

Problem 3611: Find Overbooked Employees

Problem Information

Difficulty: **Medium**

Acceptance Rate: 43.68%

Paid Only: No

Tags: Database

Problem Description

Table: `employees`

+-----+-----+ | Column Name | Type | +-----+-----+ | employee_id | int | | employee_name | varchar | | department | varchar | +-----+-----+ employee_id is the unique identifier for this table. Each row contains information about an employee and their department.

Table: `meetings`

+-----+-----+ | Column Name | Type | +-----+-----+ | meeting_id | int | | employee_id | int | | meeting_date | date | | meeting_type | varchar | | duration_hours | decimal | +-----+-----+ meeting_id is the unique identifier for this table. Each row represents a meeting attended by an employee. meeting_type can be 'Team', 'Client', or 'Training'.

Write a solution to find employees who are **meeting-heavy** \- employees who spend more than 50% of their working time in meetings during any given week.

* Assume a standard work week is 40 hours * Calculate total meeting hours per employee per week (Monday to Sunday) * An employee is meeting-heavy if their weekly meeting hours > 20 hours (50% of 40 hours) * Count how many weeks each employee was meeting-heavy * Only include employees who were meeting-heavy for at least 2 weeks

Return the result table ordered by the number of meeting-heavy weeks in descending order, then by employee name in ascending order.

The result format is in the following example.

Example:

Input:

employees table:

employee_id	employee_name	department
1	Alice Johnson	Engineering
2	Bob Smith	Marketing
3	Carol Davis	Sales
4	David Wilson	Engineering
5	Emma Brown	HR

meetings table:

meeting_id	employee_id	meeting_date	meeting_type	duration_hours
1	1	2023-06-05	Team	8.0
2	1	2023-06-06	Client	6.0
3	1	2023-06-07	Training	7.0
4	1	2023-06-12	Team	12.0
5	1	2023-06-13	Client	9.0
6	2	2023-06-05	Team	15.0
7	2	2023-06-06	Client	8.0
8	2	2023-06-12	Training	10.0
9	3	2023-06-05	Team	4.0
10	3	2023-06-06	Client	3.0
11	4	2023-06-05	Team	25.0
12	4	2023-06-19	Client	22.0
13	5	2023-06-05	Training	2.0

Output:

employee_id	employee_name	department	meeting_heavy_weeks
1	Alice Johnson	Engineering	2
4	David Wilson	Engineering	2

Explanation:

* **Alice Johnson (employee_id = 1):** * Week of June 5-11 (2023-06-05 to 2023-06-11): 8.0 + 6.0 + 7.0 = 21.0 hours (> 20 hours) * Week of June 12-18 (2023-06-12 to 2023-06-18): 12.0 + 9.0 = 21.0 hours (> 20 hours) * Meeting-heavy for 2 weeks * **David Wilson (employee_id = 4):** * Week of June 5-11: 25.0 hours (> 20 hours) * Week of June 19-25: 22.0 hours (> 20 hours) * Meeting-heavy for 2 weeks * **Employees not included:** * Bob Smith (employee_id = 2): Week of June 5-11: 15.0 + 8.0 = 23.0 hours (> 20), Week of June 12-18: 10.0 hours (< 20). Only 1 meeting-heavy week * Carol Davis (employee_id = 3): Week of June 5-11: 4.0 +

3.0 = 7.0 hours (< 20). No meeting-heavy weeks * Emma Brown (employee_id = 5): Week of June 5-11: 2.0 hours (< 20). No meeting-heavy weeks

The result table is ordered by meeting_heavy_weeks in descending order, then by employee 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
```