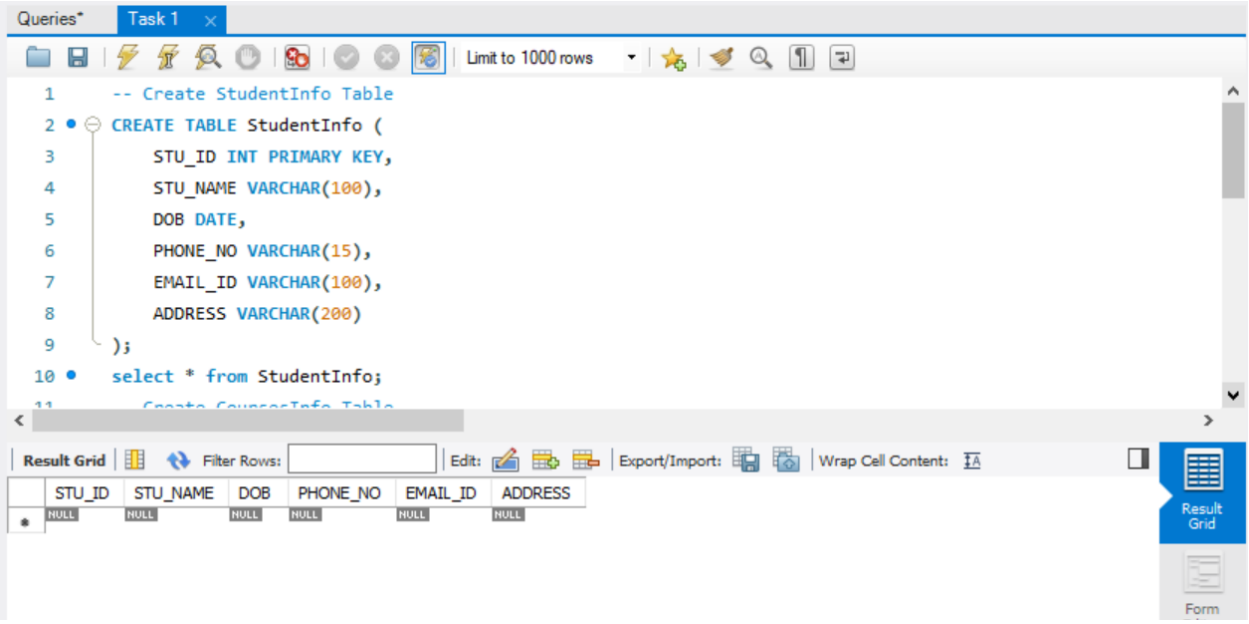
OUTPUT SHEET

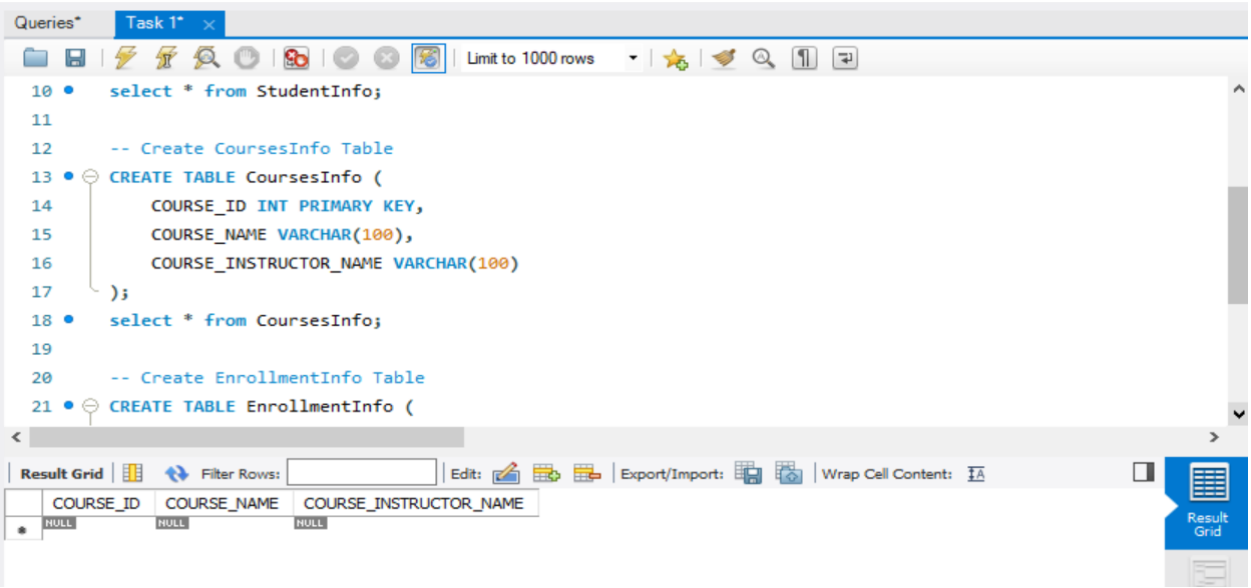
Project Title: Academic Management System (using SQL)

1. Database Creation:

a) Create the StudentInfo table with columns STU_ ID, STU_NAME, DOB, PHONE_NO, EMAIL_ID,ADDRESS.



b) Create the CoursesInfo table with columns COURSE_ID, COURSE_NAME,COURSE_INSTRUCTOR NAME.



c) Create the EnrollmentInfo with columns ENROLLMENT_ID, STU_ ID, COURSE_ID, ENROLL_STATUS(Enrolled/Not Enrolled). The FOREIGN KEY constraint in the EnrollmentInfo table references the STU_ID column in the StudentInfo table and the COURSE_ID column in the CoursesInfo table.

```sql
-- Create EnrollmentInfo Table
CREATE TABLE EnrollmentInfo (
    ENROLLMENT_ID INT PRIMARY KEY,
    STU_ID INT,
    COURSE_ID INT,
    ENROLL_STATUS VARCHAR(20) CHECK (ENROLL_STATUS IN ('Enrolled', 'Not Enrolled')),
    FOREIGN KEY (STU_ID) REFERENCES StudentInfo(STU_ID),
    FOREIGN KEY (COURSE_ID) REFERENCES CoursesInfo(COURSE_ID)
);
select * from EnrollmentInfo;

INSERT INTO StudentInfo (STU_ID, STU_NAME, DOB, PHONE_NO, EMAIL_ID, ADDRESS) VALUES
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

ENROLLMENT_ID	STU_ID	COURSE_ID	ENROLL_STATUS
NULL	NULL	NULL	NULL

2. Data Creation:

Insert some sample data for StudentInfo table , CoursesInfo table, EnrollmentInfo with respective fields.

a)Insert Sample Data into StudentInfo:

```sql
INSERT INTO StudentInfo (STU_ID, STU_NAME, DOB, PHONE_NO, EMAIL_ID, ADDRESS) VALUES
(1, 'Sucharita', '2000-04-15', '9876543210', 'suchargs@gmail.com', '123 Maple Street'),
(2, 'Madhuri', '1999-07-22', '9123456780', 'madhuri@gmail.com', '456 Elm Street'),
(3, 'Rama', '2001-01-10', '9987654321', 'rama@gmail.com', '789 Oak Street');

INSERT INTO CoursesInfo (COURSE_ID, COURSE_NAME, COURSE_INSTRUCTOR_NAME) VALUES
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

STU_ID	STU_NAME	DOB	PHONE_NO	EMAIL_ID	ADDRESS
1	Sucharita	2000-04-15	9876543210	suchargs@gmail.com	123 Maple Street
2	Madhuri	1999-07-22	9123456780	madhuri@gmail.com	456 Elm Street
3	Rama	2001-01-10	9987654321	rama@gmail.com	789 Oak Street

b) **Insert Sample Data into CoursesInfo**:

```sql
INSERT INTO CoursesInfo (COURSE_ID, COURSE_NAME, COURSE_INSTRUCTOR_NAME) VALUES
(101, 'Mathematics', 'Dr. Newton'),
(102, 'Physics', 'Dr. Tesla'),
(103, 'Chemistry', 'Dr. Curie');

INSERT INTO EnrollmentInfo (ENROLLMENT_ID, STU_ID, COURSE_ID, ENROLL_STATUS) VALUES
(1, 1, 101, 'Enrolled'),
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

COURSE_ID	COURSE_NAME	COURSE_INSTRUCTOR_NAME
101	Mathematics	Dr. Newton
102	Physics	Dr. Tesla
103	Chemistry	Dr. Curie

c) **Insert Sample Data into EnrollmentInfo**:

```
INSERT INTO EnrollmentInfo (ENROLLMENT_ID, STU_ID, COURSE_ID, ENROLL_STATUS) VALUES
(1, 1, 101, 'Enrolled'),
(2, 2, 102, 'Enrolled'),
(3, 3, 103, 'Not Enrolled'),
(4, 1, 102, 'Enrolled'),
(5, 2, 103, 'Enrolled');

SELECT si.STU_NAME, si.PHONE_NO, si.EMAIL_ID, ei.ENROLL_STATUS
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

ENROLLMENT_ID	STU_ID	COURSE_ID	ENROLL_STATUS
1	1	101	Enrolled
2	2	102	Enrolled
3	3	103	Not Enrolled
4	1	102	Enrolled
5	2	103	Enrolled

3) Retrieve the Student Information

a) Write a query to retrieve student details, such as student name, contact informations, and Enrollment status.

```
SELECT STU_NAME, PHONE_NO, EMAIL_ID, ENROLL_STATUS
FROM StudentInfo
JOIN EnrollmentInfo ON StudentInfo.STU_ID = EnrollmentInfo.STU_ID;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

STU_NAME	PHONE_NO	EMAIL_ID	ENROLL_STATUS
Sucharita	9876543210	suchargs@gmail.com	Enrolled
Sucharita	9876543210	suchargs@gmail.com	Enrolled
Madhuri	9123456780	madhuri@gmail.com	Enrolled
Madhuri	9123456780	madhuri@gmail.com	Enrolled
Rama	9987654321	rama@gmail.com	Not Enrolled

b) Write a query to retrieve a list of courses in which a specific student is enrolled.

```
SELECT COURSE_NAME
FROM CoursesInfo
JOIN EnrollmentInfo ON CoursesInfo.COURSE_ID = EnrollmentInfo.COURSE_ID
WHERE EnrollmentInfo.STU_ID = 1 AND ENROLL_STATUS = 'Enrolled';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

COURSE_NAME
Mathematics
Physics

c) Write a query to retrieve course information, including course name, instructor information.

```
SELECT COURSE_NAME, COURSE_INSTRUCTOR_NAME
FROM CoursesInfo;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

COURSE_NAME	COURSE_INSTRUCTOR_NAME
Mathematics	Dr. Newton
Physics	Dr. Tesla
Chemistry	Dr. Curie

d) Write a query to retrieve course information for a specific course .

```
SELECT COURSE_NAME, COURSE_INSTRUCTOR_NAME
FROM CoursesInfo
WHERE COURSE_ID = 101;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

COURSE_NAME	COURSE_INSTRUCTOR_NAME
Mathematics	Dr. Newton

e) Write a query to retrieve course information for multiple courses.

```
SELECT COURSE_NAME, COURSE_INSTRUCTOR_NAME
FROM CoursesInfo
WHERE COURSE_ID IN (101, 102);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

COURSE_NAME	COURSE_INSTRUCTOR_NAME
Mathematics	Dr. Newton
Physics	Dr. Tesla

f) Test the queries to ensure accurate retrieval of student information. (execute the queries and verify the results against the expected output.)

The query is executed point wise and the screenshot of the output is presented against each point.

4. Reporting and Analytics (Using joining queries)

a) Write a query to retrieve the number of students enrolled in each course

```sql
SELECT COURSE_NAME, COUNT(EnrollmentInfo.STU_ID) AS Enrolled_Students
FROM CoursesInfo
JOIN EnrollmentInfo ON CoursesInfo.COURSE_ID = EnrollmentInfo.COURSE_ID
WHERE ENROLL_STATUS = 'Enrolled'
GROUP BY COURSE_NAME;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

COURSE_NAME	Enrolled_Students
Mathematics	1
Physics	2
Chemistry	1

b) Write a query to retrieve the list of students enrolled in a specific course

```sql
SELECT STU_NAME
FROM StudentInfo
JOIN EnrollmentInfo ON StudentInfo.STU_ID = EnrollmentInfo.STU_ID
WHERE EnrollmentInfo.COURSE_ID = 102 AND ENROLL_STATUS = 'Enrolled';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

STU_NAME
Madhuri
Sucharita

c) Write a query to retrieve the count of enrolled students for each instructor.

```sql
SELECT COURSE_INSTRUCTOR_NAME, COUNT(EnrollmentInfo.STU_ID) AS Enrolled_Students
FROM CoursesInfo
JOIN EnrollmentInfo ON CoursesInfo.COURSE_ID = EnrollmentInfo.COURSE_ID
WHERE ENROLL_STATUS = 'Enrolled'
GROUP BY COURSE_INSTRUCTOR_NAME;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

COURSE_INSTRUCTOR_NAME	Enrolled_Students
Dr. Newton	1
Dr. Tesla	2
Dr. Curie	1

d) Write a query to retrieve the list of students who are enrolled in multiple courses

```sql
SELECT STU_NAME, COUNT(DISTINCT COURSE_ID) AS Courses_Enrolled
FROM StudentInfo
JOIN EnrollmentInfo ON StudentInfo.STU_ID = EnrollmentInfo.STU_ID
WHERE ENROLL_STATUS = 'Enrolled'
GROUP BY STU_NAME
HAVING COUNT(DISTINCT COURSE_ID) > 1;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

STU_NAME	Courses_Enrolled
Madhuri	2
Sucharita	2

e) Write a query to retrieve the courses that have the highest number of enrolled students(arranging from highest to lowest)

```sql
SELECT COURSE_NAME, COUNT(EnrollmentInfo.STU_ID) AS Enrolled_Students
FROM CoursesInfo
JOIN EnrollmentInfo ON CoursesInfo.COURSE_ID = EnrollmentInfo.COURSE_ID
WHERE ENROLL_STATUS = 'Enrolled'
GROUP BY COURSE_NAME
ORDER BY Enrolled_Students DESC;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

COURSE_NAME	Enrolled_Students
Physics	2
Mathematics	1
Chemistry	1

Query Explanation

1. Database Creation

a) Create StudentInfo Table

```
CREATE TABLE StudentInfo (
   STU_ID INT PRIMARY KEY,
   STU_NAME VARCHAR(100),
   DOB DATE,
   PHONE_NO VARCHAR(15),
   EMAIL_ID VARCHAR(100),
   ADDRESS VARCHAR(200)
);
```

Explanation: -

- A unique identifier for each student, set as the primary key.
- A column to store the student's name, with a maximum length of 100 characters.
- A column to store the student's date of birth.
- A column to store the student's phone number, with a maximum length of 15 characters.
- A column to store the student's email address.
- A column to store the student's address, with a maximum length of 200 characters.

b) Create CoursesInfo Table

```
CREATE TABLE CoursesInfo (
   COURSE_ID INT PRIMARY KEY,
   COURSE_NAME VARCHAR(100),
   COURSE_INSTRUCTOR_NAME VARCHAR(100)
);
```

Explanation: -

- A unique identifier for each course, set as the primary key.
- A column to store the course name, with a maximum length of 100 characters.
- A column to store the course instructor's name, with a maximum length of 100 characters.

c) Create EnrollmentInfo Table

```
CREATE TABLE EnrollmentInfo (
   ENROLLMENT_ID INT PRIMARY KEY,
   STU_ID INT,
   COURSE_ID INT,
```

```
    ENROLL_STATUS VARCHAR(20) CHECK (ENROLL_STATUS IN ('Enrolled', 'Not
Enrolled')),
    FOREIGN KEY (STU_ID) REFERENCES StudentInfo(STU_ID),
    FOREIGN KEY (COURSE_ID) REFERENCES CoursesInfo(COURSE_ID)
);
```

Explanation: -

- A unique identifier for each enrollment record, set as the primary key.
- References the `STU_ID` in the `StudentInfo` table (foreign key).
- References the `COURSE_ID` in the `CoursesInfo` table (foreign key).
- A column to store the enrollment status, restricted to 'Enrolled' or 'Not Enrolled'.
- Links to the primary key in the `StudentInfo` table.
- Links to the primary key in the `CoursesInfo` table.

2. Data Creation

a) Insert Sample Data into StudentInfo

```
INSERT INTO StudentInfo (STU_ID, STU_NAME, DOB, PHONE_NO, EMAIL_ID,
ADDRESS) VALUES
(1, 'Sucharita', '2000-04-15', '9876543210', 'suchargs@gmail.com', '123 Maple Street'),
(2, 'Madhuri', '1999-07-22', '9123456780', 'madhuri@gmail.com', '456 Elm Street'),
(3, 'Rama', '2001-01-10', '9987654321', 'rama@gmail.com', '789 Oak Street');
```

Explanation:-

- Inserts a student named 'Sucharita' with her details.
- Inserts a student named 'Madhuri' with his details.
- Inserts a student named 'Rama' with his details.

b) Insert Sample Data into CoursesInfo

```
INSERT INTO CoursesInfo (COURSE_ID, COURSE_NAME,
COURSE_INSTRUCTOR_NAME) VALUES
(101, 'Mathematics', 'Dr. Newton'),
(102, 'Physics', 'Dr. Tesla'),
(103, 'Chemistry', 'Dr. Curie');
```

Explanation:-

- Inserts a course named Mathematics taught by Dr. Newton.
- Inserts a course named Physics taught by Dr. Tesla.
- Inserts a course named Chemistry taught by Dr. Curie.

c) Insert Sample Data into EnrollmentInfo

```
INSERT INTO EnrollmentInfo (ENROLLMENT_ID, STU_ID, COURSE_ID,
ENROLL_STATUS) VALUES
(1, 1, 101, 'Enrolled'),
(2, 2, 102, 'Enrolled'),
(3, 3, 103, 'Not Enrolled'),
(4, 1, 102, 'Enrolled'),
(5, 2, 103, 'Enrolled');
```

Explanation:-

- Records Sucharita's enrollment in Mathematics as 'Enrolled'.
- Records Madhuri's enrollment in Physics as 'Enrolled'.
- Records Rama's status in Chemistry as 'Not Enrolled'.
- Records Sucharita's enrollment in Physics as 'Enrolled'.
- Records Madhuri's enrollment in Chemistry as 'Enrolled'.

3. Retrieve Student Information

a) Retrieve Student Details

```
SELECT STU_NAME, PHONE_NO, EMAIL_ID, ENROLL_STATUS
FROM StudentInfo
JOIN EnrollmentInfo ON StudentInfo.STU_ID = EnrollmentInfo.STU_ID;
```

Explanation:-

- Fetches student details (STU_NAME, PHONE_NO, EMAIL_ID) and their ENROLL_STATUS.
- Joins StudentInfo with EnrollmentInfo using STU_ID.

b) Retrieve Courses for a Specific Student

```
SELECT COURSE_NAME
FROM CoursesInfo
JOIN EnrollmentInfo ON CoursesInfo.COURSE_ID = EnrollmentInfo.COURSE_ID
WHERE EnrollmentInfo.STU_ID = 1 AND ENROLL_STATUS = 'Enrolled';
```

Explanation:-

- Fetches the COURSE_NAME for the student with STU_ID = 1 (Sucharita).
- Only includes courses where the ENROLL_STATUS is 'Enrolled'.

c) Retrieve All Course Information

```
SELECT COURSE_NAME, COURSE_INSTRUCTOR_NAME
```

```
FROM CoursesInfo;
```

Explanation:-

- Fetches all courses (COURSE_NAME) and their instructors (COURSE_INSTRUCTOR_NAME).

d) Retrieve Information for a Specific Course

```
SELECT COURSE_NAME, COURSE_INSTRUCTOR_NAME
FROM CoursesInfo
WHERE COURSE_ID = 101;
```

Explanation:-

- Fetches the name and instructor of the course with COURSE_ID = 101 (Mathematics).

e) Retrieve Information for Multiple Courses

```
SELECT COURSE_NAME, COURSE_INSTRUCTOR_NAME
FROM CoursesInfo
WHERE COURSE_ID IN (101, 102);
```

Explanation:-

- Fetches the name and instructor for courses with COURSE_ID 101 (Mathematics) and 102 (Physics).

4. Reporting and Analytics

a) Number of Students Enrolled in Each Course

```
SELECT COURSE_NAME, COUNT(EnrollmentInfo.STU_ID) AS Enrolled_Students
FROM CoursesInfo
JOIN EnrollmentInfo ON CoursesInfo.COURSE_ID = EnrollmentInfo.COURSE_ID WHERE
ENROLL_STATUS = 'Enrolled'
GROUP BY COURSE_NAME;
```

Explanation:-

- SELECT COURSE_NAME, COUNT(EnrollmentInfo.STU_ID) AS Enrolled_Students
  Selects the course name (COURSE_NAME) and counts the number of students (STU_ID) enrolled in each course. The result is labeled as Enrolled_Students.
- FROM CoursesInfo
  Specifies the CoursesInfo table as the base table for this query.
- JOIN EnrollmentInfo ON CoursesInfo.COURSE_ID = EnrollmentInfo.COURSE_ID

Joins CoursesInfo with EnrollmentInfo using the COURSE_ID column. This ensures the data is combined for courses and their enrollment records.

- WHERE ENROLL_STATUS = 'Enrolled'
  Filters the rows to include only those where the enrollment status is 'Enrolled'.
- GROUP BY COURSE_NAME
  Groups the result by COURSE_NAME, so the COUNT function is calculated for each course separately.

b) List of Students Enrolled in a Specific Course

Explanation:-

```
SELECT STU_NAME
FROM StudentInfo
JOIN EnrollmentInfo ON StudentInfo.STU_ID = EnrollmentInfo.STU_ID
WHERE EnrollmentInfo.COURSE_ID = 102 AND ENROLL_STATUS = 'Enrolled';
```

- SELECT STU_NAME
  Selects the names of students (STU_NAME) enrolled in the specified course.
- FROM StudentInfo
  Specifies the StudentInfo table as the base table for the query.
- JOIN EnrollmentInfo ON StudentInfo.STU_ID = EnrollmentInfo.STU_ID
  Joins StudentInfo with EnrollmentInfo using the STU_ID column to associate students with their enrollments.
- WHERE EnrollmentInfo.COURSE_ID = 102 AND ENROLL_STATUS = 'Enrolled'
  Filters the rows to include only those where the course ID is 102 and the enrollment status is 'Enrolled'.

c) Count of Students for Each Instructor

```
SELECT COURSE_INSTRUCTOR_NAME, COUNT(EnrollmentInfo.STU_ID) AS
Enrolled_Students
FROM CoursesInfo
JOIN EnrollmentInfo ON CoursesInfo.COURSE_ID = EnrollmentInfo.COURSE_ID
WHERE ENROLL_STATUS = 'Enrolled'
GROUP BY COURSE_INSTRUCTOR_NAME;
```

Explanation:-

- SELECT COURSE_INSTRUCTOR_NAME, COUNT(EnrollmentInfo.STU_ID) AS Enrolled_Students
  Selects the course instructor's name (COURSE_INSTRUCTOR_NAME) and counts the students (STU_ID) enrolled in their courses, labeling it as Enrolled_Students.

- FROM CoursesInfo
  Specifies the CoursesInfo table as the base table.
- JOIN EnrollmentInfo ON CoursesInfo.COURSE_ID = EnrollmentInfo.COURSE_ID
  Joins CoursesInfo with EnrollmentInfo using COURSE_ID.
- WHERE ENROLL_STATUS = 'Enrolled'
  Filters the rows to include only those with the enrollment status 'Enrolled'.
- GROUP BY COURSE_INSTRUCTOR_NAME
  Groups the result by instructor name, so the COUNT function aggregates students for each instructor.

d) Students Enrolled in Multiple Courses

```
SELECT STU_NAME, COUNT(DISTINCT COURSE_ID) AS Courses_Enrolled
FROM StudentInfo
JOIN EnrollmentInfo ON StudentInfo.STU_ID = EnrollmentInfo.STU_ID
WHERE ENROLL_STATUS = 'Enrolled' GROUP BY STU_NAME
HAVING COUNT(DISTINCT COURSE_ID) > 1;
```

Explanation:-

- SELECT STU_NAME, COUNT(DISTINCT COURSE_ID) AS Courses_Enrolled
  Selects the student name (STU_NAME) and counts the number of distinct courses (COURSE_ID) they are enrolled in. The result is labeled as Courses_Enrolled.
- FROM StudentInfo
  Specifies the StudentInfo table as the base table.
- JOIN EnrollmentInfo ON StudentInfo.STU_ID = EnrollmentInfo.STU_ID
  Joins StudentInfo with EnrollmentInfo to link students with their enrollments.
- WHERE ENROLL_STATUS = 'Enrolled'
  Filters the rows to include only those where the enrollment status is 'Enrolled'.
- GROUP BY STU_NAME
  Groups the result by student name, so the COUNT function calculates for each student.
- HAVING COUNT(DISTINCT COURSE_ID) > 1
  Filters the grouped results to include only students enrolled in more than one course.

e) Courses with the Highest Number of Students

```
SELECT COURSE_NAME, COUNT(EnrollmentInfo.STU_ID) AS Enrolled_Students
FROM CoursesInfo
JOIN EnrollmentInfo ON CoursesInfo.COURSE_ID = EnrollmentInfo.COURSE_ID
WHERE ENROLL_STATUS = 'Enrolled'
GROUP BY COURSE_NAME
ORDER BY Enrolled_Students DESC;
```

Explanation:-

- SELECT COURSE_NAME, COUNT(EnrollmentInfo.STU_ID) AS Enrolled_Students
  Selects the course name (COURSE_NAME) and counts the number of students (STU_ID) enrolled in the course, labeling it as Enrolled_Students.
- FROM CoursesInfo
  Specifies the CoursesInfo table as the base table.
- JOIN EnrollmentInfo ON CoursesInfo.COURSE_ID = EnrollmentInfo.COURSE_ID
  Joins CoursesInfo with EnrollmentInfo using COURSE_ID.
- WHERE ENROLL_STATUS = 'Enrolled'
  Filters the rows to include only those where the enrollment status is 'Enrolled'.
- GROUP BY COURSE_NAME
  Groups the result by course name, so the COUNT function calculates for each course.
- ORDER BY Enrolled_Students DESC
  Orders the result in descending order of enrolled students, showing the course with the most students first.