DBMS Lab End-Semester

106119029, Dipesh Kafle

Contents

I'm using postgresql for all the things.

SQL Part (Question a)

a. Consider the employee data.

Employee (employee-id, employee-name, street, city)

Works (employee-name, company-name, salary)

Company (company-name, city)

Manages (employee-name, manager-name)

(i) Find the name of all employees who work for State Bank of India.

(ii) Find the names and cities of residence of all employees who work for State Bank of India.

(iii) Create a view for employee based on salary in ascending order.

(iv) Find all employees in the database who earn more than every employee of UCO Bank.

(v) Find the employee in the database who earn minimum in state bank of India

(vi) Create a Function that displays the employee who earn maximum in SBI.

(vii) Write a procedure which takes the city as input parameter and lists the names of all employees belonging to that city.

(viii) Write a function that will display the number of employees with salary more than 50k.

(ix) Write a procedure raise_sal which increases the salary of an employee. It accepts an employee number and salary increase amount.

(x) Create a Procedure that displays the details of managers working in a bank

Creating Schema and Putting dummy entries

```sql
-- Employee
drop table if exists Employee cascade;
CREATE TABLE Employee(
    empid INT NOT NULL,
    empname VARCHAR(50) NOT NULL,
    street VARCHAR(150) NOT NULL,
    city VARCHAR(150) NOT NULL,
    PRIMARY KEY (empid)
);
--  Company
drop table if exists Company cascade;
CREATE TABLE Company(
    companyname VARCHAR(50) NOT NULL,
    city VARCHAR(50) NOT NULL,
    PRIMARY KEY (companyname, city)
);
-- Works
drop table if exists Works cascade;
CREATE TABLE Works(
    empname VARCHAR(50) NOT NULL,
    companyname VARCHAR(50) NOT NULL,
    salary INT NOT NULL,
    PRIMARY KEY (empname, companyname)
);
-- Manages
drop table if exists Manages cascade;
CREATE TABLE Manages(
    empname VARCHAR(50) NOT NULL,
    managername VARCHAR(50) NOT NULL
);
--
--
-- Insertions
--
--
--
--
INSERT INTO employee(empid, empname, street, city)
VALUES (1, 'Dipesh', 'abc', 'Trichy');
INSERT INTO employee(empid, empname, street, city)
VALUES (2, 'Apurva', 'xyz', 'Delhi');
INSERT INTO employee(empid, empname, street, city)
VALUES (3, 'Hari', 'pqr', 'Kathmanu');
INSERT INTO employee(empid, empname, street, city)
VALUES (4, 'Ram', 'mnop', 'Chennai');
INSERT INTO employee(empid, empname, street, city)
VALUES (5, 'Ishika', 'Thane', 'Mumbai');
INSERT INTO employee(empid, empname, street, city)
VALUES (6, 'Hedhav', 'Pune 1', 'Pune');
INSERT INTO works(empname, companyname, salary)
VALUES('Dipesh', 'State Bank Of India', 95000);
INSERT INTO works(empname, companyname, salary)
VALUES('Hedhav', 'State Bank Of India', 100000);
INSERT INTO works(empname, companyname, salary)
VALUES('Apurva', 'UCO Bank', 50500);
INSERT INTO works(empname, companyname, salary)
VALUES('Ram', 'Google', 5220);
INSERT INTO works(empname, companyname, salary)
VALUES('Ishika', 'Google', 5000);
```

```
INSERT INTO works(empname, companyname, salary)
VALUES('Hari', 'UCO Bank', 5400);
INSERT INTO company(companyname, city)
VALUES('Google', 'Chennai');
INSERT INTO company(companyname, city)
VALUES('UCO Bank', 'Bangalore');
INSERT INTO company(companyname, city)
VALUES('State Bank Of India', 'Hyderabad');
INSERT INTO manages(empname, managername)
VALUES('Dipesh', 'Mary');
INSERT INTO manages(empname, managername)
VALUES('Apurva', 'Joe');
INSERT INTO manages(empname, managername)
VALUES('Hari', 'Sita');
INSERT INTO manages(empname, managername)
VALUES('Ishika', 'Piyal');
INSERT INTO manages(empname, managername)
VALUES('Ram', 'Gita');
INSERT INTO manages(empname, managername)
VALUES('Hedhav', 'Shweta');
--
--
--
```

(i) Find the name of all employees who work for State Bank of India.

Code

```
-- (i) Find the name of all employees who work for State Bank of India.
select employee.empname
from employee,
    works
where employee.empname = works.empname
    and works.companyname = 'State Bank Of India'
```

Output

```
 empname
---------
 Dipesh
 Hedhav
(2 rows)
```

(ii) Find the names and cities of residence of all employees who work for State Bank of India.

Code

```
-- (ii) Find the names and cities of residence of all employees who work for State Bank of India.
select employee.empname,
    employee.city
from employee,
    works
where employee.empname = works.empname
    and works.companyname = 'State Bank Of India';
```

Output

```
 empname |  city
---------+--------
 Dipesh  | Trichy
 Hedhav  | Pune
(2 rows)
```

(iii) Create a view for employee based on salary in ascending order.

Code

```
-- (iii) Create a view for employee based on salary in ascending order.
create view myview as
select employee.empname,
    works.salary,
    works.companyname,
    employee.street,
    employee.city
from employee,
    works
where employee.empname = works.empname
order by salary asc;
select *
from myview;
```

Output

```
select *
from myview;
 empname | salary |     companyname     | street |   city
---------+--------+---------------------+--------+----------
 Ishika  |   5000 | Google              | Thane  | Mumbai
 Ram     |   5220 | Google              | mnop   | Chennai
 Hari    |   5400 | UCO Bank            | pqr    | Kathmanu
 Apurva  |  50500 | UCO Bank            | xyz    | Delhi
 Dipesh  |  95000 | State Bank Of India | abc    | Trichy
 Hedhav  | 100000 | State Bank Of India | Pune 1 | Pune
```

```
(6 rows)
```

(iv) Find all employees in the database who earn more than every employee of UCO Bank.

Code

```sql
-- (iv) Find all employees in the database who earn more than every employee of UCO Bank.
select empname
from works
where salary > (
        select max(salary)
        from works
        where companyname = 'UCO Bank'
    )
```

Output

```
select empname
from works
where salary > (
        select max(salary)
        from works
        where companyname = 'UCO Bank'
    );
 empname
---------
 Dipesh
 Hedhav
(2 rows)
```

(v) Find the employee in the database who earn minimum in state bank of India

Code

```sql
-- (v) Find the employee in the database who earn minimum in state bank of India
select empname
from works
where salary =(
        select min(salary)
        from works
        where companyname = 'State Bank Of India'
    )
```

Output

```
select empname
from works
where salary =(
        select min(salary)
        from works
        where companyname = 'State Bank Of India'
    );
 empname
---------
 Dipesh
(1 row)
```

(vi) Create a Function that displays the employee who earn maximum in SBI.

Code

```
-- (vi) Create a Function that displays the employee who earn maximum in SBI.
drop function if exists max_earner_sbi;
CREATE OR REPLACE FUNCTION max_earner_sbi() RETURNS Table(employee_name varchar(200))
LANGUAGE plpgsql as $$ begin return query
select empname as employee_name
from works
where salary = (
        select max(salary)
        from works
        where companyname = 'State Bank Of India'
    );
end;
$$;
select *
from max_earner_sbi()
```

Output

```
select *
from max_earner_sbi();
 employee_name
---------------
 Hedhav
(1 row)
```

(vii) Write a procedure which takes the city as input parameter and lists the names of all employees belonging to that city.

Code

```
-- (vii) Write a procedure which takes the city as input parameter and lists the names of all
-- employees belonging to that city.
Drop procedure if exists city_employee;
CREATE OR REPLACE PROCEDURE city_employee(
        city_name varchar(200),
        INOUT names varchar(200) default null
    ) LANGUAGE plpgsql AS $$ BEGIN
select empname into names
from employee
where city = city_name;
end;
$$;
Call city_employee('Trichy');
Call city_employee('Delhi')
```

Output

```
Call city_employee('Trichy');
 names
--------
 Dipesh
(1 row)

Call city_employee('Delhi');
 names
--------
 Apurva
(1 row)
```

(viii) Write a function that will display the number of employees with salary more than 50k.

Code

```
-- (viii) Write a function that will display the number of employees with salary more than 50k.
drop function if exists fiftyk_plus_earners;
CREATE OR REPLACE FUNCTION fiftyk_plus_earners(out number_of_employees int)
LANGUAGE plpgsql as $$ begin
SELECT count(*) into number_of_employees
FROM works
WHERE works.salary > 50000;
end;
$$;
select *
from fiftyk_plus_earners()
```

Output

```
select *
from fiftyk_plus_earners();
 number_of_employees
---------------------
```

```
                  3
(1 row)
```

(ix) Write a procedure raise_sal which increases the salary of an employee. It accepts an employee number and salary increase amount.

Code

```
-- (ix) Write a procedure raise_sal which increases the salary of an employee. It accepts an
-- employee number and salary increase amount.
drop procedure if exists raise_sal;
CREATE PROCEDURE raise_sal(emp_no int, amount int) LANGUAGE plpgsql AS $$ begin
UPDATE works
SET salary = salary + amount
WHERE empname = (
        SELECT empname
        FROM employee
        WHERE empid = emp_no
    );
end;
$$;

-- Before
select *
from Works;
-- raise_sal for empid 1,which is Dipesh by 10k
CALL raise_sal(1, 10000);
-- After
select *
from Works
```

Output

```
CREATE PROCEDURE raise_sal(emp_no int, amount int) LANGUAGE plpgsql AS $$ begin
UPDATE works
SET salary = salary + amount
WHERE empname = (
        SELECT empname
        FROM employee
        WHERE empid = emp_no
    );
end;
$$;
CREATE PROCEDURE
select *
from Works;
 empname |     companyname     | salary
```

```
---------+---------------------+--------
 Dipesh  | State Bank Of India |  95000
 Hedhav  | State Bank Of India | 100000
 Apurva  | UCO Bank            |  50500
 Ram     | Google              |   5220
 Ishika  | Google              |   5000
 Hari    | UCO Bank            |   5400
(6 rows)

CALL raise_sal(1, 10000);
CALL
select *
from Works;
 empname |     companyname     | salary
---------+---------------------+--------
 Hedhav  | State Bank Of India | 100000
 Apurva  | UCO Bank            |  50500
 Ram     | Google              |   5220
 Ishika  | Google              |   5000
 Hari    | UCO Bank            |   5400
 Dipesh  | State Bank Of India | 105000
(6 rows)
```

(x) Create a Procedure that displays the details of managers working in a bank

Code

```
-- (x)  Create a Procedure that displays the details of managers working in a bank
drop procedure if exists manager_details;
CREATE PROCEDURE manager_details(
    bank_name varchar(200),
    INOUT managers varchar(200) default null
) LANGUAGE plpgsql AS $$ begin
SELECT array_agg(managername) into managers
FROM employee,
    works,
    company,
    Manages
WHERE employee.empname = works.empname
    AND works.companyname = company.companyname
    AND employee.empname = Manages.empname
    AND company.companyname = bank_name;
end;
$$;
Call manager_details('UCO Bank');
Call manager_details('State Bank Of India');
```

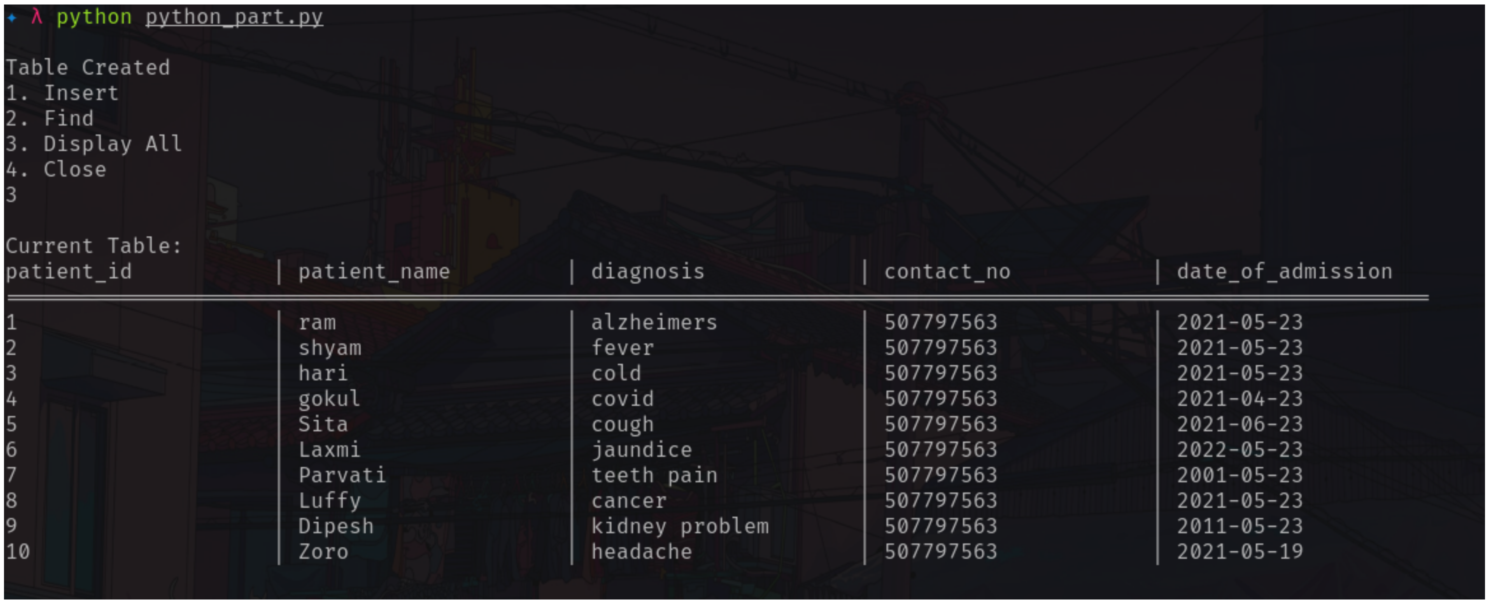
Output

```
Call manager_details('UCO Bank');
  managers
------------
 {Joe,Sita}
(1 row)

Call manager_details('State Bank Of India');
   managers
---------------
 {Mary,Shweta}
(1 row)
```

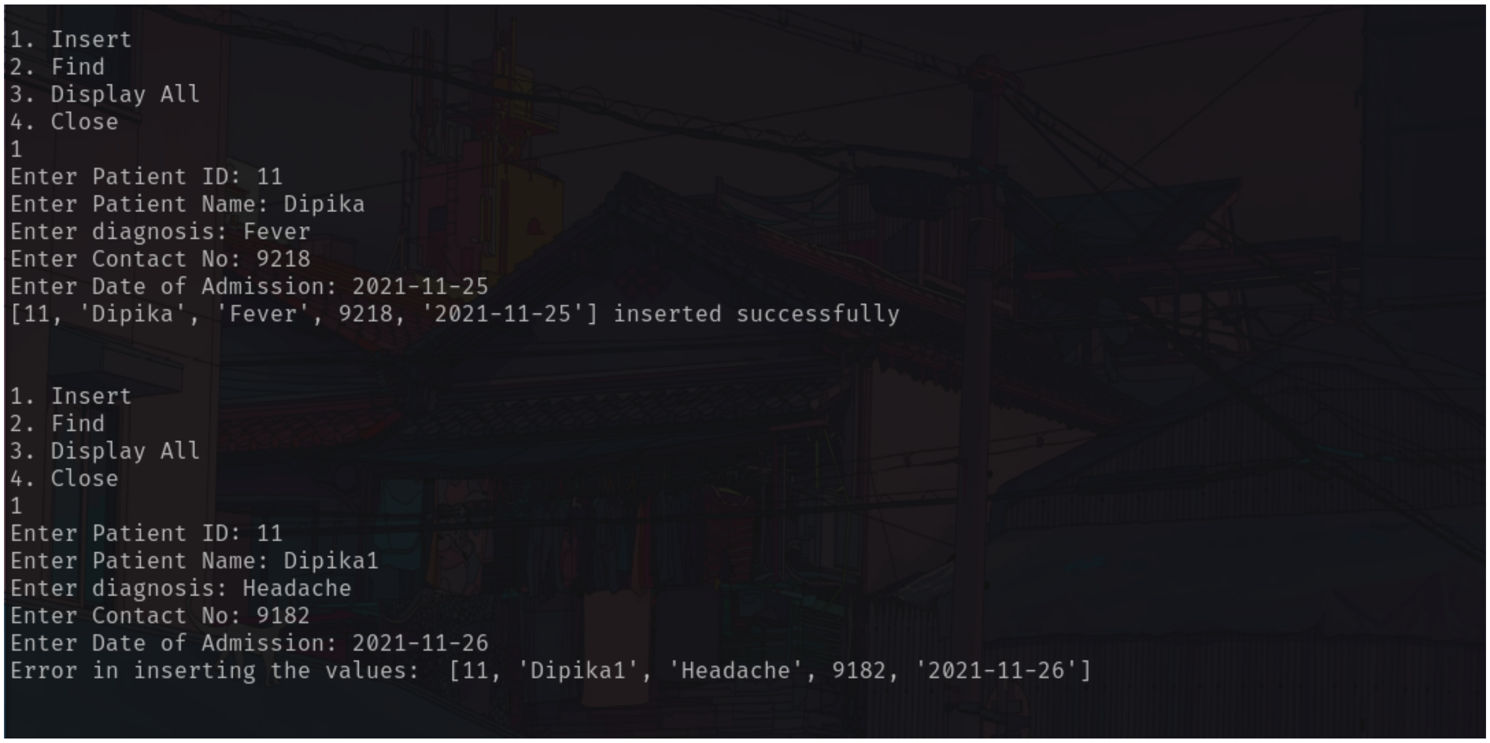
Python + SQL (Question b)

```
import psycopg2
from typing import List, Dict


class DatabaseSystem:
    def __init__(self, dbname: str, user: str, password: str):
        self.con = psycopg2.connect(
            host='localhost',
            database=dbname,
            user=user,
            password=password
        )
        self.cur = self.con.cursor()

    def createTable(self, tableName: str, query: str):
        self.cur.execute("DROP TABLE IF EXISTS " + tableName)
        self.cur.execute(query)
        print("\nTable Created")

    def insert(self, tableName: str, values: List, print_=False):
        query = "INSERT INTO " + tableName + " VALUES("
        query += ",".join(["%s"] * len(values))
        query += ")"
        try:
            self.cur.execute(query, values)
            if(print_):
                print(values, "inserted successfully")
        except:
            self.con.rollback()
            print("Error in inserting the values: ", values)

    def find(self, tableName: str, patient_id: int):
        self.cur.execute("SELECT * FROM " + tableName +
                         " WHERE  patient_id= %s", [patient_id])
        self.printCurrentCursor()

    def displayAll(self, tableName: str):
        print("\nCurrent Table:")
        self.cur.execute("SELECT * FROM " + tableName)
        self.printCurrentCursor()

    def printCurrentCursor(self):
        rows = self.cur.fetchall()
        columns = [desc[0] for desc in self.cur.description]
        header = " | ".join(["{:<20}"]*len(columns))
        if len(rows) == 0:
            return
        print(header.format(*columns))
        print('='*(20 * len(rows[0]) + 3*(len(rows[0])-1)))
        for row in rows:
            print(header.format(*row))

    def closeConnection(self):
        self.con.commit()
        self.con.close()


databaseName = "lab_exam"
username = "dipesh"
password = ""

database = DatabaseSystem(databaseName, username, password)

# b. Design a simple database for Patient details Management System using Python and MySQL

# The insert module must be able to accept the patient_id, patient_ name, diagnosis,
# Contact_no, Date_of_admission and store it in the database.
# The find module must be able to accept the Patient_id of the patient and display all the
# details of the corresponding patient.


# Patient Schema
#
# Patient_id - primary key
# Patient_name
# diagnosis
# contact_no
# date_of_admission
#

createTableQuery = '''CREATE TABLE IF NOT EXISTS
    Patients(patient_id INT PRIMARY KEY,
        patient_name VARCHAR(255),
        diagnosis VARCHAR(255),
        contact_no bigint, date_of_admission VARCHAR(20))'''


database.createTable("patients", createTableQuery)

database.insert('patients', [1, 'ram',
                             'alzheimers', '0507797563', '2021-05-23'])
database.insert('patients', [2, 'shyam', 'fever', '0507797563', '2021-05-23'])
database.insert('patients', [3, 'hari', 'cold', '0507797563', '2021-05-23'])
database.insert('patients', [4, 'gokul', 'covid', '0507797563', '2021-04-23'])
database.insert('patients', [5, 'Sita',
                             'cough', '0507797563', '2021-06-23'])
database.insert('patients', [6, 'Laxmi',
                             'jaundice', '0507797563', '2022-05-23'])
database.insert('patients', [7, 'Parvati', 'teeth pain',
                             '0507797563', '2001-05-23'])
database.insert('patients', [8, 'Luffy', 'cancer',
                             '0507797563', '2021-05-23'])
database.insert('patients', [9, 'Dipesh',
                             'kidney problem', '0507797563', '2011-05-23'])
database.insert('patients', [10, 'Zoro', 'headache',
                             '0507797563', '2021-05-19'])


def menu():
    print("1. Insert")
    print("2. Find ")
    print("3. Display All")
    print("4. Close")
    choice = int(input())
    if choice == 1:
        patient_id = int(input("Enter Patient ID: "))
        patient_name = input("Enter Patient Name: ")
        diagnosis = input("Enter diagnosis: ")

        contact_no = int(input("Enter Contact No: "))
        date_of_admission = input("Enter Date of Admission: ")
        database.insert('patients', [
                        patient_id, patient_name, diagnosis, contact_no, date_of_admission], True)
        print()
        print()
    elif choice == 2:
        patient_id = int(input("Enter patient id: "))
        database.find("patients", patient_id)
        print()
        print()
    elif choice == 3:
        database.displayAll('patients')
        print()
        print()
    elif choice == 4:
        database.closeConnection()
        exit()
    else:
        print("\nInvalid choice")
        print()
        print()
    menu()


menu()
```

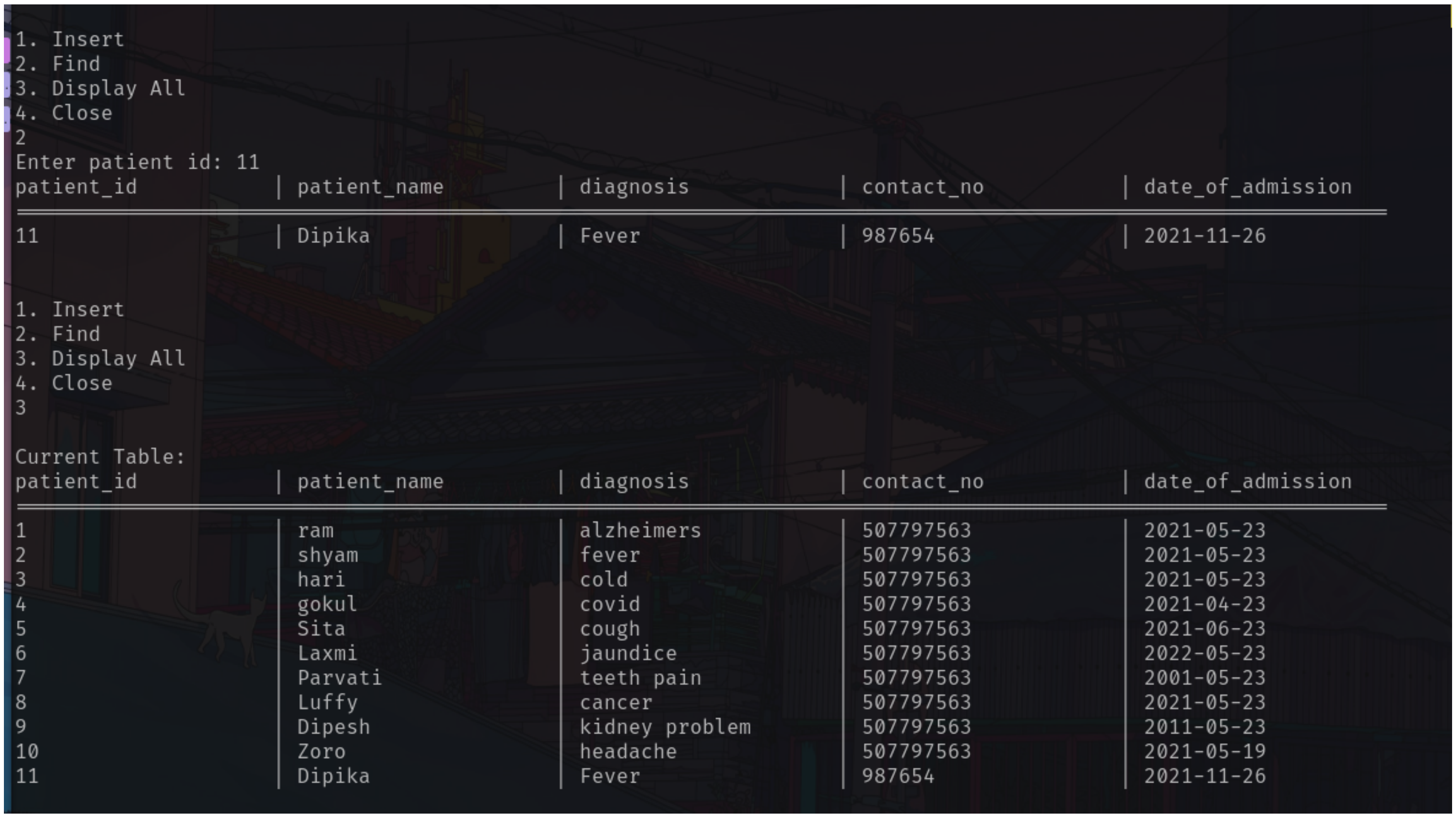
Output



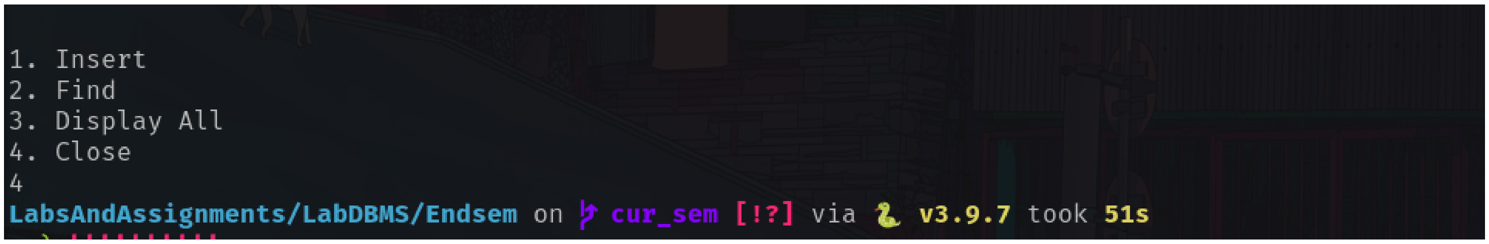
```
✦ λ python python_part.py

Table Created
1. Insert
2. Find
3. Display All
4. Close
3

Current Table:
patient_id            | patient_name          | diagnosis             | contact_no            | date_of_admission
==================================================================================================================
1                     | ram                   | alzheimers            | 507797563             | 2021-05-23
2                     | shyam                 | fever                 | 507797563             | 2021-05-23
3                     | hari                  | cold                  | 507797563             | 2021-05-23
4                     | gokul                 | covid                 | 507797563             | 2021-04-23
5                     | Sita                  | cough                 | 507797563             | 2021-06-23
6                     | Laxmi                 | jaundice              | 507797563             | 2022-05-23
7                     | Parvati               | teeth pain            | 507797563             | 2001-05-23
8                     | Luffy                 | cancer                | 507797563             | 2021-05-23
9                     | Dipesh                | kidney problem        | 507797563             | 2011-05-23
10                    | Zoro                  | headache              | 507797563             | 2021-05-19
```



```
1. Insert
2. Find
3. Display All
4. Close
1
Enter Patient ID: 11
Enter Patient Name: Dipika
Enter diagnosis: Fever
Enter Contact No: 9218
Enter Date of Admission: 2021-11-25
[11, 'Dipika', 'Fever', 9218, '2021-11-25'] inserted successfully


1. Insert
2. Find
3. Display All
4. Close
1
Enter Patient ID: 11
Enter Patient Name: Dipika1
Enter diagnosis: Headache
Enter Contact No: 9182
Enter Date of Admission: 2021-11-26
Error in inserting the values:  [11, 'Dipika1', 'Headache', 9182, '2021-11-26']
```



```
1. Insert
2. Find
3. Display All
4. Close
2
Enter patient id: 11
patient_id            | patient_name          | diagnosis             | contact_no            | date_of_admission
══════════════════════════════════════════════════════════════════════════════════════════════════════════════
11                    | Dipika                | Fever                 | 987654                | 2021-11-26


1. Insert
2. Find
3. Display All
4. Close
3

Current Table:
patient_id            | patient_name          | diagnosis             | contact_no            | date_of_admission
══════════════════════════════════════════════════════════════════════════════════════════════════════════════
1                     | ram                   | alzheimers            | 507797563             | 2021-05-23
2                     | shyam                 | fever                 | 507797563             | 2021-05-23
3                     | hari                  | cold                  | 507797563             | 2021-05-23
4                     | gokul                 | covid                 | 507797563             | 2021-04-23
5                     | Sita                  | cough                 | 507797563             | 2021-06-23
6                     | Laxmi                 | jaundice              | 507797563             | 2022-05-23
7                     | Parvati               | teeth pain            | 507797563             | 2001-05-23
8                     | Luffy                 | cancer                | 507797563             | 2021-05-23
9                     | Dipesh                | kidney problem        | 507797563             | 2011-05-23
10                    | Zoro                  | headache              | 507797563             | 2021-05-19
11                    | Dipika                | Fever                 | 987654                | 2021-11-26
```



```
1. Insert
2. Find
3. Display All
4. Close
4
LabsAndAssignments/LabDBMS/Endsem on  cur_sem [!?] via 🐍 v3.9.7 took 51s
```