# SQL interview

April 2025

LeetCode SQL Questions

## Easy

### Recyclable and Low Fat Products (1757)

Identify products that are both low fat and recyclable by filtering the `Products` table where `low_fats = 'Y'` and `recyclable = 'Y'`.

### Find Customer Referee (584)

Retrieve names of customers who were not referred by the customer with `id = 2`, considering cases where `referee_id` is `NULL` or not equal to 2.

### Big Countries (595)

Select countries with an area of at least 3,000,000 $km^2$ or a population of at least 25 million by applying appropriate `WHERE` conditions.

### Article Views I (1148)

Find authors who have viewed their own articles by joining the `Views` table with itself where `author_id = viewer_id`.

### Invalid Tweets (1683)

Identify tweets where the content length exceeds 15 characters by using the `LENGTH()` function in the `WHERE` clause.

### Replace Employee ID With The Unique Identifier (1378)

Join the `Employees` table with the `EmployeeUNI` table using a `LEFT JOIN` to replace employee IDs with their unique identifiers, showing `NULL` where no match exists.

## Product Sales Analysis I (1068)

Combine `Sales` and `Product` tables to list each sale's product name, year, and price by joining on `product_id`.

## Customer Who Visited but Did Not Make Any Transactions (1581)

Identify customers who visited but didn't make transactions by finding `visit_id`s in `Visits` not present in `Transactions`.

## Rising Temperature (197)

Find days where the temperature was higher than the previous day by comparing each day's temperature with the prior day's using a self-join or window function.

## Average Time of Process per Machine (1661)

Calculate the average processing time per machine by pairing `'start'` and `'end'` activities and computing the average duration.

## Employee Bonus (577)

List employees with bonuses less than 1000 by joining `Employee` and `Bonus` tables and filtering on the bonus amount.

## Students and Examinations (1280)

Count the number of times each student attended each exam by joining `Students`, `Subjects`, and `Examinations` tables and aggregating attendance.

## Not Boring Movies (620)

Select movies with odd IDs and descriptions not containing `'boring'`, ordering the results by rating in descending order.

## Average Selling Price (1069)

Compute the average selling price for each product by dividing the total revenue by the total quantity sold.

## Project Employees I (1075)

Report the average experience years of employees for each project by joining `Project` and `Employee` tables and calculating the average.

### Percentage of Users Attended a Contest (1633)

Calculate the percentage of users who attended at least one contest by dividing the number of distinct users in the `Contests` table by the total number of users.

### Number of Unique Subjects Taught by Each Teacher (2356)

Determine the count of unique subjects each teacher teaches by grouping the `Teacher` table by `teacher_id` and counting distinct `subject_id`s.

### User Activity for the Past 30 Days I (1141)

Count the number of users who performed activities in the past 30 days by filtering the `Activity` table based on the date and counting distinct users.

### Classes More Than 5 Students (596)

Identify classes with more than five students by grouping the `Courses` table by `class` and using the `HAVING` clause to filter.

### Find Followers Count (1729)

Count the number of followers each user has by grouping the `Followers` table by `user_id` and counting entries.

### Biggest Single Number (619)

Find the largest number that appears only once in the `MyNumbers` table by grouping and filtering for counts equal to one, then selecting the maximum.

### The Number of Employees Which Report to Each Employee (1731)

Count the number of direct reports each employee has by grouping the `Employee` table by `managerId` and counting.

### Primary Department for Each Employee (1789)

Assign each employee to their primary department by selecting the department with the highest priority or earliest assignment.

### Triangle Judgement (1757)

Determine if three lengths can form a triangle by checking if the sum of any two sides is greater than the third.

### Employees Whose Manager Left the Company (1731)

Find employees whose managers are no longer with the company by identifying `managerId`s not present in the `Employee` table.

### Fix Names in a Table (1667)

Standardize the names in the `Users` table by capitalizing the first letter and making the rest lowercase using string functions.

### Patients With a Condition (1527)

Select patients whose conditions contain the word `'DIAB1'` by using the `LIKE` operator with appropriate wildcards.

### Delete Duplicate Emails (196)

Remove duplicate email entries by identifying and deleting records where the email appears more than once, keeping only the one with the smallest `id`.

### Group Sold Products By The Date (1484)

Group products sold by date by aggregating the `Sales` table based on the `sale_date` and listing products sold on each date.

### List the Products Ordered in a Period (1327)

List products ordered within a specific date range by filtering the `Orders` table based on the `order_date`.

## Medium

### Customers Who Bought All Products (1045)

Identify customers who have purchased every product available by comparing their purchases to the full list of product IDs. Use `GROUP BY` and `HAVING` to match the count of distinct products per customer against the total product count.

### Consecutive Numbers (180)

Find IDs with three or more consecutive entries with the same number by self-joining the table with offset rows. Check that `num` values are equal across three consecutive IDs to detect patterns.

### Product Price at a Given Date (1164)

Retrieve the latest price of products before a specified date using `WHERE` and `ORDER BY` with `LIMIT 1` per product. This involves filtering records prior to the given date and selecting the most recent price for each product.

### Last Person to Fit in the Bus (1747)

From a list of people with weights and queue order, determine the last person who can fit in the bus before reaching the weight limit. Use cumulative sums to simulate boarding and identify the last person who fits within the total weight capacity.

### Count Salary Categories (1633)

Categorize employees by salary ranges (`Low`, `Average`, `High`) and count the number in each category. Use a `CASE` statement within `COUNT` to split the data into the defined ranges.

### Exchange Seats (626)

Swap the seating positions of students in adjacent pairs while keeping the last student in place if there's an odd number. Use a self-join and modulo logic to conditionally swap seat numbers.

### Movie Rating (1341)

Find users with the highest average movie rating and the movies with the highest average ratings by joining user/movie/rating tables. Group by user or movie and use `ORDER BY` with `LIMIT 1` to get top results.

### Restaurant Growth (1322)

Track new customers visiting a restaurant each day and compute their cumulative total using `DISTINCT` and a running sum. Useful for understanding customer acquisition trends over time.

### Friend Requests II: Who Has the Most Friends (1789)

Count how many unique friend requests each person has made or received, and find the user with the most. Combine sender and receiver roles and group by user to count total interactions.

### Investments in 2016 (1741)

Calculate total investments grouped by month for the year 2016. Use `DATE_FORMAT` or equivalent functions to extract months and group financial data.

# Hard

## Department Top Three Salaries (185)

Return each department's top three highest-paid employees using a dense ranking partitioned by department. Apply the `DENSE_RANK()` window function and filter on rank values $\leq$ `3`.