

-- Create Students table

CREATE TABLE

```
Students (  
    student_id INT PRIMARY KEY,  
    student_name VARCHAR(100),  
    student_major VARCHAR(100)  
);
```

-- Create Courses table

CREATE TABLE

```
Courses (  
    course_id INT PRIMARY KEY,  
    course_name VARCHAR(100),  
    course_description VARCHAR(255)  
);
```

-- Create Enrollments table

CREATE TABLE

```
Enrollments (  
    enrollment_id INT PRIMARY KEY,  
    student_id INT,  
    course_id INT,  
    enrollment_date DATE,  
    FOREIGN KEY (student_id) REFERENCES Students (student_id),  
    FOREIGN KEY (course_id) REFERENCES Courses (course_id)  
);
```

-- Insert data into Students table

INSERT INTO

```
Students (student_id, student_name, student_major)
```

VALUES

```
(1, 'Alice', 'Computer Science'),  
(2, 'Bob', 'Biology'),  
(3, 'Charlie', 'History'),  
(4, 'Diana', 'Mathematics');
```

-- Insert data into Courses table

INSERT INTO

```
Courses (course_id, course_name, course_description)
```

VALUES

```
(  
    101,  
    'Introduction to CS',  
    'Basics of Computer Science'  
),  
(102, 'Biology Basics', 'Fundamentals of Biology'),  
(
```

```
103,  
  'World History',  
  'Historical events and cultures'  
) ,  
(104, 'Calculus I', 'Introduction to Calculus'),  
(105, 'Data Structures', 'Advanced topics in CS');
```

-- Insert data into Enrollments table

```
INSERT INTO  
  Enrollments (  
    enrollment_id,  
    student_id,  
    course_id,  
    enrollment_date  
  )  
VALUES  
  (1, 1, 101, '2023-01-15'),  
  (2, 2, 102, '2023-01-20'),  
  (3, 3, 103, '2023-02-01'),  
  (4, 1, 105, '2023-02-05'),  
  (5, 4, 104, '2023-02-10'),  
  (6, 2, 101, '2023-02-12'),  
  (7, 3, 105, '2023-02-15'),  
  (8, 4, 101, '2023-02-20'),  
  (9, 1, 104, '2023-03-01'),  
  (10, 2, 104, '2023-03-05');
```

--Queries

-- 1. Inner Join: Retrieve the list of students and their enrolled courses

SELECT

s.student\_id AS student\_id,

s.student\_name AS student\_name,

c.course\_name AS course\_name

FROM

students s

INNER JOIN enrollments e ON s.student\_id = e.student\_id

INNER JOIN courses c ON e.course\_id = c.course\_id;

123 student_id ▼	ABC student_name ▼	ABC course_name ▼
1	Alice	Introduction to CS
2	Bob	Biology Basics
3	Charlie	World History
1	Alice	Data Structures
4	Diana	Calculus I
2	Bob	Introduction to CS
3	Charlie	Data Structures
4	Diana	Introduction to CS
1	Alice	Calculus I
2	Bob	Calculus I

-- 2. Left Join: List all students and their enrolled courses, including those who haven't enrolled in any course

SELECT

s.student\_id AS student\_id,  
s.student\_name AS student\_name,  
c.course\_name AS course\_name

FROM

students s

LEFT JOIN enrollments e ON s.student\_id = e.student\_id

LEFT JOIN courses c ON e.course\_id = c.course\_id;

123 student_id	ABC student_name	ABC course_name
1	Alice	Introduction to CS
2	Bob	Biology Basics
3	Charlie	World History
1	Alice	Data Structures
4	Diana	Calculus I
2	Bob	Introduction to CS
3	Charlie	Data Structures
4	Diana	Introduction to CS
1	Alice	Calculus I
2	Bob	Calculus I

-- 3. Right Join: Display all courses and the students enrolled in each course, including courses with no enrolled students

SELECT

c.course\_id AS course\_id,  
c.course\_name AS course\_name,  
s.student\_name AS student\_name

FROM

courses c

RIGHT JOIN enrollments e ON c.course\_id = e.course\_id

RIGHT JOIN students s ON e.student\_id = s.student\_id;

123 course_id	ABC course_name	ABC student_name
101	Introduction to CS	Alice
102	Biology Basics	Bob
103	World History	Charlie
105	Data Structures	Alice
104	Calculus I	Diana
101	Introduction to CS	Bob
105	Data Structures	Charlie
101	Introduction to CS	Diana
104	Calculus I	Alice
104	Calculus I	Bob

-- 4. Self Join: Find pairs of students who are enrolled in at least one common course

```
SELECT

s1.student_name AS student_name_1,

s2.student_name AS student_name_2,

e1.course_id AS course_id

FROM

enrollments e1

JOIN enrollments e2 ON e1.course_id = e2.course_id

AND e1.student_id < e2.student_id

JOIN students s1 ON e1.student_id = s1.student_id

JOIN students s2 ON e2.student_id = s2.student_id;
```

ABC student_name_1	ABC student_name_2	123 course_id
Alice	Bob	101
Alice	Diana	101
Bob	Diana	101
Alice	Diana	104
Alice	Bob	104
Bob	Diana	104
Alice	Charlie	105

-- 5. Complex Join: Retrieve students who are enrolled in 'Introduction to CS' but not in 'Data Structures'

```
SELECT

s.student_id AS student_id,

s.student_name AS student_name

FROM

students s

INNER JOIN enrollments e1 ON s.student_id = e1.student_id

INNER JOIN courses c1 ON e1.course_id = c1.course_id

AND c1.course_name = 'Introduction to CS'

WHERE

s.student_id NOT IN (

SELECT
```

```
s.student_id
FROM
students s
INNER JOIN enrollments e2 ON s.student_id = e2.student_id
INNER JOIN courses c2 ON e2.course_id = c2.course_id
AND c2.course_name = 'Data Structures'
);
```

123 student_id	ABC student_name
2	Bob
4	Diana