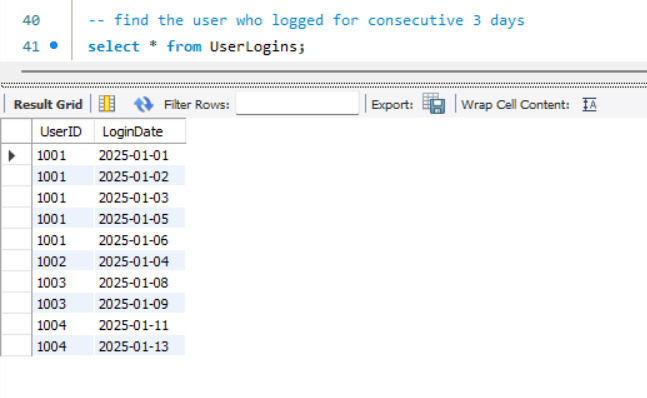
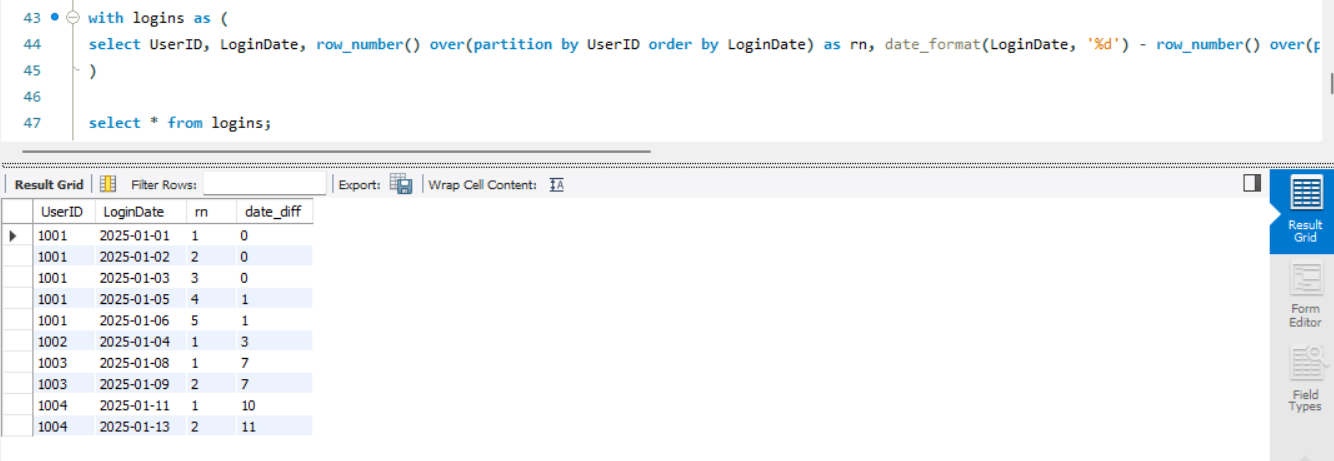
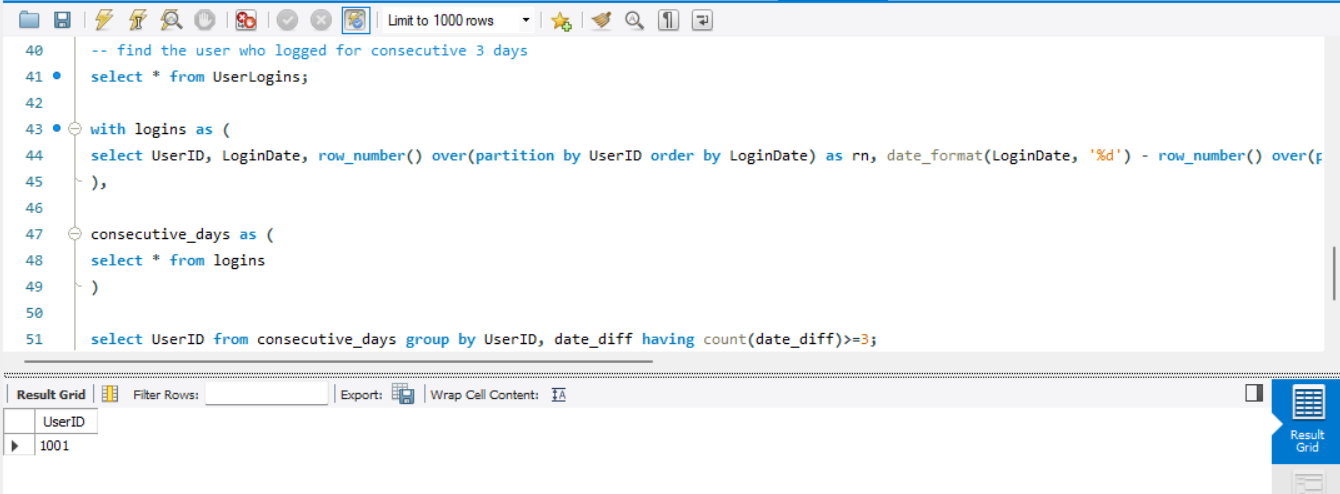
🔍 SQL Challenge: \*\*Find Users Who Logged In for at least 3 Consecutive Days\*\*  
  
💻 Querying with MySQL | Window Functions | Date Logic  
Recently, I worked on an interesting SQL scenario where I needed to identify users who logged in for 3 consecutive days. Here's how I tackled it using MySQL window functions and some smart logic. 👇   
  
🧩 Problem:   
Given a \*\*UserLogins\*\* table (📸 see Screenshot 1), identify users who have logged in for at least 3 consecutive days.   
  
🧠 Approach:  
I used a common SQL trick: if you subtract the row number from the login date (converted to day), consecutive logins will form groups with the same difference. Here's a breakdown:   
  
📌 Query Logic:   
```  
WITH logins AS (  
SELECT   
UserID,   
LoginDate,   
ROW\_NUMBER() OVER(PARTITION BY UserID ORDER BY LoginDate) AS rn,  
DATE\_FORMAT(LoginDate, '%d') - ROW\_NUMBER() OVER(PARTITION BY UserID ORDER BY LoginDate) AS date\_diff  
),  
consecutive\_days AS (  
SELECT \* FROM logins  
)  
SELECT UserID   
FROM consecutive\_days   
GROUP BY UserID, date\_diff   
HAVING COUNT(date\_diff) >= 3;   
```  
  
🔍 Breakdown:   
• CTE logins: Assigns a row number to each login per user based on date.  
• date\_diff logic: If logins are consecutive, the difference between date and row number remains the same.  
• Grouping by UserID and date\_diff, and filtering those with count >= 3 gives us users with 3+ consecutive logins.   
  
✅ Result: User 1001 logged in for at least 3 consecutive days! (📸 Screenshot 3)   
  
📷 Screenshots:   
🖼️ Screenshot 1: Sample Data (UserLogins table)

  
🖼️ Screenshot 2: Intermediate step showing row\_number and date\_diff



🖼️ Screenshot 3: Final result – user(s) who met the condition

  
  
💡This small yet powerful use-case shows how SQL window functions can be leveraged to solve temporal data challenges efficiently!   
  
🔗 Let me know your thoughts, or if you'd solve it differently!  
[**hashtag#SQL**](https://www.linkedin.com/search/results/all/?keywords=%23sql&origin=HASH_TAG_FROM_FEED) [**hashtag#MySQL**](https://www.linkedin.com/search/results/all/?keywords=%23mysql&origin=HASH_TAG_FROM_FEED) [**hashtag#WindowFunctions**](https://www.linkedin.com/search/results/all/?keywords=%23windowfunctions&origin=HASH_TAG_FROM_FEED) [**hashtag#DataEngineering**](https://www.linkedin.com/search/results/all/?keywords=%23dataengineering&origin=HASH_TAG_FROM_FEED) [**hashtag#ConsecutiveLogins**](https://www.linkedin.com/search/results/all/?keywords=%23consecutivelogins&origin=HASH_TAG_FROM_FEED) [**hashtag#SQLChallenge**](https://www.linkedin.com/search/results/all/?keywords=%23sqlchallenge&origin=HASH_TAG_FROM_FEED) [**hashtag#LearningByDoing**](https://www.linkedin.com/search/results/all/?keywords=%23learningbydoing&origin=HASH_TAG_FROM_FEED) [**hashtag#DataAnalytics**](https://www.linkedin.com/search/results/all/?keywords=%23dataanalytics&origin=HASH_TAG_FROM_FEED) [**hashtag#CodeWithMe**](https://www.linkedin.com/search/results/all/?keywords=%23codewithme&origin=HASH_TAG_FROM_FEED) [**hashtag#LinkedInLearning**](https://www.linkedin.com/search/results/all/?keywords=%23linkedinlearning&origin=HASH_TAG_FROM_FEED)