📘 Advanced SQL – Lesson 2: Temporary Tables

🎯 What You'll Learn

In this lesson, you'll learn how to:

- Create and use **temporary tables (temp tables)**
- Insert data into temp tables from existing queries
- Optimize performance by caching intermediate results
- Manage temp tables across multiple query executions or stored procedures

---

🧠 What is a Temporary Table?

A **temporary table** is a short-lived table used to store data **only during your SQL session**. It works much like a normal table but is automatically deleted when your session ends.

Unlike CTEs or subqueries, temp tables:

- Can be reused multiple times across queries
- Live in memory or tempdb during execution
- Allow you to separate complex logic and improve query performance

---

✏️ Why Use Temp Tables?

Temp tables are incredibly helpful when:

- Working with **large datasets** or complex queries

- Reusing filtered or transformed data across multiple steps
- Reducing **repetition and execution time** in stored procedures or batch queries

💡 Use them to “cache” intermediate results and avoid repeating expensive calculations.

🛠️ Key Operations

- Create with `CREATE TABLE #TempTableName`
- Insert data manually or via `INSERT INTO ... SELECT`
- Query like a regular table (`SELECT * FROM #TempTable`)
- Drop when no longer needed (`DROP TABLE IF EXISTS`)

📂 Real-World Use Case

Imagine needing to join two large tables and calculate averages or counts — multiple times. Instead of recalculating those joins each time:

1. Store the result in a temp table
2. Query from the temp table in subsequent steps
3. Save time and improve performance

Also useful in **stored procedures**, where re-running a script could otherwise cause temp table conflicts — solved by adding a safety check like `DROP TABLE IF EXISTS`.

⚠️ Tips & Best Practices

- Always **drop** a temp table if it might already exist to avoid errors
- Use temp tables instead of repeating expensive joins or aggregations
- Know they are **session-specific** — they vanish after your query ends
- For persistent intermediate storage, consider using **permanent staging tables** instead

---

📌 Recap

✅ Temp tables act like real tables — but are **temporary**
✅ Store intermediate or transformed data to optimize performance
✅ Use in **multi-step logic**, especially in stored procedures
✅ Add `DROP TABLE IF EXISTS` to avoid conflicts during repeated runs