Lab 3 DBMS

106119029

Question 1

Consider the COVID-19 database in India. Students are advised to read all the questions and then create a table with necessary details.

Patient (P_ID, Patient_Name, Sex, Age, Area, City, H_ID)

Test_Report (T_ID, P_ID ,H_ID, Reporting_date, Test_result, Discharge_date)

Hospital (H_ID, Hospital_Name, Location, State)

1.Create a procedure to display the details of a patient record for a given Patient ID.

2.Create a procedure to add details of a new patient record into patient table.

3.Write a procedure that lists the highest cases reported in a district of any particular state. Use the procedure named Find_highest which finds the highest cases for the given State.

4.Write a procedure to list the hospital, which has fastest recovery.

5. Create a procedure to delete a record from patient table

6.Write a function to display the patient details from the Patient table.

7.Writea function to list the state, which has reported with maximum child COVID cases.

8.Write a function to find the hotpot area in a district based on the Test results.

9.Write a function to display total number of male and female patients tested for COVID of which how many are reported with positive in a particular state.

10.Write a function to display the average days for the recovery of child, adults and senior citizen of a particular hospital.

Creating Tables

```
-- drop tables if exists cascade
drop table if exists Patient cascade;
drop table if exists Test_Report cascade;
drop table if exists Hospital cascade;

-- CREATE a table named Patient with P_id, Patient_name, sex, age, area, city, H_id,
CREATE table Patient(
    P_id int primary key,
    Patient_name varchar(20),
```

```
    sex varchar(1),
    age int,
    area varchar(20),
    city varchar(20),
    H_id int
);


-- CREATE a table named Test_Report with T_id, P_id, H_id, reporting_date, test_result, dis
CREATE table Test_Report(
    T_id int primary key,
    P_id int,
    H_id int,
    reporting_date date,
    test_result varchar(20),
    discharge_date date
);

-- CREATE a table named Hospital with H_id, Hospital_Name, Location, State
CREATE table Hospital(
    H_id int primary key,
    Hospital_Name varchar(20),
    Location varchar(20),
    State varchar(20)
);


-- Alter table Hospital H_id is foriegn key to Hospital
alter table Patient add constraint FK_H_id foreign key(H_id)
references Hospital(H_id);

-- Alter Table Test_Report P_id is foriegn key to Patient, H_id is foriegn key to Hospital
alter table Test_Report add constraint FK_P_id foreign key(P_id)
references Patient(P_id);
alter table Test_Report add constraint FK_H_id foreign key(H_id)
references Hospital(H_id);
```

Populating Tables

```
-- Inserting Hospitals
insert into Hospital values(1, 'Apollo Hospital', 'City1', 'Maharashtra');
insert into Hospital values(2, 'Hospital2', 'City2', 'Maharashtra');
insert into Hospital values(3, 'Hospital3', 'Delhi', 'State2');
insert into Hospital values(4, 'Hospital4', 'City4', 'State3');
insert into Hospital values(5, 'Hospital5', 'City5', 'State4');
```

```
-- Inserting Patients
insert into Patient values(1, 'Patient1', 'M', 13, 'Area1', 'City1', 1);
insert into Patient values(2, 'Patient2', 'F', 14, 'Area1', 'City1', 1);
insert into Patient values(3, 'Patient3', 'M', 17, 'Area2', 'City1', 2);
insert into Patient values(4, 'Patient4', 'F', 20, 'Area2', 'City1', 3);
insert into Patient values(5, 'Patient5', 'M', 25, 'Area2', 'City1', 3);
insert into Patient values(6, 'Patient6', 'F', 35, 'Area1', 'City2', 2);
insert into Patient values(7, 'Patient7', 'M', 47, 'Area1', 'City2', 4);
insert into Patient values(8, 'Patient8', 'F', 59, 'Area1', 'Delhi', 5);
insert into Patient values(9, 'Patient9', 'M', 65, 'Area2', 'Delhi', 1);
insert into Patient values(10, 'Patient10', 'F', 75, 'Area1', 'City4', 2);
insert into Patient values(11, 'Patient11', 'F', 20, 'Area1', 'City4', 2);
insert into Patient values(12, 'Patient12', 'M', 13, 'Area1', 'City1', 1);
insert into Patient values(13, 'Patient13', 'M', 59, 'Area1', 'Delhi', 5);

-- Insert test_reports for the given patients AND hospitals
insert into Test_Report values(1, 1, 1, '2021-05-01', 'Positive', '2021-05-20');
insert into Test_Report values(2, 2, 1, '2021-05-02', 'Positive', '2021-05-25');
insert into Test_Report values(3, 3, 2, '2021-05-03', 'Negative', '2021-05-13');
insert into Test_Report values(4, 4, 3, '2021-05-04', 'Negative', '2021-05-10');
insert into Test_Report values(5, 5, 3, '2021-05-05', 'Negative', '2021-05-13');
insert into Test_Report values(6, 6, 2, '2021-07-06', 'Positive', '2021-08-03');
insert into Test_Report values(7, 7, 4, '2021-07-07', 'Positive', '2021-08-07');
insert into Test_Report values(8, 8, 5, '2021-07-08', 'Positive', '2021-07-28');
insert into Test_Report values(9, 9, 1, '2021-07-09', 'Positive', '2021-08-05');
insert into Test_Report values(10, 10, 2, '2021-07-10', 'Negative', '2021-08-09');
insert into Test_Report values(11, 11, 2, '2021-07-15', 'Negative', '2021-08-05');
insert into Test_Report values(12, 12, 1, '2021-05-01', 'Positive', '2021-05-20');
insert into Test_Report values(13, 13, 5, '2021-07-01', 'Positive', '2021-07-28');
```

1.1

Code

```
-------------------------------------------------------------------------------
-- 1.Create a procedure to display the details of a
-- patient record for a given Patient ID.
-------------------------------------------------------------------------------

CREATE PROCEDURE DisplayPatientProc(
    pid int,
    INOUT _P_id int default null,
    INOUT _Patient_name varchar(20) default null,
    INOUT _sex varchar(1) default null,
    INOUT _age int default null,
    INOUT _area varchar(20) default null,
```

```
    INOUT _city varchar(20) default null,
    INOUT _H_id int default null
    )
LANGUAGE plpgsql AS
$$
begin
    select * INTO _P_id,_Patient_name,_sex,_age,_area,_city,_H_id from Patient
        where P_id = pid;
end;
$$;

CALL DisplayPatientProc(3);
```

Output

```
CALL DisplayPatientProc(3);
 _p_id | _patient_name | _sex | _age | _area | _city | _h_id
-------+---------------+------+------+-------+-------+-------
     3 | Patient3      | M    |   17 | Area2 | City1 |     2
(1 row)
```

1.2

Code

```
-------------------------------------------------------------------------------
-- 2. Create a procedure to add details of a new patient record into patient table.
-------------------------------------------------------------------------------
DROP PROCEDURE IF EXISTS AddPatientProc;

CREATE PROCEDURE AddPatientProc(
    P_id int,
    Patient_name varchar(20),
    sex varchar(1),
    age int,
    area varchar(20),
    city varchar(20),
    H_id int)
LANGUAGE plpgsql AS
$$
begin
    insert into Patient values(P_id,Patient_name,sex,age,area,city,H_id);
end;
$$;
```

```
CALL AddPatientProc(15, 'Patient15', 'F', 20, 'Area1', 'City4', 3);
select * from patient where p_id = 15;
```

Output

1.3

Code

```
-------------------------------------------------------------------------------
-- 3.Write a procedure that lists the highest cases
-- reported in a district of any particular state.
-------------------------------------------------------------------------------
DROP PROCEDURE IF EXISTS Find_highest;

CREATE PROCEDURE Find_highest(_state varchar(20), INOUT no_of_cases int default 0)
LANGUAGE plpgsql AS
$$
begin
    select count(*) as cases INTO no_of_cases
    FROM Hospital, Test_Report
    WHERE Test_Report.H_id = Hospital.H_id and Hospital.State = _state
    GROUP BY Hospital.location
    ORDER BY cases DESC
    LIMIT 1;
end;
$$;

CALL Find_highest('Maharashtra');
```

Output

```
DROP PROCEDURE IF EXISTS Find_highest;
DROP PROCEDURE
CREATE PROCEDURE Find_highest(_state varchar(20), INOUT no_of_cases int default 0)
LANGUAGE plpgsql AS
$$
begin
    select count(*) as cases INTO no_of_cases
    FROM Hospital, Test_Report
    WHERE Test_Report.H_id = Hospital.H_id and Hospital.State = _state
    GROUP BY Hospital.location
    ORDER BY cases DESC
    LIMIT 1;
```

```
end;
$$;
CREATE PROCEDURE
CALL Find_highest('Maharashtra');
 no_of_cases
-------------
           4
(1 row)
```

1.4

Code

```
-------------------------------------------------------------------------------------
-- 4. Write a procedure to list the hospital, which has fastest recovery
-------------------------------------------------------------------------------------
DROP PROCEDURE IF EXISTS FastestHospital;

CREATE PROCEDURE FastestHospital(INOUT name_of_hospital varchar(20) default 'none')
LANGUAGE plpgsql AS
$$
begin
    SELECT Hospital_Name INTO name_of_hospital
    FROM Test_Report, Hospital
    WHERE Test_Report.H_id = Hospital.H_id AND test_result = 'Negative'
    GROUP BY Hospital_Name ORDER by avg(discharge_date - reporting_date)
    LIMIT 1;
end;
$$;

CALL FastestHospital();
```

Output

```
DROP PROCEDURE IF EXISTS FastestHospital;
DROP PROCEDURE
CREATE PROCEDURE FastestHospital(INOUT name_of_hospital varchar(20) default 'none')
LANGUAGE plpgsql AS
$$
begin
    SELECT Hospital_Name INTO name_of_hospital
    FROM Test_Report, Hospital
    WHERE Test_Report.H_id = Hospital.H_id AND test_result = 'Negative'
    GROUP BY Hospital_Name ORDER by avg(discharge_date - reporting_date)
    LIMIT 1;
end;
$$;
```

```
CREATE PROCEDURE
CALL FastestHospital();
 name_of_hospital
------------------
 Hospital3
(1 row)
```

1.5

Code

```
-------------------------------------------------------------------------------------
-- 5. Create a procedure to delete a record from patient table
-------------------------------------------------------------------------------------
DROP PROCEDURE IF EXISTS DeletePatientProc;

CREATE PROCEDURE DeletePatientProc(Pid int)
LANGUAGE plpgsql AS
$$
begin
    delete from Test_Report where P_id = Pid;
    delete from Patient where P_id = Pid;
end;
$$;

select * from patient where p_id = 15;
CALL DeletePatientProc(15);
select * from patient where p_id = 15;
```

Output

```
DROP PROCEDURE IF EXISTS DeletePatientProc;
psql:lab3.sql:207: NOTICE:  procedure deletepatientproc() does not exist, skipping
DROP PROCEDURE
CREATE PROCEDURE DeletePatientProc(Pid int)
LANGUAGE plpgsql AS
$$
begin
    delete from Test_Report where P_id = Pid;
    delete from Patient where P_id = Pid;
end;
$$;
CREATE PROCEDURE
select * from patient where p_id = 15;
 p_id | patient_name | sex | age | area  | city  | h_id
------+--------------+-----+-----+-------+-------+------
   15 | Patient15    | F   |  20 | Area1 | City4 |    3
```

```
(1 row)

CALL DeletePatientProc(15);
CALL
select * from patient where p_id = 15;
 p_id | patient_name | sex | age | area | city | h_id
------+--------------+-----+-----+------+------+------
(0 rows)
```

1.6

Code

```
-------------------------------------------------------------------------------------
-- 6. Write a function to display the patient details from the Patient table.
-------------------------------------------------------------------------------------
DROP FUNCTION IF EXISTS DisplayPatient;
CREATE or REPLACE FUNCTION DisplayPatient()
    RETURNS Table(P_id int,
    Patient_name varchar(20),
    sex varchar(1),
    age int,
    area varchar(20),
    city varchar(20),
    H_id int)
    language plpgsql
as
$$
begin
    return query select * from Patient;
end;
$$;

SELECT * FROM DisplayPatient();
```

Output

```
DROP FUNCTION IF EXISTS DisplayPatient;
DROP FUNCTION
CREATE or REPLACE FUNCTION DisplayPatient()
    RETURNS Table(P_id int,
    Patient_name varchar(20),
    sex varchar(1),
    age int,
    area varchar(20),
    city varchar(20),
    H_id int)
```

```
    language plpgsql
as
$$
begin
    return query select * from Patient;
end;
$$;
CREATE FUNCTION
SELECT * FROM DisplayPatient();
 p_id | patient_name | sex | age | area  | city  | h_id
------+--------------+-----+-----+-------+-------+------
    1 | Patient1     | M   |  13 | Area1 | City1 |    1
    2 | Patient2     | F   |  14 | Area1 | City1 |    1
    3 | Patient3     | M   |  17 | Area2 | City1 |    2
    4 | Patient4     | F   |  20 | Area2 | City1 |    3
    5 | Patient5     | M   |  25 | Area2 | City1 |    3
    6 | Patient6     | F   |  35 | Area1 | City2 |    2
    7 | Patient7     | M   |  47 | Area2 | City2 |    4
    8 | Patient8     | F   |  59 | Area1 | Delhi |    5
    9 | Patient9     | M   |  65 | Area2 | Delhi |    1
   10 | Patient10    | F   |  75 | Area1 | City4 |    2
   11 | Patient11    | F   |  20 | Area1 | City4 |    2
(11 rows)
```

1.7

Code

```
-------------------------------------------------------------------------------
-- 7.Write a function to list the state, which has
-- reported with maximum child COVID cases.
-------------------------------------------------------------------------------

DROP FUNCTION IF EXISTS DisplayStateMaxCovid;

CREATE or REPLACE FUNCTION DisplayStateMaxCovid()
RETURNS Table(_State varchar(20),_Cases bigint)
language plpgsql
as
$$
BEGIN
    RETURN query
    SELECT State, count(*)
    FROM Patient, Test_Report, Hospital
    WHERE Patient.P_id = Test_Report.P_id
        AND Test_Report.H_id = Hospital.H_id AND age < 18
```

```
    GROUP BY State
    order by count(*) desc
    LIMIT 1;
END;
$$;

SELECT * FROM DisplayStateMaxCovid();
```

Output

```
SELECT * FROM DisplayStateMaxCovid();
   _state    | _cases
-------------+--------
 Maharashtra |      3
(1 row)
```

1.8

Code

```
-------------------------------------------------------------------------------------
-- 8)  Write a function to find the hotspot area in
-- a district based on the Test results.
-------------------------------------------------------------------------------------
DROP FUNCTION IF EXISTS DisplayHotspotArea;

CREATE or REPLACE FUNCTION DisplayHotspotArea()
RETURNS Table(_City varchar(20),_Area varchar(20), _Cases bigint)
language plpgsql
as
$$
BEGIN
    DROP view IF EXISTS xyz;
    CREATE view xyz as
    SELECT area, city, count(*) as cases
    FROM Patient, Test_Report
    WHERE Patient.P_id = Test_Report.P_id AND test_result = 'Positive'
    GROUP BY city, area;

    RETURN query
    SELECT DISTINCT temp.city, xyz.area, temp.cases
    FROM (
            SELECT city, max(cases) as cases
            FROM xyz
            GROUP BY city
        ) as temp, xyz
    WHERE temp.cases = xyz.cases;
```

```
END;
$$;

SELECT * FROM DisplayHotspotArea();
```

Output

```
DROP FUNCTION IF EXISTS DisplayHotspotArea;
DROP FUNCTION
CREATE or REPLACE FUNCTION DisplayHotspotArea()
RETURNS Table(_City varchar(20),_Area varchar(20), _Cases bigint)
language plpgsql
as
$$
BEGIN
    DROP view IF EXISTS xyz;
    CREATE view xyz as
    SELECT area, city, count(*) as cases
    FROM Patient, Test_Report
    WHERE Patient.P_id = Test_Report.P_id AND test_result = 'Positive'
    GROUP BY city, area;

    RETURN query
    SELECT DISTINCT temp.city, xyz.area, temp.cases
    FROM (
            SELECT city, max(cases) as cases
            FROM xyz
            GROUP BY city
        ) as temp, xyz
    WHERE temp.cases = xyz.cases;
END;
$$;
CREATE FUNCTION
SELECT * FROM DisplayHotspotArea();
psql:lab3.sql:299: NOTICE:  view "xyz" does not exist, skipping
 _city | _area | _cases
-------+-------+--------
 City1 | Area1 |      3
 City2 | Area1 |      2
 Delhi | Area1 |      2
(3 rows)
```

1.9

Code

```
-------------------------------------------------------------------------------
-- 9th Write a function to display total number of
-- male and female patients tested for COVID of
-- which how many are reported with positive in a particular state.
-------------------------------------------------------------------------------

DROP FUNCTION IF EXISTS GenderWisePatients;

CREATE or REPLACE FUNCTION GenderWisePatients(_state varchar(20),
    out total_people bigint,
    out male_count bigint,
    out female_count bigint)
language plpgsql
as
$$
BEGIN
    select count(*) into total_people
    FROM Patient, Hospital
    WHERE Hospital.H_id = Patient.H_id and Hospital.state = _state;

    select count(*) into male_count
    FROM Patient, Hospital, Test_Report
    WHERE Hospital.H_id = Patient.H_id
        and Test_Report.P_id = Patient.P_id
        and Test_Report.test_result = 'Positive'
        and Patient.sex = 'M'
        and Hospital.State = _state;

    select count(*) into female_count
    FROM Patient, Hospital, Test_Report
    WHERE Hospital.H_id = Patient.H_id
        and Test_Report.P_id = Patient.P_id
        and Test_Report.test_result = 'Positive'
        and Patient.sex = 'F'
        and Hospital.State = _state;
END;
$$;

select * FROM GenderWisePatients('Maharashtra');
```

Output

```
select * FROM GenderWisePatients('Maharashtra');
 total_people | male_count | female_count
--------------+------------+--------------
            7 |          2 |            2
```

```
(1 row)
```

1.10

Code

```
-----------------------------------------------------------------------
-- 10.Write a function to display the average days
-- for the recovery of child, adults and senior
-- citizen of a particular hospital.
-----------------------------------------------------------------------

DROP FUNCTION IF EXISTS RecoveryDays;

CREATE or REPLACE FUNCTION RecoveryDays(h_name varchar(20),
    out child bigint,
    out adult bigint,
    out senior bigint)
language plpgsql
as
$$
BEGIN
    SELECT avg(discharge_date - reporting_date) into child
    FROM Patient, Hospital, Test_Report
    WHERE Hospital.H_id = Patient.H_id
        and Test_Report.P_id = Patient.P_id
        AND test_result = 'Negative'
        AND age < 18
    GROUP BY Hospital_Name
    HAVING Hospital_Name = h_name;

    SELECT avg(discharge_date - reporting_date) into adult
    FROM Patient, Hospital, Test_Report
    WHERE Hospital.H_id = Patient.H_id
        and Test_Report.P_id = Patient.P_id
        AND test_result = 'Negative'
        AND age >= 18 AND age < 61
    GROUP BY Hospital_Name
    HAVING Hospital_Name = h_name;

    SELECT avg(discharge_date - reporting_date) into senior
    FROM Patient, Hospital, Test_Report
    WHERE Hospital.H_id = Patient.H_id
        and Test_Report.P_id = Patient.P_id
        AND test_result = 'Negative'
        AND age > 60
```

```
    GROUP BY Hospital_Name
    HAVING Hospital_Name = h_name;
END;
$$;

SELECT * FROM RecoveryDays('Hospital2');
```

Output

```
DROP FUNCTION IF EXISTS RecoveryDays;
DROP FUNCTION
CREATE or REPLACE FUNCTION RecoveryDays(h_name varchar(20),
    out child bigint,
    out adult bigint,
    out senior bigint)
language plpgsql
as
$$
BEGIN
    SELECT avg(discharge_date - reporting_date) into child
    FROM Patient, Hospital, Test_Report
    WHERE Hospital.H_id = Patient.H_id
        and Test_Report.P_id = Patient.P_id
        AND test_result = 'Negative'
        AND age < 18
    GROUP BY Hospital_Name
    HAVING Hospital_Name = h_name;

    SELECT avg(discharge_date - reporting_date) into adult
    FROM Patient, Hospital, Test_Report
    WHERE Hospital.H_id = Patient.H_id
        and Test_Report.P_id = Patient.P_id
        AND test_result = 'Negative'
        AND age >= 18 AND age < 61
    GROUP BY Hospital_Name
    HAVING Hospital_Name = h_name;

    SELECT avg(discharge_date - reporting_date) into senior
    FROM Patient, Hospital, Test_Report
    WHERE Hospital.H_id = Patient.H_id
        and Test_Report.P_id = Patient.P_id
        AND test_result = 'Negative'
        AND age > 60
    GROUP BY Hospital_Name
    HAVING Hospital_Name = h_name;
END;
```

```
$$;
CREATE FUNCTION
SELECT * FROM RecoveryDays('Hospital2');
 child | adult | senior
-------+-------+--------
    10 |    21 |     30
(1 row)
```