Lab 4 CS254

Name: Abhiraj Mengade
Roll no: 201cs102

Q1. Create a db of Lecturers with 10 tuples which consist of First name, last name, age, city, state, pin code, subject, salary and years of experience.

```
CREATE TABLE Lectures(
    First_name varchar(255),
    Last_name varchar(255),
    Age int,
    City varchar(255),
    State varchar(255),
    Pin_code varchar(20),
    Subject varchar(255),
    Salary int,
    Years_of_experience int);

INSERT INTO Lectures VALUES
    ("Hello", "Boy", 19, "Bangalore", "Karnataka", "560010", "C", 25000,
1),
    ("Boo", "Girl", 26, "Mumbai", "Maharashtra", "410010", "OS", 20000,
3),
    ("Ram", "Di", 23, "Mangalore", "Karnataka", "575025", "C", 22000, 0),
    ("Trans", "Long", 27, "Delhi", NULL, "101010", "DBMS", 25000, 4),
    ("John", "Cena", 25, "Bangalore", "Karnataka", "560020", "C", 24000,
3),
    ("Robert", "Ford", 24, "Pune", "Maharashtra", "410020", "OS", 20000,
3),
    ("Augustus", "Roman", 22, "Jaipur", "Rajasthan", "301010", "DBMS",
21000, 2),
```

```
    ("Archie", "Andrews", 20, "Jaisalmer", "Rajasthan", "301020", "OS",
19000, 0),
    ("Flaber", "Fisk", 27, "Mumbai", "Maharashtra", "410030", "DBMS",
22000, 1),
    ("Tony", "Stark", 22, "Bangalore", "Karnataka", "560030", "C", 30000,
3);

SELECT First_name, Last_name, Age, Salary FROM Lectures WHERE Age >= 25
AND Salary >=20000;
```

a.) Write a query to find the salary where age <=25 and salary >=20000

```
CREATE TABLE Lectures(
    First_name varchar(255),
    Last_name varchar(255),
    Age int,
    City varchar(255),
    State varchar(255),
    Pin_code varchar(20),
    Subject varchar(255),
    Salary int,
    Years_of_experience int);

INSERT INTO Lectures VALUES
    ("Hello", "Boy", 19, "Bangalore", "Karnataka", "560010", "C", 25000, 1),
    ("Boo", "Girl", 26, "Mumbai", "Maharashtra", "410010", "OS", 20000, 3),
    ("Ram", "Di", 23, "Mangalore", "Karnataka", "575025", "C", 22000, 0),
    ("Trans", "Long", 27, "Delhi", NULL, "101010", "DBMS", 25000, 4),
    ("John", "Cena", 25, "Bangalore", "Karnataka", "560020", "C", 24000, 3),
    ("Robert", "Ford", 24, "Pune", "Maharashtra", "410020", "OS", 20000, 3),
    ("Augustus", "Roman", 22, "Jaipur", "Rajasthan", "301010", "DBMS", 21000, 2),
    ("Archie", "Andrews", 20, "Jaisalmer", "Rajasthan", "301020", "OS", 19000, 0),
    ("Flaber", "Fisk", 27, "Mumbai", "Maharashtra", "410030", "DBMS", 22000, 1),
    ("Tony", "Stark", 22, "Bangalore", "Karnataka", "560030", "C", 30000, 3);

SELECT First_name, Last_name, Age, Salary FROM Lectures WHERE Age >= 25 AND Salary >=2000(
```

STDIN

Input for the program (Optional)

Output:

First_name	Last_name	Age	Salary
Boo	Girl	26	20000
Trans	Long	27	25000
John	Cena	25	24000
Flaber	Fisk	27	22000

b.) Write a query to print the lecturers between the ages of 25-35.

```
--b.)
SELECT * FROM Lectures WHERE Age BETWEEN 25 AND 35;
```

```sql
SELECT * FROM Lectures WHERE Age BETWEEN 25 AND 35;
```

STDIN

Input for the program (Optional)

Output:

```
First_name      Last_name       Age     City    State
Boo     Girl    26      Mumbai  Maharashtra     410010
Trans   Long    27      Delhi   NULL    101010  DBMS
John    Cena    25      Bangalore       Karnataka
Flaber  Fisk    27      Mumbai  Maharashtra     410030
```

c.) Check the experience of a lecturer, if their experience is greater than 2 years increment their salary by 20%

```sql
-- c.)
UPDATE Lectures SET Salary = Salary * 1.2 WHERE Years_of_experience > 2;
SELECT * FROM Lectures;
```

```sql
UPDATE Lectures SET Salary = Salary * 1.2 WHERE Years_of_experience > 2;
SELECT * FROM Lectures;
```

STDIN

Input for the program (Optional)

Output:

```
First_name      Last_name       Age     City    State
Hello   Boy     19      Bangalore       Karnataka
Boo     Girl    26      Mumbai  Maharashtra     410010
Ram     Di      23      Mangalore       Karnataka
Trans   Long    27      Delhi   NULL    101010  DBMS
John    Cena    25      Bangalore       Karnataka
Robert  Ford    24      Pune    Maharashtra     410020
Augustus        Roman   22      Jaipur  Rajasthan
Archie  Andrews 20      Jaisalmer       Rajasthan
Flaber  Fisk    27      Mumbai  Maharashtra     410030
Tony    Stark   22      Bangalore       Karnataka
```

d.) List the names of the lecturers who are not from Karnataka.

```sql
-- d.)
SELECT State FROM WHERE State !=
"Karnataka"
```

```
SELECT First_name, Last_name, State FROM Lectures WHERE State != "Karnataka";
```

STDIN

Input for the program (Optional)

Output:

```
First_name      Last_name       State
Boo     Girl    Maharashtra
Robert  Ford    Maharashtra
Augustus        Roman   Rajasthan
Archie  Andrews Rajasthan
Flaber  Fisk    Maharashtra
```

e.) Create one more column address and print the address combining city, state and pin code.

```
-- e.)
ALTER TABLE Lectures ADD Address varchar(255);
UPDATE Lectures SET Address = CONCAT(City, ", ", State, ", ", "Pin code: ", Pin_code);
SELECT First_name, Last_name, City, State, Pin_code, Address FROM Lectures;
```

```
ALTER TABLE Lectures ADD Address varchar(255);
UPDATE Lectures SET Address = CONCAT(City, ", ", State, ", ", "Pin code: ", Pin_code);
SELECT First_name, Last_name, City, State, Pin_code, Address FROM Lectures;
```

STDIN

Input for the program (Optional)

Output:

```
First_name      Last_name       City    State   Pin_code
Hello   Boy     Bangalore       Karnataka       560010
Boo     Girl    Mumbai  Maharashtra     410010  Mumbai,
Ram     Di      Mangalore       Karnataka       575025
Trans   Long    Delhi   NULL    101010  NULL
John    Cena    Bangalore       Karnataka       560020
Robert  Ford    Pune    Maharashtra     410020  Pune, Ma
Augustus        Roman   Jaipur  Rajasthan       301010
Archie  Andrews Jaisalmer       Rajasthan       301020
Flaber  Fisk    Mumbai  Maharashtra     410030  Mumbai,
Tony    Stark   Bangalore       Karnataka       560030
```

f.) Find the sum of salaries of all the lecturers in the table and also find out minimum, maximum and average salary.

```
-- f.)
SELECT SUM(Salary), MIN(Salary), MAX(Salary), AVG(Salary) FROM Lectures;
```

```
SELECT SUM(Salary), MIN(Salary), MAX(Salary), AVG(Salary) FROM Lectures;
```

STDIN

Input for the program (Optional)

Output:

```
SUM(Salary)     MIN(Salary)     MAX(Salary)     AVG(Sal
251800  19000   36000   25180.0000
```

g.) Find out the youngest and oldest lecturer in your table.

```
-- f.)
SELECT SUM          MIN          MAX          AVG          FROM
```

```
SELECT * FROM Lectures WHERE Age = (SELECT MAX(Age) FROM Lectures) OR Age = (SELECT MIN(A
```

STDIN

Input for the program (Optional)

Output:

```
First_name      Last_name       Age     City    State
Hello   Boy     19      Bangalore       Karnataka
Trans   Long    27      Delhi   NULL    101010  DBMS
Flaber  Fisk    27      Mumbai  Maharashtra     410030
```

h.) One of the subject “C” was replaced with “python”, write a query to do the same in the table and also print the names of lecturers and their subject after replacement.

```
--h.)
```

```
UPDATE Lectures SET Subject = "python" WHERE Subject = "C";
SELECT First_name, Last_name, Subject FROM Lectures;
SELECT                          Subject FROM
```

```
UPDATE Lectures SET Subject = "python" WHERE Subject = "C";
SELECT First_name, Last_name, Subject FROM Lectures;
```

STDIN

Input for the program (Optional)

Output:

```
First_name      Last_name       Subject
Hello   Boy     python
Boo     Girl    OS
Ram     Di      python
Trans   Long    DBMS
John    Cena    python
Robert  Ford    OS
Augustus        Roman   DBMS
Archie  Andrews OS
Flaber  Fisk    DBMS
Tony    Stark   python
```

i.) Write a query to retrieve the lecturers whose salary is greater than 20000 and name starts with ‘a’.

```
--i.)
SELECT * FROM Lectures WHERE Salary > 20000 AND First_name like "a%";
```

```
SELECT * FROM Lectures WHERE Salary > 20000 AND First_name like "a%";
```

STDIN

Input for the program (Optional)

Output:

```
First_name      Last_name       Age     City    State
Augustus        Roman   22      Jaipur  Rajasthan
```

j.) Write a query to retrieve the lecturers whose experience is above 2 years and first name has ‘s’.

```
--j.)
SELECT * FROM Lectures WHERE Years_of_experience > 2 AND First_name like
"%s%";
```

```
SELECT * FROM Lectures WHERE Years_of_experience > 2 AND First_name like "%s%";
```

STDIN

Input for the program (Optional)

Output:

```
First_name      Last_name       Age     City    State
Trans   Long    27      Delhi   NULL    101010  DBMS
```

Q2. Create a database of movies consisting of Movie id, Movie title, Actor, actress, year, Rating (out of 5), budget, location and Director.

```
CREATE TABLE movies
    (Movie_id int PRIMARY KEY,
    Movie_title varchar(255),
    Actor varchar(255),
    Actress varchar(255),
    Year int,
    Rating decimal(2,1) CHECK(Rating>=0 AND Rating<=5),
    Budget int,
    location varchar(255),
    Director varchar(255));
```

a.) Write a query to print the movies which have the same actress.

```
INSERT INTO movies VALUES
    (1, "The Shawshank Redemption", "Tim Robbins", NULL, 1994, 4.6,
200000, "London", "Frank Darabont"),
    (2, "Its a Wonderful Life", "James Stewart", "Donna Reed", 1946, 4.3,
150000, "London", "Frank Capra"),
```

```
    (3, "The Pianist", "Adrien Brody", "Emilia Fox", 2002, 4.2, 200000,
"New York", "Roman Polanski"),
    (4, "Casino", "Robert De Niro", "Sharon Stone", 1995, 4.1, 90000,
"London", "Martin Scorsese"),
    (5, "Hera Pheri", "Akshay Kumar", "Tabu", 2000, 4, 120000, "India",
"Priyadarshan");
    (6, "Sample", "John", "Donna Reed", 1990, 4.2, 100000, "London",
"Kim");


SELECT * FROM movies WHERE Actress = "Donna Reed";
```

```
SELECT * FROM movies WHERE Actress = "Donna Reed";
```

STDIN

Input for the program (Optional)

Output:

```
Movie_id        Movie_title     Actor   Actress Year
2       Its a Wonderful Life    James Stewart   Donna Re
6       Sample  John    Donna Reed      1990    4.2
```

b.) Write a query to print the movies with a budget greater than 1,00,000 and has an actors name starting with A.

```
SELECT * FROM          WHERE          = "Donna Reed"
```

```
SELECT * FROM movies WHERE Budget > 100000 AND Actor like "A%";
```

STDIN

Input for the program (Optional)

Output:

```
Movie_id        Movie_title     Actor   Actress Year
3       The Pianist     Adrien Brody    Emilia Fox
5       Hera Pheri      Akshay Kumar    Tabu    2000
```

c.) Write a query to filter the movies which were shot in location London and have rating above 4.

```
SELECT * FROM movies WHERE location="London" AND Rating > 4;
```

```
SELECT * FROM movies WHERE location="London" AND Rating > 4;
```

STDIN

Input for the program (Optional)

Output:

```
Movie_id        Movie_title     Actor   Actress Year
1       The Shawshank Redemption        Tim Robbins
2       Its a Wonderful Life    James Stewart   Donna
4       Casino  Robert De Niro  Sharon Stone    1995
6       Sample  John    Donna Reed      1990    4.2
```

d.) Print the average rating of the movies released after 1990 and find the most and least rated movie.

```
SELECT AVG(Rating) FROM movies WHERE Year>1990;
SELECT * FROM movies WHERE Rating = (SELECT MAX(Rating) FROM movies) OR
Rating = (SELECT MIN(Rating) FROM movies);
```

```
SELECT AVG(Rating) FROM movies WHERE Year>1990;
```

STDIN

Input for the program (Optional)

Output:

```
AVG(Rating)
4.22500
```

e.) Update the rating of the movie directed by a particular director with 5 ratings.

```
UPDATE movies SET Rating = 5 WHERE Director = "Priyadarshan";
```

```
UPDATE movies SET Rating = 5 WHERE Director = "Priyadarshan";
SELECT * from movies;
```

STDIN

Input for the program (Optional)

Output:

```
Movie_id        Movie_title     Actor   Actress Year
1       The Shawshank Redemption        Tim Robbins
2       Its a Wonderful Life    James Stewart   Donna
3       The Pianist     Adrien Brody    Emilia Fox
4       Casino  Robert De Niro  Sharon Stone    1995
5       Hera Pheri      Akshay Kumar    Tabu    2000
6       Sample  John    Donna Reed      1990    4.2
```

Q3. Create a student grading database system consisting of:

STUDENT(USN, SName,

Address, Phone, Gender)

IAMARKS(USN, Subcode, Subject name,

Test1, Test2, Test3, FinalIA)

```
CREATE TABLE STUDENT(
    USN int PRIMARY KEY,
    SName varchar(255),
    Address varchar(255),
    Phone varchar(20),
    Gender varchar(10) CHECK (Gender IN ('Male', 'Female', 'Others')));


CREATE TABLE IAMARKS(
    USN int,
    Subcode int,
    `Subject name` varchar(255),
    Test1 int CHECK (Test1 BETWEEN 0 AND 10),
    Test2 int CHECK (Test2 BETWEEN 0 AND 10),
    Test3 int CHECK (Test3 BETWEEN 0 AND 10),
    FinalIA int CHECK (FinalIA BETWEEN 0 AND 30),
```

```
    FOREIGN KEY (USN) REFERENCES STUDENT (USN));

CREATE TRIGGER calc_finalIA BEFORE INSERT ON IAMARKS FOR EACH ROW SET
NEW.FinalIA = NEW.Test1 + NEW.Test2 + NEW.Test3;

CREATE TRIGGER update_finalIA BEFORE UPDATE ON IAMARKS FOR EACH ROW SET
NEW.FinalIA = NEW.Test1 + NEW.Test2 + NEW.Test3;
```

a.) Categorize students based on the following criterion and print thetable by adding a category column in the student table. If FinalIA = 30 to 20 then CA T = ‘Outstanding’ If FinalIA= 20 to 10 then CA T = ‘Average’ If FinalIA< 10 then CA T = ‘Weak’

```
SELECT *,
    CASE
    WHEN FinalIA BETWEEN 20 AND 30 THEN 'Outstanding'
    WHEN FinalIA BETWEEN 10 AND 20 THEN 'Average'
    ELSE 'Weak'
    END
    AS CAT FROM IAMARKS;
```