SQL Aggregation Challenge: The Blunder

Problem Statement:

Samantha, a bright data analyst, faced a peculiar problem. Her keyboard's '0' key was broken, leading to a critical error in calculating average monthly salaries. She proceeded with her calculations, inadvertently removing all zeros from the salary figures.

> ### Task:
>
> - Find the **actual average salary.**
> - Find the **miscalculated average salary** (with zeros removed).
> - Calculate the **error (difference), rounded up** to the next integer.

```
Output = CEIL(actual_avg - miscalculated_avg)
```

Made with GAMMA

Unraveling "The Blunder": Step-by-Step Approach

1

Calculate Actual Average

Determine the true average salary using `AVG(salary)`.

2

Remove Zeros from Salaries

Convert salary to a string, then use `REPLACE(CAST(salary AS CHAR), '0', '')`.

3

Handle Empty Values

Use `NULLIF(..., '')` to convert empty strings (salaries that were only zeros) to NULL.

4

Convert Back to Numeric

Cast the cleaned string back to an unsigned integer: `CAST(... AS UNSIGNED)`.

5

Calculate Miscalculated Average

Compute the average of these modified salaries: `AVG(modified_salary)`.

6

Compute Final Error

Subtract and round up: `CEIL(actual_avg - miscalculated_avg)`.

Made with GAMMA

The Elegant SQL Solution (MySQL)

```
SELECT
  CEIL(AVG(salary) - AVG(CAST(NULLIF(REPLACE(CAST(salary AS CHAR), '0', ''), '') AS UNSIGNED))) AS error
FROM
  employees;
```

This concise query leverages several SQL functions to address each step of "The Blunder" challenge. It first computes the actual average, then meticulously transforms the salary data to simulate Samantha's miscalculation, and finally determines the difference, rounding up as required.

Pro Tip: Always test your SQL queries with diverse datasets, especially edge cases, to ensure robustness.

Made with GAMMA

Key Concepts & Learnings from the Challenge

CAST

Crucial for changing data types, such as converting numbers to strings for manipulation and back to numbers for calculations.

REPLACE

Used for removing specific characters (like '0' in this case) from a string, a common task in data cleaning.

NULLIF

Essential for preventing errors by converting empty strings (results of removing all characters) into NULL, which `AVG()` conveniently ignores.

AVG

The aggregate function to compute the arithmetic mean, intelligently ignoring NULL values.

CEIL

Ensures the final error is always rounded up to the nearest whole integer, as specified by the problem.

📌 Takeaway: Approaching complex SQL problems by breaking them into smaller, manageable transformations simplifies the solution process and improves readability.

Made with GAMMA