Problem 3626: Find Stores with Inventory Imbalance

Problem Information

Difficulty: Medium
Acceptance Rate: 0.00%
Paid Only: No

Problem Description

Table:

stores

+-------------+---------+ | Column Name | Type | +-------------+---------+ | store_id | int | | store_name | varchar | | location | varchar | +-------------+---------+ store_id is the unique identifier for this table. Each row contains information about a store and its location.

Table:

inventory

+-------------+---------+ | Column Name | Type | +-------------+---------+ | inventory_id| int | | store_id | int | | product_name| varchar | | quantity | int | | price | decimal | +-------------+---------+ inventory_id is the unique identifier for this table. Each row represents the inventory of a specific product at a specific store.

Write a solution to find stores that have

inventory imbalance

- stores where the most expensive product has lower stock than the cheapest product.

For each store, identify the

most expensive product

(highest price) and its quantity

For each store, identify the

cheapest product

(lowest price) and its quantity

A store has inventory imbalance if the most expensive product's quantity is

less than

the cheapest product's quantity

Calculate the

imbalance ratio

as (cheapest_quantity / most_expensive_quantity)

Round

the imbalance ratio to

2

decimal places

Only include stores that have

at least

3

different products

Return

the result table ordered by imbalance ratio in

descending

order, then by store name in

ascending

order

.

The result format is in the following example.

Example:

Input:

stores table:

store_id	store_name	location
1	Downtown Tech	New York
2	Suburb Mall	Chicago
3	City Center	Los Angeles
4	Corner Shop	Miami
5	Plaza Store	Seattle

inventory table:

inventory_id	store_id	product_name	quantity	price
1	1	Laptop	5	999.99
2	1	Mouse	50	19.99
3	1	Keyboard	25	79.99
4	1	Monitor	15	299.99
5	2	Phone	3	699.99
6	2	Charger	100	25.99
7	2	Case	75	15.99
8	2	Headphones	20	149.99
9	3	Tablet	2	499.99
10	3	Stylus	80	29.99
11	3	Cover	60	39.99
12	4	Watch	10	299.99
13	4	Band	25	49.99
14	5	Camera	8	599.99
15	5	Lens	12	199.99

Output:

store_id	store_name	location	most_exp_product	cheapest_product	imbalance_ratio

```
+----------+----------------+-------------+------------------+--------------------+------------------+ | 3 | City
Center | Los Angeles | Tablet | Stylus | 40.00 | | 1 | Downtown Tech | New York | Laptop |
Mouse | 10.00 | | 2 | Suburb Mall | Chicago | Phone | Case | 25.00 |
+----------+----------------+-------------+------------------+--------------------+------------------+
```

Explanation:

Downtown Tech (store_id = 1):

Most expensive product: Laptop ($999.99) with quantity 5

Cheapest product: Mouse ($19.99) with quantity 50

Inventory imbalance: 5 < 50 (expensive product has lower stock)

Imbalance ratio: 50 / 5 = 10.00

Has 4 products (≥ 3), so qualifies

Suburb Mall (store_id = 2):

Most expensive product: Phone ($699.99) with quantity 3

Cheapest product: Case ($15.99) with quantity 75

Inventory imbalance: 3 < 75 (expensive product has lower stock)

Imbalance ratio: 75 / 3 = 25.00

Has 4 products (≥ 3), so qualifies

City Center (store_id = 3):

Most expensive product: Tablet ($499.99) with quantity 2

Cheapest product: Stylus ($29.99) with quantity 80

Inventory imbalance: 2 < 80 (expensive product has lower stock)

Imbalance ratio: 80 / 2 = 40.00

Has 3 products (≥ 3), so qualifies

Stores not included:

Corner Shop (store_id = 4): Only has 2 products (Watch, Band) - doesn't meet minimum 3 products requirement

Plaza Store (store_id = 5): Only has 2 products (Camera, Lens) - doesn't meet minimum 3 products requirement

The Results table is ordered by imbalance ratio in descending order, then by store name 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
```

Oracle:

```
/* Write your PL/SQL query statement below */
```

Pandas:

```
import pandas as pd

def find_inventory_imbalance(stores: pd.DataFrame, inventory: pd.DataFrame)
-> pd.DataFrame:
```

Solutions

MySQL Solution:

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

MS SQL Server Solution:

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

PostgreSQL Solution:

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

Oracle Solution:

```
/* Write your PL/SQL query statement below */
```

Pandas Solution:

```
import pandas as pd

def find_inventory_imbalance(stores: pd.DataFrame, inventory: pd.DataFrame)
-> pd.DataFrame:
```