, m.TherapeuticClass;
-- 16
SELECT
    P.IID,
    P.FullName AS PatientName,
    MIN(CA.Date) AS NextAppointmentDate
FROM
    Patient P
    JOIN ClinicalActivity CA ON P.IID = CA.IID
    JOIN Appointment A ON CA.CAID = A.CAID
WHERE
    A.Status = 'Scheduled'
    AND CA.Date >= CURRENT_DATE
GROUP BY
    P.IID, P.FullName
ORDER BY
```

```
    NextAppointmentDate;
-- 17
SELECT
    p.IID,
    p.FullName,
    COUNT(e.CAID) AS Total_Emergency,
    MAX(c.Date) AS Latest_Emergency,
    DATEDIFF(CURRENT_DATE, MAX(c.Date)) AS Days_Since_Last_Emergency
FROM Patient p
JOIN ClinicalActivity c ON p.IID = c.IID
JOIN Emergency e ON c.CAID = e.CAID
GROUP BY p.IID, p.FullName
HAVING COUNT(e.CAID) >= 2
   AND MAX(c.Date) >= DATE_SUB(CURRENT_DATE, INTERVAL 14 DAY);

-- Query 18 --
select
 H.city,
 H.name as HospitalName,
 count(A.CAID) as CompletedAPP_ID
from Hospital as H
 Join Department D on H.HID = D.HID
 Join ClinicalActivity CA on CA.DEP_ID = D.DEP_ID
 Join Appointment A on A.CAID = CA.CAID
where
 A.Status = "Completed"
    and CA.Date >= date_sub(current_date(), interval 90 day)
Group by
 H.city
order by H.city, CompletedAPP_ID DESC;

-- 19
SELECT  S.MID, H.City, MIN(S.UnitPrice) AS min_price, MAX(S.UnitPrice) AS max_price
FROM Hospital H
JOIN Stock S ON S.HID = H.HID
GROUP BY H.City, S.MID
HAVING (MAX(S.UnitPrice) - MIN(S.UnitPrice)) * 100.0 / MIN(S.UnitPrice) > 30;

-- query 20
SELECT *
FROM Stock
WHERE Qty < 0 OR UnitPrice <= 0;
```

6 Discussion

During the completion of this lab, we encountered several challenges, particularly in the normalization stage. We experienced some confusion regarding which functional depen-

dencies should be included and how they affected the decomposition process. Additionally, certain SQL queries were difficult to implement due to their complexity, especially those involving advanced conditions and aggregations. Nevertheless, the lab was highly beneficial, as it provided a deeper understanding of the underlying concepts and strengthened our practical skills in both normalization and SQL.

7 Conclusion

The work carried out in this lab was highly valuable. We learned about the significant importance of normalization and how neglecting it can lead to various issues, including redundancy and inconsistencies within the database. We also strengthened our knowledge of SQL, its different components, its syntax, and its practical use in managing and querying data. In addition, this lab enhanced our understanding of database management as a whole. It helped clarify the connections between all the steps we have completed so far in this project, from the conceptual design to the actual implementation phase.