Name: Mayank

Course: B.Sc. (H) Computer Science

Roll No.: 20221468

Submitted to: Kamlesh Kumar Raghuvanshi

Practical : Database Management System

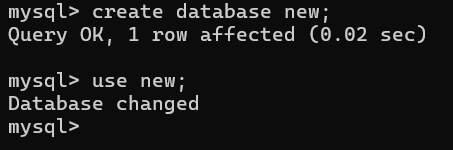
**Creating Database and Tables**

Creating database:

**Query**: create database new;

Use new;

**Output:**



Create table student:

**Query**:

CREATE TABLE STUDENT (

RollNo CHAR(6) PRIMARY KEY,

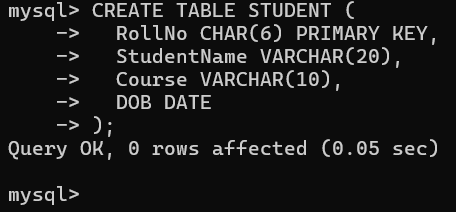
StudentName VARCHAR(20),

Course VARCHAR(10),

DOB DATE

);

**Output:**

****

Create table Society:

**Query**:

CREATE TABLE SOCIETY (

SocID CHAR(6) PRIMARY KEY,

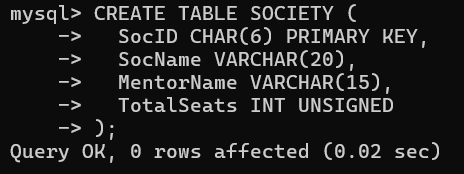
SocName VARCHAR(20),

MentorName VARCHAR(15),

TotalSeats INT UNSIGNED

);

**Output:**

****

Create table Enrollment:

**Query:**

CREATE TABLE ENROLLMENT (

RollNo CHAR(6),

SID CHAR(6),

DateOfEnrollment DATE,

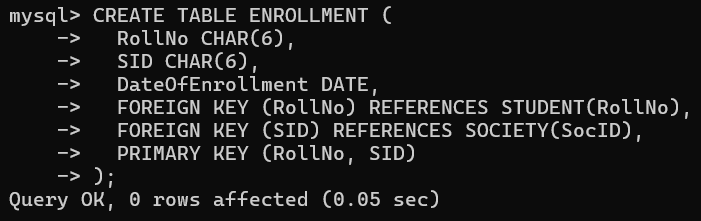
FOREIGN KEY (RollNo) REFERENCES STUDENT(RollNo),

FOREIGN KEY (SID) REFERENCES SOCIETY(SocID),

PRIMARY KEY (RollNo, SID)

);

**Output:**

****

**I. Create and use the following student-society database schema for a college to answer the given (sample) queries using the standalone SQL editor.**

Q1. Retrieve names of students enrolled in any society.

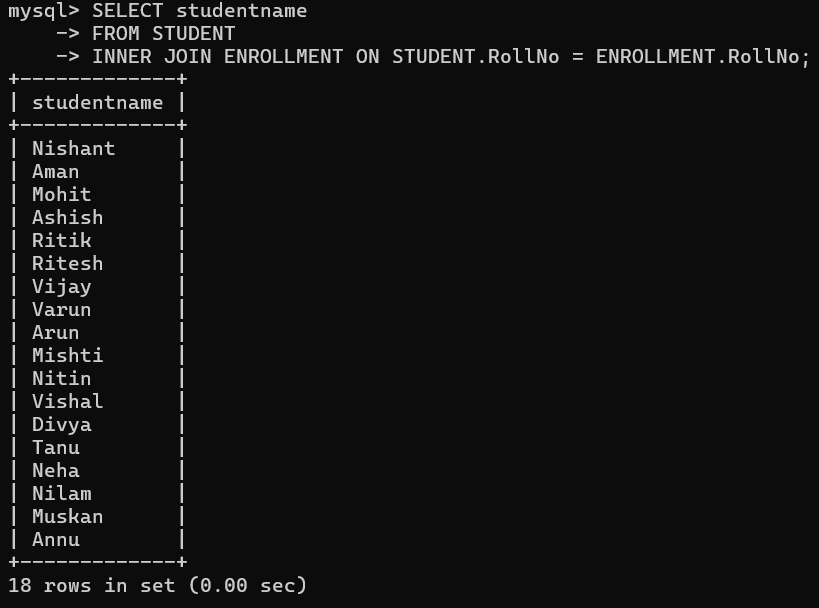
**Query:**

SELECT studentname

FROM STUDENT

INNER JOIN ENROLLMENT ON STUDENT.RollNo = ENROLLMENT.RollNo;

**Output:**

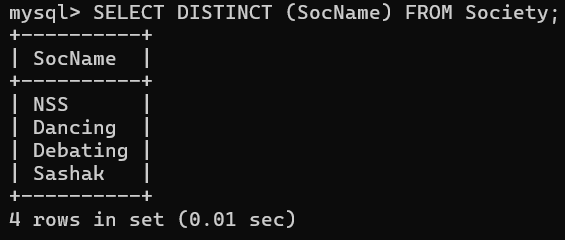
****

Q2. Retrieve all society names.

**Query:**

SELECT DISTINCT (SocName) FROM Society;

**Output:**

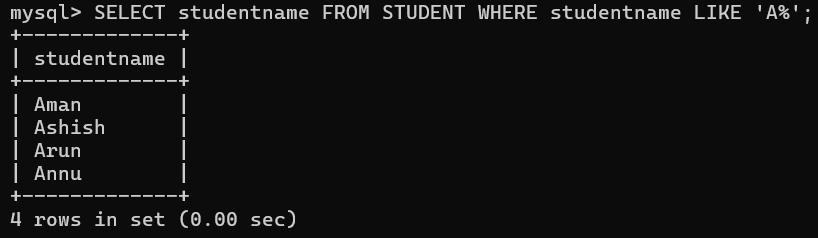
****

Q3. Retrieve students' names starting with letter ‘A’.

**Query:**

SELECT studentname FROM STUDENT WHERE studentname LIKE 'A%';

**Output:**

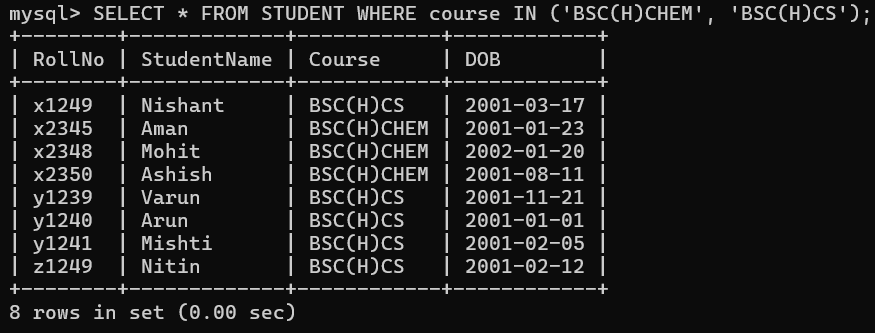


Q4. Retrieve students' details studying in courses ‘computer science’ or ‘chemistry’.

**Query:**

SELECT \* FROM STUDENT WHERE course IN ('BSC(H)CHEM', 'BSC(H)CS');

**Output:**

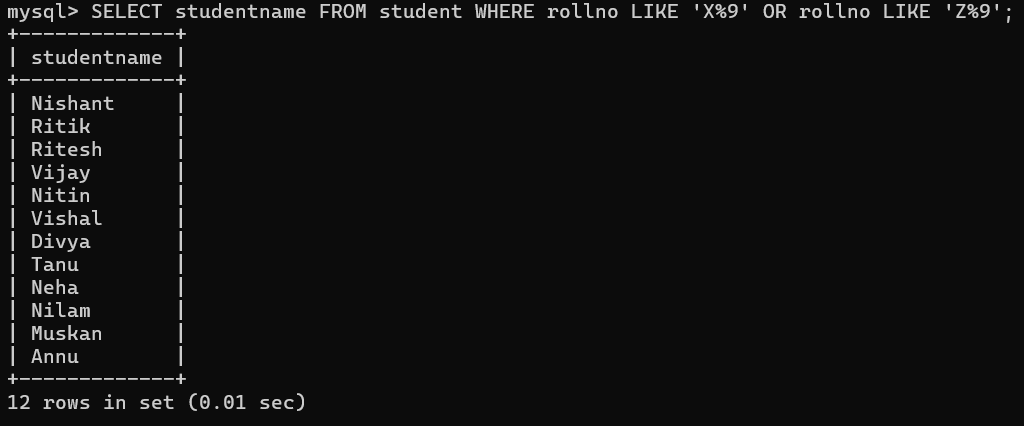
****

Q5. Retrieve students’ names whose roll no either starts with ‘X’ or ‘Z’ and ends with ‘9’

**Query:**

SELECT studentname FROM student WHERE rollno LIKE 'X%9' OR rollno LIKE 'Z%9';

**Output:**

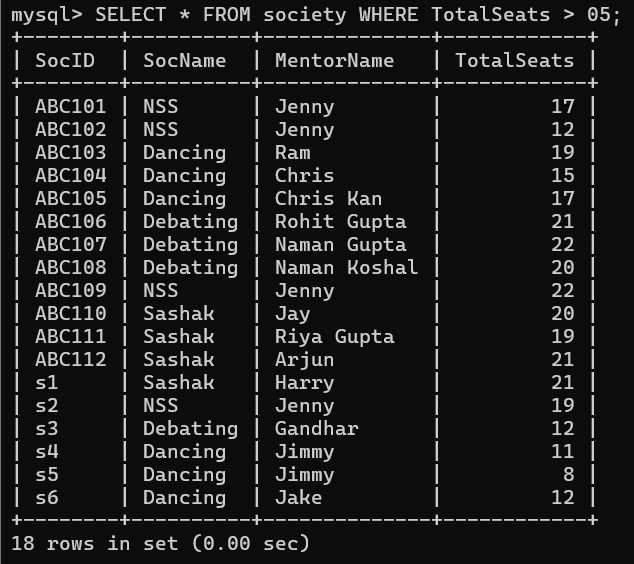
****

Q6. Find society details with more than N TotalSeats where N is to be input by the user

**Query:**

SELECT \* FROM society WHERE TotalSeats > 05;

**Output:**

****

Q7. Update society table for mentor name of a specific society

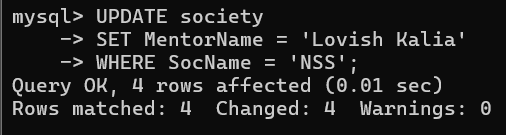
**Query:**

UPDATE society

SET MentorName = 'Lovish Kalia'

WHERE SocName = 'NSS';

**Output:**

****

Q8. Find society names in which more than five students have enrolled.

**Query:**

SELECT SocName

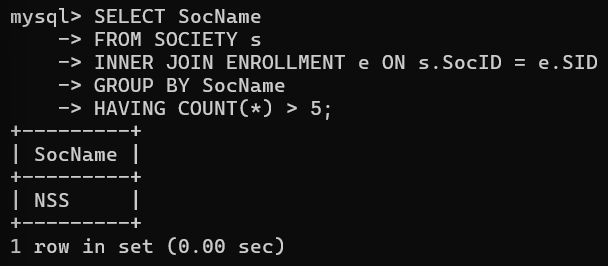
FROM SOCIETY s

INNER JOIN ENROLLMENT e ON s.SocID = e.SID

GROUP BY SocName

HAVING COUNT(\*) > 5;

**Output:**

****

Q9. Find the name of the youngest student enrolled in society 'NSS'.

**Query:**

SELECT StudentName, DOB

FROM STUDENT s

INNER JOIN ENROLLMENT e ON s.RollNo = e.RollNo

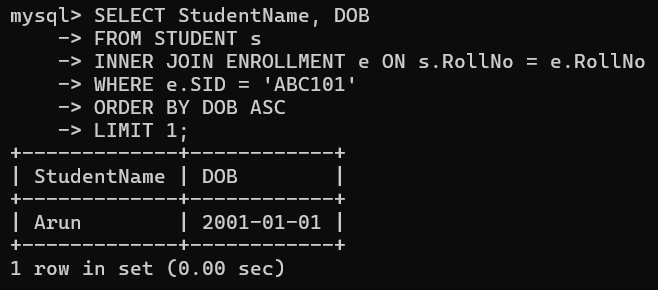
WHERE e.SID = 'ABC101'

ORDER BY DOB DESC

LIMIT 1;

**Output:**

**\*\*Note\*\*:** SID for Society ‘NSS’ is ABC101

****

Q10. Find the name of most popular society (on the basis of enrolled students)

**Query:**

SELECT SocName

FROM SOCIETY s

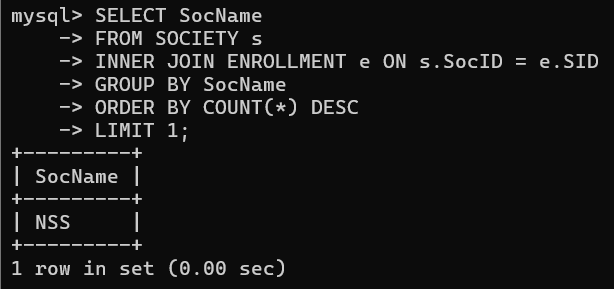
INNER JOIN ENROLLMENT e ON s.SocID = e.SID

GROUP BY SocName

ORDER BY COUNT(\*) DESC

LIMIT 1;

**Output:**

****

Q11. Find the name of two least popular societies (on the basis of enrolled students)

**Query:**

SELECT SocName

FROM SOCIETY s

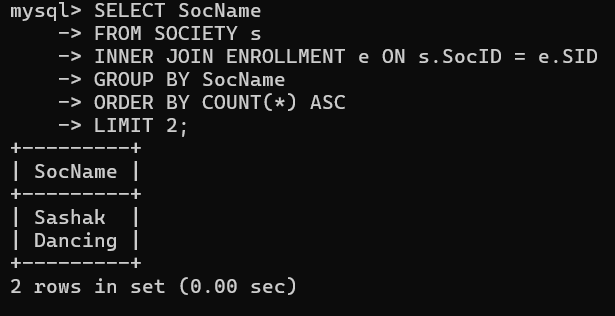
INNER JOIN ENROLLMENT e ON s.SocID = e.SID

GROUP BY SocName

ORDER BY COUNT(\*) ASC

LIMIT 2;

**Output:**

****

Q12. Find the student names who are not enrolled in any society

**Query:**

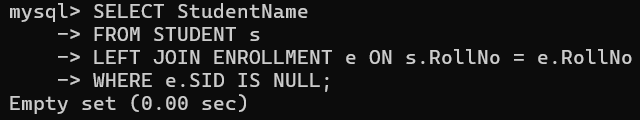
SELECT StudentName

FROM STUDENT s

LEFT JOIN ENROLLMENT e ON s.RollNo = e.RollNo

WHERE e.SID IS NULL;

**Output:**



Q13. Find the student names enrolled in at least two societies

**Query:**

SELECT StudentName

FROM STUDENT s

INNER JOIN ENROLLMENT e1 ON s.RollNo = e1.RollNo

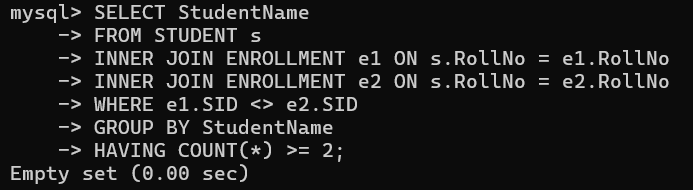
INNER JOIN ENROLLMENT e2 ON s.RollNo = e2.RollNo

WHERE e1.SID <> e2.SID

GROUP BY StudentName

HAVING COUNT(\*) >= 2;

**Output:**

****

Q14. Find society names in which maximum students are enrolled

**Query:**

SELECT SocName, COUNT(\*) AS TotalEnrolledStudents

FROM SOCIETY s

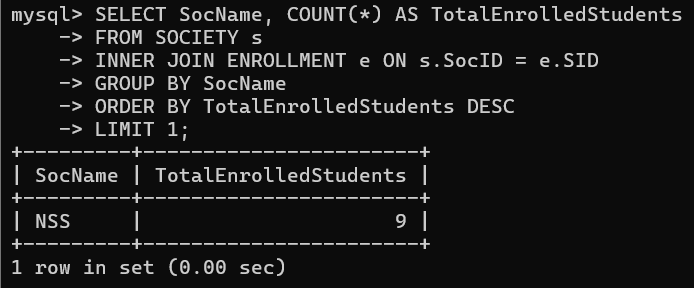
INNER JOIN ENROLLMENT e ON s.SocID = e.SID

GROUP BY SocName

ORDER BY TotalEnrolledStudents DESC

LIMIT 1;

**Output:**



Q15. Find names of all students who have enrolled in any society and society names in which at least one student has enrolled

**Query:**

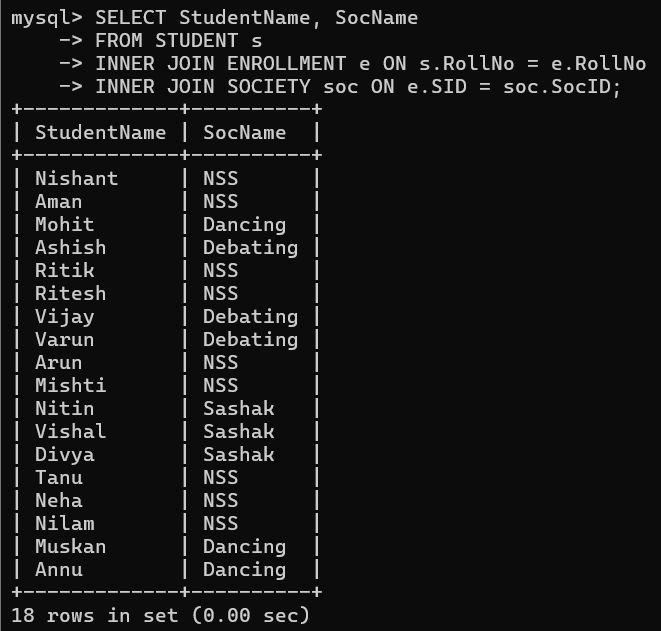
SELECT StudentName, SocName

FROM STUDENT s

INNER JOIN ENROLLMENT e ON s.RollNo = e.RollNo

INNER JOIN SOCIETY soc ON e.SID = soc.SocID;

**Output:**



Q16. Find names of students who are enrolled in any of the three societies ‘Debating’, ‘Dancing’and ‘Sashak’

**Query:**

SELECT StudentName

FROM student s

INNER JOIN ENROLLMENT e ON s.RollNo = e.RollNo

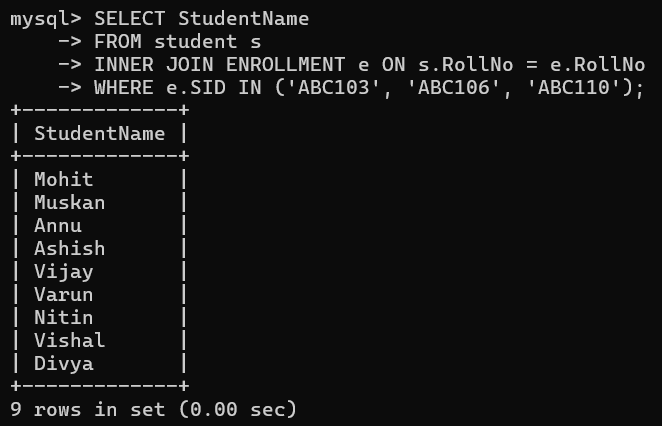
WHERE e.SID IN ('ABC103', 'ABC106', 'ABC110');

**Output:**

**\*\*Note\*\*:** SID for 'Debating' : ABC103,

'Dancing' : ABC106,

'Sashakt' : ABC110



Q17. Find society names such that its mentor has a name with ‘Gupta’ in it.

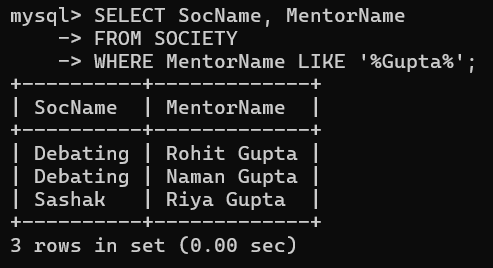
**Query:**

SELECT SocName, MentorName

FROM SOCIETY

WHERE MentorName LIKE '%Gupta%';

**Output:**

****

Q18. Find the society names in which the number of enrolled students is only 10% of its capacity.

**Query:**

SELECT SocName, MAX(TotalSeats) AS TotalSeats, COUNT(\*) AS EnrolledStudents

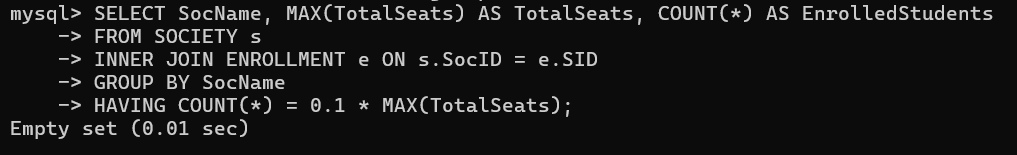
FROM SOCIETY s

INNER JOIN ENROLLMENT e ON s.SocID = e.SID

GROUP BY SocName

HAVING COUNT(\*) = 0.1 \* MAX(TotalSeats);

**Output:**

****

Q19. Display the vacant seats for each society.

**Query:**

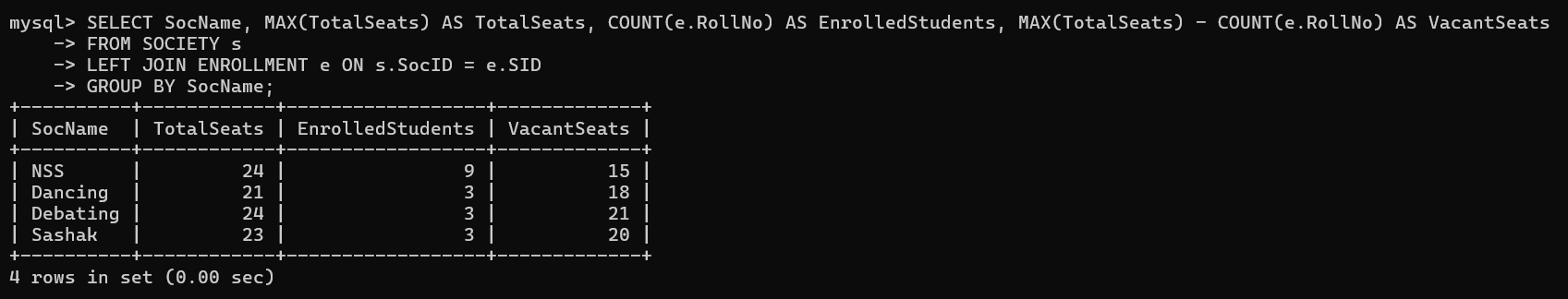
SELECT SocName, MAX(TotalSeats) AS TotalSeats, COUNT(e.RollNo) AS EnrolledStudents, MAX(TotalSeats) - COUNT(e.RollNo) AS VacantSeats

FROM SOCIETY s

LEFT JOIN ENROLLMENT e ON s.SocID = e.SID

GROUP BY SocName;

**Output:**

****

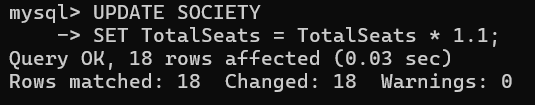
Q20. Increment Total Seats of each society by 10%

**Query:**

UPDATE SOCIETY

SET TotalSeats = TotalSeats \* 1.1;

**Output:**

****

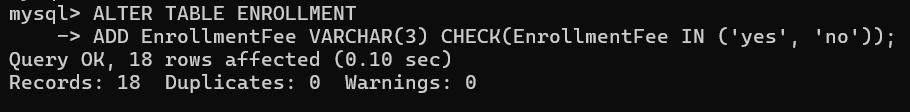
Q21. Add the enrollment fees paid (‘yes’/’No’) field in the enrollment table

**Query:**

ALTER TABLE ENROLLMENT

ADD EnrollmentFee VARCHAR(3) CHECK(EnrollmentFee IN ('yes', 'no'));

**Output:**

****

Q22. Update date of enrollment of society id ‘s1’ to ‘2018-01-15’, ‘s2’ to current date and ‘s3’ to ‘2018-01-02’.

**Query:**

UPDATE ENROLLMENT

SET DateOfEnrollment = '2018-01-15'

WHERE SID = 'sl';

UPDATE ENROLLMENT

SET DateOfEnrollment = CURDATE()

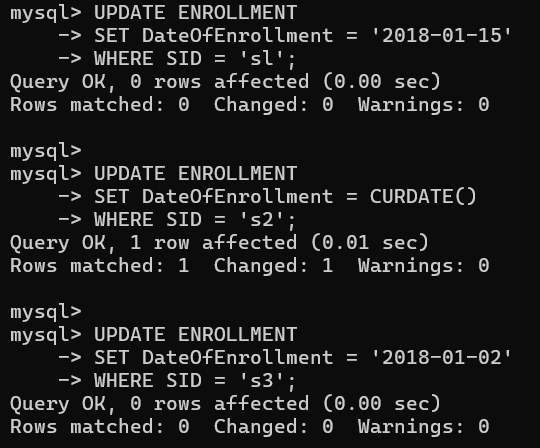
WHERE SID = 's2';

UPDATE ENROLLMENT

SET DateOfEnrollment = '2018-01-02'

WHERE SID = 's3';

**Output:**

****

Q23. Create a view to keep track of society names with the total number of students enrolled in it.

**Query:**

CREATE VIEW SocietyEnrollmentView

AS

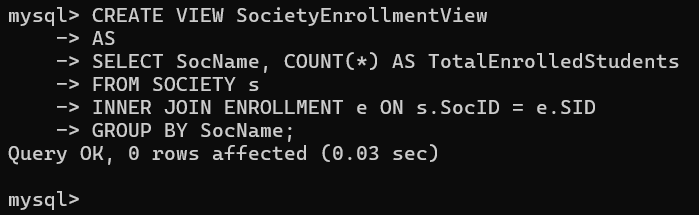
SELECT SocName, COUNT(\*) AS TotalEnrolledStudents

FROM SOCIETY s

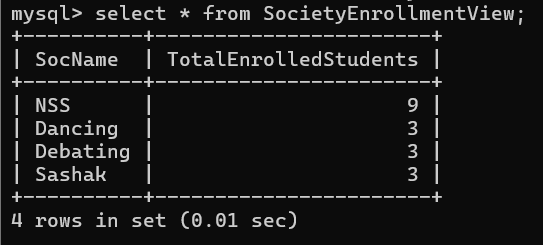
INNER JOIN ENROLLMENT e ON s.SocID = e.SID

GROUP BY SocName;

**Output:**

****

**To View the Table** : select \* from SocietyEnrollmentView;

****

Q24. Find student names enrolled in all the societies.

**Query:**

SELECT StudentName

FROM STUDENT

WHERE RollNo IN (

SELECT RollNo

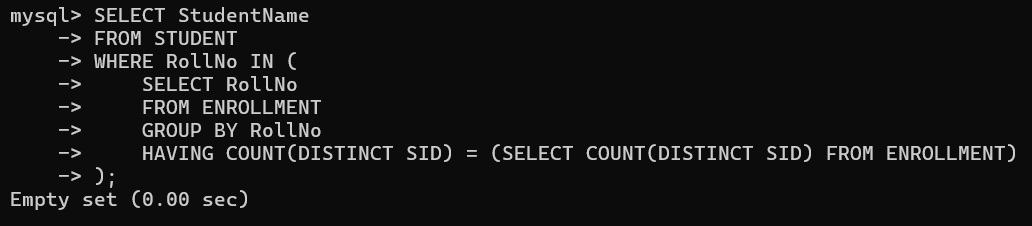
FROM ENROLLMENT

GROUP BY RollNo

HAVING COUNT(DISTINCT SID) = (SELECT COUNT(DISTINCT SID) FROM ENROLLMENT)

);

**Output:**

****

Q25. Count the number of societies with more than 5 students enrolled in it

**Query:**

SELECT COUNT(\*) AS TotalSocieties

FROM (

SELECT s.SocID

FROM SOCIETY s

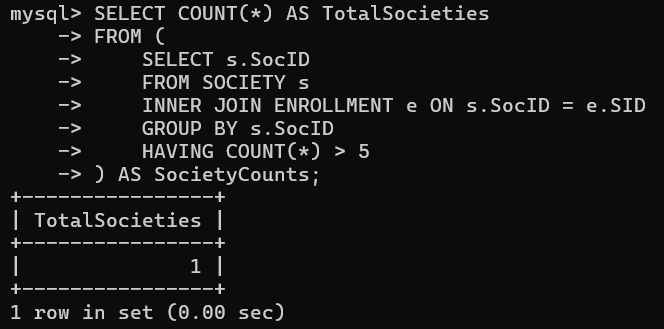
INNER JOIN ENROLLMENT e ON s.SocID = e.SID

GROUP BY s.SocID

HAVING COUNT(\*) > 5

) AS SocietyCounts;

**Output:**

****

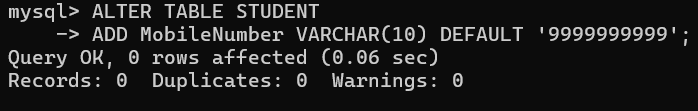
Q26. Add column Mobile number in student table with default value ‘9999999999

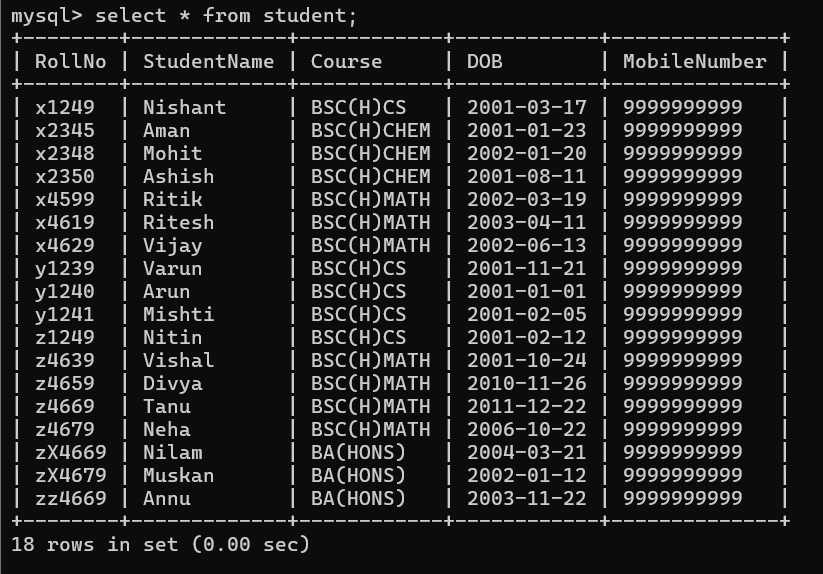
**Query:**

ALTER TABLE STUDENT

ADD MobileNumber VARCHAR(10) DEFAULT '9999999999';

**Output:**

****

****

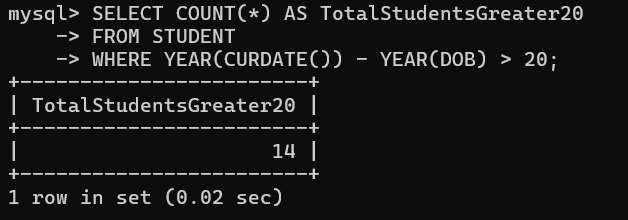
Q27. Find the total number of students whose age is > 20 years.

**Query:**  
SELECT COUNT(\*) AS TotalStudentsGreater20

FROM STUDENT

WHERE YEAR(CURDATE()) - YEAR(DOB) > 20;

**Output:**

****

Q28. Find names of students who are born in 2001 and are enrolled in at least one society

**Query:**

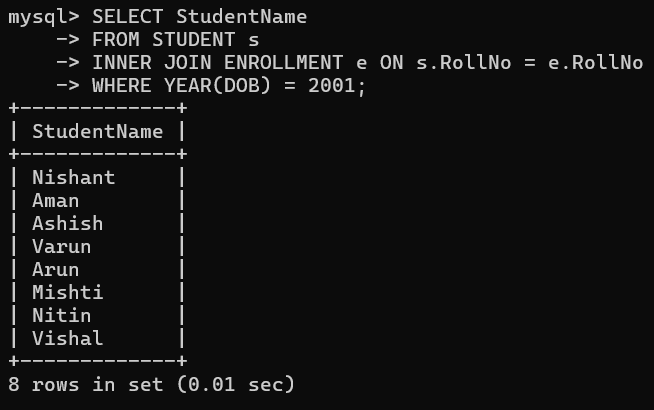
SELECT StudentName

FROM STUDENT s

INNER JOIN ENROLLMENT e ON s.RollNo = e.RollNo

WHERE YEAR(DOB) = 2001;

**Output:**

****

Q29. Count all societies whose name starts with ‘S’ and ends with ‘t’ and at least 5 students are enrolled in the society.

**Query:**

SELECT COUNT(\*) AS TotalMatchingSocieties

FROM SOCIETY s

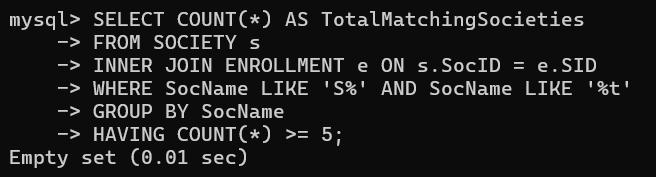
INNER JOIN ENROLLMENT e ON s.SocID = e.SID

WHERE SocName LIKE 'S%' AND SocName LIKE '%t'

GROUP BY SocName

HAVING COUNT(\*) >= 5;

**Output:**

****

Q30. Display the following information:

Society name, Mentor name, Total Capacity, Total Enrolled ,Unfilled Seats

**Query:**

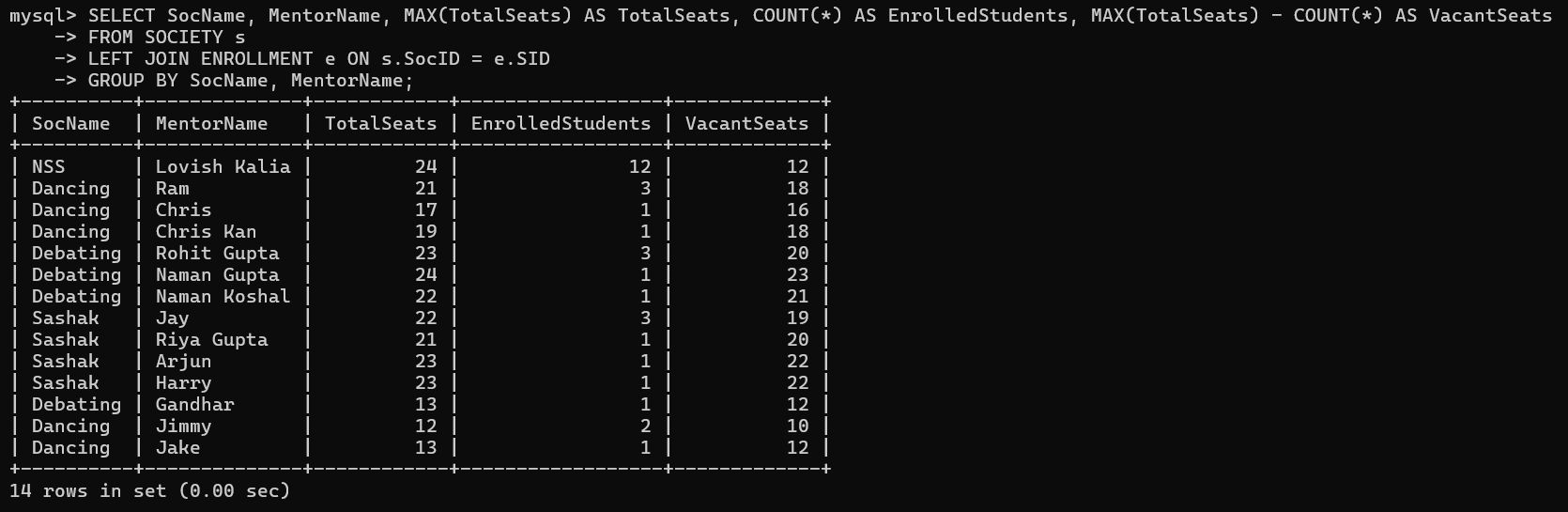
SELECT SocName, MentorName, MAX(TotalSeats) AS TotalSeats, COUNT(\*) AS EnrolledStudents, MAX(TotalSeats) - COUNT(\*) AS VacantSeats

FROM SOCIETY s

LEFT JOIN ENROLLMENT e ON s.SocID = e.SID

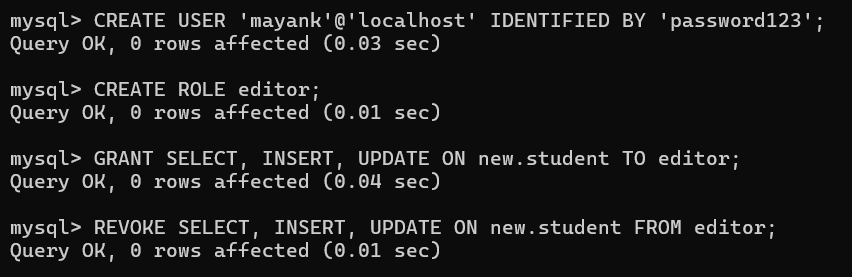
GROUP BY SocName, MentorName;

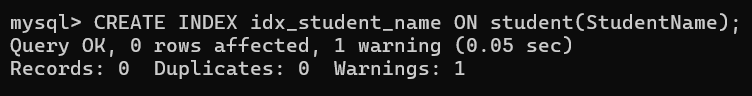
**Output:**

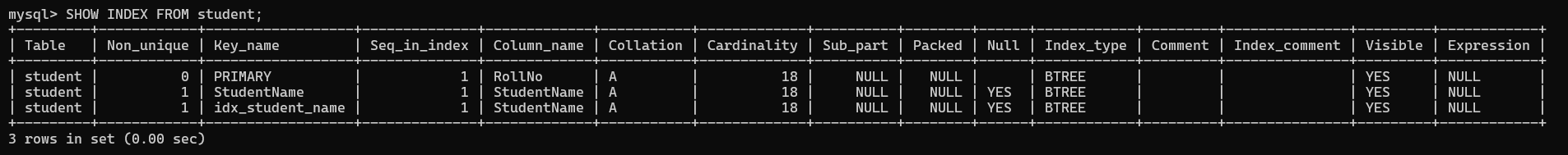
****

**II. Do the following database administration commands:** create user, create role, grant privileges to a role, revoke privileges from a role, create index

**Output**:

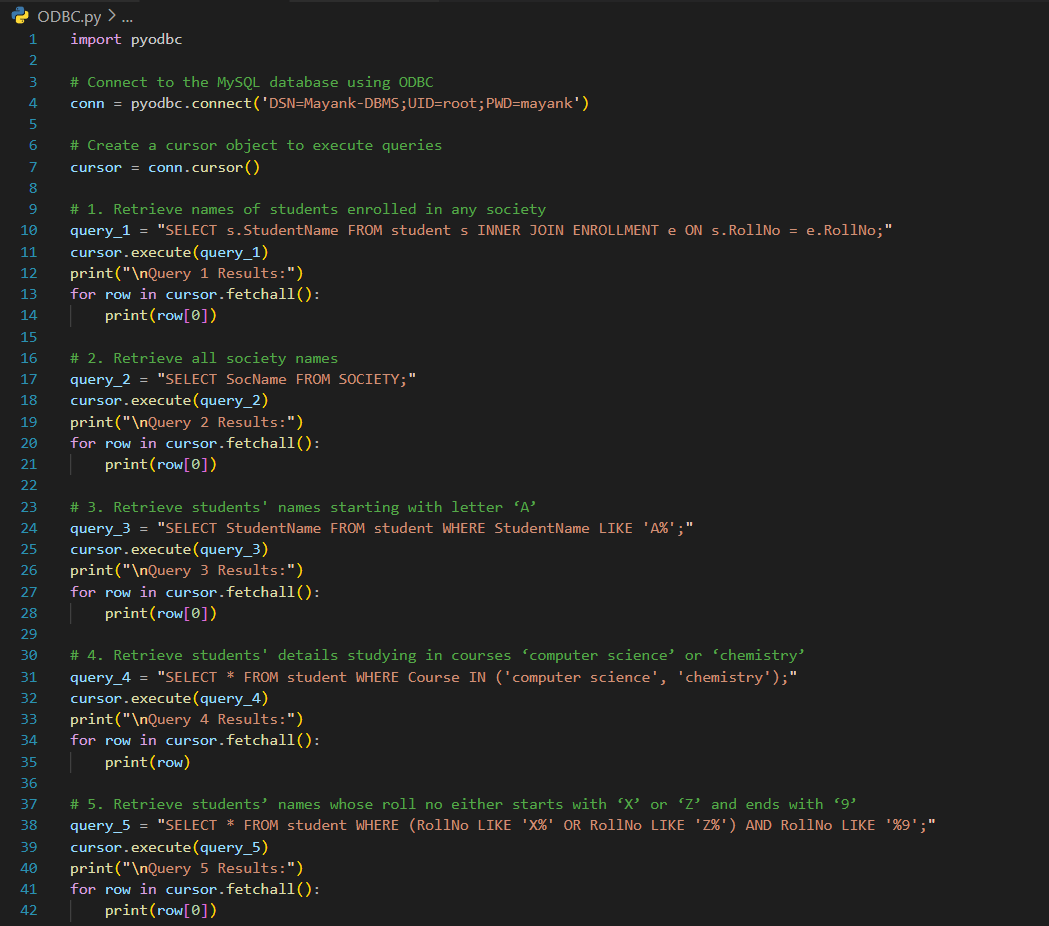


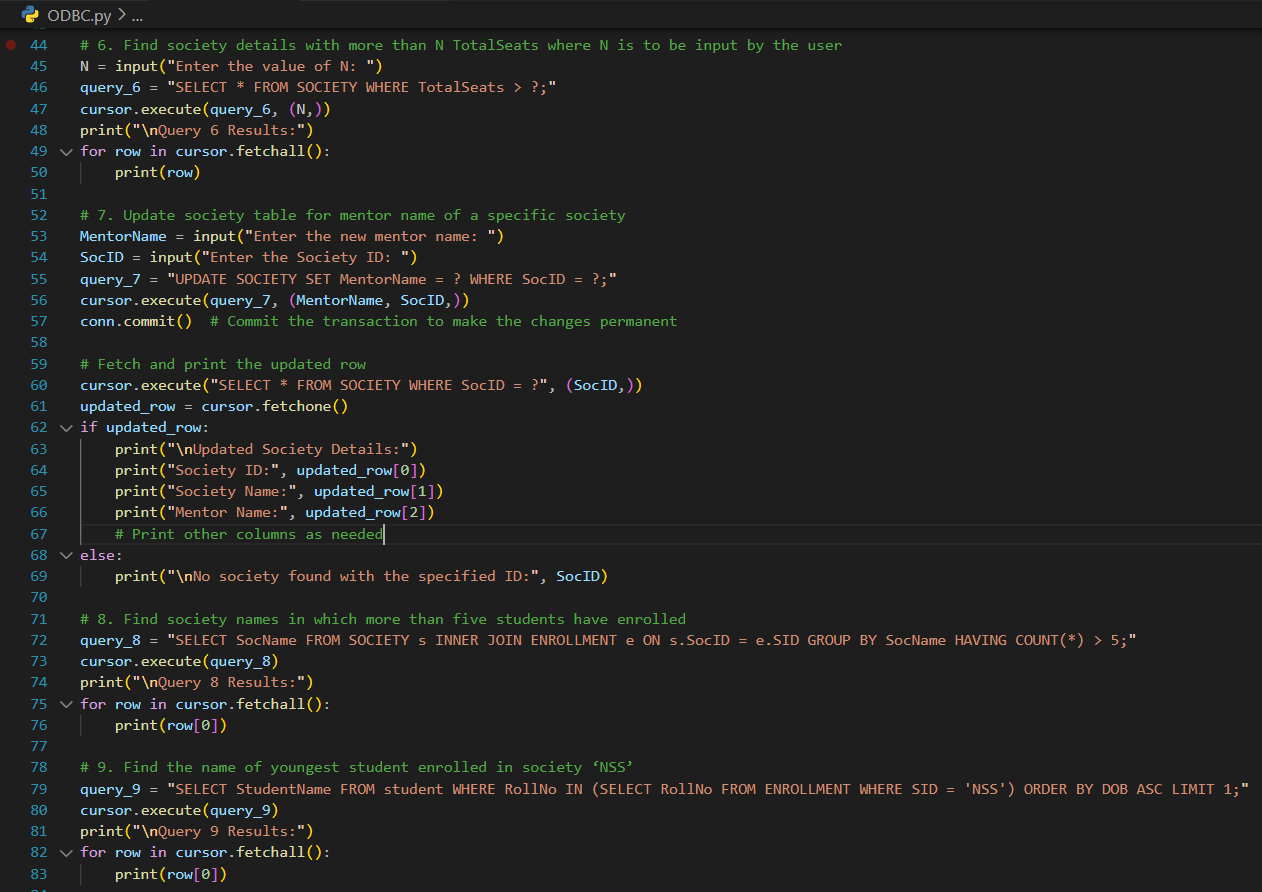


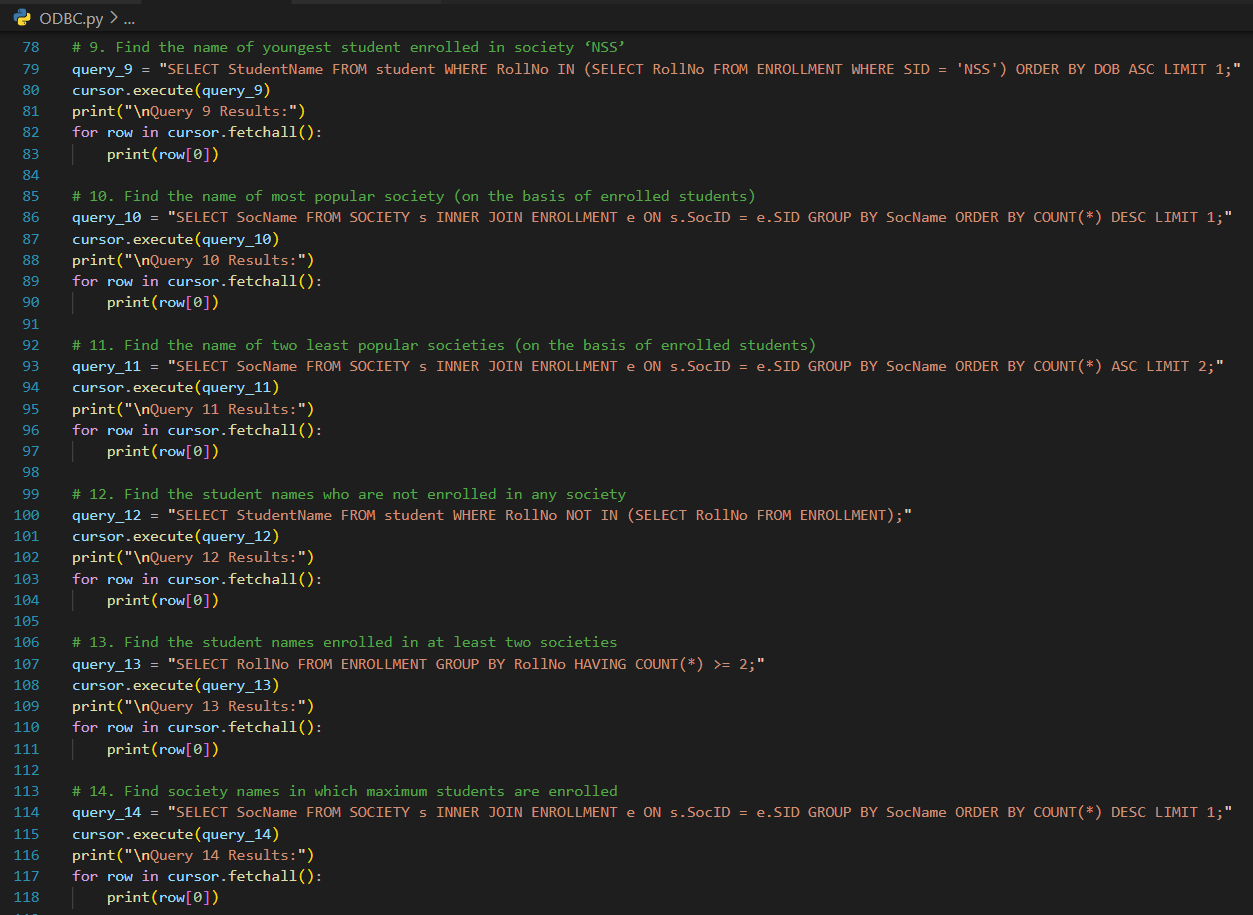


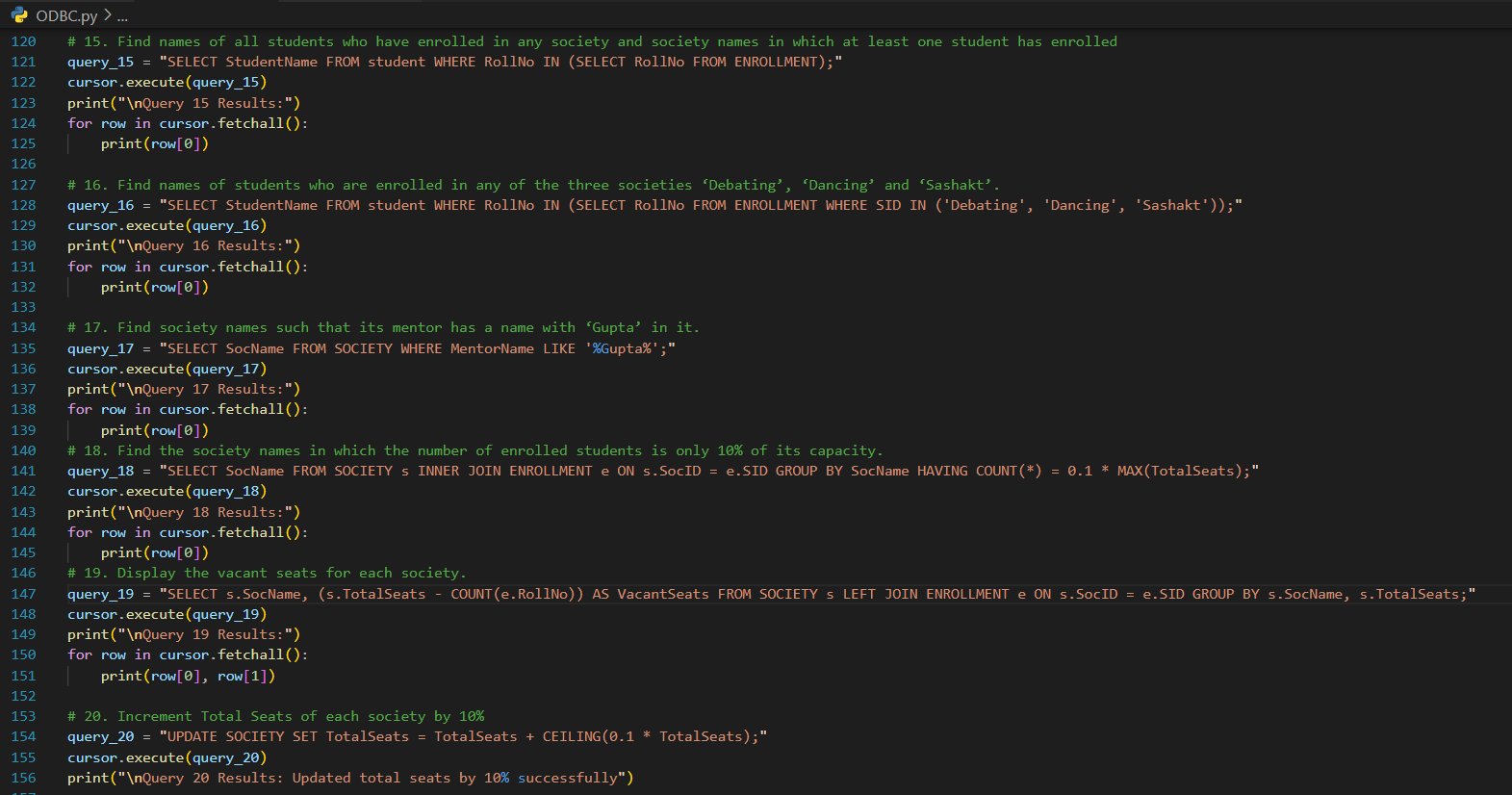
**III.** Execute queries given in part I through a high-level language using ODBC connection.

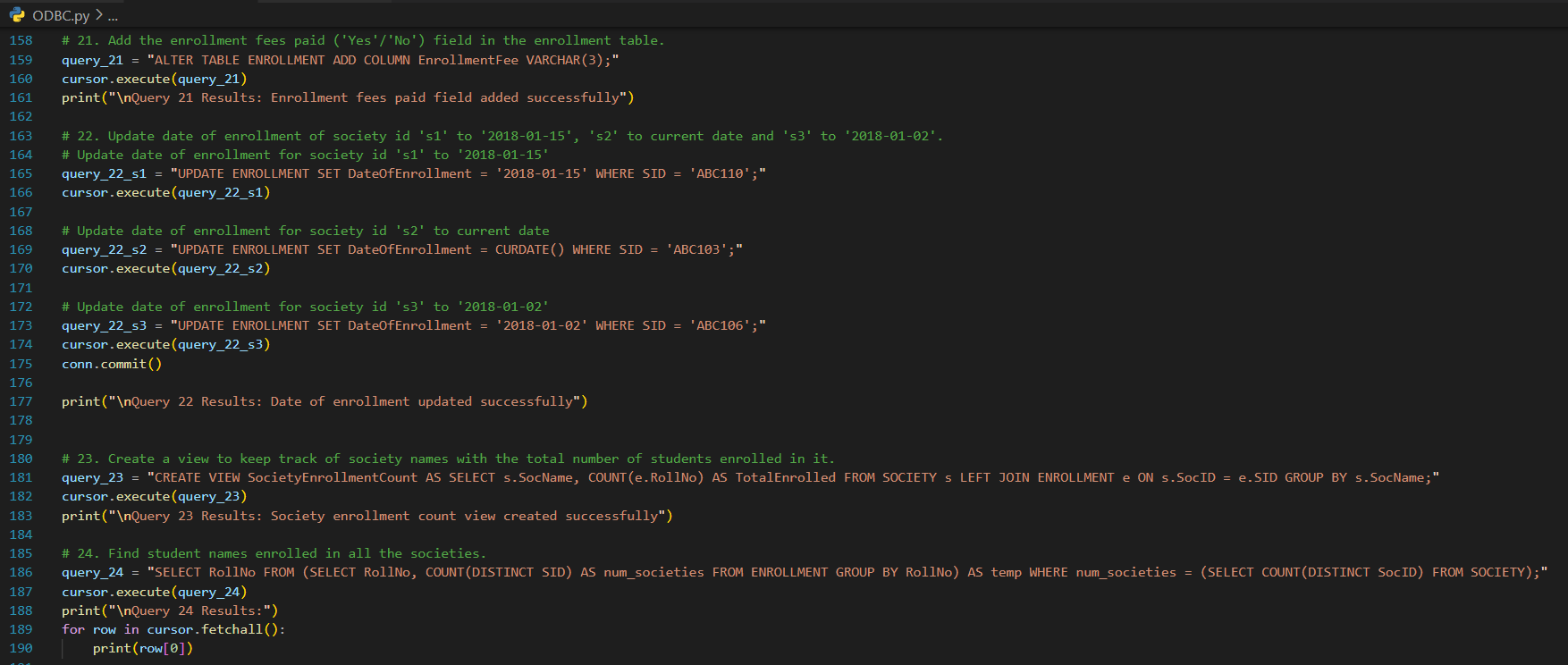
**CODE:**

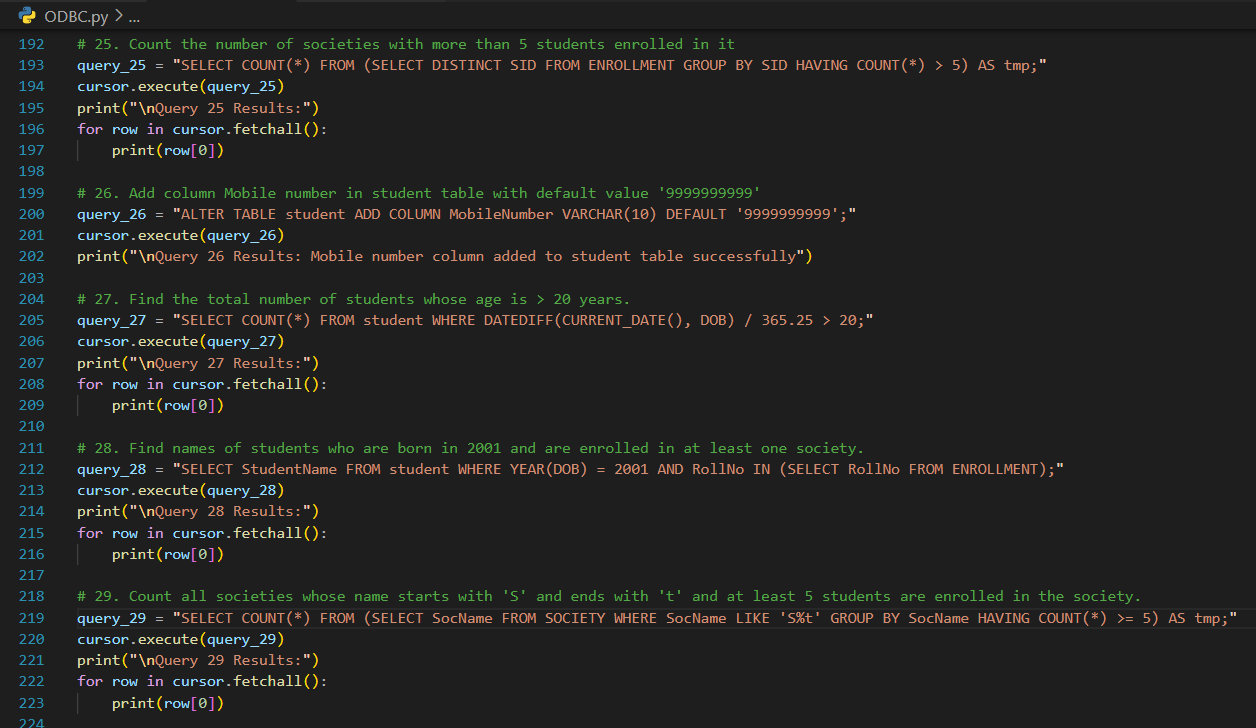
****

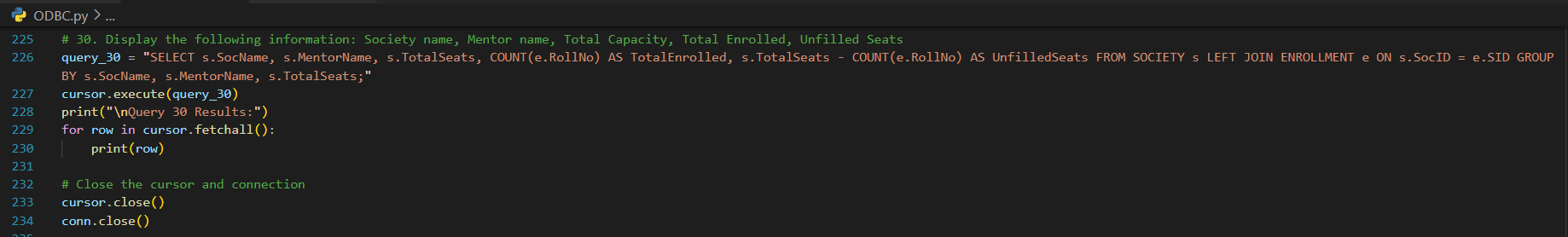
****



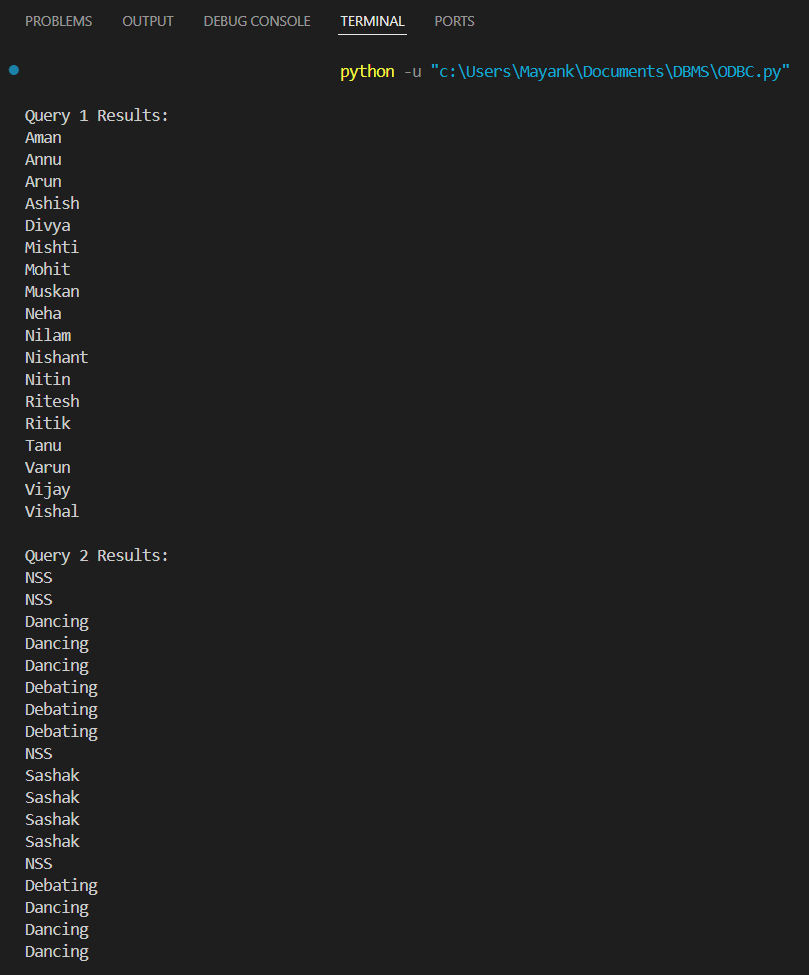


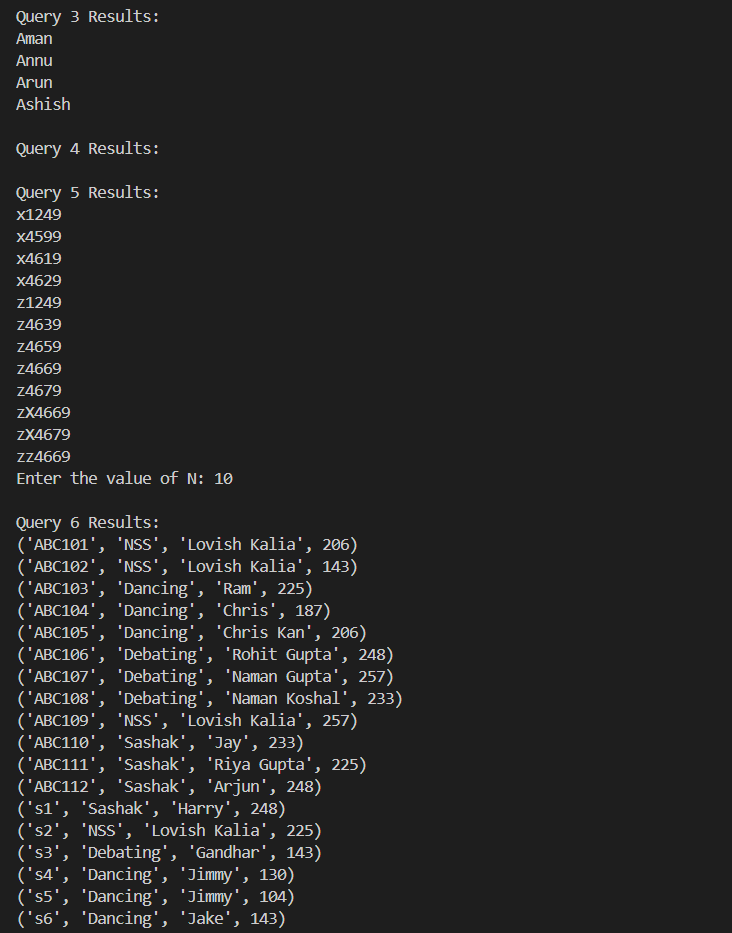


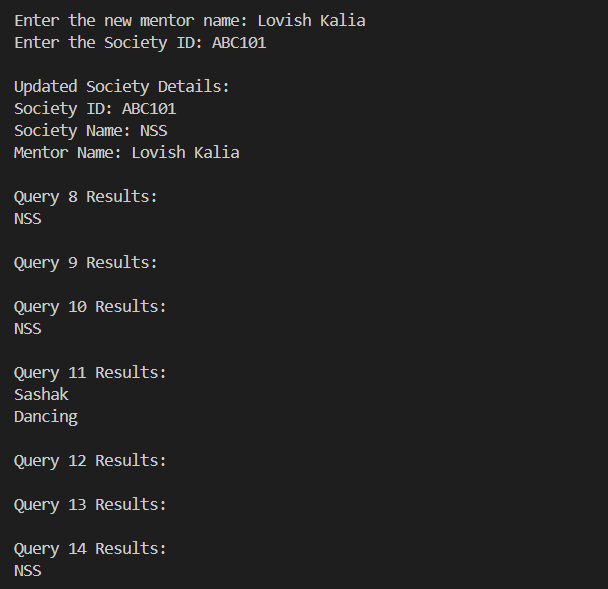




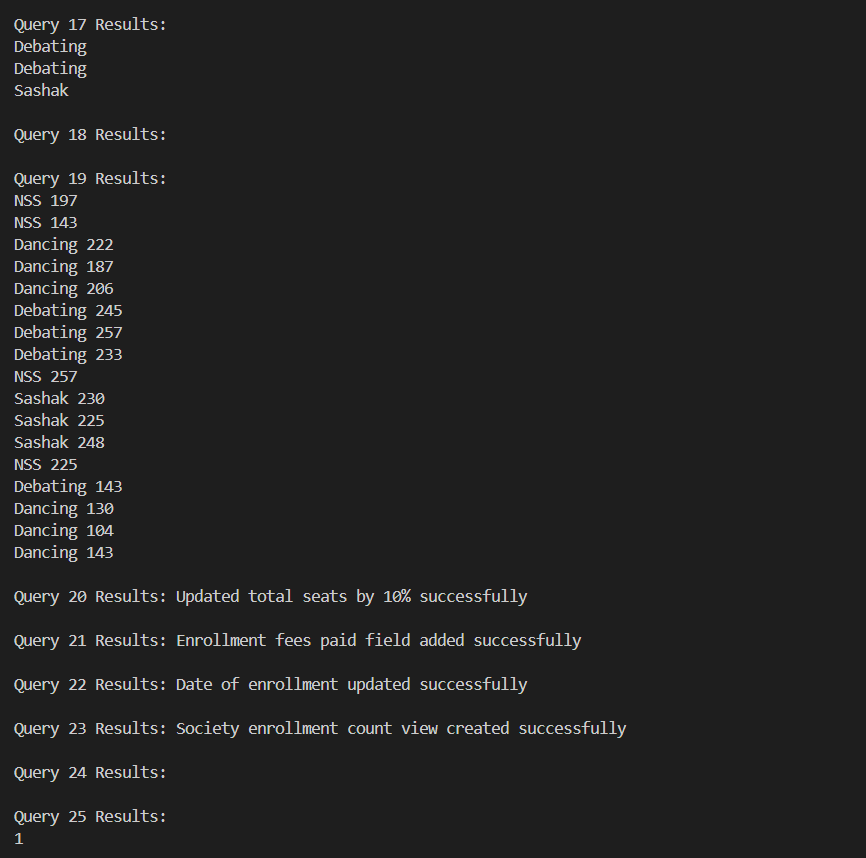
**Output:**

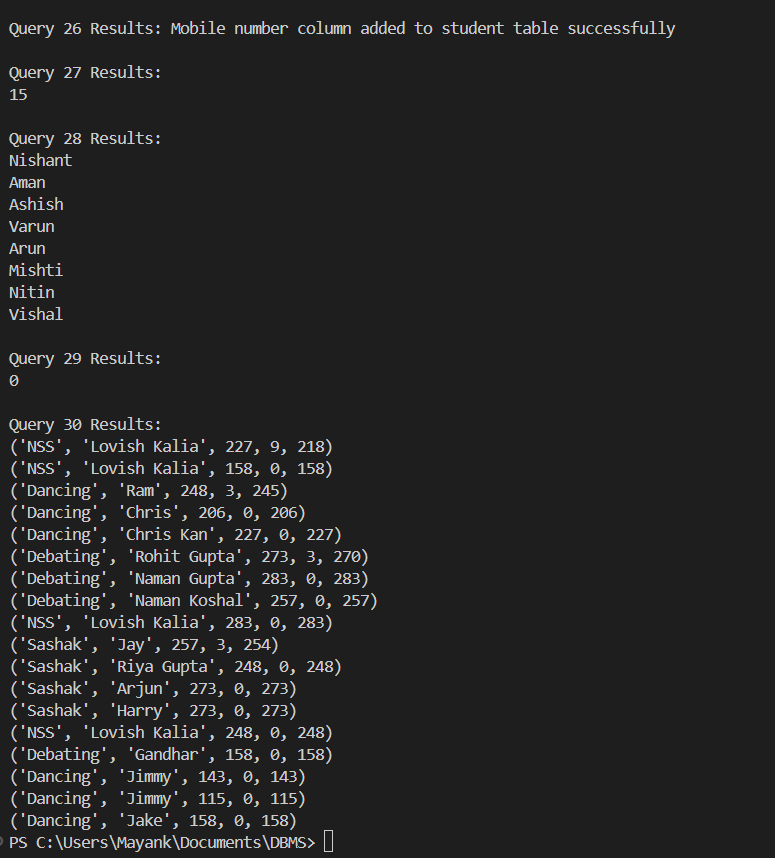
****

****

****

****

****

****