SQL 100 Days Challenge – Day 54 Reflection

Topic: E-commerce Customer, Orders, and Payments Analytics

Dataset: Customers, Products, Orders, OrderDetails, Payments

Practice Experience:

- Today's set of queries felt **easy and straightforward**, giving me a chance to reinforce key concepts with confidence.
- Questions covered customer spending, product popularity, order status comparisons, monthly revenue, repeat customers, and payment methods.
- The **Bonus Challenge** compared **NOT EXISTS vs LEFT JOIN … IS NULL** approaches to find customers with no orders — a nice touch to understand query performance.

Key Learnings:

1. **Top Customers:** Used SUM + ORDER BY to rank by spending.
2. **Inactive Customers:** Detected with LEFT JOIN and null checks.
3. **AOV (Average Order Value):** Per-customer analysis using AVG().
4. **Product Popularity:** Identified most revenue-generating products.
5. **Completed vs Cancelled Orders:** Applied CASE WHEN to calculate counts and percentages.
6. **Monthly Revenue:** Grouped with YEAR, MONTH, and DATENAME for trend insights.
7. **Order Diversity:** Analyzed customers with orders across multiple statuses.
8. **Window Functions:** Used RANK() to list top products by revenue.
9. **Repeat Customers:** Identified with HAVING COUNT() > 1.
10. **Payment Method Analysis:** Found the highest-revenue method and average contribution.
11. **Bonus – Query Optimization:** Compared NOT EXISTS vs LEFT JOIN, understanding their trade-offs in clarity, performance, and flexibility.

Insights:

- **Alice (USA)** emerged as a top spender with repeat orders.
- **Electronics** (Laptop, Smartphone) dominated in revenue compared to Accessories or Stationery.
- **Credit Cards** and **PayPal** were strong contributors to revenue.
- Cancellation ratios were low, showing stable order flow.
- Both NOT EXISTS and LEFT JOIN approaches had their own strengths — performance and clarity vs flexibility.

Skills Reinforced:

- Aggregations (SUM, AVG, COUNT)
- CASE WHEN for percentages and classification
- Window functions (RANK)
- Date grouping (YEAR, MONTH, DATENAME)
- Query optimization thinking (NOT EXISTS vs LEFT JOIN)

Personal Note:
Today was **light yet productive**. The questions gave me a chance to refine SQL basics while practicing performance-oriented thinking. It was satisfying to apply logic smoothly without much struggle. The bonus exercise reminded me that writing queries isn't just about results, but also about **efficiency and maintainability**.

Next Steps:

- Deep dive into **index performance testing** on larger datasets.
- Explore hybrid queries mixing window functions with optimization.
- Compare query plans for NOT EXISTS vs LEFT JOIN on big tables.