Section1 : Select

- **Problem Name:** 1757. Recyclable and Low Fat Products
- **Problem Link:** LeetCode URL
- **SQL Solution:**

```
SELECT product_id
FROM Products
WHERE low_fats = 'Y' AND recyclable = 'Y';
```

- **Problem Name:** 584. Find Customer Referee
- **Problem Link:** LeetCode URL
- **SQL Solution:**

```
SELECT name
FROM Customer
WHERE referee_id != 2 OR referee_id IS NULL;
```

Section2 : Basic Joins

- **Problem Name:** 1378. Replace Employee ID With The Unique Identifier
- **Problem Link:** LeetCode URL
- **SQL Solution:**

```
SELECT i.unique_id, e.name
FROM Employees e
LEFT JOIN EmployeeUNI i
ON e.id = i.id;
```

- **Problem Name:** 1068. Product Sales Analysis I
- **Problem Link:** LeetCode URL
- **SQL Solution:**

```
SELECT product_name, year, price
FROM Sales
JOIN Product
ON Sales.product_id = Product.product_id;
```

Section3 : Basic Aggregate Functions

- **Problem Name:** 620. No Boring Movies
- **Problem Link:** LeetCode URL
- **SQL Solution:**

```
SELECT id, movie, description, rating
FROM Cinema
WHERE id % 2 = 1
AND description!='boring'
ORDER BY rating DESC;
```

- **Problem Name:** 1251. Average Selling Price
- **Problem Link:** LeetCode URL
- **SQL Solution:**

```
SELECT p.product_id,
ROUND(IFNULL(SUM(u.units*p.price)/SUM(u.units), 0), 2) AS
average_price
FROM Prices p
LEFT JOIN UnitsSold u
ON p.product_id = u.product_id
AND u.purchase_date BETWEEN start_date AND end_date
GROUP BY product_id
```

Section4 : Sorting and Grouping

- **Problem Name:** 2356. Number of Unique Subjects Taught By Each Teacher
- **Problem Link:** LeetCode URL
- **SQL Solution:**

```
SELECT teacher_id, count(DISTINCT subject_id) AS cnt
FROM Teacher
GROUP BY teacher_id;
```

- **Problem Name:** 1141. User Activity for the Past 30 Days I
- **Problem Link:** LeetCode URL
- **SQL Solution:**

```
SELECT activity_date as day, COUNT(DISTINCT user_id) AS
active_users
FROM Activity
WHERE activity_date BETWEEN '2019-06-28' AND '2019-07-27'
```

```
GROUP BY activity_date;
```

Section5 : Advanced Select and Joins

- **Problem Name:** 1731. The Number of Employees Which Report to Each Employee
- **Problem Link:** LeetCode URL
- **SQL Solution:**

```
SELECT m.employee_id, m.name, COUNT(e.employee_id) AS reports_count, ROUND(AVG(e.age)) AS average_age
FROM Employees e
JOIN Employees m ON e.reports_to = m.employee_id
GROUP BY m.employee_id, m.name
ORDER BY m.employee_id;
```

- **Problem Name:** 610. Triangle Judgement
- **Problem Link:** LeetCode URL
- **SQL Solution:**

```
SELECT x, y, z,
    CASE
        WHEN x+y>z AND y+z>x AND z+x>y THEN 'Yes'
        ELSE 'No'
    END AS triangle
FROM Triangle;
```

Section6 : Subqueries

- **Problem Name:** 1978. Employees Whose Manager Left the Company
- **Problem Link:** LeetCode URL
- **SQL Solution:**

```
SELECT e1.employee_id
FROM Employees e1
LEFT JOIN Employees e2 ON e1.manager_id = e2.employee_id
WHERE e1.salary < 30000 AND e1.manager_id IS NOT NULL
AND e2.employee_id IS NULL
ORDER BY e1.employee_id
```

- **Problem Name:** 626. Exchange Seats

- **Problem Link:** LeetCode URL
- **SQL Solution:**

```
SELECT
    CASE
        WHEN id % 2 = 1 AND id != (SELECT MAX(id) FROM Seat)
            THEN id + 1
        WHEN id % 2 = 0
            THEN id - 1
        ELSE id
    END AS id, student
FROM Seat
ORDER BY id ASC;
```

Section7 : Advanced String Functions / Regex / Clause

- **Problem Name:** 1667. Fix Names in a Table
- **Problem Link:** LeetCode URL
- **SQL Solution:**

```
SELECT user_id, CONCAT(
        UPPER(LEFT(name,  1)),
        LOWER(SUBSTRING(name, 2))) AS name
FROM Users
ORDER BY user_id;
```

- **Problem Name:** 1527. Patients With a Condition
- **Problem Link:** LeetCode URL
- **SQL Solution:**

```
SELECT patient_id, patient_name, conditions
FROM Patients
WHERE conditions LIKE 'DIAB1%'
     OR conditions LIKE '% DIAB1%';
```