



University  
Mohammed VI  
Polytechnic



# Deliverable #2: MNHS database conceptual design

Data Management Course  
UM6P College of Computing

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

---

## Team Information

|                 |   |
|-----------------|---|
| Team Name       | Groupe2   |
| Member 1        | Abir Fakhreddine  |
| Member 2        | Malak El Assali   |
| Member 3        | Nada El Farissi   |
| Member 4        | Amine Chrif   |
| Member 5        | Anass Fertat  |
| Member 6        | Yasser Hallou   |
| Repository Link | <a href="https://github.com/beaNoBeebea">https://github.com/beaNoBeebea</a> |

---

# 1 Introduction

The MNHS needs a database to manage its different entities (patients, staff, hospitals, departments, appointments, prescriptions, medications, insurance, billing, emergencies, etc.) and the different interactions between them. Our work consists of transforming the conceptual schema of the MNHS database into a logical relational model by defining the tables, attributes, primary, and foreign keys, as well as finding ways to implement integrity constraints and discussing when not possible.

# 2 Requirements

In this deliverable, we represent the database in a relational model, discuss the constraints that could not be represented, provide some SQL code snippets and experiment with our database through a query.

The relational model will include tables representing all of the database's entities such as **Staff**, **Contact\_location**, **Hospital**, **Department** and **Clinical\_Activity**. Every table must have a primary key and contain all attributes of its entity with the right data types.

The relational model should also represent all relationships, whether it is a Many-To-Many, a One-To-Many or a One-To-One relationship between entities.

The constraints that could not be represented will also be mentioned, detailing the issues faced.

The relational schema is implemented in SQL, ensuring that all tables, constraints, and relationships are properly defined. In addition, the required query is implemented, and its corresponding output is presented to validate the functionality of the relational model.

# 3 Methodology

We started off with the conversion of all entities (**Patient**, **Staff**, **Hospital**, **Department**, **Clinical\_Activity**, etc.) into tables with their primary key and appropriate attributes with their data types.

The hierarchy of the **Staff** entity (**Practitioner**, **Caregiving\_Staff**, and **Technical\_Staff**) was represented by putting a foreign key referencing the superclass in each of the subclasses.

The hierarchy of the **Clinical\_Activity** entities (**Appointment** and **Emergency**) by putting a foreign key referencing the superclass in each of the subclasses.

The Many-To-Many relationships were represented by creating tables with as a primary key the tuple composed of the two primary keys of the two entities participating in the relationship. Taking as an example the **Has\_Contact\_Location** relationship with a primary key composed of the primary key of the **Patient** entity and the **Contact\_Location** entity. (other examples: **Insurance\_Covers**, **Include\_Medication**, **Stock**..)

The One-To-Many relationships were represented by putting foreign keys referencing the "one" side of the relationship in the "many" side of the relationship. Let's take as an example the relationship between **Department** and **Hospital** which was modeled by referencing **Hospital** using a foreign key in the **Department** entity.

Representing the One-To-One relationships by referencing one of the two entities in the other using a foreign key as is the case of the relationship Generates between **Clinical\_Activity** and **Expense**, since we put **Clinical\_Activity**'s primary key as a foreign key in **expense**.

As for the total participation constraints, we encountered some issues that will be discussed in detail in the Discussion section.

## 4 Implementation & Results

### Relational Schema

#### Entities

**Patient**(IID: integer, *Name*: string, *Sex*: string, *Birth*: date, *Blood\_Group*: string, *CIN*: string, *Phone*: string)

- IID is the PRIMARY KEY
- CIN is unique (by design as a national ID, but also enforced by a database constraint)
- Name is NOT NULL
- Sex is NOT NULL and restricted to values 'M' and 'F'
- Blood\_Group is restricted to values: 'A+', 'A-', 'B+', 'B-', 'AB+', 'AB-', 'O+', 'O-'

**Contact\_Location**(CLID: string, *city*: string, *province*: string, *street*: string, *number*: integer, *postal\_code*: string, *phone\_number*: string)

- CLID is the PRIMARY KEY

**Hospital**(HID: string, *Name*: string, *City*: string, *Region*: string)

- HID is the PRIMARY KEY
- Name is NOT NULL

**Department**(DEP\_ID: string, *Name*: string, *Specialty*: string, *HID*: string)

- DEP\_ID is the PRIMARY KEY
- HID is a FOREIGN KEY referencing Hospital(HID)
- HID is NOT NULL, as it enforces a one-to-many relationship: every department must belong to exactly one hospital. The NOT NULL constraint ensures referential integrity in the one-to-many relationship.

**Staff**(STAFF\_ID: string, *Name*: string, *status*: string)

- STAFF\_ID is the PRIMARY KEY
- Name is NOT NULL
- This is the parent table in an ISA hierarchy

*Practitioner*(STAFF\_ID: string, *License\_Number*: string, *Specialty*: string)

- STAFF\_ID is both the PRIMARY KEY and a FOREIGN KEY referencing Staff(STAFF\_ID)
- This is a child table of the previous one

*Caregiving*(STAFF\_ID: string, *Grade*: string, *Ward*: string)

- STAFF\_ID is both the PRIMARY KEY and a FOREIGN KEY referencing Staff(STAFF\_ID)
- This is a child table of the previous one

*Technical*(STAFF\_ID: string, *Modality*: string, *Certifications*: string)

- STAFF\_ID is both the PRIMARY KEY and a FOREIGN KEY referencing Staff(STAFF\_ID)
- This is a child table of the previous one

**Insurance**(InsID: string, *Type*: string)

- InsID is the PRIMARY KEY
- Domain: *Type* in {CNOPS, CNSS, RAMED, private}.

**Clinical\_Activity** (CAID: string, *Date*: date, *Time*: time, *DEP\_ID*: string, *STAFF\_ID*: string, *IID*: string, *EX\_ID*: string)

- CAID is the PRIMARY KEY
- DEP\_ID is a FOREIGN KEY referencing Department(DEP\_ID)
- STAFF\_ID is a FOREIGN KEY referencing Staff(STAFF\_ID)
- IID is a FOREIGN KEY referencing Patient(IID)
- EX\_ID is a FOREIGN KEY referencing Expense(EX\_ID)
- A clinical activity occurs in exactly one department, so DEP\_ID is NOT NULL
- A clinical activity is linked to exactly one staff member, so STAFF\_ID is NOT NULL
- A clinical activity has exactly one patient, so IID is NOT NULL
- A clinical activity generates exactly one expense, so EX\_ID is NOT NULL
- Additional note: Clinical\_Activity is in a one-to-one relationship with Expense, and the lack of correct modelization of this relationship will be discussed later on

- Finally, Clinical\_Activity is the parent entity to an ISA hierarchy

*Appointment*(CAID: string, *Reason*: string, *Outcome*: string, *Status*: string)

- An appointment is a clinical activity.
- Domain: *Status*  $\in \{Scheduled, Completed, Cancelled\}$ .
- CAID is a FOREIGN KEY referencing Clinical\_Activity(CAID), and is also the relation's PRIMARY KEY, as this is a child entity to Clinical\_Activity

*Emergency*(CAID: string, *Triage\_Level*: string, *Outcome*: string)

- An emergency is a clinical activity.
- CAID is a FOREIGN KEY referencing Clinical\_Activity(CAID), and is also the relation's PRIMARY KEY, as this is a child entity to Clinical\_Activity

**Expense**(EX\_ID: string, *Total*: decimal, *InsID*: string)

- Each expense is generated by exactly one clinical activity.
- Each expense is attached to at least one insurance.
- InsID is a FOREIGN KEY referencing Insurance(InsID), it is NOT NULL, as Expense and Insurance are in a one-to-many relationship
- As mentionned previously, Clinical Activity and Expense are in a relationship. So, the logical representation would be to add a foreign key referencing Clinical\_Activity here as well. However, this would put us in a "chicken or egg" dilemma, as we couldn't create any of two relation instances from these relations without first creating the other. So, this is a constraint we can't represent with the tools we currently have.

**Prescription**(PID: string, *Date\_Issued*: date, *CAID*: string)

- A prescription is generated by exactly one clinical activity.
- CAID is a FOREIGN KEY referencing Clinical\_Activity(CAID). It is UNIQUE and NOT NULL, as every Prescription is linked to exactly one Clinical\_Activity, and each Clinical\_Activity is linked to at most one Clinical\_Activity.

**Medication**(DrugID: string, *Name*: string, *Class*: string, *Form*: string, *Strength*: string, *Active\_Ingredient*: string, *Manufacturer*: string)

- DrugID is the PRIMARY KEY

## Relationships

### One-to-Many Relationships

Foreign keys in Department, Clinical\_Activity, and Prescription represent one-to-many relationships.

## Many-to-Many Relationships

**Has\_Contact\_Location**(IID: string, CLID: string)

- (CLID,IID) is the PRIMARY KEY
- IID is a FOREIGN KEY referencing Patient(IID)
- CLID is a FOREIGN KEY referencing Contact\_Location(CLID)

**Insurance\_Covers**(InsID: string, IID: string)

- (InsID,IID) is the PRIMARY KEY
- IID is a FOREIGN KEY referencing Patient(IID)
- InsID is a FOREIGN KEY referencing Insurance(InsID)

**Include\_Medication**(PID: string, DrugID: string, *dosage*: decimal, *duration*: integer)

- (PID,DrugID) is the PRIMARY KEY
- PID is a FOREIGN KEY referencing Prescription(PID)
- DrugID is a FOREIGN KEY referencing Medication(DrugID)

**Stock**(DrugID: string, HID: string, *Stock\_Timestamp*: datetime, *Qty*: integer, *Unit\_Price*: decimal, *Reorder\_Level*: integer)

- (HID,DrugID) is the PRIMARY KEY
- HID is a FOREIGN KEY referencing Hospital(HID)
- DrugID is a FOREIGN KEY referencing Medication(DrugID)

**Work\_In**(STAFF\_ID: string, DEP\_ID: string)

- (STAFF\_ID,DEP\_ID) is the PRIMARY KEY
- STAFF\_ID is a FOREIGN KEY referencing Staff(STAFF\_ID)
- DEP\_ID is a FOREIGN KEY referencing Department(DEP\_ID)
- This relationship has a total participation constraint on the side of Staff, this cannot be represented with the tools we currently possess, as we cannot prove that somehow every single instance of Staff exists in at least one instance of Work\_In

## 4.1 Partial Implementation of the Relational Schema in SQL

```
CREATE DATABASE LAB3;
USE LAB3;
```

```
CREATE TABLE Patient (
    IID INT PRIMARY KEY,
    CIN CHAR(11) UNIQUE,
    Name VARCHAR(50) NOT NULL,
    Sex ENUM( 'M' , 'F' ) NOT NULL,
    Birth DATE,
    Bloodgroup ENUM( 'A+' , 'A-' , 'B+' , 'B-' , 'AB+' , 'AB-' , 'O+' , 'O-' ) ,
    Phone VARCHAR(20)
);
```

```
CREATE TABLE Hospital(
    HID CHAR(11) PRIMARY KEY,
    Name VARCHAR(50) NOT NULL,
    City VARCHAR(50) ,
    Region VARCHAR(50)
);
```

```
CREATE TABLE Department(
    DEP_ID CHAR(11) PRIMARY KEY,
    Name VARCHAR(50) NOT NULL,
    Speciality VARCHAR(50) ,
    HID CHAR(11) NOT NULL, — ONE-TO-MANY
    FOREIGN KEY (HID) REFERENCES Hospital(HID)
);
```

```
CREATE TABLE Staff (
    STAFF_ID CHAR(11) PRIMARY KEY,
    Name VARCHAR(100) NOT NULL,
    status VARCHAR(20)
);
```

```
CREATE TABLE Insurance (
    InsID CHAR(11) PRIMARY KEY,
    Type VARCHAR(50)
);
```

```
CREATE TABLE Expense (
    EX_ID CHAR(11) PRIMARY KEY,
    TOTAL DECIMAL,
    InsID CHAR(11) NOT NULL, — ONE-TO-MANY
    /* —EXPENSE IS IN A ONE-TO-ONE RELATIONSHIP WITH CLINICAL
    ACTIVITY, SO, LOGICALLY, WE SHOULD ADD A NOT NULL REFERENCE
    —TO CLINICAL ACTIVITY HERE AS WELL, THIS HOWEVER CREATES AN EGG AND
    CHICKEN PROBLEM SO WE ABSTAIN*/
    FOREIGN KEY (InsID) REFERENCES Insurance(InsID)
);
```

```
CREATE TABLE Clinical_Activity (
    CAID CHAR(11) PRIMARY KEY,
    Date DATE,
    Time TIME,
    DEP_ID CHAR(11) NOT NULL,
    STAFF_ID CHAR(11) NOT NULL,
    EX_ID CHAR(11) NOT NULL,
    IID INT NOT NULL,
    FOREIGN KEY (DEP_ID) REFERENCES Department(DEP_ID),
    FOREIGN KEY (STAFF_ID) REFERENCES Staff(STAFF_ID),
    FOREIGN KEY (IID) REFERENCES Patient(IID),
    FOREIGN KEY (EX_ID) REFERENCES Expense(EX_ID)
);
```

```
CREATE TABLE Appointment(
    CAID CHAR(11) PRIMARY KEY,
    Status VARCHAR(50),
    Reason VARCHAR(50),
    FOREIGN KEY (CAID) REFERENCES Clinical_Activity(CAID)
    ON DELETE CASCADE
);
```

```
INSERT INTO Patient VALUES
(1, 'CIN001', 'Aymane-Tahiri', 'M', '2003-06-15', 'A+', '0612345678'),
(2, 'CIN002', 'Sara-El-Fassi', 'F', '1999-03-21', 'B-', '0623456789'),
(3, 'CIN003', 'Youssef-Idrissi', 'M', '1988-11-02', 'O+', '0634567890');
INSERT INTO Hospital VALUES
('H1', 'Mohammed-VI-Hospital', 'Benguerir', 'Marrakech-Safi'),
('H2', 'Avicenne-Hospital', 'Casablanca', 'Casablanca-Settat'),
('H3', 'CHU-Rabat', 'Rabat', 'Rabat-Sale-Kenitra');
```

```
INSERT INTO Department VALUES
```



```
( 'D1' , 'Cardiology' , 'Heart' , 'H1' ),  
( 'D2' , 'Neurology' , 'Brain' , 'H2' ),  
( 'D3' , 'Orthopedics' , 'Bones' , 'H3' );
```

**INSERT INTO Staff VALUES**

```
( 'S1' , 'Dr.-Amina-Rahimi' , 'Active' ),  
( 'S2' , 'Dr.-Karim-El-Mansouri' , 'Active' ),  
( 'S3' , 'Dr.-Yasmine-Berrada' , 'Active' );
```

**INSERT INTO Insurance VALUES**

```
( 'I1' , 'Basic' ),  
( 'I2' , 'Premium' ),  
( 'I3' , 'VIP' );
```

**INSERT INTO Expense VALUES**

```
( 'E1' , 100 , 'I1' ),  
( 'E2' , 200 , 'I2' ),  
( 'E3' , 150 , 'I3' );
```

**INSERT INTO Clinical\_Activity VALUES**

```
( 'CA1' , '2025-10-12' , '10:00:00' , 'D1' , 'S1' , 'E1' , 1 ),  
( 'CA2' , '2025-10-13' , '11:30:00' , 'D2' , 'S2' , 'E2' , 2 ),  
( 'CA3' , '2025-10-14' , '09:15:00' , 'D3' , 'S3' , 'E3' , 3 );
```

**INSERT INTO Appointment VALUES**

```
( 'CA1' , 'Scheduled' , 'Routine-check-up' ),  
( 'CA2' , 'Completed' , 'MRI-follow-up' ),  
( 'CA3' , 'Scheduled' , 'Knee-pain-consultation' );
```

## 4.2 SQL Query Experimentation

To validate part of the relational schema implementation, we executed a query that lists the name of patients with scheduled appointments in the city of Benguerir.

```
SELECT p.Name AS Patient_Name  
FROM Patient p  
JOIN Clinical_Activity ca ON p.IID = ca.IID  
JOIN Department d ON ca.DEP_ID = d.DEP_ID  
JOIN Hospital h ON d.HID = h.HID  
JOIN Appointment a ON ca.CAID = a.CAID  
WHERE h.City = 'Benguerir' AND a.Status = 'Scheduled';
```

### 4.3 Query Output and Interpretation

The output below represents the result of executing the above query on the MNHS database.

| Patient_Name  |
|---------------|
| Aymane Tahiri |

Figure 1: Result of the SQL query showing the name of patients with scheduled appointments in the city of Benguerir

## 5 Discussion

The main challenges we faced were the following:

- The representation of the total participation constraint. For example in the **Works\_in** relationship we have total participation of the **Staff** entity, however we could only represent the fact that it's a many-to-many relationship by creating a new table for the relationship without modeling the fact that a **Staff** member must work in at least one **Department**.
- Another challenge is the representation of the one-to-one relationship **Generates** between **Clinical\_Activity** and **Expense**. Logically, we could add a foreign key in both entities. However, doing so creates a circular dependency also called the "chicken or egg" dilemma. We cannot insert one without the other already existing. This restriction cannot be fully represented with the tools currently available. Therefore, only one foreign key (from **Expense** to **Clinical\_Activity**) is implemented, enforcing only the fact that every **Expense** is linked to exactly one **Clinical\_Activity**.

## 6 Conclusion

The relational design developed throughout this work models at best the entities, attributes, and relationships of the MNHS database enforcing primary and foreign key constraints to preserve referential integrity. The relational schema was implemented in SQL, and the required queries were executed to validate the correctness of the model.

Some semantic constraints, such as total participation and mutually dependent one-to-one relationships, could not be fully represented within the limits of the tools currently available. Nevertheless, the final schema provides a consistent and functional structure that accurately reflects the original ER model.