Tasks from LeetCode

https://leetcode.com/problemset/database/

Task 175 (LEFT JOIN).

SOLUTION:

Task 196 (DELETE duplicates)

SOLUTION:

Task 586 (SELECT TOP, MAX)

SOLUTION:

Task 577 (LEFT JOIN)

SOLUTION:

Task 175 (LEFT JOIN).

Table: Person

```
+--------------------+-----------+
| Column Name  | Type     |
+--------------------+-----------+
| personId         | int        |
| lastName         | varchar  |
| firstName        | varchar  |
+--------------------+------------+
```

personId is the primary key (column with unique values) for this table.
This table contains information about the ID of some persons and their
first and last names.

Table: Address

```
+-------------------+------------+
| Column Name  | Type     |
+-------------------+------------+
| addressId        | int        |
| personId         | int        |
| city               | varchar  |
| state              | varchar  |
+--------------------+-----------+
```

addressId is the primary key (column with unique values) for this table.
Each row of this table contains information about the city and state of
one person with ID = PersonId.

***Write a solution to report the first name, last name, city, and state
of each person in the Person table. If the address of a personId is
not present in the Address table, report null instead.***
Return the result table in any order.

SOLUTION:

```sql
SELECT p.firstname, p.lastname, ad.city, ad.state
FROM person AS p LEFT JOIN address AS ad
ON p.personid = ad.personid
```

Task 196 (DELETE duplicates)

Table: Person

```
+------------------+----------+
| Column Name | Type     |
+------------------+----------+
| id               | int      |
| email            | varchar  |
+------------------+----------+
```

id is the primary key (column with unique values) for this table.
Each row of this table contains an email. The emails will not contain uppercase letters.

Write a solution to delete all duplicate emails, keeping only one unique email with the smallest id.
For SQL users, please note that you are supposed to write a DELETE statement and not a SELECT one.

SOLUTION:

```sql
DELETE p FROM Person p, Person q
WHERE p.Id>q.Id AND p.email=q.email
```

Task 586 (SELECT TOP, MAX)

Table: Orders

```
+-----------------------+----------+
| Column Name           | Type     |
+-----------------------+----------+
| order_number          | int      |
| customer_number       | int      |
+-----------------------+----------+
```

order_number is the primary key (column with unique values) for this table.
This table contains information about the order ID and the customer ID.

Write a solution to find the customer_number for the customer who has placed the largest number of orders.
The test cases are generated so that exactly one customer will have placed more orders than any other customer.

SOLUTION:

```sql
SELECT TOP 1 MAX(customer_number) AS
customer_number
FROM orders
GROUP BY customer_number
ORDER BY customer_number DESC
```

Task 577 (LEFT JOIN)

Table: Employee

```
+-------------------+-----------+
| Column Name  | Type     |
+-------------------+-----------+
| empId            | int       |
| name             | varchar |
| supervisor       | int       |
| salary           | int       |
+-------------------+-------------+
```

empId is the column with unique values for this table.
Each row of this table indicates the name and the ID of an employee in addition to their salary and the id of their manager.

Table: Bonus

```
+-------------------+-------+
| Column Name  | Type |
+-------------------+-------+
| empId            | int   |
| bonus            | int   |
+-------------------+-------+
```

empId is the column of unique values for this table.
empId is a foreign key (reference column) to empId from the Employee table.
Each row of this table contains the id of an employee and their respective bonus.

Write a solution to report the name and bonus amount of each employee with a bonus less than 1000.
Return the result table in any order.

SOLUTION:

```sql
SELECT e.name, b.bonus
FROM employee e LEFT JOIN bonus b ON
e.empId=b.empId
WHERE b.bonus<1000 OR b.bonus is null
```