Challenging Puzzles For the Brave

In this section you'll be working with the tables you created in assignment 1. The problems in this section are challenging! Before starting these problems, you should have solved all of the previous assignments in the course. Getting the questions right in this section means you're a SQL Expert!

The questions in this section will require you to use the tables you created in assignment 1. Those tables are: students, student_enrollment, courses, professors and teach. Remember all assignments are mandatory, especially assignment 1. If you completed assignment 1, you should have these tables in your database already. If have not completed all of the previous assignments in this course, this section will not be feasible for you. Go back and complete all other problems and then come back here for a good challenge!

Questions for this assignment

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.

Do not scroll past here without trying out the assignment yourself

Instructor Solutions for this assignment

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

You may have thought about the following query at first, but this is not correct:

```
SELECT * FROM students

WHERE student_no IN (SELECT student_no

                     FROM student_enrollment

                     WHERE course_no != 'CS180')

                    ORDER BY student_name
```

The above query is incorrect because it does not answer the question "Who does not take CS180?". Instead, it answers the question "Who takes a course that is not CS180?" The correct result should include students who take no courses as well as students who take courses but none of them CS180.

2 CORRECT ANSWERS BELOW:

Answer A:

```
SELECT * FROM students

WHERE student_no NOT IN (

   SELECT student_no

   FROM student_enrollment

   WHERE course_no = 'CS180'

   );
```

Answer B: Bonus points if you can understand the below solution.

```
SELECT s.student_no, s.student_name, s.age
FROM students s LEFT JOIN student_enrollment se
   ON s.student_no = se.student_no
GROUP BY s.student_no, s.student_name, s.age
HAVING MAX(CASE WHEN se.course_no = 'CS180'
        THEN 1 ELSE 0 END) = 0
```

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

The following query looks promising as a solution but returns the wrong result!

```
SELECT *
FROM students
WHERE student_no IN (SELECT student_no
                FROM student_enrollment
                WHERE course_no != 'CS110'
                AND course_no != 'CS107')
```

2 CORRECT ANSWERS BELOW:

Solution A:

```
SELECT s.*

FROM students s, student_enrollment se

WHERE s.student_no = se.student_no

AND 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')
```

Solution A uses a self join on the student_enrollment table so that those students are narrowed down that take both CS110 and CS107 in the subquery. The outer query filters for those student_no that are not the ones retrieved from the subquery.

Solution B:

```
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 se.course_no IN ('CS110', 'CS107')

    THEN 1 ELSE 0 END ) = 1
```

In solution B, a CASE expression is used with the aggregate SUM function to find students who take either CS110 or CS107, but not both.

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

You may have thought about the below query to solve this problem but this will not give you the correct result:

```
SELECT s.*

FROM students s, student_enrollment se

WHERE s.student_no = se.student_no

AND se.course_no = 'CS220'
```

We want to see those students who only take CS220 and no other course. The above query returns students who take CS220 but these students could also be taking other courses and that is why this query doesn't work.

CORRECT ANSWERS BELOW:

Solution A:

```
SELECT s.*

FROM students s, student_enrollment se

WHERE s.student_no = se.student_no

AND s.student_no NOT IN ( SELECT student_no

                FROM student_enrollment

                WHERE course_no != 'CS220')
```

In Solution A, the subquery returns all students that take a course other than CS220. The outer query gets all students regardless of what course they take. In essence, the subquery finds all students who take a course that is not CS220. The outer query returns all student who are not amongst those that take a course other than CS220. At this point, the only available students are those who actually take CS220 or take nothing at all.

Solution B:

```
SELECT s.*

FROM students s, student_enrollment se1,

    (SELECT student_no FROM student_enrollment

     GROUP BY student_no

     HAVING count(*) = 1) se2

WHERE s.student_no = se1.student_no

AND se1.student_no = se2.student_no

AND se1.course_no = 'CS220'
```

Solution B uses subquery to get those students who take only a single course and since it's in the from clause, it's considered a source of data just like a table. This is also called an inline view if you recall. So the student_no from the inline view is joined with the outer query and we filter for only those students that take the course CS220. So this query returns that one student that takes CS220 and no other course.

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.

SOLUTION:

```
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
```

Use the COUNT function to determine which students take no more than 2 courses. Students that don't take any courses are being excluded anyway because of the join.

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

SOLUTION:

```
SELECT s1.*

FROM students s1

WHERE 2 >= (SELECT count(*)

        FROM students s2

        WHERE s2.age < s1.age)
```

Using the aggregate function COUNT and a correlated subquery as shown in the solution above, you can retrieve the students who are older than zero, one or two other students.