**1. SELECT + CASE + Expressions**

1. Show all products along with a status column: 'Expensive' if price > 1000, 'Moderate' if between 500–1000, and 'Cheap' otherwise.

**🔹 2. Aggregate Functions + Filtering**

1. Find the top 2 suppliers (by total purchases) in each year using a window function.
2. For each customer, show their total spending and the number of different products purchased.

**🔹 3. GROUP BY + HAVING (Advanced)**

1. List users who handled more than 3 inventory actions in a single day.
2. Show product categories where the average rating across all their products is below 3.5.

**🔹 4. JOINs (Complex)**

1. List all products and the total quantity sold, including products that have never been sold (use LEFT JOIN).
2. Get a list of suppliers who never supplied products that were sold (join Purchases → Products → SalesDetails).

**🔹 5. Subqueries (Advanced)**

1. Find the name of the product with the **second highest average rating**.
2. List products that were **never purchased** and **never reviewed**.
3. Find warehouses that store **every product from category X** (relational division).

**🔹 6. Views**

1. Create a view that returns, for each product, total stock, total sold, and remaining quantity.
2. Create a view to show monthly purchase totals per supplier, including months with no purchases.

**🔹 7. Stored Procedures**

1. Create a procedure to get all sales by customer ID and optional date range.
2. Create a procedure to insert a review only if no previous review exists for that product and customer.

**🔹 8. Functions (Scalar & Table-Valued)**

1. Write a scalar function that returns 'High', 'Medium', or 'Low' based on stock quantity.
2. Create a table-valued function that returns all sales of a given product with customer names.

**🔹 9. Triggers**

1. Write a trigger that prevents inserting a purchase if the supplier is inactive.
2. Create a trigger to automatically update a "last\_updated" column when product price is modified.

**🔹 10. Indexes + Performance**

1. Create a covering index on SalesDetails(product\_id, sale\_id, price) to improve reporting queries.
2. Create a filtered index on Inventory\_Log(action) where action = 'OUT'.