📘 SQL 100 Days Challenge – Day 58 Reflection

✅ Day Overview

Today's practice was centered on **Banking Transactions & Fraud Detection**. The queries covered everything from transaction summaries and customer insights to fraud detection logic using **window functions, CTEs, and conditional checks**.

This session reinforced how SQL can be applied in **financial risk management** and **fraud monitoring systems**.

📝 Learnings & Key Highlights

1. **Aggregations**
    - Used COUNT, SUM, AVG, MIN, and MAX to summarize customer-level transactions.
    - Realized that COUNT(TransactionID) and COUNT(DISTINCT TransactionID) produce the same result here since TransactionID is a **Primary Key**.

2. **Withdrawal Analysis**
    - Identified accounts with **average withdrawal > 2000**.
    - Querying became straightforward with practice on grouping and filtering (HAVING).

3. **Geographic Insights**
    - Detected customers transacting in multiple countries using COUNT(DISTINCT Location) + STRING_AGG().

4. **Window Functions & CTEs**
    - Used RANK() to extract **largest transactions per account**.
    - Running balance calculation (Question 6) showed how deposits and withdrawals affect accounts over time.
    - CTE + window logic felt natural now due to consistent practice.

5. **Date Functions Challenge**
    - Got confused initially between DATEPART() and DATEDIFF() when calculating weekly withdrawals.
    - Eventually applied DATEPART(WEEK, TransDate) correctly.

6. **Fraud & Risk Detection**
    - Built queries for **high-risk withdrawals** (>70% of account balance).
    - Joined with **FraudAlerts** table to link flagged transactions with reasons.
    - Bonus Question (fraud risk accounts) was very challenging, but it gave deep insight into **complex fraud scenarios**:
        - 2+ fraud alerts in last 30 days
        - Transactions from 3+ cities in a month
        - Risk levels classified as *Extreme, High, Medium, Low*

💡 Reflection

- Consistency is truly paying off — writing complex CTEs and window queries feels much easier now.
- Handling **fraud detection** problems showed how SQL powers **real-world banking systems**.
- The **Bonus Question** was tough but rewarding — it made me think in terms of both **time windows** and **multi-condition logic**.
- Key lesson: **Break complex problems into smaller CTEs and solve step by step**.

🚀 What's Next?

- Strengthen handling of **date functions** (DATEADD, DATEDIFF, DATEPART) since they're crucial for time-based queries.
- Keep practicing **risk detection** and **customer profiling queries** — very useful for financial domain analytics.