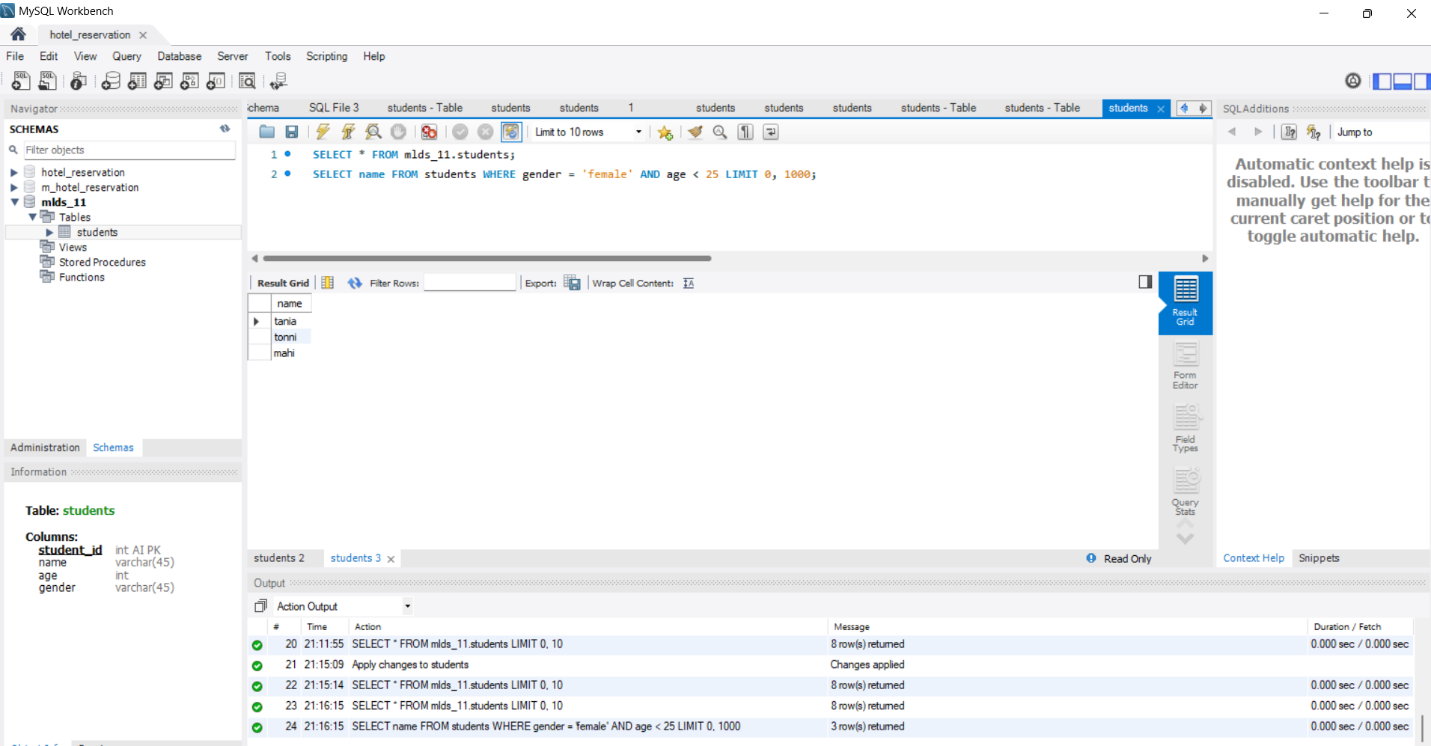
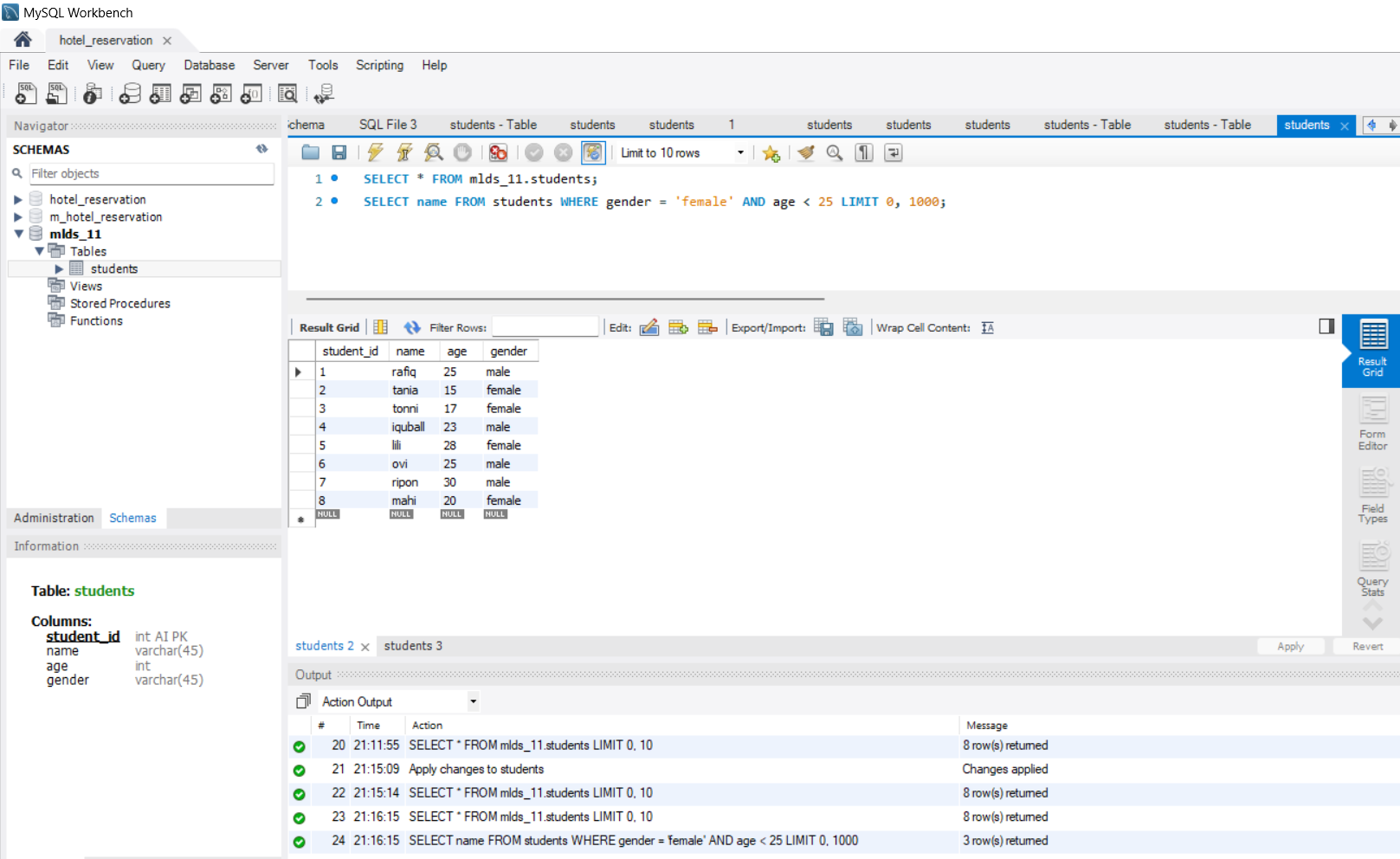
**Problem 1: SELECT and WHERE**

**Ques: You have a table named `students` with columns `student\_id`, `name`, `age`, and `gender`. Write a query to select the names of female students who are below 25 years of age.**

SELECT name

FROM students WHERE gender = ‘female’ AND age<25;

**Problem 2: ORDER BY, GROUP BY, and AGGREGATE FUNCTIONS**

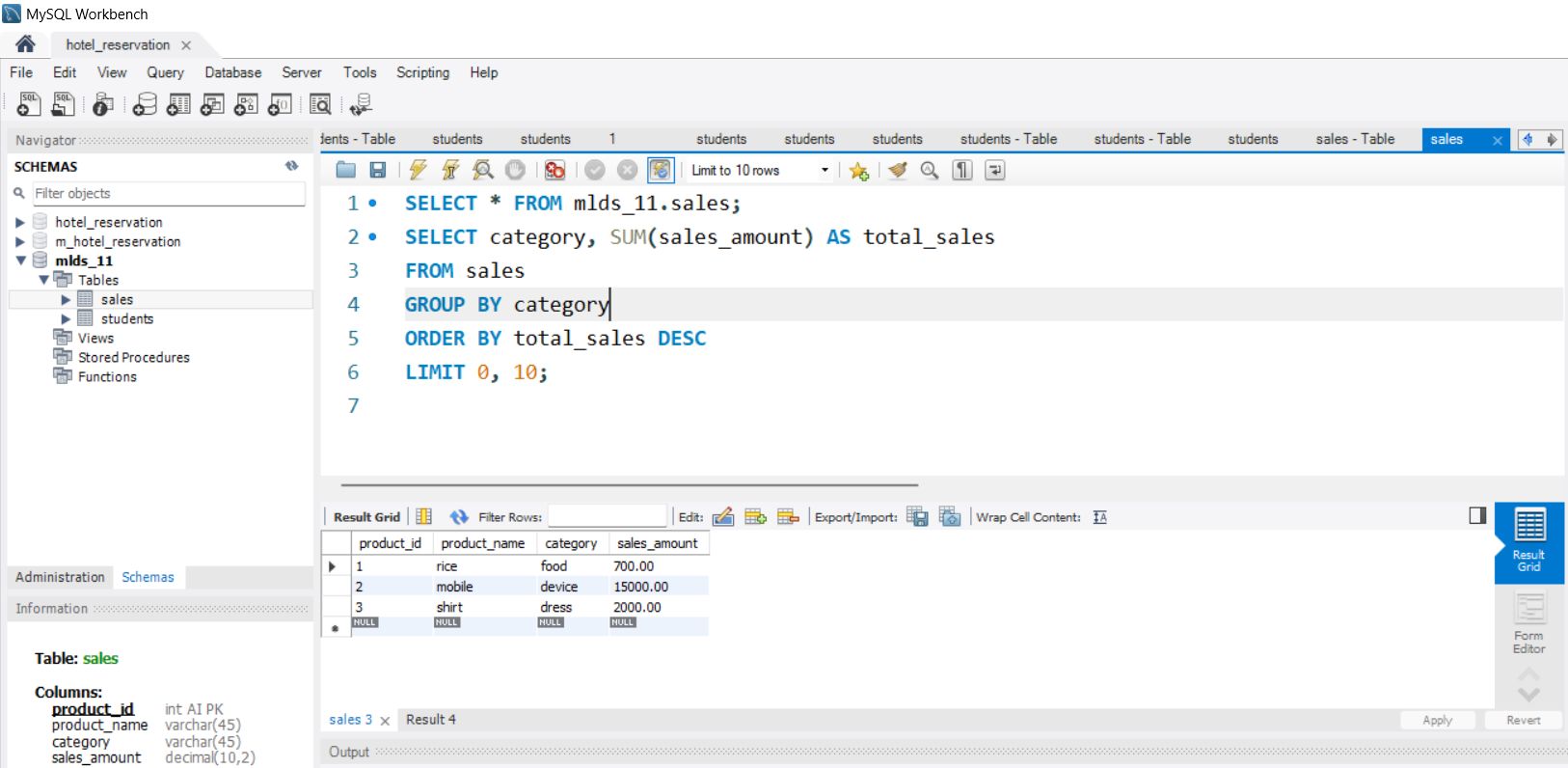
**Ques: Consider a table named `sales` with columns `product\_id`, `product\_name`, `category`, and `sales\_amount`. Write a query to find the total sales amount for each category, and display the results in descending order of total sales amount.**

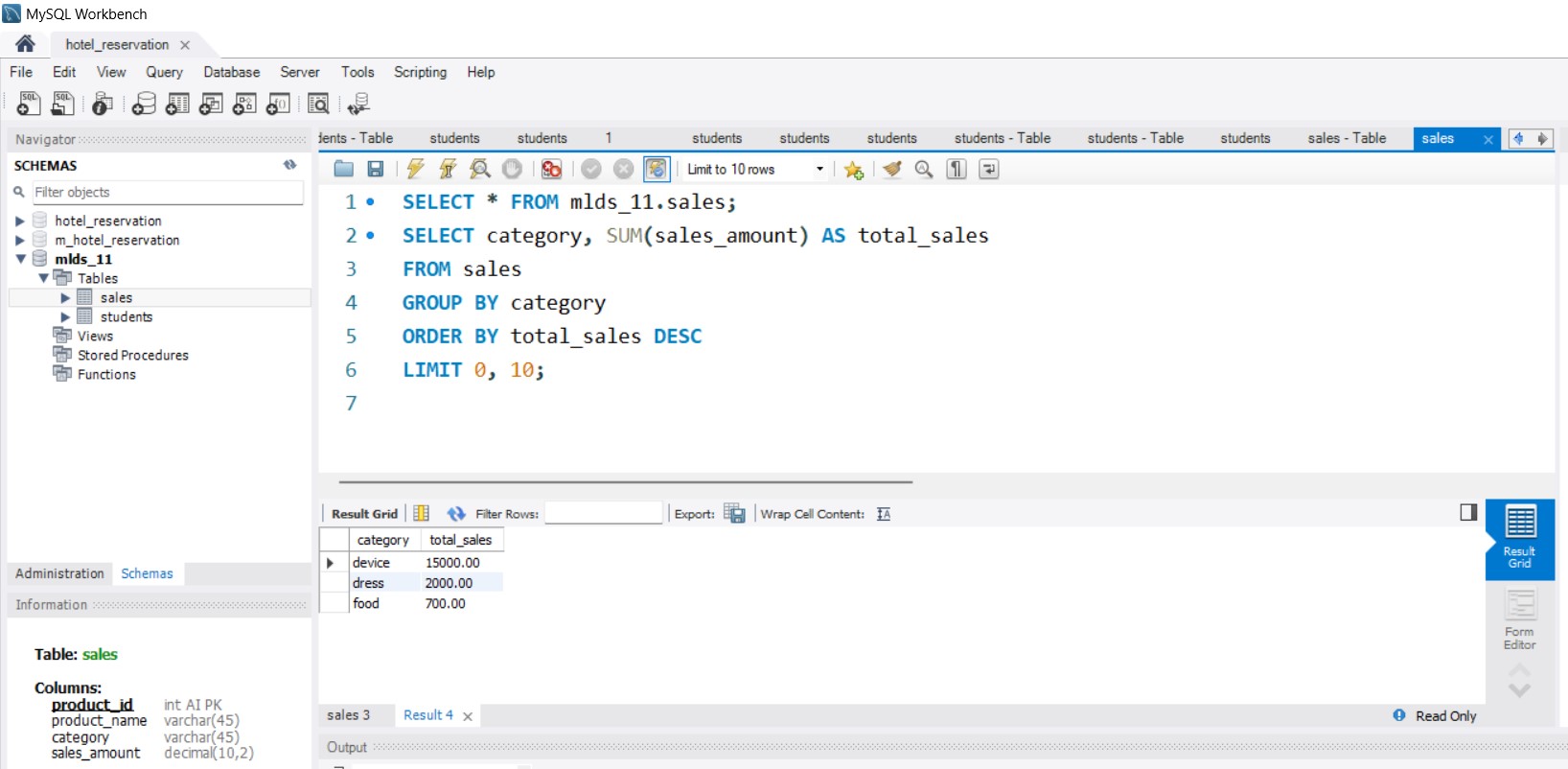
SELECT category, SUM (sales\_amount) AS total\_sales

FROM sales

GROUP BY category

ORDER BY total\_sales DESC;





**Problem 3: JOIN, WHERE, and LIKE**

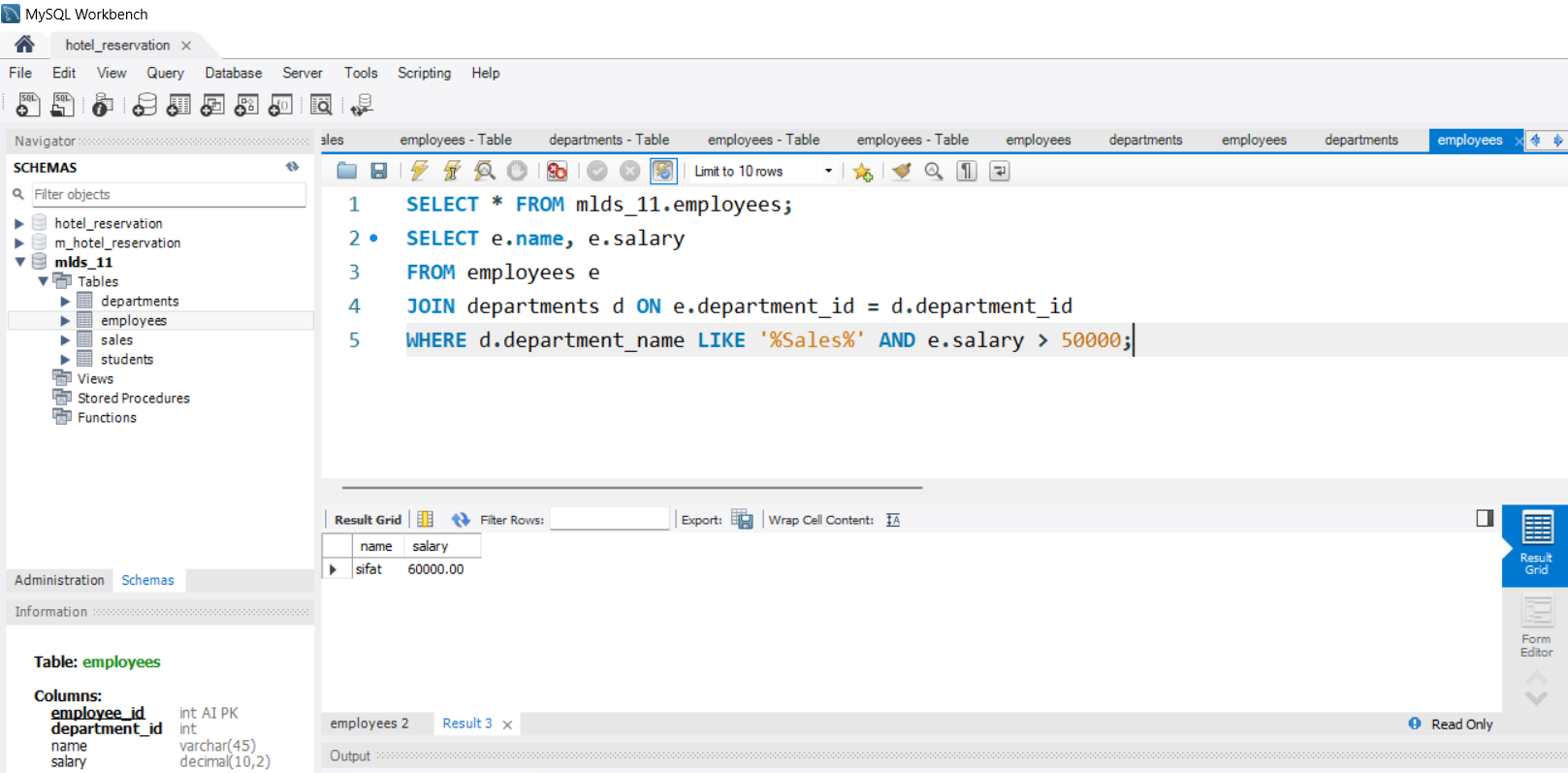
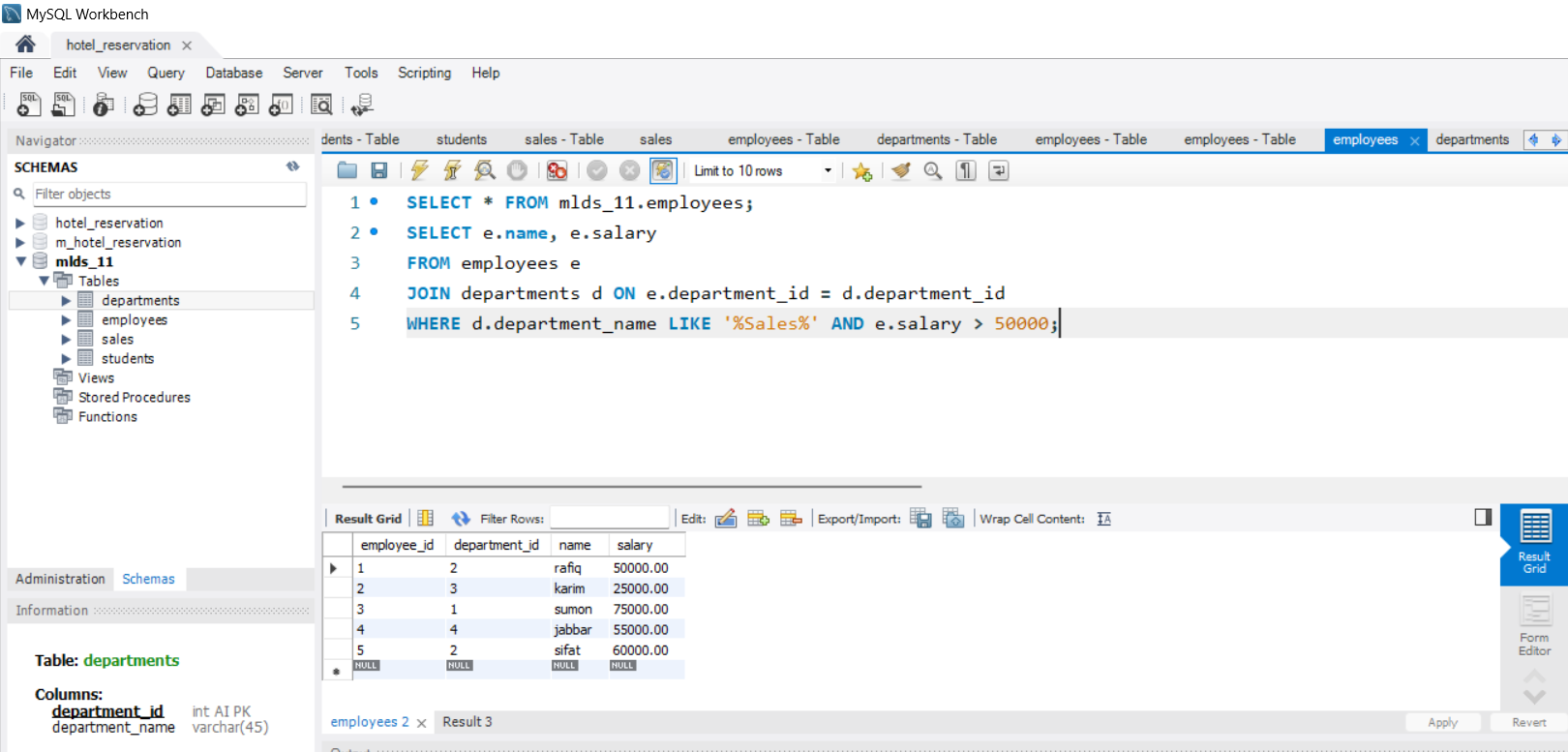
**Ques: Suppose you have two tables named `employees` and `departments`, where `employees` contains columns `employee\_id`, `name`, `department\_id`, and `salary`, and `departments` contains columns `department\_id` and `department\_name`. Write a query to select the names and salaries of employees who work in the 'Sales' department and whose salaries are greater than $50000.**

SELECT e.name, e.salary

FROM employees e

JOIN departments d ON e.department\_id = d.department\_id

WHERE d.department\_name LIKE '%Sales%' AND e.salary > 50000;

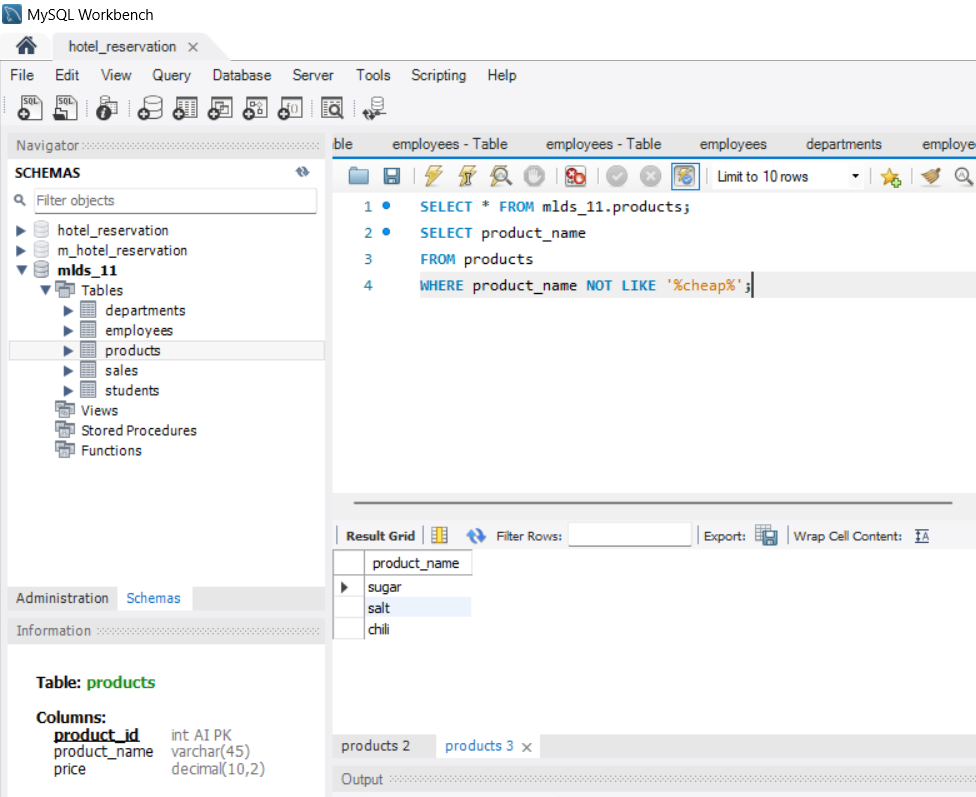
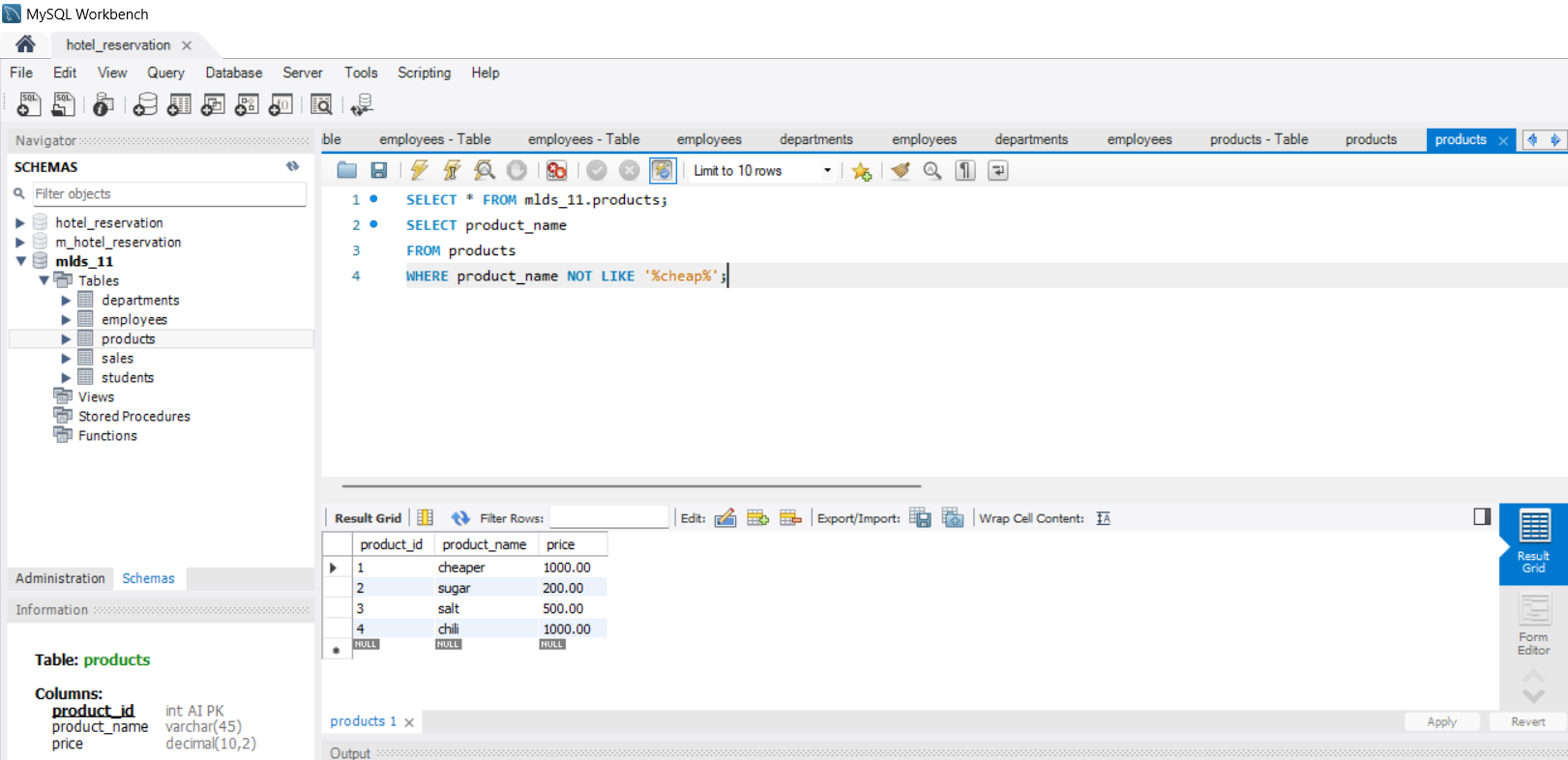


**Problem 4: NOT, Wildcards, and LIKE**

**Ques: Assume you have a table named `products` with columns `product\_id`, `product\_name`, and `price`. Write a query to select the names of products that do not contain the word 'cheap' in their names.**

SELECT product\_name

FROM products

WHERE product\_name NOT LIKE '%cheap%';

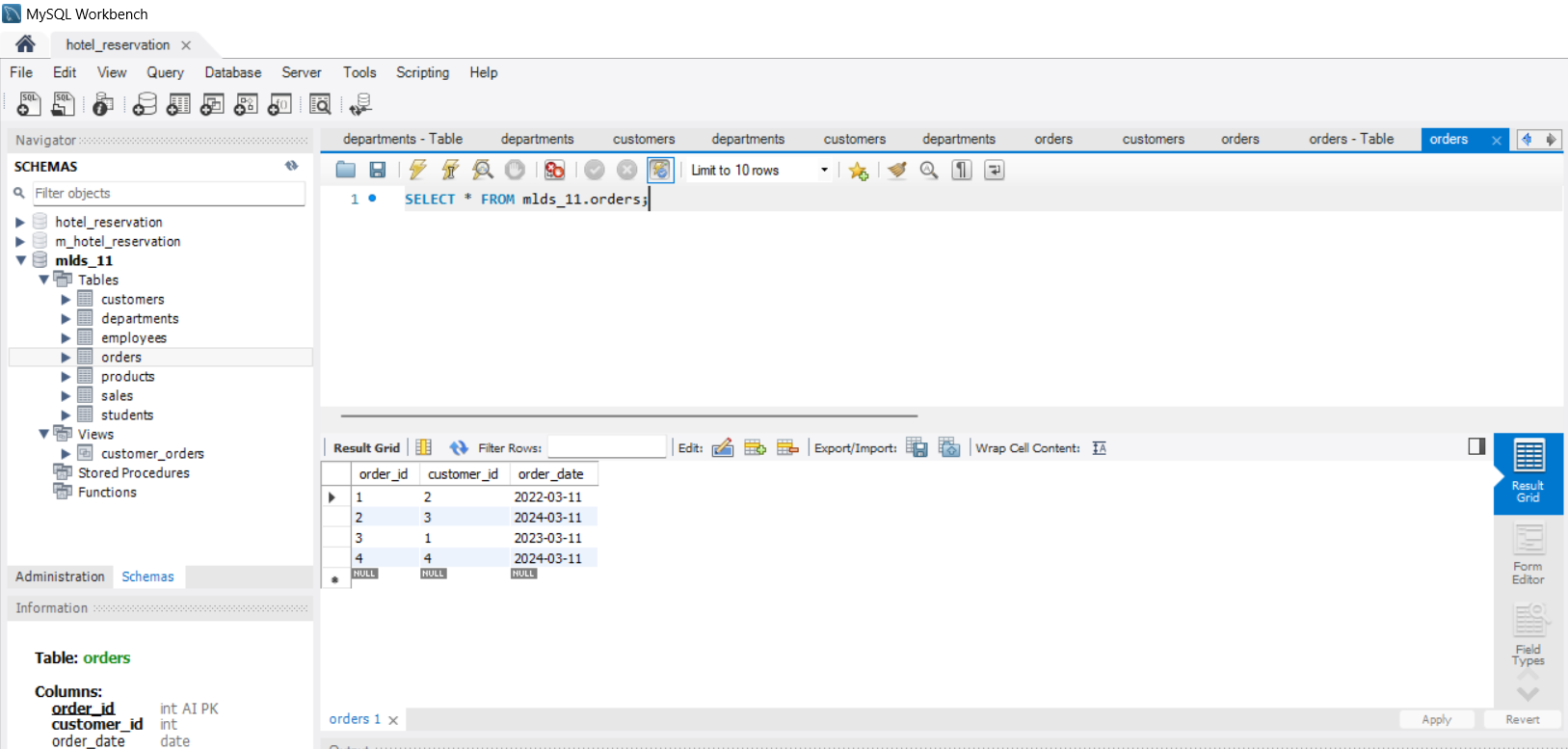
**Problem 5: Views and Joins**

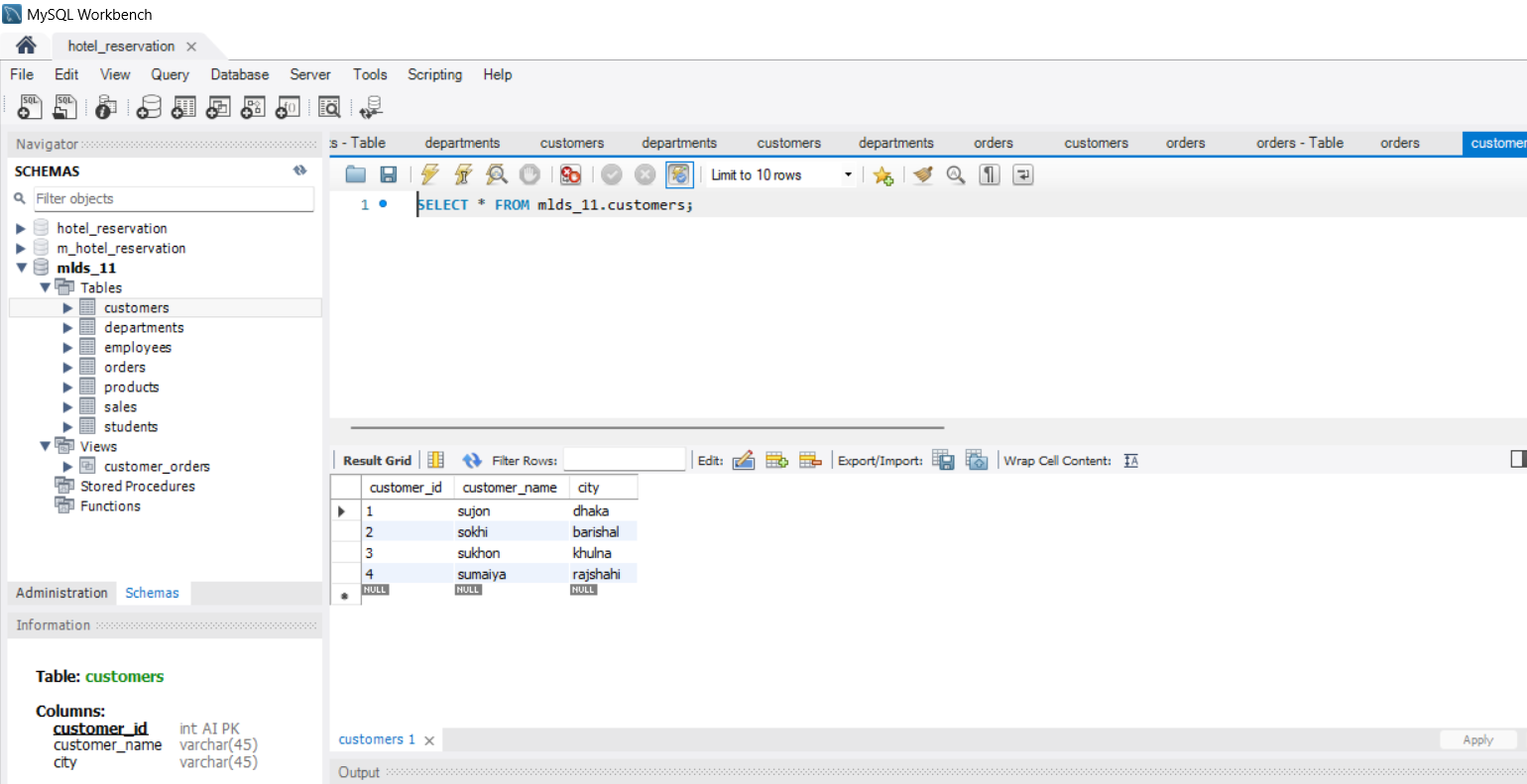
**Ques: Consider two tables: `orders` with columns `order\_id`, `customer\_id`, and `order\_date`, and `customers` with columns `customer\_id`, `customer\_name`, and `city`. Create a view named `customer\_orders` that displays the customer name, order ID, and order date for each order, along with the city of the customer.**

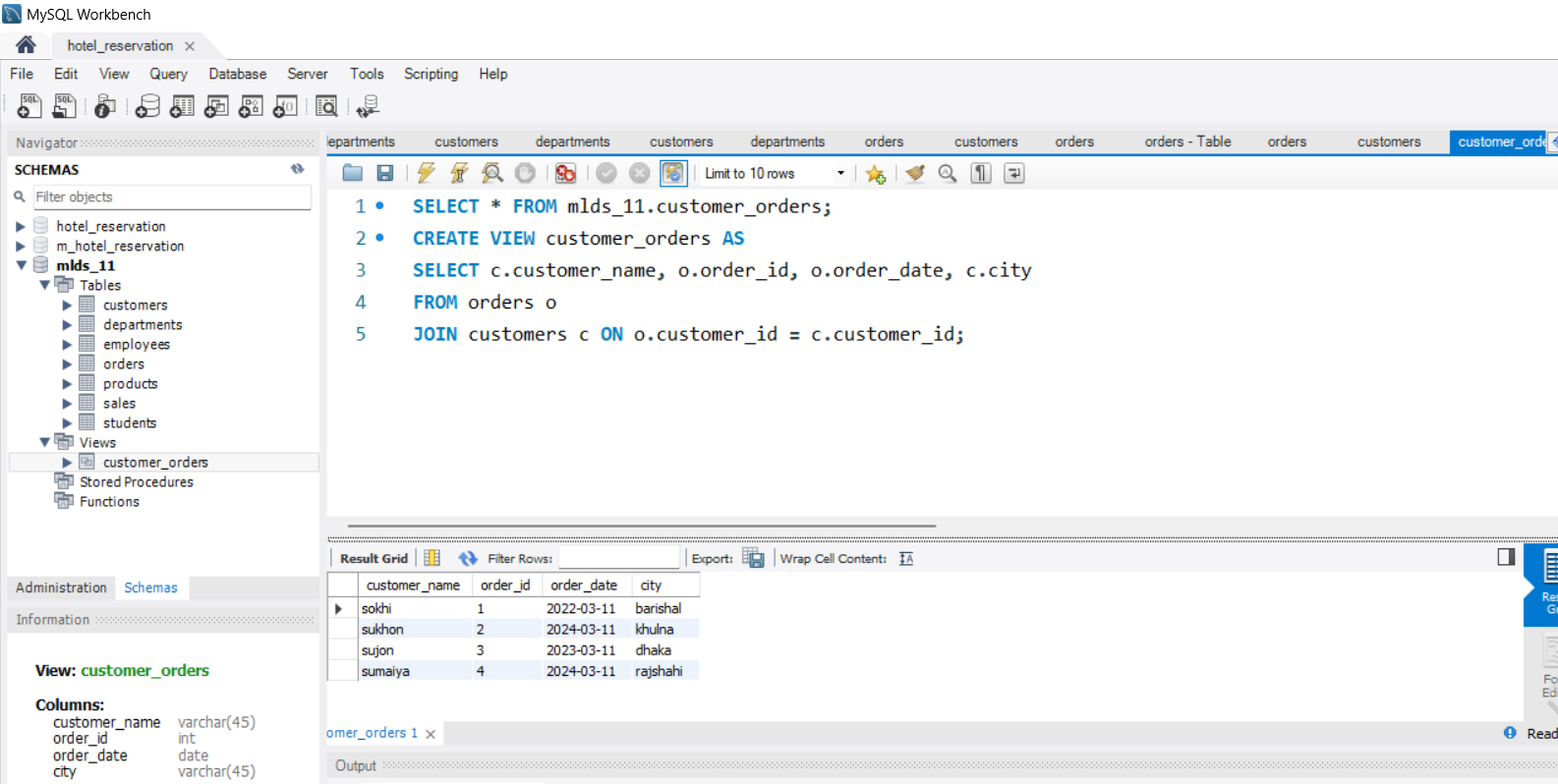
CREATE VIEW customer\_orders AS

SELECT c.customer\_name, o.order\_id, o.order\_date, c.city

FROM orders o

JOIN customers c ON o.customer\_id = c.customer\_id;





**Problem 6: Event**

**Ques: Design an event in MySQL that runs every day at midnight (12 AM) and updates the `sales` table (Problem 2) by increasing the sales amount of each product by 5%.**

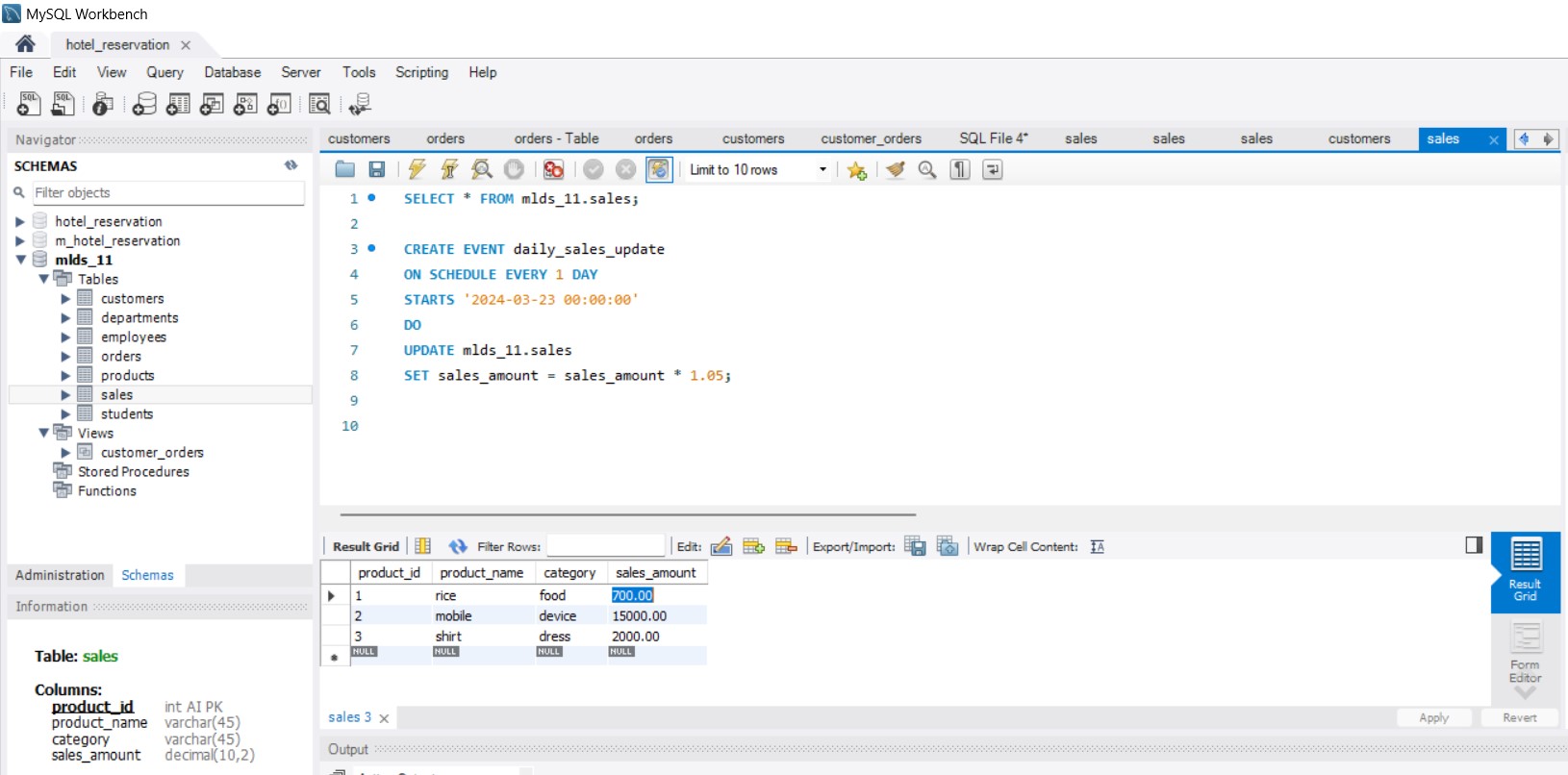
CREATE EVENT daily\_sales\_update

ON SCHEDULE EVERY 1 DAY

STARTS 'YYYY-MM-DD 00:00:00'

DO

UPDATE sales

SET sales\_amount = sales\_amount \* 1.05;

**Problem 7: Normalization**

**Ques: You have a table named `customers` with columns `customer\_id`, `customer\_name`, `address`, `city`, and `country`. Identify any normalization issues in this table and propose a normalized schema to address them.**

CREATE TABLE customers (

customer\_id INT PRIMARY KEY,

customer\_name VARCHAR(255)

);

CREATE TABLE addresses (

address\_id INT PRIMARY KEY,

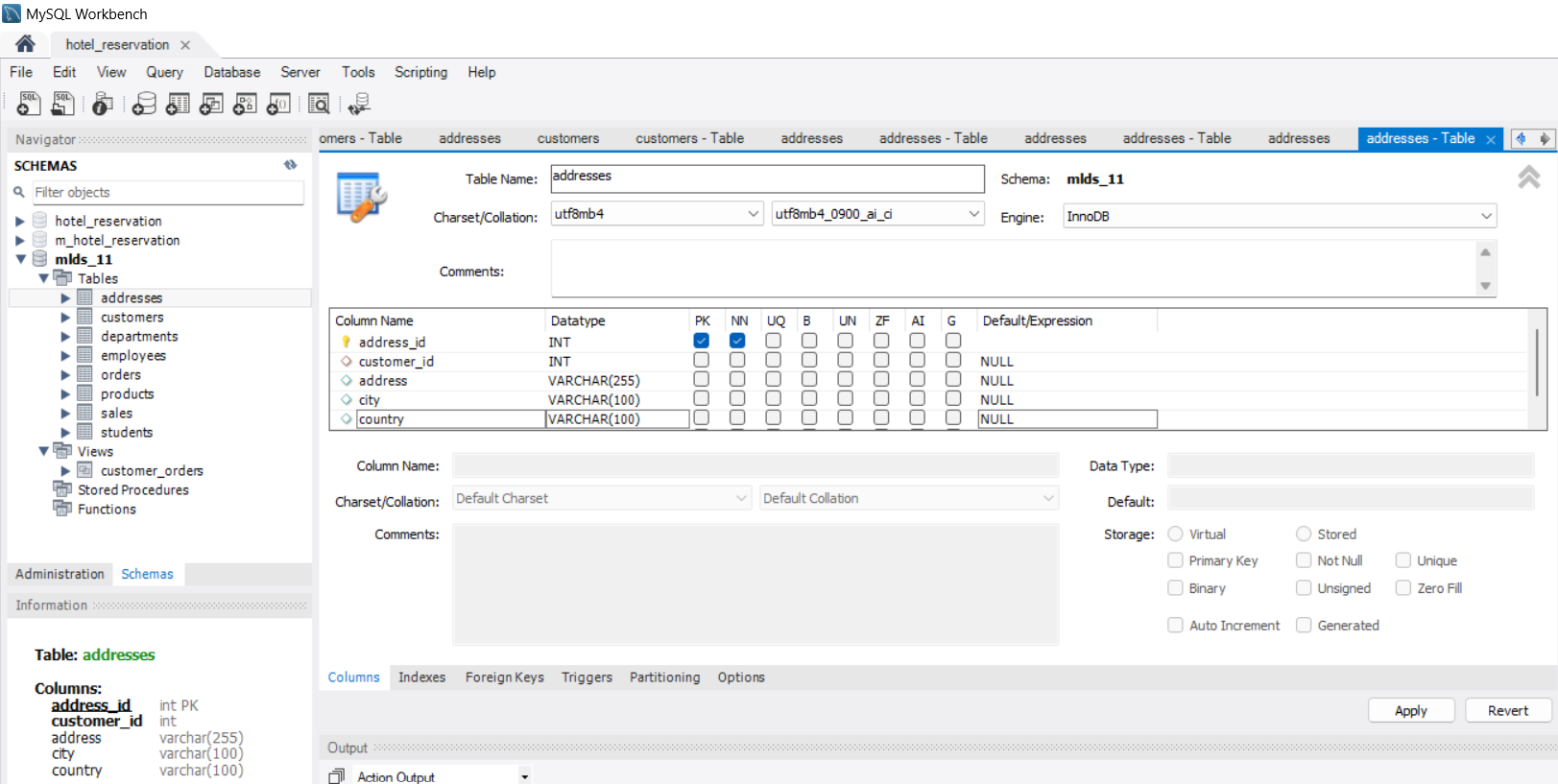
customer\_id INT,

address VARCHAR(255),

city VARCHAR(100),

country VARCHAR(100),

FOREIGN KEY (customer\_id) REFERENCES customers(customer\_id)

);

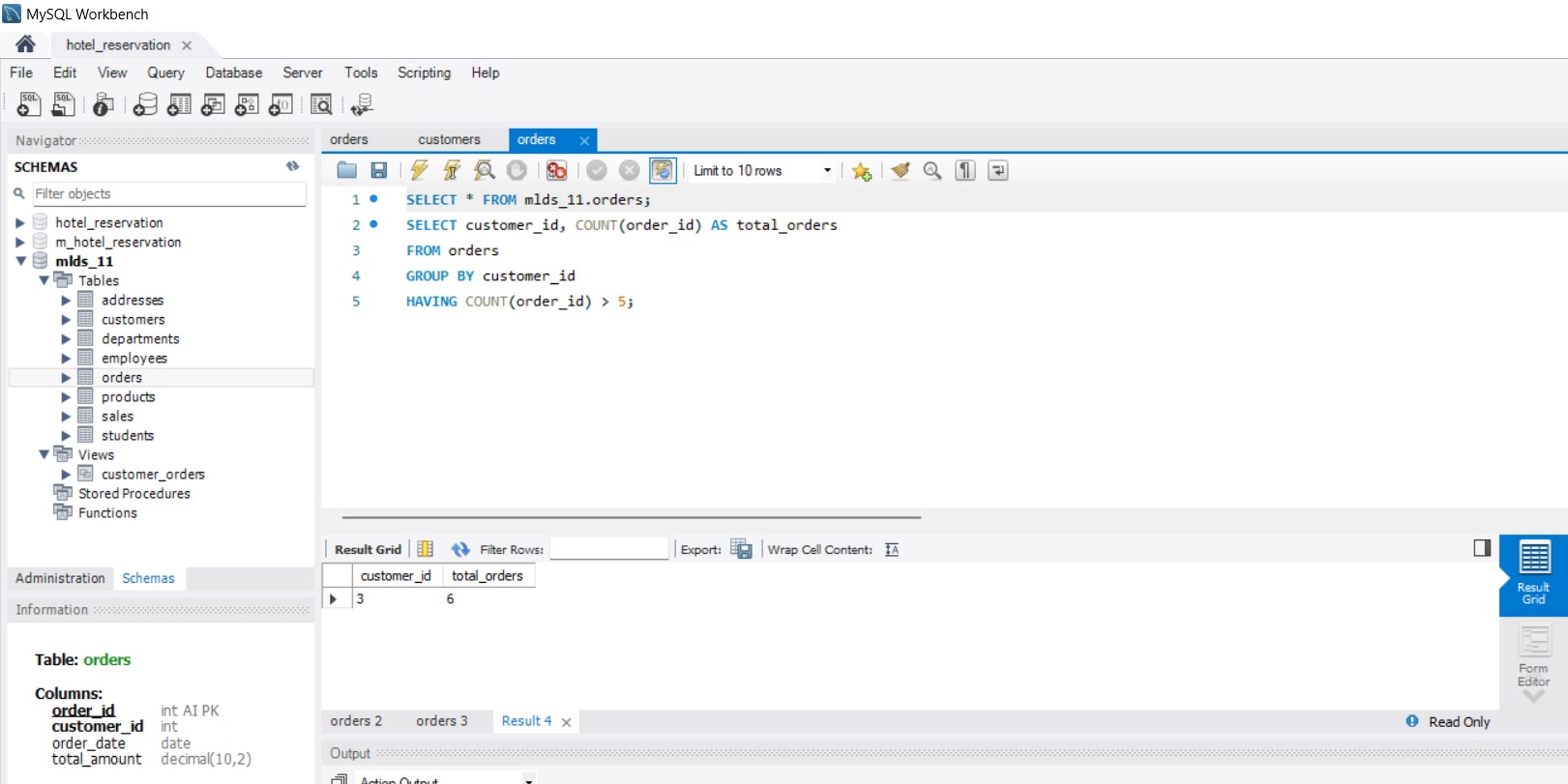
**Problem 8: GROUP BY and HAVING**

**Ques: Consider a table named `orders` with columns `order\_id`, `customer\_id`, `order\_date`, and `total\_amount`. Write a query to find the total number of orders placed by each customer who has placed more than 5 orders.**

SELECT customer\_id, COUNT(order\_id) AS total\_orders

FROM orders

GROUP BY customer\_id

HAVING COUNT(order\_id) > 5;