# Codtech IT Solutions — SQL Internship Task 2

Task: Data Analysis with Complex Queries

Intern Name: Madhugna Kadem

Intern ID: CT04DN607

## 1. Objective

The objective of this task is to perform advanced data analysis using SQL. Specifically, the task requires the use of window functions, subqueries, and common table expressions (CTEs) to generate insightful reports that reveal trends or patterns in the data. This exercise will improve the understanding of advanced SQL operations that are commonly used in real-world data analysis and reporting.

## 2. Sample Dataset Description

To demonstrate the capabilities of advanced SQL, we will use a sample sales dataset. The dataset includes two tables:  
  
- sales: Contains sales transactions including sale\_id, product\_id, quantity, sale\_date, and amount.  
- products: Contains product information including product\_id, product\_name, and category.

## 3. SQL Queries and Analysis

### SQL to Create Tables and Insert Sample Data:

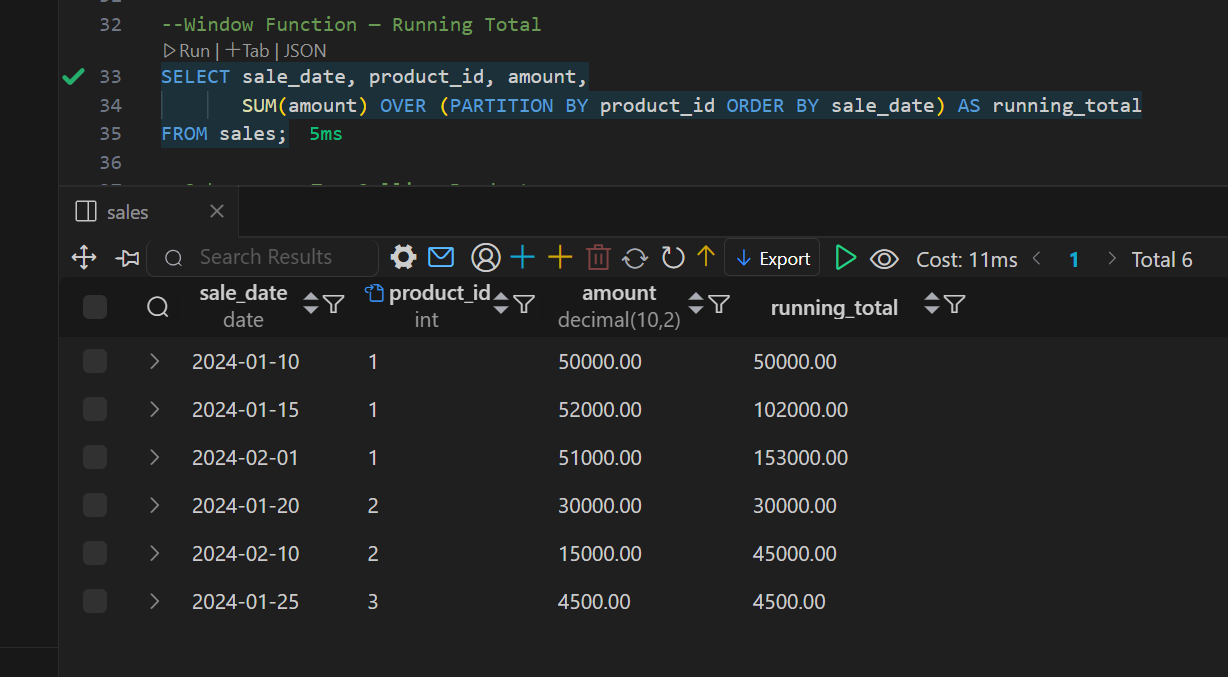
-- Create tables  
CREATE TABLE products (  
 product\_id INT PRIMARY KEY,  
 product\_name VARCHAR(50),  
 category VARCHAR(50)  
);  
  
CREATE TABLE sales (  
 sale\_id INT PRIMARY KEY,  
 product\_id INT,  
 quantity INT,  
 sale\_date DATE,  
 amount DECIMAL(10,2),  
 FOREIGN KEY (product\_id) REFERENCES products(product\_id)  
);  
  
-- Insert sample data into products  
INSERT INTO products VALUES  
(1, 'Laptop', 'Electronics'),  
(2, 'Mobile', 'Electronics'),  
(3, 'Shoes', 'Footwear');  
  
-- Insert sample data into sales  
INSERT INTO sales VALUES  
(101, 1, 1, '2024-01-10', 50000.00),  
(102, 1, 1, '2024-01-15', 52000.00),  
(103, 2, 2, '2024-01-20', 30000.00),  
(104, 3, 3, '2024-01-25', 4500.00),  
(105, 1, 1, '2024-02-01', 51000.00),  
(106, 2, 1, '2024-02-10', 15000.00);

### Window Function: Running Total of Sales

SELECT sale\_date, product\_id, amount,  
 SUM(amount) OVER (PARTITION BY product\_id ORDER BY sale\_date) AS running\_total  
FROM sales;

Explanation: This query calculates the running total of sales amount for each product over time using the window function SUM() OVER(). It helps in identifying cumulative revenue trends.

Output Screenshot:

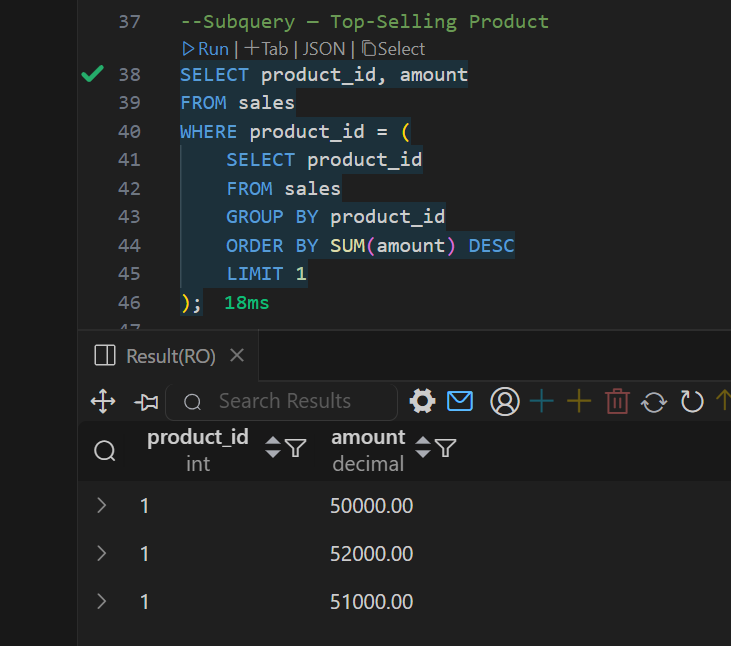


### Subquery: Top-Selling Product by Total Sales

SELECT product\_id, amount  
FROM sales  
WHERE product\_id = (  
 SELECT product\_id  
 FROM sales  
 GROUP BY product\_id  
 ORDER BY SUM(amount) DESC  
 LIMIT 1  
);

Explanation: This subquery identifies the product with the highest total sales. It is useful for targeting bestsellers and allocating marketing resources accordingly.

Output Screenshot:

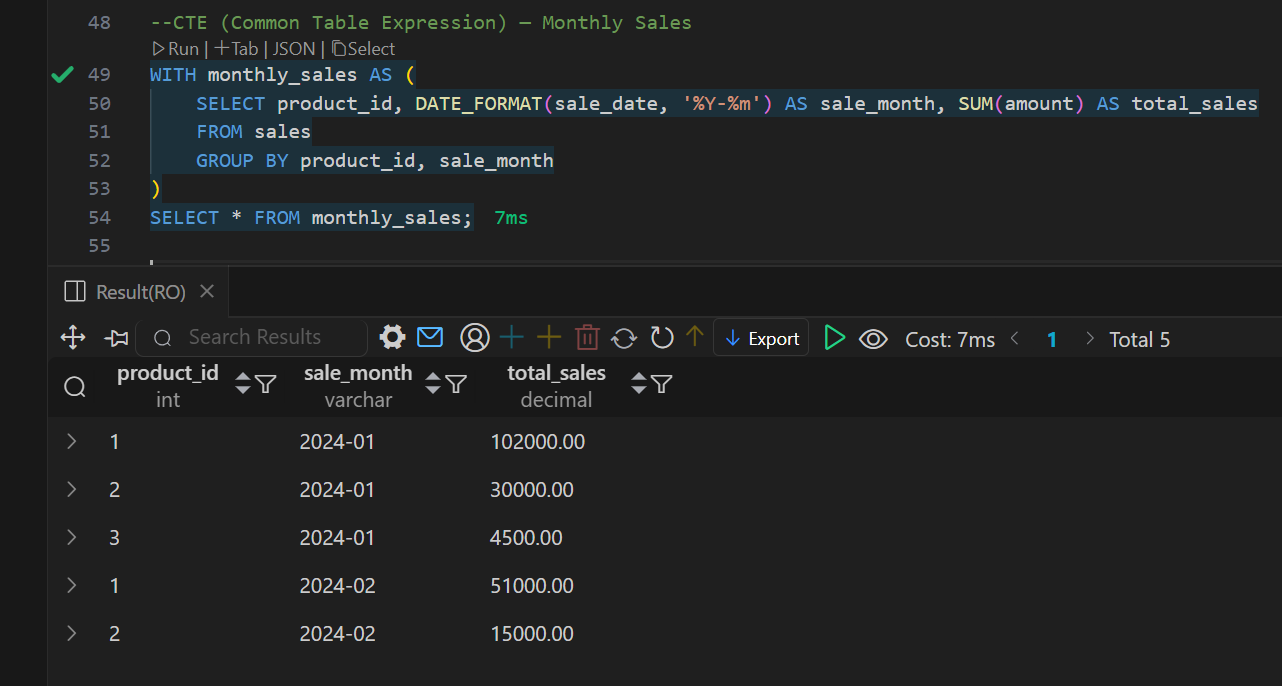


### Common Table Expression (CTE): Monthly Sales per Product

WITH monthly\_sales AS (  
 SELECT product\_id, DATE\_FORMAT(sale\_date, '%Y-%m') AS sale\_month, SUM(amount) AS total\_sales  
 FROM sales  
 GROUP BY product\_id, sale\_month  
)  
SELECT \* FROM monthly\_sales;

Explanation: This CTE simplifies the query logic by precomputing monthly sales. The result can be used for visualizing monthly sales patterns or further aggregated analysis.

Output Screenshot:



## 4. Summary

This task helped demonstrate the power of advanced SQL techniques in uncovering data insights. Window functions are great for trends over time, subqueries help filter complex conditions, and CTEs allow for readable and reusable logic. These queries form the foundation for complex reports and dashboards used in data analysis projects.