**Question: 1**

What will be the result of the query 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.

**Answer:**

create table runners (

id int,

name varchar(30)

);

insert into runners values

(1, 'John Doe'),

(2, 'Jane Doe'),

(3, 'Alice Jones'),

(4, 'Bobby Louis'),

(5, 'Lisa Romero');

create table races (

id int,

event varchar(30),

winner\_id int

);

insert into races values

(1, '100 meter dash', 2),

(2, '500 meter dash', 3),

(3, 'cross-country', 2),

(4, 'triathalon', null);

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

The subquery ('SELECT winner\_id FROM races') will return [2, 3, NULL]. The 'NOT IN' condition will then filter out rows from the runners table where the id is 2, 3, or NULL. This means the query will ''ideally'' return the following rows:

| id | name |

|----|--------------|

| 1 | John Doe |

| 4 | Bobby Louis |

| 5 | Lisa Romero |

**The issue with this query is that it does not handle 'NULL' values in the 'winner\_id' column properly and it will return nothing.** If we want to include rows where 'winner\_id' is 'NULL', we need to handle them explicitly.

An alternative version of the query that avoids this issue is:

SELECT \* FROM runners

WHERE id NOT IN (SELECT winner\_id FROM races WHERE winner\_id IS NOT NULL)

This query adds a condition 'winner\_id IS NOT NULL' to include rows where 'winner\_id' is 'NOT NULL'. This ensures that rows with 'NOT NULL' in the 'winner\_id' column are included in the result set.

**Question: 2**

Write a query to fetch values in table test\_a that are and not in test\_b without using the NOT keyword.

**Answer:**

**c**reate table test\_a (

id numeric

);

insert into test\_a values

(10),

(20),

(30),

(40),

(50)

;

create table test\_b (

id numeric

);

insert into test\_b values

(10),

(30),

(50);

Query Answer -

SELECT test\_a.id

FROM test\_a

LEFT JOIN test\_b ON test\_a.id = test\_b.id

WHERE test\_b.id IS NULL;

**Question: 3**

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.

**Answer:**

create table users (

user\_id int,

username varchar(30),

primary key(user\_id )

);

insert into users values

(1, 'John Doe'),

(2, 'Jane Doe'),

(3, 'Alice Jones'),

(4, 'Lisa Romero');

create table training\_details (

user\_training\_id int,

user\_id int,

training\_id int,

training\_date date,

foreign key (user\_id) references users(user\_id)

);

insert into training\_details values

(1, 1, 1, "2015-08-02"),

(2, 2, 1, "2015-08-03"),

(3, 3, 2, "2015-08-02"),

(4, 4, 2, "2015-08-04"),

(5, 2, 2, "2015-08-03"),

(6, 1, 1, "2015-08-02"),

(7, 3, 2, "2015-08-04"),

(8, 4, 3, "2015-08-03"),

(9, 1, 4, "2015-08-03"),

(10, 3, 1, "2015-08-02"),

(11, 4, 2, "2015-08-04"),

(12, 3, 2, "2015-08-02"),

(13, 1, 1, "2015-08-02"),

(14, 4, 3, "2015-08-03");

Query Answer -

SELECT

u.user\_id,

u.username,

td.training\_id,

COUNT(\*) AS num\_lessons\_taken

FROM

users u

INNER JOIN

training\_details td ON u.user\_id = td.user\_id

GROUP BY

u.user\_id,

td.training\_id,

td.training\_date

HAVING

COUNT(\*) > 1

ORDER BY

td.training\_date DESC;