DATABASE MANAGEMENT SYSTEM

PROJECT REPORT:

Organ Donation and Procurement Network Management System

SUBMITTED BY:

SANCHI SINGH

2021UCS1565

PROBLEM STATEMENT:

Problem Statement:
Organ transplantation is a medical procedure in which an organ is removed from one body and placed in the body of a recipient, to replace a damaged or missing organ. The donor and recipient may be at the same location, or organs may be transported from a donor site to another location.
Organ Donation and Procurement Organizations play a pivotal role in today's medical institutions. Such organizations are responsible for the evaluation and procurement of organs for organ transplantation. These organizations represent the front-line of organ procurement, having direct contact with the hospital and the family of a recently deceased donor. The work of such organizations includes to identify the best candidates for the available organs and to coordinate with the medical institutions to decide on each organ recipient. They are also responsible for educating the public to increase the awareness of and participation in the organ donation process. Also, it keeps

track of all transplantation operations carried till date.

The Organ Donation and Procurement Network Management System is a database management system that uses database technology to construct, maintain and manipulate various kinds of data about a person's donation or procurement of a particular organ. It maintains a comprehensive medical history and other critical information like blood group, age, etc of every person in the database design. In short, it maintains a database containing statistical information regarding network of organ donation and procurement of different countries.

REQUIREMENT ANALYSIS:

REQUIREMENT ANALYSIS:

Organ Wastage is a major issue that can only be solved by having a proper database of all Patient and Donors in a well-formed way, that can be processed easily. Records of donor and patients are created when a person donates or procures an organ from a Medical Institution. Records may include the following information:-

1. Personal Information
2. Medical History
3. Medical insuarance, if any
4. Allergies to any medicine, if any
5. The need for an organ presently
5. Medical Insurance provided by any private or government insurers.
7. Address

This record serves a variety of purposes and is critical to the proper functioning of Organ Donation and Procurement Network, especially in today's complicated health care environment. These records provide statistical information regarding the number of organs needed and available at a particular point of time. It is essential for planning, evaluating and coordinating organ donation and procurement

Basic Steps in Implementation :

- Every user has an account with can only be registered by a government certified

hospital, which will keep all the information as defined in Problem Statement.

- Only Hospitals are eligible to request for a donation or procurement

transaction.

• Government organizations will keep a watch on the pairing of donors and Patients and can approve a transplantation operation if all the rules are satisfied.

• Collecting Statistical Data through the history of Transplantation Transaction.

ER Analysis: Identifying Entity Sets and Relationship Sets:

Entity Sets:

1. USER:

1. User ID
2. Name
3. Date of birth
4. Phone Number (multi-valued)
5. Medical Insurance
6. Medical History
7. Address

2. PATIENT:

1. Patient_ID
2. Organ Required
3. Reason of procurement
4. User_ID (foreign key)

3. DONOR:

1. Donor_ID

2. Organ Donated

3. Reason of donation

4. User_ID (foreign key)

4. Organ Available:

1. Organ_ID

2. Organ Name

3. Donor_ID (foreign key)

5. Organization:

1. Organization ID

2. Organization Name

3. Location

4. Government approved organization or not

5. Phone Number (multi-valued)

6. Doctor:

1. Doctor ID

2. Doctor Name

3. Phone Number (multi-valued)

7. Organization Head:

1. Head Name

2. Date of Joining

3. Term Length

Relationship Sets:

1. Donates – The act of donation of an organ from a donor

1. Date - Date of donation

2. Procures - The act of procuring an organ by the patient

3. Transaction

1. Date of transaction

2. Status – whether the surgery was successful or not

4. Organ Donated -The organ donated by an donor, which is then stored in Organ_available table.

5. Attended By -The transplantation performed by doctor – procuring an organ from a donor and transplanting it to the patient by surgery.

6. Registers - Donor is registered in which organization

7. Works in – The organization where the doctor works.

8. Headed By – The organization is headed by which person

FUNCTIONAL DEPENDENCIES:

Tables and their Functional Dependencies :-

1) User(User_ID, Name, Date_of_birth, Medical_Insurance, Medical_History, Street, City, State)
FD={User_ID → Name, Date_of_birth, Medical Insurance, Medical History, Street, City, State}

2) User_phone_no(User_ID, phone_no)
FD={User_ID -> phone_no}
{User_ID} is foreign key constraint

3) Patient(Patient_ID, organ_req, reason_of_procurement, Doctor_ID,User_ID)
FD={Patient_ID, organ_req -> reason_of_procurement, Doctor_ID, User_ID}
{User_ID, Doctor_ID} are foreign key constraints

4) Donor(Donor_ID, organ_donated, reason_of_donation, Organization_ID, User_ID)
FD={Donor_ID, organ_donated -> reason_of_donation, Organization_ID, User_ID}
{User_ID, Organization_ID} are foreign key constraints

5) Organ Available(Organ_ID,Organ_name, Donor_ID)
FD={Organ_ID -> Organ_name,Donor_ID}

{Donor_ID} is foreign key constraint

6) Transaction(Patient_ID, Organ_ID, Donor_ID, Date_of_transaction, Status)
FD={Patient_ID, Organ_ID -> Donor_ID,Date_of_transaction, Status}
{Patient_ID, Donor_ID} are foreign 'ey constraints

7) Organi-ation(Organization_ID, Organization_name, Location, Government_approved)
FD={Organization_ID -> Organization_name, Location, Government_approved}

8) Organization_phone_no(Organization_ID, phone_no)
FD={Organization_ID -> phone_no}
{Organization_ID} are foreign key constraints

9) Doctor(Doctor_ID, Doctor_name, Department_name, Organization_id)
FD={Doctor_ID -> Doctor_name, Organization_id}
{Organization_ID} is foreign key constraint

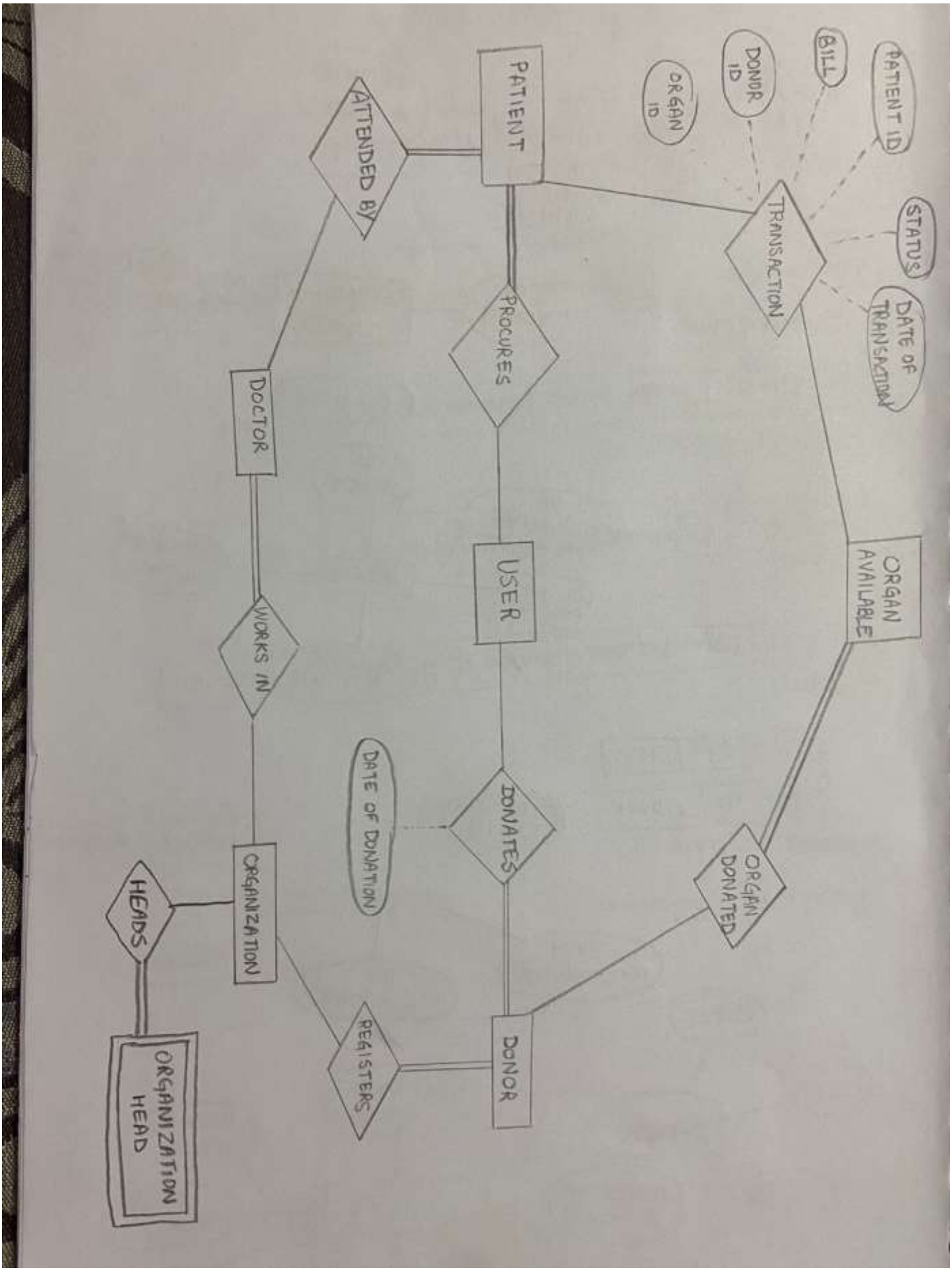
10) Doctor_phone_no(Doctor_ID, phone_no)
FD={Doctor_ID -> phone_no}
{Doctor_ID} is foreign key

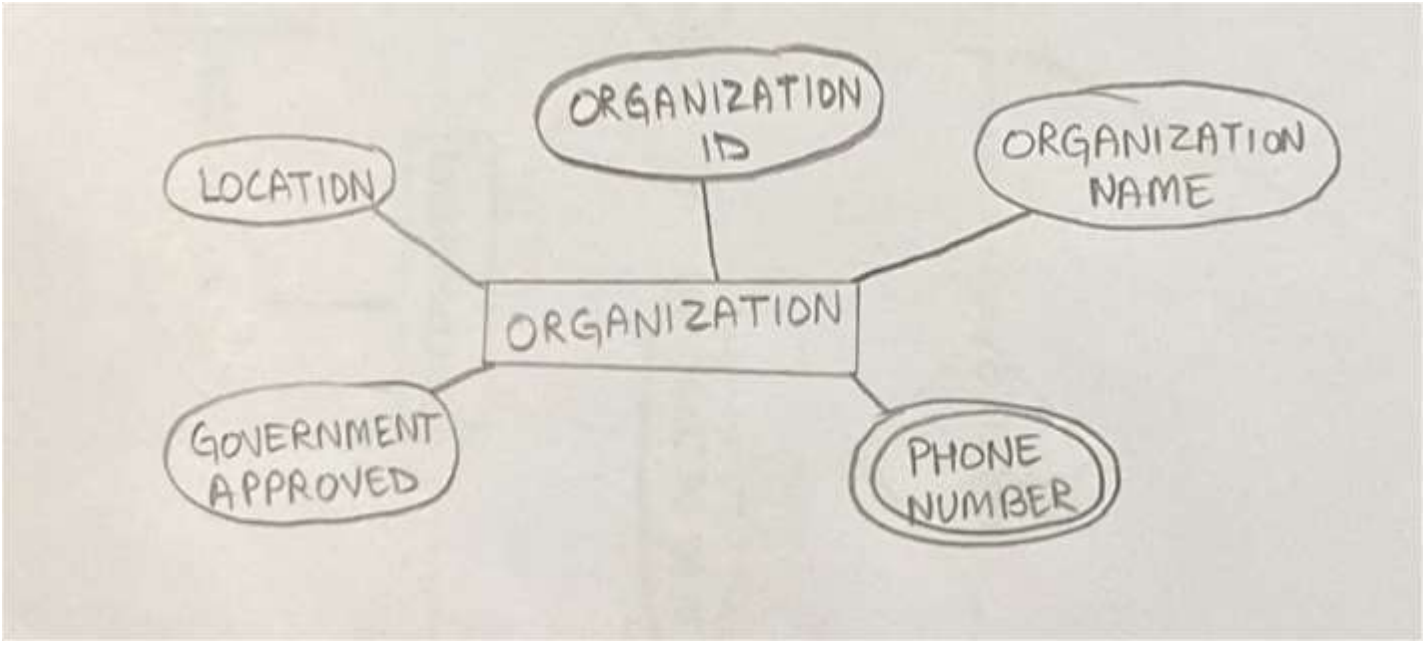
constraint

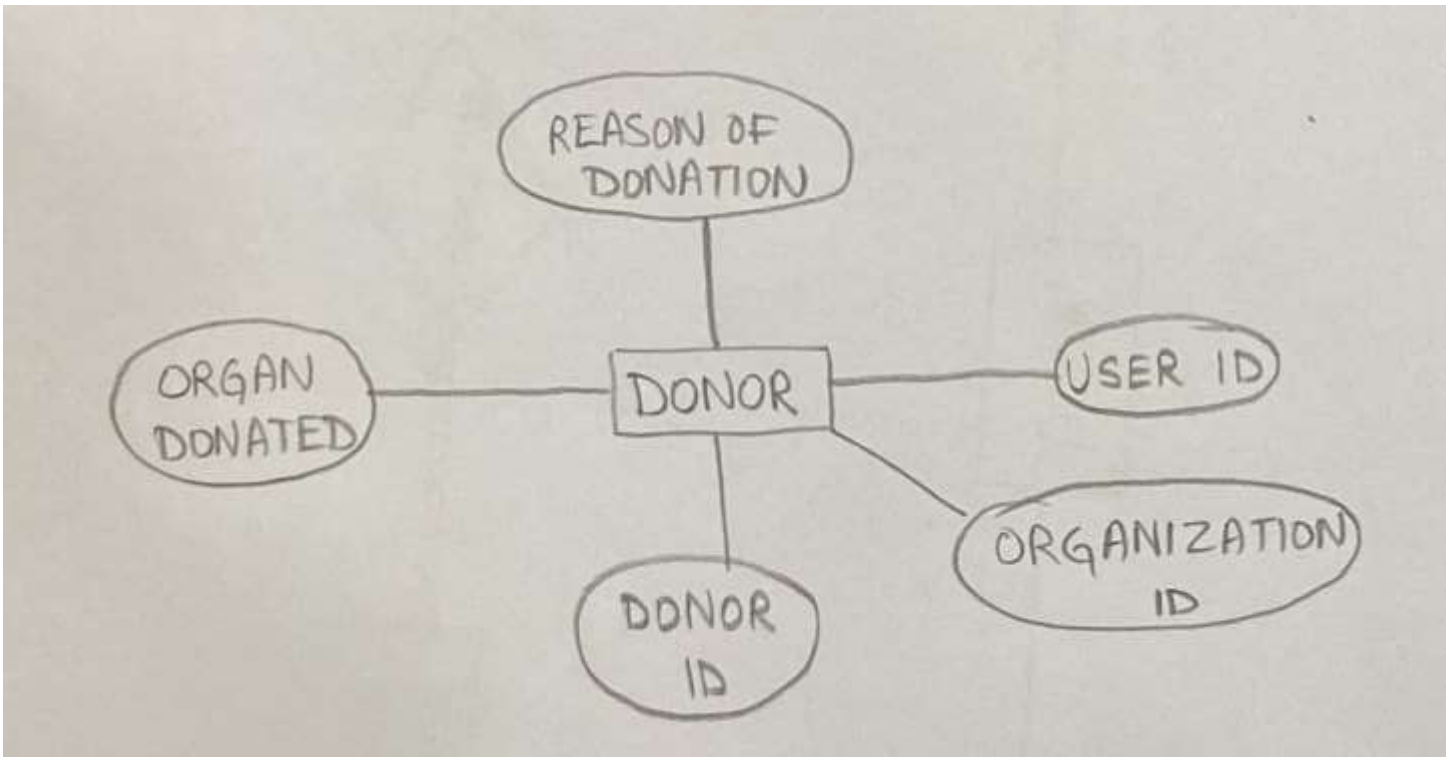
11) Organization_head(Organization_ID, Employee_ID, Name, Date_of_joining, Term_length)

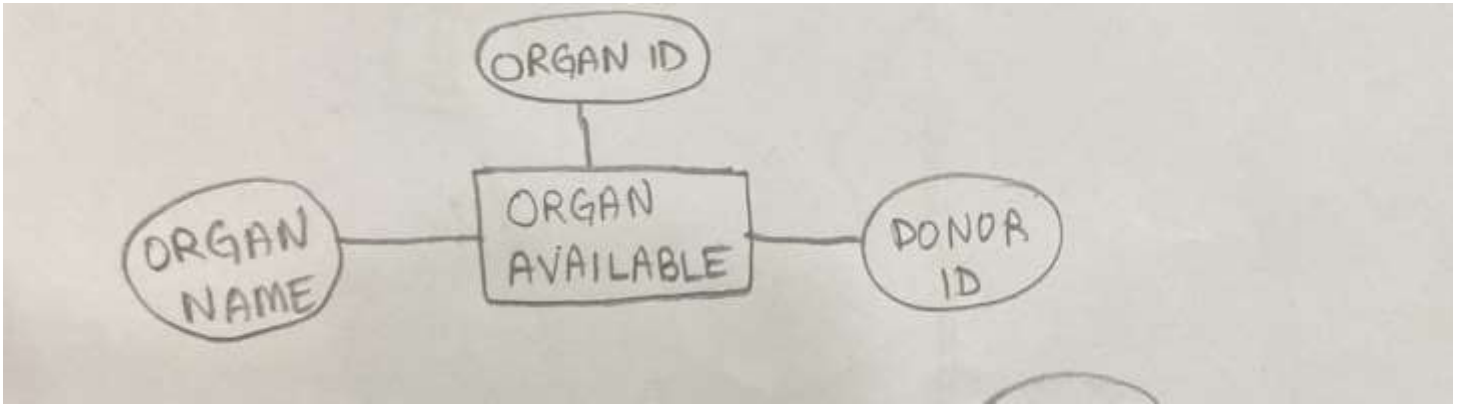
FD={Organization_ID, Employee_ID -> Name, Date_of_joining, Term_length}

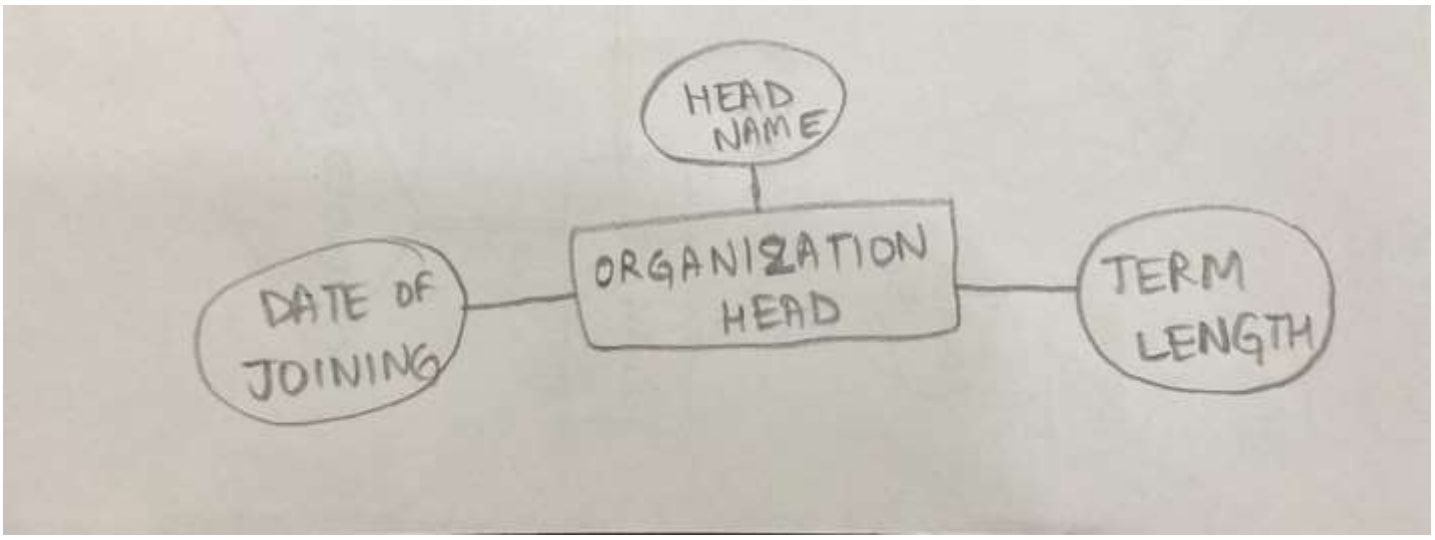
ER MODEL:

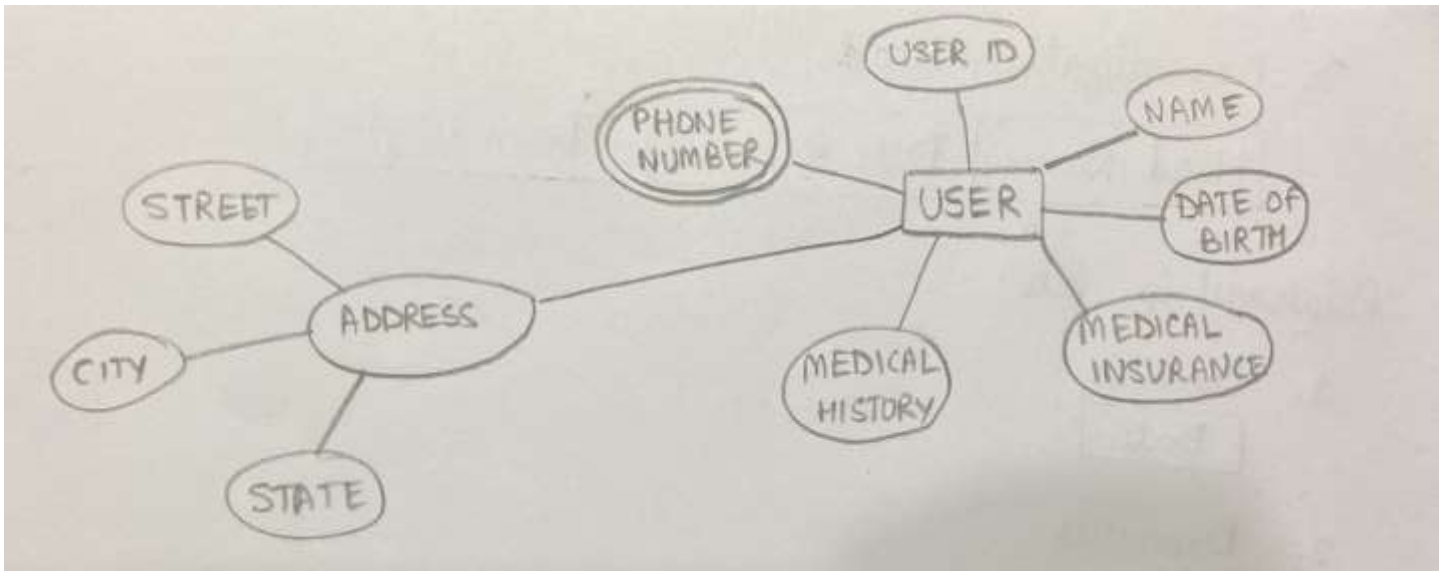

PATIENT ID
STATUS
BILL
DONOR ID
ORGAN ID
DATE OF TRANSACTION
TRANSACTION
PATIENT
ATTENDED BY
PROCURES
ORGAN AVAILABLE
ORGAN DONATED
DOCTOR
USER
WORKS IN
DONATES
DATE OF DONATION
ORGANIZATION
REGISTERS
DONOR
HEADS
ORGANIZATION HEAD


ORGANIZATION ID
LOCATION
ORGANIZATION NAME
ORGANIZATION
GOVERNMENT APPROVED
PHONE NUMBER

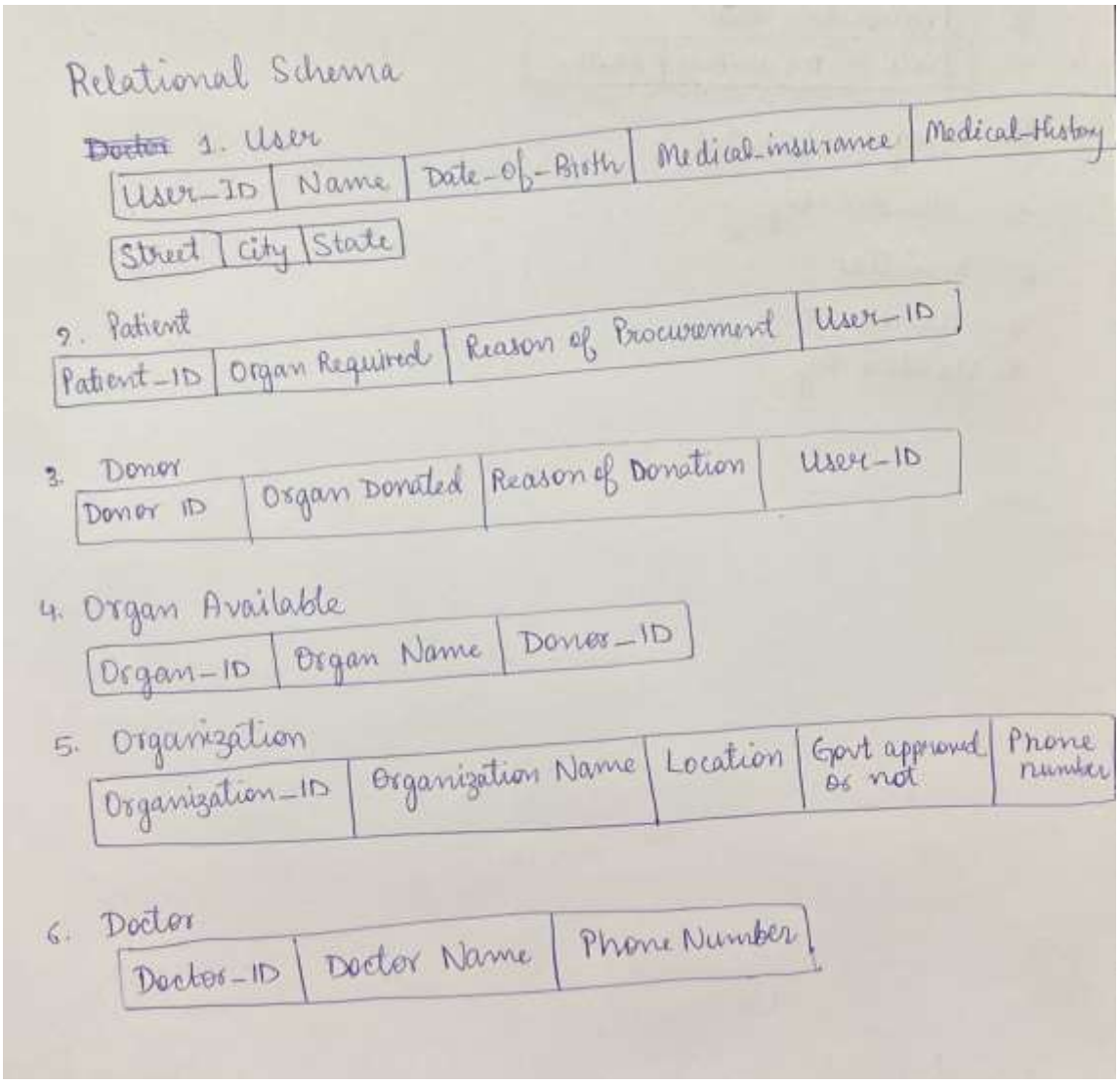

REASON OF DONATION
ORGAN DONATED
DONOR
USER ID
ORGANIZATION ID
DONOR ID

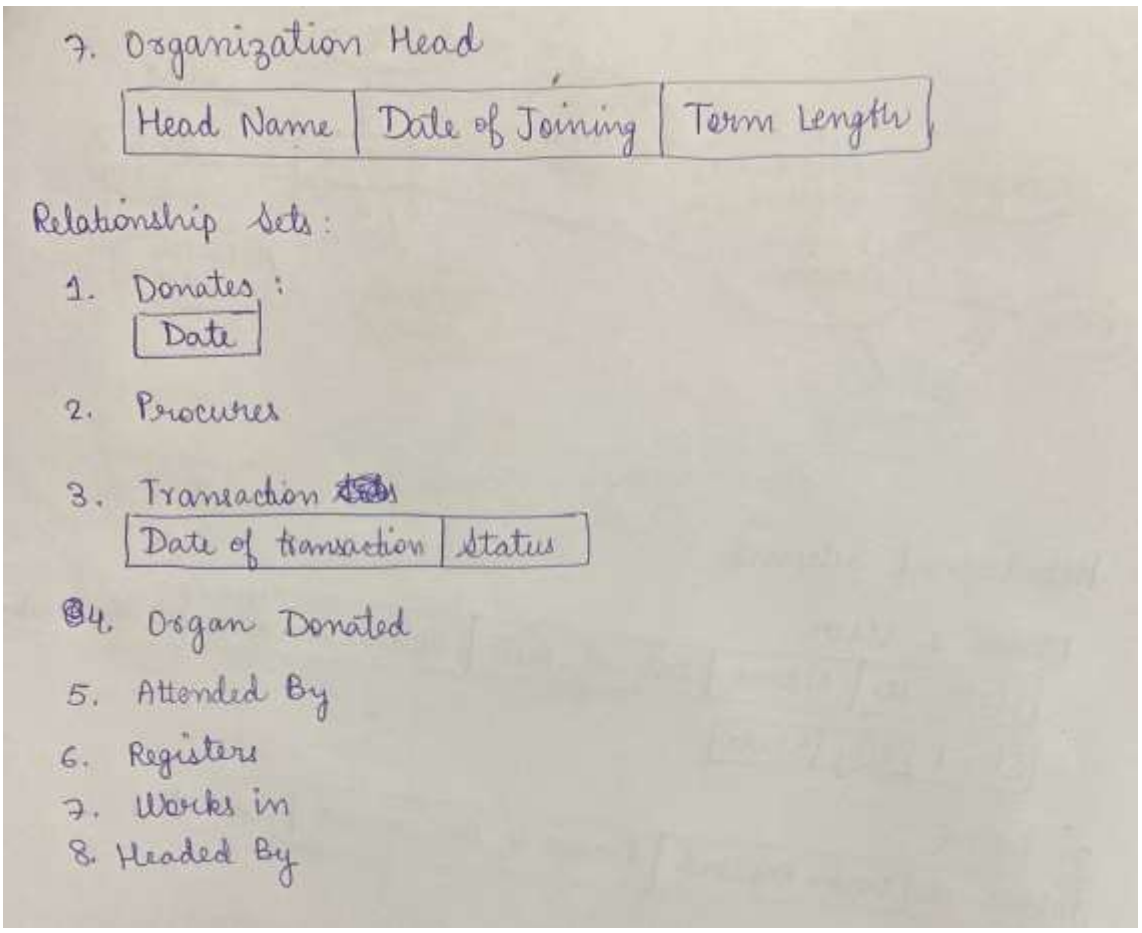

ORGAN ID
ORGAN NAME
ORGAN AVAILABLE
DONOR ID


HEAD NAME
DATE OF JOINING
ORGANIZATION HEAD
TERM LENGTH


USER ID
PHONE NUMBER
NAME
STREET
USER
DATE OF BIRTH
ADDRESS
CITY
MEDICAL HISTORY
MEDICAL INSURANCE
STATE

RELATIONAL SCHEMA:





CREATE TABLE QUERIES IN PYTHON:

```
import mysql.connector
mydb = mysql.connector.connect(
  host='localhost',
  user='root',
  password='0212',
  database = 'DBMS_PROJECT'
)
mycursor=mydb.cursor()

mycursor.execute("CREATE TABLE login( username VARCHAR(20) NOT NULL), password VARCHAR(20)
NOT NULL")
sql="INSERT INTO login (username,password) VALUES (%s,%s)"
val=("admin","admin")
mycursor.execute(sql, val)
mycursor.execute("""CREATE TABLE User(
    User_ID int NOT NULL PRIMARY KEY,
    Name varchar(20) NOT NULL,
    Date_of_Birth date NOT NULL,
    Medical_insurance int,
    Medical_history varchar(20),
    Street varchar(20),
    City varchar(20),
    State varchar(20),
    )""")


mycursor.execute("""CREATE TABLE User_phone_no(
    User_ID INT NOT NULL PRIMARY KEY,
    phone_no VARCHAR(15),
    FOREIGN KEY(User_ID) REFERENCES User(User_ID) ON DELETE CASCADE)""")


mycursor.execute("""CREATE TABLE Organization(
  Organization_ID int NOT NULL,
  Organization_name varchar(20) NOT NULL,
  Location varchar(20),
  Government_approved int, # 0 or 1
  PRIMARY KEY(Organization_ID)
)""")

mycursor.execute("""CREATE TABLE Doctor(
  Doctor_ID int NOT NULL,
  Doctor_Name varchar(20) NOT NULL,
  Department_Name varchar(20) NOT NULL,
  organization_ID int NOT NULL,
  FOREIGN KEY(organization_ID) REFERENCES Organization(organization_ID) ON DELETE CASCADE,
  PRIMARY KEY(Doctor_ID)
)""")

mycursor.execute("""CREATE TABLE Patient(
    Patient_ID int NOT NULL,
    organ_req varchar(20) NOT NULL,
```

```
    reason_of_procurement varchar(20),
    Doctor_ID int NOT NULL,
    User_ID int NOT NULL,
    FOREIGN KEY(User_ID) REFERENCES User(User_ID) ON DELETE CASCADE,
    FOREIGN KEY(Doctor_ID) REFERENCES Doctor(Doctor_ID) ON DELETE CASCADE,
    PRIMARY KEY(Patient_Id, organ_req)
)""")

mycursor.execute(""" CREATE TABLE Donor(
  Donor_ID int NOT NULL,
  organ_donated varchar(20) NOT NULL,
  reason_of_donation varchar(20),
  Organization_ID int NOT NULL,
  User_ID int NOT NULL,
  FOREIGN KEY(User_ID) REFERENCES User(User_ID) ON DELETE CASCADE,
  FOREIGN KEY(Organization_ID) REFERENCES Organization(Organization_ID) ON DELETE CASCADE,
  PRIMARY KEY(Donor_ID, organ_donated)
)""")

mycursor.execute("""CREATE TABLE Organ_available(
  Organ_ID int NOT NULL AUTO_INCREMENT,
  Organ_name varchar(20) NOT NULL,
  Donor_ID int NOT NULL,
  FOREIGN KEY(Donor_ID) REFERENCES Donor(Donor_ID) ON DELETE CASCADE,
  PRIMARY KEY(Organ_ID)
)""")

mycursor.execute("""CREATE TABLE Transaction(
  Patient_ID int NOT NULL,
  Organ_ID int NOT NULL,
  Donor_ID int NOT NULL,
  Date_of_transaction date NOT NULL,
  Status int NOT NULL, #0 or 1
  FOREIGN KEY(Patient_ID) REFERENCES Patient(Patient_ID) ON DELETE CASCADE,
  FOREIGN KEY(Donor_ID) REFERENCES Donor(Donor_ID) ON DELETE CASCADE,
  PRIMARY KEY(Patient_ID,Organ_ID)
)""")

mycursor.execute("""CREATE TABLE Organization_phone_no(
  Organization_ID int NOT NULL,
  Phone_no varchar(15),
  FOREIGN KEY(Organization_ID) REFERENCES Organization(Organization_ID) ON DELETE CASCADE
)""")

mycursor.execute("""CREATE TABLE Doctor_phone_no(
  Doctor_ID int NOT NULL,
  Phone_no varchar(15),
  FOREIGN KEY(Doctor_ID) REFERENCES Doctor(Doctor_ID) ON DELETE CASCADE
)""")

mycursor.execute("""CREATE TABLE Organization_head(
  Organization_ID int NOT NULL,
  Employee_ID int NOT NULL,
  Name varchar(20) NOT NULL,
  Date_of_joining date NOT NULL,
  Term_length int NOT NULL,
  FOREIGN KEY(Organization_ID) REFERENCES Organization(Organization_ID) ON DELETE CASCADE,
  PRIMARY KEY(Organization_ID,Employee_ID)
```

```
)""")

mycursor.execute("DROP TRIGGER IF EXISTS ADD_DONOR")
qrystr="""CREATE TRIGGER ADD_DONOR
AFTER INSERT ON DONOR
FOR EACH ROW BEGIN INSERT INTO Organ_available(Organ_name, Donor_ID)
VALUES (new.organ_donated, new.Donor_ID); end;"""

mycursor.execute(qrystr)



qry2="""CREATE TRIGGER ADD_DONOR_LOG AFTER INSERT
 on Donor for each row begin insert into log values
  (now(), concat("Inserted new Donor", cast(new.Donor_Id as char))); end;"""


mycursor.execute("""create trigger UPD_DONOR_LOG
after update
on Donor
for each row
begin
insert into log values
(now(), concat("Updated Donor Details", cast(new.Donor_Id as char)));
end;""")

mycursor.execute("""create trigger DEL_DONOR_LOG
after delete
on Donor
for each row
begin
insert into log values
(now(), concat("Deleted Donor ", cast(old.Donor_Id as char)));
end; """)

mycursor.execute(""" create trigger ADD_PATIENT_LOG
after insert
on Patient
for each row
begin
insert into log values
(now(), concat("Inserted new Patient ", cast(new.Patient_Id as char)));
end; """)

mycursor.execute("""create trigger UPD_PATIENT_LOG
after update
on Patient
for each row
begin
insert into log values
(now(), concat("Updated Patient Details ", cast(new.Patient_Id as char)));
end; """)


mycursor.execute("""create trigger DEL_PATIENT_LOG
after delete
on Donor
for each row
```

```
begin
insert into log values
(now(), concat("Deleted Patient ", cast(old.Donor_Id as char)));
end """)

mycursor.execute("""create trigger ADD_TRANSACTION_LOG
after insert
on Transaction
for each row
begin
insert into log values
(now(), concat("Added Transaction :: Patient ID : "
, cast(new.Patient_ID as char),
"; Donor ID : " ,cast(new.Donor_ID as char)));
end """)

mycursor.execute("""INSERT INTO User
 VALUES(10,'Random1','2000-01-01',1,NULL,'Street 1','City 1','State 1')""")
mycursor.execute("""INSERT INTO User
 VALUES(20,'Random2','2000-01-02',1,NULL,'Street 2','City 2','State 2')""")
```

OUTPUT:

```
Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import mysql.connector
>>> mydb= mysql.connector.connect(
...   host='localhost',
...   user='root',
...   password='0212',
...   database = 'DBMS_PROJECT'
... )
>>> mycursor=mydb.cursor()
>>> mycursor.execute("Show tables;")
>>>
>>> myresult = mycursor.fetchall()
>>>
>>> for x in myresult:
...     print(x)
...
('doctor',)
('donor',)
('log',)
('login',)
('organ_available',)
('organization',)
('organization_head',)
('organization_phone_no',)
('patient',)
('transaction',)
('user',)
('user_phone_no',)
>>>
```

INSERT VALUE QUERIES IN PYTHON:

```
import mysql.connector
mydb = mysql.connector.connect(
  host='localhost',
  user='root',
  password='0212',
```

```
  database = 'DBMS_PROJECT'
)
mycursor=mydb.cursor()
sql = """INSERT INTO doctors
 (Doctor_ID,Doctor_Name, Department_Name, organization_ID)
 VALUES (%s, %s,%s,%s)"""
val=[(1,'Doctor-1','Department-1',78),
(2,'Doctor-2','Department-2',10),
(3,'Doctor-3','Department-3',61),
(4,'Doctor-4','Department-4',26),
(5,'Doctor-5','Department-5',11),
(6,'Doctor-6','Department-6',99),
(7,'Doctor-7','Department-7',54),
(8,'Doctor-8','Department-8',44),
(9,'Doctor-9','Department-9',57),
(10,'Doctor-10','Department-10',31),
(11,'Doctor-11','Department-11',5),
(12,'Doctor-12','Department-12',11),
(13,'Doctor-13','Department-13',36),
(14,'Doctor-14','Department-14',22),
(15,'Doctor-15','Department-15',42),
(16,'Doctor-16','Department-16',73),
(17,'Doctor-17','Department-17',80),
(18,'Doctor-18','Department-18',77),
(19,'Doctor-19','Department-19',6),
(20,'Doctor-20','Department-20',87)
]
mycursor.executemany(sql, val)

mydb.commit()

insertdonor="""INSERT INTO donors
(Donor_ID,organ_donated,reason_of_donation,Organzation_ID,User_ID)
VALUES (%s,%s,%s,%s,%s)"""
val1=[
(1,'Heart','Reason-1',97,90),
(2,'Pancreas','Reason-2',79,41),
(3,'Pancreas','Reason-3',1,95),
(4,'Intestine','Reason-4',60,96),
(5,'Kidney','Reason-5',69,72),
(6,'Pancreas','Reason-6',1,89),
(7,'Kidney','Reason-7',51,43),
(8,'Kidney','Reason-8',53,61),
(9,'Heart','Reason-9',57,16),
(10,'Heart','Reason-10',24,50),
(11,'Kidney','Reason-11',8,92),
(12,'Pancreas','Reason-12',64,58),
(13,'Pancreas','Reason-13',28,45),
(14,'Pancreas','Reason-14',10,75),
(15,'Heart','Reason-15',50,53),
(16,'Intestine','Reason-16',27,31),
(17,'Intestine','Reason-17',72,94),
(18,'Intestine','Reason-18',97,7),
(19,'Pancreas','Reason-19',69,67),
(20,'Intestine','Reason-20',40,28),
]
mycursor.executemany(insertdonor, val1)
mydb.commit()
```

```
sqlorg="""INSERT INTO Organization
(Organization_ID,Organization_name,Location,Government_approved)
VALUES(%s,%s,%s,%s)"""
val2=[
(1, 'Organization-1','New Delhi',1),
(2, 'Organization-2','Mumbai',0),
(3, 'Organization-3','Kolkata',0),
(4, 'Organization-4','Kolkata',1),
(5, 'Organization-5','Ahmedabad',1),
(6, 'Organization-6','Kolkata',0),
(7, 'Organization-7','Kolkata',0),
(8, 'Organization-8','Ahmedabad',0),
(9, 'Organization-9','Kolkata',1),
(10, 'Organization-10','Ahmedabad',1),
(11, 'Organization-11','Ahmedabad',1),
(12, 'Organization-12','Mumbai',0),
(13, 'Organization-13','Kolkata',0),
(14, 'Organization-14','Ahmedabad',1),
(15, 'Organization-15','Ahmedabad',0),
(16, 'Organization-16','Kolkata',0),
(17, 'Organization-17','Kolkata',1),
(18, 'Organization-18','Mumbai',1),
(19, 'Organization-19','Ahmedabad',1),
(20, 'Organization-20','Ahmedabad',1)
]
mycursor.executemany(sqlorg, val2)
mydb.commit()

sqlpatient="""INSERT INTO Patient
(Patient_ID,organ_req,reason_of_procurement,Doctor_ID,User_ID)
VALUES (%s,%s,%s,%s,%s)"""
val3=[
(1,'Heart','Reason-1',63,48),
(2,'Kidney','Reason-2',62,11),
(3,'Pancreas','Reason-3',72,84),
(4,'Kidney','Reason-4',87,36),
(5,'Heart','Reason-5',44,13),
(6,'Lung','Reason-6',71,52),
(7,'Intestine','Reason-7',63,85),
(8,'Intestine','Reason-8',42,83),
(9,'Lung','Reason-9',41,52),
(10,'Kidney','Reason-10',16,8),
(11,'Kidney','Reason-11',91,95),
(12,'Pancreas','Reason-12',70,58),
(13,'Intestine','Reason-13',81,44),
(14,'Heart','Reason-14',3,94),
(15,'Kidney','Reason-15',94,30),
(16,'Lung','Reason-16',95,97),
(17,'Heart','Reason-17',7,2),
(18,'Kidney','Reason-18',89,82),
(19,'Kidney','Reason-19',25,24),
(20,'Pancreas','Reason-20',11,23)
]
mycursor.executemany(sqlpatient, val3)
mydb.commit()

sqluser="""INSERT INTO User
```

```
(User_ID,Name,Date_Of_Birth, Medical_insurance, Medical_history,Street, City, State)
VALUES (%s,%s,%s,%s,%s,%s,%s,%s)"""
val4=[
( 1 ,'Name-1','1978-8-21',1,'NIL','Street-1','New Delhi','Delhi'),
( 2 ,'Name-2','1975-12-10',0,'NIL','Street-2','Mumbai','Maharashtra'),
( 3 ,'Name-3','1976-6-4',0,'NIL','Street-3','Mumbai','Maharashtra'),
( 4 ,'Name-4','1985-10-13',1,'NIL','Street-4','Ahmedabad','Gujarat'),
( 5 ,'Name-5','1983-10-12',1,'NIL','Street-5','Kolkata','West Bengal'),
( 6 ,'Name-6','1977-1-18',1,'NIL','Street-6','Kolkata','West Bengal'),
( 7 ,'Name-7','1976-2-26',0,'NIL','Street-7','New Delhi','Delhi'),
( 8 ,'Name-8','1973-4-12',1,'NIL','Street-8','Mumbai','Maharashtra'),
( 9 ,'Name-9','1976-11-1',0,'NIL','Street-9','Mumbai','Maharashtra'),
( 10 ,'Name-10','1978-11-18',1,'NIL','Street-10','New Delhi','Delhi'),
( 11 ,'Name-11','1975-1-6',1,'NIL','Street-11','Mumbai','Maharashtra'),
( 12 ,'Name-12','1983-11-1',1,'NIL','Street-12','Mumbai','Maharashtra'),
( 13 ,'Name-13','1983-1-9',1,'NIL','Street-13','New Delhi','Delhi'),
( 14 ,'Name-14','1975-10-12',1,'NIL','Street-14','Mumbai','Maharashtra'),
( 15 ,'Name-15','1977-9-23',0,'NIL','Street-15','Ahmedabad','Gujarat'),
( 16 ,'Name-16','1982-11-29',1,'NIL','Street-16','New Delhi','Delhi'),
( 17 ,'Name-17','1974-3-19',0,'NIL','Street-17','Mumbai','Maharashtra'),
( 18 ,'Name-18','1973-10-27',0,'NIL','Street-18','New Delhi','Delhi'),
( 19 ,'Name-19','1980-3-18',0,'NIL','Street-19','Kolkata','West Bengal'),
( 20 ,'Name-20','1978-8-15',1,'NIL','Street-20','Kolkata','West Bengal')
]
mycursor.executemany(sqluser, val4)
mydb.commit()

sqltrans="""INSERT INTO transaction
( Patient_ID, Organ_ID, Donor_ID, Date_of_transaction, Status)
 VALUES (%s,%s,%s,%s,%s)"""
val5=[
( 22,7,7,'2014-9-19',0),
( 97,19,19,'2013-4-30',1),
( 156,154,154,'2017-4-10',1),
( 113,110,110,'2013-9-28',1),
( 73,36,36,'2017-3-27',0),
( 108,28,28,'2015-8-1',0),
( 111,164,164,'2012-4-2',1),
( 86,184,184,'2013-11-11',0),
( 34,106,106,'2014-3-12',1),
( 51,149,149,'2017-1-29',0),
( 69,97,97,'2015-12-21',1),
( 174,77,77,'2013-8-4',0),
( 68,17,17,'2012-5-15',1),
( 32,119,119,'2017-5-22',0),
( 79,76,76,'2015-12-23',0),
( 183,16,16,'2014-3-5',0),
( 60,186,186,'2014-9-2',1),
( 142,44,44,'2016-6-8',1),
( 199,10,10,'2016-9-28',1),
( 109,181,181,'2014-5-22',0)
]
mycursor.executemany(sqltrans, val5)
mydb.commit()

sqlphone="""INSERT INTO User_Phone_no(User_ID, phone_number) VALUES (%s,%s,)"""
val6=[
(1,'Kidney','Reason-1',85,6),
```

```
(2,'Pancreas','Reason-2',13,10),
(3,'Heart','Reason-3',44,39),
(4,'Lung','Reason-4',88,49),
(5,'Lung','Reason-5',80,60),
(6,'Intestine','Reason-6',49,66),
(7,'Heart','Reason-7',8,90),
(8,'Heart','Reason-8',44,27),
(9,'Pancreas','Reason-9',39,84),
(10,'Lung','Reason-10',51,66),
(11,'Pancreas','Reason-11',57,40),
(12,'Kidney','Reason-12',11,68),
(13,'Kidney','Reason-13',47,32),
(14,'Intestine','Reason-14',51,43),
(15,'Heart','Reason-15',38,85),
(16,'Lung','Reason-16',32,70),
(17,'Lung','Reason-17',65,69),
(18,'Kidney','Reason-18',21,85),
(19,'Lung','Reason-19',28,58),
(20,'Pancreas','Reason-20',23,68)
]
mycursor.executemany(sqlphone, val6)
mydb.commit()
```

OUTPUT:

```
>>> mycursor.execute("SELECT*FROM User")
>>> result=mycursor.fetchall()
>>> for row in result:
...     print(row)
...     print("\n")
...
(1, 'Name-1', datetime.date(1978, 8, 21), 1, 'NIL', 'Street-1', 'New Delhi', 'Delhi')


(2, 'Name-2', datetime.date(1975, 12, 10), 0, 'NIL', 'Street-2', 'Mumbai', 'Maharashtra')


(3, 'Name-3', datetime.date(1976, 6, 4), 0, 'NIL', 'Street-3', 'Mumbai', 'Maharashtra')


(4, 'Name-4', datetime.date(1985, 10, 13), 1, 'NIL', 'Street-4', 'Ahmedabad', 'Gujarat')


(5, 'Name-5', datetime.date(1983, 10, 12), 1, 'NIL', 'Street-5', 'Kolkata', 'West Bengal')


(6, 'Name-6', datetime.date(1977, 1, 18), 1, 'NIL', 'Street-6', 'Kolkata', 'West Bengal')


(7, 'Name-7', datetime.date(1976, 2, 26), 0, 'NIL', 'Street-7', 'New Delhi', 'Delhi')


(8, 'Name-8', datetime.date(1973, 4, 12), 1, 'NIL', 'Street-8', 'Mumbai', 'Maharashtra')


(9, 'Name-9', datetime.date(1976, 11, 1), 0, 'NIL', 'Street-9', 'Mumbai', 'Maharashtra')


(10, 'Name-10', datetime.date(1978, 11, 18), 1, 'NIL', 'Street-10', 'New Delhi', 'Delhi')


(11, 'Name-11', datetime.date(1975, 1, 6), 1, 'NIL', 'Street-11', 'Mumbai', 'Maharashtra')


(12, 'Name-12', datetime.date(1983, 11, 1), 1, 'NIL', 'Street-12', 'Mumbai', 'Maharashtra')
```

```
(13, 'Name-13', datetime.date(1983, 1, 9), 1, 'NIL', 'Street-13', 'New Delhi', 'Delhi')

(14, 'Name-14', datetime.date(1975, 10, 12), 1, 'NIL', 'Street-14', 'Mumbai', 'Maharashtra')

(15, 'Name-15', datetime.date(1977, 9, 23), 0, 'NIL', 'Street-15', 'Ahmedabad', 'Gujarat')

(16, 'Name-16', datetime.date(1982, 11, 29), 1, 'NIL', 'Street-16', 'New Delhi', 'Delhi')

(17, 'Name-17', datetime.date(1974, 3, 19), 0, 'NIL', 'Street-17', 'Mumbai', 'Maharashtra')

(18, 'Name-18', datetime.date(1973, 10, 27), 0, 'NIL', 'Street-18', 'New Delhi', 'Delhi')

(19, 'Name-19', datetime.date(1980, 3, 18), 0, 'NIL', 'Street-19', 'Kolkata', 'West Bengal')

(20, 'Name-20', datetime.date(1978, 8, 15), 1, 'NIL', 'Street-20', 'Kolkata', 'West Bengal')
```

```
>>> mycursor.execute("SELECT*FROM DOCTOR")
>>> result=mycursor.fetchall()
>>> for row in result:
...     print(row)
...     print("\n")
...
(2, 'Doctor-2', 'Department-2', 10)


(3, 'Doctor-3', 'Department-3', 61)


(4, 'Doctor-4', 'Department-4', 26)


(5, 'Doctor-5', 'Department-5', 11)


(6, 'Doctor-6', 'Department-6', 99)


(7, 'Doctor-7', 'Department-7', 54)


(8, 'Doctor-8', 'Department-8', 44)


(9, 'Doctor-9', 'Department-9', 57)


(10, 'Doctor-10', 'Department-10', 31)


(11, 'Doctor-11', 'Department-11', 5)


(12, 'Doctor-12', 'Department-12', 11)
```

```
(13, 'Doctor-13', 'Department-13', 36)

(14, 'Doctor-14', 'Department-14', 22)

(15, 'Doctor-15', 'Department-15', 42)

(16, 'Doctor-16', 'Department-16', 73)

(17, 'Doctor-17', 'Department-17', 80)

(18, 'Doctor-18', 'Department-18', 77)

(19, 'Doctor-19', 'Department-19', 6)

(20, 'Doctor-20', 'Department-20', 87)
```

```
>>> mycursor.execute("SELECT*FROM Patient")
>>> result=mycursor.fetchall()
>>> for row in result:
...     print(row)
...     print("\n")
...
(1, 'Heart', 'Reason-1', 63, 48)


(2, 'Kidney', 'Reason-2', 62, 11)


(3, 'Pancreas', 'Reason-3', 72, 84)


(4, 'Kidney', 'Reason-4', 87, 36)


(5, 'Heart', 'Reason-5', 44, 13)


(6, 'Lung', 'Reason-6', 71, 52)


(7, 'Intestine', 'Reason-7', 63, 85)


(8, 'Intestine', 'Reason-8', 42, 83)


(9, 'Lung', 'Reason-9', 41, 52)


(10, 'Kidney', 'Reason-10', 16, 8)


(11, 'Kidney', 'Reason-11', 91, 95)


(12, 'Pancreas', 'Reason-12', 70, 58)
```

```
(13, 'Intestine', 'Reason-13', 81, 44)

(14, 'Heart', 'Reason-14', 3, 94)

(15, 'Kidney', 'Reason-15', 94, 30)

(16, 'Lung', 'Reason-16', 95, 97)

(17, 'Heart', 'Reason-17', 7, 2)

(18, 'Kidney', 'Reason-18', 89, 82)

(19, 'Kidney', 'Reason-19', 25, 24)

(20, 'Pancreas', 'Reason-20', 11, 23)
```

```
>>> mycursor.execute("SELECT*FROM Donor")
>>> result=mycursor.fetchall()
>>> for row in result:
...     print(row)
...     print("\n")
...
(1, 'Heart', 'Reason-1', 97, 90)

(2, 'Pancreas', 'Reason-2', 79, 41)

(3, 'Pancreas', 'Reason-3', 1, 95)

(4, 'Intestine', 'Reason-4', 60, 96)

(5, 'Kidney', 'Reason-5', 69, 72)

(6, 'Pancreas', 'Reason-6', 1, 89)

(7, 'Kidney', 'Reason-7', 51, 43)

(8, 'Kidney', 'Reason-8', 53, 61)

(9, 'Heart', 'Reason-9', 57, 16)

(10, 'Heart', 'Reason-10', 24, 50)

(11, 'Kidney', 'Reason-11', 8, 92)

(12, 'Pancreas', 'Reason-12', 64, 58)
```

```
(13, 'Pancreas', 'Reason-13', 28, 45)

(14, 'Pancreas', 'Reason-14', 10, 75)

(15, 'Heart', 'Reason-15', 50, 53)

(16, 'Intestine', 'Reason-16', 27, 31)

(17, 'Intestine', 'Reason-17', 72, 94)

(18, 'Intestine', 'Reason-18', 97, 7)

(19, 'Pancreas', 'Reason-19', 69, 67)

(20, 'Intestine', 'Reason-20', 40, 28)
```

```
>>> mycursor.execute("SELECT*FROM Organization")
>>> result=mycursor.fetchall()
>>> for row in result:
...     print(row)
...     print("\n")
...
(1, 'Organization-1', 'New Delhi', 1)

(2, 'Organization-2', 'Mumbai', 0)

(3, 'Organization-3', 'Kolkata', 0)

(4, 'Organization-4', 'Kolkata', 1)

(5, 'Organization-5', 'Ahmedabad', 1)

(6, 'Organization-6', 'Kolkata', 0)

(7, 'Organization-7', 'Kolkata', 0)

(8, 'Organization-8', 'Ahmedabad', 0)

(9, 'Organization-9', 'Kolkata', 1)

(10, 'Organization-10', 'Ahmedabad', 1)

(11, 'Organization-11', 'Ahmedabad', 1)

(12, 'Organization-12', 'Mumbai', 0)
```

```
(13, 'Organization-13', 'Kolkata', 0)

(14, 'Organization-14', 'Ahmedabad', 1)

(15, 'Organization-15', 'Ahmedabad', 0)

(16, 'Organization-16', 'Kolkata', 0)

(17, 'Organization-17', 'Kolkata', 1)

(18, 'Organization-18', 'Mumbai', 1)

(19, 'Organization-19', 'Ahmedabad', 1)

(20, 'Organization-20', 'Ahmedabad', 1)
```

```
>>> mycursor.execute("SELECT*FROM Transaction")
>>> result=mycursor.fetchall()
>>> for row in result:
...     print(row)
...     print("\n")
...
(1, 93, 93, datetime.date(2012, 4, 12), 0)


(4, 155, 155, datetime.date(2015, 5, 29), 0)


(6, 58, 58, datetime.date(2016, 10, 26), 0)


(7, 172, 172, datetime.date(2012, 10, 10), 0)


(8, 71, 71, datetime.date(2016, 5, 2), 0)


(10, 147, 147, datetime.date(2016, 3, 8), 0)


(12, 145, 145, datetime.date(2016, 11, 26), 0)


(19, 4, 4, datetime.date(2014, 11, 10), 0)


(20, 129, 129, datetime.date(2014, 10, 10), 0)


(21, 144, 144, datetime.date(2012, 12, 11), 0)


(22, 7, 7, datetime.date(2014, 9, 19), 0)


(26, 23, 23, datetime.date(2013, 8, 24), 1)
```

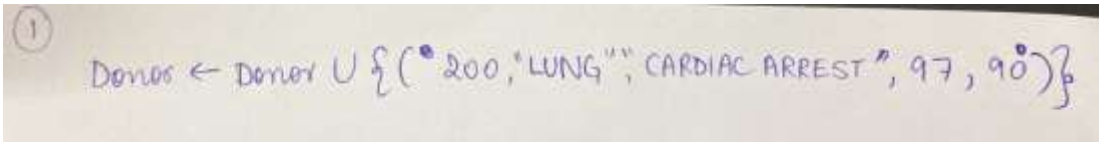
```
(28, 143, 143, datetime.date(2017, 12, 13), 0)

(30, 82, 82, datetime.date(2014, 7, 25), 1)

(31, 152, 152, datetime.date(2014, 9, 4), 1)

(32, 119, 119, datetime.date(2017, 5, 22), 0)

(33, 162, 162, datetime.date(2016, 12, 16), 0)

(34, 106, 106, datetime.date(2014, 3, 12), 1)

(36, 123, 123, datetime.date(2016, 12, 3), 0)

(37, 91, 91, datetime.date(2016, 7, 13), 1)
```

QUERIES IMPLEMENTING OUR REQUIREMENTS IN PYTHON:

1.ADD NEW ENTRY TO DONOR TABLE:

```
>>> insertdonor="""INSERT INTO donor (Donor_ID,organ_donated,reason_of_donation,Organization_ID,User_ID) VALUES (%s,%s,%s,%s,%s)"""
>>> insert=[200,'LUNG','CARDIAC ARREST',97,90]
>>> mycursor.execute(insertdonor,insert)
>>> mydb.commit()
>>>
>>> mycursor.execute("SELECT*FROM Donor")
>>> r=mycursor.fetchall()
>>> for x in r:
...     print(x)
...     print("\n")
```

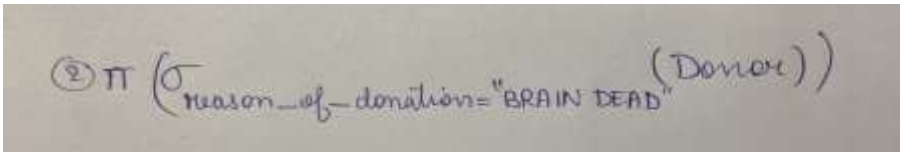
```
(200, 'LUNG', 'CARDIAC ARREST', 97, 90)
```



2. FIND DONORS WHO WERE BRAIN DEAD:

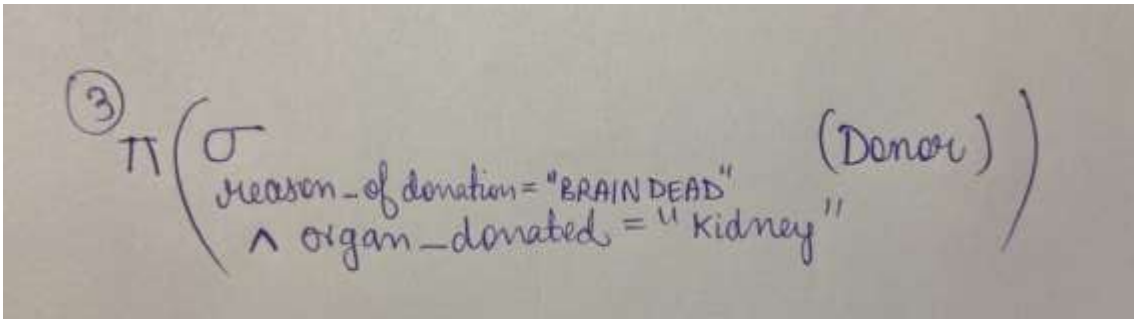
```
>>> mycursor.execute("SELECT*FROM Donor WHERE reason_of_donation='BRAIN DEAD'")
>>> myresult=mycursor.fetchall()
>>> for x in myresult:
...     print(x)
...     print("\n")
...
(2, 'Pancreas', 'BRAIN DEAD', 79, 41)


(6, 'Pancreas', 'BRAIN DEAD', 1, 89)


(9, 'Heart', 'BRAIN DEAD', 57, 16)


(13, 'Pancreas', 'BRAIN DEAD', 28, 45)


(16, 'Intestine', 'BRAIN DEAD', 27, 31)


(18, 'Intestine', 'BRAIN DEAD', 97, 7)


(24, 'Lung', 'BRAIN DEAD', 89, 50)


(27, 'Lung', 'BRAIN DEAD', 33, 92)


(29, 'Intestine', 'BRAIN DEAD', 62, 58)


(32, 'Lung', 'BRAIN DEAD', 11, 72)


(34, 'Heart', 'BRAIN DEAD', 22, 29)
```

```
(42, 'Kidney', 'BRAIN DEAD', 85, 36)

(45, 'Heart', 'BRAIN DEAD', 5, 33)

(48, 'Lung', 'BRAIN DEAD', 54, 42)

(51, 'Intestine', 'BRAIN DEAD', 55, 44)

(53, 'Kidney', 'BRAIN DEAD', 9, 17)

(55, 'Lung', 'BRAIN DEAD', 25, 35)

(59, 'Lung', 'BRAIN DEAD', 6, 57)

(62, 'Intestine', 'BRAIN DEAD', 9, 94)

(65, 'Pancreas', 'BRAIN DEAD', 79, 32)

(72, 'Lung', 'BRAIN DEAD', 55, 36)

(75, 'Intestine', 'BRAIN DEAD', 5, 27)

(77, 'Pancreas', 'BRAIN DEAD', 46, 28)

(80, 'Kidney', 'BRAIN DEAD', 64, 98)
```



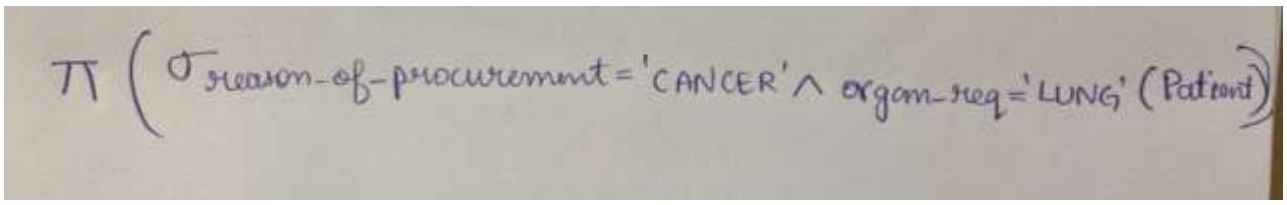
3. FIND THE DONORS WHO DONATED THEIR KIDNEY WHERE THE REASON OF DONATION IS BRAIN DEAD

```
>>> mycursor.execute("SELECT*FROM Donor WHERE reason_of_donation='BRAIN DEAD' AND organ_donated='Kidney'")
>>> myresult=mycursor.fetchall()
>>> for x in myresult:
...     print(x)
...     print("\n")
...
(42, 'Kidney', 'BRAIN DEAD', 85, 36)


(53, 'Kidney', 'BRAIN DEAD', 9, 17)


(80, 'Kidney', 'BRAIN DEAD', 64, 98)


(87, 'Kidney', 'BRAIN DEAD', 38, 96)
```

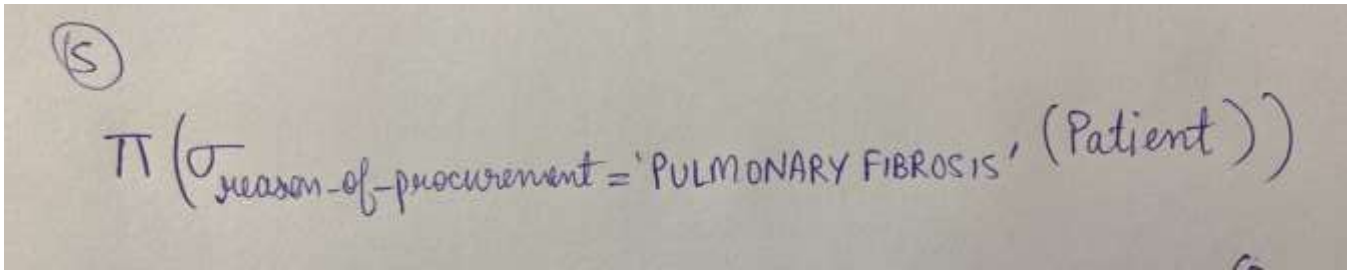


4. FIND THE PATIENTS WHO NEED A LUNG TRANSPLANT WHO ALSO SUFFER FROM CANCER:

```
>>> mycursor.execute("SELECT*FROM Patient WHERE reason_of_procurement='CANCER' AND organ_req='LUNG'")
>>> r=mycursor.fetchall()
>>> for x in r:
...     print(x)
...     print("\n")
...
(26, 'Lung', 'CANCER', 94, 8)


(50, 'Lung', 'CANCER', 95, 4)


(91, 'Lung', 'CANCER', 17, 56)


(133, 'Lung', 'CANCER', 48, 58)


(136, 'Lung', 'CANCER', 27, 19)


(168, 'Lung', 'CANCER', 71, 21)


(180, 'Lung', 'CANCER', 15, 80)
```



5.LIST THE ROWS FROM THE PATIENT TABLE THAT HAS PATIENTS SUFFERING FROM PULMONARY FIBROSIS



```
>>> mycursor.execute("SELECT*FROM Patient WHERE reason_of_procurement='PULMONARY FIBROSIS'")
>>> r=mycursor.fetchall()
>>> for x in r:
...     print(x)
...     print("\n")
...
(105, 'Lung', 'PULMONARY FIBROSIS', 44, 8)


(111, 'Lung', 'PULMONARY FIBROSIS', 80, 42)


(119, 'Lung', 'PULMONARY FIBROSIS', 59, 82)


(156, 'Lung', 'PULMONARY FIBROSIS', 62, 84)


(162, 'Lung', 'PULMONARY FIBROSIS', 62, 58)


(170, 'Lung', 'PULMONARY FIBROSIS', 20, 26)


(173, 'Lung', 'PULMONARY FIBROSIS', 97, 23)


(192, 'Lung', 'PULMONARY FIBROSIS', 49, 26)


(198, 'Lung', 'PULMONARY FIBROSIS', 82, 71)


(199, 'Lung', 'PULMONARY FIBROSIS', 86, 31)
```

6. LIST ALL THE DONORS WHO DONATED THEIR PANCREAS AFTER SUFFERING CARDIAC ARREST

```
>>> mycursor.execute("SELECT*FROM Donor WHERE reason_of_donation='CARDIAC ARREST' AND organ_donated='Pancreas'")
>>> r=mycursor.fetchall()
>>> for x in r:
...     print(x)
...     print("\n")
...
(12, 'Pancreas', 'CARDIAC ARREST', 64, 58)


(14, 'Pancreas', 'CARDIAC ARREST', 10, 75)


(19, 'Pancreas', 'CARDIAC ARREST', 69, 67)


(21, 'Pancreas', 'CARDIAC ARREST', 97, 40)


(28, 'Pancreas', 'CARDIAC ARREST', 67, 93)


(38, 'Pancreas', 'CARDIAC ARREST', 12, 28)


(71, 'Pancreas', 'CARDIAC ARREST', 46, 89)


(79, 'Pancreas', 'CARDIAC ARREST', 65, 71)


(112, 'Pancreas', 'CARDIAC ARREST', 38, 14)


(131, 'Pancreas', 'CARDIAC ARREST', 8, 91)


(139, 'Pancreas', 'CARDIAC ARREST', 11, 67)


(167, 'Pancreas', 'CARDIAC ARREST', 58, 1)
```

(6) $\Pi\left(\sigma_{\text{reason\_of\_donation}='\text{CARDIAC ARREST}' \wedge \text{organ donated}='\text{PANCREAS}'}(\text{Donor})\right)$