# Leet Code – SQL

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 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 an SQL query to report the **first login date** for each player.

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

The query result format is in the following example.

**select player\_id, event\_date as first\_login from (**

**select a.\*,**

**dense\_rank() over(partition by player\_id order by event\_date) as rnk**

**from Activity a) temp**

**where temp.rnk = '1';**

**607. Sales Person**

Easy

30954Add to ListShare

SQL Schema

Table: SalesPerson

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

| Column Name | Type |

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

| sales\_id | int |

| name | varchar |

| salary | int |

| commission\_rate | int |

| hire\_date | date |

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

sales\_id is the primary key column for this table.

Each row of this table indicates the name and the ID of a salesperson alongside their salary, commission rate, and hire date.

Table: Company

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

| Column Name | Type |

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

| com\_id | int |

| name | varchar |

| city | varchar |

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

com\_id is the primary key column for this table.

Each row of this table indicates the name and the ID of a company and the city in which the company is located.

Table: Orders

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

| Column Name | Type |

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

| order\_id | int |

| order\_date | date |

| com\_id | int |

| sales\_id | int |

| amount | int |

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

order\_id is the primary key column for this table.

com\_id is a foreign key to com\_id from the Company table.

sales\_id is a foreign key to com\_id from the SalesPerson table.

Each row of this table contains information about one order. This includes the ID of the company, the ID of the salesperson, the date of the order, and the amount paid.

Write an SQL query to report the names of all the salespersons who did not have any orders related to the company with the name **"RED"**.

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

The query result format is in the following example.

**select name from SalesPerson**

**where sales\_id not in (**

**select sales\_id from**

**Company join Orders using(com\_id)**

**where Company.name = 'RED'**

**);**

**1484. Group Sold Products By The Date**

Easy

31323Add to ListShare

SQL Schema

Table Activities:

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

| Column Name | Type |

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

| sell\_date | date |

| product | varchar |

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

There is no primary key for this table, it may contain duplicates.

Each row of this table contains the product name and the date it was sold in a market.

Write an SQL query to find for each date the number of different products sold and their names.

The sold products names for each date should be sorted lexicographically.

Return the result table ordered by sell\_date.

The query result format is in the following example.

**Example 1:**

**Input:**

Activities table:

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

| sell\_date | product |

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

| 2020-05-30 | Headphone |

| 2020-06-01 | Pencil |

| 2020-06-02 | Mask |

| 2020-05-30 | Basketball |

| 2020-06-01 | Bible |

| 2020-06-02 | Mask |

| 2020-05-30 | T-Shirt |

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

**Output:**

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

| sell\_date | num\_sold | products |

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

| 2020-05-30 | 3 | Basketball,Headphone,T-shirt |

| 2020-06-01 | 2 | Bible,Pencil |

| 2020-06-02 | 1 | Mask |

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

**Explanation:**

For 2020-05-30, Sold items were (Headphone, Basketball, T-shirt), we sort them lexicographically and separate them by a comma.

For 2020-06-01, Sold items were (Pencil, Bible), we sort them lexicographically and separate them by a comma.

For 2020-06-02, the Sold item is (Mask), we just return it.

**select sell\_date,**

**count(distinct product) as num\_sold,**

**group\_concat(distinct product order by product asc) as products**

**from Activities**

**group by sell\_date**

**order by sell\_date**

**608. Tree Node**

Medium

40332Add to ListShare

SQL Schema

Table: Tree

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

| Column Name | Type |

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

| id | int |

| p\_id | int |

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

id is the primary key column for this table.

Each row of this table contains information about the id of a node and the id of its parent node in a tree.

The given structure is always a valid tree.

Each node in the tree can be one of three types:

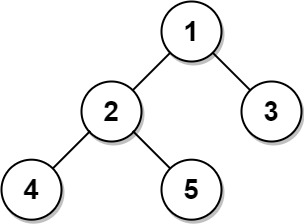
* **"Leaf"**: if the node is a leaf node.
* **"Root"**: if the node is the root of the tree.
* **"Inner"**: If the node is neither a leaf node nor a root node.

Write an SQL query to report the type of each node in the tree.

Return the result table **ordered** by id **in ascending order**.

The query result format is in the following example.

**Example 1:**



**Input:**

Tree table:

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

| id | p\_id |

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

| 1 | null |

| 2 | 1 |

| 3 | 1 |

| 4 | 2 |

| 5 | 2 |

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

**Output:**

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

| id | type |

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

| 1 | Root |

| 2 | Inner |

| 3 | Leaf |

| 4 | Leaf |

| 5 | Leaf |

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

**Explanation:**

Node 1 is the root node because its parent node is null and it has child nodes 2 and 3.

Node 2 is an inner node because it has parent node 1 and child node 4 and 5.

Nodes 3, 4, and 5 are leaf nodes because they have parent nodes and they do not have child nodes.

**select id,**

**case**

**when p\_id is null then 'Root'**

**when id in (select distinct p\_id from Tree) then 'Inner'**

**else 'Leaf'**

**end as type**

**from Tree;**

**1393. Capital Gain/Loss**

Medium

19521Add to ListShare

SQL Schema

Table: Stocks

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

| Column Name | Type |

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

| stock\_name | varchar |

| operation | enum |

| operation\_day | int |

| price | int |

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

(stock\_name, operation\_day) is the primary key for this table.

The operation column is an ENUM of type ('Sell', 'Buy')

Each row of this table indicates that the stock which has stock\_name had an operation on the day operation\_day with the price.

It is guaranteed that each 'Sell' operation for a stock has a corresponding 'Buy' operation in a previous day. It is also guaranteed that each 'Buy' operation for a stock has a corresponding 'Sell' operation in an upcoming day.

Write an SQL query to report the **Capital gain/loss** for each stock.

The **Capital gain/loss** of a stock is the total gain or loss after buying and selling the stock one or many times.

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

The query result format is in the following example.

**Example 1:**

**Input:**

Stocks table:

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

| stock\_name | operation | operation\_day | price |

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

| Leetcode | Buy | 1 | 1000 |

| Corona Masks | Buy | 2 | 10 |

| Leetcode | Sell | 5 | 9000 |

| Handbags | Buy | 17 | 30000 |

| Corona Masks | Sell | 3 | 1010 |

| Corona Masks | Buy | 4 | 1000 |

| Corona Masks | Sell | 5 | 500 |

| Corona Masks | Buy | 6 | 1000 |

| Handbags | Sell | 29 | 7000 |

| Corona Masks | Sell | 10 | 10000 |

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

**Output:**

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

| stock\_name | capital\_gain\_loss |

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

| Corona Masks | 9500 |

| Leetcode | 8000 |

| Handbags | -23000 |

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

**Explanation:**

Leetcode stock was bought at day 1 for 1000$ and was sold at day 5 for 9000$. Capital gain = 9000 - 1000 = 8000$.

Handbags stock was bought at day 17 for 30000$ and was sold at day 29 for 7000$. Capital loss = 7000 - 30000 = -23000$.

Corona Masks stock was bought at day 1 for 10$ and was sold at day 3 for 1010$. It was bought again at day 4 for 1000$ and was sold at day 5 for 500$. At last, it was bought at day 6 for 1000$ and was sold at day 10 for 10000$. Capital gain/loss is the sum of capital gains/losses for each ('Buy' --> 'Sell') operation = (1010 - 10) + (500 - 1000) + (10000 - 1000) = 1000 - 500 + 9000 = 9500$.

**# Write your MySQL query statement below**

**#select stock\_name, operation, sum(price)**

**#from Stocks**

**#group by stock\_name, operation**

**#order by stock\_name, operation;**

**with sell\_price as**

**(select stock\_name, sum(price) as price from Stocks**

**where operation = 'Sell'**

**group by stock\_name**

**),**

**buy\_price as**

**(select stock\_name, sum(price) as price from Stocks**

**where operation = 'Buy'**

**group by stock\_name**

**)**

**select stock\_name, sp.price - bp.price as capital\_gain\_loss**

**from sell\_price sp join buy\_price bp using(stock\_name)**