🎥 Video Summary: Solving Complex Data Problems with Derived Tables

---

🧠 What Is a Derived Table?

- A **derived table** is a **query nested inside another query**.
- It behaves like a **virtual table** created temporarily to support the **outer query**.
- It is **not stored** in the database and is **discarded** after the query runs.

🔍 Why Use Derived Tables?

- Useful when the **required data doesn't exist** in a single table.
- Helps **simplify complex queries** by breaking them into **modular parts**.
- Makes queries **easier to read, write, and maintain**.

📊 Example: Donor Analysis

- Liz, a data analyst, needs to identify:
  - **Megadonors**: donated **over $10,000** in the past year.
  - **Frequent donors**: donated **more than 3 times**, regardless of amount.
- She uses a **derived table** to:
  - Group donations by **donor ID**.
  - Aggregate **total amount** and **donation count**.
- The **outer query** then filters for megadonors and frequent donors using the derived table.

✅ Benefits of Derived Tables

1. **Simplifies complex logic** into manageable parts.
2. **Improves readability** and maintainability of queries.
3. **Supports advanced calculations** and custom views in visualization tools.
4. Enables **custom reports** and **data-driven insights**.

⚠️ Limitations of Derived Tables

1. **Performance impact**: Must be **recreated each time** the query runs.
2. **Not persistent**: Cannot be reused across queries unless redefined.
3. **Not stored**: Exists only during query execution.

📌 Final Takeaway

Derived tables are a **powerful tool** for cloud data analysts to solve complex problems. However, they should be used **strategically**, balancing **performance** with **business needs**.