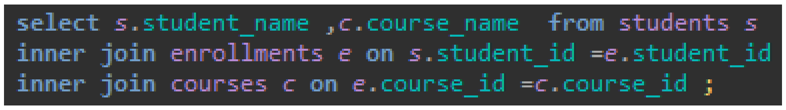
Assignment 3:

```
-- 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'),
```

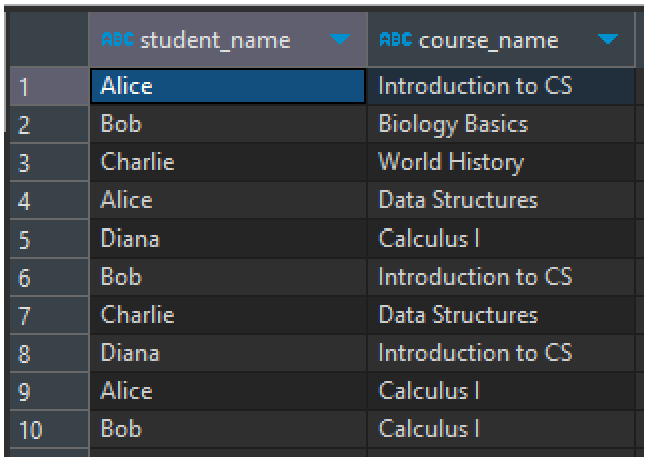
(2, 2, 102, '2023-01-20'),
(3, 3, 103, '2023-02-01'),
(4, 1, 105, '2023-02-05'),
(5, 4, 104, '2023-02-10'),
(6, 2, 101, '2023-02-12'),
(7, 3, 105, '2023-02-15'),
(8, 4, 101, '2023-02-20'),
(9, 1, 104, '2023-03-01'),
(10, 2, 104, '2023-03-05');

1. Inner Join:

Question: Retrieve the list of students and their enrolled courses.



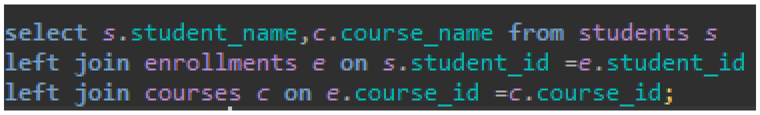
```
select s.student_name ,c.course_name  from students s
inner join enrollments e on s.student_id =e.student_id
inner join courses c on e.course_id =c.course_id ;
```



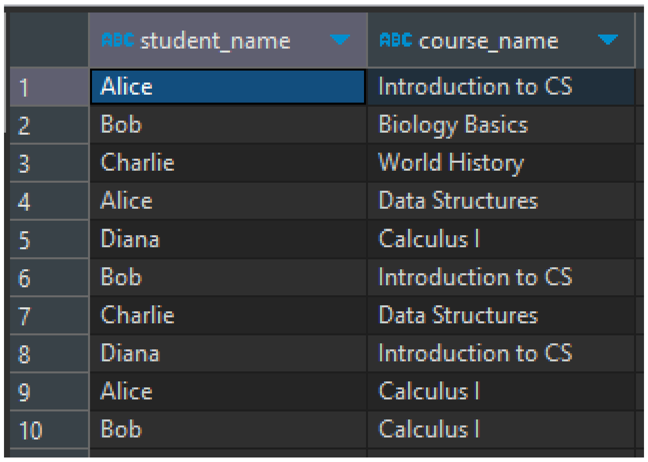
	student_name	course_name
1	Alice	Introduction to CS
2	Bob	Biology Basics
3	Charlie	World History
4	Alice	Data Structures
5	Diana	Calculus I
6	Bob	Introduction to CS
7	Charlie	Data Structures
8	Diana	Introduction to CS
9	Alice	Calculus I
10	Bob	Calculus I

2. Left Join:

Question: List all students and their enrolled courses, including those who haven't enrolled in any course.



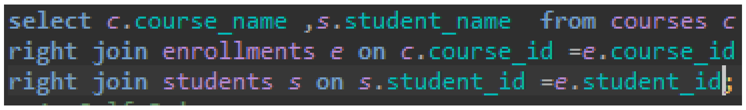
```sql
select s.student_name,c.course_name from students s
left join enrollments e on s.student_id =e.student_id
left join courses c on e.course_id =c.course_id;
```



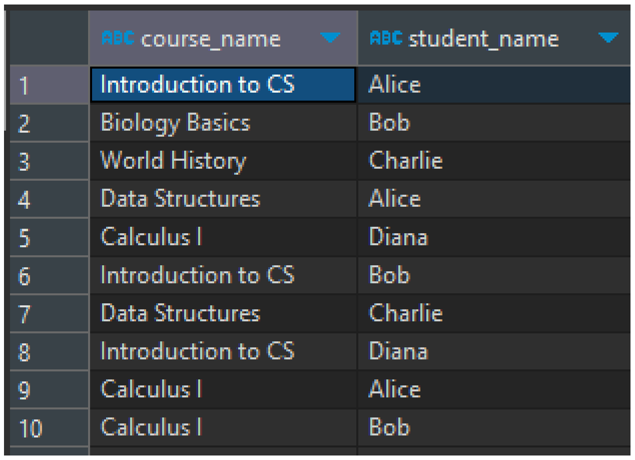
	student_name	course_name
1	Alice	Introduction to CS
2	Bob	Biology Basics
3	Charlie	World History
4	Alice	Data Structures
5	Diana	Calculus I
6	Bob	Introduction to CS
7	Charlie	Data Structures
8	Diana	Introduction to CS
9	Alice	Calculus I
10	Bob	Calculus I

3. Right Join:

Question: Display all courses and the students enrolled in each course, including courses with no enrolled students.



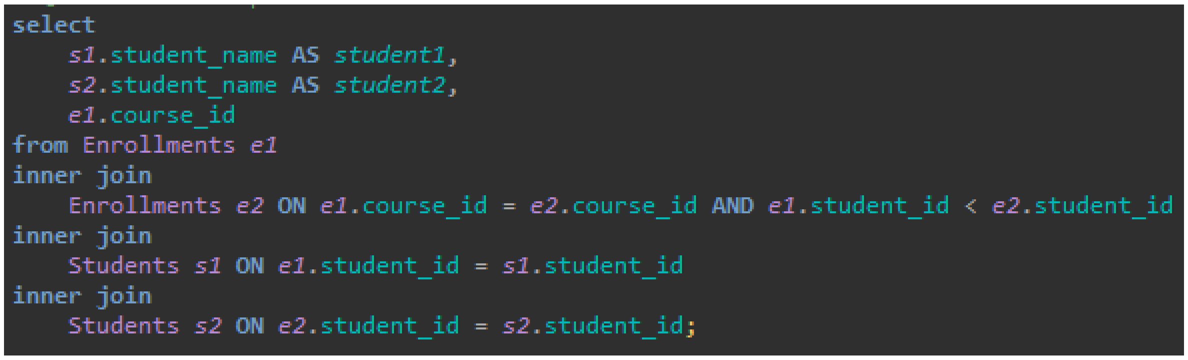
```sql
select c.course_name ,s.student_name  from courses c
right join enrollments e on c.course_id =e.course_id
right join students s on s.student_id =e.student_id;
```



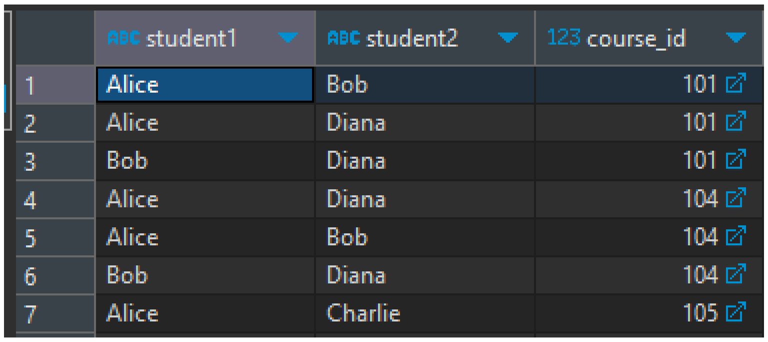
	course_name	student_name
1	Introduction to CS	Alice
2	Biology Basics	Bob
3	World History	Charlie
4	Data Structures	Alice
5	Calculus I	Diana
6	Introduction to CS	Bob
7	Data Structures	Charlie
8	Introduction to CS	Diana
9	Calculus I	Alice
10	Calculus I	Bob

4. Self Join:

Question: Find pairs of students who are enrolled in at least one common course.



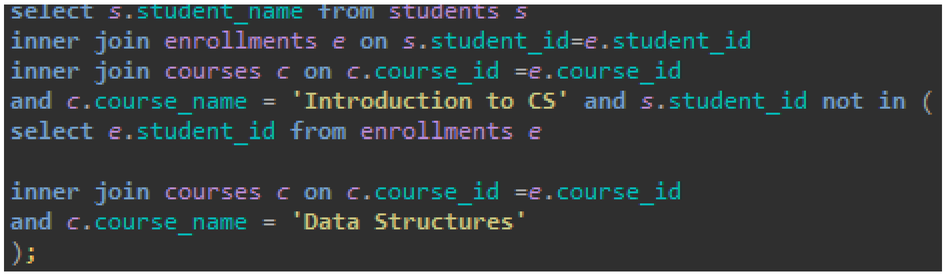
```sql
select
    s1.student_name AS student1,
    s2.student_name AS student2,
    e1.course_id
from Enrollments e1
inner join
    Enrollments e2 ON e1.course_id = e2.course_id AND e1.student_id < e2.student_id
inner join
    Students s1 ON e1.student_id = s1.student_id
inner join
    Students s2 ON e2.student_id = s2.student_id;
```



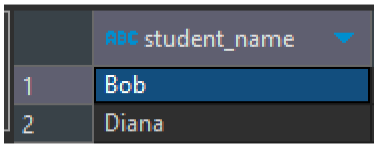
	student1	student2	course_id
1	Alice	Bob	101
2	Alice	Diana	101
3	Bob	Diana	101
4	Alice	Diana	104
5	Alice	Bob	104
6	Bob	Diana	104
7	Alice	Charlie	105

5. Complex Join:

Question: Retrieve students who are enrolled in 'Introduction to CS' but not in 'Data Structures'.



```sql
select s.student_name from students s
inner join enrollments e on s.student_id=e.student_id
inner join courses c on c.course_id =e.course_id
and c.course_name = 'Introduction to CS' and s.student_id not in (
select e.student_id from enrollments e

inner join courses c on c.course_id =e.course_id
and c.course_name = 'Data Structures'
);
```

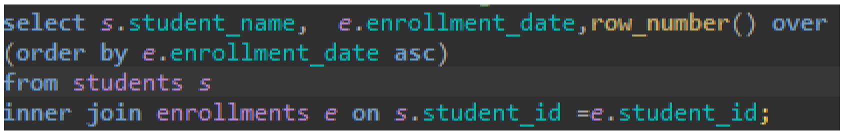


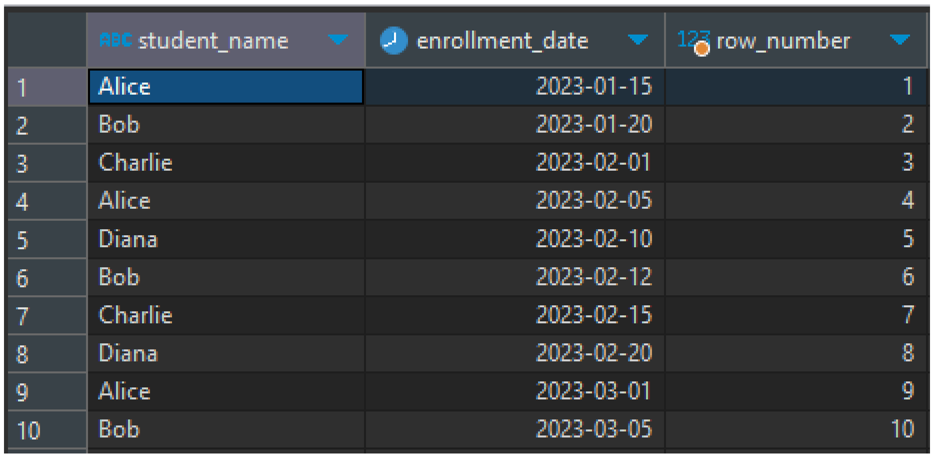
	student_name
1	Bob
2	Diana

Windows function:

1. Using `ROW_NUMBER()`:

Question: List all students along with a row number based on their enrollment date in ascending order.

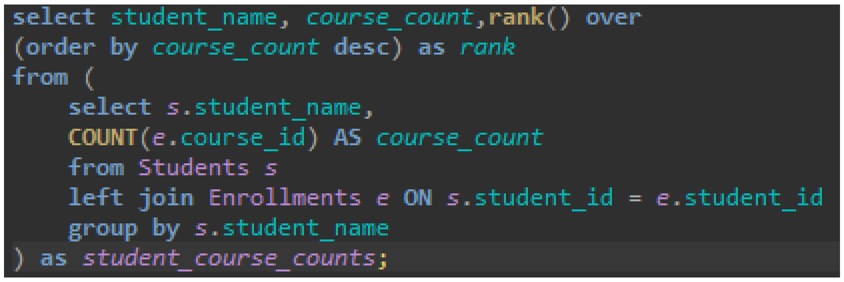

```
select s.student_name,  e.enrollment_date,row_number() over
(order by e.enrollment_date asc)
from students s
inner join enrollments e on s.student_id =e.student_id;
```

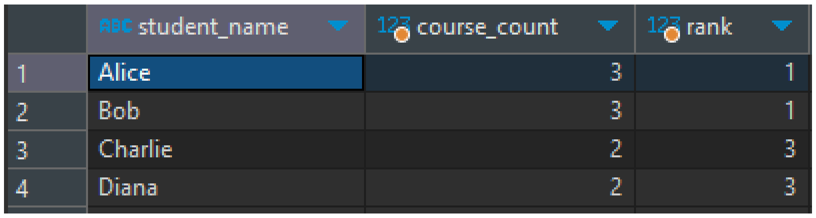


	student_name	enrollment_date	row_number
1	Alice	2023-01-15	1
2	Bob	2023-01-20	2
3	Charlie	2023-02-01	3
4	Alice	2023-02-05	4
5	Diana	2023-02-10	5
6	Bob	2023-02-12	6
7	Charlie	2023-02-15	7
8	Diana	2023-02-20	8
9	Alice	2023-03-01	9
10	Bob	2023-03-05	10

2. Using `RANK()`:

Question: Rank students based on the number of courses they are enrolled in, handling ties by assigning the same rank.

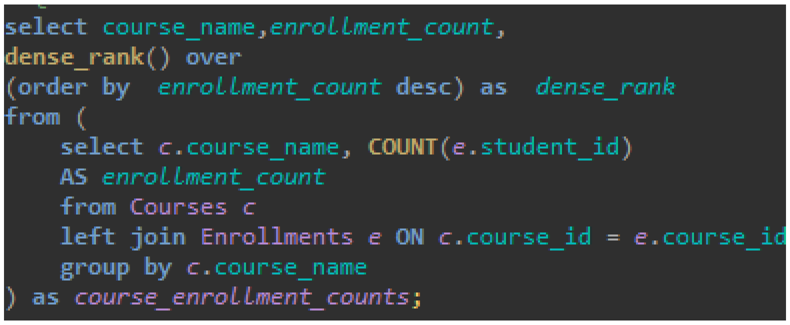

```
select student_name, course_count,rank() over
(order by course_count desc) as rank
from (
    select s.student_name,
    COUNT(e.course_id) AS course_count
    from Students s
    left join Enrollments e ON s.student_id = e.student_id
    group by s.student_name
) as student_course_counts;
```



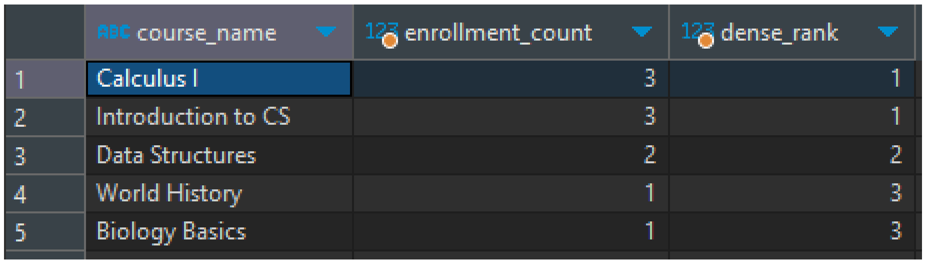
	student_name	course_count	rank
1	Alice	3	1
2	Bob	3	1
3	Charlie	2	3
4	Diana	2	3

3. Using `DENSE_RANK()`:

Question: Determine the dense rank of courses based on their enrollment count across all students



```
select course_name,enrollment_count,
dense_rank() over
(order by  enrollment_count desc) as  dense_rank
from (
    select c.course_name, COUNT(e.student_id)
    AS enrollment_count
    from Courses c
    left join Enrollments e ON c.course_id = e.course_id
    group by c.course_name
) as course_enrollment_counts;
```



	course_name	enrollment_count	dense_rank
1	Calculus I	3	1
2	Introduction to CS	3	1
3	Data Structures	2	2
4	World History	1	3
5	Biology Basics	1	3