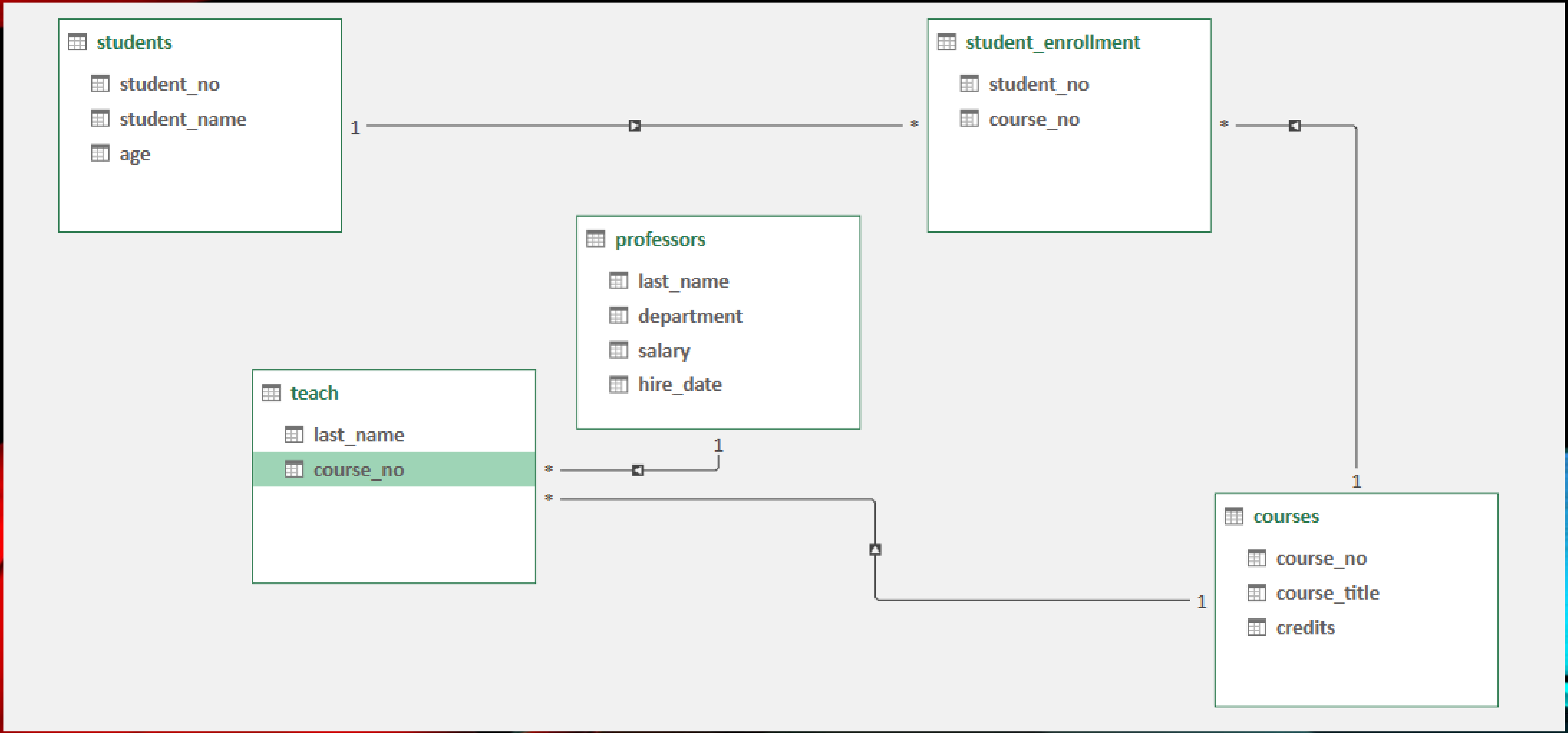
Entity Relationship Model of College Database



students
- student_no
- student_name
- age

student_enrollment
- student_no
- course_no

professors
- last_name
- department
- salary
- hire_date

teach
- last_name
- course_no

courses
- course_no
- course_title
- credits

THIS IS THE COLLEGE DATABASE

- Total five Tables

1. Students
2. Student enrollment
3. Courses
4. Professors
5. Teach

SIMPLE QUESTIONS

Q 1; Write a query to display the names of those students that are between ages of 18 and 20

Q 2; Write a query to display all of those students that contain the letters "ch" in their name or their name ends with letters "nd".

Q 3; Write a query to display the name of those students that have the letters "ae" or "ph" in their name and are NOT 19 years old

Q 4; Write a query that lists the names of students sorted by their age from largest to smallest

Q 5; Write a query that displays the names and ages of the top 4 oldest students

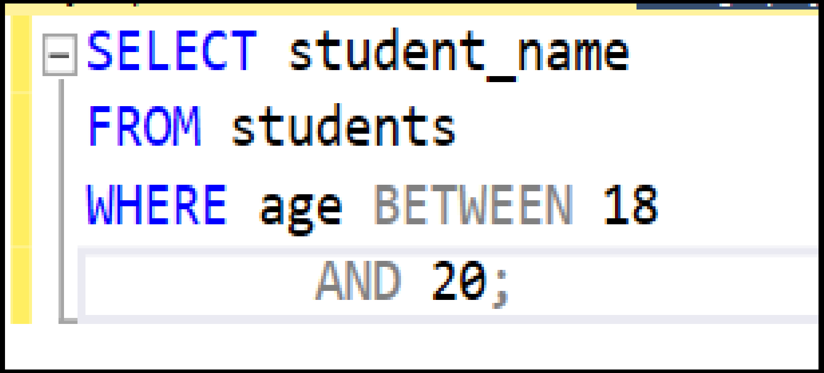
Q6; Write a query that returns students base on the following criteria

The student must not be older than age 20

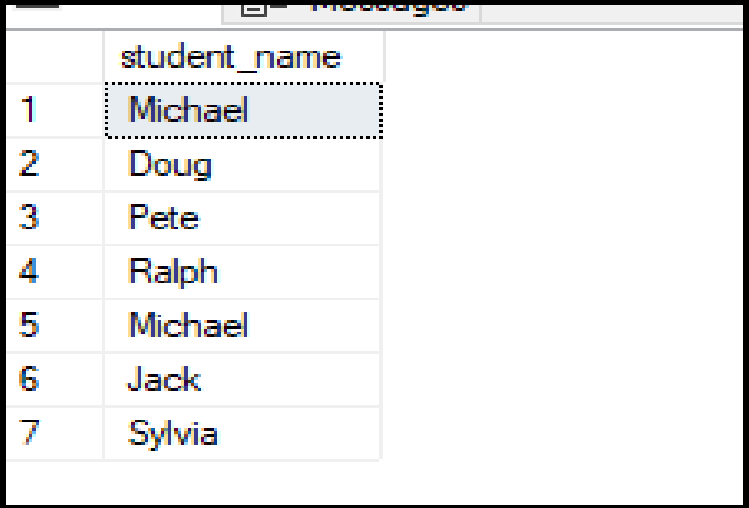
if their student no is either 3 and 5 or their student no is 7.

your query should also return students older than age 20 but in that case they must have a student no that is at least 4.

Q 1; WRITE A QUERY TO DISPLAY THE NAMES OF THOSE STUDENTS THAT ARE BETWEEN AGES OF 18 AND 20

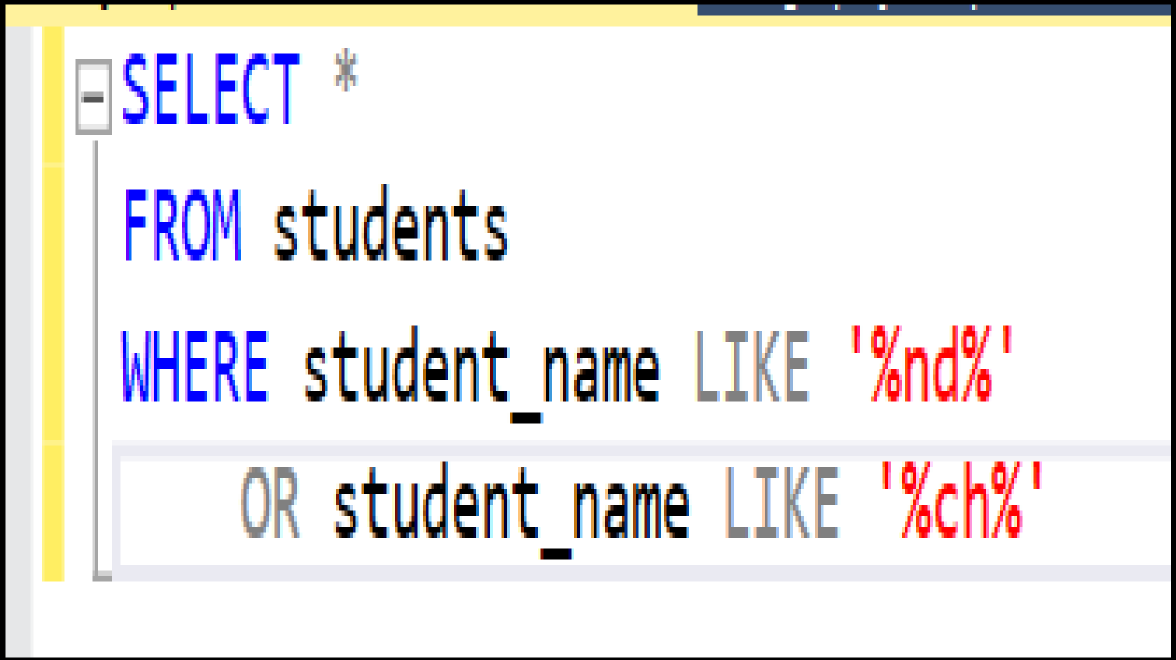


```
SELECT student_name
FROM students
WHERE age BETWEEN 18
        AND 20;
```

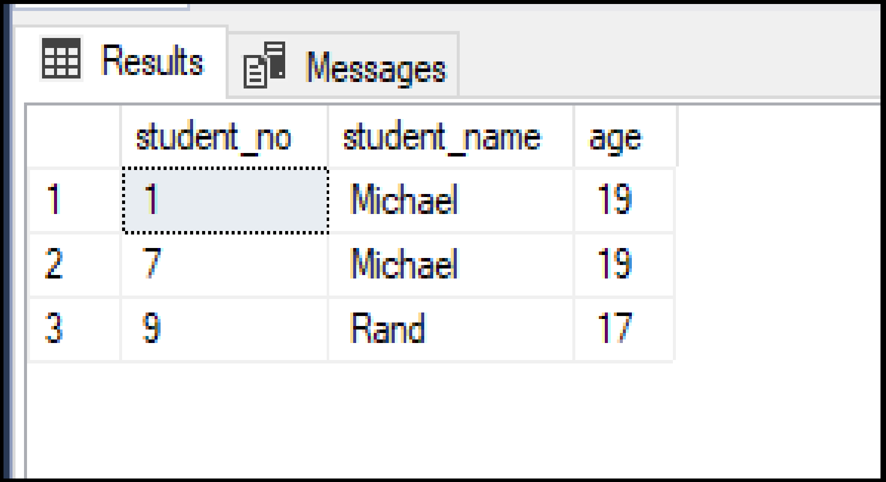


	student_name
1	Michael
2	Doug
3	Pete
4	Ralph
5	Michael
6	Jack
7	Sylvia

Q 2; WRITE A QUERY TO DISPLAY ALL OF THOSE STUDENTS THAT CONTAIN THE LETTERS "CH" IN THEIR NAME OR THEIR NAME ENDS WITH LETTERS "ND".

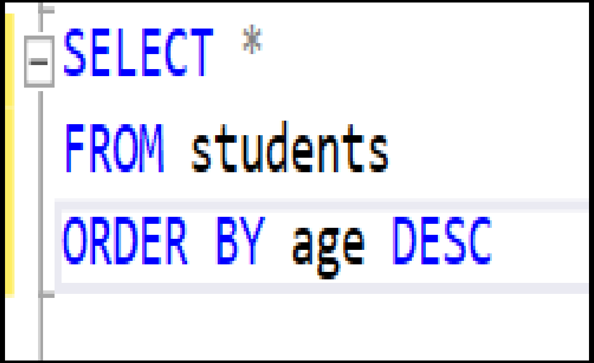


```
SELECT *
FROM students
WHERE student_name LIKE '%nd%'
    OR student_name LIKE '%ch%'
```

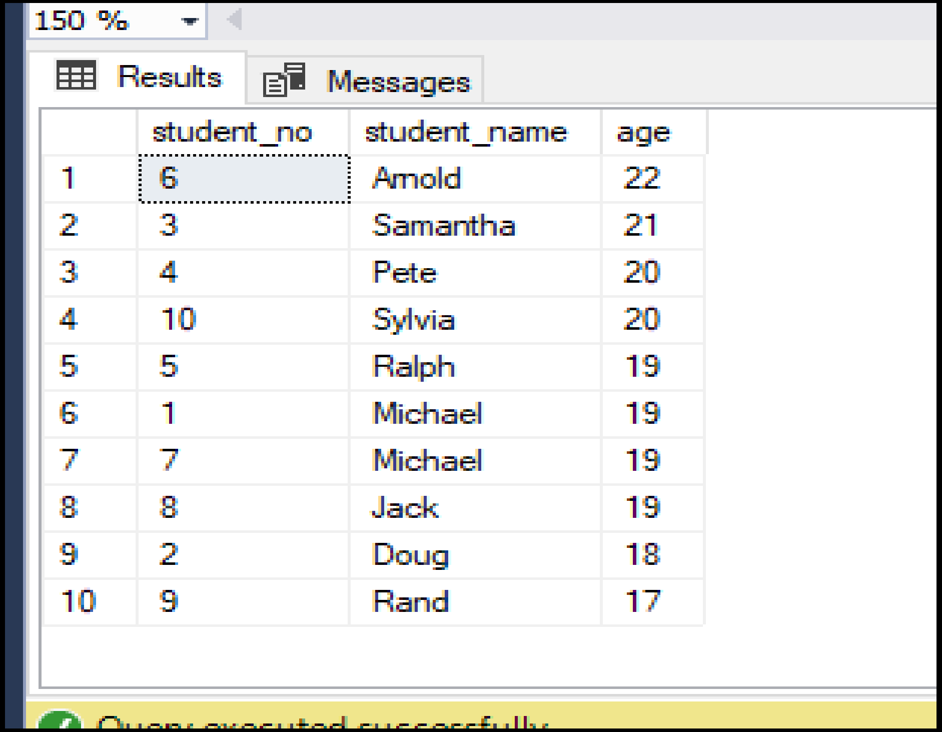


	student_no	student_name	age
1	1	Michael	19
2	7	Michael	19
3	9	Rand	17

Q 4; WRITE A QUERY THAT LISTS THE NAMES OF STUDENTS SORTED BY THEIR AGE FROM LARGEST TO SMALLEST



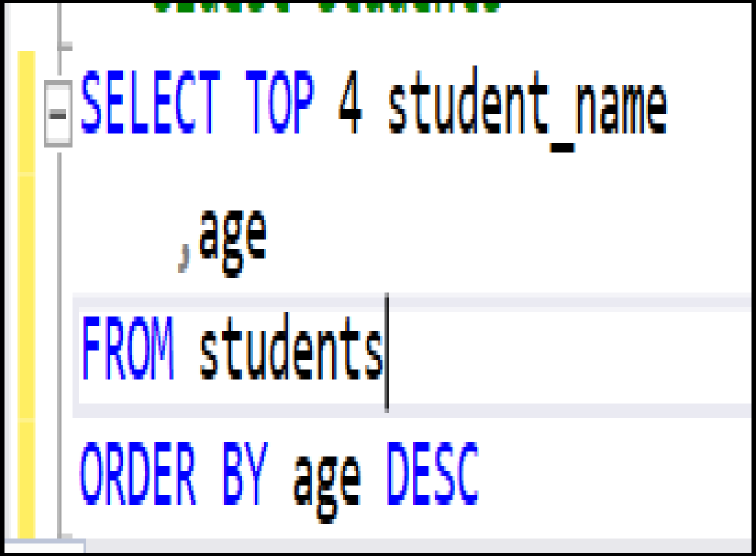
```sql
SELECT *
FROM students
ORDER BY age DESC
```



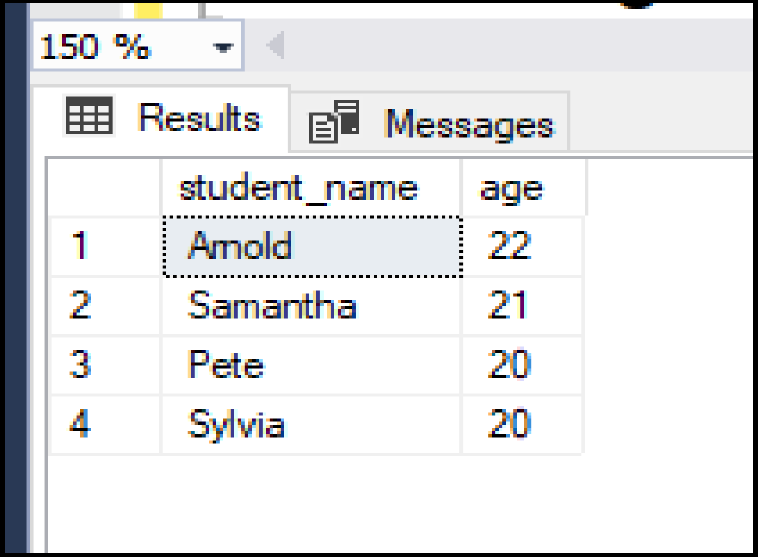
	student_no	student_name	age
1	6	Arnold	22
2	3	Samantha	21
3	4	Pete	20
4	10	Sylvia	20
5	5	Ralph	19
6	1	Michael	19
7	7	Michael	19
8	8	Jack	19
9	2	Doug	18
10	9	Rand	17

Query executed successfully.

Q 5; WRITE A QUERY THAT DISPLAYS THE NAMES AND AGES OF THE TOP 4 OLDEST STUDENTS

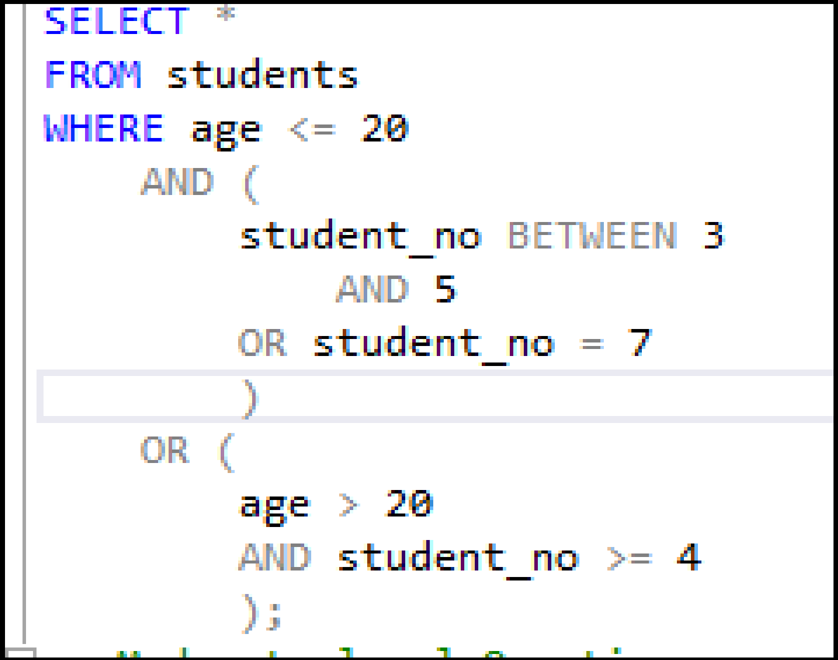


```sql
SELECT TOP 4 student_name
      ,age
FROM students
ORDER BY age DESC
```

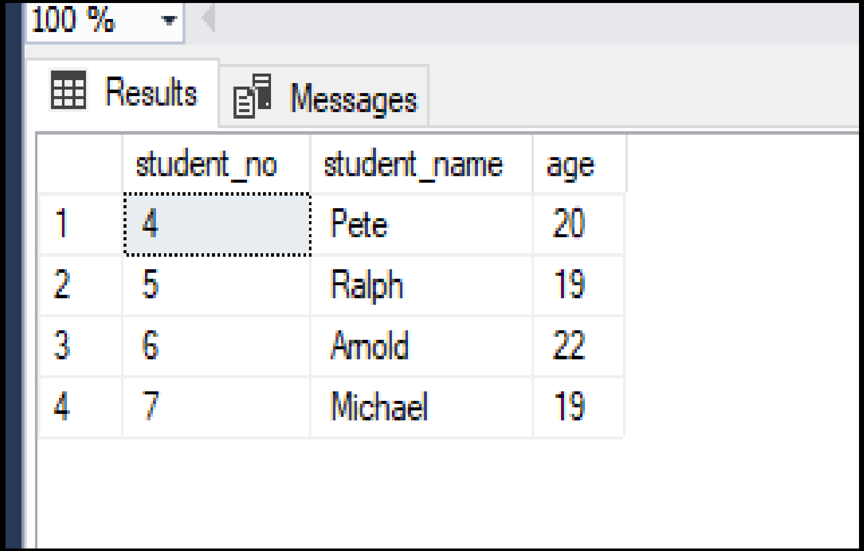


	student_name	age
1	Arnold	22
2	Samantha	21
3	Pete	20
4	Sylvia	20

Q6; WRITE A QUERY THAT RETURNS STUDENTS BASE ON THE FOLLOWING CRITERIA
THE STUDENT MUST NOT BE OLDER THAN AGE 20
IF THEIR STUDENT NO IS EITHER 3 AND 5 OR THEIR STUDENT NO IS 7.
YOUR QUERY SHOULD ALSO RETURN STUDENTS OLDER THAN AGE 20 BUT IN THAT
CASE THEY MUST HAVE A STUDENT NO THAT IS AT LEAST 4.



```
SELECT *
FROM students
WHERE age <= 20
    AND (
        student_no BETWEEN 3
            AND 5
        OR student_no = 7
        )
    OR (
        age > 20
        AND student_no >= 4
        );
```



	student_no	student_name	age
1	4	Pete	20
2	5	Ralph	19
3	6	Arnold	22
4	7	Michael	19

- MODERATE LEVEL QUESTIONS;

1. Write a query that finds students who do not take CS180.

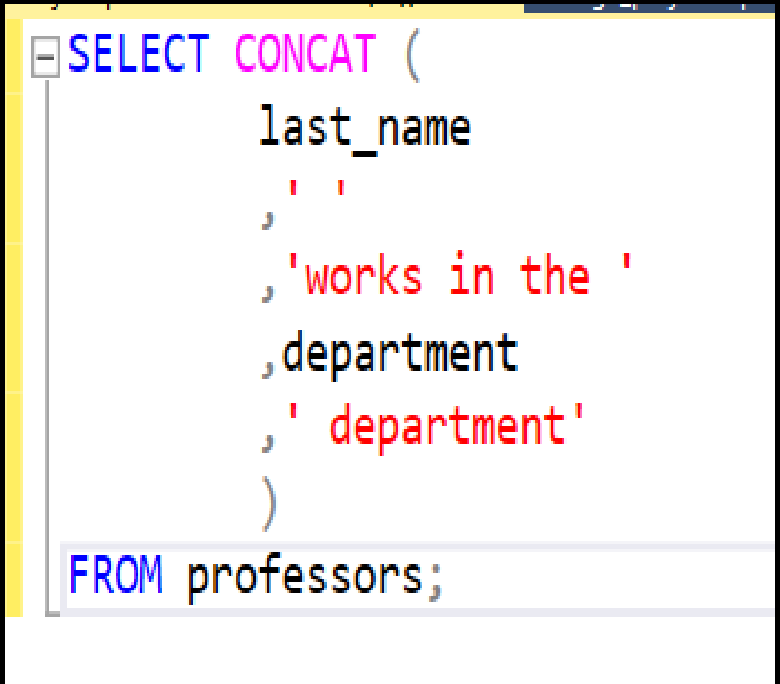
2. Write a query to find students who take CS110 or CS107 but not both.

3. Write a query to find students who take CS220 and no other courses.

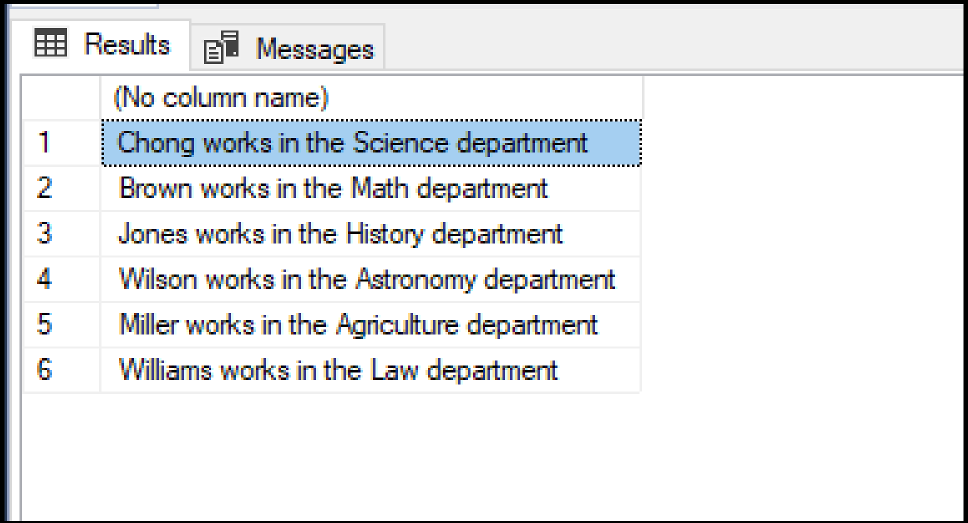
4. Write a query that finds those students who take at most 2 courses. Your query should exclude students that don't take any courses as well as those that take more than 2 course.

5. Write a query to find students who are older than at most two other students.

Q 1; WRITE A QUERY AGAINST THE PROFESSORS TABLE THAT CAN OUTPUT THE FOLLOWING IN THE RESULT
" CHONG WORKS IN THE SCIENCE DEPARTMENT”

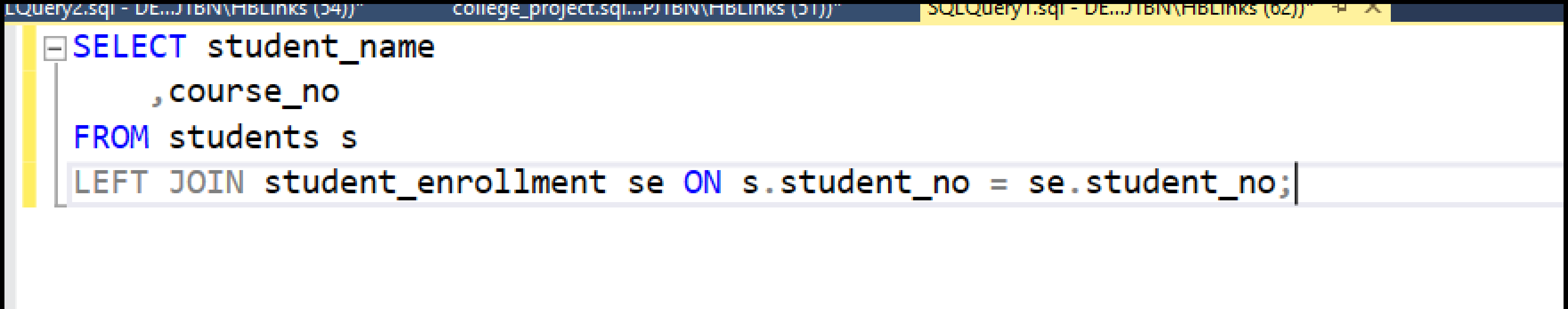


```
SELECT CONCAT (
        last_name
        ,' '
        ,'works in the '
        ,department
        ,' department'
        )
FROM professors;
```

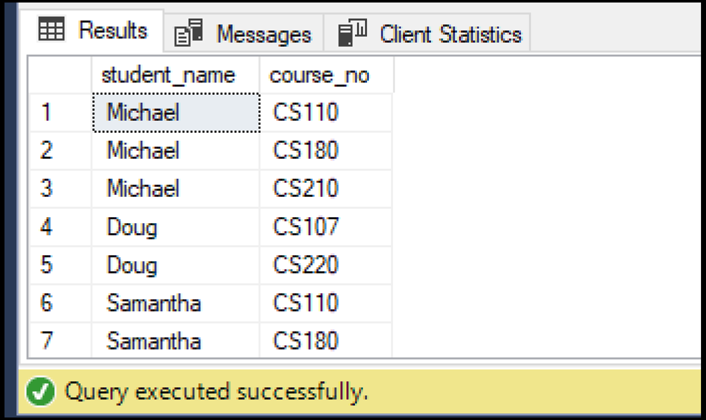


	(No column name)
1	Chong works in the Science department
2	Brown works in the Math department
3	Jones works in the History department
4	Wilson works in the Astronomy department
5	Miller works in the Agriculture department
6	Williams works in the Law department

Q; WRITE A QUERY THAT RETURNS ALL OF THE STUDENTS AS WELL AS ANY COURSES THEY MAY OR MAY NOT BE TAKING;


LQuery2.sql - DE...JTBN\HBLinks (54))*
college_project.sql...PJTBN\HBLinks (51))*
SQLQuery1.sql - DE...JTBN\HBLinks (62))*

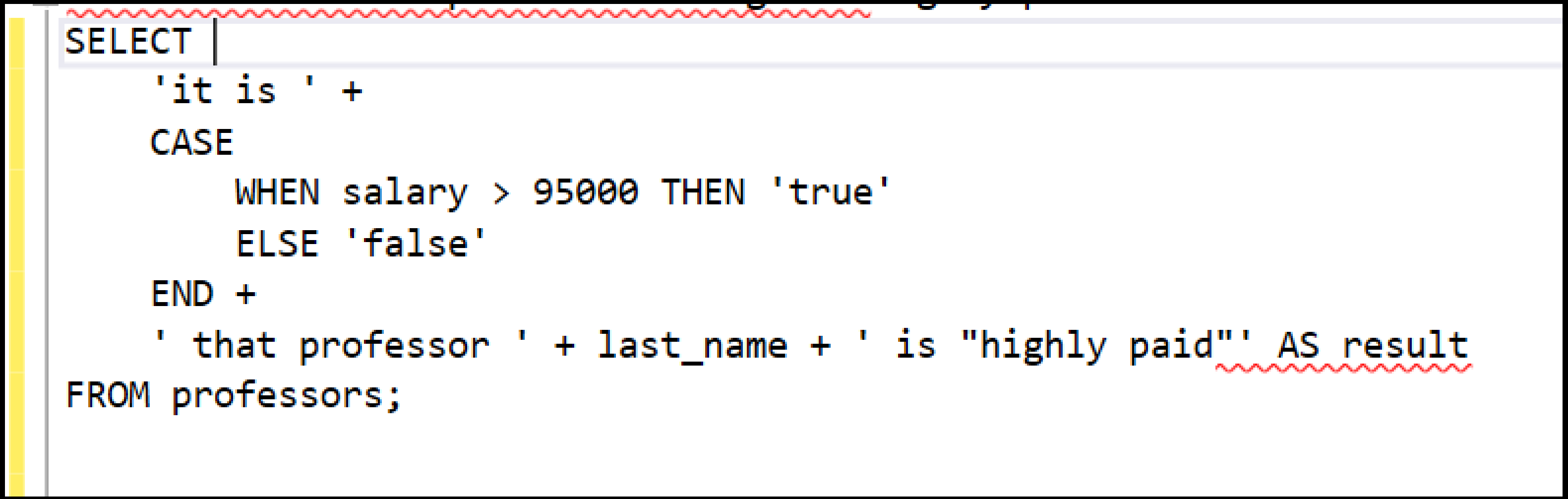
```sql
SELECT student_name
      ,course_no
FROM students s
LEFT JOIN student_enrollment se ON s.student_no = se.student_no;
```



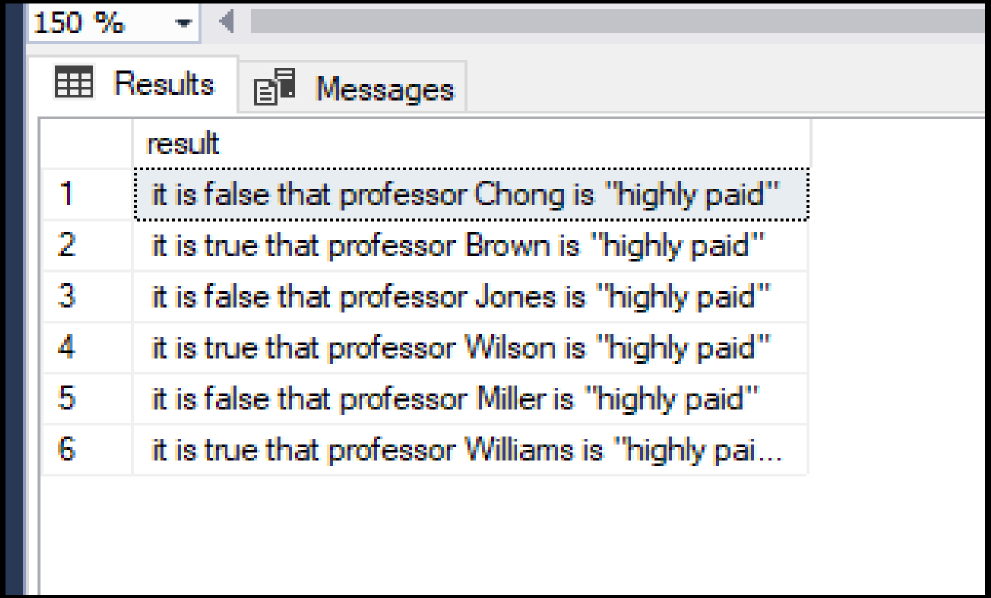
	student_name	course_no
1	Michael	CS110
2	Michael	CS180
3	Michael	CS210
4	Doug	CS107
5	Doug	CS220
6	Samantha	CS110
7	Samantha	CS180

Query executed successfully.

Q 2; Write a SQL query against the professors table that would return the following result
"it is false that professor chong is "highly paid"

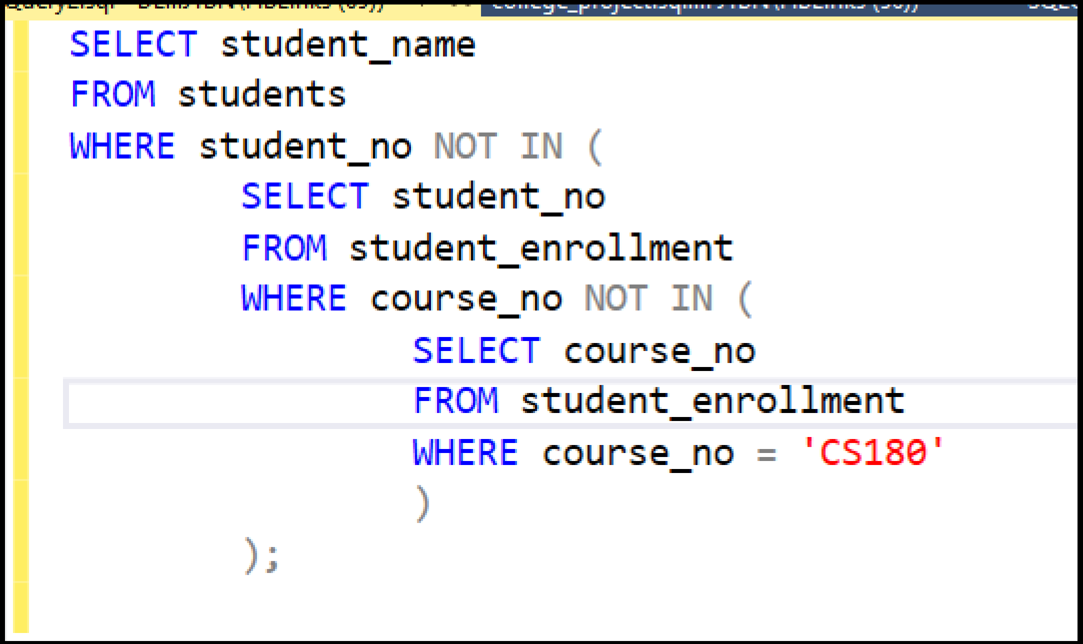


```
SELECT
    'it is ' +
    CASE
        WHEN salary > 95000 THEN 'true'
        ELSE 'false'
    END +
    ' that professor ' + last_name + ' is "highly paid"' AS result
FROM professors;
```

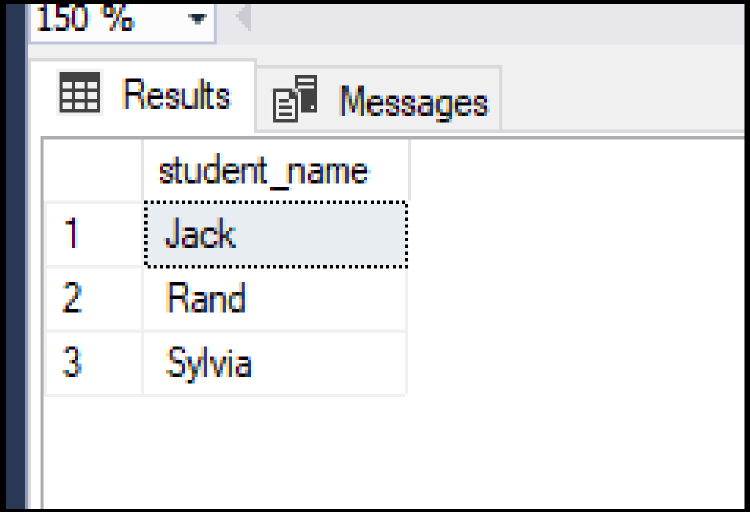


	result
1	it is false that professor Chong is "highly paid"
2	it is true that professor Brown is "highly paid"
3	it is false that professor Jones is "highly paid"
4	it is true that professor Wilson is "highly paid"
5	it is false that professor Miller is "highly paid"
6	it is true that professor Williams is "highly pai...

Q; WRITE A QUERY THAT FINDS THOSE STUDENTS WHO DO NOT TAKE CS180 COURSE

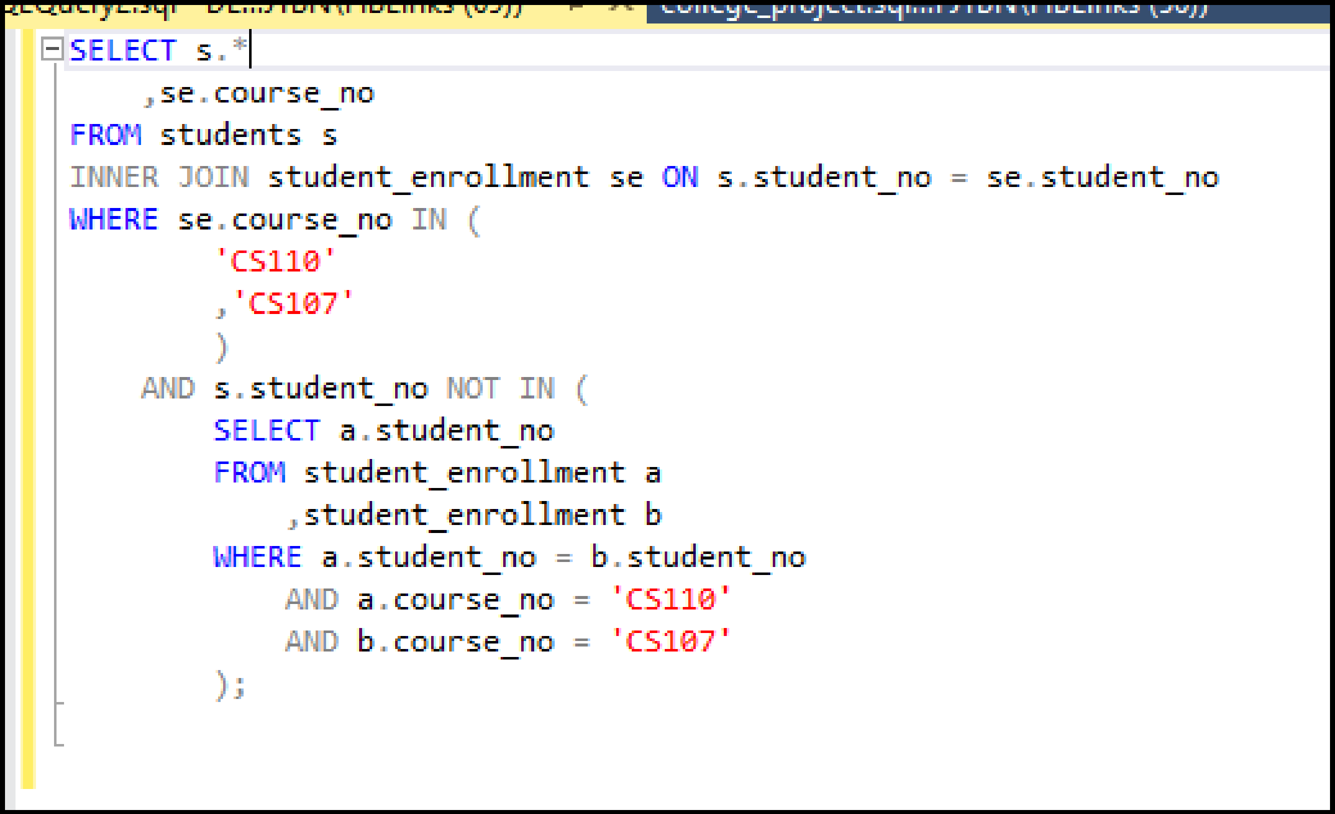


```
SELECT student_name
FROM students
WHERE student_no NOT IN (
        SELECT student_no
        FROM student_enrollment
        WHERE course_no NOT IN (
                SELECT course_no
                FROM student_enrollment
                WHERE course_no = 'CS180'
                )
        );
```

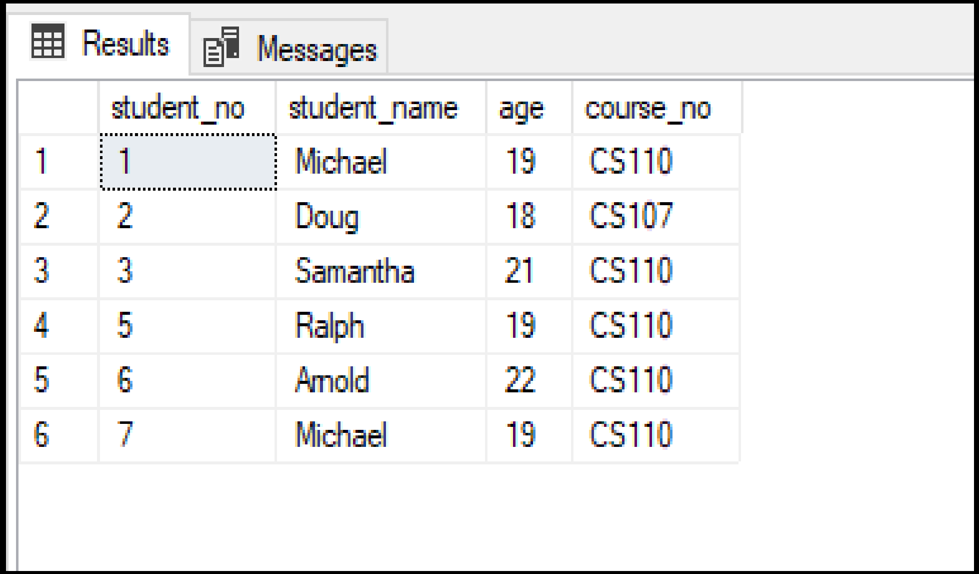


	student_name
1	Jack
2	Rand
3	Sylvia

Q; WRITE A QUERY TO FIND THE STUDENTS WHO TAKE CS110 OR CS107 BUT NOT TAKE BOTH

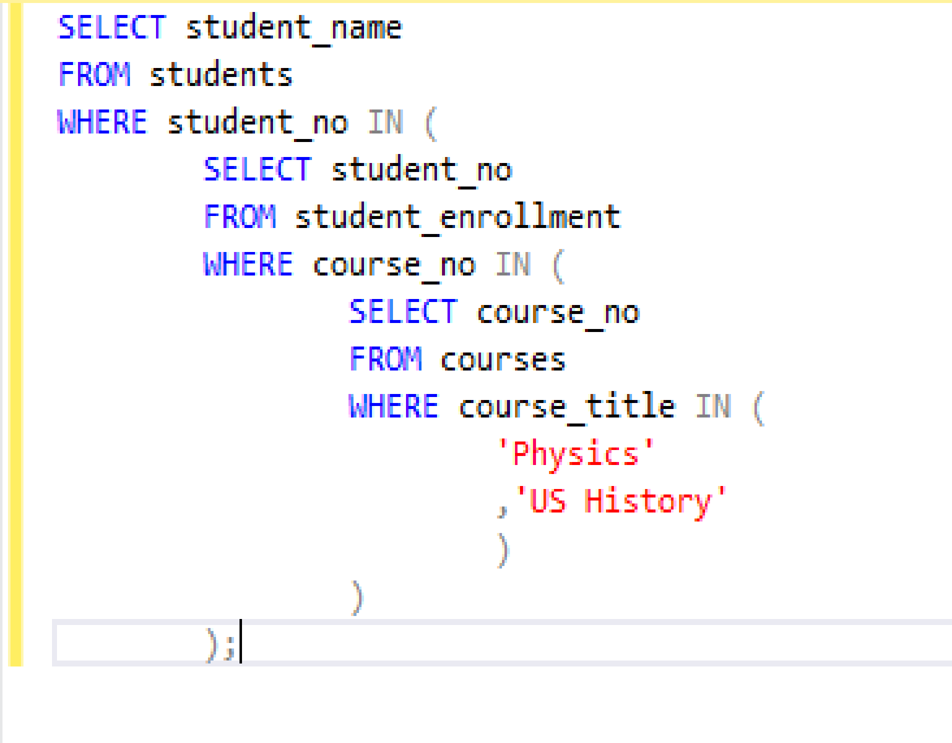


```sql
SELECT s.*
     ,se.course_no
FROM students s
INNER JOIN student_enrollment se ON s.student_no = se.student_no
WHERE se.course_no IN (
        'CS110'
        ,'CS107'
        )
    AND s.student_no NOT IN (
        SELECT a.student_no
        FROM student_enrollment a
            ,student_enrollment b
        WHERE a.student_no = b.student_no
            AND a.course_no = 'CS110'
            AND b.course_no = 'CS107'
        );
```

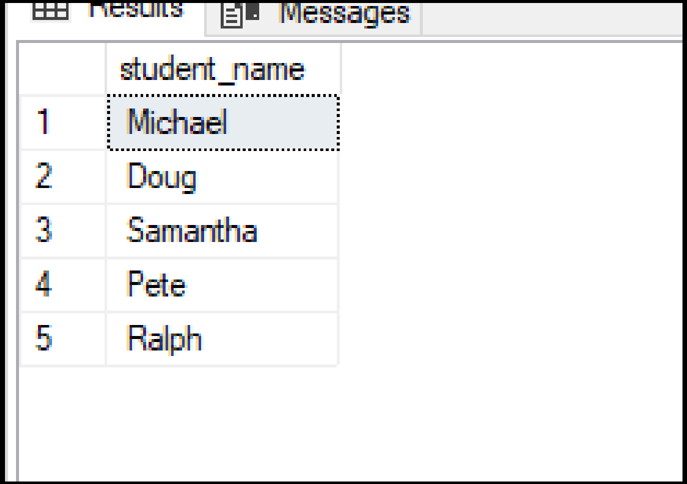


	student_no	student_name	age	course_no
1	1	Michael	19	CS110
2	2	Doug	18	CS107
3	3	Samantha	21	CS110
4	5	Ralph	19	CS110
5	6	Arnold	22	CS110
6	7	Michael	19	CS110

Q; USING SUBQUERIES ONLY, WRITE A SQL STATEMENT THAT RETURNS THE NAMES OF THOSE STUDENTS THAT ARE TAKING THE COURSES PHYSICS AND US HISTORY.

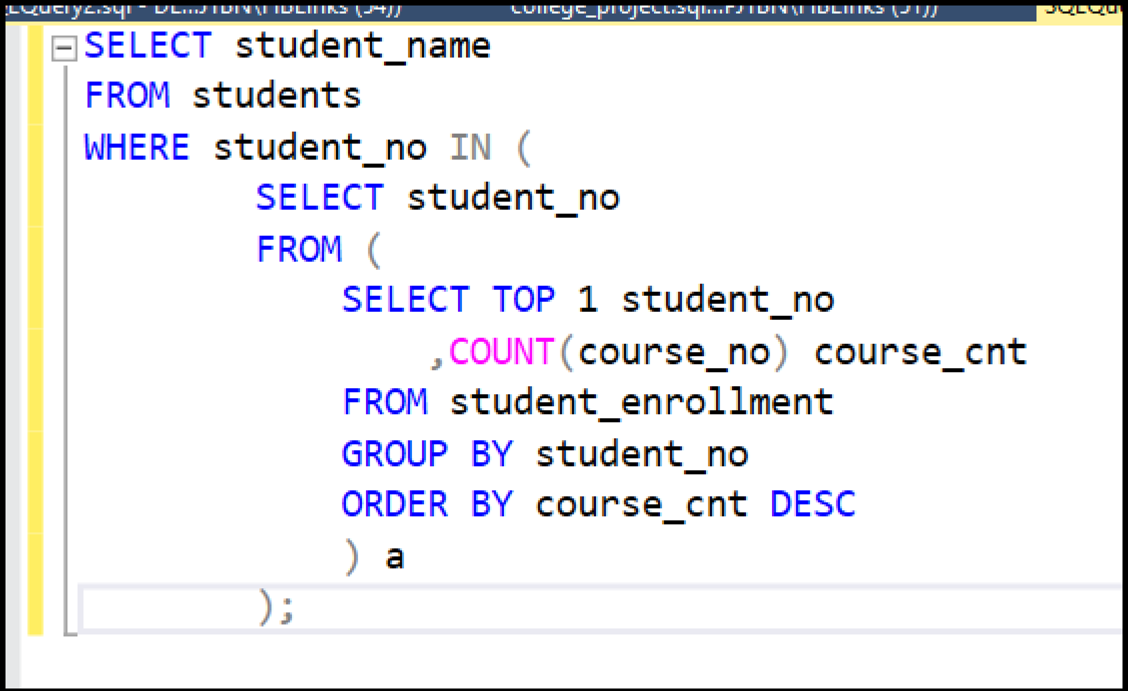


```sql
SELECT student_name
FROM students
WHERE student_no IN (
        SELECT student_no
        FROM student_enrollment
        WHERE course_no IN (
                SELECT course_no
                FROM courses
                WHERE course_title IN (
                        'Physics'
                        ,'US History'
                        )
                )
        );
```

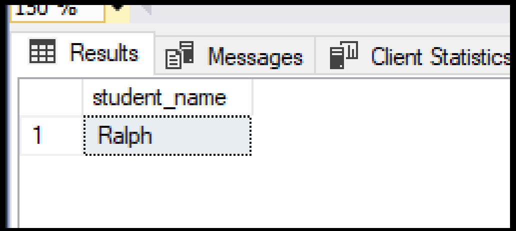


```
   student_name
1  Michael
2  Doug
3  Samantha
4  Pete
5  Ralph
```

Q; THE NAME OF THE STUDENT THAT TAKES THE HIGHEST NUMBER OF COURSES

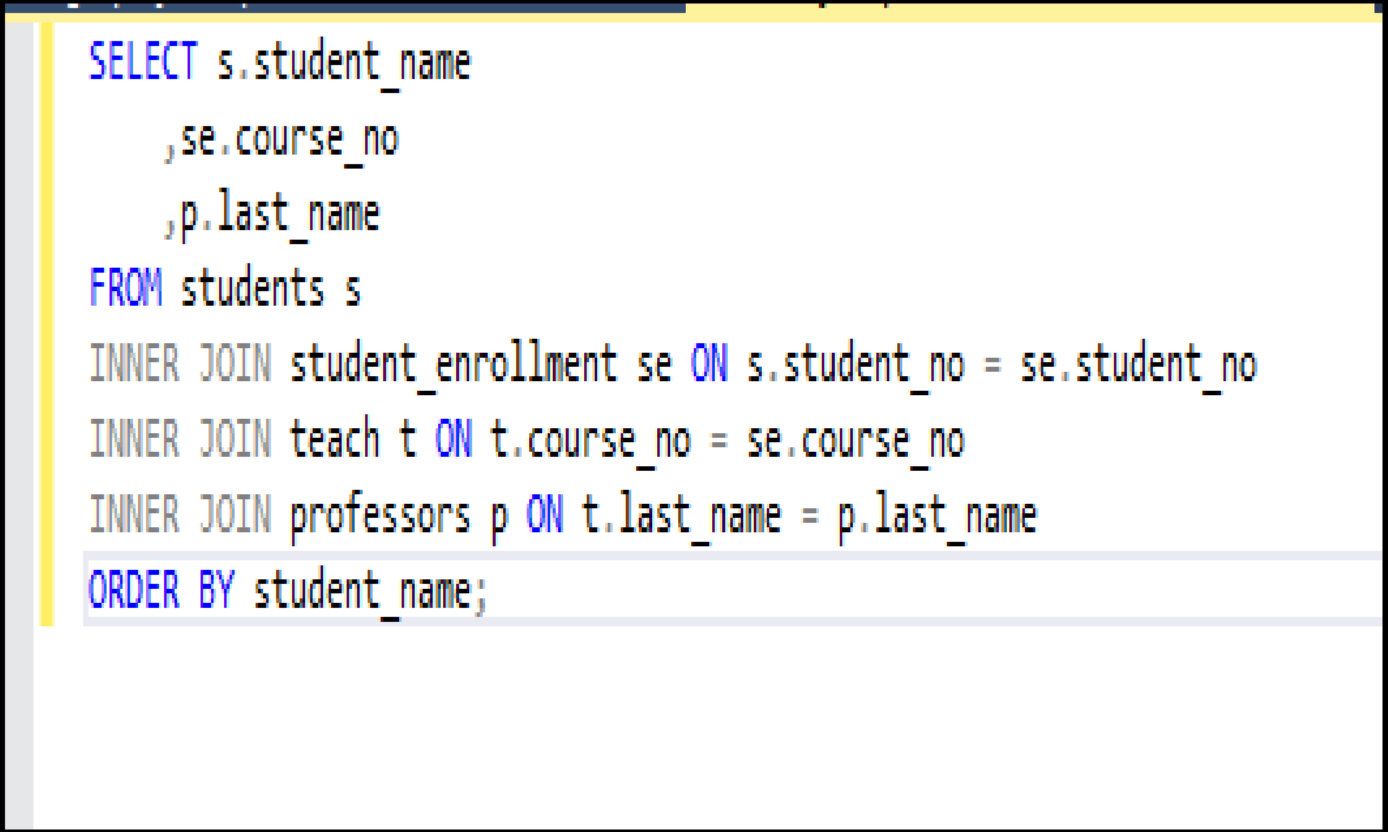


```sql
SELECT student_name
FROM students
WHERE student_no IN (
        SELECT student_no
        FROM (
            SELECT TOP 1 student_no
                ,COUNT(course_no) course_cnt
            FROM student_enrollment
            GROUP BY student_no
            ORDER BY course_cnt DESC
            ) a
        );
```

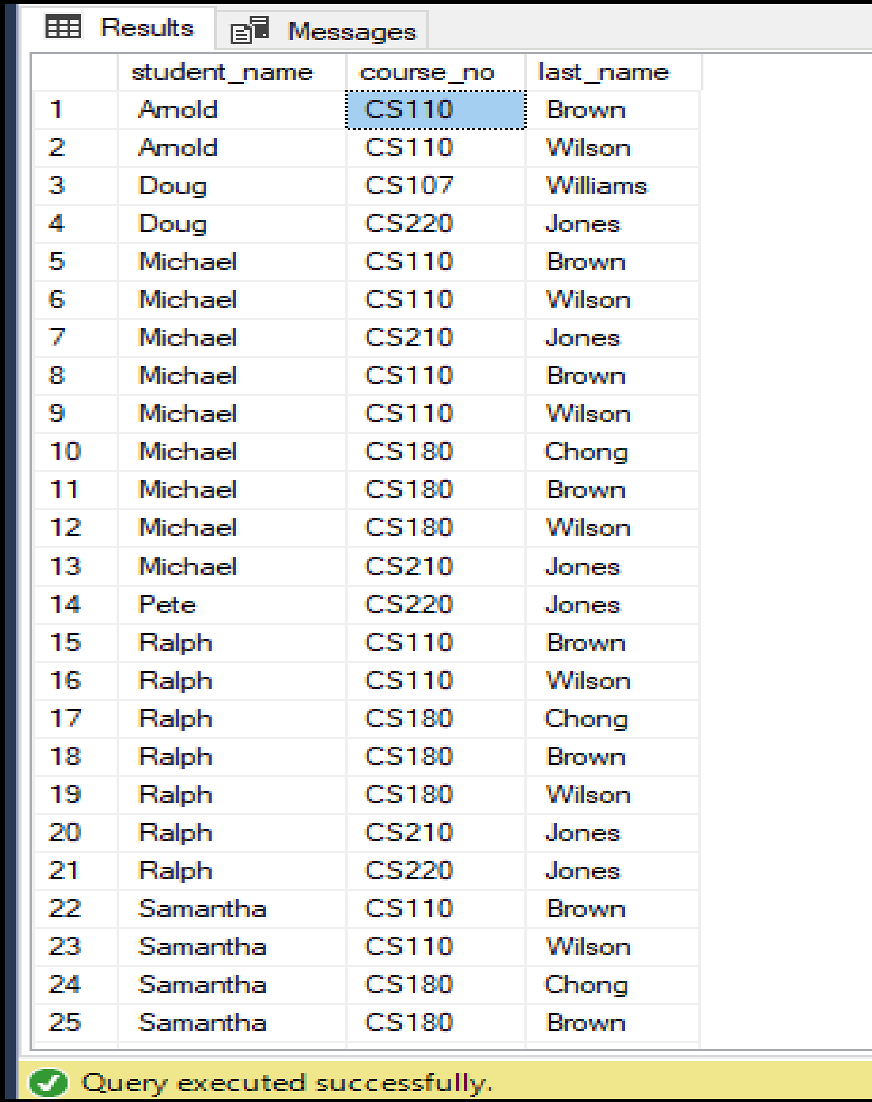


```
   student_name
1  Ralph
```

Q; WRITE A QUERY THAT SHOWS THE STUDENT'S NAME THE COURSES THE STUDENT IS TAKING AND THE PROFESSORS THAT TEACH THAT COURSE.



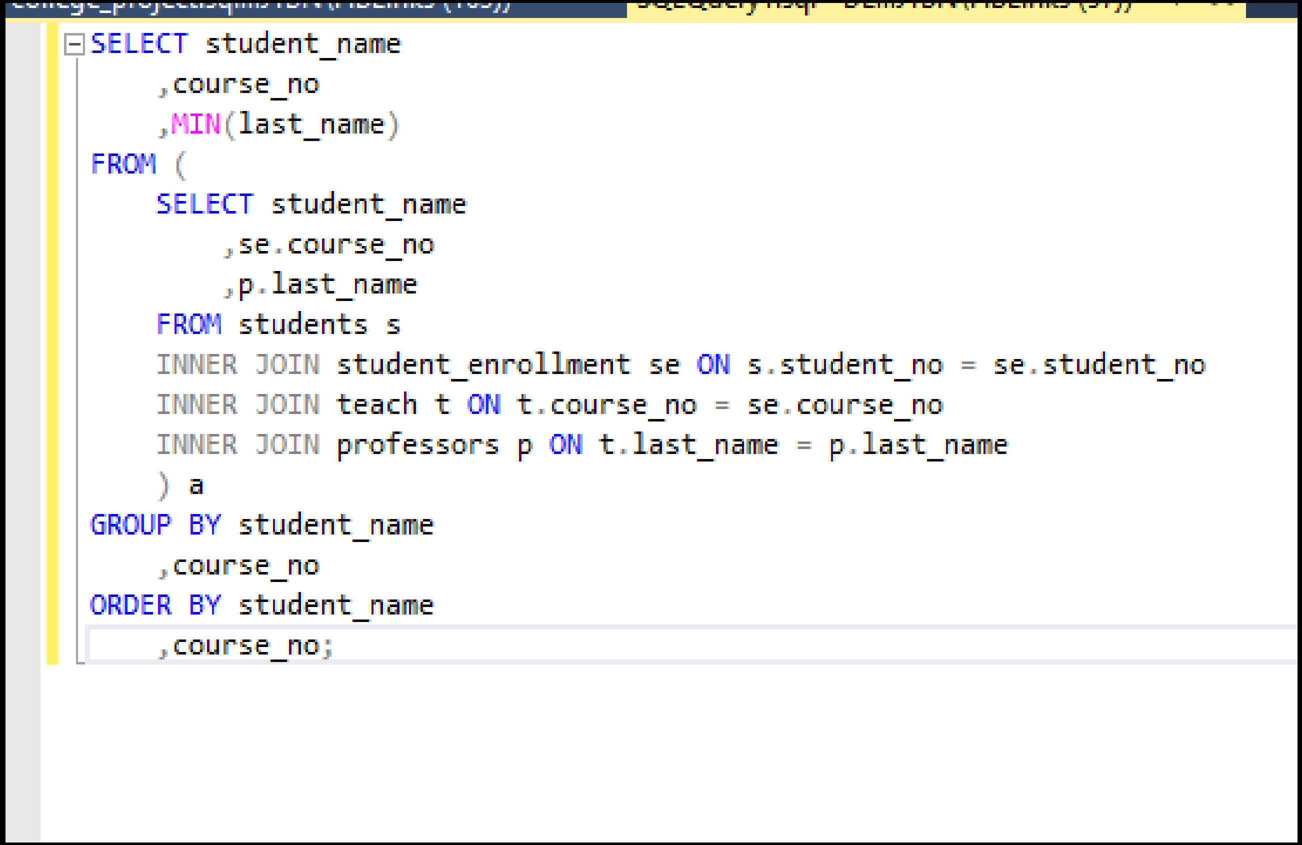
```
SELECT s.student_name
    ,se.course_no
    ,p.last_name
FROM students s
INNER JOIN student_enrollment se ON s.student_no = se.student_no
INNER JOIN teach t ON t.course_no = se.course_no
INNER JOIN professors p ON t.last_name = p.last_name
ORDER BY student_name;
```



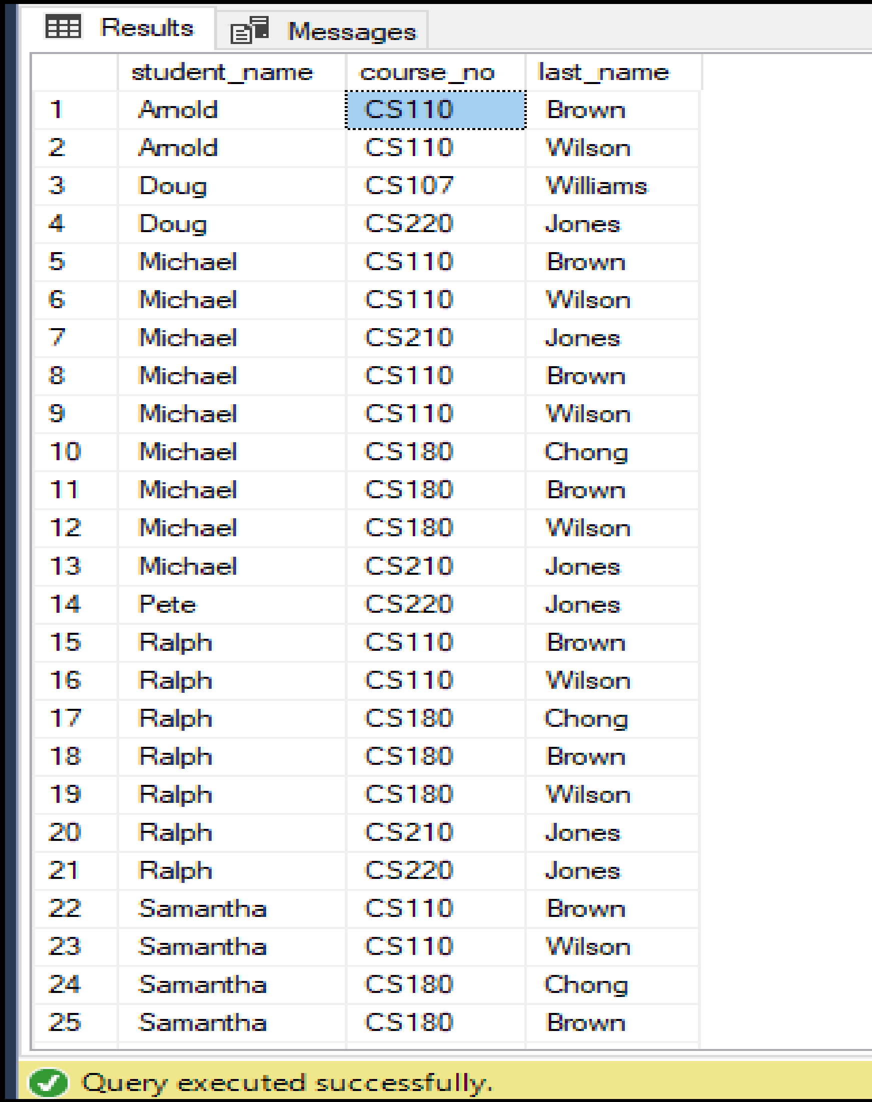
	student_name	course_no	last_name
1	Arnold	CS110	Brown
2	Arnold	CS110	Wilson
3	Doug	CS107	Williams
4	Doug	CS220	Jones
5	Michael	CS110	Brown
6	Michael	CS110	Wilson
7	Michael	CS210	Jones
8	Michael	CS110	Brown
9	Michael	CS110	Wilson
10	Michael	CS180	Chong
11	Michael	CS180	Brown
12	Michael	CS180	Wilson
13	Michael	CS210	Jones
14	Pete	CS220	Jones
15	Ralph	CS110	Brown
16	Ralph	CS110	Wilson
17	Ralph	CS180	Chong
18	Ralph	CS180	Brown
19	Ralph	CS180	Wilson
20	Ralph	CS210	Jones
21	Ralph	CS220	Jones
22	Samantha	CS110	Brown
23	Samantha	CS110	Wilson
24	Samantha	CS180	Chong
25	Samantha	CS180	Brown

✓ Query executed successfully.

SECOND METHOD



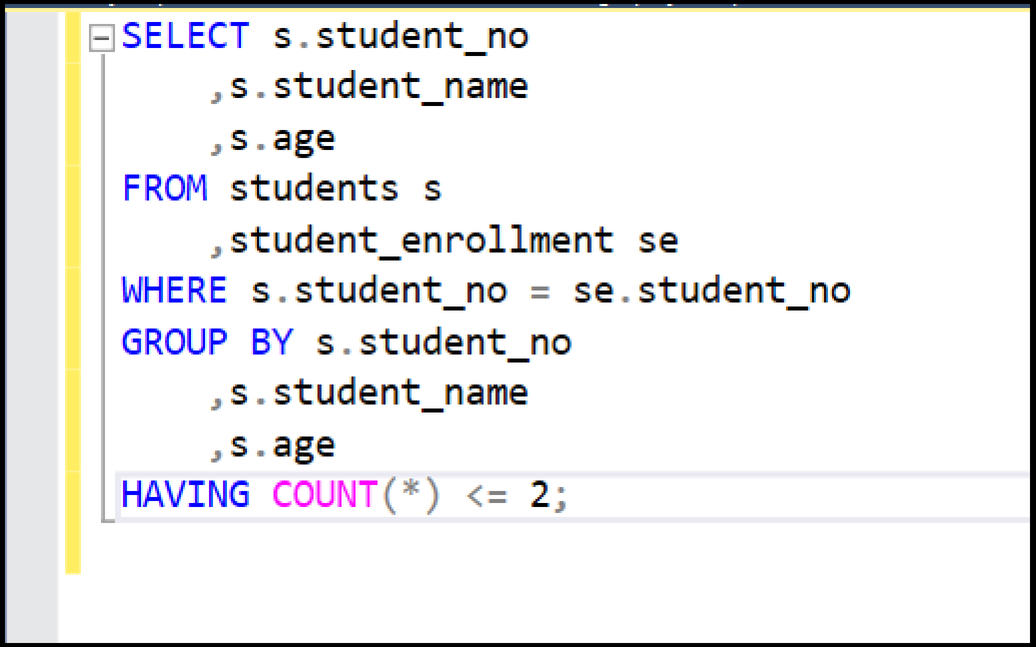
```sql
SELECT student_name
    ,course_no
    ,MIN(last_name)
FROM (
    SELECT student_name
        ,se.course_no
        ,p.last_name
    FROM students s
    INNER JOIN student_enrollment se ON s.student_no = se.student_no
    INNER JOIN teach t ON t.course_no = se.course_no
    INNER JOIN professors p ON t.last_name = p.last_name
    ) a
GROUP BY student_name
    ,course_no
ORDER BY student_name
    ,course_no;
```



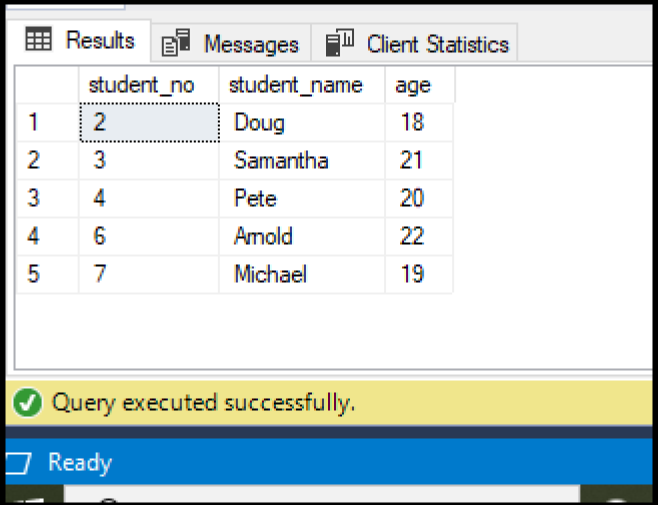
	student_name	course_no	last_name
1	Arnold	CS110	Brown
2	Arnold	CS110	Wilson
3	Doug	CS107	Williams
4	Doug	CS220	Jones
5	Michael	CS110	Brown
6	Michael	CS110	Wilson
7	Michael	CS210	Jones
8	Michael	CS110	Brown
9	Michael	CS110	Wilson
10	Michael	CS180	Chong
11	Michael	CS180	Brown
12	Michael	CS180	Wilson
13	Michael	CS210	Jones
14	Pete	CS220	Jones
15	Ralph	CS110	Brown
16	Ralph	CS110	Wilson
17	Ralph	CS180	Chong
18	Ralph	CS180	Brown
19	Ralph	CS180	Wilson
20	Ralph	CS210	Jones
21	Ralph	CS220	Jones
22	Samantha	CS110	Brown
23	Samantha	CS110	Wilson
24	Samantha	CS180	Chong
25	Samantha	CS180	Brown

✓ Query executed successfully.

Q; WRITE A QUERY THAT FINDS THOSE STUDENTS WHO TAKE AT MOST 2 COURSES. YOUR QUERY SHOULD EXCLUDE STUDENTS THAT DON'T TAKE ANY COURSES AS WELL AS THOSE THAT TAKE MORE THAN 2 COURSES.



```
SELECT s.student_no
     ,s.student_name
     ,s.age
FROM students s
     ,student_enrollment se
WHERE s.student_no = se.student_no
GROUP BY s.student_no
     ,s.student_name
     ,s.age
HAVING COUNT(*) <= 2;
```

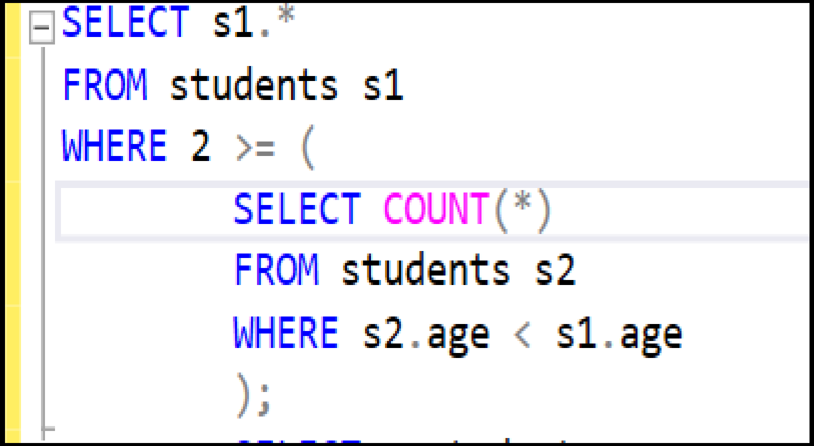


	student_no	student_name	age
1	2	Doug	18
2	3	Samantha	21
3	4	Pete	20
4	6	Arnold	22
5	7	Michael	19

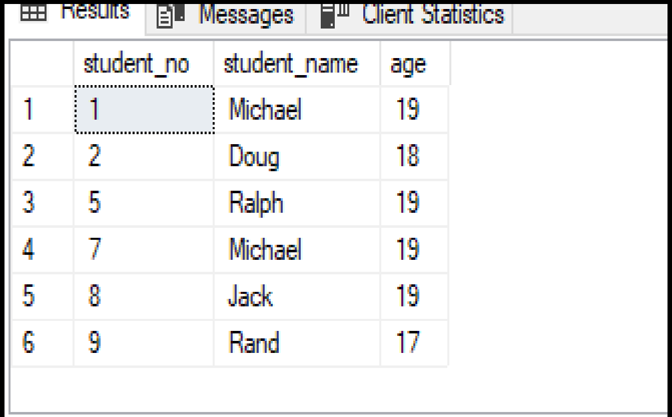
Query executed successfully.

Ready

WRITE A QUERY TO FIND STUDENTS WHO ARE OLDER THAN AT MOST TWO OTHER STUDENTS.

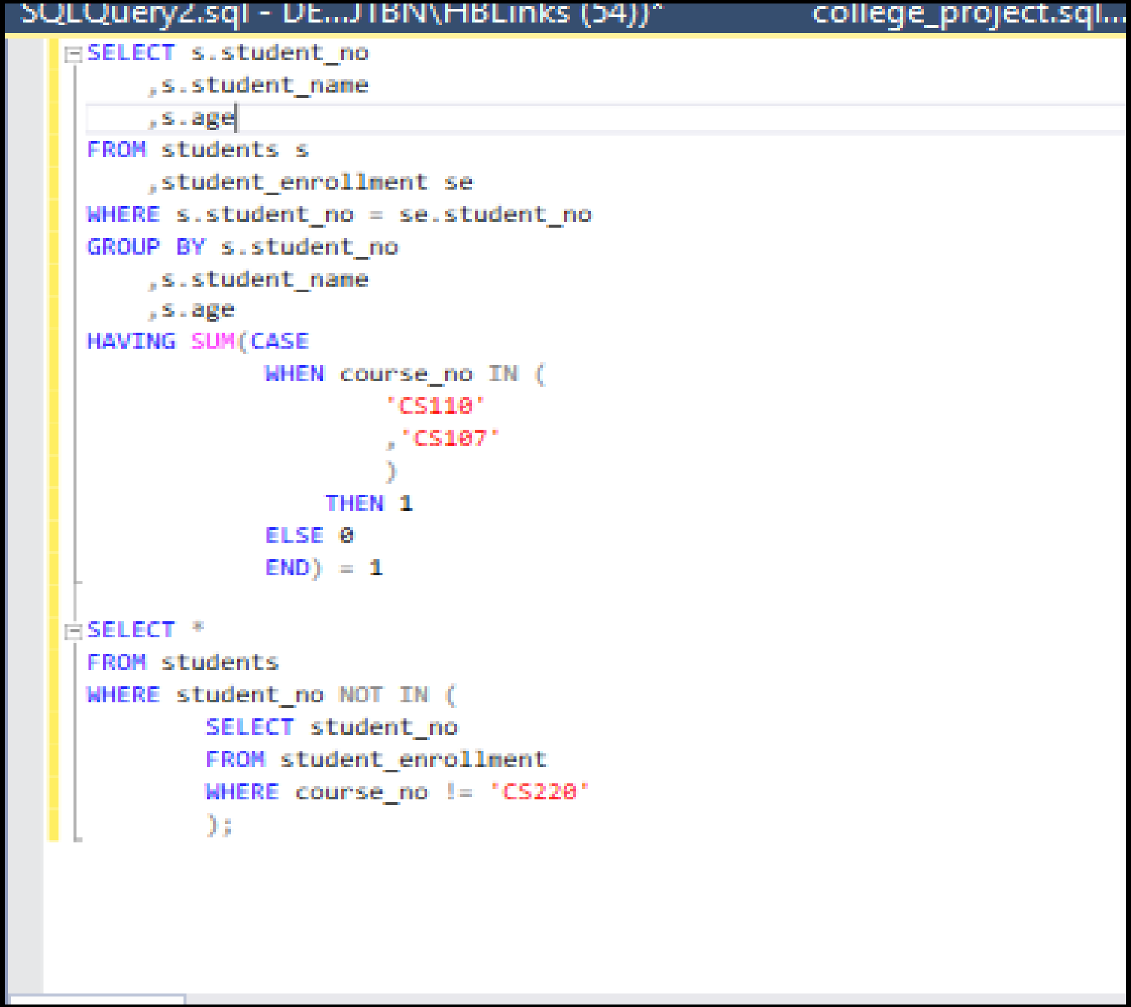


```sql
SELECT s1.*
FROM students s1
WHERE 2 >= (
        SELECT COUNT(*)
        FROM students s2
        WHERE s2.age < s1.age
        );
```



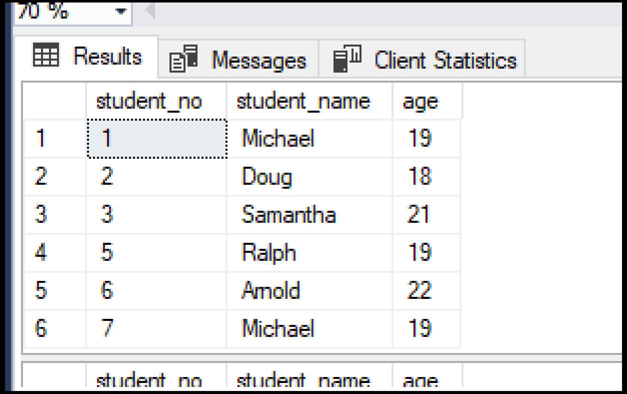
	student_no	student_name	age
1	1	Michael	19
2	2	Doug	18
3	5	Ralph	19
4	7	Michael	19
5	8	Jack	19
6	9	Rand	17

WRITE A QUERY TO FIND STUDENTS WHO TAKE CS110 OR CS107 BUT NOT BOTH.



```sql
SELECT s.student_no
    ,s.student_name
    ,s.age
FROM students s
    ,student_enrollment se
WHERE s.student_no = se.student_no
GROUP BY s.student_no
    ,s.student_name
    ,s.age
HAVING SUM(CASE
            WHEN course_no IN (
                    'CS110'
                    ,'CS107'
                    )
                THEN 1
            ELSE 0
            END) = 1

SELECT *
FROM students
WHERE student_no NOT IN (
        SELECT student_no
        FROM student_enrollment
        WHERE course_no != 'CS220'
        );
```



	student_no	student_name	age
1	1	Michael	19
2	2	Doug	18
3	3	Samantha	21
4	5	Ralph	19
5	6	Arnold	22
6	7	Michael	19

	student_no	student_name	age