Savitribai Phule Pune University

S.Y. B.C.A. (Science) (Semester-III) Practical Examination

BCA 235: s(Database Management Systems II Laboratory)

Duration: 3Hrs. **Max Marks: 35+15=50**

Note: - 1. Read the questions carefully and insert data in the database accordingly.
2. Insert sufficient number of records in the database.
3. No query should generate empty output.
4. For count queries output should be more than 2 records. (If asked)

Create the following database in 3NF using PostgresSQL. [Total Marks: 10]

Q1) Consider the following database of Bus-Transport System. Many buses run on one route. Driversare allotted to the buses shift-wise.

Bus (Bus_no int , capacity int , depot_name varchar (20))

Route (Route_no int, source varchar (20), destination varchar(20), no_of_stations int)

Driver (Driver_no int , driver_name varchar(20), license_no int, address varchar(20), age int , salary float)

Relationship:

Bus and Route related with many to one relationship.

Bus and Driver related with many to many relationship with descriptive attributes, Shift – it can be 1 (Morning) or 2 (Evening) and Date_of_duty_allotted.

Constraints: Primary key, license_no must be unique, Bus capacity should not be null.

Create a View: [**10]**

1. To display driver names working in both shifts.
2. To display route details on which Bus_no 101 is running.

Q.2) Using above database solve following questions: **[Total Marks: 20]**

1. Write a trigger after deleting the bus record which has capacity < 20. Display the appropriate message. **[10]**

2. Write a cursor to display details of buses running on route_no = 1. **[10]**

Q.3) External Viva **[05]**

Q.4) Internal Evaluation **[15]**

BUS-TRANSPORT SYSTEM

CREATE TABLE Bus (Bus_no INT PRIMARY KEY, capacity INT NOT NULL, depot_name VARCHAR(20));

CREATE TABLE Route (Route_no INT PRIMARY KEY, source VARCHAR(20), destination VARCHAR(20), no_of_stations INT);

CREATE TABLE Driver (Driver_no INT PRIMARY KEY, driver_name VARCHAR(20), license_no INT UNIQUE, address VARCHAR(20), age INT, salary DECIMAL);

CREATE TABLE Bus_Driver (Bus_no INT, Driver_no INT, Shift INT CHECK (Shift IN (1, 2)), Date_of_duty_allotted DATE, PRIMARY KEY (Bus_no, Driver_no, Shift, Date_of_duty_allotted), FOREIGN KEY (Bus_no) REFERENCES Bus(Bus_no), FOREIGN KEY (Driver_no) REFERENCES Driver(Driver_no));

CREATE TABLE Bus_Route (Bus_no INT PRIMARY KEY, Route_no INT, FOREIGN KEY (Bus_no) REFERENCES Bus(Bus_no), FOREIGN KEY (Route_no) REFERENCES Route(Route_no));

INSERT INTO Route VALUES (1, 'Mumbai', 'Pune', 5), (2, 'Nashik', 'Mumbai', 7), (3, 'Ahmednagar', 'Aurangabad', 6), (4, 'Pune', 'Nagpur', 10), (5, 'Mumbai', 'Goa', 8);

INSERT INTO Bus VALUES (101, 30, 'Depot A'), (102, 40, 'Depot B'), (103, 50, 'Depot C'), (104, 35, 'Depot D'), (105, 30, 'Depot E');

INSERT INTO Driver VALUES (1, 'Rajesh', 123456, 'Mumbai', 45, 25000), (2, 'Amit', 654321, 'Pune', 35, 18000), (3, 'Sunil', 789456, 'Nashik', 50, 22000), (4, 'Suresh', 111222, 'Aurangabad', 40, 24000), (5, 'Mahesh', 222333, 'Nagpur', 29, 26000), (6, 'Anil', 333444, 'Goa', 55, 27000);

INSERT INTO Bus_Route VALUES (101, 1), (102, 2), (103, 3), (104, 4), (105, 5);

INSERT INTO Bus_Driver VALUES (101, 1, 1, '2024-10-10'), (101, 1, 2, '2024-10-10'), (102, 2, 1, '2024-10-11'), (102, 3, 2, '2024-10-11'), (103, 4, 1, '2024-10-12'), (103, 5, 2, '2024-10-12'), (104, 5, 1, '2024-10-13'), (104, 6, 2, '2024-10-13'), (105, 1, 1, '2024-10-14'), (105, 6, 2, '2024-10-14');

Q.1) Create a View:

1. CREATE VIEW Drivers_Both_Shifts AS SELECT d.driver_name FROM Driver d JOIN Bus_Driver bd1 ON d.Driver_no = bd1.Driver_no AND bd1.Shift = 1 JOIN Bus_Driver bd2 ON d.Driver_no = bd2.Driver_no AND bd2.Shift = 2 GROUP BY d.driver_name;
   SELECT * FROM Drivers_Both_Shifts;

2. CREATE VIEW Route_Bus_101 AS SELECT r.Route_no, r.source, r.destination, r.no_of_stations FROM Route r JOIN Bus_Route br ON r.Route_no = br.Route_no WHERE br.Bus_no = 101;
   SELECT * FROM Route_Bus_101;

Q.2) Using above database solve following questions:

1.
```
CREATE OR REPLACE FUNCTION after_delete_bus()
RETURNS TRIGGER AS $$
BEGIN
   IF OLD.capacity < 20 THEN
      RAISE NOTICE 'Bus with capacity less than 20 has been
deleted.';
   END IF;
   RETURN OLD;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER trigger_after_delete_bus
AFTER DELETE ON Bus
FOR EACH ROW
EXECUTE FUNCTION after_delete_bus();

INSERT INTO Bus VALUES (103, 15, 'Depot C');

DELETE FROM Bus WHERE Bus_no = 103;

DELETE FROM Bus WHERE Bus_no = 101;
```

2.
```
CREATE OR REPLACE FUNCTION display_buses_on_route_1()
RETURNS VOID AS $$
DECLARE
   rec RECORD;
BEGIN
   FOR rec IN
      SELECT b.Bus_no, b.capacity, b.depot_name
      FROM Bus b
      JOIN Bus_Route br ON b.Bus_no = br.Bus_no
      WHERE br.Route_no = 1
   LOOP
      RAISE NOTICE 'Bus No: %, Capacity: %, Depot: %',
rec.Bus_no, rec.capacity, rec.depot_name;
   END LOOP;

   IF NOT FOUND THEN
      RAISE NOTICE 'No buses found on Route No: 1';
   END IF;
END;
$$ LANGUAGE plpgsql;

SELECT display_buses_on_route_1();
```