Query-1: Add a New Student

Output:

```
Select * from Students;

student_id      name    email   phone_number    enrollment_date
1        Liam Gentry    lima@example.com        1234567890      2024-10-18
2        Lyric Glass    lyric@example.com       0987654321      2024-10-18
3       Julio Stokes    julio@example.com       0987654321      2024-10-18
```

Explanation:

This SQL statement inserts a new record into the Students table. The student's name is Julio Stokes, with the email julio@example.com and the phone number 0987654321. The INSERT INTO statement is used to add these values to the respective columns in the table.

Query-2: Enroll a Student in a Course

Output:

```
Select * from Enrollments;
enrollment_id   student_id      course_id       enrollment_date
1       1       1       2024-10-18
2       2       2       2024-10-18
3       1       2       2024-10-18
```

Explanation:

This SQL statement enrolls the student with student_id = 1 into the course with course_id = 2. The INSERT INTO statement is used to add a new record to the Enrollments table, linking the student and the course by their respective IDs.

Query-3: Assign a Grade to a Student

Output:

```
Select * from Grades;
grade_id        student_id      course_id       grade
1       1       1       A
2       2       2       B
3       1       2       A
```

Explanation:

This SQL statement assigns a grade of 'A' to the student with student_id = 1 for the course with course_id = 2. The INSERT INTO statement adds a new record to the Grades table, linking the student's ID, the course ID, and the assigned grade.

Query-4: List All Students Enrolled in a Course

```
name    course_name
 Liam Gentry    Mathematics
```

Explanation:

This SQL query retrieves a list of all students who are enrolled in course 1 along with the course name. It selects the student's name from the Students table and the course name from the Courses table. The Enrollments table acts as a link between the Students and Courses tables, and the query filters for records where the course_id is 1, meaning it only shows students enrolled in that specific course.

Query-5: View Student's Grades

Output:

```
name    course_name     grade
 Liam Gentry    Mathematics     A
```

Explanation:

This SQL query retrieves the grades of student 1 across all the courses they are enrolled in. It selects the student's name, the course name, and the corresponding grade from the Grades table. The Students and Courses tables are joined to provide the student's details and course names, and the query filters for records where the student_id is 1, showing only the grades for that specific student.

Query-6: Update a Student's Contact Information

Output:

```
Select * from Students;
student_id      name    email   phone_number    enrollment_date
1       Liam Gentry     lima@example.com        9876543210      2024-10-18
2       Lyric Glass     lyric@example.com       0987654321      2024-10-18
```

Explanation:

This SQL statement updates the phone_number of the student with student_id = 1 to '9876543210'. The UPDATE statement modifies the existing record in the Students table where the student_id matches the specified value. It ensures that only the phone number for student 1 is changed.

Query-7: Remove a Student from a Course

Output:

```
Select * from Enrollments;
enrollment_id   student_id      course_id       enrollment_date
1       1       1       2024-10-18
2       2       2       2024-10-18
```

Explanation:

This SQL statement deletes the enrollment record for the student with student_id = 1 in the course with course_id = 2. The DELETE FROM statement removes the matching record from the Enrollments table where both the student and course IDs match the specified values, effectively unenrolling the student from that course.

Query-8: View All Courses Taught by a Specific Teacher

Output:

```
course_name
Physics
```

Explanation:

This SQL query retrieves the names of all courses taught by the teacher with teacher_id = 2. The SELECT statement is used to get the course_name from the Courses table, and the query is filtered by teacher_id, so it only returns courses that are assigned to the specific teacher.

Query-9: Count the Number of Students in a Course

Output:

```
total_students
1
```

Explanation:

This SQL query counts the total number of students enrolled in course 1. The COUNT(*) function returns the total number of records in the Enrollments table that match the condition course_id = 1. The result is labeled as total_students to show how many students are currently enrolled in that specific course.

Query-10: List Students Who Have Not Yet Been Assigned a Grade

Output:

```
name
 Liam Gentry
```

Explanation:

This SQL query retrieves the names of students who are not enrolled in course 2 or who have not received a grade for that course. It uses a LEFT JOIN to combine the Students table with the Grades table based on student_id and filters for records where the grade is NULL. This means it lists students who either have no corresponding entry in the Grades table for course 2, indicating they haven't received a grade or are not enrolled in the course at all.

Query-11: Calculate the Average Grade for a Course

Output:

```
avg_grade
4.0000
```

Explanation:

This SQL query calculates the average grade point for all students enrolled in course 1.

It uses a CASE statement to convert letter grades into numerical values (A = 4, B = 3, C = 2, D = 1, and any other grade is considered as 0). The AVG() function then computes the **average**

of these numerical values. The result is labeled as avg_grade and represents the overall performance of students in the specified course based on their grades.

Query-12: Find the Highest Grade Assigned in a Course

Output:

```
highest_grade
B
```

Explanation:

This SQL query retrieves the highest grade achieved by students in course 2. It uses the MAX() function to find the maximum value in the grade column from the Grades table, specifically filtering for records where course_id = 2. The result is labeled as highest_grade, indicating the top grade received in that particular course.

Query-13: List Students with the Same Grade in a Course

Output:

```
Output not generated for insufficient data
```

Explanation:

This SQL query identifies the names of students and their grades for course 1, but only for those grades that have been assigned to more than one student. It joins the Grades table with the Students table based on student_id. The GROUP BY clause groups the results by grade, and the HAVING clause filters the grouped results to only include grades that appear more than once. This helps to find common grades among students in that specific course.

Query-14: Assign a Teacher to a New Course

Output:

```
Select * from Courses;
course_id       course_name     teacher_id
1       Mathematics     1
2       Physics 2
3       Chemistry       3
```

Explanation:

This SQL code inserts a new record into the Courses table. It specifies that a course named "Chemistry" is being added, and it associates this course with a teacher whose ID is 3. The INSERT INTO statement is used to add new rows to a table, and the VALUES clause provides the data to be inserted into the respective columns of the table.

Query-15: Find the Total Number of Courses Each Student is Enrolled In

Output:

```
name    total_courses
 Liam Gentry     1
 Lyric Glass     1
```

Explanation:

This SQL query retrieves the names of students along with the total number of courses they are enrolled in. It joins the Enrollments table with the Students table based on the student_id to associate each enrollment with the corresponding student. The COUNT(e.course_id) function counts the number of courses for each student, and the results are grouped by the student's name using the GROUP BY clause. This allows for a summary of course enrollments per student.

Query-16: List All Courses a Student is Enrolled In

Output:

```
course_name
Mathematics
```

Explanation:

This SQL query retrieves the names of courses in which a specific student, identified by student_id = 1, is enrolled. It does so by joining the Enrollments table with the Courses table on the course_id. The join operation connects each enrollment record with its corresponding course record, allowing the query to select the course names associated with the specified student. The WHERE clause filters the results to include only the courses for the specified student.

Query-17: Delete a Course and All Related Data

Output:

```
Select * from Courses;
Select * from Enrollments;
Select * from Grades;
course_id       course_name     teacher_id
1       Mathematics     1
enrollment_id   student_id      course_id       enrollment_date
1       1       1       2024-10-18
grade_id        student_id      course_id       grade
1       1       1       A
```

Explanation:

The provided SQL code consists of three statements that delete data related to a specific course identified by course_id = 2. The first statement removes the course from the Courses table. The second statement deletes any enrollment records associated with that course from the Enrollments table. Finally, the third statement eliminates any grades recorded for that course from the Grades table. This cascading deletion ensures that all references to the course are removed from the database, preventing orphaned records in the related tables.