DESCRIPTION

Table: Delivery

+-----------------------------+---------+

| Column Name | Type |

+-----------------------------+---------+

| delivery_id | int |

| customer_id | int |

| order_date | date |

| customer_pref_delivery_date | date |

+-----------------------------+---------+

delivery_id is the column of unique values of this table.

The table holds information about food delivery to customers that make orders at some date and specify a preferred delivery date (on the same order date or after it).

If the customer's preferred delivery date is the same as the order date, then the order is called **immediate;** otherwise, it is called **scheduled**.

The **first order** of a customer is the order with the earliest order date that the customer made. It is guaranteed that a customer has precisely one first order.

Write a solution to find the percentage of immediate orders in the first orders of all customers, **rounded to 2 decimal places**.

The result format is in the following example.

Example 1:

Input:

Delivery table:

+-------------+-------------+------------+----------------------------+

| delivery_id | customer_id | order_date | customer_pref_delivery_date |

+-------------+-------------+------------+----------------------------+

| 1 | 1 | 2019-08-01 | 2019-08-02 |

| 2 | 2 | 2019-08-02 | 2019-08-02 |

| 3 | 1 | 2019-08-11 | 2019-08-12 |

| 4 | 3 | 2019-08-24 | 2019-08-24 |

| 5 | 3 | 2019-08-21 | 2019-08-22 |

| 6 | 2 | 2019-08-11 | 2019-08-13 |

| 7 | 4 | 2019-08-09 | 2019-08-09 |

+-------------+-------------+------------+----------------------------+

Output:

+----------------------+

| immediate_percentage |

+----------------------+

| 50.00 |

+----------------------+

Explanation:

The customer id 1 has a first order with delivery id 1 and it is scheduled.

The customer id 2 has a first order with delivery id 2 and it is immediate.

The customer id 3 has a first order with delivery id 5 and it is scheduled.

The customer id 4 has a first order with delivery id 7 and it is immediate.

Hence, half the customers have immediate first orders.

SOLUTION

MySQL:

- Select query_name, calculate quality using AVG(), and round the result to 2 decimals using ROUND()
- Calculate poor_query_percentage using IF() (if rating is less than 3, then 1, else 0), add up using SUM(), and round the result to 2 decimals using ROUND()
- GROUP BY query_name

TO DO

PostgreSQL:

- Select query_name, calculate quality using SUM() and COUNT(), and round the result to 2 decimals using ROUND()
- Calculate poor_query_percentage using CASE (when rating is less than 3, then 1, else 0), add up using SUM(), and round the result to 2 decimals using ROUND()
- GROUP BY query_name

```
WITH t1 AS(
    SELECT customer_id, MIN(order_date) first_order
    FROM DELIVERY
    GROUP BY 1)
SELECT ROUND(AVG(CASE WHEN d.order_date = d.customer_pref_delivery_date THEN 1 ELSE 0 END) * 100, 2)
immediate_percentage
FROM Delivery d
JOIN t1
ON t1.customer_id = d.customer_id AND t1.first_order = d.order_date;
```