# **Problem 1 (Meta Hard Level)**

A table named “famous” has two columns called user id and follower id. It represents each user ID has a particular follower ID. These follower IDs are also users of #Facebook / Meta. Then, find the famous percentage of each user.

Famous Percentage = number of followers a user has / total number of users on the platform.

**Explanation:**

1. distinct\_users CTE: Combines user\_id and follower\_id using UNION to get all unique users on the platform. This helps us determine the total number of users.
2. follower\_count CTE: Counts the number of followers for each user\_id by grouping the rows in the famous table. This gives a list of users with their follower counts.
3. Final SELECT Statement: Uses the data from follower\_count and distinct\_users to calculate the famous percentage for each user.

**MySQL Solution:**



**MSSQL Solution:**



# **Problem 2 (Amazon Hard Level)**

Given a table 'sf\_transactions' of purchases by date, calculate the month-over-month percentage change in revenue. The output should include the year-month date (YYYY-MM) and percentage change, rounded to the 2nd decimal point, and sorted from the beginning of the year to the end of the year. The percentage change column will be populated from the 2nd month forward and calculated as

.

**Explanation:**

1. MonthlyRevenue CTE: Aggregates the total revenue for each month using FORMAT to convert the created\_at date to the format YYYY-MM.
2. RevenueChange CTE: Adds a column previous\_revenue using the LAG function, which fetches the total revenue of the previous month for each row.
3. Final SELECT: Calculates the percentage change as ((total\_revenue - previous\_revenue) / previous\_revenue) \* 100. The ROUND function ensures the percentage is rounded to two decimal places. The output is ordered by year\_month to display the data chronologically.

**MSSQL Server Solution:**

****

**MySQL Solution:**

****

# **Problem 3 (Google Medium Level)**

You are analyzing a social network dataset at Google. Your task is to find mutual friends between two users, Karl and Hans. There is only one user named Karl and one named Hans in the dataset.  
  
The output should contain 'user\_id' and 'user\_name' columns.

**Explanation:**

1. The CTEs (karl\_friends and hans\_friends) efficiently find all friends for Karl and Hans, respectively.
2. The main query joins these CTEs with the users table to find the users who are present in both Karl's and Hans's friend lists (mutual friends).



# **Problem 4 (Uber Hard Level)**

Some forecasting methods are extremely simple and surprisingly effective. Naïve forecast is one of them. To create a naïve forecast for "distance per dollar" (defined as distance\_to\_travel / monetary\_cost), first sum the "distance to travel" and "monetary cost" values monthly. This gives the actual value for the current month. For the forecasted value, use the previous month's value. After obtaining both actual and forecasted values, calculate the root mean squared error (RMSE) using the formula

Report the RMSE rounded to two decimal places.



**MySQL Solution:**



**MSSQL Solution:**



# **Problem 5 (Microsoft Medium Level)**

Given a list of projects and employees mapped to each project, calculate by the amount of project budget allocated to each employee. The output should include the project title and the project budget rounded to the closest integer. Order your list by projects with the highest budget per employee first.

Explanation:

1. Joining Tables: The initial step involves joining the ms\_projects and ms\_emp\_projects tables on the project ID to combine project details (including titles and budgets) with employee assignments.
2. Grouping and Aggregating: The data is then grouped by project title and budget, allowing for the calculation of budget per employee by dividing the total budget of each project by the count of employees assigned to that project.
3. Rounding and Ordering: Finally, the computed budget per employee is rounded to the nearest integer, and the results are ordered in descending order to prioritize projects with the highest budget allocation per employee.

