2025BATCH

**HealthHub**

**DBMS PROJECT**

**4th SEMESTER**

**Submitted by-:**

Ishita Garg 102203992

Akshita Pathak 102203796

Ayushi Rathore 102203816

Aarushi Bajaj 102203820

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **S NO.** | **CONTENT** | **PAGE NO.** |
| 1 | Requirements | 3 |
| 2 | ER Diagram & assumptions | 4 |
| 3 | Relational Schema | 5 |
| 4 | Normalization | 6 |
| 5 | SQL statements for table creation | 8 |
| 6 | SQL statements for insertion commands | 11 |
| 7 | PL/SQL Code | 12 |

**REQUIREMENTS**

The doctor-patient management system is a software application that is designed to manage the interactions between doctors and patients. The system allows doctors to manage their appointments, prescriptions, and medical records for their patients. Patients can use the system to schedule appointments, view their medical records, and receive prescriptions.

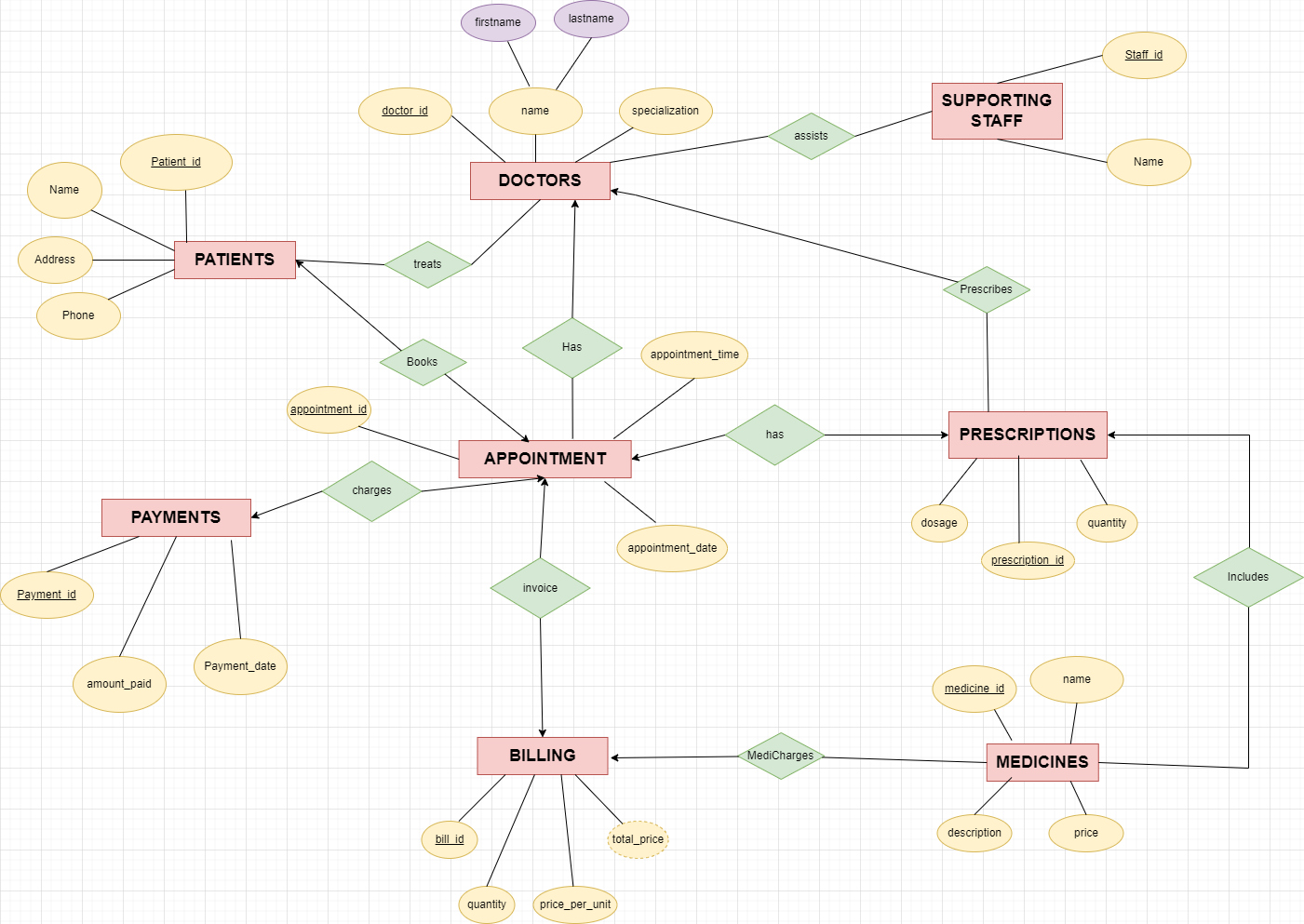
The system is implemented using a database to store the doctor, patient, appointment, prescription, billing, and payment information. The system also includes a user interface that allows doctors and patients to interact with the database.

The doctor-patient management system can be used in a variety of healthcare settings, such as hospitals, clinics, and private practices. The system can improve the efficiency and accuracy of managing patient information, reduce errors, and enhance communication between doctors and patients.

To commercialize the system, it can be marketed to healthcare organizations and private practices. The system can be sold as a subscription-based service or a one-time purchase. The system can also be customized to meet the specific needs of healthcare organizations and practices, which can provide an additional source of revenue.

In summary, the doctor-patient management system is a valuable tool for managing patient information, improving communication, and enhancing the overall quality of healthcare. It can be implemented in various healthcare settings and can be commercialized through subscriptions or customized solutions for organizations and practices.

# MODELLING OF REQUIREMENTS AS ER-DIAGRAM



**ASSUMPTIONS**:

A Doctor can have many Appointments, but each Appointment is with only one Doctor.

A Patient can have only one Appointment and each Appointment has only one Patient.

A Prescription is associated with one Appointment and can have many Medicines.

Many Medicines can be associated with one Prescription and one Billing.

Many supporting staff assist many Doctors.

A Billing is associated with one Appointment and can have many Medicines.

A Payment is associated with one Appointment.

# ER- DIAGRAM INTO RELATIONAL SCHEMA

## Patients

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| patient\_id | name | | | address | | phone | | | doctor\_id | | doctor\_name | | |
| specialization | | | appointment\_date | | | | appointment\_time | | medicine\_name | | dosage | |
| quantity | | price\_per\_unit | | | total\_price | | | payment\_amount | | payment\_date | |

# AFTER NORMALIZATION

## Doctors

|  |  |  |  |
| --- | --- | --- | --- |
| doctor\_id | name | Specialization | staff\_id |

Primary Key: doctor \_id

Foreign Keys: staff\_id

## Patients

|  |  |  |  |
| --- | --- | --- | --- |
| patient\_id | name | address | phone |

∙ Primary Key: patient\_id

## Appointments

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| appointment\_id | doctor\_id | patient\_id | appointment\_date | appointment\_time |

Primary Key: appointment\_id

Foreign Keys: doctor\_id , patient\_id

## Medicines

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| medicine\_id | name | description | price | Bill\_id |

Primary Key: medicine \_id

Foreign Keys: bill\_id

## Prescriptions

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| prescrip tion\_id | appointme nt\_id | medicine\_ id | dosage | quantity |

Primary Key: prescription\_id

Foreign Keys:appointment\_id, medicine\_id

## Billing

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| bill\_id | appointment\_id | medicine\_id | quantity | price\_pe r\_unit | total\_pri ce |

Primary Key: bill\_id

Foreign Keys: appointment\_id, medicine\_id

## Payments

|  |  |  |  |
| --- | --- | --- | --- |
| payment\_id | appointment\_id | amount\_paid | payment\_date |

∙ Primary Key: payment\_id

∙ Foreign Keys:appointment\_id

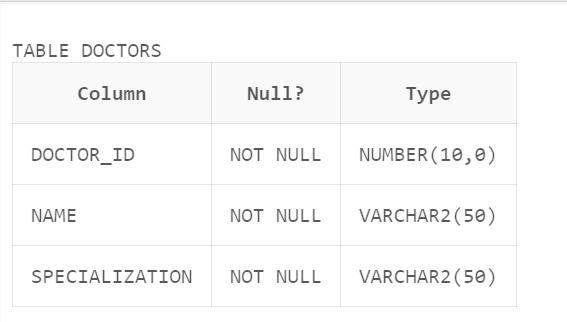
## Supporting\_Staff

|  |  |
| --- | --- |
| staff \_id | name |

∙ Primary Key: staff\_id

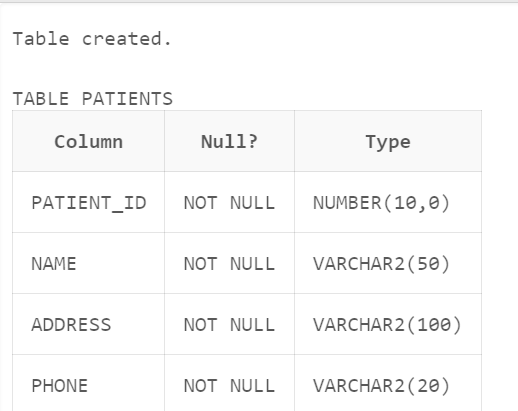
# SQL STATEMENTS FOR TABLE CREATION

CREATE TABLE Doctors ( doctor\_id NUMBER(10) PRIMARY KEY, name VARCHAR2(50) NOT NULL, specialization VARCHAR2(50) NOT NULL);



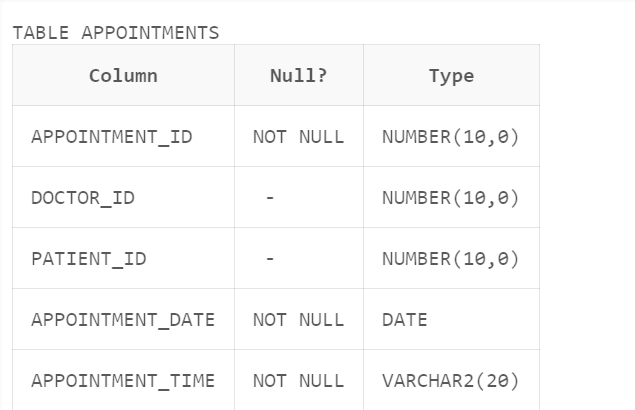
CREATE TABLE Patients ( patient\_id NUMBER(10) PRIMARY KEY, name VARCHAR2(50) NOT NULL, address VARCHAR2(100) NOT NULL, phone VARCHAR2(20) NOT NULL

);



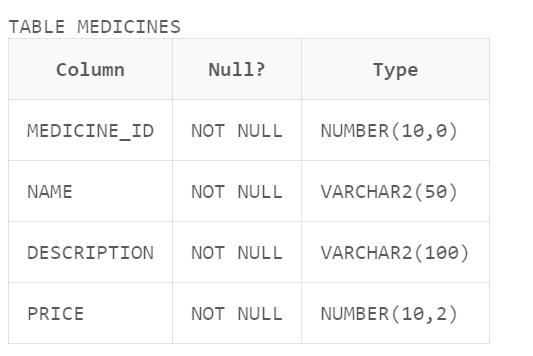
CREATE TABLE Appointments ( appointment\_id NUMBER(10) PRIMARY KEY, doctor\_id NUMBER(10) REFERENCES Doctors(doctor\_id), patient\_id NUMBER(10) REFERENCES Patients(patient\_id), appointment\_date DATE NOT NULL, appointment\_time VARCHAR2(20) NOT NULL

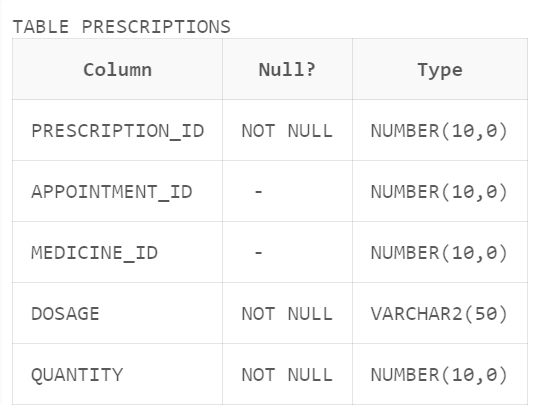
);



CREATE TABLE Medicines ( medicine\_id NUMBER(10) PRIMARY KEY, name VARCHAR2(50) NOT NULL, description VARCHAR2(100) NOT NULL, price NUMBER(10, 2) NOT NULL

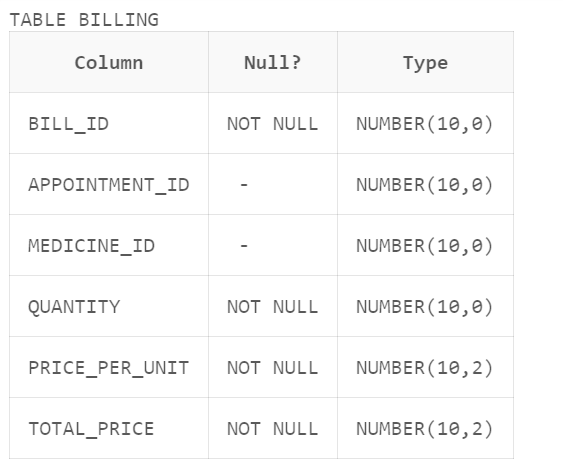
);



CREATE TABLE Prescriptions ( prescription\_id NUMBER(10) PRIMARY KEY, appointment\_id NUMBER(10) REFERENCES Appointments(appointment\_id), medicine\_id NUMBER(10) REFERENCES Medicines(medicine\_id), dosage VARCHAR2(50) NOT NULL, quantity NUMBER(10) NOT NULL); 

CREATE TABLE Billing ( bill\_id NUMBER(10) PRIMARY KEY, appointment\_id NUMBER(10) REFERENCES Appointments(appointment\_id), medicine\_id NUMBER(10) REFERENCES Medicines(medicine\_id), quantity NUMBER(10) NOT NULL, price\_per\_unit NUMBER(10, 2) NOT NULL, total\_price NUMBER(10, 2) NOT NULL

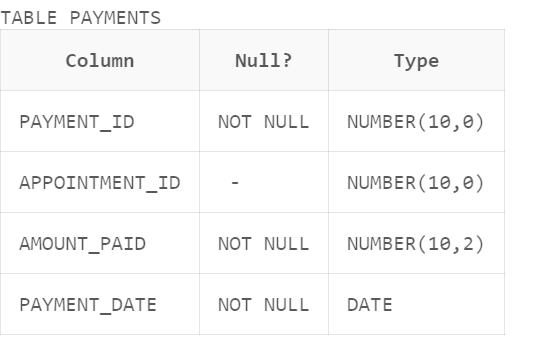
);



CREATE TABLE Payments ( payment\_id NUMBER(10) PRIMARY KEY, appointment\_id NUMBER(10) REFERENCES Appointments(appointment\_id), amount\_paid NUMBER(10, 2) NOT NULL, payment\_date DATE NOT NULL

);

SQL



Create table supporting\_staff(staff\_id number primary key,name varchar(30));



# STATEMENTS FOR INSERT COMMANDS

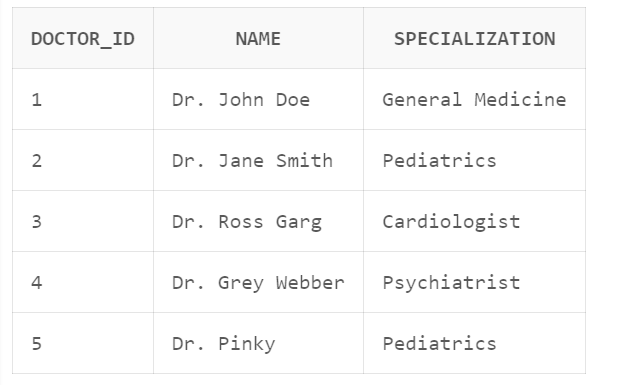
INSERT INTO Doctors (doctor\_id, name, specialization) VALUES (1, 'Dr. John Doe', 'General Medicine');

INSERT INTO Doctors (doctor\_id, name, specialization) VALUES (2, 'Dr. Jane Smith', 'Pediatrics');

INSERT INTO Doctors (doctor\_id, name, specialization) VALUES (3, 'Dr. Ross Garg', 'Cardiologist');

INSERT INTO Doctors (doctor\_id, name, specialization) VALUES (4, 'Dr. Grey Webber', 'Psychiatrist');

INSERT INTO Doctors (doctor\_id, name, specialization) VALUES (5, 'Dr. Pinky', 'Pediatrics');



INSERT INTO Patients (patient\_id, name, address, phone) VALUES (1, 'Smith', '123 Main St, Anytown, USA', '555-1234');

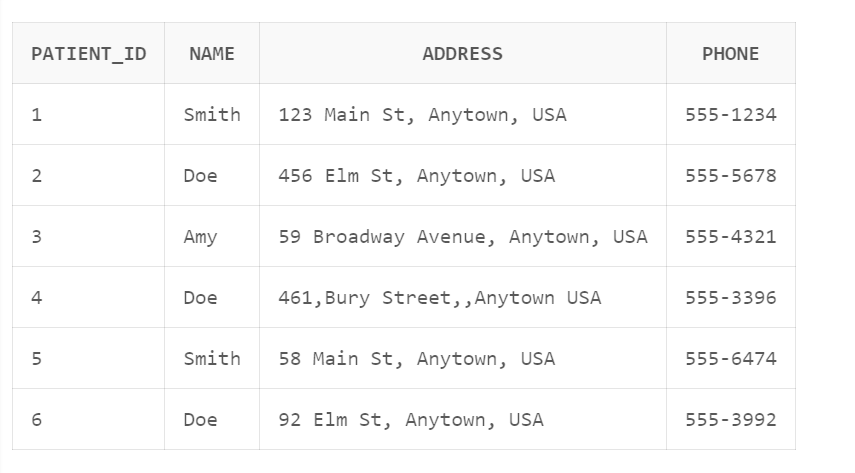
INSERT INTO Patients (patient\_id, name, address, phone) VALUES (2, 'Doe', '456 Elm St, Anytown, USA', '555-5678');

INSERT INTO Patients (patient\_id, name, address, phone) VALUES (3, 'Amy', '59 Broadway Avenue, Anytown, USA', '555-4321');

INSERT INTO Patients (patient\_id, name, address, phone) VALUES (2, 'Doe', '461,Bury Street,,Anytown USA', '555-3396');

INSERT INTO Patients (patient\_id, name, address, phone) VALUES (1, 'Smith', '58 Main St, Anytown, USA', '555-6474');

INSERT INTO Patients (patient\_id, name, address, phone) VALUES (2, 'Doe', '92 Elm St, Anytown, USA', '555-3992');

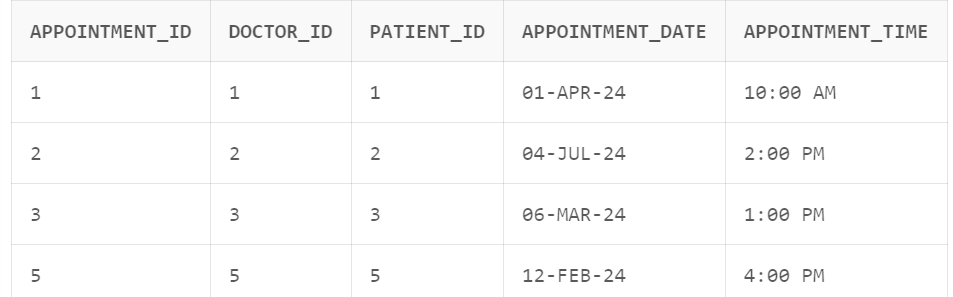


INSERT INTO Appointments (appointment\_id, doctor\_id, patient\_id, appointment\_date, appointment\_time) VALUES (1, 1, 1, TO\_DATE('2024-04-1', 'YYYY-MM-DD'), '10:00 AM');

INSERT INTO Appointments (appointment\_id, doctor\_id, patient\_id, appointment\_date, appointment\_time) VALUES (2, 2, 2, TO\_DATE('2024-07-4', 'YYYY-MM-DD'), '2:00 PM');

INSERT INTO Appointments (appointment\_id, doctor\_id, patient\_id, appointment\_date, appointment\_time) VALUES (3, 3, 3, TO\_DATE('2024-03-6', 'YYYY-MM-DD'), '1:00 PM');

INSERT INTO Appointments (appointment\_id, doctor\_id, patient\_id, appointment\_date, appointment\_time) VALUES (5, 5, 5, TO\_DATE('2024-02-12', 'YYYY-MM-DD'), '4:00 PM');



INSERT INTO Medicines (medicine\_id, name, description, price) VALUES (1, 'Tylenol', 'Pain reliever', 5.99);

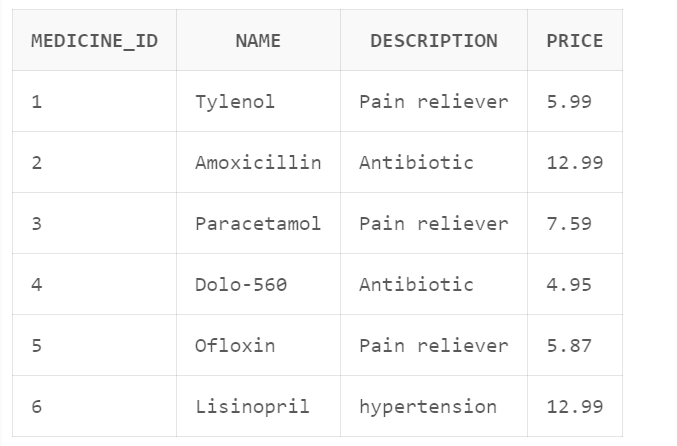
INSERT INTO Medicines (medicine\_id, name, description, price) VALUES (2, 'Amoxicillin', 'Antibiotic',12.99);

INSERT INTO Medicines (medicine\_id, name, description, price) VALUES (3, 'Paracetamol', 'Pain reliever', 7.59);

INSERT INTO Medicines (medicine\_id, name, description, price) VALUES (4, 'Dolo-560', 'Antibiotic',4.95);

INSERT INTO Medicines (medicine\_id, name, description, price) VALUES (5, 'Ofloxin', 'Pain reliever', 5.87);

INSERT INTO Medicines (medicine\_id, name, description, price) VALUES (6, 'Lisinopril','hypertension',12.99);

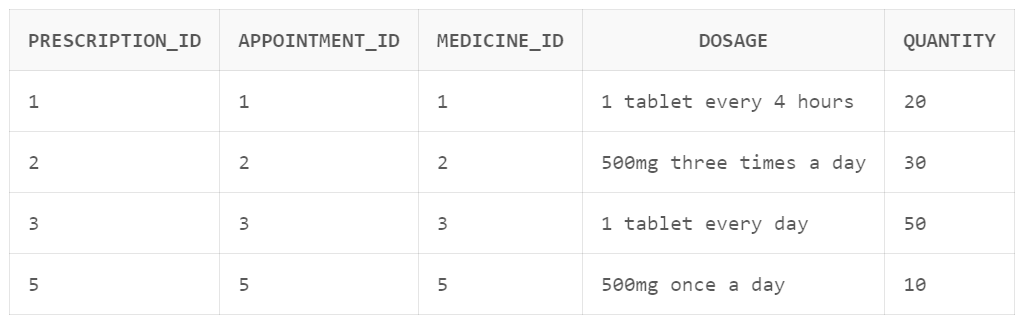


INSERT INTO Prescriptions (prescription\_id, appointment\_id, medicine\_id, dosage, quantity) VALUES (1, 1, 1, '1 tablet every 4 hours', 20);

INSERT INTO Prescriptions (prescription\_id, appointment\_id, medicine\_id, dosage, quantity) VALUES (2,2, 2, '500mg three times a day', 30);

INSERT INTO Prescriptions (prescription\_id, appointment\_id, medicine\_id, dosage, quantity) VALUES (3, 3, 3, '1 tablet every day', 50);

INSERT INTO Prescriptions (prescription\_id, appointment\_id, medicine\_id, dosage, quantity) VALUES (5,5, 5, '500mg once a day', 10);

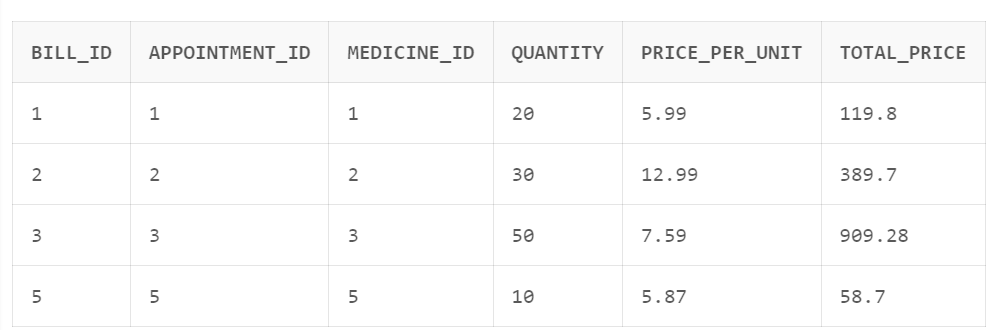


INSERT INTO Billing (bill\_id, appointment\_id, medicine\_id, quantity, price\_per\_unit, total\_price) VALUES (1, 1, 1, 20, 5.99, 119.80);

INSERT INTO Billing (bill\_id, appointment\_id, medicine\_id, quantity, price\_per\_unit, total\_price) VALUES (2, 2, 2, 30, 12.99, 389.70);

INSERT INTO Billing (bill\_id, appointment\_id, medicine\_id, quantity, price\_per\_unit, total\_price) VALUES (3, 3, 3, 50, 7.59, 909.282);

INSERT INTO Billing (bill\_id, appointment\_id, medicine\_id, quantity, price\_per\_unit, total\_price) VALUES (5, 5, 5, 10, 5.87, 58.7);

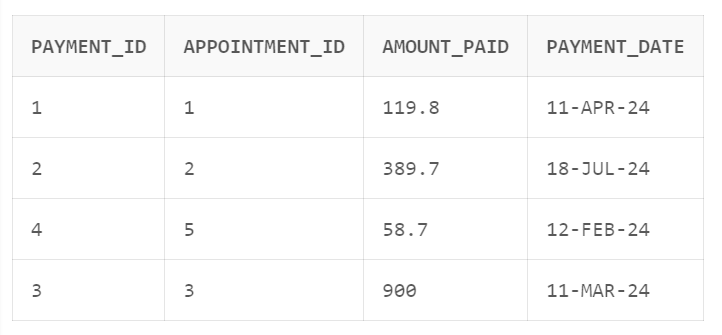


INSERT INTO Payments (payment\_id, appointment\_id, amount\_paid, payment\_date) VALUES (1, 1, 119.80, TO\_DATE('2024-04-11', 'YYYY-MM-DD'));

INSERT INTO Payments (payment\_id, appointment\_id, amount\_paid, payment\_date) VALUES (2, 2, 389.70, TO\_DATE('2024-07-18', 'YYYY-MM-DD'));

INSERT INTO Payments (payment\_id, appointment\_id, amount\_paid, payment\_date) VALUES (3, 3, 900.00, TO\_DATE('2024-03-11', 'YYYY-MM-DD'));

INSERT INTO Payments (payment\_id, appointment\_id, amount\_paid, payment\_date) VALUES (4, 5, 58.7, TO\_DATE('2024-02-12', 'YYYY-MM-DD'));

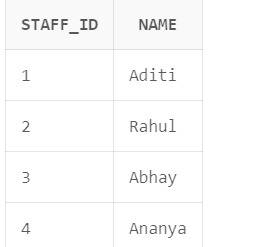


Insert into supporting\_staff values(1,'Aditi');

Insert into supporting\_staff values(2,'Rahul');

Insert into supporting\_staff values(3,'Abhay');

Insert into supporting\_staff values(4,'Ananya');



# PL/SQL CODE

## PROCEDURE

### procedure to delete a record of a patient with id 5

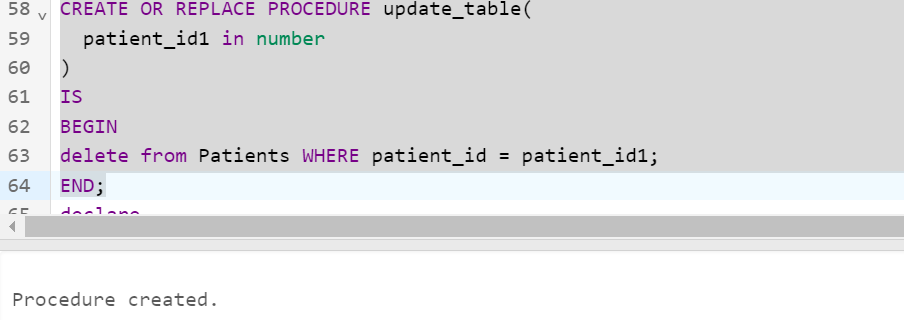
CREATE OR REPLACE PROCEDURE update\_table(patient\_id1 in number)

IS

BEGIN

delete from Patients WHERE patient\_id = patient\_id1;

END;



## CURSOR WITH FUNCTION

### Patient’s name extraction using patient id using cursors

CREATE OR REPLACE FUNCTION get\_patient\_info(patient\_id1 IN NUMBER)

RETURN char

Is

cursor c is select\*from patients;

rec c%rowtype;

name1 char;

begin

for rec in c loop

select name into name1 from patients where patient\_id=patient\_id1;

end loop;

return name1;

END;

Declare

name char;

pno number:=2;

begin

name:=get\_patient\_info(pno);

dbms\_output.put\_line('patient name for id 2 is'||name);

end;

## EXCEPTION HANDLING WITH FUNCTION

## total billing using appointment\_id using exception handling

CREATE OR REPLACE FUNCTION get\_total\_billing\_amount(appointment\_id IN NUMBER)

RETURN NUMBER

IS

total\_amount NUMBER(10,2);

BEGIN

SELECT SUM(total\_price) INTO total\_amount

FROM Billing

WHERE appointment\_id = appointment\_id;

EXCEPTION

When total\_amount IS NULL THEN

Dbms\_output.put\_line( 'No billing information found for the given appointment ID.');

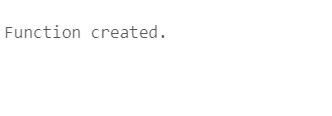
END IF;

RETURN total\_amount;

WHEN OTHERS THEN

Dbms\_output.put\_line ( 'An error occurred while retrieving the billing information: ' );

 END;



## TRIGGER WITH FUNCTION

### updating billing info using triggers

-- create a function to update billing when a prescription is added

CREATE OR REPLACE FUNCTION update\_billing()

RETURN TRIGGER

IS

v\_price number;

BEGIN-- get the medicine price

SELECT price INTO v\_price FROM Medicines

WHERE medicine\_id = NEW.medicine\_id;-- calculate the total price

v\_total\_price := v\_price \* NEW.quantity;-- update the billing table

INSERT INTO Billing (bill\_id, appointment\_id, medicine\_id, quantity, price\_per\_unit, total\_price)

VALUES(bill\_seq.NEXTVAL, NEW.appointment\_id, NEW.medicine\_id, NEW.quantity, v\_price, v\_total\_price);-- return the new row

RETURN NEW;

EXCEPTION

WHEN OTHERS THEN-- handle any exceptions that occur

Dbms\_output.put\_line ('Error updating billing: ' | SQLERRM);

END;-- create a trigger to call the function when a prescription is added

CREATE OR REPLACE TRIGGER prescription\_added

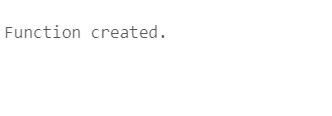
AFTER INSERT ON Prescriptions

FOR EACH ROW

BEGIN

update\_billing();

 END;



# CONCLUSIONS

In conclusion, the doctor-patient management system represents a significant advancement in healthcare technology, offering streamlined management of patient interactions, appointments, prescriptions, and medical records. Its implementation can lead to improved efficiency, accuracy, and communication between healthcare providers and patients, ultimately enhancing the quality of care delivered. By offering customizable solutions and flexible commercial models, such as subscription-based services or one-time purchases, this system can cater to the diverse needs of healthcare organizations and practices. Overall, it stands as a valuable asset in modernizing healthcare delivery and optimizing patient outcomes.

Top of Form

# 

# References

* <https://drive.google.com/file/d/1DABTfDAGois13Zw9hNT9rwY6HE66V-vu/view?usp=sharing>
* <https://www.geeksforgeeks.org/how-to-design-a-database-for-healthcare-management-system/>
* <https://medium.com/@apoorvchowdhry55/hospital-management-system-f21b978a1b8c>
* <https://chat.openai.com/c/ee4fbf42-3ec8-41c6-b17b-bc29d408c58b>
* <https://github.com/topics/hospital-management-system>