

Question #1

Problem List < > Submit Premium

Testcase Test Result

Case 1

Input

```
ActorDirector =
| actor_id | director_id | timestamp |
| ----- | ----- | ----- |
| 1 | 1 | 0 |
| 1 | 1 | 1 |
| 1 | 1 | 2 |
| 1 | 2 | 3 |
| 1 | 2 | 4 |
| 2 | 1 | 5 |
```

View more

Output

```
| actor_id | director_id |
| ----- | ----- |
| 1 | 1 |
```

Expected

```
| actor_id | director_id |
| ----- | ----- |
| 1 | 1 |
```

Contribute a testcase

Description Editorial Solutions Submissions

1050. Actors and Directors Who Cooperated At Least Three Times Solved

Easy Topics Companies

SQL Schema > Pandas Schema >

Table: ActorDirector

Column Name	Type
actor_id	int
director_id	int
timestamp	int

timestamp is the primary key (column with unique values) for this table.

Write a solution to find all the pairs (`actor_id`, `director_id`) where the actor has cooperated with the director at least three times.

Return the result table in **any order**.

The result format is in the following example.

751 50 8 Online

Accepted Runtime: 85 ms

Case 1

Input

```
ActorDirector =
| actor_id | director_id | timestamp |
| ----- | ----- | ----- |
| 1 | 1 | 0 |
| 1 | 1 | 1 |
| 1 | 1 | 2 |
| 1 | 2 | 3 |
| 1 | 2 | 4 |
| 2 | 1 | 5 |
```

Mariam_Zoair183 submitted at Nov 05, 2025 18:29

Runtime

364 ms | Beats **39.89%**

Analyze Complexity

Code | MySQL

```
SELECT actor_id, director_id
FROM ActorDirector
GROUP BY actor_id, director_id
HAVING COUNT(*) >= 3;
```

Question #2

[Description](#) | [Editorial](#) | [Solutions](#) | [Submissions](#)

1667. Fix Names in a Table

[Easy](#) | [Topics](#) | [Companies](#)

[SQL Schema](#) > [Pandas Schema](#) >

Table: Users

Column Name	Type
user_id	int
name	varchar

user_id is the primary key (column with unique values) for this table. This table contains the ID and the name of the user. The name consists of only lowercase and uppercase characters.

Write a solution to fix the names so that only the first character is uppercase and the rest are lowercase.

Return the result table ordered by `user_id`.

The result format is in the following example.

Example 1:

1K 116

• 14 Online

Accepted Runtime: 86 ms

Case 1

Input

Users =	
user_id	name
-----	-----
1	alice
2	b0B

Output

user_id	name
1	Alice
2	Bob

Expected

user_id	name
1	Alice
2	Bob

MySQL v Auto

```
</> Code
1 SELECT
2   | user_id,
3   | CONCAT(UPPER(LEFT(name, 1)), LOWER(SUBSTRING(name, 2))) AS name
4 FROM Users
5 ORDER BY user_id;
```

Saved Ln 1, Col 1

Testcase > [Test Result](#)

Accepted Runtime: 86 ms

Case 1

Input

Users =

user_id	name
1	aLice
2	b0B

Output

Question #3

175. Combine Two Tables

Solved 0

MySQL v Auto

```

1 SELECT
2   p.firstName,
3   p.lastName,
4   a.city,
5   a.state
6 FROM Person p
7 LEFT JOIN Address a
8   ON p.PersonId = a.PersonId;

```

Saved Ln 1, Col 1

Testcase | Test Result

Accepted Runtime: 101 ms

Case 1

Input

Person =

personId	lastName	firstName
1	Wang	Allen
2	Alice	Bob

Address =

addressId	personId	city	state
1	2	New York City	New York
2	3	Leetcode	California

• 101 Online

Output

firstName	lastName	city	state
Allen	Wang	null	null
Bob	Alice	New York City	New York

Expected

firstName	lastName	city	state
Allen	Wang	null	null
Bob	Alice	New York City	New York

Question #4

176. Second Highest Salary

Medium Topics Companies

SQL Schema > Pandas Schema >

Table: Employee

```
+-----+  
| Column Name | Type |  
+-----+  
| id          | int   |  
| salary       | int   |  
+-----+  
id is the primary key (column with unique values) for this table.  
Each row of this table contains information about the salary of an employee.
```

Write a solution to find the second highest **distinct** salary from the `Employee` table. If there is no second highest salary, return `null` (return `None` in Pandas).

The result format is in the following example.

Example 1:

Input:
Employee table:
4.1K 369 78 Online

Testcase Test Result

Accepted Runtime: 123 ms

Case 1 Case 2

Input

```
Employee =  
| id | salary |  
| -- | ----- |  
| 1  | 100   |  
| 2  | 200   |  
| 3  | 300   |
```

Output

```
| SecondHighestSalary |  
| ----- |  
| 200   |
```

Expected

```
| SecondHighestSalary |  
| ----- |  
| 200   |
```

MySQL ✓ Auto

```
1 SELECT  
2   (SELECT MAX(salary)  
3    FROM Employee  
4   WHERE salary < (SELECT MAX(salary) FROM Employee)  
5 ) AS SecondHighestSalary;
```

Saved Ln 1, Col 1

Testcase Test Result

Accepted Runtime: 123 ms

Case 1 Case 2

Input

```
Employee =  
| id | salary |  
| -- | ----- |  
| 1  | 100   |  
| 2  | 200   |  
| 3  | 300   |
```

Output

```
| SecondHighestSalary |  
| ----- |  
| 200   |
```

Contribute a testcase

Question #5

1327. List the Products Ordered in a Period

[Easy](#) [Topics](#) [Companies](#)
[SQL Schema](#) > [Pandas Schema](#) >

Table: Products

Column Name	Type
product_id	int
product_name	varchar
product_category	varchar

product_id is the primary key (column with unique values) for this table.
This table contains data about the company's products.

Table: Orders

Column Name	Type
product_id	int
order_date	date
unit	int

528 | 76 | ⚡ | 7 Online

✓ Code

MySQL v Auto

```

1 SELECT
2   p.product_name,
3   SUM(o.unit) AS unit
4 FROM Products p
5 JOIN Orders o
6   ON p.product_id = o.product_id
7 WHERE o.order_date BETWEEN '2020-02-01' AND '2020-02-29'
8 GROUP BY p.product_name
9 HAVING SUM(o.unit) >= 100;

```

Saved

Ln 1, Col 1

✓ Testcase | » Test Result

Accepted Runtime: 89 ms

 Case 1

Input

Products =

product_id	product_name	product_category
1	Leetcode Solutions	Book
2	Jewels of Stringology	Book
3	HP	Laptop
4	Lenovo	Laptop
5	iPhone Mirroring Code Kit	T-shirt

✓ Testcase | » Test Result

Accepted Runtime: 89 ms

 Case 1

Input

Products =

product_id	product_name	product_category
1	Leetcode Solutions	Book
2	Jewels of Stringology	Book
3	HP	Laptop
4	Lenovo	Laptop
5	Leetcode Kit	T-shirt

Orders =

product_id	order_date	unit
1	2020-02-05	60
1	2020-02-10	70
2	2020-01-18	30
2	2020-02-11	80
3	2020-02-17	2
3	2020-02-24	3

▼ View more

Output

Orders =

product_id	order_date	unit
1	2020-02-05	60
1	2020-02-10	70
2	2020-01-18	30
2	2020-02-11	80
3	2020-02-17	2
3	2020-02-24	3

▼ View more

Output

product_name	unit
Leetcode Solutions	130
Leetcode Kit	100

Expected

product_name	unit
Leetcode Solutions	130
Leetcode Kit	100

» Contribute a testcase

Question #6

[Description](#) | [Editorial](#) | [Solutions](#) | [Submissions](#)

1378. Replace Employee ID With The Unique Identifier

[Easy](#) | [Topics](#) | [Companies](#)

[SQL Schema](#) > [Pandas Schema](#) >

Table: Employees

Column Name	Type
<code>id</code>	<code>int</code>
<code>name</code>	<code>varchar</code>

`id` is the primary key (column with unique values) for this table.
Each row of this table contains the `id` and the name of an employee in a company.

Table: EmployeeUNI

Column Name	Type
<code>id</code>	<code>int</code>
<code>unique_id</code>	<code>int</code>

(`id`, `unique_id`) is the primary key (combination of columns with unique

values) for this table.

1.9K 199 ⚡ 39 Online

[Testcase](#) | [Test Result](#)

Input

Employees =	
<code>id</code>	<code>name</code>
—	—
1	Alice
7	Bob
11	Meir
90	Winston
3	Jonathan

EmployeeUNI =	
<code>id</code>	<code>unique_id</code>
—	—
3	1
11	2
90	3

Output

unique_id	name
—	—
null	Alice
null	Bob
2	Meir
3	Winston
1	Jonathan

EmployeeUNI =	
<code>id</code>	<code>unique_id</code>
—	—
3	1
11	2
90	3

Output

unique_id	name
—	—
null	Alice
null	Bob
2	Meir
3	Winston
1	Jonathan

Expected

unique_id	name
—	—
null	Alice
null	Bob
2	Meir
3	Winston
1	Jonathan

</> Code

MySQL | Auto

```
1 SELECT
2   eu.unique_id,
3   e.name
4   FROM Employees e
5   LEFT JOIN EmployeeUNI eu
6   ON e.id = eu.id;
```

Saved

Ln 1, Col 1

[Testcase](#) | [Test Result](#)

Accepted Runtime: 98 ms

Case 1

Input

Employees =

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

39 Online

•

EmployeeUNI =

id	unique_id
—	—
3	1
11	2
90	3

•

Output

unique_id	name
—	—
null	Alice
null	Bob
2	Meir
3	Winston
1	Jonathan

•

Expected

unique_id	name
—	—
null	Alice
null	Bob
2	Meir
3	Winston
1	Jonathan

© 2024 LeetCode Ltd. All rights reserved. LeetCode is a trademark of LeetCode Ltd. All other trademarks are the property of their respective owners.

Question #7

Description | Editorial | Solutions | Submissions

550. Game Play Analysis IV

Medium Topics Companies

SQL Schema > Pandas Schema >

Table: Activity

Column Name	Type
player_id	int
device_id	int
event_date	date
games_played	int

(player_id, event_date) is the primary key (combination of columns with unique values) of this table.

This table shows the activity of players of some games.

Each row is a record of a player who logged in and played a number of games (possibly 0) before logging out on someday using some device.

Write a solution to report the fraction of players that logged in again on the day after the day they first logged in, rounded to 2 decimal places. In other words, you need to determine the number of players who logged in on the day immediately following their initial login, and divide it by the number of total players.

The result format is in the following example.

1.4K 271 • 52 Online

MySQL Auto
Saved Ln 1, Col 1

</> Code

```
1 -- Fraction of players who returned the day after their first login
2 SELECT
3   ROUND(AVG(CASE WHEN a2.player_id IS NULL THEN 0 ELSE 1 END), 2) AS fraction
4 FROM (
5   SELECT player_id, MIN(event_date) AS first_login
6   FROM Activity
7   GROUP BY player_id
8 ) f
9 LEFT JOIN Activity a2
  ON f.player_id = a2.player_id
  AND f.first_login = a2.event_date - INTERVAL 1 DAY
```

Testcase | Test Result

Accepted Runtime: 114 ms

Case 1

Input

Activity =			
player_id	device_id	event_date	games_played
1	2	2016-03-01	5
1	2	2016-03-02	6
2	3	2017-06-25	1
3	1	2016-03-02	0
3	4	2018-07-03	5

Output

fraction

0.33

Input

Activity =
player_id device_id event_date games_played
----- ----- ----- -----
1 2 2016-03-01 5
1 2 2016-03-02 6
2 3 2017-06-25 1
3 1 2016-03-02 0
3 4 2018-07-03 5

Output

fraction

0.33

Expected

fraction

0.33

Question # 8

1075. Project Employees I

[Easy](#) [Topics](#) [Companies](#)
[SQL Schema](#) > [Pandas Schema](#) >

Table: Project

```
+-----+-----+
| Column Name | Type   |
+-----+-----+
| project_id  | int    |
| employee_id | int    |
+-----+-----+
(project_id, employee_id) is the primary key of this table.
employee_id is a foreign key to Employee table.
Each row of this table indicates that the employee with employee_id is
working on the project with project_id.
```

Table: Employee

```
+-----+-----+
| Column Name | Type   |
+-----+-----+
| employee_id | int    |
| name        | varchar |
| experience_years | int   |
+-----+-----+
945 130 19 Online
```

Code

MySQL v Auto

```
1 SELECT
2   | p.project_id,
3   | ROUND(AVG(e.experience_years), 2) AS average_years
4   FROM Project p
5   JOIN Employee e
6   | ON p.employee_id = e.employee_id
7   GROUP BY p.project_id;
```

Saved

Ln 1, Col 1

Testcase | Test Result

Accepted Runtime: 88 ms

Case 1

Input

```
Project =
| project_id | employee_id |
| ----- | ----- |
| 1          | 1          |
| 1          | 2          |
| 1          | 3          |
| 2          | 1          |
| 2          | 4          |
```

Testcase | Test Result

```
Project =
| project_id | employee_id |
| ----- | ----- |
| 1          | 1          |
| 1          | 2          |
| 1          | 3          |
| 2          | 1          |
| 2          | 4          |
```

```
Employee =
| employee_id | name    | experience_years |
| ----- | ----- | ----- |
| 1          | Khaled  | 3          |
| 2          | Ali     | 2          |
| 3          | John    | 1          |
| 4          | Doe    | 2          |
```

Output

```
| project_id | average_years |
| ----- | ----- |
| 1          | 2          |
| 2          | 2.5        |
```

Output

```
| project_id | average_years |
| ----- | ----- |
| 1          | 2          |
| 2          | 2.5        |
```

Expected

```
| project_id | average_years |
| ----- | ----- |
| 1          | 2          |
| 2          | 2.5        |
```

Question #9

185. Department Top Three Salaries

Hard Topics Companies

SQL Schema > Pandas Schema >

Table: Employee

Column Name	Type
<code>id</code>	<code>int</code>
<code>name</code>	<code>varchar</code>
<code>salary</code>	<code>int</code>
<code>departmentId</code>	<code>int</code>

`id` is the primary key (column with unique values) for this table.
`departmentId` is a foreign key (reference column) of the ID from the `Department` table.
Each row of this table indicates the ID, name, and salary of an employee. It also contains the ID of their department.

Table: Department

Column Name	Type
<code>id</code>	<code>int</code>
<code>name</code>	<code>varchar</code>

MySQL

```
</code> Code
MySQL Auto

1 WITH Ranked AS (
2     SELECT
3         d.name      AS Department,
4         e.name      AS Employee,
5         e.salary    AS Salary,
6         DENSE_RANK() OVER (
7             PARTITION BY e.departmentId
8             ORDER BY e.salary DESC
9         ) AS salary_rank
)
Saved Ln 1, Col 1

Testcase > Test Result
Accepted Runtime: 123 ms

Case 1

Input
Employee =
| id | name | salary | departmentId |
| -- | ---- | ----- | ----- |
| 1 | Joe | 85000 | 1 |
| 2 | Henry | 80000 | 2 |
| 3 | Sam | 60000 | 2 |
| 4 | Max | 90000 | 1 |
```

185. Department Top Three Salaries

Hard Topics Companies

SQL Schema > Pandas Schema >

Table: Employee

Column Name	Type
id	int
name	varchar
salary	int
departmentId	int

id is the primary key (column with unique values) for this table.
departmentId is a foreign key (reference column) of the ID from the Department table.
Each row of this table indicates the ID, name, and salary of an employee. It also contains the ID of their department.

Table: Department

Column Name	Type
id	int
name	varchar

53 Online

```
-- 
  JOIN Department d
  ON e.departmentId = d.id
)
SELECT
  Department,
  Employee,
  Salary
FROM Ranked
WHERE salary_rank <= 3
ORDER BY Department, Employee, Salary DESC;
```

Saved Ln 1, Co

Testcase | Test Result

Accepted Runtime: 123 ms

Case 1

Input

Employee =

id	name	salary	departmentId
1	Joe	85000	1
2	Henry	80000	2
3	Sam	60000	2
4	Max	90000	1
5	Janet	69000	1

Ln 1, Co

Input

Employee =
id name salary departmentId
-- ----- ----- -----
1 Joe 85000 1
2 Henry 80000 2
3 Sam 60000 2
4 Max 90000 1
5 Janet 69000 1
6 Randy 85000 1

[View more](#)

Department =

id name
-- -----
1 IT
2 Sales



Output

Department Employee Salary
----- ----- -----
IT Max 90000
IT Joe 85000
IT Randy 85000
IT Will 70000
Sales Henry 80000
Sales Sam 60000



Expected

Department Employee Salary
----- ----- -----
IT Joe 85000
Sales Henry 80000
Sales Sam 60000
IT Max 90000
IT Randy 85000
IT Will 70000

