36_Tweets' Rolling Averages Medium - Solution

Source - https://datalemur.com/questions/rolling-average-tweets

Running Notes

- calculate the 3-day rolling average of tweets for each user.
- Output the user ID, tweet date, and rolling averages rounded to 2 decimal places
- In this case, we want to determine how the tweet count for each user changes over a 3-day period.

```
SELECT *,
AVG(tweet_count) OVER(
PARTITION BY user_id
)
FROM tweets
ORDER BY user_id, tweet_date;
```

The code uptill now just calculates the average tweets for each user but this is not rolling average, for rolling average I need to ORDER BY the tweet_date and also write a code for the “3-day” window

window functions perform complex calculations across sets of rows (termed as a “window”) related to the current row.

💡 **Window functions** are different from aggregate functions because the rows aren't grouped into a single output. In a window function, each row can remain separate, but the function has access to more than just the data in the current row.

💡 `ROWS BETWEEN` is used to specify a window frame in relation to the current row.

`ROWS BETWEEN` syntax:

```
OVER ([PARTITION BY <columns>] [ORDER BY <columns>] [ROWS BETWEEN <lower_bound> AND <upper_bound>])
```

The bounds in `ROWS BETWEEN` can be anyone of these five things:

- `UNBOUNDED PRECEDING`: All rows before the current row.
- `n PRECEDING`: *n* rows before the current row.
- `CURRENT ROW`: Just the current row.
- `n FOLLOWING`: *n* rows after the current row.
- `UNBOUNDED FOLLOWING`: All rows after the current row.

The goal is to calculate how a user's
tweet count changes over a 3-day period, focusing on the most recent 3 days (including the current day).
requirement : to see how things change
over the past 3 days

```
SELECT user_id,tweet_date,
ROUND(AVG(tweet_count) OVER(
PARTITION BY user_id
ORDER BY tweet_date
ROWS BETWEEN 2 PRECEDING AND CURRENT ROW
```

```
),2) AS rolling_avg_3d
FROM tweets
ORDER BY user_id, tweet_date;
```

Written By,
Harshee Pitroda