
# American International University - Bangladesh (AIUB)

COURSE: ADVANCE DATABASE MANAGEMENT SYSTEM [A]

Summer, 2022-2023

COURSE TEACHER: JUENA AHMED NOSHIN

PROJECT REPORT

## Project Name: Doctor Appointment System

## Group Members:

| Name | ID | Contributions |
|---|---|---|
| MD. SAHILUR RAHMAN | 20-43257-1 | UML Diagram, Project Proposal, Introduction, Schema Diagram, Scenario Description. |
| MD. FAHAD KHAN | 20-43328-1 | Interface, Er Diagram, Project Proposal, introduction, Scenario Description, Final Project Connection. |
| MD. MOSHIUR RAHMAN | 19-41448-3 | Table Creation, Data Insertion, Query Writing, PL/SQL Writing Relational Algebra. |
| PULOK SARKER | 19-41641-3 | Normalization. |

# Contents:

# 1. **Introduction:**

The Doctor Appointment System project aims to revolutionize the way patients schedule appointments with doctors, streamlining the entire process and enhancing the overall efficiency of healthcare facilities. This innovative system will provide a user-friendly platform for patients to conveniently book appointments with their preferred doctors, while also facilitating seamless scheduling and management for medical professionals and administrators. This project seeks to address these challenges by harnessing the power of technology to create a robust, efficient, and user-centric appointment management system.

The Doctor Appointment System will provide a seamless and intuitive interface for patients to search for doctors based on their specialties, availability, and other criteria. Patients will have access to comprehensive doctor profiles that include qualifications, availability, and patient reviews, enabling them to make informed decisions about their healthcare providers. Furthermore, the system will empower patients to book appointments with their chosen doctors effortlessly. Through real-time availability updates, patients can select suitable time slots, receive confirmation, and even get reminders for their appointments via email or SMS. This streamlined approach to appointment scheduling will enhance patient satisfaction and convenience, ultimately improving healthcare outcomes.

For doctors and administrators, the Doctor Appointment System will simplify the process of managing appointments and optimizing resource utilization. Doctors will have an intuitive interface to manage their availability, view patient medical histories, and communicate securely with their patients. Administrators will benefit from a comprehensive dashboard to oversee doctor profiles, monitor appointment schedules, and generate insightful reports for effective decision-making.

By leveraging modern technology, user-friendly interfaces, and efficient scheduling algorithms, the Doctor Appointment System project will transform the way healthcare appointments are managed. It will enhance the patient experience, reduce administrative burden, optimize resource allocation, and improve overall efficiency in healthcare facilities.

In summary, the Doctor Appointment System project seeks to bridge the gap between patients and healthcare providers, creating a seamless and efficient appointment management system. Through this project, we aim to enhance patient satisfaction, streamline administrative processes, and contribute to a more patient-centric and effective healthcare ecosystem.

# 2. Project Proposal:

## 1. Introduction:

The Doctor Appointment System is an innovative project aimed at streamlining the process of scheduling and managing appointments between doctors and patients. This system will provide a user-friendly interface for patients to conveniently book appointments with their preferred doctors, while also facilitating efficient scheduling and management for medical professionals and administrators.

## 2. Objectives:

- Develop a user-friendly desktop-based platform for patients to book appointments with doctors.
- Implement an intuitive and efficient scheduling system for doctors and administrators.
- Improve the overall efficiency of the appointment management process, reducing wait times and enhancing patient satisfaction.
- Ensure data security and privacy to protect sensitive medical information.
- Provide an accessible and scalable solution that can accommodate a growing number of users and healthcare providers.

## 3. Features:

### a. Patient Interface:

- login functionality for patients.
- Search functionality to find doctors based on specialty, availability, etc. View doctor profiles, including qualifications, availability, and view system-based prescription and reports.
- Book appointments with selected doctors, considering available time slots.
- Patient see their bill automatically.

### b. Doctor Interface:

- login functionality for doctors.
- View appointments.
- Create Prescription and add medicines.
- Access patient medical history and previous visit information.

### c. Admin Interface:

- User registration and login functionality for administrators.
- Manage doctor profiles, including adding, editing, or removing doctors.
- Monitor and manage the appointments, ensuring optimal utilization of resources.

- Create Report for patients.
- Make salary for doctor.
- Make bills for patients.

### d. User Interface

- User can make their appointments.
- User can search doctor.

## 4. Software:

The Doctor Appointment System will be developed using a modern technology stack, including but not limited to:

- Oracle 10g
- Visual Code Studio 2022
- Draw.io
- C#

## 5. Project Timeline:

The project will be divided into the following phases:

- Requirement gathering and analysis.
- System design.
- Integration and testing.
- Deployment and user acceptance testing.
- Maintenance and support.

## 6. Budget:

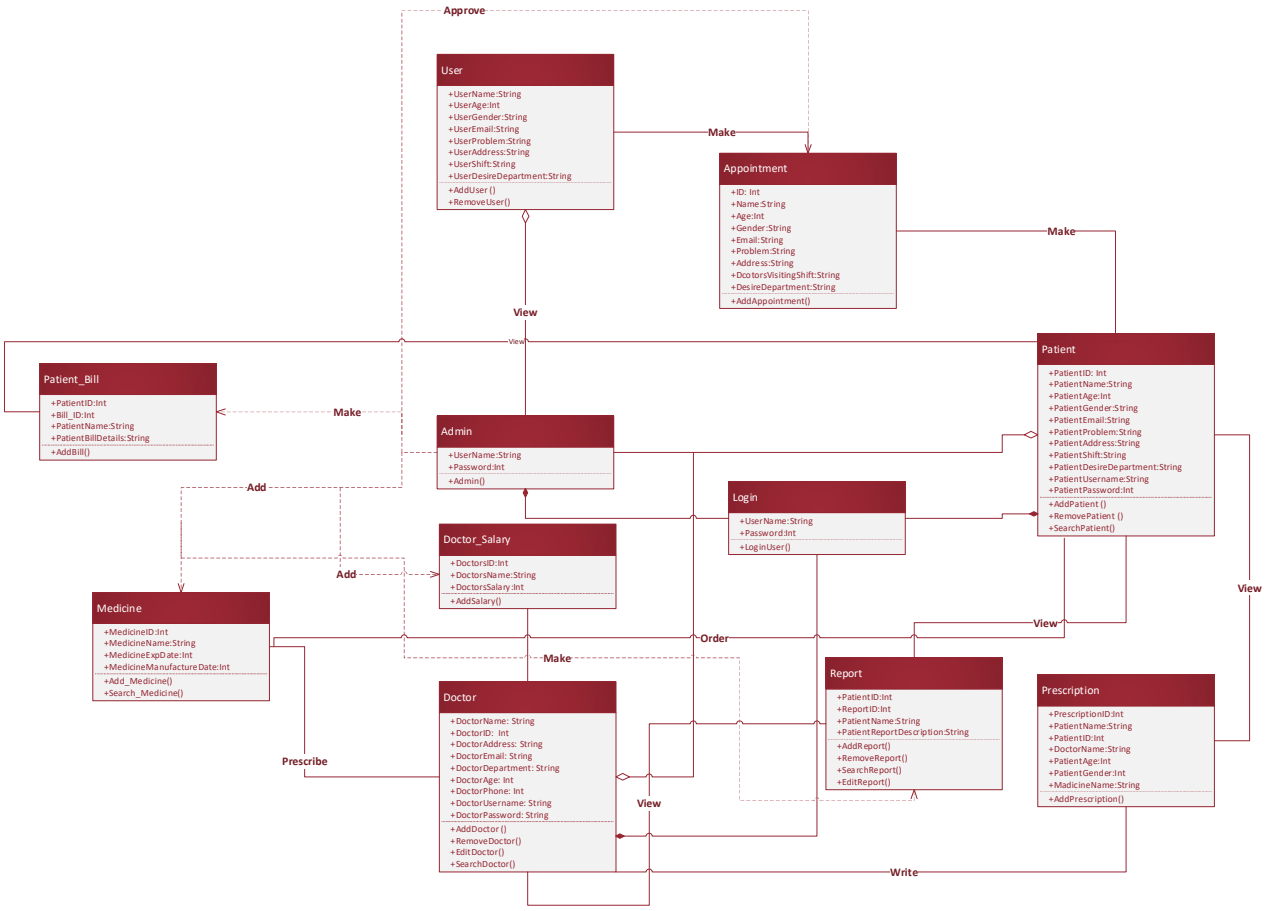
The project budget will include costs associated with development, testing, deployment, infrastructure, and ongoing maintenance and support. A detailed budget breakdown will be provided after the requirements analysis phase.

## 7. Conclusion:

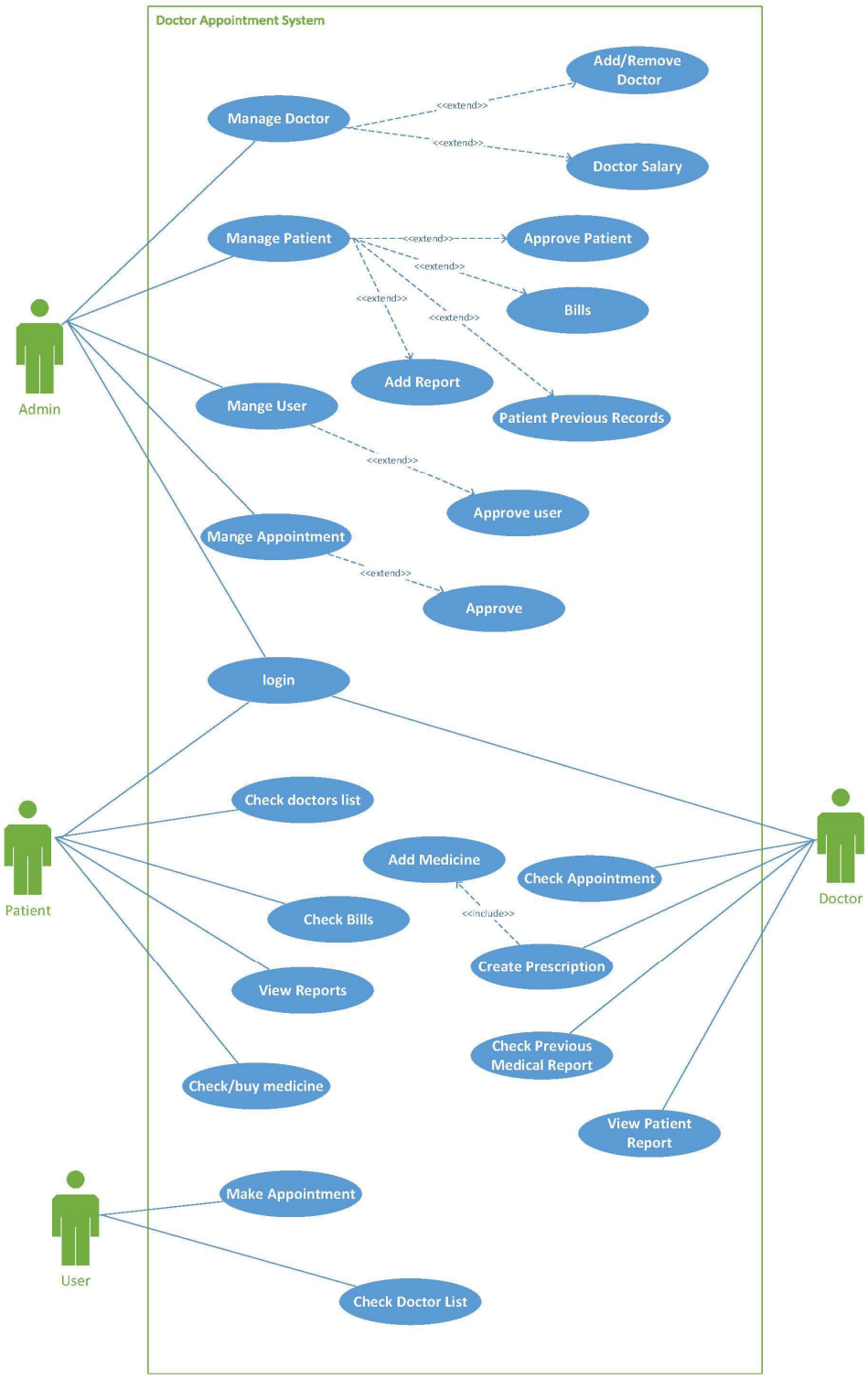
The Doctor Appointment System aims to revolutionize the way patients book appointments and streamline the appointment management process for doctors and administrators. By implementing this system, we will enhance patient experience, improve healthcare resource utilization, and increase overall efficiency in healthcare facilities.

# 3. UML Diagrams:

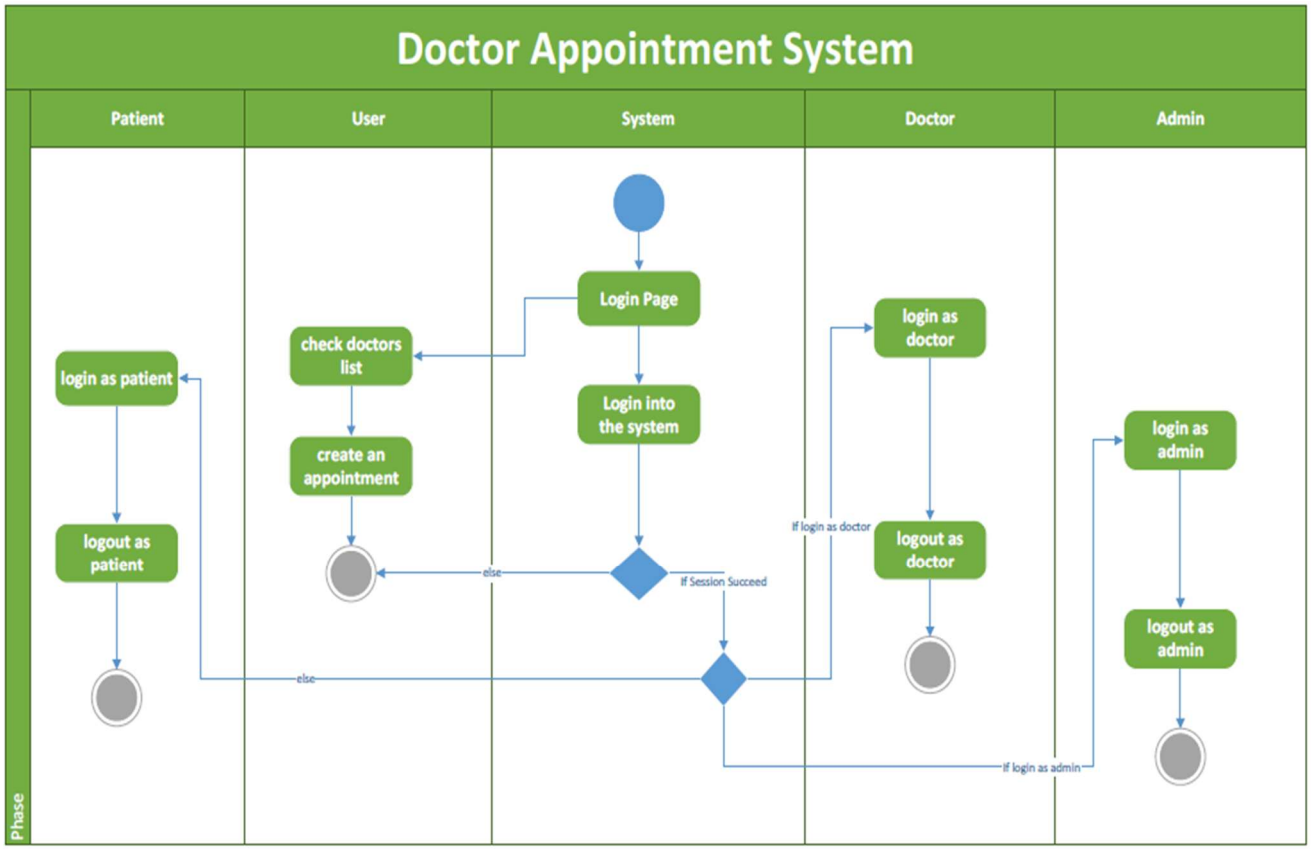
## 1. Class Diagram



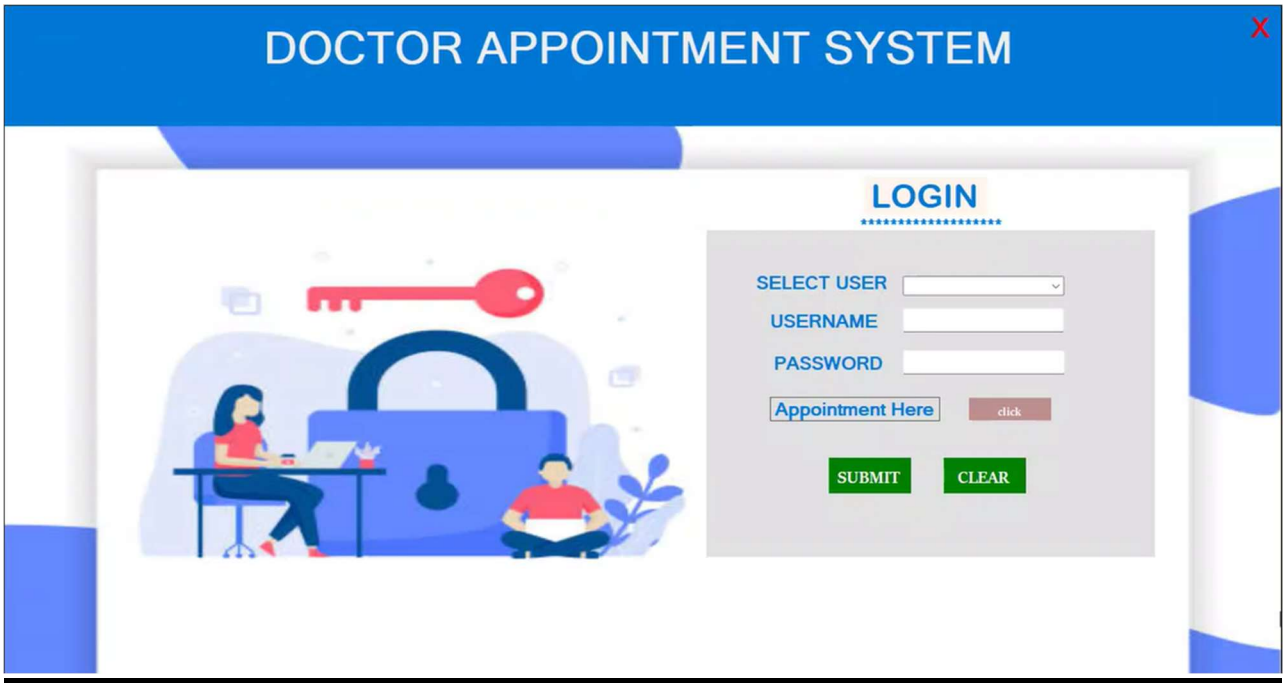
# 2. Use Case Diagram

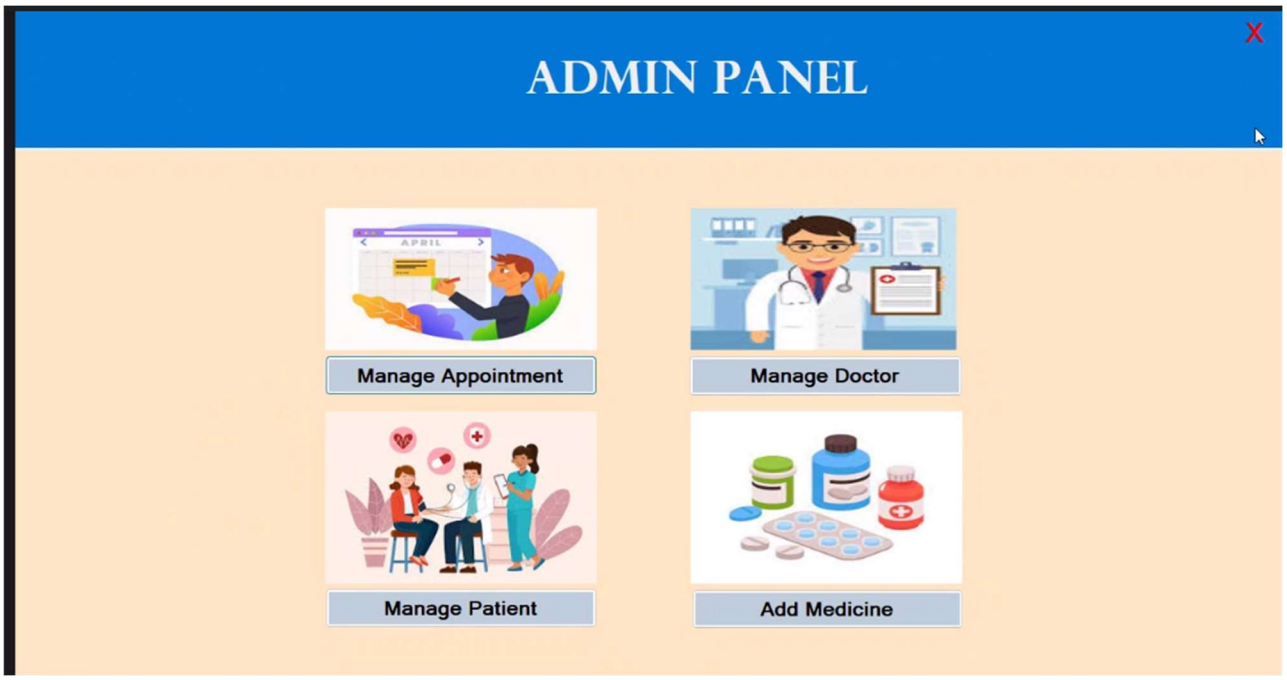


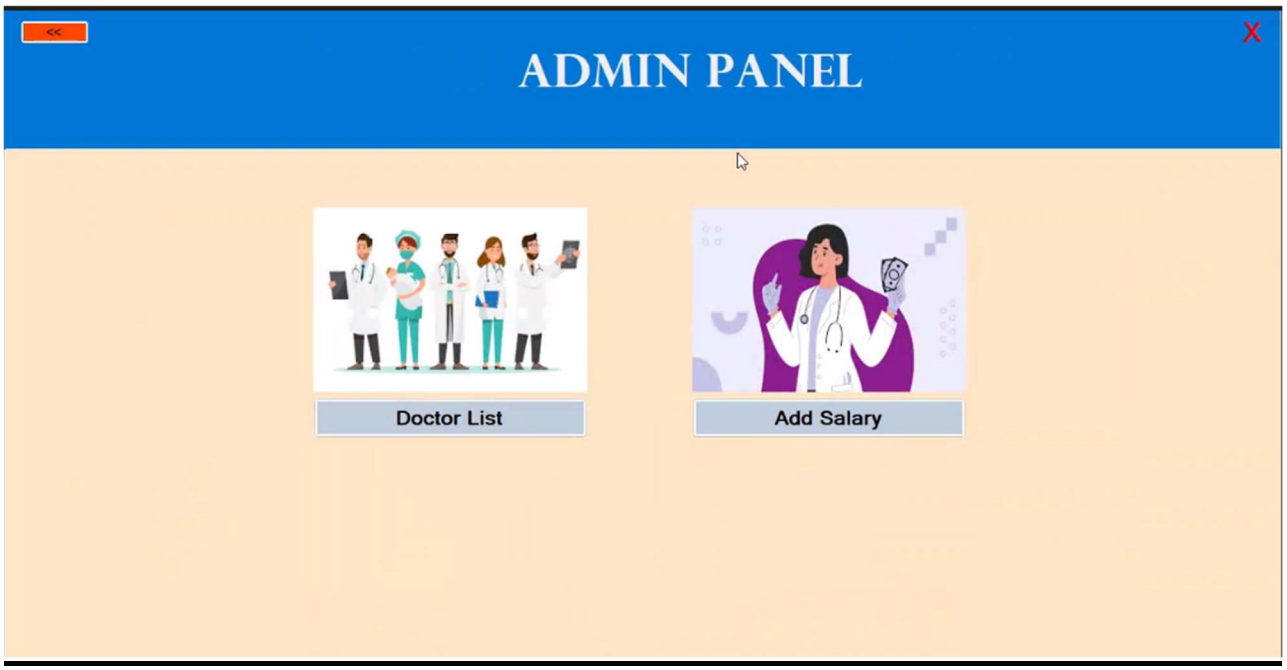
# 3. Activity Diagram

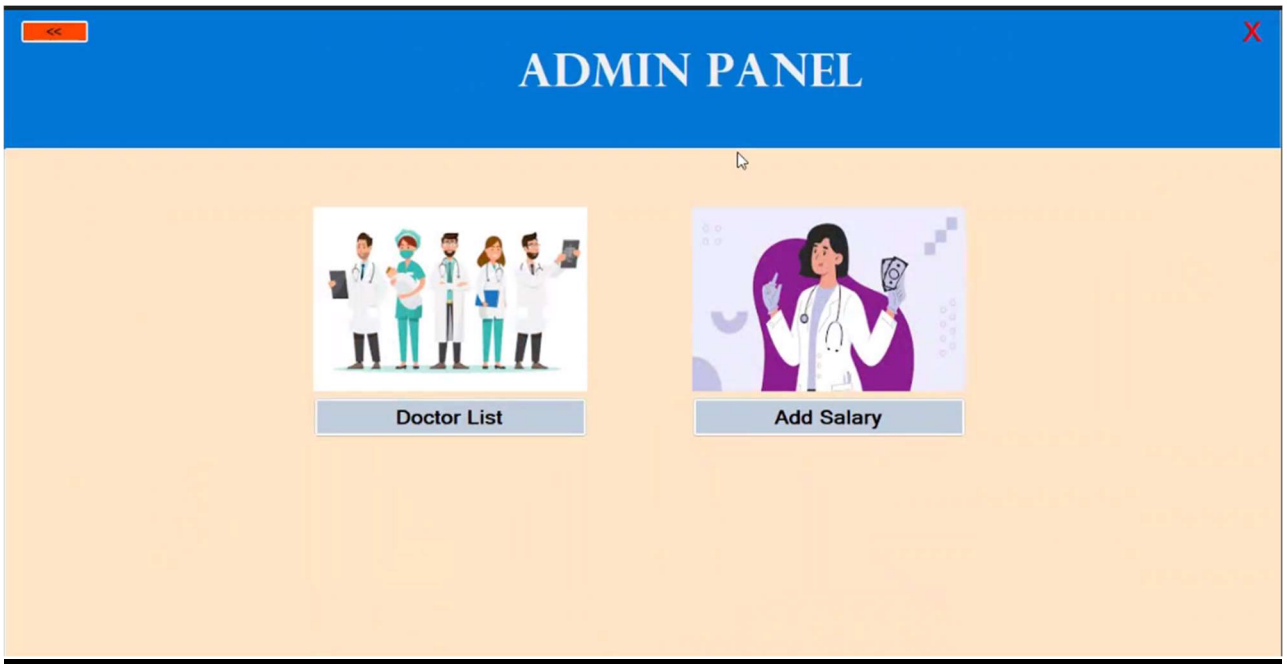



# 4. User Interface

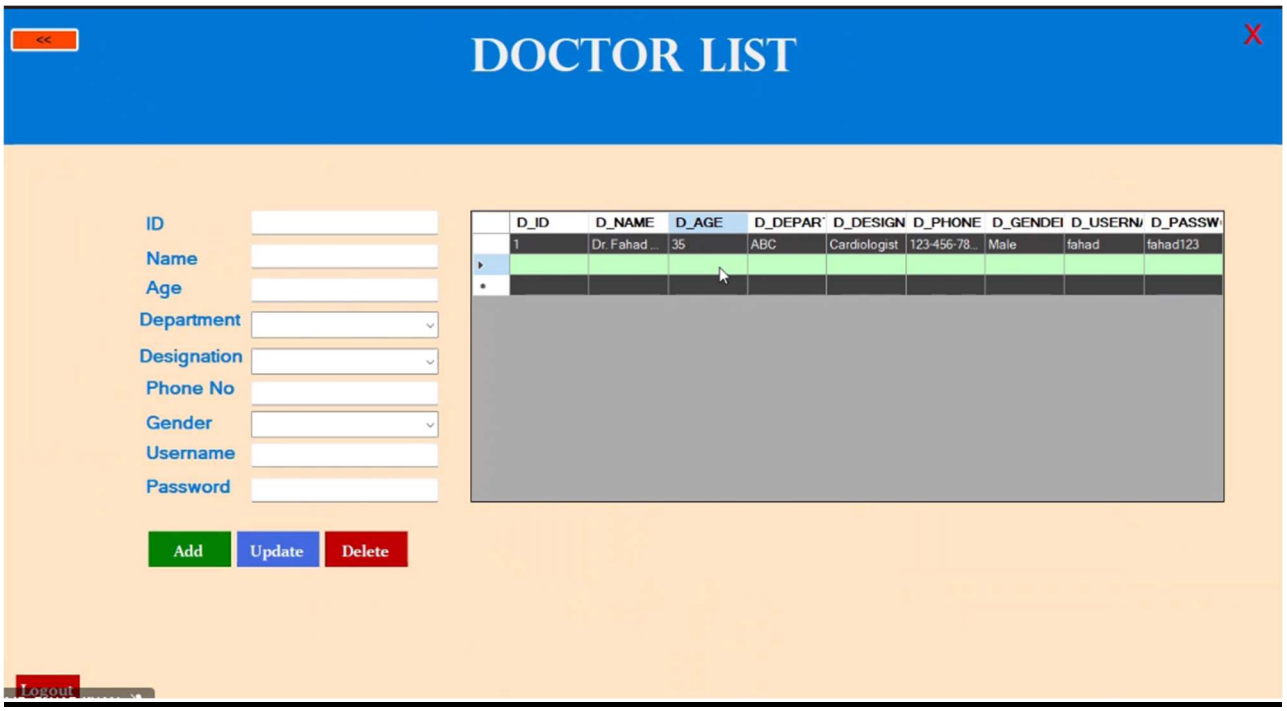


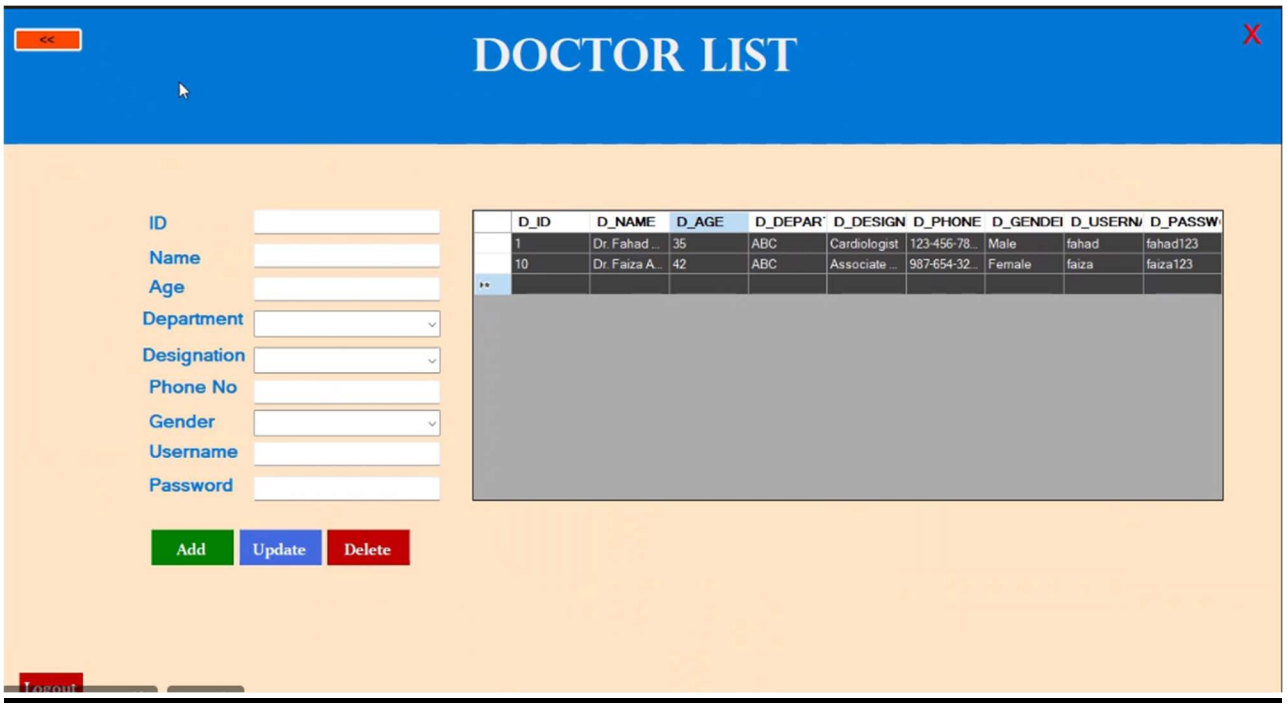

ADMIN PANEL
X
APRIL
Manage Appointment
Manage Doctor
Manage Patient
Add Medicine

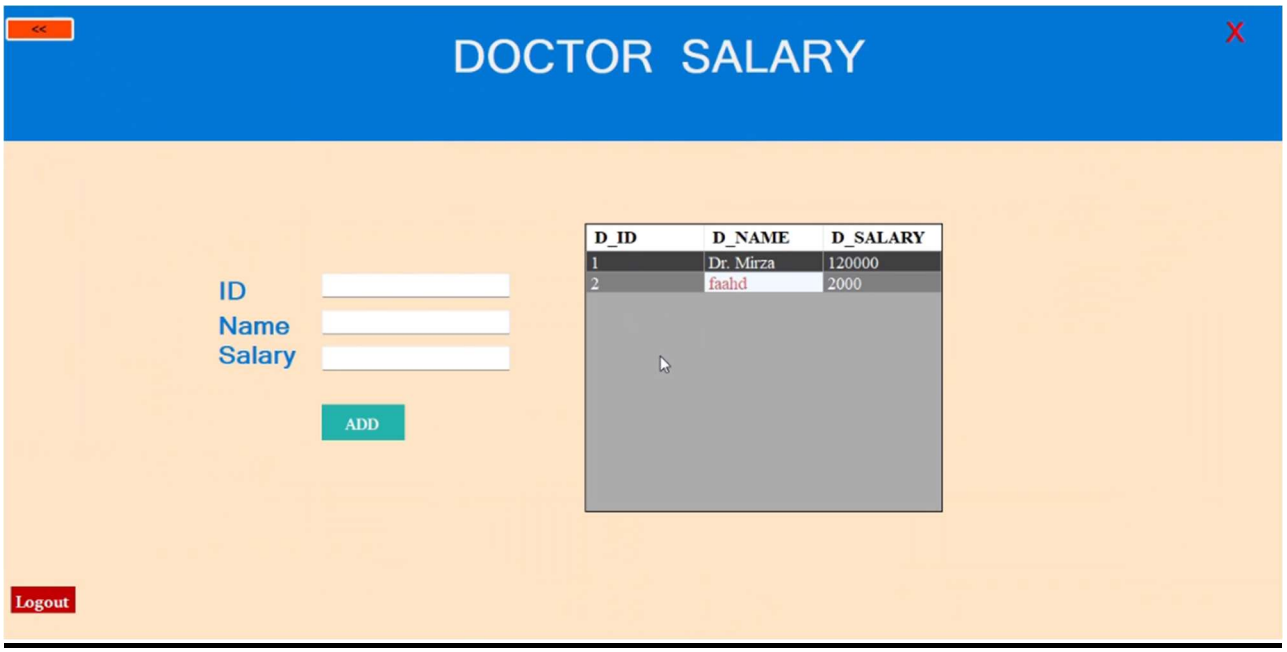

<<
ADMIN PANEL
X
Doctor List
Add Salary

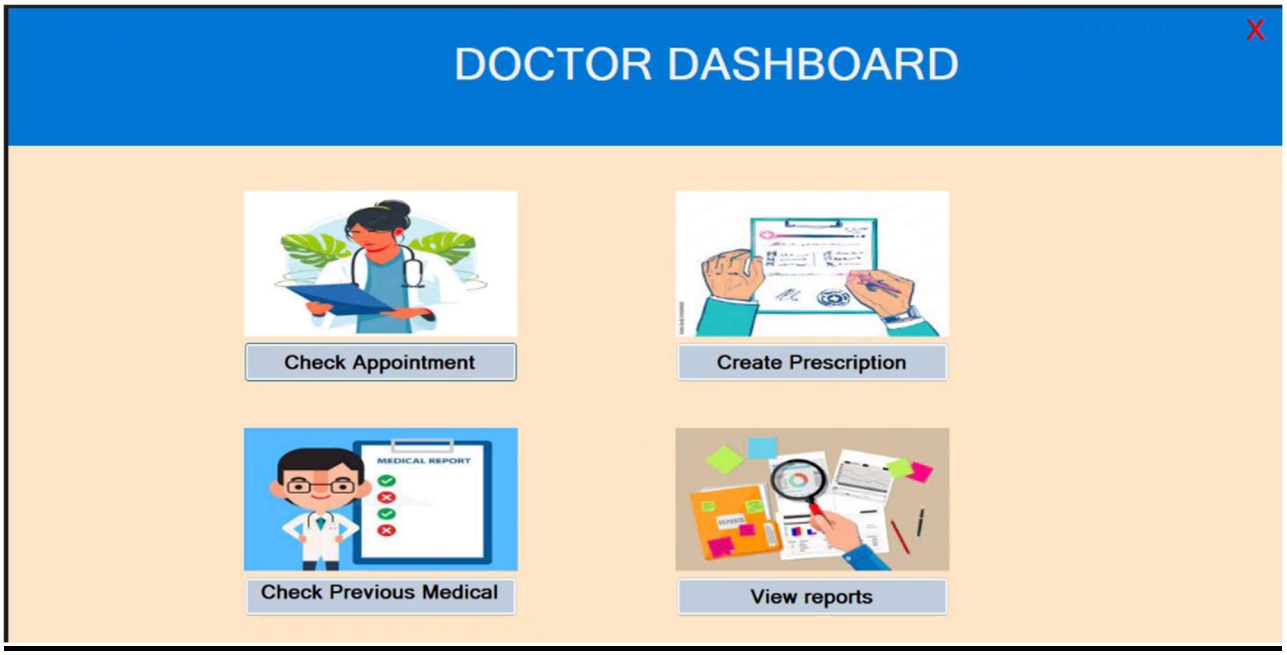

<<
X
ADMIN PANEL
Doctor List
Add Salary

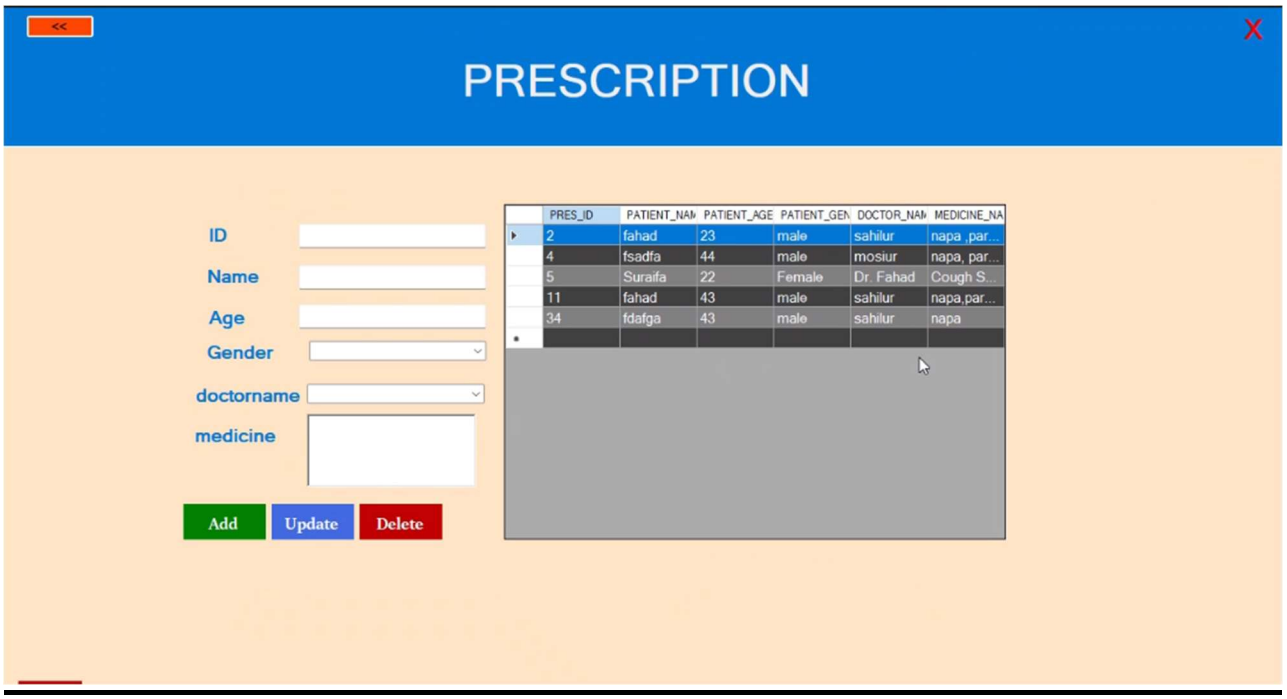

<<
X
APPOINTMENTS
<<
A_ID A_PNAME A_EMAIL A_ADDRESS A_DPT A_PHONE A_PROBLEM A_SHIFT A_AGE A_GENDER
1 Khurshed Alam alam67@gmail.com kuril dhaka 555-123-4567 Chest Pain Morning 45 Male
2 Rahim rahim@yahoo.... mirpur dhaka 555-987-6543 Fractured Arm Afternoon 28 Male
3 Adrita Islam adrita@gmail.com bailyroad dhaka 555-555-7890 Rash Morning 8 Female
4 Sakib Hassan hassan@gmail... dhanmondi dh... 555-987-6543 Fever Afternoon 6 Male
5 Alex Hassan alex@hotmail.com motijhil dhaka 555-555-1111 Flu Morning 37 Male
Accept
Reject

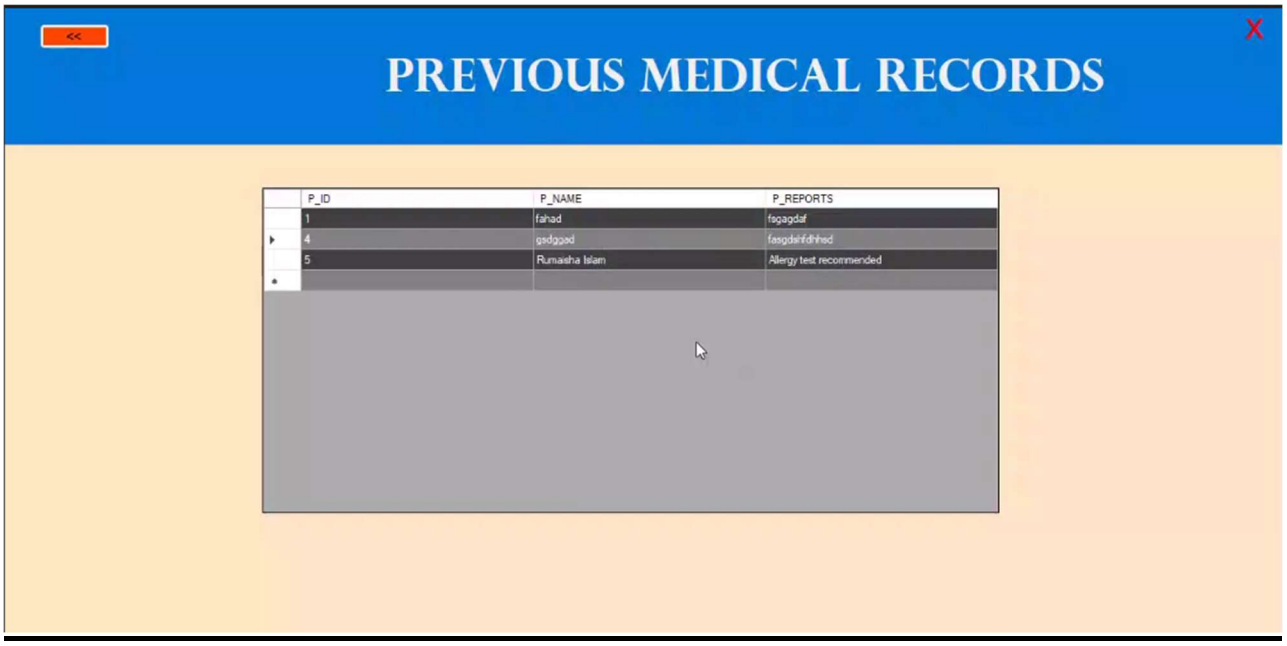

DOCTOR LIST
ID
Name
Age
Department
Designation
Phone No
Gender
Username
Password
Add
Update
Delete
Logout
D_ID D_NAME D_AGE D_DEPAR D_DESIGN D_PHONE D_GENDEI D_USERNA D_PASSW
1 Dr. Fahad ... 35 ABC Cardiologist 123-456-78... Male fahad fahad123

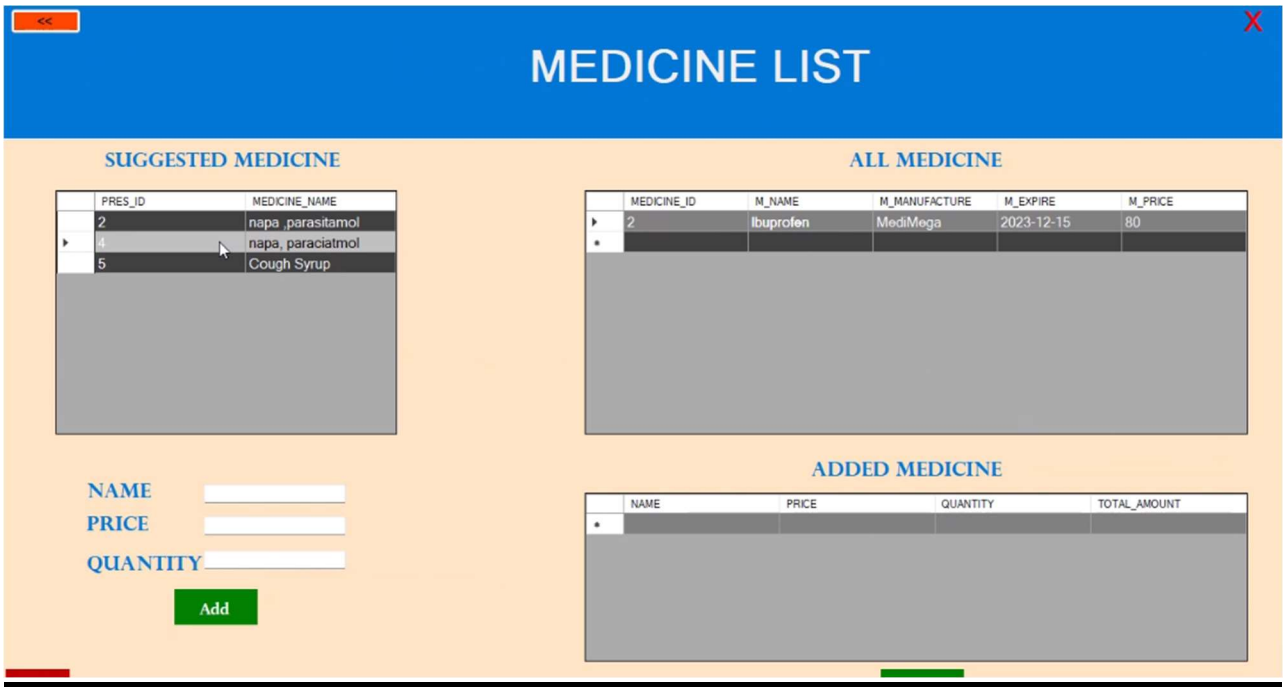

DOCTOR LIST
ID
Name
Age
Department
Designation
Phone No
Gender
Username
Password
Add
Update
Delete
Logout
D_ID D_NAME D_AGE D_DEPAR D_DESIGN D_PHONE D_GENDEI D_USERNA D_PASSW
1 Dr. Fahad ... 35 ABC Cardiologist 123-456-78... Male fahad fahad123
10 Dr. Faiza A... 42 ABC Associate ... 987-654-32... Female faiza faiza123

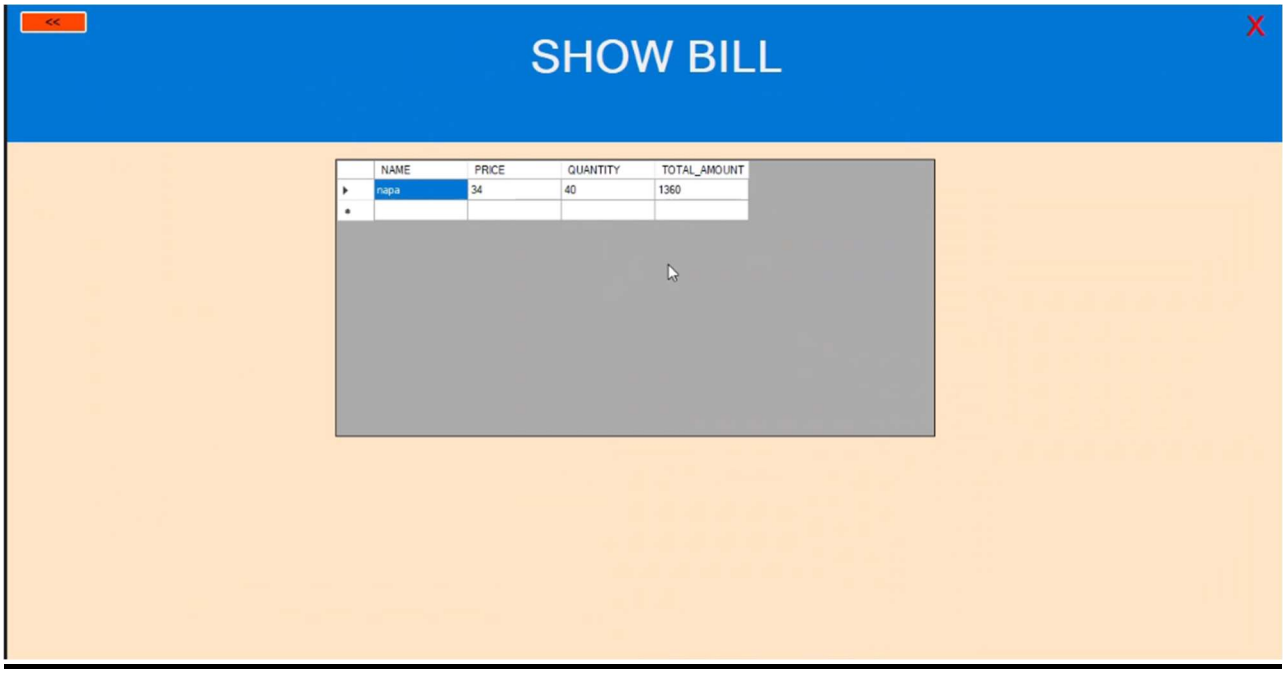

<<
DOCTOR SALARY
X
ID
Name
Salary
ADD
D_ID
D_NAME
D_SALARY
1
Dr. Mirza
120000
2
faahd
2000
Logout


DOCTOR DASHBOARD
X
Check Appointment
Create Prescription
MEDICAL REPORT
Check Previous Medical
View reports


<<
X
PRESCRIPTION
ID
Name
Age
Gender
doctorname
medicine
Add
Update
Delete
PRES_ID PATIENT_NAM PATIENT_AGE PATIENT_GEN DOCTOR_NAM MEDICINE_NA
2 fahad 23 male sahilur napa ,par...
4 fsadfa 44 male mosiur napa, par...
5 Suraifa 22 Female Dr. Fahad Cough S...
11 fahad 43 male sahilur napa,par...
34 fdafga 43 male sahilur napa


<<
X
PREVIOUS MEDICAL RECORDS
P_ID P_NAME P_REPORTS
1 fahad fagagdaf
4 qsdgqad fasgdslnfdhlnsd
5 Rumaisha Islam Allergy test recommended

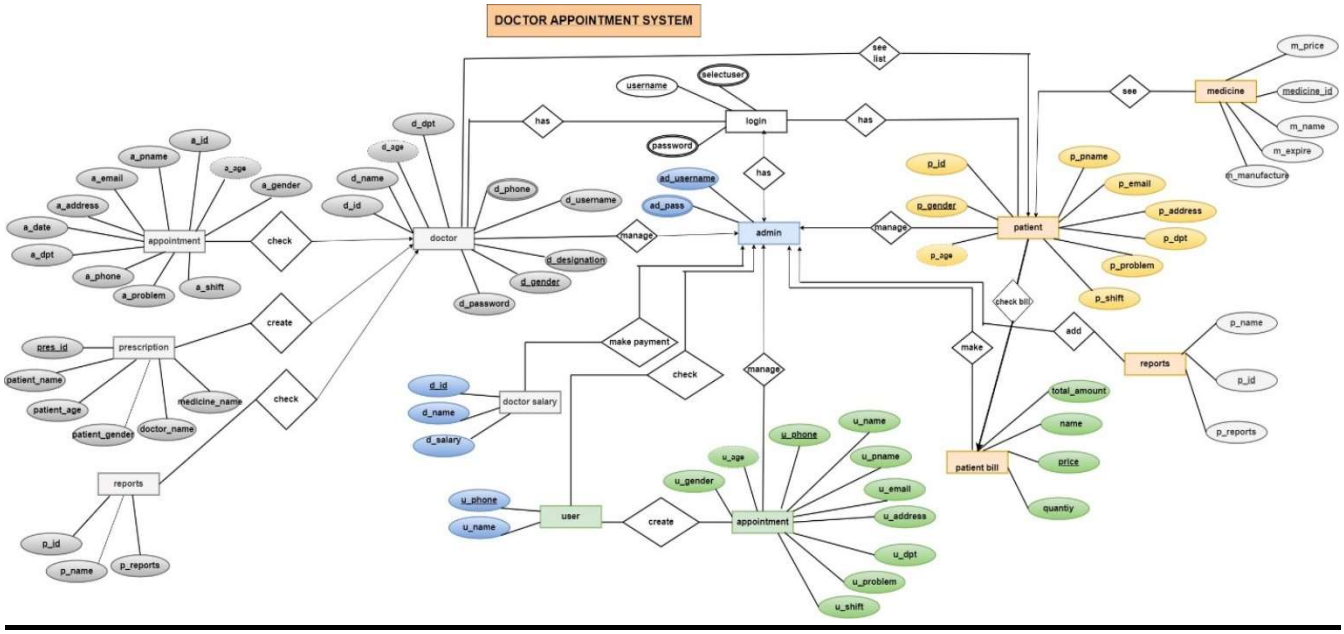

MEDICINE LIST
SUGGESTED MEDICINE
PRES_ID
MEDICINE_NAME
2
napa ,parasitamol
napa, paraciatmol
5
Cough Syrup
ALL MEDICINE
MEDICINE_ID
M_NAME
M_MANUFACTURE
M_EXPIRE
M_PRICE
2
Ibuprofen
MediMega
2023-12-15
80
NAME
PRICE
QUANTITY
Add
ADDED MEDICINE
NAME
PRICE
QUANTITY
TOTAL_AMOUNT


SHOW BILL
NAME
PRICE
QUANTITY
TOTAL_AMOUNT
napa
34
40
1360

# 5. Scenario Description

In a doctor appointment system, there are four types of users: admin, patient, doctor, and regular user. Admin manages all the users within the system. Users can make appointments and view the list of available doctors. Users can create appointments using their usernames and phone numbers. Each patient is identified by a unique ID. When a user makes an appointment, they are placed in the waiting list. Once the admin approves the appointment, the user becomes a patient. The admin also has the authority to reject appointments. Patient details include their name, email, gender, age, medical issues, and address. Patients can access the list of available doctors, as well as view the available medicine list. The system allows patients to access their submitted reports. Upon purchasing medicine, patients can view the total bill and print a receipt. Doctors are assigned unique IDs and their information includes name, designation, age, department, phone number, email, and gender. Doctors can view patient details and reports submitted by the admin. They can manage patient lists, write prescriptions, and add medicines to those prescriptions. Each prescription is identified by a unique prescription ID. The admin has assigned username and password. Admins are responsible for creating, updating, and deleting doctor profiles. They also handle salary management for doctors. Patients can purchase medicine from a suggested list provided by the doctor. The selected medicines contribute to an automatically calculated total. Patients have the option to print a receipt if desired.

# 6. ER Diagram



# 7. Normalization

## Doctor

**Check-Appointment**

UNF

a_id, a_pname, a_email, a_address, a_data, a_department, a_phone, a_problem, a_shift, a_age, a_gender, d_id, d_name, d_age, d_dpt, d_phone, d_username,d_department, d_designation, d_gender, d_password

1NF

a_id (PK), a_pname, a_email, a_address, a_data, a_phone, a_problem, a_shift, a_age, a_gender (PK), d_name, d_age, d_dpt, d_phone, d_username,d_department, d_designation, d_gender, d_password

2NF

a_id (PK), a_pname, a_email, a_address, a_phone, a_problem, a_shift, a_age, a_gender,

d_id (PK), d_name, d_age, d_phone, d_username,d_department, d_designation, d_gender, d_password,

3NF

a_id (PK), a_pname, a_email, a_address, a_phone, a_problem, a_shift, a_age, a_gender, d_id (FK)

d_id (PK), d_name, d_age, d_phone, d_username,d_department, d_designation, d_gender, d_password,

**Create- Prescrption**

UNF

pres_id, patient_name, patient_age, medicine_name, patient_gender, doctor_name, d_id, d_name, d_age, d_phone, d_username,d_department, d_designation, d_gender, d_password,

1NF

pres_id(PK), patient_name, patient_age, medicine_name, patient_gender, doctor_name, d_id(PK), d_name, d_age, d_phone, d_username,d_department, d_designation, d_gender, d_password,

2NF

pres_id(PK), patient_name, patient_age, medicine_name, patient_gender, doctor_name,

d_id(PK), d_name, d_age, d_phone, d_username,d_department, d_designation, d_gender, d_password,

3NF

pres_id(PK), patient_name, patient_age, medicine_name, patient_gender, doctor_name,

d_id(PK), d_name, d_age, d_phone, d_username,d_department, d_designation, d_gender, d_password,

**Check-Report**

UNF

p_id, p_name, p_report d_id, d_name, d_age, d_dpt, d_phone, d_username,d_department, d_designation, d_gender, d_password

1NF

p_id (PK), p_name, p_report, d_id (PK), d_name, d_age, d_dpt, d_phone, d_username,d_department, d_designation, d_gender, d_password

2NF

p_id (PK), p_name

d_id (PK), d_name, d_age, d_dpt, d_phone, d_username,d_department, d_designation, d_gender, d_password

3NF

p_id (PK), p_name

d_id (PK), d_name, d_age, d_phone, d_username,d_department, d_designation, d_gender, d_password

# Patient

**list view- Patient view doctor list**

UNF

p_id, p_pname, p_email, p_address, p_data, p_department, p_phone, p_problem, p_shift, p_age, p_gender, d_id, d_name, d_age, d_dpt, d_phone, d_username,d_department, d_designation, d_gender, d_password

1NF

p_id (PK), p_pname, p_email, p_address, p_data, p_phone, p_problem, p_shift, p_age, p_gender, d_id (PK), d_name, d_age, d_dpt, d_phone, d_username,d_department, d_designation, d_gender, d_password

2NF

p_id (PK), p_pname, p_email, p_address, p_phone, p_problem, p_shift, p_age, p_gender

d_id (PK), d_name, d_age, d_phone, d_username,d_department, d_designation, d_gender, d_password,

3NF

p_id (PK), p_pname, p_email, p_address, p_phone,p_shift, p_problem, p_age, p_gender,

d_id (PK), d_name, d_age, d_phone, d_username,d-department, d_designation, d_gender, d_password,

**see-medicine**

1.UNF

medicine_id, m_name, m_manufacture, m_expire, m_price, p_id, p_pname, p_email, p_address, p_data, p_department, p_phone, p_problem, p_shift, p_age, p_gender

2. 1NF

medicine_id(PK), m_name, m_manufacture, m_expire, m_price, p_id(PK), p_pname, p_email, p_address, p_data, p_department, p_phone, p_problem, p_shift, p_age, p_gender

2NF

medicine_id(PK), m_name, m_manufacture, m_expire, m_price,

p_id(PK), p_pname, p_email, p_address, p_data, p_department, p_phone, p_problem, p_shift, p_age, p_gender

3NF

medicine_id(PK), m_name, m_manufacture, m_expire, m_price

p_id(PK), p_pname, p_email, p_address, p_data, p_department,p_shift, p_phone, p_problem, p_age, p_gender,

**Check (Report)**

UNF

p_id, p_name, p_reports, p_id, p_pname, p_email, p_address, p_data, p_department, p_phone, p_problem, p_shift, p_age, p_gender

1NF:

p_report, p_id(PK), p_pname, p_email, p_address, p_data, p_department, p_phone, p_problem, p_shift, p_age, p_gender

2NF

p_id(PK), p_pname, p_email, p_address, p_data, p_department, p_phone, p_problem, p_shift, p_age, p_gender

p_report, r_id(PK),

3NF

p_id(PK), p_pname, p_email, p_address, p_data, p_department,p_shift, p_phone, p_problem, p_age, p_gender,

p_report, r_id(PK)

**check Bill**

UNF

Total_amount,name, quantity,price, p_id, p_pname, p_email, p_address, p_data, p_department, p_phone, p_problem, p_shift, p_age, p_gender

1NF

Total_amount,name, quantity,price, p_id(PK), p_pname, p_email, p_address, p_data, p_department, p_phone, p_problem, p_shift, p_age, p_gender

2NF

Total_amount,name, quantity,price

p_id(PK), p_pname, p_email, p_address, p_data, p_department, p_phone, p_problem, p_shift, p_age, p_gender

3NF

Total_amount,name, quantity,price

p_id, p_pname, p_email, p_address, p_data, p_department, p_shift, p_phone, p_problem, p_age, p_gender

## Admin

**Manage-Doctor**

UNF

d_id, d_name, d_age, d_dpt, d_phone, d_username,d_department, d_designation, d_gender, d_password, ad_username, ad_pass, ad_id

1NF

d_id(PK), d_name, d_age, d_dpt, d_phone, d_username, d_department, d_designation, d_gender, d_password, ad_username, ad_pass, ad_id(PK)

2NF

d_id(PK), d_name, d_age, d_dpt, d_phone, d_username, d_department, d_designation, d_gender, d_password

ad_username, ad_pass, ad_id(PK)

3NF

d_id(PK), d_name, d_age, d_phone, d_username, d_department, d_designation, d_gender, d_password

ad_username, ad_pass, ad_id(PK)

**Manage-Patient**

UNF

p_id, p_pname, p_email, p_address, p_data, p_department, p_phone, p_problem, p_shift, p_age, p_gender, ad_username, ad_pass, ad_id

1NF

p_id(PK), p_pname, p_email, p_address, p_data, p_department, p_phone, p_problem, p_shift, p_age, p_gender, ad_username, ad_pass, ad_id(PK)

2NF

p_id(PK), p_pname, p_email, p_address, p_data, p_department, p_phone, p_problem, p_shift, p_age, p_gender

ad_username, ad_pass, ad_id(PK)

3NF

p_id(PK), p_pname, p_email, p_address, p_data, p_department,p_shift, p_phone, p_problem, p_age, p_gender

ad_username, ad_pass, ad_id(PK)

Manage Appointment

UNF

u_id, u_pname, u_email, u_address, u_data, u_department, u_phone, u_problem, u_shift, u_age, u_gender, ad_username, ad_pass, ad_id

1NF

u_id(PK), u_pname, u_email, u_address, u_data, u_department, u_phone, u_problem, u_shift, u_age, u_gender, ad_username, ad_pass, ad_id(PK)

2NF

u_id(PK), u_pname, u_email, u_address, u_data, u_department, u_phone, u_problem, u_shift, u_age, u_gender,

ad_username, ad_pass, ad_id(PK)

3NF

u_id(PK), u_pname, u_email, u_address, u_data, u_department, u_phone,p_shift, u_problem, u_age, u_gender,

ad_username, ad_pass, ad_id(PK)

**Add Report**

UNF

ad_username, ad_pass, ad_id, p_name, p_id, p_reports

1NF

ad_username, ad_pass, ad_id(PK), p_name, p_id(PK), p_reports

2NF

ad_username, ad_pass, ad_id(PK)

p_name, p_id(PK), p_reports

3NF

ad_username, ad_pass, ad_id(PK)

p_name, p_id, p_reports

**Make bills**

UNF

ad_username, ad_pass, ad_id, total_amount,name, quantity,price

1NF

ad_username, ad_pass, ad_id(PK), total_amount,name, quantity,price

2NF

ad_username, ad_pass, ad_id(PK)

total_amount,name, quantity,price

3NF

ad_username, ad_pass, ad_id(PK)

total_amount,name, quantity,price

**Check User**

UNF

u_phone, u_name, ad_username, ad_pass, ad_id

1NF

u_phone, u_name, ad_username, ad_pass, ad_id(PK)

2NF

u_phone, u_name

ad_username, ad_pass, ad_id(PK)

3NF

u_phone, u_name

ad_username, ad_pass, ad_id(PK)

**make payment-doctor**

UNF

d_id, d_name, d_salary, ad_username, ad_pass, ad_id

1NF

d_id(PK), d_name, d_salary, ad_username, ad_pass, ad_id(PK)

2NF

d_id(PK), d_name, d_salary

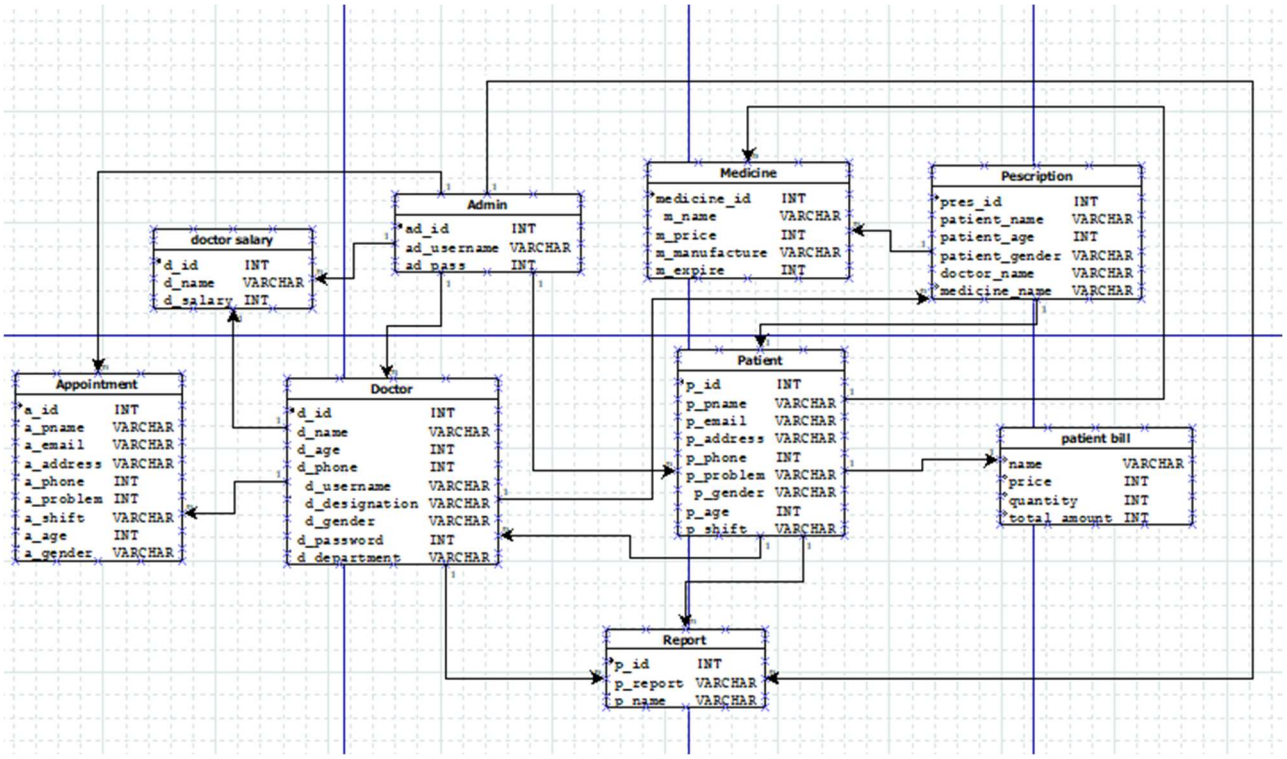
ad_username, ad_pass, ad_id(PK)

3NF

d_id(PK), d_name, d_salary

ad_username, ad_pass, ad_id(PK)

## 8. Schema Diagram




## 9. Table Creation:

CREATE USER Doctor IDENTIFIED BY doctor;

GRANT ALL PRIVILEGES TO Doctor; CREATE USER Patient IDENTIFIED BY patient;

GRANT ALL PRIVILEGES TO Patient; CREATE USER Admin IDENTIFIED BY admin;

GRANT ALL PRIVILEGES TO Admin;

CREATE ROLE doctor;

CREATE ROLE patient

CREATE ROLE admin;

GRANT SELECT, INSERT, UPDATE, DELETE ON Add_User TO Doctor;

GRANT SELECT, INSERT, UPDATE, DELETE ON Add_User TO Patient;

GRANT SELECT, INSERT, UPDATE, DELETE ON Add_User TO Admin;

GRANT doctor TO Doctor;

```
GRANT patient TO Patient;
GRANT admin TO Admin;
CREATE TABLE Doctors (
   d_id INT PRIMARY KEY,
   d_name VARCHAR2(50),
   d_age INT,
   d_department VARCHAR2(50),
   d_designation VARCHAR2(50),
   d_phone VARCHAR2(20),
   d_gender VARCHAR2(10),
   d_username VARCHAR2(20),
   d_password VARCHAR2(70)
);

CREATE TABLE Appointment (
   a_id INT PRIMARY KEY,
   a_pname VARCHAR2(50),
   a_email VARCHAR2(70),
   a_address VARCHAR2(250),
   a_dpt VATCHAR2(50);
   a_phone VARCHAR2(20),
   a_problem VARCHAR2(250),
   a_shift VARCHAR2(20),
   a_age INT,
   a_gender VARCHAR2(10)
);

CREATE TABLE Medicine (
   Medicine_id INT PRIMARY KEY,
   m_name VARCHAR2(50),
   m_manufacture VARCHAR2(50),
```

```
    m_expire VARCHAR2(50),
    m_price NUMBER(10, 2)
);

CREATE TABLE Patient (
    p_id INT PRIMARY KEY,
    p_pname VARCHAR2(50),
    p_email VARCHAR2(70),
    p_address VARCHAR2(250),
    p_dpt VATCHAR2(50);
    p_phone VARCHAR2(20),
    p_problem VARCHAR2(250),
     p_shift VARCHAR2(50),
    p_age NUMBER,
    p_gender VARCHAR2(10)
);

CREATE TABLE admin (
    ad_id NUMBER PRIMARY KEY,
    ad_username VARCHAR2(50),
    ad_pass NUMBER
);

CREATE TABLE doctor_salary (
    d_id NUMBER,
    d_name VARCHAR2(50),
    d_salary NUMBER
);

CREATE TABLE reports (
    p_id NUMBER,
```

```
    p_name VARCHAR2(50),
    p_reports VARCHAR2(250)
);

CREATE TABLE Prescription (
    PRES_ID NUMBER,
    patient_name VARCHAR2(50),
    patient_age VARCHAR2(50),
    patient_gender VARCHAR2(50),
    doctor_name VARCHAR2(50),
    medicine_name VARCHAR2(50)
);
CREATE TABLE patient_bill (
   name VARCHAR2(50),
   price number,
   quantity number,
   total_amount number
);
```

## 10. Data Insertion

**Data Insertion**

```
-- Insert data into Doctors table
INSERT INTO Doctors (d_id, d_name, d_age, d_department, d_designation, d_phone, d_gender,
d_username, d_password) VALUES
    (1, 'Dr. Fahad Khan', 35, 'Cardiology', 'Cardiologist', '123-456-7890', 'Male', 'fahad', 'fahad123');
INSERT INTO Doctors (d_id, d_name, d_age, d_department, d_designation, d_phone, d_gender,
d_username, d_password) VALUES
    (2, 'Dr. Faiza Alam', 42, 'Orthopedics', 'Orthopedic Surgeon', '987-654-3210', 'Female', 'faiza', 'faiza123');
INSERT INTO Doctors (d_id, d_name, d_age, d_department, d_designation, d_phone, d_gender,
d_username, d_password) VALUES
    (3, 'Dr. Mridha', 29, 'Pediatrics', 'Pediatrician', '555-111-2222', 'Male', 'mridha', 'mridha123');
INSERT INTO Doctors (d_id, d_name, d_age, d_department, d_designation, d_phone, d_gender,
d_username, d_password) VALUES
```

(4, 'Dr. Pulok Sarkar', 50, 'Dermatology', 'Dermatologist', '555-333-4444', 'Male', 'pulok', 'pulok123');

INSERT INTO Doctors (d_id, d_name, d_age, d_department, d_designation, d_phone, d_gender, d_username, d_password) VALUES

(5, 'Dr. Liza', 38, 'Gynecology', 'Gynecologist', '555-555-5555', 'Female', 'liza123', 'liza123');

describe doctors

Object Type **TABLE** Object **DOCTORS**

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| DOCTORS | D_ID | NUMBER | 22 | - | 0 | 1 | - | - | - |
| | D_NAME | VARCHAR2 | 50 | - | - | - | ✓ | - | - |
| | D_AGE | NUMBER | 22 | - | 0 | - | ✓ | - | - |
| | D_DEPARTMENT | VARCHAR2 | 50 | - | - | - | ✓ | - | - |
| | D_DESIGNATION | VARCHAR2 | 50 | - | - | - | ✓ | - | - |
| | D_PHONE | VARCHAR2 | 20 | - | - | - | ✓ | - | - |
| | D_GENDER | VARCHAR2 | 10 | - | - | - | ✓ | - | - |
| | D_USERNAME | VARCHAR2 | 20 | - | - | - | ✓ | - | - |
| | D_PASSWORD | VARCHAR2 | 70 | - | - | - | ✓ | - | - |

1 - 9

Select * from doctors

| D_ID | D_NAME | D_AGE | D_DEPARTMENT | D_DESIGNATION | D_PHONE | D_GENDER | D_USERNAME | D_PASSWORD |
|---|---|---|---|---|---|---|---|---|
| 1 | Dr. Fahad Khan | 35 | Cardiology | Cardiologist | 123-456-7890 | Male | fahad | fahad123 |
| 2 | Dr. Faiza Alam | 42 | Orthopedics | Orthopedic Surgeon | 987-654-3210 | Female | faiza | faiza123 |
| 3 | Dr. Mridha | 29 | Pediatrics | Pediatrician | 555-111-2222 | Male | mridha | mridha123 |
| 4 | Dr. Pulok Sarkar | 50 | Dermatology | Dermatologist | 555-333-4444 | Male | pulok | pulok123 |
| 5 | Dr. Liza | 38 | Gynecology | Gynecologist | 555-555-5555 | Female | liza123 | liza123 |

5 rows returned in 0.01 seconds Download

-- Insert data into Appointment table

INSERT INTO Appointment (a_id, a_pname, a_email, a_address, a_dpt, a_phone, a_problem, a_shift, a_age, a_gender) VALUES

(1, 'Khurshed Alam', 'alam67@gmail.com', 'kuril dhaka','Cardiology', '555-123-4567', 'Chest Pain', 'Morning', 45, 'Male');

INSERT INTO Appointment (a_id, a_pname, a_email, a_address, a_dpt, a_phone, a_problem, a_shift, a_age, a_gender) VALUES

(2, 'Rahim', 'rahim@yahoo.com', 'mirpur dhaka','Cardiology' ,'555-987-6543', 'Fractured Arm', 'Afternoon', 28, 'Male');

INSERT INTO Appointment (a_id, a_pname, a_email, a_address, a_dpt, a_phone, a_problem, a_shift, a_age, a_gender) VALUES

(3, 'Adrita Islam', 'adrita@gmail.com', 'bailyroad dhaka','Orthopedics' ,'555-555-7890', 'Rash', 'Morning', 8, 'Female');

INSERT INTO Appointment (a_id, a_pname, a_email, a_address, a_dpt, a_phone, a_problem, a_shift, a_age, a_gender) VALUES

(4, 'Sakib Hassan', 'hassan@gmail.com', 'dhanmondi dhaka','Pediatrics', '555-987-6543', 'Fever', 'Afternoon', 6, 'Male');

INSERT INTO Appointment (a_id, a_pname, a_email, a_address, a_dpt, a_phone, a_problem, a_shift, a_age, a_gender) VALUES

(5, 'Alex Hassan', 'alex@hotmail.com', 'motijhil dhaka','Pediatrics', '555-555-1111', 'Flu', 'Morning', 37, 'Male');

describe appointment

Object Type **TABLE** Object **APPOINTMENT**

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| APPOINTMENT | A_ID | NUMBER | 22 | - | 0 | 1 | - | - | - |
| | A_PNAME | VARCHAR2 | 50 | - | - | - | ✓ | - | - |
| | A_EMAIL | VARCHAR2 | 70 | - | - | - | ✓ | - | - |
| | A_ADDRESS | VARCHAR2 | 250 | - | - | - | ✓ | - | - |
| | A_DPT | VARCHAR2 | 50 | - | - | - | ✓ | - | - |
| | A_PHONE | VARCHAR2 | 20 | - | - | - | ✓ | - | - |
| | A_PROBLEM | VARCHAR2 | 250 | - | - | - | ✓ | - | - |
| | A_SHIFT | VARCHAR2 | 20 | - | - | - | ✓ | - | - |
| | A_AGE | NUMBER | 22 | - | 0 | - | ✓ | - | - |
| | A_GENDER | VARCHAR2 | 10 | - | - | - | ✓ | - | - |

1 - 10

select * from appointment

| A_ID | A_PNAME | A_EMAIL | A_ADDRESS | A_DPT | A_PHONE | A_PROBLEM | A_SHIFT | A_AGE | A_GENDER |
|---|---|---|---|---|---|---|---|---|---|
| 1 | Khurshed Alam | alam67@gmail.com | kuril dhaka | Cardiology | 555-123-4567 | Chest Pain | Morning | 45 | Male |
| 2 | Rahim | rahim@yahoo.com | mirpur dhaka | Cardiology | 555-987-6543 | Fractured Arm | Afternoon | 28 | Male |
| 3 | Adrita Islam | adrita@gmail.com | bailyroad dhaka | Orthopedics | 555-555-7890 | Rash | Morning | 8 | Female |
| 4 | Sakib Hassan | hassan@gmail.com | dhanmondi dhaka | Pediatrics | 555-987-6543 | Fever | Afternoon | 6 | Male |
| 5 | Alex Hassan | alex@hotmail.com | motijhil dhaka | Pediatrics | 555-555-1111 | Flu | Morning | 37 | Male |

5 rows returned in 0.00 seconds Download

-- Insert data into Medicine table

INSERT INTO Medicine (medicine_id, m_name, m_manufacture, m_expire, m_price) VALUES

(1, 'Aspirin', 'PharmaCorp', '2024-08-31', 5);

INSERT INTO Medicine (medicine_id, m_name, m_manufacture, m_expire, m_price) VALUES

(2, 'Ibuprofen', 'MediMega', '2023-12-15', 8);

INSERT INTO Medicine (medicine_id, m_name, m_manufacture, m_expire, m_price) VALUES

(3, 'Paracetamol', 'MediPharm', '2024-10-15', 4);

```
INSERT INTO Medicine (medicine_id, m_name, m_manufacture, m_expire, m_price) VALUES
   (4, 'Antihistamine', 'HealthMeds', '2023-11-30', 7);
INSERT INTO Medicine (medicine_id, m_name, m_manufacture, m_expire, m_price) VALUES
   (5, 'Cough Syrup', 'RemedyCare', '2023-09-20', 6);
```

describe medicine

Object Type **TABLE** Object **MEDICINE**

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| MEDICINE | MEDICINE_ID | NUMBER | 22 | - | 0 | 1 | - | - | - |
| | M_NAME | VARCHAR2 | 50 | - | - | - | ✓ | - | - |
| | M_MANUFACTURE | VARCHAR2 | 50 | - | - | - | ✓ | - | - |
| | M_EXPIRE | VARCHAR2 | 50 | - | - | - | ✓ | - | - |
| | M_PRICE | NUMBER | - | 10 | 2 | - | ✓ | - | - |

1 - 5

select * from medicine

| MEDICINE_ID | M_NAME | M_MANUFACTURE | M_EXPIRE | M_PRICE |
|---|---|---|---|---|
| 1 | Aspirin | PharmaCorp | 2024-08-31 | 5 |
| 2 | Ibuprofen | MediMega | 2023-12-15 | 8 |
| 3 | Paracetamol | MediPharm | 2024-10-15 | 4 |
| 4 | Antihistamine | HealthMeds | 2023-11-30 | 7 |
| 5 | Cough Syrup | RemedyCare | 2023-09-20 | 6 |

5 rows returned in 0.00 seconds Download

```
--insert data into patient table

INSERT INTO Patient (p_id, p_pname, p_email, p_address, p_phone, p_problem,p_shift, p_age, p_gender)

VALUES (2, 'Fahad Khan', 'fahad@gmail.com', 'kuril dhaka', '555-5678', 'Fractured wrist','Morning', 32, 'Male');

INSERT INTO Patient (p_id, p_pname, p_email, p_address, p_phone, p_problem,p_shift, p_age, p_gender)

VALUES (3, 'Fairoz Sharmin', 'michael@gmail.com', 'kuril dhaka', '555-9876', 'Pregnancy checkup','Evening', 28, 'Female');

INSERT INTO Patient (p_id, p_pname, p_email, p_address, p_phone, p_problem,p_shift, p_age, p_gender)

VALUES (4, 'Emily Hassan', 'emily@yahoo.com', 'mirpur dhaka', '555-5555', 'Pediatric visit','Afternoon', 7, 'Female');

INSERT INTO Patient (p_id, p_pname, p_email, p_address, p_phone, p_problem,p_shift, p_age, p_gender)
```

VALUES (5, 'Poluk Sarkar', 'pulok@gmail.com', 'dhanmondi dhaka', '555-4321', 'Skin rash','Morning', 60, 'Male');

Describe patient

Object Type **TABLE** Object **PATIENT**

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| PATIENT | P_ID | NUMBER | 22 | - | 0 | 1 | - | - | - |
| | P_PNAME | VARCHAR2 | 50 | - | - | - | ✓ | - | - |
| | P_EMAIL | VARCHAR2 | 70 | - | - | - | ✓ | - | - |
| | P_ADDRESS | VARCHAR2 | 250 | - | - | - | ✓ | - | - |
| | P_DPT | VARCHAR2 | 50 | - | - | - | ✓ | - | - |
| | P_PHONE | VARCHAR2 | 20 | - | - | - | ✓ | - | - |
| | P_PROBLEM | VARCHAR2 | 250 | - | - | - | ✓ | - | - |
| | P_SHIFT | VARCHAR2 | 20 | - | - | - | ✓ | - | - |
| | P_AGE | NUMBER | 22 | - | 0 | - | ✓ | - | - |
| | P_GENDER | VARCHAR2 | 10 | - | - | - | ✓ | - | - |

1 - 10

Select * from patient

| P_ID | P_PNAME | P_EMAIL | P_ADDRESS | P_DPT | P_PHONE | P_PROBLEM | P_SHIFT | P_AGE | P_GENDER |
|---|---|---|---|---|---|---|---|---|---|
| 2 | Fahad Khan | fahad@gmail.com | kuril dhaka | Cardiology | 555-5678 | Fractured wrist | Morning | 32 | Male |
| 3 | Fairoz Sharmin | michael@gmail.com | kuril dhaka | Cardiology | 555-9876 | Pregnancy checkup | Evening | 28 | Female |
| 5 | Poluk Sarkar | pulok@gmail.com | dhanmondi dhaka | Orthopedics | 555-4321 | Skin rash | Morning | 60 | Male |
| 4 | Emily Hassan | emily@yahoo.com | mirpur dhaka | Dermatology | 555-5555 | Pediatric visit | Afternoon | 7 | Female |

4 rows returned in 0.00 seconds Download

-- Insert data into admin table

INSERT INTO admin (ad_id, ad_username, ad_pass) VALUES

(1, 'admin1', 123456);

INSERT INTO admin (ad_id, ad_username, ad_pass) VALUES

(2, 'admin2', 987654);

Describe admin

Object Type **TABLE** Object **ADMIN**

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| ADMIN | AD_ID | NUMBER | 22 | - | - | 1 | - | - | - |
| | AD_USERNAME | VARCHAR2 | 50 | - | - | - | ✓ | - | - |
| | AD_PASS | NUMBER | 22 | - | - | - | ✓ | - | - |

1 - 3

Select * from admin

| AD_ID | AD_USERNAME | AD_PASS |
|---|---|---|
| 1 | admin1 | 123456 |
| 2 | admin2 | 987654 |

2 rows returned in 0.00 seconds Download

```
-- Insert data into doctor_salary table
INSERT INTO doctor_salary (d_id, d_name, d_salary) VALUES
   (1, 'Dr. Mirza', 120000);
INSERT INTO doctor_salary (d_id, d_name, d_salary) VALUES
   (2, 'Dr. Sohel', 135000);
INSERT INTO doctor_salary (d_id, d_name, d_salary) VALUES
   (3, 'Dr. Fahad', 98000);
INSERT INTO doctor_salary (d_id, d_name, d_salary) VALUES
   (4, 'Dr. Pulok', 150000);
INSERT INTO doctor_salary (d_id, d_name, d_salary) VALUES
   (5, 'Dr. Raisa', 110000);
```

Describe doctor_salary

Object Type **TABLE** Object **DOCTOR_SALARY**

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| DOCTOR_SALARY | D_ID | NUMBER | 22 | - | - | - | ✓ | - | - |
| | D_NAME | VARCHAR2 | 50 | - | - | - | ✓ | - | - |
| | D_SALARY | NUMBER | 22 | - | - | - | ✓ | - | - |

1 - 3

Select * from doctor_salary

| D_ID | D_NAME | D_SALARY |
|---|---|---|
| 1 | Dr. Mirza | 120000 |
| 2 | Dr. Sohel | 135000 |
| 3 | Dr. Fahad | 98000 |
| 4 | Dr. Pulok | 150000 |
| 5 | Dr. Raisa | 110000 |

5 rows returned in 0.00 seconds Download

```
-- Insert data into reports table
INSERT INTO reports (p_id, p_name, p_reports) VALUES
   (1, 'Akbor Ali', 'X-ray report shows no issues');
INSERT INTO reports (p_id, p_name, p_reports) VALUES
   (2, 'Mehedi Hasan', 'Blood test results are normal');
INSERT INTO reports (p_id, p_name, p_reports) VALUES
   (3, 'Sarah Rahman', 'Blood test results pending');
INSERT INTO reports (p_id, p_name, p_reports) VALUES
   (4, 'Mirza Mehedi', 'X-ray shows mild infection');
INSERT INTO reports (p_id, p_name, p_reports) VALUES
   (5, 'Rumaisha Islam', 'Allergy test recommended');
```

Describe reports

Object Type **TABLE** Object **REPORTS**

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| REPORTS | P_ID | NUMBER | 22 | - | - | - | ✓ | - | - |
| | P_NAME | VARCHAR2 | 50 | - | - | - | ✓ | - | - |
| | P_REPORTS | VARCHAR2 | 250 | - | - | - | ✓ | - | - |
| | | | | | | | | | 1 - 3 |

Select * from reports

| P_ID | P_NAME | P_REPORTS |
|---|---|---|
| 1 | Akbor Ali | X-ray report shows no issues |
| 2 | Mehedi Hasan | Blood test results are normal |
| 3 | Sarah Rahman | Blood test results pending |
| 4 | Mirza Mehedi | X-ray shows mild infection |
| 5 | Rumaisha Islam | Allergy test recommended |

5 rows returned in 0.00 seconds Download

-- Insert data into Prescription table

INSERT INTO Prescription (PRES_ID, patient_name, patient_age, patient_gender, doctor_name, medicine_name) VALUES

(1, 'Rumaisha', '32', 'Female', 'Dr. Fahad', 'Aspirin');

INSERT INTO Prescription (PRES_ID, patient_name, patient_age, patient_gender, doctor_name, medicine_name) VALUES

(2, 'Fahim', '45', 'Male', 'Dr. Mirza', 'Ibuprofen');

INSERT INTO Prescription (PRES_ID, patient_name, patient_age, patient_gender, doctor_name, medicine_name) VALUES

(3, 'Sarah', '28', 'Female', 'Dr. Sahil', 'Paracetamol');

INSERT INTO Prescription (PRES_ID, patient_name, patient_age, patient_gender, doctor_name, medicine_name) VALUES

(4, 'Fahad', '40', 'Male', 'Dr. Moshiur', 'Antihistamine');

INSERT INTO Prescription (PRES_ID, patient_name, patient_age, patient_gender, doctor_name, medicine_name) VALUES

(5, 'Suraifa', '22', 'Female', 'Dr. Fahad', 'Cough Syrup');

Describe prescription

Object Type **TABLE** Object **PRESCRIPTION**

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| PRESCRIPTION | PRES_ID | NUMBER | 22 | - | - | - | ✓ | - | - |
| | PATIENT_NAME | VARCHAR2 | 50 | - | - | - | ✓ | - | - |
| | PATIENT_AGE | VARCHAR2 | 50 | - | - | - | ✓ | - | - |
| | PATIENT_GENDER | VARCHAR2 | 50 | - | - | - | ✓ | - | - |
| | DOCTOR_NAME | VARCHAR2 | 50 | - | - | - | ✓ | - | - |
| | MEDICINE_NAME | VARCHAR2 | 50 | - | - | - | ✓ | - | - |

1 - 6

Select * from prescription

| PRES_ID | PATIENT_NAME | PATIENT_AGE | PATIENT_GENDER | DOCTOR_NAME | MEDICINE_NAME |
|---|---|---|---|---|---|
| 1 | Rumaisha | 32 | Female | Dr. Fahad | Aspirin |
| 2 | Fahim | 45 | Male | Dr. Mirza | Ibuprofen |
| 3 | Sarah | 28 | Female | Dr. Sahil | Paracetamol |
| 4 | Fahad | 40 | Male | Dr. Moshiur | Antihistamine |
| 5 | Suraifa | 22 | Female | Dr. Fahad | Cough Syrup |

5 rows returned in 0.01 seconds Download

-- Insert data into patient_bill table with detailed information

INSERT INTO patient_bill (name, price, quantity, total_amount)

VALUES ('Medication A', 20.00, 3, 60.00);

INSERT INTO patient_bill (name, price, quantity, total_amount)

VALUES ('Treatment B', 50.00, 1, 50.00);

INSERT INTO patient_bill (name, price, quantity, total_amount)

VALUES ('Lab Test X', 25.00, 2, 50.00);

INSERT INTO patient_bill (name, price, quantity, total_amount)

VALUES ('Surgery Y', 200.00, 1, 200.00);

INSERT INTO patient_bill (name, price, quantity, total_amount)

VALUES ('Consultation Z', 50.00, 1, 50.00);

Describe patient_bill

Object Type **TABLE** Object **PATIENT_BILL**

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| PATIENT_BILL | NAME | VARCHAR2 | 50 | - | - | - | ✓ | - | - |
| | PRICE | NUMBER | 22 | - | - | - | ✓ | - | - |
| | QUANTITY | NUMBER | 22 | - | - | - | ✓ | - | - |
| | TOTAL_AMOUNT | NUMBER | 22 | - | - | - | ✓ | - | - |

1 - 4

Select * from patient_bill

| NAME | PRICE | QUANTITY | TOTAL_AMOUNT |
|---|---|---|---|
| Medication A | 20 | 3 | 60 |
| Treatment B | 50 | 1 | 50 |
| Lab Test X | 25 | 2 | 50 |
| Surgery Y | 200 | 1 | 200 |
| Consultation Z | 50 | 1 | 50 |

5 rows returned in 0.00 seconds Download

# 10. Query Writing:

## 3 single-row-function Question:

1. Retrieve the system doctor names in upper case.

   **Answer:** SELECT UPPER(d_name) AS Capital_name
   FROM Doctors;

| CAPITAL_NAME |
|---|
| DR. FAHAD KHAN |
| DR. FAIZA ALAM |
| DR. MRIDHA |
| DR. PULOK SARKAR |
| DR. LIZA |

5 rows returned in 0.00 seconds Download

2. Question: Retrieve the full names of CA_Doctors users along with the length of their names.

   Answer: SELECT d_name AS full_name, LENGTH(d_name) AS name_length
   FROM Doctors;

| FULL_NAME | NAME_LENGTH |
|---|---|
| Dr. Fahad khan | 14 |
| Dr. Faiza Alam | 14 |
| Dr. Mridha | 10 |
| Dr. Pulok Sarkar | 16 |
| Dr. Liza | 8 |

5 rows returned in 0.00 seconds Download

3. Question: Retrieve Female Patients.

   Answer: SELECT p_id, p_pname, p_age FROM Patient WHERE p_gender = 'Female';

| P_ID | P_PNAME | P_AGE |
|---|---|---|
| 3 | Fairoz Sharmin | 28 |
| 4 | Emily Hassan | 7 |

2 rows returned in 0.00 seconds Download

## 3 group function

1. **Question:** Calculate the average length of CA_Doctors passwords.

   **Answer:** SELECT AVG(LENGTH(d_password)) AS avg_password_length
   FROM Doctors;

| AVG_PASSWORD_LENGTH |
|---|
| 8 |

1 rows returned in 0.00 seconds Download

2. Question: Find the maximum value of CA_Doctors mobile numbers.

   **Answer:** SELECT MAX(d_phone) AS max_mobile_number FROM Doctors;

| MAX_MOBILE_NUMBER |
|---|
| 987-654-3210 |

1 rows returned in 0.00 seconds Download

3. Question: Retrieve the count of CA_Doctors records.

**Answer:** SELECT COUNT(*) AS record_count FROM Doctors;

| RECORD_COUNT |
| --- |
| 5 |

1 rows returned in 0.00 seconds Download

## 3 subquery

1. **Question:** Retrieve the Doctors users who have the same gender as any Appointment user.

   Answer: SELECT d_gender FROM Doctors WHERE d_gender IN (SELECT a_gender FROM Appointment);

| D_GENDER |
| --- |
| Male |
| Male |
| Female |
| Female |

4 rows returned in 0.00 seconds Download

2. **Question:** Retrieve the Doctors users who have a name length greater than the average name length of all Appointment users.

   **Answer:** SELECT d_name
   FROM Doctors
   WHERE LENGTH(d_name) > (
   SELECT AVG(LENGTH(a_pname))
   FROM Appointment);

| D_NAME |
| --- |
| Dr. Fahad khan |
| Dr. Faiza Alam |
| Dr. Pulok Sarkar |

3 rows returned in 0.01 seconds Download

3. **Question:** Retrieve the Doctors users who have a phone that is not present in the Appointment table.

   **Answer:** SELECT d_phone
   FROM Doctors

```
WHERE d_phone NOT IN (
SELECT DISTINCT a_phone
FROM Appointment);
```

| D_PHONE |
|---|
| 555-555-5555 |
| 555-111-2222 |
| 555-333-4444 |
| 987-654-3210 |
| 123-456-7890 |

5 rows returned in 0.00 seconds Download

## 3 joining

1. **Question:** retrieves information about patients and any prescriptions they have received

   **Answer:** SELECT P.p_pname, P.p_age, P.p_gender, PR.doctor_name, PR.medicine_name
   FROM Patient P
   LEFT JOIN Prescription PR ON P.p_id = PR.PRES_ID;

| P_PNAME | P_AGE | P_GENDER | DOCTOR_NAME | MEDICINE_NAME |
|---|---|---|---|---|
| Fahad Khan | 32 | Male | Dr. Mirza | Ibuprofen |
| Fairoz Sharmin | 28 | Female | Dr. Sahil | Paracetamol |
| Emily Hassan | 7 | Female | Dr. Moshiur | Antihistamine |
| Poluk Sarkar | 60 | Male | Dr. Fahad | Cough Syrup |

4 rows returned in 0.01 seconds Download

2. **Question:** Retrieve the SA_Add_User users along with their corresponding D_Add_User user, if any, based on the same id.
   **Answer:** SELECT d.d_name AS d_name, a.a_pname AS a_pname
   FROM Doctors d
   LEFT JOIN Appointment a ON d.d_id = a.a_id;

| D_NAME | A_PNAME |
|---|---|
| Dr. Fahad khan | Khurshed Alam |
| Dr. Faiza Alam | Rahim |
| Dr. Mridha | Adrita Islam |
| Dr. Pulok Sarkar | Sakib Hassan |
| Dr. Liza | Alex Hassan |

5 rows returned in 0.00 seconds Download

3. **Question:** Retrieve the names and ages of doctors from a specific department.
   **Answer:** SELECT d.d_name, d.d_age
   FROM Doctors d
   LEFT JOIN department dp ON d.dpt_id = dp.dpt_id
   WHERE dp.dpt_id = 2;

| D_NAME | D_AGE |
|---|---|
| Dr. Faiza Alam | 42 |

1 rows returned in 0.00 seconds Download

## 3 View

Question: Create a view that displays doctors with their salaries who earn more than a certain amount.

Answer: CREATE VIEW High_Earning_Doctors AS
SELECT d_id, d_name, d_salary
FROM doctor_salary
WHERE d_salary > 10000;

```
View created.
```

0.02 seconds

select * from High_Earning_Doctors;

| D_ID | D_NAME | D_SALARY |
|---|---|---|
| 1 | Dr. Mirza | 120000 |
| 2 | Dr. Sohel | 135000 |
| 3 | Dr. Fahad | 98000 |
| 4 | Dr. Pulok | 150000 |
| 5 | Dr. Raisa | 110000 |

5 rows returned in 0.00 seconds Download

**Question:** Create a view that lists patients under the age of 30.

**Answer:** CREATE VIEW Young_Patients AS
SELECT p_id, p_pname AS patient_name, p_age
FROM Patient
WHERE p_age < 30;

```
View created.

0.02 seconds
```

select * from  Young_Patients;

| P_ID | PATIENT_NAME | P_AGE |
|---|---|---|
| 3 | Fairoz Sharmin | 28 |
| 4 | Emily Hassan | 7 |

2 rows returned in 0.00 seconds Download

**Question:** This view combines information from the Patient and Appointment tables to provide a comprehensive overview of patients and their appointments.

**Answer:** CREATE VIEW PatientAppointments AS
SELECT P.p_id, P.p_pname, P.p_age, P.p_gender,
A.a_id, A.a_dpt, A.a_problem, A.a_shift
FROM Patient P
LEFT JOIN Appointment A ON P.p_id = A.a_id;

```
View created.

0.02 seconds
```

select * from PatientAppointments

| P_ID | P_PNAME | P_AGE | P_GENDER | A_ID | A_DPT | A_PROBLEM | A_SHIFT |
|---|---|---|---|---|---|---|---|
| 2 | Fahad Khan | 32 | Male | 2 | Cardiology | Fractured Arm | Afternoon |
| 3 | Fairoz Sharmin | 28 | Female | 3 | Orthopedics | Rash | Morning |
| 4 | Emily Hassan | 7 | Female | 4 | Pediatrics | Fever | Afternoon |
| 5 | Poluk Sarkar | 60 | Male | 5 | Pediatrics | Flu | Morning |

4 rows returned in 0.00 seconds Download

## 3 synonyms

**Question:** Create a synonym for the Doctors table.

**Answer:** CREATE SYNONYM Doctors_sym FOR Doctors;

```
Synonym created.

0.00 seconds
```

SELECT * FROM Doctors_sym;

| D_ID | D_NAME | D_AGE | D_DEPARTMENT | D_DESIGNATION | D_PHONE | D_GENDER | D_USERNAME | D_PASSWORD |
|---|---|---|---|---|---|---|---|---|
| 1 | Dr. Fahad Khan | 35 | Cardiology | Cardiologist | 123-456-7890 | Male | fahad | fahad123 |
| 2 | Dr. Faiza Alam | 42 | Orthopedics | Orthopedic Surgeon | 987-654-3210 | Female | faiza | faiza123 |
| 3 | Dr. Mridha | 29 | Pediatrics | Pediatrician | 555-111-2222 | Male | mridha | mridha123 |
| 4 | Dr. Pulok Sarkar | 50 | Dermatology | Dermatologist | 555-333-4444 | Male | pulok | pulok123 |
| 5 | Dr. Liza | 38 | Gynecology | Gynecologist | 555-555-5555 | Female | liza123 | liza123 |

5 rows returned in 0.00 seconds Download

**Question:** Create a synonym for the Appointment table.

**Answer:** CREATE SYNONYM Appointment_sym FOR Appointment;

```
Synonym created.

0.00 seconds
```

SELECT * FROM Appointment_sym;

| A_ID | A_PNAME | A_EMAIL | A_ADDRESS | A_PHONE | A_PROBLEM | A_SHIFT | A_AGE | A_GENDER |
|---|---|---|---|---|---|---|---|---|
| 1 | Khurshed Alam | alam67@gmail.com | kuril dhaka | 555-123-4567 | Chest Pain | Morning | 45 | Male |
| 2 | Rahim | rahim@yahoo.com | mirpur dhaka | 555-987-6543 | Fractured Arm | Afternoon | 28 | Male |
| 3 | Adrita Islam | adrita@gmail.com | bailyroad dhaka | 555-555-7890 | Rash | Morning | 8 | Female |
| 4 | Sakib Hassan | hassan@gmail.com | dhanmondi dhaka | 555-987-6543 | Fever | Afternoon | 6 | Male |
| 5 | Alex Hassan | alex@hotmail.com | motijhil dhaka | 555-555-1111 | Flu | Morning | 37 | Male |

5 rows returned in 0.00 seconds Download

**Question:** Create a synonym for the Patient table.

**Answer:** CREATE SYNONYM Patient_sym FOR Patient;

```
Synonym created.

0.00 seconds
```

SELECT * FROM Patient_sym;

| P_ID | P_PNAME | P_EMAIL | P_ADDRESS | P_PHONE | P_PROBLEM | P_SHIFT | P_AGE | P_GENDER |
|---|---|---|---|---|---|---|---|---|
| 1 | Sahilur Rahman | sahil@yahoo.com | mirpur dhaka | 555-1234 | Heart issues | Afternoon | 45 | Male |
| 2 | Fahad Khan | fahad@gmail.com | kuril dhaka | 555-5678 | Fractured wrist | Morning | 32 | Male |
| 3 | Fairoz Sharmin | michael@gmail.com | kuril dhaka | 555-9876 | Pregnancy checkup | Evening | 28 | Female |
| 4 | Emily Hassan | emily@yahoo.com | mirpur dhaka | 555-5555 | Pediatric visit | Afternoon | 7 | Female |
| 5 | Poluk Sarkar | pulok@gmail.com | dhanmondi dhaka | 555-4321 | Skin rash | Morning | 60 | Male |

5 rows returned in 0.00 seconds Download

# 11.PL/SQL:

## 3 Function

**Question:** Create a function that retrieves the total number of appointments for a given doctor based on their id from the Doctors and Appointment tables.

**Answer:**

```
CREATE OR REPLACE FUNCTION totaldoctor
RETURN number IS
t_doctor number(30) := 0;
BEGIN
SELECT count(*) into t_doctor
FROM Doctors;
RETURN t_doctor;
END;
```

```
Function created.

0.02 seconds
```

```
DECLARE
c number(30);
BEGIN
c := totaldoctor();
dbms_output.put_line('Total Number of Doctors: ' || c);
```

```
Total Number of Doctors: 5

Statement processed.

0.00 seconds
```

**Question:** Create a function that retrieves the total salary of a doctor based on their ID.

**Answer:** CREATE OR REPLACE FUNCTION get_doctor_salary(d_id_in NUMBER)

```
RETURN NUMBER
IS
   doctor_salary NUMBER;
BEGIN
   SELECT d_salary INTO doctor_salary
   FROM doctor_salary
   WHERE d_id = d_id_in;
   RETURN doctor_salary;
EXCEPTION
   WHEN NO_DATA_FOUND THEN
      RETURN NULL;
END;
/
```

```
Function created.

0.00 seconds
```

```
DECLARE
   doctor_id NUMBER := 2; -- Change this to the desired doctor's ID
   salary NUMBER;
BEGIN
```

```
  salary := get_doctor_salary(doctor_id);

  IF salary IS NOT NULL THEN
    DBMS_OUTPUT.PUT_LINE('Doctor ' || doctor_id || ' salary: ' || salary);
  ELSE
    DBMS_OUTPUT.PUT_LINE('Doctor with ID ' || doctor_id || ' not found.');
  END IF;
END;
/
```

```
Doctor 2 salary: 135000
Statement processed.
0.00 seconds
```

**Question:** Create a function that that calculates the total bill amount for a given patient's ID.

**Answer:**

```
CREATE OR REPLACE FUNCTION calculate_patient_bill(p_id_in NUMBER)
RETURN NUMBER
IS
  total_amount NUMBER := 0;
BEGIN
  FOR bill_rec IN (SELECT price, quantity FROM patient_bill WHERE name IS NOT NULL) -- Modify the WHERE clause as needed
  LOOP
    total_amount := total_amount + (bill_rec.price * bill_rec.quantity);
  END LOOP;

  RETURN total_amount;
EXCEPTION
  WHEN NO_DATA_FOUND THEN
    RETURN NULL;
END;
```

```
/
```

```
Function created.

0.02 seconds
```

```
DECLARE
   patient_id NUMBER := 3; -- Change this to the desired patient's ID
   total_bill NUMBER;
BEGIN
   total_bill := calculate_patient_bill(patient_id);


   IF total_bill IS NOT NULL THEN
      DBMS_OUTPUT.PUT_LINE('Patient ' || patient_id || ' total bill: ' || total_bill);
   ELSE
      DBMS_OUTPUT.PUT_LINE('Patient with ID ' || patient_id || ' not found.');
   END IF;
END;
/
```

```
Patient 3 total bill: 410

Statement processed.

0.00 seconds
```

# 3 Procedure

**Question:** Create a procedure that updates the department of a doctor based on their ID.

**Answer:**

```
CREATE OR REPLACE PROCEDURE
  update_doctor_department(
    d_id_in NUMBER,
    new_department_in VARCHAR2
)
IS
BEGIN
    UPDATE Doctors
    SET d_department = new_department_in
    WHERE d_id = d_id_in;


    COMMIT;
    DBMS_OUTPUT.PUT_LINE('Doctor ' || d_id_in || ' department
  updated to: ' || new_department_in);
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Doctor with ID ' || d_id_in || ' not
  found.');
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
END;
```

```
Procedure created.

0.00 seconds
```

```
BEGIN
  update_doctor_department(2, 'Neurology');
END;
```

```
Doctor 2 department updated to: Neurology

Statement processed.

0.02 seconds
```

**Question:** Create a procedure that adds a new medicine to the Medicine table.

**Answer:**

```
CREATE OR REPLACE PROCEDURE add_new_medicine(
    m_id_in NUMBER,
    m_name_in VARCHAR2,
    m_manufacture_in VARCHAR2,
    m_expire_in VARCHAR2,
    m_price_in NUMBER
)
IS
BEGIN
    INSERT INTO Medicine (Medicine_id, m_name, m_manufacture,
  m_expire, m_price)
    VALUES (m_id_in, m_name_in, m_manufacture_in, m_expire_in,
  m_price_in);

    COMMIT;
```

```
    DBMS_OUTPUT.PUT_LINE('New medicine added: ' || m_name_in);
  EXCEPTION
    WHEN OTHERS THEN
      DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
  END;
  /
```

```
Procedure created.
0.00 seconds
```

```
BEGIN
   add_new_medicine(6, 'Antacid', 'HealthMeds', '2024-12-31', 10.50);
END;
/
```

```
New medicine added: Antacid
Statement processed.
0.00 seconds
```

**Question:** Create a procedure that deletes a prescription from the Prescription table based on the prescription ID.

**Answer** CREATE OR REPLACE PROCEDURE generate_patient_report(

```
    p_id_in NUMBER
  )
  IS
    report_text VARCHAR2(4000);
  BEGIN
    SELECT p_reports INTO report_text
```

```
    FROM reports
    WHERE p_id = p_id_in;

    DBMS_OUTPUT.PUT_LINE('Patient Report for ID ' || p_id_in || ':');
    DBMS_OUTPUT.PUT_LINE(report_text);
 EXCEPTION
    WHEN NO_DATA_FOUND THEN
       DBMS_OUTPUT.PUT_LINE('Report for Patient ID ' || p_id_in || ' not
  found.');
    WHEN OTHERS THEN
       DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
 END;
 /
```

```
Procedure created.

0.00 seconds
```

```
BEGIN
   generate_patient_report(3);
END;
/
```

```
Patient Report for ID 3:
Blood test results pending

Statement processed.

0.01 seconds
```

# 3 Record

**Question:** Create a PL/SQL block that retrieve records from a table and display their values. Here's an example using the Doctors table.

**Answer**: DECLARE

```
  doctor_record Doctors%ROWTYPE;
BEGIN
  SELECT * INTO doctor_record
  FROM Doctors
  WHERE d_id = 6; -- Change the ID as needed

  DBMS_OUTPUT.PUT_LINE('Doctor ID: ' || doctor_record.d_id);
  DBMS_OUTPUT.PUT_LINE('Doctor Name: ' || doctor_record.d_name);
  DBMS_OUTPUT.PUT_LINE('Department: ' || doctor_record.d_department);
  -- Add more output lines for other columns

EXCEPTION
  WHEN NO_DATA_FOUND THEN
    DBMS_OUTPUT.PUT_LINE('Doctor not found.');
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
END;
/
```

```
Doctor not found.

Statement processed.

0.00 seconds
```

**Question:** Create a PL/SQL block that retrieves records from the Doctors table and displays information for a range of IDs.

**Answer**:

```
DECLARE
  doctor_record Doctors%ROWTYPE;
  CURSOR doctor_cursor IS
    SELECT * FROM Doctors WHERE d_id BETWEEN 1 AND 3; -- Adjust the range as needed
BEGIN
  OPEN doctor_cursor;
  LOOP
    FETCH doctor_cursor INTO doctor_record;
    EXIT WHEN doctor_cursor%NOTFOUND;

    DBMS_OUTPUT.PUT_LINE('Doctor ID: ' || doctor_record.d_id);
    DBMS_OUTPUT.PUT_LINE('Doctor Name: ' || doctor_record.d_name);
    DBMS_OUTPUT.PUT_LINE('Department: ' || doctor_record.d_department);
    -- Add more output lines for other columns
    DBMS_OUTPUT.PUT_LINE('--------------------');
  END LOOP;
  CLOSE doctor_cursor;

EXCEPTION
  WHEN NO_DATA_FOUND THEN
    DBMS_OUTPUT.PUT_LINE('No doctors found.');
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
END;
/
```

```
Doctor ID: 1
Doctor Name: Dr. Fahad Khan
Department: Cardiology
-------------------
Doctor ID: 2
Doctor Name: Dr. Faiza Alam
Department: Neurology
-------------------
Doctor ID: 3
Doctor Name: Dr. Mridha
Department: Pediatrics
-------------------

Statement processed.

0.00 seconds
```

**Question:** Create a PL/SQL block that retrieves and displays records from the Medicine table based on a specific condition.

**Answer**:

```
DECLARE
  medicine_record Medicine%ROWTYPE;
  CURSOR medicine_cursor IS
    SELECT * FROM Medicine WHERE m_price > 5.00; -- Adjust the condition as needed
BEGIN
  OPEN medicine_cursor;
  LOOP
    FETCH medicine_cursor INTO medicine_record;
    EXIT WHEN medicine_cursor%NOTFOUND;

    DBMS_OUTPUT.PUT_LINE('Medicine ID: ' || medicine_record.Medicine_id);
    DBMS_OUTPUT.PUT_LINE('Medicine Name: ' || medicine_record.m_name);
    DBMS_OUTPUT.PUT_LINE('Manufacture: ' || medicine_record.m_manufacture);
    -- Add more output lines for other columns
    DBMS_OUTPUT.PUT_LINE('-------------------');
  END LOOP;
  CLOSE medicine_cursor;

EXCEPTION
```

```
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('No medicines found.');
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
END;
/
```

```
Medicine ID: 6
Medicine Name: Antacid
Manufacture: HealthMeds
-------------------
Medicine ID: 2
Medicine Name: Ibuprofen
Manufacture: MediMega
-------------------
Medicine ID: 4
Medicine Name: Antihistamine
Manufacture: HealthMeds
-------------------
Medicine ID: 5
Medicine Name: Cough Syrup
Manufacture: RemedyCare
-------------------

Statement processed.

0.01 seconds
```

## 3 Cursor

**Question:** Create a cursor to retrieve doctor information from the Doctor table.

**Answer**:

```
DECLARE
    CURSOR doctor_cursor IS
        SELECT * FROM Doctors;
    doctor_record Doctors%ROWTYPE;
BEGIN
    OPEN doctor_cursor;
    LOOP
        FETCH doctor_cursor INTO doctor_record;
        EXIT WHEN doctor_cursor%NOTFOUND;
```

```
    DBMS_OUTPUT.PUT_LINE('Doctor ID: ' || doctor_record.d_id);
    DBMS_OUTPUT.PUT_LINE('Doctor Name: ' || doctor_record.d_name);
    DBMS_OUTPUT.PUT_LINE('Department: ' || doctor_record.d_department);
    -- Add more output lines for other columns
    DBMS_OUTPUT.PUT_LINE('--------------------');
  END LOOP;
  CLOSE doctor_cursor;

EXCEPTION
  WHEN NO_DATA_FOUND THEN
    DBMS_OUTPUT.PUT_LINE('No doctors found.');
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
END;
/
```

```
Doctor ID: 1
Doctor Name: Dr. Fahad Khan
Department: Cardiology
--------------------
Doctor ID: 2
Doctor Name: Dr. Faiza Alam
Department: Neurology
--------------------
Doctor ID: 3
Doctor Name: Dr. Mridha
Department: Pediatrics
--------------------
Doctor ID: 4
Doctor Name: Dr. Pulok Sarkar
Department: Dermatology
--------------------
Doctor ID: 5
Doctor Name: Dr. Liza
Department: Gynecology
--------------------

Statement processed.
```

0.00 seconds

**Question:** Create a Cursor for Medicine Table with Price Filter.

**Answer**:

```
DECLARE
  CURSOR medicine_cursor IS
    SELECT * FROM Medicine WHERE m_price > 5.00;
  medicine_record Medicine%ROWTYPE;
BEGIN
  OPEN medicine_cursor;
  LOOP
    FETCH medicine_cursor INTO medicine_record;
    EXIT WHEN medicine_cursor%NOTFOUND;

    DBMS_OUTPUT.PUT_LINE('Medicine ID: ' || medicine_record.Medicine_id);
    DBMS_OUTPUT.PUT_LINE('Medicine Name: ' || medicine_record.m_name);
    DBMS_OUTPUT.PUT_LINE('Manufacture: ' || medicine_record.m_manufacture);
    -- Add more output lines for other columns
    DBMS_OUTPUT.PUT_LINE('--------------------');
  END LOOP;
  CLOSE medicine_cursor;

EXCEPTION
  WHEN NO_DATA_FOUND THEN
    DBMS_OUTPUT.PUT_LINE('No medicines found.');
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
END;
/
```

```
Medicine ID: 6
Medicine Name: Antacid
Manufacture: HealthMeds
-------------------
Medicine ID: 2
Medicine Name: Ibuprofen
Manufacture: MediMega
-------------------
Medicine ID: 4
Medicine Name: Antihistamine
Manufacture: HealthMeds
-------------------
Medicine ID: 5
Medicine Name: Cough Syrup
Manufacture: RemedyCare
-------------------

Statement processed.

0.00 seconds
```

**Question:** Create a cursor to retrieve medicine information from the Medicine table.

**Answer**: DECLARE

```
  CURSOR bill_cursor IS
    SELECT * FROM patient_bill;
  bill_record patient_bill%ROWTYPE;
BEGIN
  OPEN bill_cursor;
  LOOP
    FETCH bill_cursor INTO bill_record;
    EXIT WHEN bill_cursor%NOTFOUND;

    DBMS_OUTPUT.PUT_LINE('Bill Name: ' || bill_record.name);
    DBMS_OUTPUT.PUT_LINE('Price: ' || bill_record.price);
    -- Add more output lines for other columns
    DBMS_OUTPUT.PUT_LINE('-------------------');
  END LOOP;
  CLOSE bill_cursor;
```

```
EXCEPTION
   WHEN NO_DATA_FOUND THEN
      DBMS_OUTPUT.PUT_LINE('No bills found.');
   WHEN OTHERS THEN
      DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
END;
/
```

```
Bill Name: Medication A
Price: 20
--------------------
Bill Name: Treatment B
Price: 50
--------------------
Bill Name: Lab Test X
Price: 25
--------------------
Bill Name: Surgery Y
Price: 200
--------------------
Bill Name: Consultation Z
Price: 50
--------------------

Statement processed.
```

## 3 Trigger

**Question:** Create a trigger that Update Doctor's Age Automatically.

**Answer**: CREATE OR REPLACE TRIGGER update_doctor_age

BEFORE INSERT ON Doctors

FOR EACH ROW

DECLARE

birthdate DATE;

BEGIN

birthdate := SYSDATE - (:NEW.d_age * 365);

:NEW.d_age := EXTRACT(YEAR FROM birthdate);

END;

/

```
Trigger created.

0.01 seconds
```

**Question:** Create a trigger that will prevents the deletion of a doctor if there are active appointments associated with them..

**Answer**: CREATE OR REPLACE TRIGGER prevent_delete_doctor

BEFORE DELETE ON Doctors

FOR EACH ROW

DECLARE

  appointment_count NUMBER;

BEGIN

  SELECT COUNT(*) INTO appointment_count

  FROM Appointment

  WHERE a_dpt = :OLD.d_department;

  IF appointment_count > 0 THEN

    RAISE_APPLICATION_ERROR(-20001, 'Cannot delete doctor with
active appointments.');

  END IF;

END;

/

```
Package created.

0.00 seconds
Trigger created.

0.01 seconds
```

**Question:** Create a trigger that will automatically updates the total amount in the patient_bill table whenever a new item is added to the bill.

**Answer**: CREATE OR REPLACE TRIGGER update_total_amount

BEFORE INSERT ON patient_bill

FOR EACH ROW

BEGIN

:NEW.total_amount := :NEW.price * :NEW.quantity;

END;

/

```
Trigger created.

0.00 seconds
```

/

## 3 Package

**Question:** Create a package that provides functions to validate user credentials and retrieve user roles.

**Answer**: CREATE OR REPLACE PACKAGE authentication_pkg AS

FUNCTION is_valid_user(username VARCHAR2, password VARCHAR2) RETURN BOOLEAN;

FUNCTION get_user_role(username VARCHAR2) RETURN VARCHAR2;

END authentication_pkg;

```
/
Package created.

0.00 seconds
CREATE OR REPLACE PACKAGE BODY authentication_pkg AS

  FUNCTION is_valid_user(username VARCHAR2, password VARCHAR2) RETURN BOOLEAN IS

    user_count NUMBER;

  BEGIN

    SELECT COUNT(*) INTO user_count

    FROM admin

    WHERE ad_username = username AND ad_pass = password;


    RETURN user_count > 0;

  END;


  FUNCTION get_user_role(username VARCHAR2) RETURN VARCHAR2 IS

    user_role VARCHAR2(20);

  BEGIN

    SELECT ad_role INTO user_role

    FROM admin

    WHERE ad_username = username;


    RETURN user_role;

  END;

END authentication_pkg;

/.
```

```
Package Body created.

0.01 seconds
```

```
BEGIN
   medicine_pkg.update_medicine_price(3, 5.00);
   COMMIT;
END;
```

```
Statement processed.

0.00 seconds
```

**Question:** Create a package that includes Package patient_pkg for Patient Operations.

**Answer**:

```
CREATE OR REPLACE PACKAGE patient_pkg AS

    PROCEDURE add_patient(p_id INT, p_pname VARCHAR2,
   p_email VARCHAR2, p_address VARCHAR2, p_dpt VARCHAR2,
   p_phone VARCHAR2, p_problem VARCHAR2, p_shift VARCHAR2,
   p_age NUMBER, p_gender VARCHAR2);

    FUNCTION get_patient_info(p_id INT) RETURN VARCHAR2;

  END patient_pkg;
```

```
Package created.

0.00 seconds
```

```
CREATE OR REPLACE PACKAGE BODY patient_pkg AS

    PROCEDURE add_patient(p_id INT, p_pname VARCHAR2,
p_email VARCHAR2, p_address VARCHAR2, p_dpt VARCHAR2,
p_phone VARCHAR2, p_problem VARCHAR2, p_shift VARCHAR2,
p_age NUMBER, p_gender VARCHAR2) IS

    BEGIN
```

```
    INSERT INTO Patient VALUES (p_id, p_pname, p_email,
p_address, p_dpt, p_phone, p_problem, p_shift, p_age, p_gender);
    COMMIT;
  END;

  FUNCTION get_patient_info(p_id INT) RETURN VARCHAR2 IS
    patient_info VARCHAR2(200);
  BEGIN
    SELECT p_pname || ' - ' || p_dpt || ' - ' || p_shift
    INTO patient_info
    FROM Patient
    WHERE p_id = p_id;

    RETURN patient_info;
  END;
  PROCEDURE update_patient_contact(p_id INT, p_phone
VARCHAR2) IS
  BEGIN
    UPDATE Patient
    SET p_phone = p_phone
    WHERE p_id = p_id;
    COMMIT;
  END;
```

```
END patient_pkg;
```

```
Package Body created.

0.00 seconds
```

```
BEGIN
  medicine_pkg.update_medicine_price(3, 5.00);
  COMMIT;
END;
```

```
Statement processed.

0.00 seconds
```

**Question:** Create a package that includes a procedure to display the medicine's name based on the given id.

**Answer**:

```
CREATE OR REPLACE PACKAGE doctor_salary_pkg AS
   PROCEDURE calculate_doctor_salary(d_id INT);
   PROCEDURE update_doctor_salary(d_id INT, new_salary NUMBER);
END doctor_salary_pkg;
/
```

```
Package created.

0.00 seconds
```

```
CREATE OR REPLACE PACKAGE BODY doctor_salary_pkg AS
  PROCEDURE calculate_doctor_salary(d_id INT) IS
    base_salary NUMBER;
    calculated_salary NUMBER;
```

```
    BEGIN
        SELECT d_salary INTO base_salary
        FROM doctor_salary
        WHERE d_id = d_id;

        -- Calculate the salary based on some formula
        calculated_salary := base_salary * 1.1;

        UPDATE doctor_salary
        SET d_salary = calculated_salary
        WHERE d_id = d_id;
    END;

    PROCEDURE update_doctor_salary(d_id INT, new_salary NUMBER) IS
    BEGIN
        UPDATE doctor_salary
        SET d_salary = new_salary
        WHERE d_id = d_id;
    END;
END doctor_salary_pkg;
/
```

```
Package Body created.

0.00 seconds
```

```
BEGIN
    medicine_pkg.update_medicine_price(3, 5.00);
    COMMIT;
END;
```

```
Statement processed.

0.00 seconds
```

# 12.Relational Algebra:

**Question:** Retrieve the Doctors users who are male.

**Answer:**

Selection:

$\sigma_{(d\_gender='Male')(Doctors)}$

**Question:** Retrieve the names and email addresses of the Appointment users.

**Answer:**

Projection:

$\pi_{(a\_pname,\ a\_email)}(Appointment)$

**Question:** Retrieve the unique email addresses from the Doctors and Appointment tables.

**Answer:**

Union:

$Doctors \cup Appointment$

**Question**: Retrieve all possible combinations of Appointment and Patient users.

**Answer**:

Cartesian Product:

$Appointment \times Patient$

**Question**: Rename the "Doctors" table to "Doctor".

**Answer:**

Rename:

$\rho_{(Doctor)(Doctors)}$

# 13.Conclusion:

The Doctor Appointment System introduces an innovative solution to address the diverse challenges faced by patients, medical professionals, and administrators throughout the appointment coordination process. By harnessing the potential of technology and intuitive interfaces, this system stands ready to revolutionize the landscape of healthcare appointment scheduling, ultimately leading to heightened patient satisfaction and a more efficient utilization of resources. As we continue the current phase of designing the system, our aspirations extend to its complete implementation, incorporating a range of additional features. We are resolutely dedicated to refining the system based on user input, with a particular focus on enhancing its user-friendliness. Our overarching vision also encompasses the integration of historical patient data within the system, enabling seamless patient identification during subsequent visits. Moreover, our strategy involves actively updating patients on their treatment progress through notifications sent via messages or email.