

# PHARMACY MANAGEMENT SYSTEM



# Abstract:

## Introduction:

The main aim of the project is to manage the database of a pharmaceutical shop efficiently. We acquire this aim by creating a database of the available medicines, suppliers of those medicines, employees working, the patients who visit the pharmacy, and the doctors of those patients.

## Objective:

The purpose of this is to improve the maintenance of the medical data in the Pharmacy. The pharmacist will use the pharmacy management system to minimize the time and resources by maintaining the details of the medication and the customers(patients) systemically so that the data can be used in the possible quickest time. At the same time, the resources which are minimized are workforce, money, papers, etc. The system is user-friendly and will help the pharmacist. This Pharmacy Management System will reduce the pharmacist's burden and make the system efficient by providing more accurate details about medicines, patients visited, and the employees working. Also, it is a user-friendly application for pharmacists, which reduces the burden and helps to manage all sections of Pharmacy like Medicine management, Billing, Patient Data, etc.

## Entities and Attributes:

Pharmacy : PharmacyID(Primary Key), PharmacyName, PharmacyAddress, PhoneNumber

Employee : EmployeeID, EmployeeName, EmployeeAddress

Medicine : MedicineID(Primary Key), ExpiryDate, Name

Bill : BillID(Primary Key), Quantity, TotalAmt

Patient : PatientID(Primary Key), PatientName, Gender, PatientAddress, DOB, Age

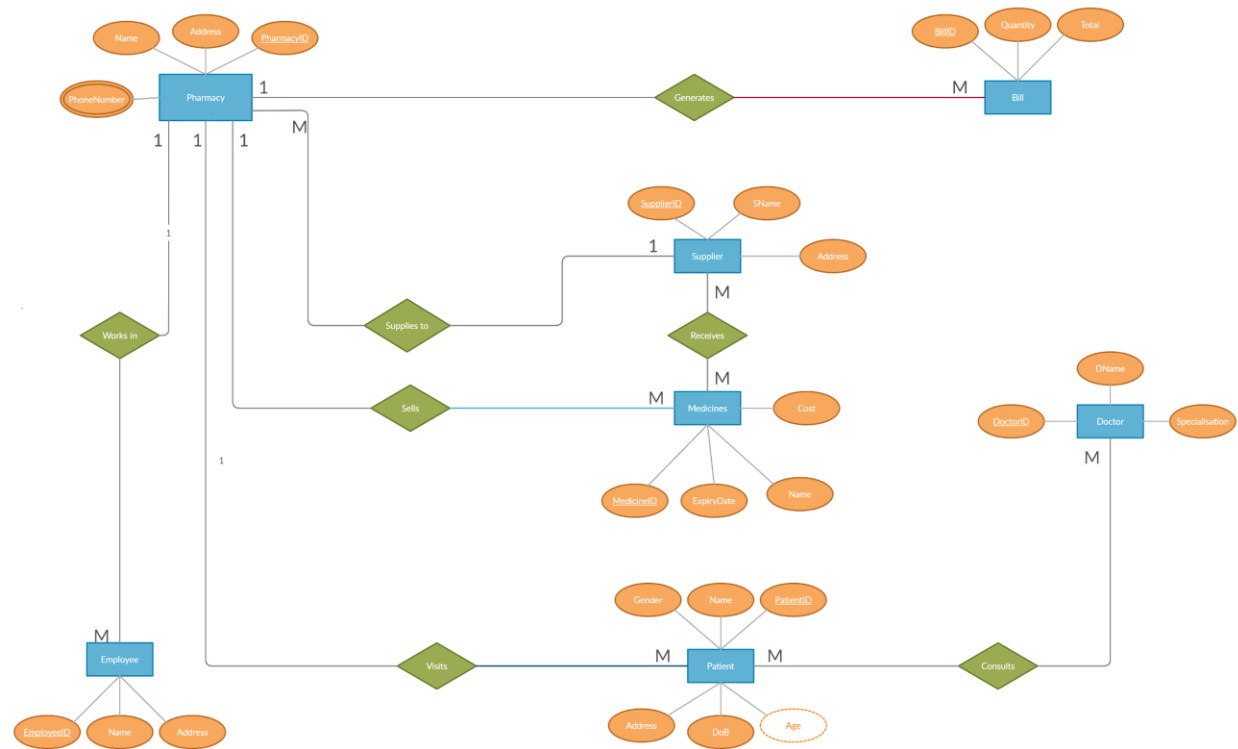
Supplier : SupplierID(Primary Key), SName, Address

Doctor : DoctorID(Primary Key), DName, Specialisation

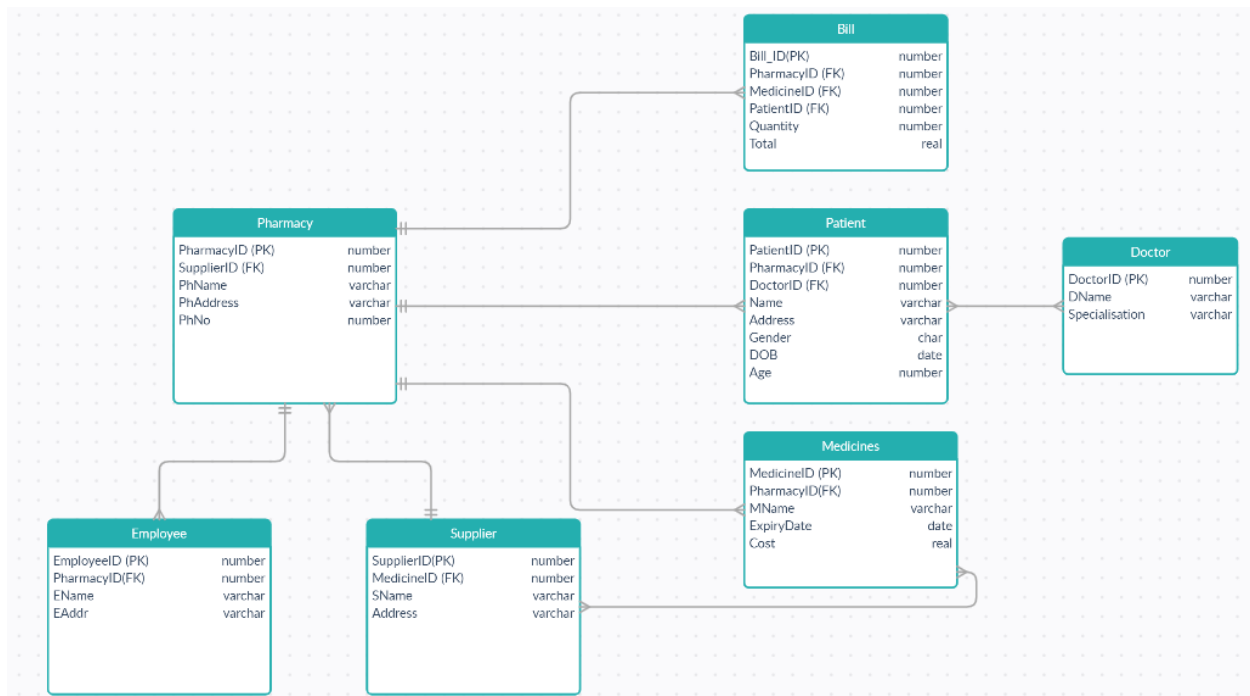
### **Relationships:**

- 1) Employee **WorksIn** Pharmacy
- 2) Pharmacy **Generates** Bills
- 3) Patient **Visits** Pharmacy
- 4) Pharmacy **Sells** Medicines
- 5) Supplier **Receives** Medicines from Manufacturer and **Supplies to** Pharmacy
- 6) Patient **Consults** the Doctor.

# ER Diagram:



# Relational/Conceptual Schema:



# Creation of Tables:

## Medicines

### Creation:

*create table medicines(mid number primary key,mname varchar(20),expirydate date,cost real);*

### Insertion:

*insert into medicines values(11,'Dolo',DATE '2021-05-15',25);*

*insert into medicines values(12,'Paracetamol',DATE '2021-09-25',10);*

*insert into medicines values(13,'Combiflam',DATE '2021-08-09',35);*

*insert into medicines values(14,'Crocine Advanced',DATE '2021-02-09',45);*

*insert into medicines values(15,'Covifor',DATE '2021-01-21',20);*

*insert into medicines values(16,'Saridon',DATE '2021-11-14',5);*

MID	MNAME	EXPIRYDATE	COST
11	Dolo	15-MAY-21	25
12	Paracetamol	25-SEP-21	10
13	Combiflam	09-AUG-21	35
14	Crocine Advanced	09-FEB-21	45
15	Covifor	21-JAN-21	20
16	Saridon	14-NOV-21	5

Download CSV

6 rows selected.

---

## **Supplier**

### **Creation:**

*create table supplier(sid number primary key, mid number, sname varchar(20), saddr varchar(20), foreign key(mid) references medicines);*

### **Insertion:**

*insert into supplier values(1221,11,'K. A. Paul','Rushikonda');*

*insert into supplier values(1331,12,'R. G. V','Maharanipeta');*

*insert into supplier values(1441,13,'Rakesh','MVP');*

*insert into supplier values(1551,14,'Shekhar','Seethamadhara');*

*insert into supplier values(1661,15,'Johnny','RTC Complex');*

*insert into supplier values(1771,16,'Harsha','Dwarakanagar');*

*insert into supplier values(1881,12,'Binod','OldPost Office');*

SID	MID	SNAME	SADDR
1221	11	K. A. Paul	Rushikonda
1331	12	R. G. V	Maharanipeta
1441	13	Rakesh	MVP
1551	14	Shekhar	Seethamadhara
1661	15	Johnny	RTC Complex
1771	16	Harsha	Dwarakanagar
1881	12	Binod	OldPost Office

Download CSV

7 rows selected.

## **Pharmacy**

### **Creation:**

*create table pharmacy(phid number primary key,sid number,phname varchar(20),phaddress varchar(30),phno number, foreign key(sid) references supplier);*

### **Insertion:**

*insert into pharmacy values(121,1221,'GIMSR','Rushikonda',9834523519);*

*insert into pharmacy values(122,1331,'KGH','Maharanipeta',9834523524);*

*insert into pharmacy values(123,1441,'Apollo','MVP',9834523520);*

*insert into pharmacy values(124,1551,'Seven Hills','Seethamadhara',9834523530);*

*insert into pharmacy values(125,1661,'KIMS','RTC Complex',9834523547);*

*insert into pharmacy values(126,1771,'Kala','Dwarakanagar',9834523551);*

*insert into pharmacy values(127,1881,'MedPlus','OldPost Office',9834523554);*



PHID	SID	PHNAME	PHADDRESS	PHNO
121	1221	GIMSR	Rushikonda	9834523519
122	1331	KGH	Maharanipeta	9834523524
123	1441	Apollo	MVP	9834523520
124	1551	Seven Hills	Seethamadhara	9834523530
125	1661	KIMS	RTC Complex	9834523547
126	1771	Kala	Dwarakanagar	9834523551
127	1881	MedPlus	OldPost Office	9834523554

Download CSV

7 rows selected.

## **Doctor**

### **Creation:**

*create table doctor(did number primary key, dname varchar(20), specialisation varchar(20));*

### **Insertion:**

*insert into doctor values(1234,'Newton','Orthopedic');*

*insert into doctor values(1235,'Trump','Dermatologist');*

*insert into doctor values(1236,'Osama','Neurologist');*

*insert into doctor values(1237,'Abraham','Cardiologist');*

*insert into doctor values(1238,'Kasab','ENT');*

DID	DNAME	SPECIALISATION
1234	Newton	Orthopedic
1235	Trump	Dermatologist
1236	Osama	Neurologist
1237	Abraham	Cardiologist
1238	Kasab	ENT

Download CSV

5 rows selected.

## **Patient**

### **Creation:**

*create table patient(pid number primary key,phid number, did number, pname varchar(30),gender varchar(1),dob date,paddress varchar(30),foreign key(phid) references pharmacy,foreign key(did) references doctor);*

### **Insertion:**

*insert into patient values(1,121,1234,'Gowtham','M','15-May-2001','Maharanipeta');*

*insert into patient values(2,122,1235,'Mahesh','M','25-Sep-2000','RTC Complex');*

*insert into patient values(3,123,1236,'Saai Ram','M','09-Aug-2001','OldPost Office');*

*insert into patient values(4,124,1237,'Anudeep','M','09-Feb-2001','MVP');*

*insert into patient values(5,125,1238,'Krishna,Vamsi','M','21-Jan-2002','Seethamadhara');*

*insert into patient values(6,126,1237,'Varun','M','14-Nov-2000','Rushikonda');*

*insert into patient values(7,127,1235,'Neeraj','M','09-Feb-2001','Dwarakanagar');*

PID	PHID	DID	PNAME	GENDER	DOB	PADDRESS
1	121	1234	Gowtham	M	15-MAY-01	Maharanipeta
2	122	1235	Mahesh	M	25-SEP-00	RTC Complex
3	123	1236	Saaii Ram	M	09-AUG-01	OldPost Office
4	124	1237	Anudeep	M	09-FEB-01	MVP
5	125	1238	Krishna,Vamsi	M	21-JAN-02	Seethamadhara
6	126	1237	Varun	M	14-NOV-00	Rushikonda
7	127	1235	Neeraj	M	09-FEB-01	Dwarakanagar

Download CSV

7 rows selected.

---

## **Employee**

### **Creation:**

*create table employee(eid number primary key,phid number,ename varchar(20),eaddress varchar(20),foreign key(phid) references pharmacy);*

### **Insertion:**

*insert into employee values(128,121,'Jack','Rushikonda');*

*insert into employee values(129,122,'Jill','Maharanipeta');*

*insert into employee values(130,123,'Rohit','MVP');*

*insert into employee values(131,124,'Sachin','Seethamadhara');*

*insert into employee values(132,125,'Virat','RTC Complex');*

*insert into employee values(133,126,'NTR','Dwarakanagar');*

*insert into employee values(134,127,'Rahul','OldPost Office');*

*insert into employee values(135,122,'Salman','Seethamadhara');*

*insert into employee values(136,124,'Pavan','RTC Complex');*

*insert into employee values(137,125,'Arjun','Dwarakanagar');*

*insert into employee values(138,123,'Prabhas','OldPost Office');*

## Queries:

### ALTER:

ALTER TABLE doctor ADD (daddr varchar(20));

### Output:

Table altered.

DID	DNAME	SPECIALISATION	DADDR
1234	Newton	Orthopedic	-
1235	Trump	Dermatologist	-
1236	Osama	Neurologist	-
1237	Abraham	Cardiologist	-
1238	Kasab	ENT	-

ALTER TABLE doctor RENAME column daddr to daddress;

### Output:

Table altered.

DID	DNAME	SPECIALISATION	DADDRESS
1234	Newton	Orthopedic	-
1235	Trump	Dermatologist	-
1236	Osama	Neurologist	-
1237	Abraham	Cardiologist	-
1238	Kasab	ENT	-

ALTER TABLE doctor MODIFY(daddress varchar(30));

### Output:

Table altered.

DID	DNAME	SPECIALISATION	DADDRESS
1234	Newton	Orthopedic	-
1235	Trump	Dermatologist	-
1236	Osama	Neurologist	-
1237	Abraham	Cardiologist	-
1238	Kasab	ENT	-

ALTER TABLE doctor DROP column address;

**Output:**

Table altered.

DID	DNAME	SPECIALISATION
1234	Newton	Orthopedic
1235	Trump	Dermatologist
1236	Osama	Neurologist
1237	Abraham	Cardiologist
1238	Kasab	ENT

**UPDATE:**

update patient set dob='29-jul-2001' WHERE pname='Neeraj';

**Output:**

1 row(s) updated.

PID	PHID	DID	PNAME	GENDER	DOB	PADDRESS
1	121	1234	Gowtham	M	15-MAY-01	Maharanipeta
2	122	1235	Mahesh	M	25-SEP-00	RTC Complex
3	123	1236	Saai Ram	M	09-AUG-01	OldPost Office
4	124	1237	Anudeep	M	09-FEB-01	MVP
5	125	1238	Krishna,Vamsi	M	21-JAN-02	Seethamadhara
6	126	1237	Varun	M	14-NOV-00	Rushikonda
7	127	1235	Neeraj	M	29-JUL-01	Dwarakanagar

**INSERT:**

insert into doctor values(1239,'Raj','RMP');

**Output:**

1 row(s) inserted.  
 EXECUTED THE DML OPERATION FOR DOCTOR TABLE  
 EXECUTED THE DML OPERATION FOR DOCTOR TABLE

DID	DNAME	SPECIALISATION
1239	Raj	RMP
1234	Newton	Orthopedic
1235	Trump	Dermatologist
1236	Osama	Neurologist
1237	Abraham	Cardiologist
1238	Kasab	ENT

### **DELETE:**

delete FROM doctor WHERE did=1239;

### **Output:**

1 row(s) deleted.  
 EXECUTED THE DML OPERATION FOR DOCTOR TABLE  
 EXECUTED THE DML OPERATION FOR DOCTOR TABLE

DID	DNAME	SPECIALISATION
1234	Newton	Orthopedic
1235	Trump	Dermatologist
1236	Osama	Neurologist
1237	Abraham	Cardiologist
1238	Kasab	ENT

### **ALL:**

SELECT \* FROM medicines WHERE cost>ALL(SELECT cost FROM medicines WHERE cost<10);

### **Output:**

MID	MNAME	EXPIRYDATE	COST
12	Paracetamol	25-SEP-21	10
15	Covifor	21-JAN-21	20
11	Dolo	15-MAY-21	25
13	Combiflam	09-AUG-21	35
14	Crocine Advanced	09-FEB-21	45

### **AND:**

SELECT \* FROM employee WHERE eaddress='RTC Complex' AND eid=132;

### **Output:**

EID	PHID	ENAME	EADDRESS
132	125	Virat	RTC Complex

**NOT IN:**

SELECT \* FROM pharmacy p WHERE p.phid NOT IN (SELECT b.phid FROM bill b);

**Output:**

PHID	SID	PHNAME	PHADDRESS	PHNO
122	1331	KGH	Maharanipeta	9834523524
126	1771	Kala	Dwarakanagar	9834523551

**ANY:**

SELECT \* FROM medicines WHERE cost=ANY(10,20);

**Output:**

MID	MNAME	EXPIRYDATE	COST
12	Paracetamol	25-SEP-21	10
15	Covifor	21-JAN-21	20

**BETWEEN:**

SELECT \* FROM patient WHERE dob BETWEEN '15-may-2000' and '15-may-2001';

**Output:**

PID	PHID	DID	PNAME	GENDER	DOB	PADDRESS
1	121	1234	Gowtham	M	15-MAY-01	Maharanipeta
2	122	1235	Mahesh	M	25-SEP-00	RTC Complex
4	124	1237	Anudeep	M	09-FEB-01	MVP
6	126	1237	Varun	M	14-NOV-00	Rushikonda

**IN:**

SELECT \* FROM supplier WHERE MID IN (11,12);

**Output:**

SID	MID	SNAME	SADDR
1221	11	K. A. Paul	Rushikonda
1331	12	R. G. V	Maharanipeta
1881	12	Binod	OldPost Office

**LIKE:**

SELECT \* FROM patient WHERE pname LIKE '%m';

**Output:**

PID	PHID	DID	PNAME	GENDER	DOB	PADDRESS
1	121	1234	Gowtham	M	15-MAY-01	Maharanipeta
3	123	1236	Saaii Ram	M	09-AUG-01	OldPost Office

### **OR:**

SELECT \* FROM employee WHERE eaddress='Rushikonda' OR eaddress='MVP';

### **Output:**

EID	PHID	ENAME	EADDRESS
128	121	Jack	Rushikonda
130	123	Rohit	MVP

### **COUNT:**

SELECT COUNT (\*) FROM bill;

### **Output:**

COUNT(*)
5

### **AVG:**

SELECT AVG(total) FROM bill;

### **Output:**

AVG(TOTAL)
272

### **MAX:**

SELECT MAX(quantity) FROM bill;

### **Output:**

MAX(QUANTITY)
15

### **MIN:**

SELECT MIN(quantity) FROM bill;

### **Output:**

MIN(QUANTITY)
2

### **SUM:**

SELECT SUM(quantity) FROM bill;

### **Output:**

SUM(QUANTITY)
44

### **Difference between MAX and MIN :**

SELECT MAX(quantity)-MIN(quantity) FROM bill;



**Output:**

MAX(QUANTITY) - MIN(QUANTITY)
13

**NOT NULL:**

SELECT dname FROM doctor WHERE specialisation is NOT NULL;

**Output:**

DNAME
Newton
Trump
Osama
Abraham
Kasab

**NULL:**

SELECT dname FROM doctor WHERE specialisation is NULL;

**Output:**

no data found

**NESTED QUERY:**

SELECT \* FROM pharmacy p WHERE p.sid=(SELECT sid FROM supplier WHERE sname='Binod');

**Output:**

PHID	SID	PHNAME	PHADDRESS	PHNO
127	1881	MedPlus	OldPost Office	9834523554

**CORELATED NESTED QUERIES:**

SELECT p.pid,p.pname FROM patient p WHERE EXISTS(SELECT \* FROM doctor d WHERE d.did=p.did and d.dname='Newton');

**Output:**

PID	PNAME
1	Gowtham

**VIEWS:**

CREATE VIEW pview(p\_id,p\_name) as SELECT pid,pname FROM patient;

**Output :**

View created.

---

P_ID	P_NAME
1	Gowtham
2	Mahesh
3	Saaii Ram
4	Anudeep
5	Krishna,Vamsi
6	Varun
7	Neeraj

**Order By:**

SELECT \* FROM medicines order by cost;

**Output:**

MID	MNAME	EXPIRYDATE	COST
16	Saridon	14-NOV-21	5
12	Paracetamol	25-SEP-21	10
15	Covifor	21-JAN-21	20
11	Dolo	15-MAY-21	25
13	Combiflam	09-AUG-21	35
14	Crocine Advanced	09-FEB-21	45

### PL/SQL:

1)

//Medicines names and its cost and Total cost//

```
declare
type medname IS VARRAY(5) of varchar(20);
type medcost IS VARRAY(5) of number;
costs medcost;
names medname;
sums number;
begin
sums:=0;
costs := medcost(20,21,22,23,24);
names := medname('Dolo','Paracetamol','Combiflam','Crocine','Saridon');
for i in 1..5 loop
dbms_output.put_line(names(i) || ' ' ||costs(i));
end loop;
for i in 1..5 loop;
sums:= sums+costs(i);
end loop;
dbms_output.put_line('Total Cost : ' ||sums);
end;
```

2)

//Details using switch case//

```
declare
options number(2)
begin
dbms_output.put_line('MENU');
dbms_output.put_line('1.Employee Details');
dbms_output.put_line('2.Medicine Details');
option := &option;
case option
when 1 then
select * from employee;
when 2 then
select * from medicines;
else
dbms_output.put_line('Doesn't exist');
end case;
end;
```

# Cursors and Triggers:

## Cursors :

```
DECLARE
  CURSOR c2 IS SELECT eid,ename FROM employee;
  ceid employee.eid%type;
  cename employee.ename%type;
BEGIN
  open c2;
LOOP
  FETCH c2 INTO ceid,cename;
  EXIT WHEN c2%notfound;
  dbms_output.put_line('Employee ' ||cename||' id is '||ceid);
END LOOP;
  close c2;
END;
/
```

```
Statement processed.
Employee  Jack   id is 128
Employee  Jill   id is 129
Employee  Rohit  id is 130
Employee  Sachin id is 131
Employee  Virat  id is 132
Employee  NTR    id is 133
Employee  Rahul  id is 134
Employee  Salman id is 135
Employee  Pavan  id is 136
Employee  Arjun  id is 137
Employee  Prabhas id is 138
```

```
DECLARE
  CURSOR c3 IS SELECT sid,sname FROM supplier;
  csid supplier.sid%type;
  csname supplier.sname%type;
BEGIN
  open c3;
LOOP
  FETCH c3 INTO csid,csname;
  EXIT WHEN c3%notfound;
  dbms_output.put_line('Supplier '||csname||' id is '||csid);
END LOOP;
  close c3;
END;
/
```

```
Statement processed.
Supplier K. A. Paul id is 1221
Supplier R. G. V id is 1331
Supplier Rakesh id is 1441
Supplier Shekhar id is 1551
Supplier Johnny id is 1661
Supplier Harsha id is 1771
Supplier Binod id is 1881
```

## **Triggers :**

```
CREATE or REPLACE TRIGGER doctor_trigger
BEFORE INSERT OR UPDATE OR DELETE ON doctor
BEGIN
dbms_output.put_line('EXECUTED THE DML OPERATION FOR DOCTOR TABLE');
END;
```

```
INSERT INTO doctor VALUES(101,'kilbil','rmp');
```

```
UPDATE doctor
SET did=102
WHERE did=101;
```

```
DELETE FROM doctor WHERE did=102;
```

```
SELECT * FROM doctor;
```

```
Trigger created.
```

```
1 row(s) inserted.
EXECUTED THE DML OPERATION FOR DOCTOR TABLE
EXECUTED THE DML OPERATION FOR DOCTOR TABLE
```

```
1 row(s) updated.
EXECUTED THE DML OPERATION FOR DOCTOR TABLE
EXECUTED THE DML OPERATION FOR DOCTOR TABLE
```

```
1 row(s) deleted.
EXECUTED THE DML OPERATION FOR DOCTOR TABLE
EXECUTED THE DML OPERATION FOR DOCTOR TABLE
```

DID	DNAME	SPECIALISATION
1234	Newton	Orthopedic
1235	Trump	Dermatologist
1236	Osama	Neurologist
1237	Abraham	Cardiologist
1238	Kasab	ENT

```
CREATE OR REPLACE TRIGGER supplier_trigger
BEFORE INSERT OR UPDATE OR DELETE ON supplier
BEGIN
dbms_output.put_line('EXECUTED THE DML OPERATION FOR SUPPLIER TABLE');
END;
```

```
INSERT INTO supplierVALUES(1000,13,'NAGARJUNA','GAJUVAKA');
```

```
UPDATE supplier
```

SET sid=1001

WHERE sid=1000;

DELETE FROM supplier WHERE sid=1001;

Trigger created.

1 row(s) inserted.

EXECUTED THE DML OPERATION FOR SUPPLIER TABLE

1 row(s) updated.

EXECUTED THE DML OPERATION FOR SUPPLIER TABLE

1 row(s) deleted.

EXECUTED THE DML OPERATION FOR SUPPLIER TABLE

SID	MID	SNAME	SADDR
1221	11	K. A. Paul	Rushikonda
1331	12	R. G. V	Maharanipeta
1441	13	Rakesh	MVP
1551	14	Shekhar	Seethamadhara
1661	15	Johnny	RTC Complex
1771	16	Harsha	Dwarakanagar
1881	12	Binod	OldPost Office

# Normalization:

Let us consider the following table and the functional dependencies:

Phid	SID	PName	Addr	Phno
121	1221	GIMSR	Rushikonda	9834523519
122	1331	KGH	Maharanipeta	9834523524
123	1441	Appolo	MVP	9834523520
124	1551	Seven Hills	Seethamadhara	9834523530
125	1661	Kims	RTC Complex	9834523547
126	1771	Kala	Dwarakanaga	9834523551
127	1881	MedPlus	Old Post Office	9834523554

Pharmacy(PhID,SID,PhName,Addr,PhNo)

FD: {**PhID,SID** -> PhName, **PhID,SID**->Addr, **PhID,SID**->PhNo }

Closure of PhID, SID is {PhID,SID,PhName, Addr, PhNo}

Candidate Key: {(PhID,SID)}

Prime Attributes are {PhID,SID}. Non-prime attributes are {PName,Addr,PhNo}.

**1NF:** Since the above table does not contain any multi-valued attribute, the table is in 1NF.

**2NF:** Conditions for 2NF are-

- i) Table should be in 1NF.
- ii) There should be no Partial Dependency.

Now we'll check for Partial Dependency. The condition for Partial Dependency is:  
LHS of all FDs should be a proper subset of Candidate Key or Super Key and RHS contains Non-Prime Attribute.

FDs are: {**PhID,SID** -> PhName, **PhID,SID**->Addr, **PhID,SID**->PhNo }

PhID,SID->PhName: LHS is not a proper subset of Candidate key, so there is no partial dependency in this FD.

PhID,SID->Addr: LHS is not a proper subset of Candidate key, so there is no partial dependency in this FD.

PhID,SID->PhNo: LHS is not a proper subset of Candidate key, so there is no partial dependency in this FD.



Hence we can say that all the non-prime attributes are fully functional dependent on Candidate Key.

Hence the table is in 2NF.

**3NF**: Conditions for 3NF are-

- i) Table should be in 2NF.
- ii) There should be no Transitive Dependency.

We can check for 3NF using the condition: LHS should be a Candidate Key or Super Key OR RHS should be a Prime Attribute.

FDs are: {**PhID,SID** -> PhName, **PhID,SID**->Addr, **PhID,SID**->PhNo }

PhID,SID->PhName: LHS is a Candidate key, so there is no transitive dependency in this FD.

PhID,SID->Addr: LHS is a Candidate key, so there is no transitive dependency in this FD.

PhID,SID->PhNo: LHS is a Candidate key, so there is no transitive dependency in this FD.

Hence the table is in 3NF.

**BCNF**: The condition for BCNF is that the LHS of the FD should be a Candidate Key.

FDs are: {**PhID,SID** -> PhName, **PhID,SID**->Addr, **PhID,SID**->PhNo }

PhID,SID->PhName: LHS is a Candidate key.

PhID,SID->Addr: LHS is a Candidate key.

PhID,SID->PhNo: LHS is a Candidate key.

Hence the table is in BCNF.

**Hence, the highest normal form of the pharmacy relation is BCNF.**

Now let us consider the below relation and FDs:

MID	PhID	MName	ExpiryDate	Cost
11	123	Dolo	15-05-2021	25
12	124	Paracetamol	25-09-2021	10
13	125	Combiflam	09-08-2021	35
14	126	Crocine Advanced	09-02-2021	45
15	127	Covifor	21-01-2021	20
16	121	Saridon	14-11-2021	5

Medicines(MID,PhID,MName,ExpiryDate,Cost)

FDs: { **MID**->PhID, **MID**-> MName, **MID,PhID**->ExpiryDate, **MID,PhID**->Cost,**PhID**-> MID }

Closure of MID is {MID,PhID,MName,ExpiryDate,Cost}

Candidate Keys: { MID }

Prime Attributes are {MID}. Non-prime attributes are {PhID,MName,ExpiryDate,Cost}.

**1NF:** Since the above table does not contain any multi-valued attribute so, the table is in 1NF.

**2NF:** Conditions for 2NF are-

- Table should be in 1NF.
- There should be no Partial Dependency.

Now we'll check for Partial Dependency. The condition for Partial Dependency is:  
LHS of all FDs should be a proper subset of Candidate Key or Super Key and RHS contains Non-Prime Attribute.

FDs are: { **MID**->PhID, **MID**-> MName, **MID,PhID**->ExpiryDate, **MID,PhID**->Cost,**PhID**-> MID }

MID->PhID: LHS is not a proper subset of Candidate key, so there is no partial dependency in this FD.

MID->MName: LHS is not a proper subset of Candidate key, so there is no partial dependency in this FD.

MID,PhID->ExpiryDate: LHS is not a proper subset of Candidate key, so there is no partial dependency in this FD.

MID,PhID->Cost: LHS is not a proper subset of Candidate key, so there is no partial dependency in this FD.

PhID-> MID: LHS is not a proper subset of Candidate key, so there is no partial dependency in this FD.

Hence we can say that all the non-prime attributes are fully functional dependent on Candidate Key.

Hence the table is in 2NF.

**3NF**: Conditions for 3NF are-

- i) Table should be in 2NF.
- ii) There should be no Transitive Dependency.

We can check for 3NF using the condition: LHS should be a Candidate Key or Super Key OR RHS should be a Prime Attribute.

FDs are: { **MID**->PhID, **MID**-> MName, **PhID**->ExpiryDate, **MID,PhID**->Cost,**PhID**-> MID }

MID->PhID: LHS is a Candidate key, so there is no transitive dependency in this FD.

MID->MName: LHS is a Candidate key, so there is no transitive dependency in this FD.

MID,PhID->ExpiryDate: LHS is a Candidate key, so there is no transitive dependency in this FD.

MID,PhID->Cost: LHS is a Candidate key, so there is no transitive dependency in this FD

PhID-> MID: LHS is not a Candidate key, but RHS is a Prime Attribute, so there is no transitive dependency in this FD.

Hence the table is in 3NF.

**BCNF**: Condition for BCNF is that the LHS Of the FD should be a Candidate Key.

FDs are: { **MID**->PhID, **MID**-> MName, **PhID**->ExpiryDate, **MID,PhID**->Cost,,**PhID**-> MID }

MID->PhID: LHS is a Candidate key

MID->MName: LHS is a Candidate key.

MID,PhID->ExpiryDate: LHS is a Candidate key.

MID,PhID->Cost: LHS is a Candidate key

PhID-> MID: LHS is not a candidate key.

Hence the table is not in BCNF.

**Hence the highest normal form of medicines relation is 3NF.**

Now let us consider the below relation and FDs.

SID	MID	Sname	Saddr
1221	11	K. A. Paul	Rushikonda
1331	12	R. G. V	Maharanipeta
1441	13	Rakesh	MVP
1551	14	Shekhar	Seethamadhara
1661	15	Johnny	RTC Complex
1771	16	Harsha	Dwarakanagar
1881	12	Binod	Old Post Office

Supplier(SID,MID,SName,SAddr)

FDs are: { **SID**->SName, **SID,MID**-> SAddr}

Closure of SID,MID is: {SID,MID,SName,SAddr}

Candidate Key is (SID,MID)

Prime Attributes are:{SID,MID}. Non Prime Attributes are: {SName,SAddr}

**1NF**: Since the above table does not contain any multi-valued attribute so, the table is in 1NF.

**2NF**: Conditions for 2NF are-

- i) Table should be in 1NF.
- ii) There should be no Partial Dependency.

Now we'll check for Partial Dependency. The condition for Partial Dependency is:  
LHS of all FDs should be a proper subset of Candidate Key or Super Key and RHS contains Non-Prime Attribute.

FDs are: { **SID**->SName, **SID,MID**-> SAddr}

FDs are: { **MID**->PhID, **MID**-> MName, **MID,PhID**->ExpiryDate, **MID,PhID**->Cost,**PhID**-> MID }

SID->SName: LHS is a proper subset of Candidate key and RHS is a non-prime attribute, so there is no partial dependency in this FD.

SID,MID-> SAddr: LHS is not a proper subset of Candidate key, so there is no partial dependency in this FD.

Since, one non-prime attribute is partially dependent, hence the table is not in 2NF

**Hence the highest normal form of supplier relation is 1NF.**

# JDBC-ODBC:

## Code:

**Form Class:** This is the Parent class and is used to generate a basic form which takes User's Role.

```
import java.io.*;
import java.sql.*;
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import java.text.DateFormat;
import java.text.SimpleDateFormat;
class Form implements ActionListener
{

    static int y=70;
    static JRadioButton tb1,tb2,tb3,tb4,tb5,tb6,tb7;
    static JFrame f=new JFrame("Pharmacy");
    static JPanel p=new JPanel(new BorderLayout());
    static JPanel dp=new JPanel();
    static JPanel rp=new JPanel();
    static JPanel tp=new JPanel();

    static JRadioButton b1,b2,b3,b4;
    static JLabel l;
    static JPasswordField p1;
    char Password[]={'P','h','a','r','m','a','c','y'};
    static char r='D';
    static Form instance=new Form();
    static Statement stmt=null;
    static JButton eb;
    static void MakeCon()
    {
        try{
            Class.forName("com.mysql.jdbc.Driver");
            Connection con=DriverManager.getConnection(
                "jdbc:mysql://localhost:3308/pharmacymanagement","root","");
            stmt=con.createStatement();
        }catch(Exception e){ System.out.println(e);}
    }

    public static void GenerateLabel(JPanel px,String s,int x,int y,int x1,int y1)
    {
        l=new JLabel(s);
        Font myFont = new Font("Serif",Font.BOLD,25);
        l.setFont(myFont);
    }
}
```

```
l.setBounds(x,y,x1,y1);
px.add(l);
}
```

```
public static void GenerateNLabel(JPanel px,String s,int x,int y,int x1,int y1)
{
    l=new JLabel(s);
    Font myFont = new Font("Serif",Font.PLAIN,25);
    l.setFont(myFont);
    l.setBounds(x,y,x1,y1);
    px.add(l);
}
```

```
public static void GenerateButton(JPanel px,int x,int y,String s)
{
    JButton b=new JButton(s);
    b.setBounds(x,y,120,30);
    b.setActionCommand(s);
    b.addActionListener(instance);
    px.add(b);
}
```

```
public static void GenerateExit(JPanel px,int x,int y)
{
    eb=new JButton("Exit");
    eb.setBounds(x,y,120,30);
    eb.setActionCommand("Exit");
    eb.addActionListener(instance);
    eb.setBackground(Color.RED);
    px.add(eb);
}
```

```
public static void main()
{
    MakeCon();
    f.setDefaultCloseOperation(JFrame.HIDE_ON_CLOSE);
    f.setSize(10000,1000);
    GenerateLabel(p,"Choose your role",550,300,300,30);
    b1=new JRadioButton("Admin");
    b1.setBounds(550,330,100,30);
    b2=new JRadioButton("Receptionist");
    b2.setBounds(550,360,100,30);
    ButtonGroup bg=new ButtonGroup();
    bg.add(b1);bg.add(b2);
    p.add(b1);
    p.add(b2);

    JLabel pl=new JLabel("Enter your Password: ");
    pl.setBounds(450,400,150,20);
    p.add(pl);
    pl.setVisible(false);
    p1=new JPasswordField(20);
    p1.setBounds(600,400,150,30);
    p.add(p1);
}
```

```

p1.setVisible(false);

b1.addActionListener(new ActionListener(){
    @Override
    public void actionPerformed(ActionEvent e){
        p1.setVisible(true);
        pl.setVisible(true);
    }
});
b2.addActionListener(new ActionListener(){
    @Override
    public void actionPerformed(ActionEvent e){
        p1.setVisible(false);
        pl.setVisible(false);
    }
});
GenerateButton(p,580,450,"Next");
GenerateExit(p,580,600);
f.setSize(10000,1000);
f.setContentPane(p);
f.setLayout(null);
f.setVisible(true);
}

public void actionPerformed(ActionEvent e)
{
    if(e.getActionCommand()=="Next")
    {
        if(b1.isSelected())
        {
            if(Arrays.equals(p1.getPassword(),Password))
            {
                f.remove(p);
                f.setContentPane(dp);
                new DBA();
            }
            else
            {
                JOptionPane.showMessageDialog(null, "Incorrect Password");
            }
        }
        if(b2.isSelected())
        {
            f.remove(p);
            f.setContentPane(rp);
            new Recep();
        }
        f.validate();
        f.repaint();
    }
    if(e.getActionCommand()=="Continue")
    {
        if(b3.isSelected())
        {

```

```

        f.setContentPane(tp);
        new ShowTables();
        GenerateButton(tp,580,450,"Back");
        GenerateExit(tp,580,600);
    }
}
if(e.getActionCommand()=="Back")
{
    f.remove(tp);
    if(r=='D')
        f.setContentPane(dp);
    if(r=='R')
        f.setContentPane(rp);
    tp.removeAll();
    y=70;
}
if(e.getActionCommand()=="Back to Main")
{
    p1.setText("");
    f.setContentPane(p);
}
if(e.getActionCommand()=="Exit")
{
    int n = JOptionPane.showConfirmDialog(null,"Are you sure?","Confirmation",
JOptionPane.YES_NO_OPTION);
    if(n==0)
    {
        p.removeAll();
        dp.removeAll();
        rp.removeAll();
        tp.removeAll();
        f.dispose();
    }
}
}
}

```



**DBA Class:** This class is used to generate a screen for the Admin.

```
import javax.swing.*.*;
import java.awt.*.*;
import java.awt.event.*;
public class DBA extends Form
{
    public DBA()
    {
        r='D';
        f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        GenerateLabel(dp,"Enter your Choice",500,100,300,30);
        b3=new JRadioButton("Show Table Data");
        b3.setBounds(500,140,200,30);
        b4=new JRadioButton("Update Table Data");
        b4.setBounds(500,180,200,30);
        b4.setVisible(false);
        b3.setSelected(true);
        ButtonGroup bg=new ButtonGroup();
        bg.add(b3);bg.add(b4);
        dp.add(b3);
        dp.add(b4);
        GenerateLabel(dp,"Choose the table:",500,250,300,30);
        tb1=new JRadioButton("Patient");
        tb1.setBounds(500,280,100,30);
        tb2=new JRadioButton("Medicines");
        tb2.setBounds(500,310,100,30);
        tb3=new JRadioButton("Bill");
        tb3.setBounds(500,340,100,30);
        tb4=new JRadioButton("Employee");
        tb4.setBounds(500,370,100,30);
        tb5=new JRadioButton("Pharmacy");
        tb5.setBounds(500,400,100,30);
        tb6=new JRadioButton("Supplier");
        tb6.setBounds(500,430,100,30);
        tb7=new JRadioButton("Doctor");
        tb7.setBounds(500,460,100,30);
        ButtonGroup tbg=new ButtonGroup();
        tbg.add(tb1);    dp.add(tb1);
        tbg.add(tb2);    dp.add(tb2);
        tbg.add(tb3);    dp.add(tb3);
        tbg.add(tb4);    dp.add(tb4);
        tbg.add(tb5);    dp.add(tb5);
        tbg.add(tb6);    dp.add(tb6);
        tbg.add(tb7);    dp.add(tb7);

        GenerateButton(dp,600,500,"Continue");
        GenerateButton(dp,450,500,"Back to Main");
    }
}
```

```
GenerateExit(dp,520,600);
```

```
f.setSize(10000,1000);
```

```
f.setLayout(null);
```

```
f.setVisible(true);
```

```
}
```

```
}
```

**Recep Class:** This class is used to generate a screen for the receptionist.

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class Recep extends Form implements ActionListener
{
    public Recep()
    {
        r='R';
        f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        GenerateLabel(rp,"Enter your Choice",500,100,300,30);
        b3=new JRadioButton("Show Table Data");
        b3.setBounds(500,140,200,30);
        b4=new JRadioButton("Update Table Data");
        b4.setBounds(500,180,200,30);
        b4.setVisible(false);
        b3.setSelected(true);
        ButtonGroup bg=new ButtonGroup();
        bg.add(b3);bg.add(b4);
        rp.add(b3);
        rp.add(b4);
        GenerateLabel(rp,"Choose the table:",500,250,300,30);
        tb1=new JRadioButton("Patient");
        tb1.setBounds(500,280,100,30);
        tb2=new JRadioButton("Medicines");
        tb2.setBounds(500,310,100,30);
        tb3=new JRadioButton("Bill");
        tb3.setBounds(500,340,100,30);
        tb4=new JRadioButton("Employee");
        tb4.setBounds(500,370,100,30);
        tb5=new JRadioButton("Pharmacy");
        tb5.setBounds(500,400,100,30);
        tb6=new JRadioButton("Supplier");
        tb6.setBounds(500,430,100,30);
        tb7=new JRadioButton("Doctor");
        tb7.setBounds(500,460,100,30);
        ButtonGroup tbg=new ButtonGroup();
        tbg.add(tb1);    rp.add(tb1);
        tbg.add(tb2);    rp.add(tb2);
        tbg.add(tb3);    rp.add(tb3);
        tbg.add(tb4);    rp.add(tb4);
        tbg.add(tb5);    rp.add(tb5);
        tbg.add(tb6);    rp.add(tb6);
        tbg.add(tb7);    rp.add(tb7);
        //tb4.setVisible(false);
        //tb5.setVisible(false);
        tb6.setVisible(false);
        tb7.setVisible(false);
        b3.addActionListener(new ActionListener(){
            @Override
            public void actionPerformed(ActionEvent e){
                tb5.setVisible(true);
            }
        });
    }
}
```

```

        //tb6.setVisible(true);
        //tb7.setVisible(true);
        tb4.setVisible(true);
    }
});
b4.addActionListener(new ActionListener(){
    @Override
    public void actionPerformed(ActionEvent e){
        tb5.setVisible(false);
        tb6.setVisible(false);
        tb7.setVisible(false);
        tb4.setVisible(false);
    }
});

GenerateButton(rp,600,500,"Continue");
GenerateButton(rp,450,500,"Back to Main");
GenerateExit(rp,520,600);
f.setSize(10000,1000);
f.setLayout(null);
f.setVisible(true);
}
}

```

**ShowTables Class:** This class is used to generate a screen which shows the table data.

```

import javax.swing.*.*;
import java.awt.*.*;
import java.sql.*;
import java.awt.event.*;
class ShowTables extends Form
{
    ShowTables()
    {
        if(tb1.isSelected())
            showPatient();
        else if(tb2.isSelected())
            showMedicines();
        else if(tb3.isSelected())
            showBill();
        else if(tb4.isSelected())
            showEmployee();
        else if(tb5.isSelected())

```

```

        showPharmacy();
    else if(tb6.isSelected())
        showSupplier();
    else if(tb7.isSelected())
        showDoctor();
    else
    {
        JOptionPane.showMessageDialog(null, "Please select a table");
        if(r=='D')
            f.setContentPane(dp);
        else
            f.setContentPane(rp);
    }
}

```

```

public static void showBill()
{
    f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    y=y+10;
    try{
        ResultSet rs=stmt.executeQuery("select * from bill;");
        GenerateLabel(tp,"BILL TABLE",570,0,500,50);
        GenerateLabel(tp,"BID",370,50,100,30);
        GenerateLabel(tp,"PhID",470,50,100,30);
        GenerateLabel(tp,"MID",570,50,100,30);
        GenerateLabel(tp,"PID",670,50,100,30);
        GenerateLabel(tp,"Quantity",770,50,100,30);
        GenerateLabel(tp,"Total",900,50,100,30);
        while(rs.next())
        {
            GenerateNLabel(tp,String.valueOf(rs.getInt(1)),370,y,100,20);
            GenerateNLabel(tp,String.valueOf(rs.getInt(2)),470,y,100,20);
            GenerateNLabel(tp,String.valueOf(rs.getInt(3)),570,y,100,20);
            GenerateNLabel(tp,String.valueOf(rs.getInt(4)),670,y,100,20);
            GenerateNLabel(tp,String.valueOf(rs.getInt(5)),770,y,100,20);
            GenerateNLabel(tp,String.valueOf(rs.getInt(6)),900,y,100,20);
            y=y+30;
        }

    }catch(Exception e){System.out.println(e);}
    f.setSize(10000,1000);
    f.setLayout(null);
    f.setVisible(true);
}

```

```

public static void showSupplier()
{
    f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    try{
        ResultSet rs=stmt.executeQuery("select * from supplier;");
        GenerateLabel(tp,"SUPPLIER TABLE",520,0,500,50);
        GenerateLabel(tp,"SID",370,50,100,30);
        GenerateLabel(tp,"MID",470,50,100,30);
        GenerateLabel(tp,"SName",570,50,200,30);
    }
}

```

```

        GenerateNLabel(tp,"SAddress",770,50,200,30);
        while(rs.next())
        {
            GenerateNLabel(tp,String.valueOf(rs.getInt(1)),370,y,100,50);
            GenerateNLabel(tp,String.valueOf(rs.getInt(2)),470,y,100,50);
            GenerateNLabel(tp,rs.getString(3),570,y,200,50);
            GenerateNLabel(tp,rs.getString(4),770,y,200,50);
            y=y+30;
        }
    }catch(Exception e){System.out.println(e);}
    f.setSize(10000,1000);
    f.setLayout(null);
    f.setVisible(true);
}

public static void showEmployee()
{
    try{
        ResultSet rs=stmt.executeQuery("select * from employee;");
        GenerateLabel(tp,"EMPLOYEE TABLE",520,0,500,50);
        GenerateLabel(tp,"EID",370,50,100,30);
        GenerateLabel(tp,"PhID",470,50,100,30);
        GenerateLabel(tp,"EName",570,50,200,30);
        GenerateLabel(tp,"EAddress",770,50,200,30);
        while(rs.next())
        {
            GenerateNLabel(tp,String.valueOf(rs.getInt(1)),370,y,100,50);
            GenerateNLabel(tp,String.valueOf(rs.getInt(2)),470,y,100,50);
            GenerateNLabel(tp,rs.getString(3),570,y,200,50);
            GenerateNLabel(tp,rs.getString(4),770,y,200,50);
            y=y+30;
        }
    }catch(Exception e){System.out.println(e);}
    f.setSize(10000,1000);
    f.setLayout(null);
    f.setVisible(true);
}

public static void showMedicines()
{
    try{
        ResultSet rs=stmt.executeQuery("select * from medicines;");
        GenerateLabel(tp,"Medicine TABLE",520,0,500,50);
        GenerateLabel(tp,"MID",370,50,100,30);
        GenerateLabel(tp,"MName",470,50,100,30);
        GenerateLabel(tp,"ExpiryDate",670,50,200,30);
        GenerateLabel(tp,"Cost",870,50,200,30);
        while(rs.next())
        {
            GenerateNLabel(tp,String.valueOf(rs.getInt(1)),370,y,100,50);
            GenerateNLabel(tp,rs.getString(2),470,y,200,50);
            GenerateNLabel(tp,rs.getDate(3).toString(),670,y,200,50);
            GenerateNLabel(tp,String.valueOf(rs.getInt(4)),870,y,100,50);
            y=y+30;
        }
    }
}

```

```

    }

    }catch(Exception e){System.out.println(e);}
    f.setSize(10000,1000);
    f.setLayout(null);
    f.setVisible(true);
}

public static void showDoctor()
{
    try{
        ResultSet rs=stmt.executeQuery("select * from doctor;");
        GenerateLabel(tp,"DOCTOR TABLE",520,0,500,50);
        GenerateLabel(tp,"DID",370,50,100,30);
        GenerateLabel(tp,"DName",470,50,200,30);
        GenerateLabel(tp,"Specialisation",670,50,200,30);
        while(rs.next())
        {
            GenerateNLabel(tp,String.valueOf(rs.getInt(1)),370,y,100,50);
            GenerateNLabel(tp,rs.getString(2),470,y,200,50);
            GenerateNLabel(tp,rs.getString(3),670,y,200,50);
            y=y+30;
        }

    }catch(Exception e){System.out.println(e);}
    f.setSize(10000,1000);
    f.setLayout(null);
    f.setVisible(true);
}

public static void showPharmacy()
{
    try{
        ResultSet rs=stmt.executeQuery("select * from pharmacy;");
        GenerateLabel(tp,"PHARMACY TABLE",570,0,500,50);
        GenerateLabel(tp,"PhID",370,50,100,30);
        GenerateLabel(tp,"SID",470,50,100,30);
        GenerateLabel(tp,"PhName",570,50,200,30);
        GenerateLabel(tp,"PhAddress",770,50,200,30);
        GenerateLabel(tp,"PhNo",970,50,100,30);
        while(rs.next())
        {
            GenerateNLabel(tp,String.valueOf(rs.getInt(1)),370,y,100,50);
            GenerateNLabel(tp,String.valueOf(rs.getInt(2)),470,y,100,50);
            GenerateNLabel(tp,rs.getString(3),570,y,200,50);
            GenerateNLabel(tp,rs.getString(4),770,y,200,50);
            GenerateNLabel(tp,String.valueOf(rs.getInt(5)),970,y,200,50);
            y=y+30;
        }

    }catch(Exception e){System.out.println(e);}
    f.setSize(10000,1000);
    f.setLayout(null);
    f.setVisible(true);
}

```

```

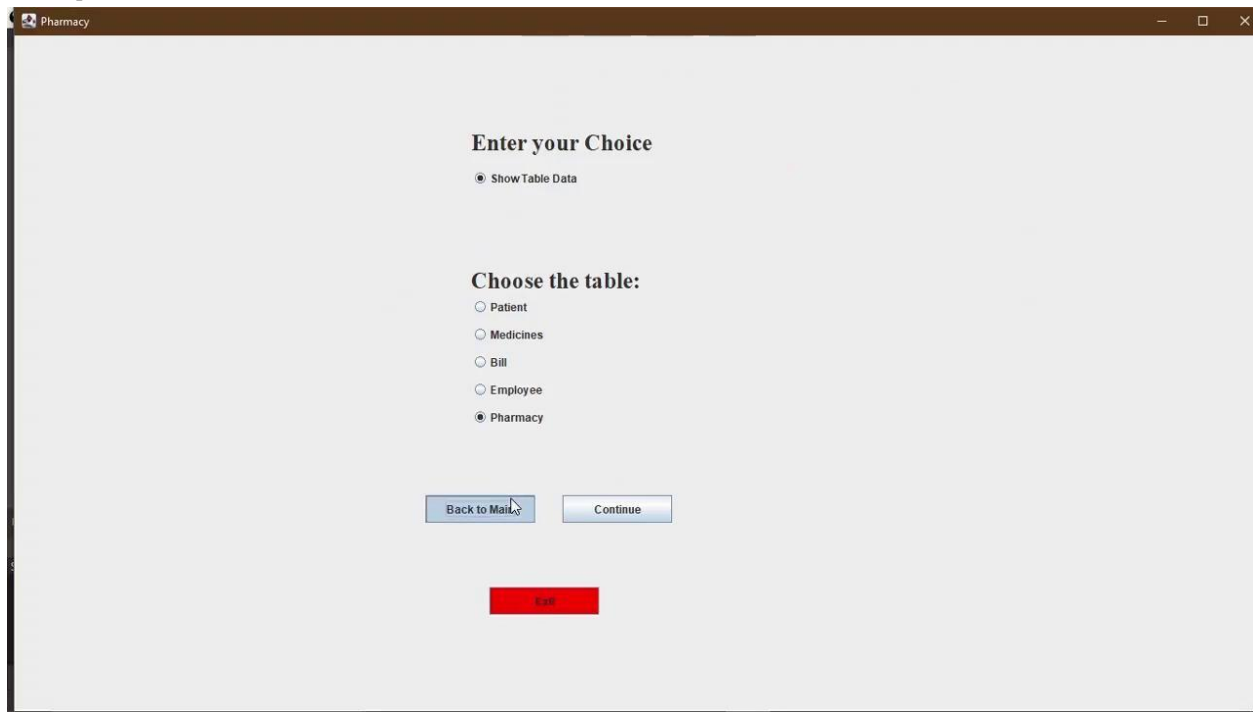
}

public static void showPatient()
{
    try{
        ResultSet rs=stmt.executeQuery("select * from patient;");
        GenerateLabel(tp,"PATIENT TABLE",570,0,500,50);
        GenerateLabel(tp,"PID",270,50,100,30);
        GenerateLabel(tp,"PhID",370,50,100,30);
        GenerateLabel(tp,"DID",470,50,100,30);
        GenerateLabel(tp,"PName",570,50,200,30);
        GenerateLabel(tp,"Gender",770,50,200,30);
        GenerateLabel(tp,"DOB",870,50,200,30);
        GenerateLabel(tp,"PAddress",1070,50,200,30);
        while(rs.next())
        {
            GenerateNLabel(tp,String.valueOf(rs.getInt(1)),270,y,100,50);
            GenerateNLabel(tp,String.valueOf(rs.getInt(2)),370,y,100,50);
            GenerateNLabel(tp,String.valueOf(rs.getInt(3)),470,y,100,50);
            GenerateNLabel(tp,rs.getString(4),570,y,200,50);
            GenerateNLabel(tp,rs.getString(5),770,y,200,50);
            GenerateNLabel(tp,rs.getDate(6).toString(),870,y,200,50);
            GenerateNLabel(tp,rs.getString(7),1070,y,200,50);
            y=y+30;
        }
    }catch(Exception e){System.out.println(e);}
    f.setSize(10000,1000);
    f.setLayout(null);
    f.setVisible(true);
}
}

```



## **Output:**



The screenshot shows a web application window with a title bar labeled "Pharmacy". The main content area has a light gray background. At the top, the text "Enter your Choice" is displayed. Below it, there is a radio button labeled "Show Table Data" which is selected. Further down, the text "Choose the table:" is shown, followed by five radio buttons: "Patient", "Medicines", "Bill", "Employee", and "Pharmacy". The "Pharmacy" radio button is selected. At the bottom, there are two buttons: "Back to Main" and "Continue". Below these buttons is a red button labeled "Save".

Note: Above is the video of the complete demonstration, kindly click on that and click on the redirecting link to view the complete output video.

Thank  
You