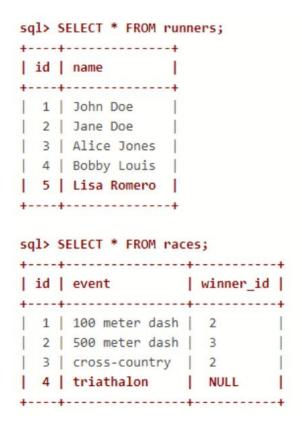
SQL Assesment

Qn.1) What will be the result of the guery below?

SELECT * FROM runners WHERE id NOT IN (SELECT winner id FROM races)

Explain your answer and also provide an alternative version of this query that will avoid the issue that it exposes.



Ans) The issue will arise with the query SELECT * FROM runners WHERE id NOT IN (SELECT winner_id FROM races) due to the presence of NULL values in the winner_id column of the races table. When comparing with NULL values, the result is UNKNOWN, and rows with NULL values in winner_id will not be excluded from the result set.

Alternative Query:

To address this issue and ensure accurate results, you can rewrite the query using a LEFT JOIN and explicitly checking for NULL values. Here's an alternative version of the query:

sql

SELECT r.*

FROM runners r

LEFT JOIN races ra ON r.id = ra.winner_id

WHERE ra.winner id IS NULL OR ra.winner id = ";

Explanation of Alternative Query:

This query uses a LEFT JOIN between the runners and races tables on the id and winner_id.

By checking for both NULL values and empty strings specifically in the winner_id column from the races table, it ensures that all runners who have not won any races are included in the result set.

```
create table test_a(id numeric);
create table test_b(id numeric);
insert into test_a(id) values
  (10),
  (20),
  (30),
  (40),
  (50);
insert into test_b(id) values
  (10),
  (30),
  (30),
  (50);
```

Write a query to fetch values in table test_a that are and not in test_b without using the NOT keyword.

Ans) SELECT a.id

FROM test_a a

LEFT JOIN test_b b ON a.id = b.id

WHERE b.id IS NULL;

Qn.3)

```
SELECT * FROM users;
user_id username
1
      John Doe
2
      Jane Don
      Alice Jones
3
      Lisa Romero
SELECT * FROM training_details;
user_training_id user_id training_id training_date
             1 1 "2015-08-02"
2 1 "2015-08-03"
1
              2
                      1
2
              3
4
2
1
                                "2015-08-02"
                     2
3
                                "2015-08-04"
                     2
                   2
1
2
3
4
1
2
2
                                "2015-08-03"
5
                                "2015-08-02"
                                "2015-08-04"
7
               3
              4
                                "2015-08-03"
8
              1
                                "2015-08-03"
9
              3
                                "2015-08-02"
10
              4
                                "2015-08-04"
11
              1
1
              3
                                 "2015-08-02"
12
                                 "2015-08-02"
13
                                 "2015-08-03"
14
```

Write a query to to get the list of users who took the a training lesson more than once in the same day, grouped by user and training lesson, each ordered from the most recent lesson date to oldest date.

```
Ans)
SELECT
 u.user_id,
 td.training_lesson,
  td.lesson_date,
  COUNT(*) AS lesson_count
FROM
  users u
  JOIN training_details td ON u.user_id = td.user_id
GROUP BY
 u.user_id,
 td.training_lesson,
  td.lesson_date
HAVING
  COUNT(*) > 1
ORDER BY
  u.user_id,
 td.training_lesson,
  td.lesson_date DESC;
```

Qn.4)

Emp_ld	Emp_name	Salary	Manager_ld
10	Anil	50000	18
11	Vikas	75000	16
12	Nisha	40000	18
13	Nidhi	60000	17
14	Priya	80000	18
15	Mohit	45000	18
16	Rajesh	90000	-
17	Raman	55000	16
18	Santosh	65000	17

Write a query to generate below output:

Manager_ld	Manager	Average_Salary_Under_Manager
16	Rajesh	65000
17	Raman	62500
18	Santosh	53750

SELECT E2.EMP_ID, E2.EMP_NAME, AVG(E1.SALARY)

FROM MANAGER_EMP E1

INNER JOIN MANAGER_EMP E2

ON E1.MANAGER_ID = E2.EMP_ID

GROUP BY E2.EMP_ID, E2.EMP_NAME