Problem 3521: Find Product Recommendation Pairs

Problem Information

Difficulty: Medium
Acceptance Rate: 61.71%
Paid Only: No
Tags: Database

Problem Description

Table: `ProductPurchases`

+-------------+------+ | Column Name | Type | +-------------+------+ | user_id | int | | product_id | int | | quantity | int | +-------------+------+ (user_id, product_id) is the unique key for this table. Each row represents a purchase of a product by a user in a specific quantity.

Table: `ProductInfo`

+-------------+---------+ | Column Name | Type | +-------------+---------+ | product_id | int | | category | varchar | | price | decimal | +-------------+---------+ product_id is the primary key for this table. Each row assigns a category and price to a product.

Amazon wants to implement the **Customers who bought this also bought...** feature based on **co-purchase patterns**. Write a solution to :

1. Identify **distinct** product pairs frequently **purchased together by the same customers** (where `product1_id` < `product2_id`) 2. For **each product pair** , determine how many customers purchased **both** products

A product pair is considered for recommendation **if** **at least** `3` **different** customers have purchased **both products**.

Return _the_ _result table ordered by**customer_count** in **descending** order, and in case of a tie, by _`product1_id` _in**ascending** order, and then by _`product2_id` _in**ascending** order_.

The result format is in the following example.

Example:

Input:

ProductPurchases table:

+---------+------------+----------+ | user_id | product_id | quantity | +---------+------------+----------+ | 1 | 101 | 2 | | 1 | 102 | 1 | | 1 | 103 | 3 | | 2 | 101 | 1 | | 2 | 102 | 5 | | 2 | 104 | 1 | | 3 | 101 | 2 | | 3 | 103 | 1 | | 3 | 105 | 4 | | 4 | 101 | 1 | | 4 | 102 | 1 | | 4 | 103 | 2 | | 4 | 104 | 3 | | 5 | 102 | 2 | | 5 | 104 | 1 | +---------+------------+----------+

ProductInfo table:

+------------+-------------+-------+ | product_id | category | price | +------------+-------------+-------+ | 101 | Electronics | 100 | | 102 | Books | 20 | | 103 | Clothing | 35 | | 104 | Kitchen | 50 | | 105 | Sports | 75 | +------------+-------------+-------+

Output:

+-------------+-------------+-------------------+-------------------+----------------+ | product1_id | product2_id | product1_category | product2_category | customer_count | +-------------+-------------+-------------------+-------------------+----------------+ | 101 | 102 | Electronics | Books | 3 | | 101 | 103 | Electronics | Clothing | 3 | | 102 | 104 | Books | Kitchen | 3 | +-------------+-------------+-------------------+-------------------+----------------+

Explanation:

* **Product pair (101, 102):** * Purchased by users 1, 2, and 4 (3 customers) * Product 101 is in Electronics category * Product 102 is in Books category * **Product pair (101, 103):** * Purchased by users 1, 3, and 4 (3 customers) * Product 101 is in Electronics category * Product 103 is in Clothing category * **Product pair (102, 104):** * Purchased by users 2, 4, and 5 (3 customers) * Product 102 is in Books category * Product 104 is in Kitchen category

The result is ordered by customer_count in descending order. For pairs with the same customer_count, they are ordered by product1_id and then product2_id in ascending order.

Code Snippets

MySQL:

```
# Write your MySQL query statement below
```

MS SQL Server:

```
/* Write your T-SQL query statement below */
```

PostgreSQL:

```
-- Write your PostgreSQL query statement below
```