Question: 1978. Employees Whose Manager Left the Company

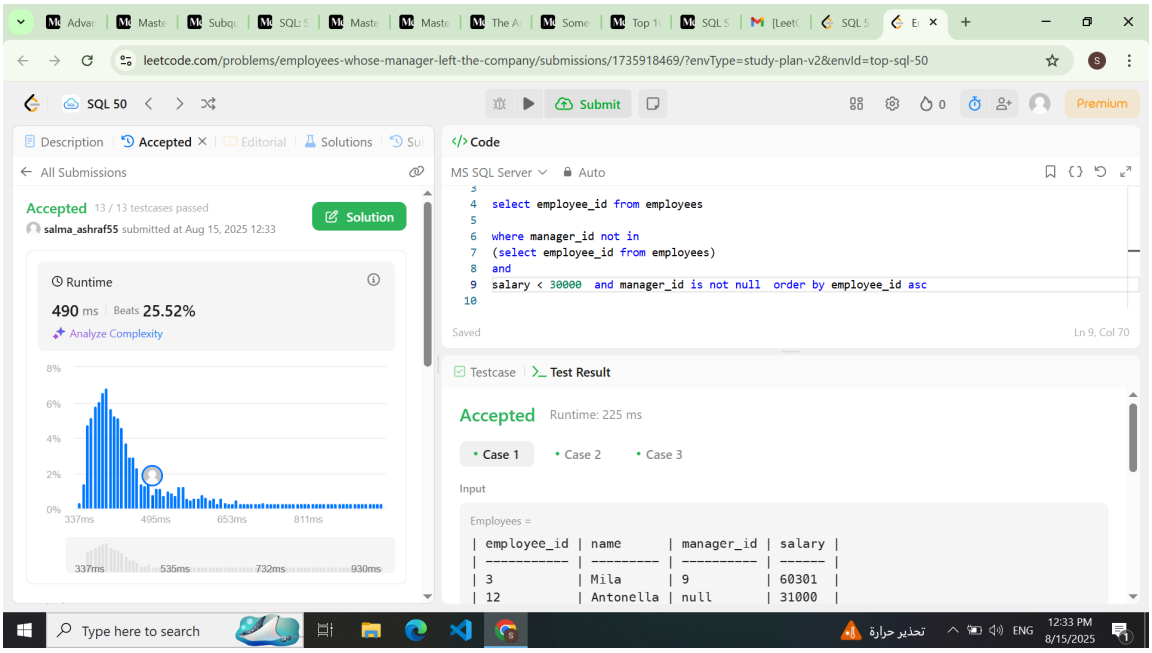
Description:

We need to find the IDs of employees whose managers have left the company.
This means:

- `manager_id` is **not null**.
- `manager_id` does **not** exist in the `employee_id` column (manager is no longer in the company).
- Employee's salary is **less than 30000**.
- Results should be ordered by `employee_id` in ascending order.

SQL Query:

```sql
SELECT employee_id
FROM employees
WHERE manager_id NOT IN (
    SELECT employee_id
    FROM employees
)
AND salary < 30000
AND manager_id IS NOT NULL
ORDER BY employee_id ASC;
```



Question: Exchange Seats

Description:

We need to rearrange students' seats according to the following rules:

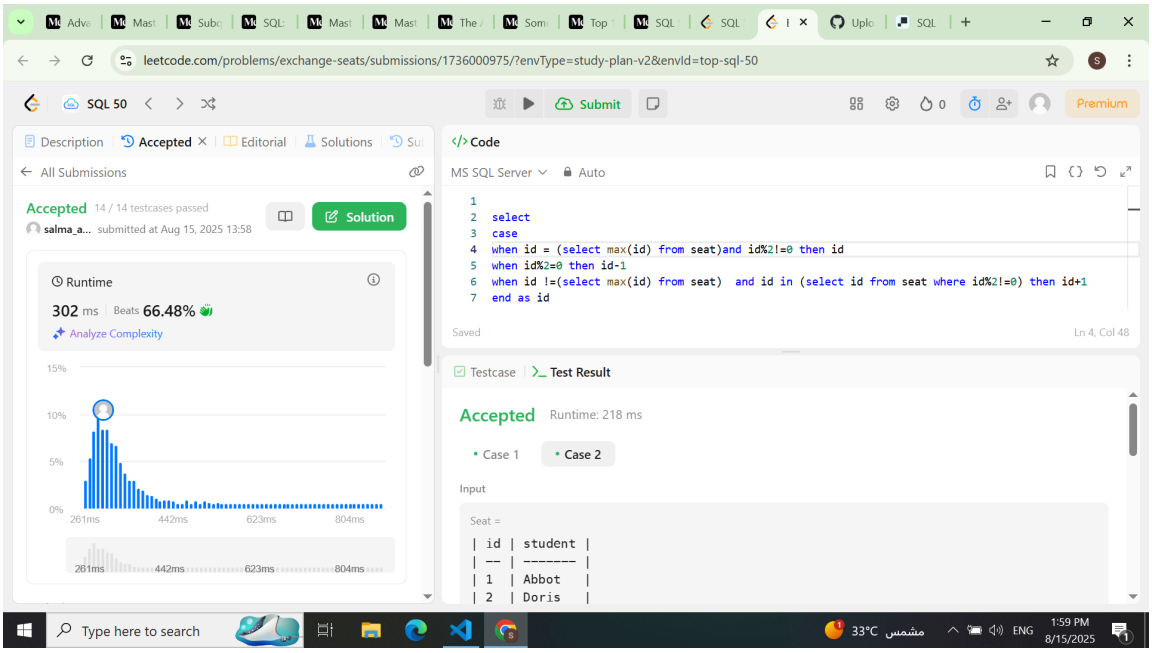
- Swap seats for every two consecutive students.
- If the number of students is odd, the last student keeps their original seat.

Logic:

- If `id` is the maximum `id` and odd → keep the same `id`.
- If `id` is even → subtract 1 (move to the previous seat).
- If `id` is odd and not the maximum `id` → add 1 (move to the next seat).

SQL Query:

```sql
SELECT
    CASE
        WHEN id = (SELECT MAX(id) FROM seat) AND id % 2 != 0 THEN id
        WHEN id % 2 = 0 THEN id - 1
        WHEN id != (SELECT MAX(id) FROM seat)
             AND id IN (SELECT id FROM seat WHERE id % 2 != 0) THEN id + 1
    END AS id,
    student
FROM seat
ORDER BY id;
```



Question: 1341. Movie Rating

Description:

We need to return two results:

1. **The name of the user** who has rated the most movies.
   - If there is a tie, choose the one with the lexicographically smaller name.
2. **The title of the highest-rated movie** in February 2020.
   - If there is a tie, choose the one with the lexicographically smaller title.

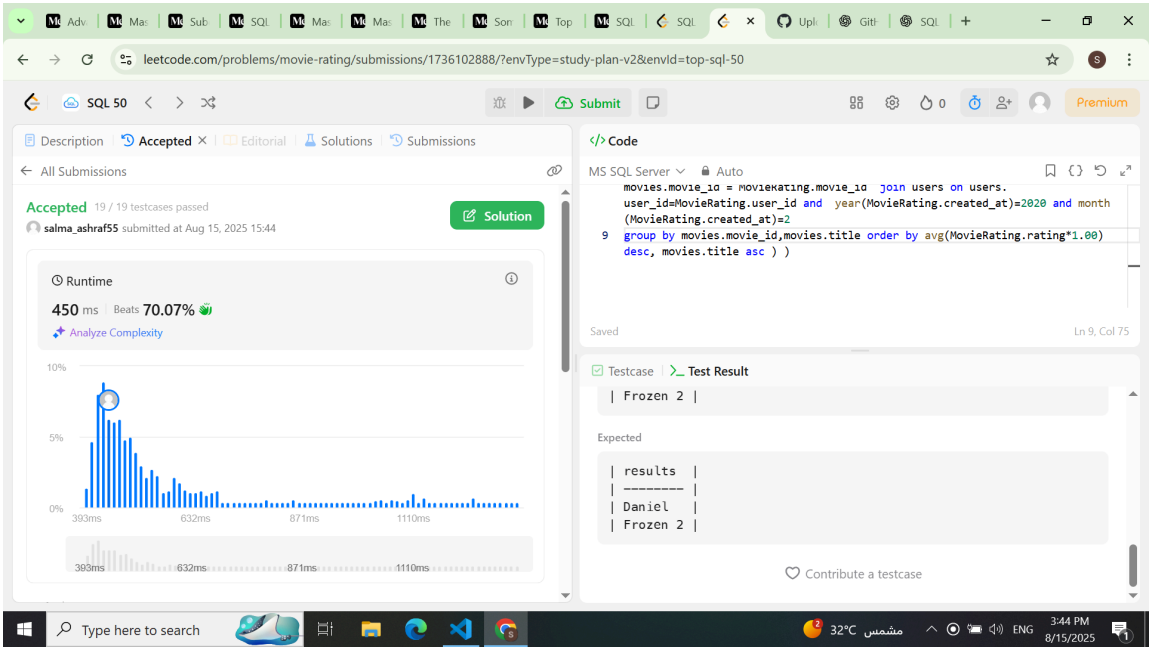
The two results should be combined vertically using `UNION ALL`.

SQL Query:

```sql
-- Part 1: User who rated the most movies
(SELECT TOP 1 name AS results
 FROM users
 WHERE user_id IN (
     SELECT TOP 1 users.user_id
     FROM MovieRating
     JOIN users
         ON users.user_id = MovieRating.user_id
     GROUP BY users.user_id, users.name
     ORDER BY COUNT(MovieRating.user_id) DESC, users.name ASC
 ))

UNION ALL

-- Part 2: Highest-rated movie in February 2020
(SELECT TOP 1 title
 FROM Movies
 WHERE movie_id IN (
     SELECT TOP 1 movies.movie_id
     FROM MovieRating
     JOIN movies
         ON movies.movie_id = MovieRating.movie_id
     JOIN users
         ON users.user_id = MovieRating.user_id
     WHERE YEAR(MovieRating.created_at) = 2020
       AND MONTH(MovieRating.created_at) = 2
     GROUP BY movies.movie_id, movies.title
     ORDER BY AVG(MovieRating.rating * 1.00) DESC, movies.title ASC
 ));
```



In [ ]:
```
!cd "D:\Videos\SIC-BD\task2 sql"
!jupyter nbconvert --to html "leetcode subqueries problems.ipynb"
```

In [ ]:
```
```