SQL Concepts for Leetcode-Style Problems

Combine Two Tables

-> Learn basic SQL JOIN with LEFT JOIN.

Second Highest Salary

-> Use LIMIT, OFFSET, and DISTINCT for ranking.

Nth Highest Salary

-> Apply DENSE_RANK() or correlated subquery.

Rank Scores

-> Learn RANK() window function.

Consecutive Numbers

-> Use LEAD() or self join for sequence detection.

Employees Earning More Than Their Managers

-> JOIN on manager-employee relation.

Duplicate Emails

-> Find duplicates with GROUP BY HAVING COUNT > 1.

Customers Who Never Order

-> Use NOT IN or LEFT JOIN ... IS NULL.

Department Highest Salary

-> Combine JOIN + GROUP BY MAX.

Department Top Three Salaries

-> Master DENSE_RANK() + partitioning.

Delete Duplicate Emails

-> Use DELETE with ROW_NUMBER() or subqueries.

Rising Temperature

-> Compare rows using LAG() or self join.

Trips and Users

-> Filter with LEFT JOIN and conditions.

Game Play Analysis I

-> Basic filtering with MIN() per player.

Game Play Analysis II

-> Use DATEDIFF() and player grouping.

Game Play Analysis III

-> GROUP BY player for max levels.

Game Play Analysis IV

-> Conditional aggregation (CASE WHEN) and early activity.

Median Employee Salary

-> Apply window functions to compute median.

Managers with >= 5 Direct Reports

-> GROUP BY + HAVING COUNT >= 5.

Median from Frequency Table

-> Use cumulative frequency logic.

Winning Candidate

-> Find max votes using GROUP BY and ORDER BY.

Employee Bonus

-> LEFT JOIN and handle NULL bonus.

Highest Answer Rate Question

-> Use JOIN and max ratio calculation.

Cumulative Salary of Employee

-> Use recursive CTEs.

Count Students per Department

-> Combine GROUP BY and JOIN.

Find Customer Referee

-> Use IS NULL and != filters.

Investments in 2016

-> Use YEAR() function and conditionals.

Most Orders by a Customer

-> Use COUNT() + ORDER BY DESC LIMIT 1.

Big Countries

-> Simple WHERE condition filters.

Classes More Than 5 Students

-> Group filtering using HAVING.

Friend Request Acceptance Rate

-> Use COUNT() and compute ratios.

Human Traffic of Stadium

-> Consecutive id + JOIN logic.

Most Friends

-> COUNT + GROUP BY + max.

Consecutive Available Seats

-> id and seat type logic.

Sales Person

-> Exclude sellers with NOT IN subquery.

Tree Node

-> Identify Root, Leaf, and Inner with LEFT JOIN.

Triangle Judgement

-> Use condition a + b > c for triangle.

Shortest Distance in a Plane

-> MIN() of distances.

Shortest Distance in a Line

-> Row-wise pairwise difference.

Second Degree Follower

-> Self joins and indirect follower path.

Avg Salary Dept vs Company

-> Use AVG(), GROUP BY, and filtering.

Students Report Geography

-> Complex joins and conditions.

Biggest Single Number

-> GROUP BY HAVING COUNT = 1.

Not Boring Movies

-> Multiple filters with WHERE.

Exchange Seats

-> Use MOD() and conditional CASE swaps.

Swap Salary

-> Use UPDATE with CASE.

Customers Who Bought All Products

-> Use NOT EXISTS logic.

Actor-Director >= 3 Co-operations

-> GROUP BY HAVING COUNT >= 3.

Product Sales Analysis I

-> Simple JOIN query.

Product Sales Analysis II

-> Aggregate and filter by product_id.