🚀 **SQL Challenge: Identify Users Who Added to Cart After a Page View on the Same Day**

🔎 Perfect for SQL interview prep or building analytical thinking!

Recently, I worked on an interesting SQL scenario simulating **e-commerce user behavior tracking**:

🎯 **Problem Statement:**

Given a table UserActivity with:

* UserID
* ActionType (Page\_View, Add\_to\_Cart, Purchase)
* ActionTimestamp

✅ The task: Find **UserIDs of customers who performed an 'Add\_to\_Cart' action strictly *after* a 'Page\_View' on the same day**. Each UserID should appear only once.

🛠️ **Approach:**  
1️ **Extract Page\_View events** into a CTE.  
2️ **Extract Add\_to\_Cart events** into another CTE.  
3️ **Join** them by UserID, ensuring both actions occurred **on the same day** (using date formatting on ActionTimestamp).  
4️ Filter where Add\_to\_Cart happened **strictly after** Page\_View.  
5️ Select **distinct UserIDs** to avoid duplicates.

```

WITH Page\_View\_Events AS (

SELECT UserID, ActionType, ActionTimestamp

FROM UserActivity

WHERE ActionType = 'Page\_View'

),

Add\_to\_Cart AS (

SELECT UserID, ActionType, ActionTimestamp

FROM UserActivity

WHERE ActionType = 'Add\_to\_Cart'

)

SELECT DISTINCT UserID

FROM Page\_View\_Events p

JOIN Add\_to\_Cart a

ON p.UserID = a.UserID

AND TO\_VARCHAR(p.ActionTimestamp, 'DD-MM-YYYY') = TO\_VARCHAR(a.ActionTimestamp, 'DD-MM-YYYY')

WHERE a.ActionTimestamp > p.ActionTimestamp;

```

🧠 **Key Takeaways:**   
✅ Mastering window functions and date manipulation is crucial in real-world SQL analytics.  
✅ Challenges like these teach **temporal event analysis**, a critical skill for BI, data science, and engineering roles.  
✅ This logic can directly power features like **funnel analysis** in e-commerce!

💬 Have you faced similar SQL challenges? Share your approach below!  
#SQL #SQLQueries #DataAnalytics #Ecommerce #PageView #AddToCart #LearningByDoing #SQLInterview #DataEngineering #PratikBawane #LinkedInLearning