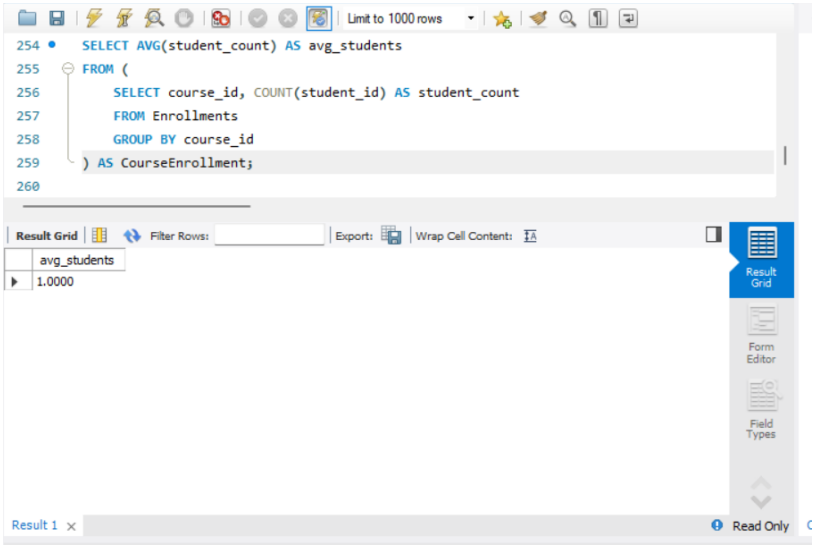
ASSIGNMENT STUDENT INFORMATION SYSTEM

Task 4. Subquery and its type:

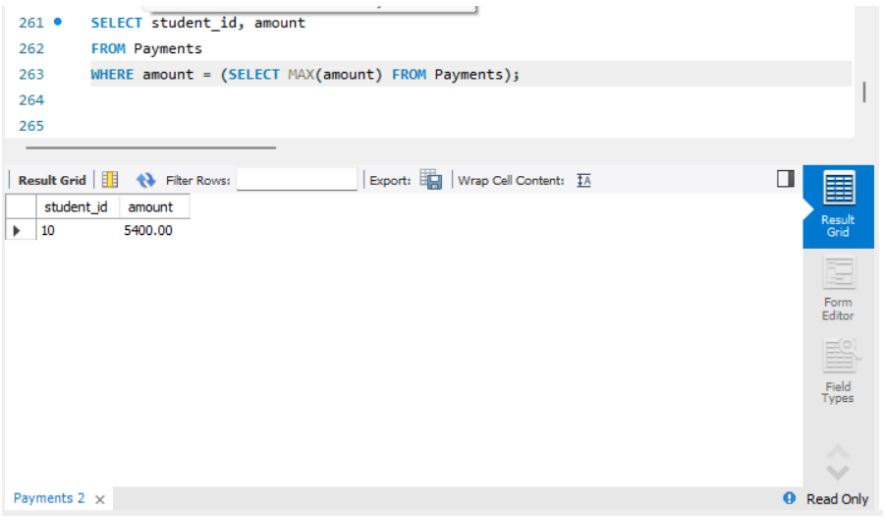
1. Write an SQL query to calculate the average number of students enrolled in each course. Use aggregate functions and subqueries to achieve this.

```
SELECT AVG(student_count) AS avg_students
FROM (
    SELECT course_id, COUNT(student_id) AS student_count
    FROM Enrollments
    GROUP BY course_id
) AS CourseEnrollment;
```



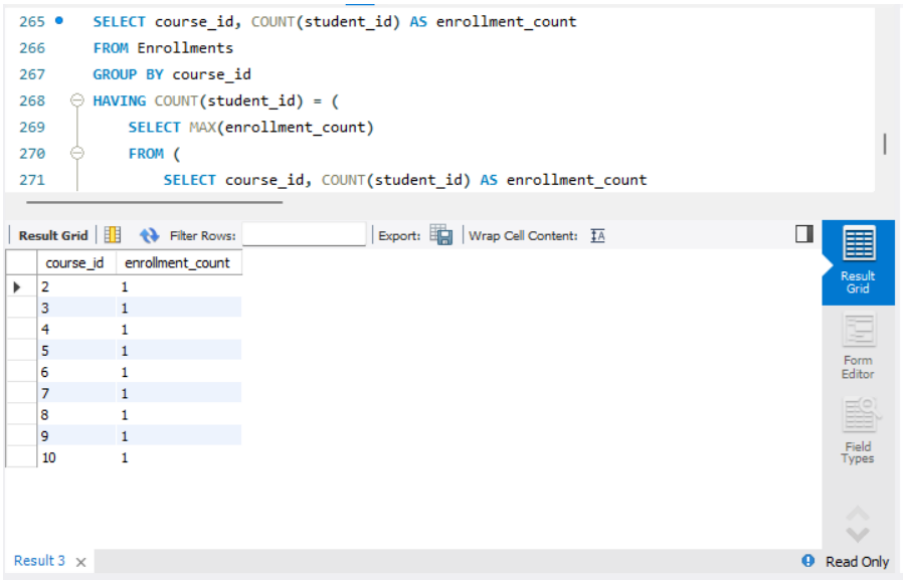
2. Identify the student(s) who made the highest payment. Use a subquery to find the maximum payment amount and then retrieve the student(s) associated with that amount.

```
SELECT student_id, amount
FROM Payments
WHERE amount = (SELECT MAX(amount) FROM Payments);
```



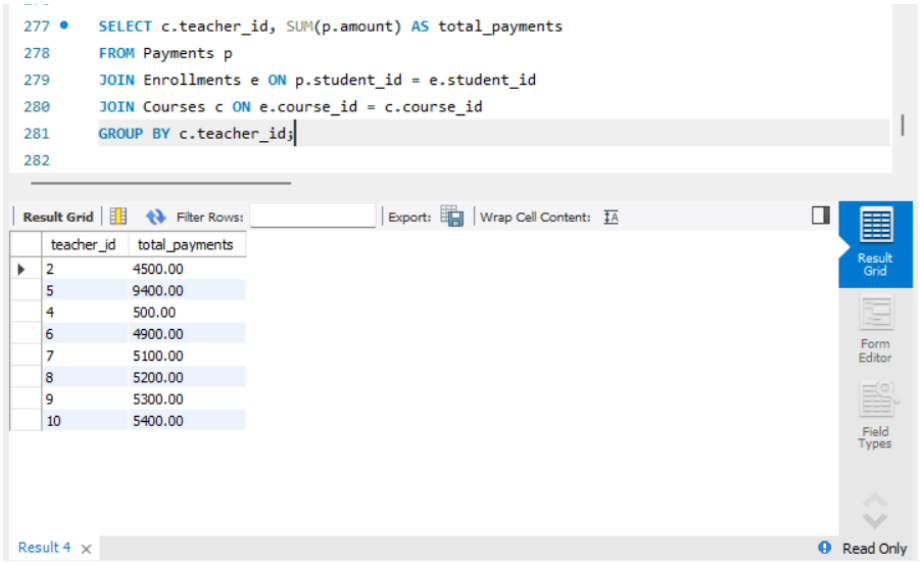
3. Retrieve a list of courses with the highest number of enrollments. Use subqueries to find the course(s) with the maximum enrollment count.

```
SELECT course_id, COUNT(student_id) AS enrollment_count
FROM Enrollments
GROUP BY course_id
HAVING COUNT(student_id) = (
    SELECT MAX(enrollment_count)
    FROM (
        SELECT course_id, COUNT(student_id) AS enrollment_count
        FROM Enrollments
        GROUP BY course_id
    ) AS CourseCounts
);
```



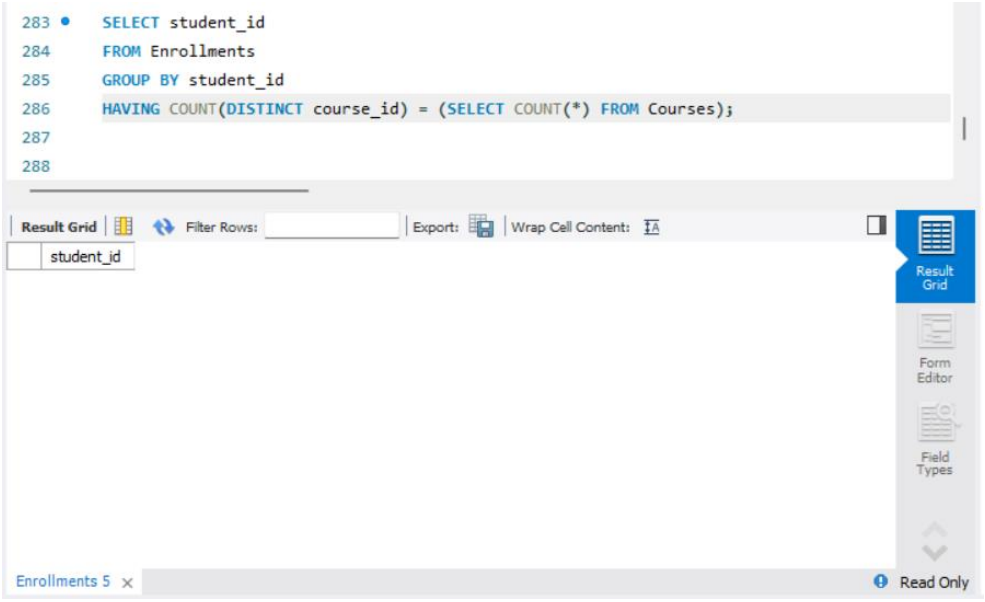
4. Calculate the total payments made to courses taught by each teacher. Use subqueries to sum payments for each teacher's courses.

```
SELECT c.teacher_id, SUM(p.amount) AS total_payments
FROM Payments p
JOIN Enrollments e ON p.student_id = e.student_id
JOIN Courses c ON e.course_id = c.course_id
GROUP BY c.teacher_id;
```



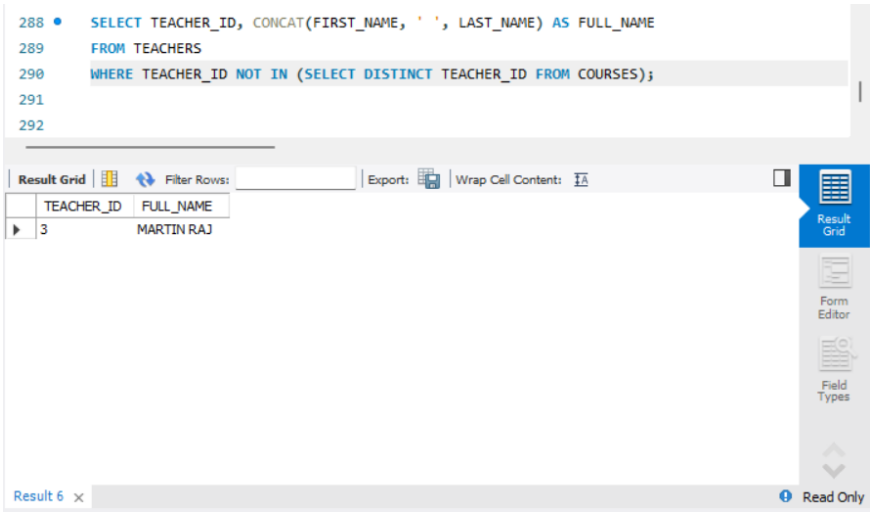
5. Identify students who are enrolled in all available courses. Use subqueries to compare a student's enrollments with the total number of courses.

```
SELECT student_id
FROM Enrollments
GROUP BY student_id
HAVING COUNT(DISTINCT course_id) = (SELECT COUNT(*) FROM Courses);
```



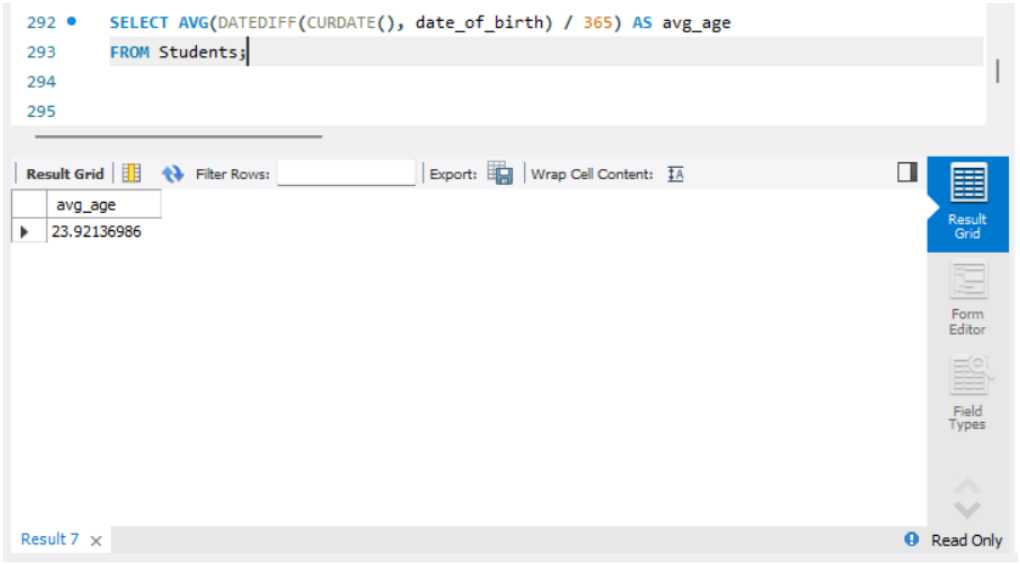
6. Retrieve the names of teachers who have not been assigned to any courses. Use subqueries to find teachers with no course assignments

```
SELECT TEACHER_ID, CONCAT(FIRST_NAME, ' ', LAST_NAME) AS FULL_NAME
FROM TEACHERS
WHERE TEACHER_ID NOT IN (SELECT DISTINCT TEACHER_ID FROM COURSES);
```



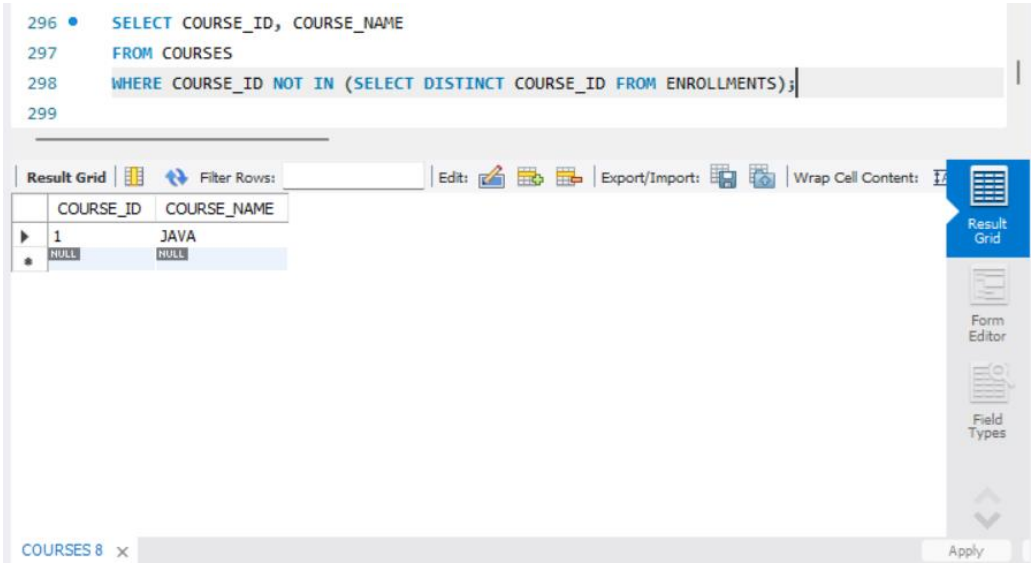
7. Calculate the average age of all students. Use subqueries to calculate the age of each student based on their date of birth.

```
SELECT AVG(DATEDIFF(CURDATE(), date_of_birth) / 365) AS avg_age
FROM Students;
```



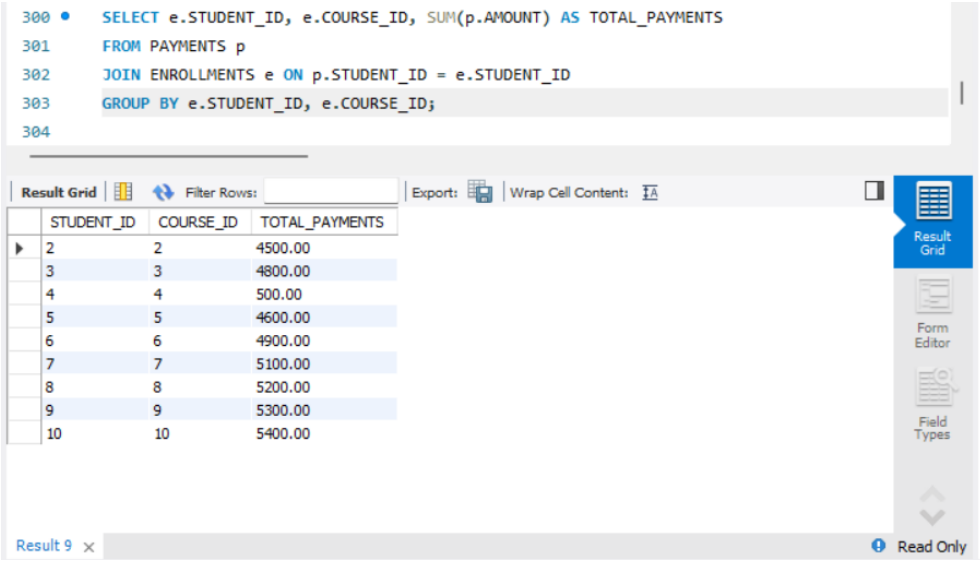
8. Identify courses with no enrollments. Use subqueries to find courses without enrollment records.

```
SELECT COURSE_ID, COURSE_NAME
FROM COURSES
WHERE COURSE_ID NOT IN (SELECT DISTINCT COURSE_ID FROM ENROLLMENTS);
```



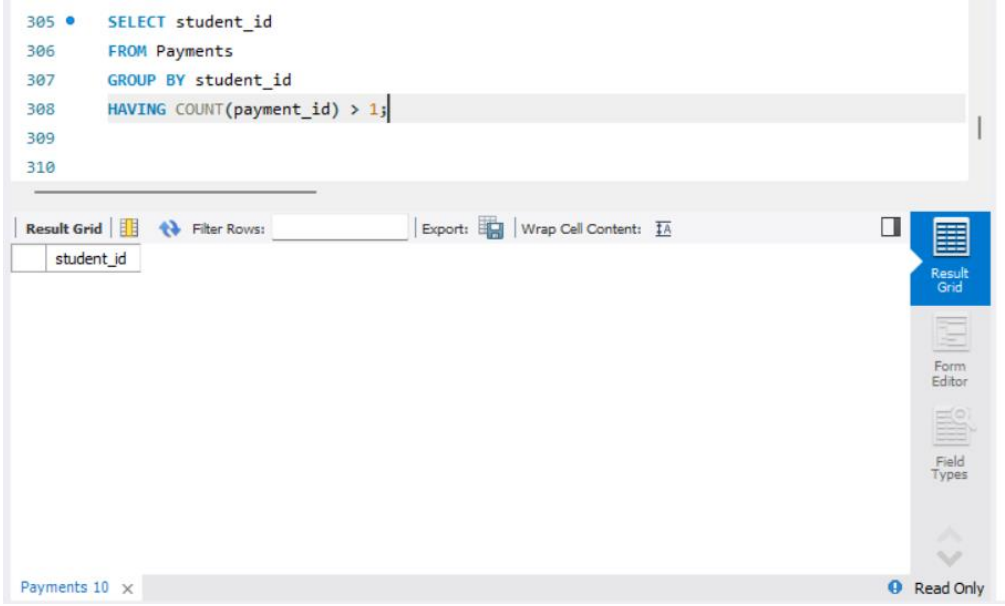
9. Calculate the total payments made by each student for each course they are enrolled in. Use subqueries and aggregate functions to sum payments.

```
SELECT e.STUDENT_ID, e.COURSE_ID, SUM(p.AMOUNT) AS TOTAL_PAYMENTS
FROM PAYMENTS p
JOIN ENROLLMENTS e ON p.STUDENT_ID = e.STUDENT_ID
GROUP BY e.STUDENT_ID, e.COURSE_ID;
```



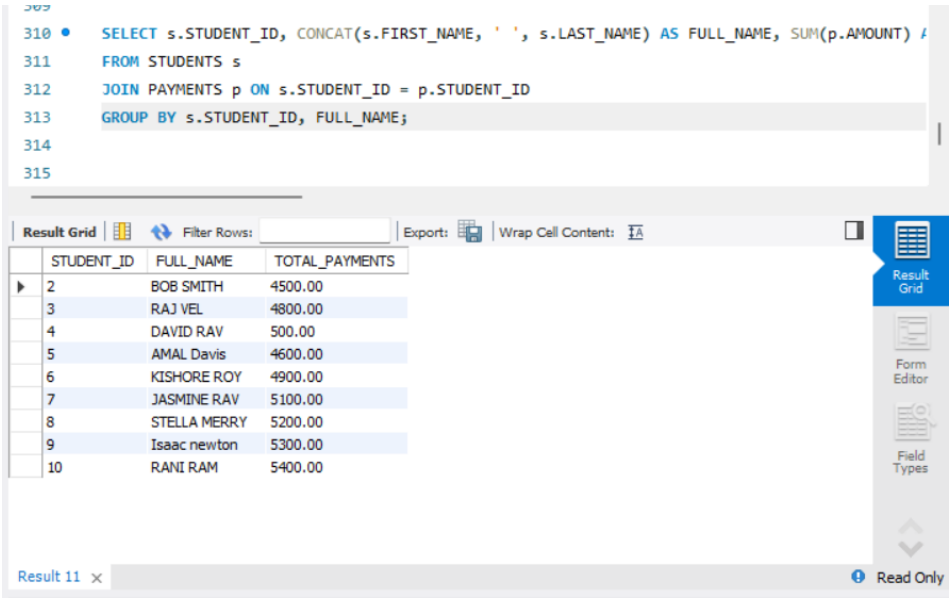
10. Identify students who have made more than one payment. Use subqueries and aggregate functions to count payments per student and filter for those with counts greater than one.

```
SELECT student_id
FROM Payments
GROUP BY student_id
HAVING COUNT(payment_id) > 1;
```



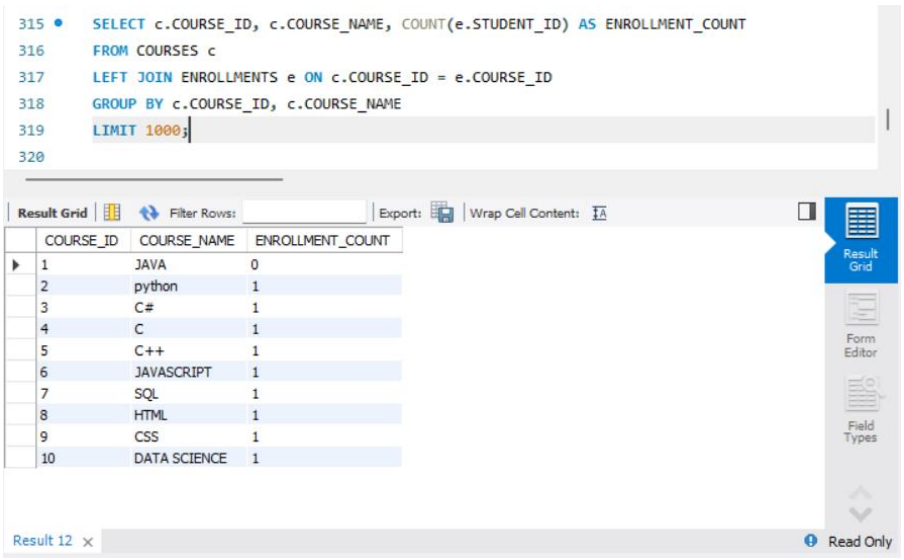
11. Write an SQL query to calculate the total payments made by each student. Join the "Students" table with the "Payments" table and use GROUP BY to calculate the sum of payments for each student.

```
SELECT s.STUDENT_ID, CONCAT(s.FIRST_NAME, ' ', s.LAST_NAME) AS FULL_NAME,
SUM(p.AMOUNT) AS TOTAL_PAYMENTS
FROM STUDENTS s
JOIN PAYMENTS p ON s.STUDENT_ID = p.STUDENT_ID
GROUP BY s.STUDENT_ID, FULL_NAME;
```



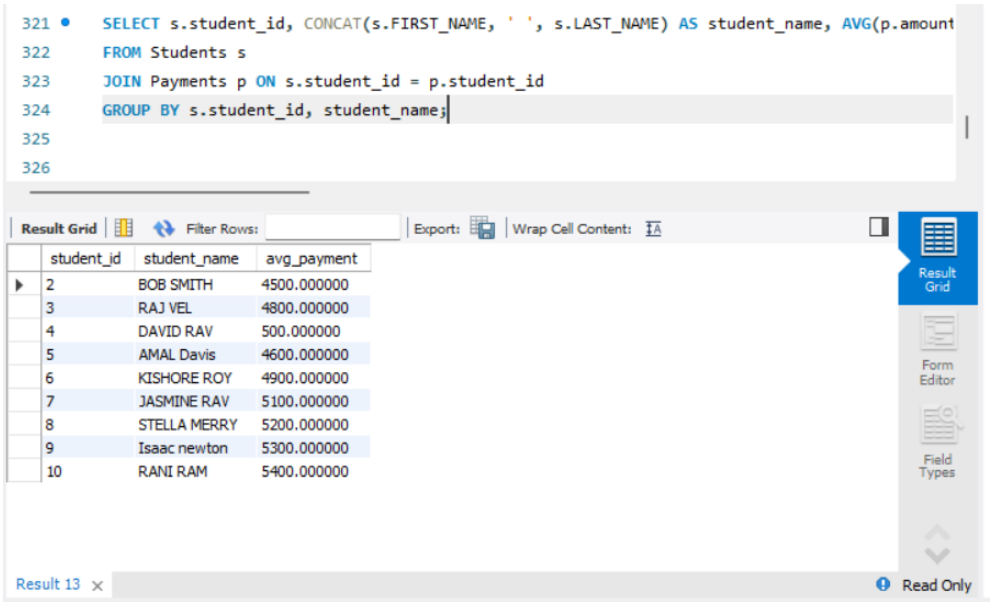
12. Retrieve a list of course names along with the count of students enrolled in each course. Use JOIN operations between the "Courses" table and the "Enrollments" table and GROUP BY to count enrollments.

```
SELECT c.COURSE_ID, 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_ID, c.COURSE_NAME
LIMIT 1000;
```



13. Calculate the average payment amount made by students. Use JOIN operations between the "Students" table and the "Payments" table and GROUP BY to calculate the average.

```
SELECT s.student_id, CONCAT(s.FIRST_NAME, ' ', s.LAST_NAME) AS student_name,
AVG(p.amount) AS avg_payment
FROM Students s
JOIN Payments p ON s.student_id = p.student_id
GROUP BY s.student_id, student_name;
```



```
321 •  SELECT s.student_id, CONCAT(s.FIRST_NAME, ' ', s.LAST_NAME) AS student_name, AVG(p.amount
322    FROM Students s
323    JOIN Payments p ON s.student_id = p.student_id
324    GROUP BY s.student_id, student_name;
325
326
```

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

student_id	student_name	avg_payment
2	BOB SMITH	4500.000000
3	RAJ VEL	4800.000000
4	DAVID RAV	500.000000
5	AMAL Davis	4600.000000
6	KISHORE ROY	4900.000000
7	JASMINE RAV	5100.000000
8	STELLA MERRY	5200.000000
9	Isaac newton	5300.000000
10	RANI RAM	5400.000000

Result Grid | Form Editor | Field Types

Result 13 | Read Only