# Leetcode SQL Prep

## Easy

### Find Customer Referee

**Handling NULL Values using filters**

Find the names of the customer that are not referred by the customer with id = 2

Customer table:

SELECT

name

FROM Customer

WHERE referee\_id != 2 OR referee\_id IS NULL

## Medium

### Confirmation Rate

Definition: number of 'confirmed' messages / by the total number of requested confirmation messages (‘confirmed’ + ‘timeout’)

**COALESCE, LEFT JOIN**

If the column or expression contains NULL values, it will substitute them with the first non-null value provided. If 0 was used as the fallback in the COALESCE function, it will return 0 when all other values are NULL.

SELECT

s.user\_id

,ROUND(coalesce(n.confirmed/d.requested,0),2) AS confirmation\_rate

FROM Signups s

LEFT JOIN

(

SELECT user\_id, SUM(1) AS confirmed

FROM Confirmations

WHERE action = "confirmed"

GROUP BY user\_id

) n ON s.user\_id = n.user\_id

LEFT JOIN

(

SELECT user\_id, SUM(1) AS requested

FROM Confirmations

GROUP BY user\_id

) d ON s.user\_id = d.user\_id

### Customers who bought all products

Write a solution to report the customer ids from the Customer table that **bought all the products in the Product table.**

Approach: Read english in the question

1. Count total number of unique products in product table
2. Use grouping to get unique product count from customer table
3. Check if customer has bought all products: If customers has bought all the products then then count of distinct product must match total number of unique products in product table

SELECT

customer\_id

FROM customer

GROUP BY customer\_id

HAVING COUNT(DISTINCT product\_key) = (SELECT COUNT(DISTINCT product\_key) FROM Product)

## Hard

### Department Top 3 Salaries

Write a solution to find the employees who are high earners in each of the departments.

Rank v/s Dense\_rank

| Rank | Dense Rank |
| --- | --- |
| Skips rank | Does not skip any rank |
| When there are ties, it assigns the same rank to tied rows, but it skips next ranks.  Eg: If two rows tie for rank 1, the next row will get rank 3 | When there are ties, it assigns the same rank to tied rows, **but it does not skip any rank.** The next row will get the next consecutive rank |

Approach

1. Rank employees by salary in each department using partition by
2. Filter out employees who have salaries ranked as 1,2 and 3 in respective departments

WITH base AS

(

SELECT

d.name AS Department

,e.name AS Employee

,salary

,DENSE\_RANK() OVER (PARTITION BY e.departmentid ORDER BY e.salary DESC) AS rnk

FROM

Employee e INNER JOIN Department d

ON e.departmentId = d.id

)

SELECT Department,Employee,salary

FROM base

WHERE rnk <= 3