LeetCode SQL 50 Study Plan - Day 1(By Sanjana Thakur)

Select Queries (5 questions)Type :easy

1.LeetCode SQL Problem 1757: Recyclable and Low Fat Products

Problem:

You are given a table Products with the following schema:

```
+-------------+---------+
| Column Name | Type    |
+-------------+---------+
| product_id  | int     |
| low_fats    | enum    |
| recyclable  | enum    |
+-------------+---------+
```

- product_id is the primary key for this table.
- low_fats is an ENUM of type ('Y', 'N') where 'Y' means this product is low fat and 'N' means it is not.
- recyclable is an ENUM of type ('Y', 'N') where 'Y' means this product is recyclable and 'N' means it is not.

Write a solution to find the IDs of products that are both low fat and recyclable. Return the result table in any order.

Example 1:

Input:

Products table:

```
+-------------+----------+------------+
| product_id  | low_fats | recyclable |
+-------------+----------+------------+
| 0           | Y        | N          |
| 1           | Y        | Y          |
| 2           | N        | Y          |
| 3           | Y        | Y          |
| 4           | N        | N          |
+-------------+----------+------------+
```

Output:

```
+-------------+
| product_id  |
+-------------+
| 1           |
| 3           |
+-------------+
```

Explanation: Only products 1 and 3 are both low fat and recyclable.

Solution:

```
SELECT product_id
FROM Products
WHERE low_fats = 'Y' AND recyclable = 'Y';
```

2. LeetCode SQL Problem 584: Find Customer Referee

Problem:

You are given a table Customer with the following schema:

sql

Copy code

```
+-------------+---------+
| Column Name | Type    |
+-------------+---------+
| id          | int     |
| name        | varchar |
| referee_id  | int     |
+-------------+---------+
```

- id is the primary key for this table.
- Each row of this table indicates the id of a customer, their name, and the id of the customer who referred them.

Write a solution to find the names of the customers that are not referred by the customer with id = 2. Return the result table in any order.

Example 1:

Input:

Customer table:

```
+----+------+------------+
| id | name | referee_id |
+----+------+------------+
| 1  | Will | null       |
| 2  | Jane | null       |
| 3  | Alex | 2          |
| 4  | Bill | null       |
| 5  | Zack | 1          |
| 6  | Mark | 2          |
+----+------+------------+
```

Output:

```
+------+
| name |
+------+
| Will |
| Jane |
| Bill |
| Zack |
+------+
```

Solution:

```
SELECT name
FROM Customer
WHERE referee_id IS NULL OR referee_id != 2;
```

3. LeetCode SQL Problem 595: Big Countries

Problem:

You are given a table World with the following schema:

```
+-------------+---------+
| Column Name | Type    |
+-------------+---------+
| name        | varchar |
| continent   | varchar |
| area        | int     |
| population  | int     |
| gdp         | bigint  |
+-------------+---------+
```

- name is the primary key for this table.
- Each row of this table gives information about a country, including its name, continent, area, population, and gdp.

A country is considered "big" if:

- It has an area of at least three million (i.e., 3,000,000 km^{2}), or
- It has a population of at least twenty-five million (i.e., 25,000,000).

Write a solution to find the name, population, and area of the big countries. Return the result table in any order.

Example 1:

Input:
World table:

```
+-------------+-----------+---------+------------+--------------+
| name        | continent | area    | population | gdp          |
+-------------+-----------+---------+------------+--------------+
| Afghanistan | Asia      | 652230  | 25500100   | 20343000000  |
| Albania     | Europe    | 28748   | 2831741    | 12960000000  |
| Algeria     | Africa    | 2381741 | 37100000   | 188681000000 |
| Andorra     | Europe    | 468     | 78115      | 3712000000   |
| Angola      | Africa    | 1246700 | 20609294   | 100990000000 |
+-------------+-----------+---------+------------+--------------+
Output:
+-------------+------------+---------+
| name        | population | area    |
+-------------+------------+---------+
| Afghanistan | 25500100   | 652230  |
| Algeria     | 37100000   | 2381741 |
+-------------+------------+---------+
```

Solution:

```
SELECT name, population, area
FROM World
WHERE area >= 3000000 OR population >= 25000000;
```

4. LeetCode SQL Problem 1148: Article Views I

Problem:

You are given a table Views with the following schema:

```
+---------------+---------+
| Column Name   | Type    |
+---------------+---------+
| article_id    | int     |
| author_id     | int     |
| viewer_id     | int     |
| view_date     | date    |
+---------------+---------+
```

- There is no primary key for this table, and it may contain duplicate rows.
- Each row in this table indicates that some viewer viewed an article (written by some author) on some date.

Write a solution to find all the authors that viewed at least one of their own articles. Return the result table sorted by id in ascending order.

Example 1:

```
Input:
Views table:
+------------+-----------+-----------+------------+
| article_id | author_id | viewer_id | view_date  |
+------------+-----------+-----------+------------+
| 1          | 3         | 5         | 2019-08-01 |
| 1          | 3         | 6         | 2019-08-02 |
| 2          | 7         | 7         | 2019-08-01 |
| 2          | 7         | 6         | 2019-08-02 |
| 4          | 7         | 1         | 2019-07-22 |
| 3          | 4         | 4         | 2019-07-21 |
| 3          | 4         | 4         | 2019-07-21 |
+------------+-----------+-----------+------------+
```

Output:

```
+------+
| id   |
+------+
| 4    |
| 7    |
+------+
```

Solution:

```
SELECT DISTINCT author_id AS id
FROM Views
WHERE author_id = viewer_id
ORDER BY id;
```

5. LeetCode SQL Problem 1683: Invalid Tweets

Problem:

You are given a table Tweets with the following schema:

```
+----------------+---------+
| Column Name    | Type    |
+----------------+---------+
| tweet_id       | int     |
| content        | varchar |
+----------------+---------+
```

- tweet_id is the primary key for this table.
- This table contains all the tweets in a social media app.

Write a solution to find the IDs of the invalid tweets. A tweet is considered invalid if the number of characters used in the content of the tweet is strictly greater than 15. Return the result table in any order.

Example 1:

Input:

Tweets table:

```
+----------+----------------------------------+
| tweet_id | content                          |
+----------+----------------------------------+
| 1        | Vote for Biden                   |
| 2        | Let us make America great again! |
+----------+----------------------------------+
Output:
+----------+
| tweet_id |
+----------+
| 2        |
+----------+
```

Explanation: Tweet 1 has a length of 14 characters and is a valid tweet. Tweet 2 has a length of 32 characters and is an invalid tweet.

Solution:

```
SELECT tweet_id
FROM Tweets
WHERE LENGTH(content) > 15;
```

Basic Joins (9 questions)/2 medium

1. Replace Employee ID With The Unique Identifier

Question:

Write a solution to show the unique ID of each user. If a user does not have a unique ID, replace it with null.

Return the result table in any order.

Input:

Employees Table:

id	name
1	Alice
7	Bob
11	Meir
90	Winston
3	Jonathan

EmployeeUNI Table:

id	unique_id
3	1
11	2
90	3

Output:

unique_id	name

null	Alice
null	Bob
2	Meir
3	Winston
1	Jonathan

Solution:

```
SELECT
   eu.unique_id,
   e.name
FROM
   Employees e
LEFT JOIN
   EmployeeUNI eu ON e.id = eu.id;
```

2. Product Sales Analysis I

Question:

Write a solution to report the product_name, year, and price for each sale_id in the Sales table.
Return the resulting table in any order.

Input:

Sales Table:

sale_id	product_id	year	quantity	price
1	100	2008	10	5000
2	100	2009	12	5000
7	200	2011	15	9000

Product Table:

product_id	product_name
100	Nokia
200	Apple
300	Samsung

Output:

product_name	year	price
Nokia	2008	5000
Nokia	2009	5000
Apple	2011	9000

Solution:

```
SELECT
   p.product_name,
   s.year,
   s.price
FROM
   Sales s
JOIN
   Product p ON s.product_id = p.product_id;
```

3. Customer Who Visited but Did Not Make Any Transactions

Question:

Write a solution to find the IDs of the users who visited without making any transactions and the number of times they made these types of visits.

Return the result table sorted in any order.

Input:

Visits Table:

visit_id	customer_id
1	23
2	9
4	30
5	54
6	96
7	54
8	54

Transactions Table:

transaction_id	visit_id	amount
2	5	310
3	5	300
9	5	200
12	1	910
13	2	970

Output:

customer_id	count_no_trans
54	2
30	1
96	1

Solution:

```
SELECT
    v.customer_id,
    COUNT(v.visit_id) AS count_no_trans
FROM
    Visits v
```

```
LEFT JOIN
   Transactions t ON v.visit_id = t.visit_id
WHERE
   t.transaction_id IS NULL
GROUP BY
   v.customer_id;
```

4. Rising Temperature

Question:

Write a solution to find all dates' IDs with higher temperatures compared to their previous dates (yesterday).

Return the result table in any order.

Input:

Weather Table:

id	recordDate	temperature
1	2015-01-01	10
2	2015-01-02	25
3	2015-01-03	20
4	2015-01-04	30

Output:

id
2
4

Solution:

```
SELECT
   W1.id
FROM
   Weather W1
JOIN
   Weather W2 ON W1.recordDate = DATE_ADD(W2.recordDate,
INTERVAL 1 DAY)
WHERE
   W1.temperature > W2.temperature;
```

5. Average Time of Process per Machine

Question:

Write a solution to find the average time each machine takes to complete a process.

The time to complete a process is the 'end' timestamp minus the 'start' timestamp. The average time is calculated by the total time to complete every process on the machine divided by the number of processes that were run.

The resulting table should have the machine_id along with the average time as processing_time, rounded to 3 decimal places.

Input:

Activity Table:

machine_id	process_id	activity_type	timestamp
0	0	start	0.712
0	0	end	1.520
0	1	start	3.140
0	1	end	4.120
1	0	start	0.550
1	0	end	1.550
1	1	start	0.430
1	1	end	1.420
2	0	start	4.100
2	0	end	4.512
2	1	start	2.500
2	1	end	5.000

Output:

machine_id	processing_time
0	0.894
1	0.995
2	1.456

Solution:

```
WITH ProcessTimes AS (
    SELECT
        machine_id,
        (MAX(CASE WHEN activity_type = 'end' THEN timestamp END) -
         MAX(CASE WHEN activity_type = 'start' THEN timestamp END))
AS process_time
    FROM Activity
    GROUP BY machine_id, process_id
)
SELECT
    machine_id,
    ROUND(AVG(process_time), 3) AS processing_time
FROM ProcessTimes
GROUP BY machine_id;
```

6. Employee Bonus(medium)

Question:
Write a solution to report the name and bonus amount of each employee with a bonus less than 1000.
Return the result table in any order.

Input:

Employee Table:

empId	name	supervisor	salary
3	Brad	null	4000
1	John	3	1000
2	Dan	3	2000
4	Thomas	3	4000

Bonus Table:

empId	bonus
2	500
4	2000

Output:

name	bonus
Brad	null
John	null
Dan	500

Solution:

```
SELECT
   e.name,
   b.bonus
FROM
   Employee e
LEFT JOIN
   Bonus b ON e.empId = b.empId
WHERE
   b.bonus < 1000 OR b.bonus IS NULL;
```

7. Students and Examinations(medium)

Question:
Write a solution to find the number of times each student attended each exam.
Return the result table ordered by student_id and subject_name.

Input:

Students Table:

student_id	student_name
1	Alice

2	Bob
13	John
6	Alex

Subjects Table:

subject_name
Math
Physics
Programming

Examinations Table:

student_id	subject_name
1	Math
1	Physics
1	Programming
2	Programming
1	Physics
1	Math
2	Math

Output:

student_id	subject_name	num_exams
1	Math	2
1	Physics	2
1	Programming	1
2	Math	1
2	Programming	1

Solution:

```
SELECT
   s.student_id,
   e.subject_name,
   COUNT(*) AS num_exams
FROM
   Examinations e
JOIN
   Students s ON e.student_id = s.student_id
GROUP BY
   s.student_id, e.subject_name
ORDER BY
   s.student_id, e.subject_name;
```