**NewYork-Presbyterian Hospital Division Database Project**

**ER MODEL USING UML NOTATION**

**A diagram of a medical application

Description automatically generated**

**Entity:**

1. Patient: Patient\_id
2. Department: Department\_id
3. Staff: can be superclass, having doctor, nurse, office people, etc. subclasses
4. Appointment:
5. TreatmentType: Surgery, etc
6. Treatment: each patient each visit, treatment history
7. Prescription: Drug\_Name
8. Billing: Billing\_Num
9. Patient
10. Payment

**RELATIONSHIP SENTENCES**

**Patient - department**

One patient gets treatments from one or many departments

One department may have zero or many patients.

**Patient - Appointment**

One patient may make zero or many appointments

One appointment allows one and only one patient to make it.

**Patient - Medical prescription**

One patient must have zero or many medical prescriptions

One Medical prescription may be prescribed to zero or many patients

**Medical prescription- Drug**

One prescription will have one or many drugs

One drug can be used by zero or many prescriptions

**Staff - Patient**

One staff must assist zero or many patients

One patient must be assisted by one or many staff

**Staff - department**

One staff belongs to one and only department.

Each department may have zero or many staff.

**Staff - Appointment**

One staff can be appointed to zero or one appointment.

One appointment may require one or many staff to serve.

**TreatmentType - Patient**

One patient may have zero or many Treatments Type.

One Treatment Type may be operated on zero or many patients.

**TreatmentType - Staff**

One Treatment Type may be assigned by one or many staff

One staff may be assigned to zero or many Treatments Type.

**Patient - TreatmentType**

One patient has one or many Treatments Type

One Treatment Type has zero or many patients

**Patient - Treatment**

One Patient may have zero or many treatments

One treatment may be recorded for one and only one patient.

**Staff - prescription**

One staff may prescribe zero or many prescriptions to patients.

One prescription may be assigned to patients by one and only one staff.

**Patient - Payment**

One patient has zero or many payments to pay.

One payment is paid by one and only patients.

**Billing - Department**

One billing is created by one or many departments.

One department creates zero or many billings.

**Billing - Payment**

One billing can collect one or many payments.

One payment is paid for one or many billings.

**Recursive Relationship**

One Manager can manage one or many staff

One staff may manage by one and only one manager

**CONVERTING THE ERD TO A RELATIONAL MODEL: ERD->RDM**

* Staff(staff\_id,department\_id,….etc)
* Staff\_Supervisor(staff\_id, supervisor\_id)
* nonmedical\_staff(staff\_id,nonmedicalStaff\_id)
* medical\_staff(staff\_id,medicalStaff\_id)
* Appointment(appointment\_id,patient\_id,date,staff\_id,…etc)
* Patients(patients\_id,patients\_name,patients\_address,patients\_PhonNum, patient\_emergencyInfo, department\_id)
* Department(department\_id,department\_name,department\_location, ...etc)
* Medical\_Prescription(prescription\_id,patients\_id,drug\_name,prescription\_date,… etc)
* Drug(drug\_name,brand)
* Treatment\_Name(treatment\_name,patients\_id,treatment\_frequency,treatment\_date,… etc)
* TreatmentType(TreatmentType\_Code,treatment\_name,…etc)
* Billing(billing\_num,patients\_id,date,billing\_amount,department\_id,…etc)
* Payment(payment\_num,payment\_date,payment\_amount,…etc) Billing\_Payment(payment\_num,billing\_num)

**NORMALIZATION**

**Payment**

Payment(payment\_num, payment\_date, payment\_amount)

Step 1: Identify the primary key (payment\_num)

Step 2: Identify FDs

FD1: payment\_num → payment\_date, payment\_amount

Step 3:

1. Do we have key? Yes, in 1NF
2. Do we have partial key problem? No

R1(payment\_num, payment\_date, payment\_amount)

1. Do we have transitive dependency? No
2. Do we have non-key determine part of key? No

The final relations are R1.

**Billing**

Billing(billing\_num, patient\_id, date, billing\_amount, department\_id)

Step 1: Identify the primary key (billing\_num)

Step 2: Identify FDs

FD1: billing\_num → date, patient\_id, billing\_amount, department\_id

FD2: patient\_id → billing\_amount

Step 3:

1. Do we have key? Yes, in 1NF

~~R1(billing\_num, date, patient\_id, billing\_amount, department\_id)~~

1. Do we have partial key problem? No
2. Do we have transitive dependency? Yes, FD2

R2(patient\_id, billing\_amount)

R3(billing\_num, date, patient\_id, department\_id)

1. Do we have non-key determine part of key? No

The final relations are R2 and R3.

**Drug**

Drug(drug\_name)

Step 1: Identify the primary key (drug\_name)

Step 2: Identify FDs

FD1: drug\_name Step

Step 3:

1. Do we have key? Yes, in 1NF

R1(drug\_name)

1. Do we have partial key problem? No, it is in 2NF
2. Do we have transitive dependency? No
3. Do we have non-key determine part of key? No

The final Relation is R1.

**Patients**

Patients(patient\_id, patients\_dapartment, patients\_name, patients\_address, patients\_phoneNum, patients\_emergencyinfo)

Step 1: Identify the primary key: patient\_id

Step 2: Identify FDs

FD1: patient\_id-> patients\_dapartment, patients\_name, patients\_address, patients\_phoneNum, patients\_emergencyinfo

FD2: patients\_name, patients\_address ->patients\_dapartment,patients\_phoneNum, patients\_emergencyinfo

Step 3:

1. Do we have key? Yes, 1NF
2. Do we have partial key dependency? No, it is in 2NF

~~R1(patient\_id, patients\_dapartment, patients\_name, patients\_address, patients\_phoneNum, patients\_emergencyinfo)~~

1. Do we have transitive dependency? Yes,

patients\_name, patients\_address ->patients\_dapartment,patients\_phoneNum, patients\_emergencyinfo

R2(patients\_name, patients\_address, patients\_dapartment,patients\_phoneNum, patients\_emergencyinfo)

R3(patient\_id, patients\_name, patients\_address)

Checked, in 3NF

1. Do we non-key attribute determines part of key? No

The final relations are R2 and R3.

**Department**

Step 1: identify primary key (department\_id,)

Step 2: FD1:department\_id->department\_name,department\_location,phone\_num,department\_class

FD2:department\_name->department\_location,phone\_num

Step3:

1. Any keys? Yes,1NF
2. Any partial dependency? No, 2NF

Checking any partial? No

~~R1 (department\_id,department\_name,department\_location,phone\_num,department\_class)~~

1. Any transitive dependency? Yes,department\_name->department\_location,phone\_num

R2(department\_name,department\_location,phone\_num)

R3(department\_id,department\_name,department\_class)

1. Is Any determinant not candidate key? No

The final relations are R2 and R3.

**Staff**

Step1: identify primary key (staff\_id)

Step2:

FD1:staff\_id->staff\_name,staff\_address,phone\_num,gender,email,department\_id

FD2:staff\_name->staff\_address，phone\_num,email

Step3:

1. Any keys?Yes,1NF
2. Any partial dependency?No,2NF

~~R1(staff\_id,staff\_name,staff\_address,phone\_num,gender,email,department\_id)~~

1. Any transitive dependency?Yes, staff\_name->staff\_address，phone\_num,email

R2(staff\_name,staff\_address，phone\_num,email)

R3(staff\_id,staff\_name, gender,department\_id)

1. Any determinant not candidate key? No

The final relations are R2 and R3

**Appiontment**

Appointment(patient\_id, date, staff\_num)

Step 1: Identify the primary key (patient\_id)

Step 2: Identify FDs

FD1: patient\_id → date, staff\_num

Step 3:

1. Do we have key? Yes, in 1NF

R1(patient\_id, date, staff\_num)

1. Do we have partial key problem? No, it is in 2NF
2. Do we have transitive dependency? No
3. Do we have non-key determine part of key? No,

The final relations are R1.

**Treatment\_Name**

Step1: identify primary key (treatment\_name)

Step2:

FD1:treatment\_name->patients\_id,treatment\_history,visit\_date,visit\_time,staff\_id

FD2:patients\_id->treatment\_history,staff\_id

1. Any keys? Yes,1NF

2. Any partial dependency?No,2NF ~~R1(treatment\_name,patients\_id,treatment\_history,visit\_date,visit\_time,staff \_id)~~

3. Any transitive dependency?Yes, patients\_id->treatment\_history,staff\_id

R2(patients\_id,treatment\_history,staff\_id )

R3(treatment\_name,patients\_id,visit\_date,visit\_time)

4. Any determinant candidate key? No

Final relations: R2, R3.

**Treatment\_Type**

Step1: identify primary key(TreatmentType\_code)

Step2:

FD1: TreatmentType\_code->treatment\_id,treatment\_name,treatment\_level,billing\_code

FD2: treatment\_id->treatment\_name

1. Any keys? Yes,1NF

2. Any partial dependency? No,2NF

~~R1(TreatmentType\_code, treatment\_id, treatment\_name, treatment\_level, billing\_code)~~

3. Any transitive dependency? Yes, treatment\_id->treatment\_name

R2(treatment\_id,treatment\_name)

R3(TreatmentType\_code,treatment\_id, treatment\_level,billing\_code)

4. Any determinant candidate key? No

Final relations: R2, R3.

**CREATING A TABLE WITH SQL**

**STAFF Table**

CREATE TABLE Staff (

staff\_id VARCHAR(50) NOT NULL,

staff\_name VARCHAR(50) NOT NULL,

staff\_address VARCHAR(200),

phone\_num VARCHAR(14),

gender VARCHAR(10),

email VARCHAR(30),

department\_id VARCHAR(50),

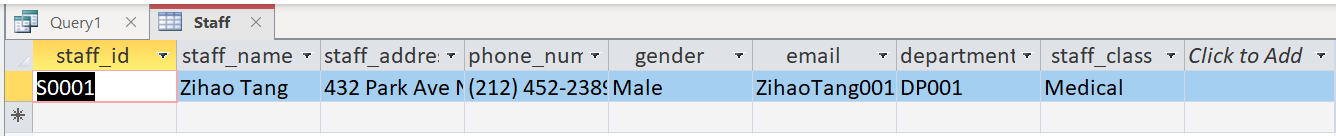
staff\_class VARCHAR(50),

CONSTRAINT pk\_Staff PRIMARY KEY(staff\_id),

CONSTRAINT fk\_Staff1 FOREIGN KEY(department\_id) REFERENCES Department(department\_id));

INSERT INTO Staff

VALUES ('S0001', 'Zihao Tang','432 Park Ave New York NY 10022', '(212) 452-2389', 'Male', 'ZihaoTang001@gmail.com', 'DP001', 'Medical');

****

**Department Table**

CREATE TABLE Department(

department\_id VARCHAR(50) NOT NULL,

department\_name VARCHAR(50) NOT NULL,

department\_location VARCHAR(200),

phone\_num VARCHAR(200),

department\_class VARCHAR(10),

CONSTRAINT pk\_Department PRIMARY KEY(department\_id)

);

INSERT INTO Department

VALUES ('DP001', 'Outpatient','B1', '(347)333-0001', 'Medical');

A screenshot of a computer

Description automatically generated

**Payment Table**

CREATE TABLE Payment(

payment\_num VARCHAR(50) NOT NULL,

payment\_date DATE,

payment\_amount NUMBER NOT NULL,

payment\_method VARCHAR(200),

billing\_num VARCHAR(50),

CONSTRAINT pk\_Payment PRIMARY KEY(payment\_num),

CONSTRAINT fk\_Payment FOREIGN KEY(billing\_num) REFERENCES Billing(billing\_num));

INSERT INTO Payment

VALUES ('PM001', '1/24/2022', 23000, 'Credit Card','B211224288');

**A screen shot of a computer

Description automatically generated**

**Appointment Table**

CREATE TABLE Appointment (

appointment\_id VARCHAR(50) NOT NULL,

Apt\_date DATE,

Apt\_time VARCHAR(20),

patients\_id VARCHAR(50),

staff\_id VARCHAR(50),

CONSTRAINT pk\_Appointment PRIMARY KEY(appointment\_id),

CONSTRAINT fk\_Appointment FOREIGN KEY(patients\_id) REFERENCES Patient(patients\_id)),

CONSTRAINT fk\_Appointment FOREIGN KEY(staff\_id) REFERENCES Staff(staff\_id));

INSERT INTO Appointment

VALUES ('APT22112111','11/21/2022','8:00:00 AM','P20060615','S0013')

A screenshot of a computer

Description automatically generated

**Patient Table**

CREATE TABLE Patient (

patients\_id VARCHAR(50) NOT NULL,

patients\_name VARCHAR(50),

patients\_address VARCHAR(200),

patients\_phoneNum VARCHAR(20),

patients\_email VARCHAR(100),

patients\_emergencyinfo VARCHAR(20),

department\_id VARCHAR(50),

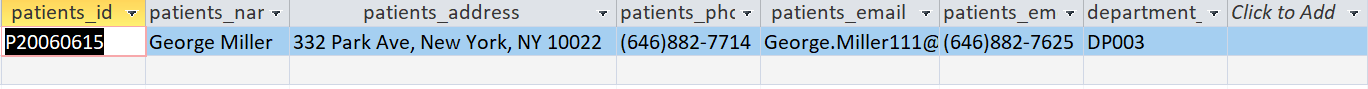
CONSTRAINT pk\_Patient PRIMARY KEY(patients\_id),

CONSTRAINT fk\_Patient FOREIGN KEY(department\_id) REFERENCES Department(department\_id)

);

INSERT INTO Patient

VALUES ('P20060615', 'George Miller', '332 Park Ave, New York, NY 10022', '(646)882-7714','George.Miller111@gmail.com', '(646)882-7625', 'DP003');



**Billing Table**

CREATE TABLE Billing (

billing\_num VARCHAR(50) NOT NULL,

billing\_date DATE,

billing\_amount NUMBER NOT NULL,

department\_id VARCHAR(50),

patients\_id VARCHAR(50),

CONSTRAINT pk\_Billing PRIMARY KEY(billing\_num),

CONSTRAINT fk\_Billing1 FOREIGN KEY(patients\_id) REFERENCES Patient(patients\_id),

CONSTRAINT fk\_Billing2 FOREIGN KEY(department\_id) REFERENCES Department(department\_id));

INSERT INTO Billing

VALUES ('B211224288', '12/24/2021', '23000.00', 'DP001', 'P20060615');

A close-up of a computer screen

Description automatically generated

**NonMedicalStaff Table**

CREATE TABLE NonMedicalStaff (

Non\_Medical\_Staff\_ID VARCHAR(50) NOT NULL,

CONSTRAINT pk\_NonMedicalStaff PRIMARY KEY(Non\_Medical\_Staff\_ID),

CONSTRAINT fk\_NonMedicalStaff FOREIGN KEY(Non\_Medical\_Staff\_ID) REFERENCES Staff(staff\_id));

INSERT INTO NonMedicalStaff

VALUES ('S0004');

A yellow box with black text

Description automatically generated

**MedicalStaff Table**

CREATE TABLE MedicalStaff (

Medical\_Staff\_ID VARCHAR(50) NOT NULL,

CONSTRAINT pk\_Medical\_Staff PRIMARY KEY(Medical\_Staff\_ID),

CONSTRAINT fk\_Medical\_Staff FOREIGN KEY(Medical\_Staff\_ID) REFERENCES staff(staff\_id)

);

INSERT INTO MedicalStaff

VALUES ('S0001');

A yellow and black text

Description automatically generated

**Prescription Table**

CREATE TABLE Prescription(

Prescription\_id VARCHAR(100) NOT NULL,

patient\_id VARCHAR(50),

patients\_name VARCHAR(100),

drug\_name VARCHAR(100),

Prescription\_date DATE,

CONSTRAINT pk\_Prescription PRIMARY KEY(Prescription\_id),

CONSTRAINT fk\_Prescription1 FOREIGN KEY(patient\_id) REFERENCES Patient(patients\_id),

CONSTRAINT fk\_Prescription2 FOREIGN KEY(drug\_name) REFERENCES Drug(drug\_name));

INSERT INTO Prescription(

VALUES (PRES202203054, 'P19120305', 'Steve Parson', Ibuprofin, '2/22/2022');

A screenshot of a computer

Description automatically generated

**Treatment\_Name Table**

CREATE TABLE Treatment\_Name(

treatment\_name VARCHAR(150) NOT NULL,

patients\_id VARCHAR(50) ,

treatment\_history VARCHAR(150),

Visit\_time DATE,

CONSTRAINT pk\_Treatment\_Name PRIMARY KEY (treatment\_name)

CONSTRAINT fk\_Treatment\_Name FOREIGN KEY(patients\_id)

REFERENCES Patient(patients\_id)

);

INSERT INTO Treatment\_Name (treatment\_name, patients\_id, treatment\_history, Visit\_time)

VALUES ("Chiropractic", "P20060615", "Chiropractic", '12/24/2021')

A screenshot of a computer

Description automatically generated

**Treatment\_Type Table**

CREATE TABLE Treatment\_Type(

treatment\_typeCode VARCHAR(150) NOT NULL,

treatment\_id NUMBER,

treatment\_name VARCHAR(150),

treatment\_level VARCHAR(150),

billing\_code VARCHAR(50),

CONSTRAINT pk\_Treatment\_Type PRIMARY KEY (treatment\_typeCode),

CONSTRAINT fk\_Treatment\_Type FOREIGN KEY(treatment\_name) REFERENCES Treatment\_Name(treatment\_name)

);

INSERT INTO Treatment\_Type (treatment\_typeCode,treatment\_id, treatment\_name, treatment\_level,billing\_code)

VALUES ("003-1073",1073, "Chiropractic", "H", "2412211073")

A screenshot of a computer screen

Description automatically generated

**SCENARIO:**

1. Write a query to display the department\_id, and total employees working in each department and display the result from the highest number of workers.

SELECT department\_id, COUNT(department\_id) AS Number\_of\_emplyees

FROM Staff

GROUP BY department\_id

ORDER BY COUNT(department\_id) DESC;

A screenshot of a computer

Description automatically generated

1. Write a query to match patients with their billing information by displaying the patient id, patient name, billing number, billing date, and billing amount, and sort the results by patient id.

SELECT P.patients\_id, patients\_name, billing\_num, billing\_date, billing\_amount

FROM Patient P INNER JOIN Billing B

ON P.patients\_id = B.patients\_id

ORDER BY P.patients\_id;

A screenshot of a computer

Description automatically generated

3. Write a query to show the patient id and name of who took the fever or cold medicine after the Nov of 2021

SELECT patient\_id,patients\_name

FROM Prescription

WHERE drug\_name = "Ibuprofin" OR "Tylenol"

AND prescription\_date > 11/01/2021;

A screenshot of a table

Description automatically generated

4, Write a query to display Non\_Medical\_Staff\_ID and Name the Gender are Female.

SELECT N.Non\_Medical\_Staff\_ID, S.staff\_name

FROM Staff AS S

RIGHT JOIN NonMedicalStaff AS N

ON S.staff\_id = N.Non\_Medical\_Staff\_ID

WHERE S.gender = 'Female';

A screenshot of a computer

Description automatically generated

5, Write a query to show patients\_id and treatment\_name, the treatment\_history are “Alexander Technique”

SELECT patients\_id, treatment\_name

FROM TREATMENT\_NAME

WHERE Treatment\_history LIKE "Alexander Technique";

A screenshot of a computer

Description automatically generated

**CONCLUSION**

The creation of the Entity-Relationship Diagram (ERD) was the most challenging and time-consuming aspect of this project. As the initial step, it represented the first major hurdle. Initially, it was daunting to consider every perspective, from medical and administrative staff to patients themselves, in order to develop a system that was both efficient and comprehensive.

To achieve this, I engaged in a process of continuous refinement, repeatedly revising the ERD to eliminate superfluous attributes and clarify key relationships. This iterative process not only enhanced my understanding of database relations, primary keys, and foreign keys but also improved my ability to link disparate instances effectively.

The transition from ERD to Relational Database Model (RDM) and Boyce-Codd Normal Form (BCNF) deepened my grasp of database structures. By defining these elements, I reached an exciting phase of the project: implementing the system in Access SQL. Here, I crafted SQL tables with defined primary and foreign keys, enabling efficient data queries. This allowed for quicker access to patient information, thereby reducing patient wait times and improving the overall efficacy of the healthcare system.

In summary, this project was a journey of overcoming initial complexities to create a user-focused database system that caters to the dynamic needs of a healthcare environment.