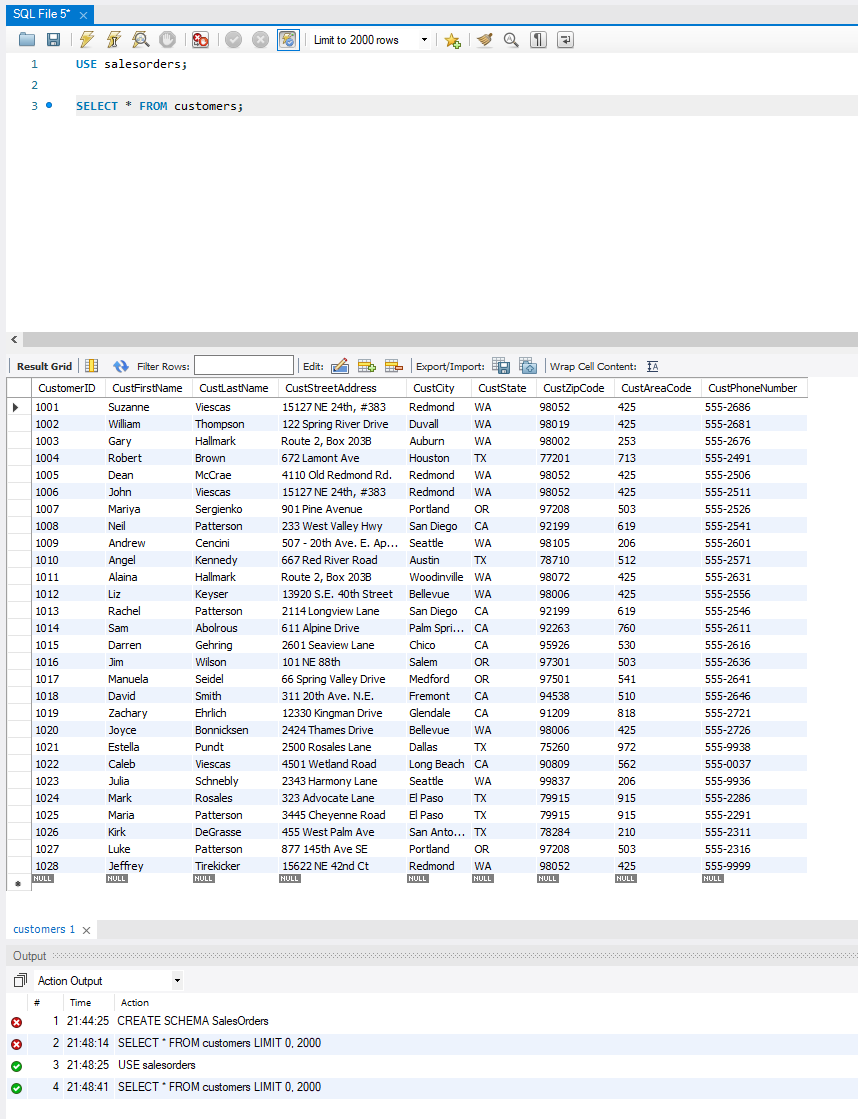
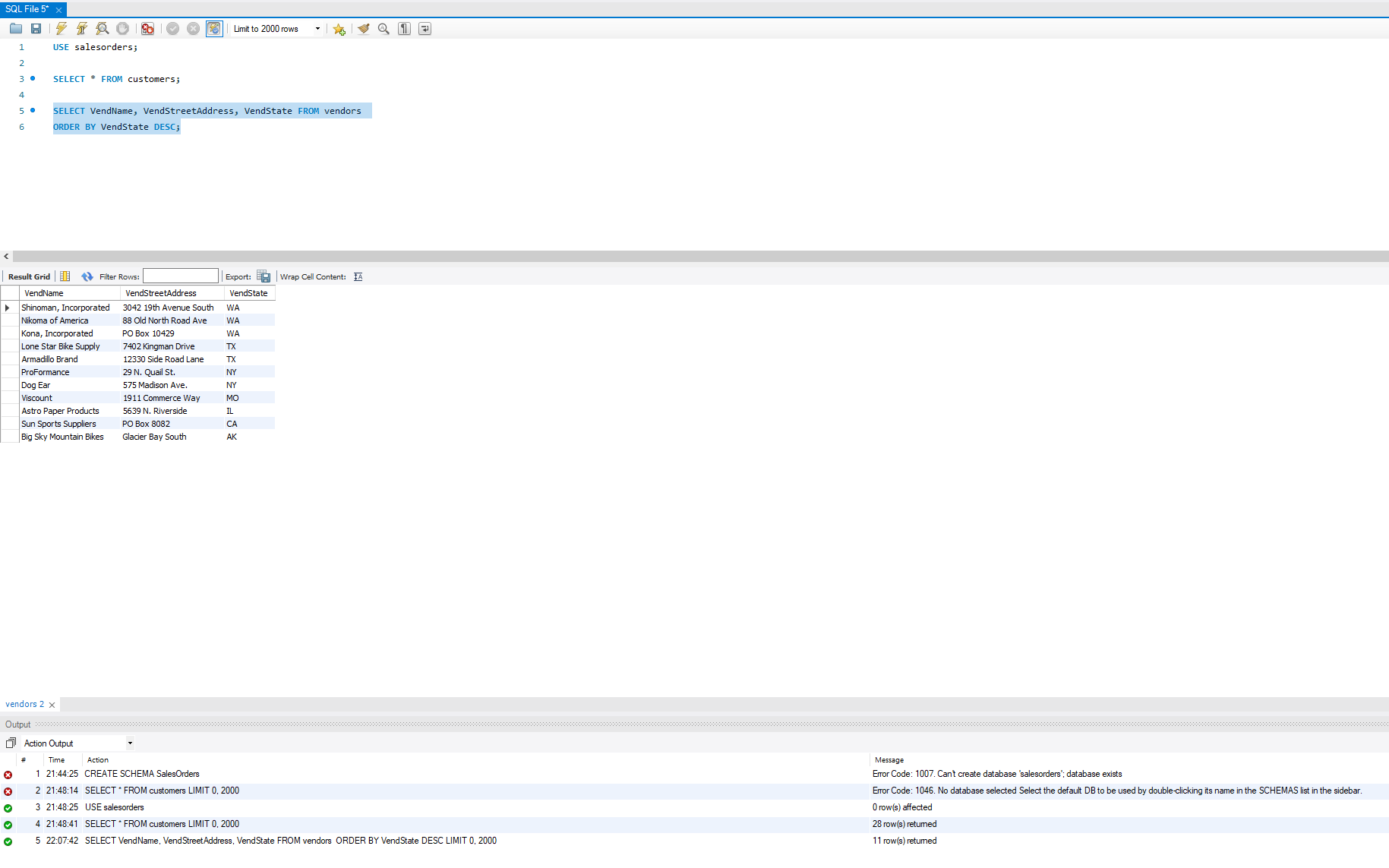
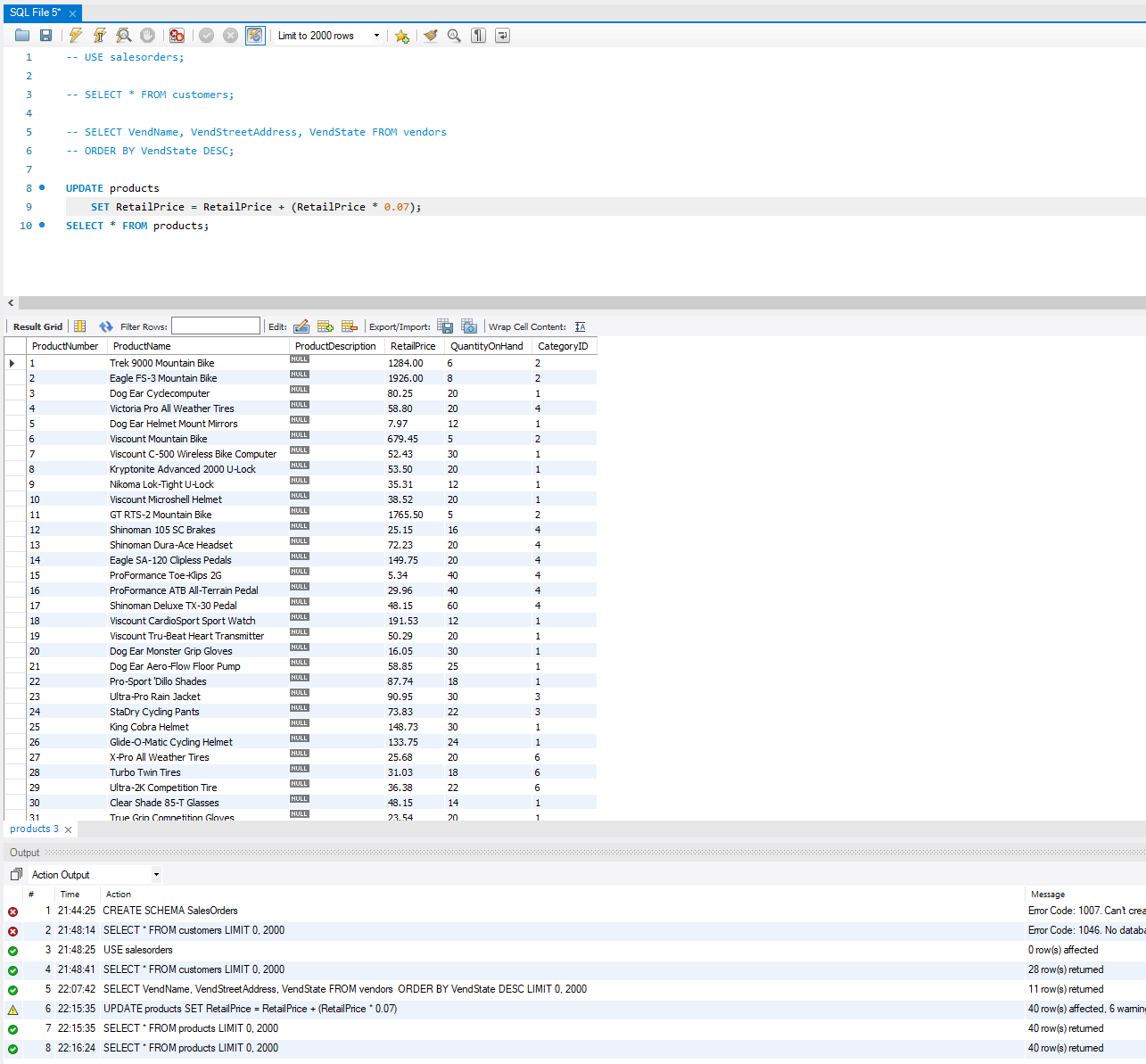
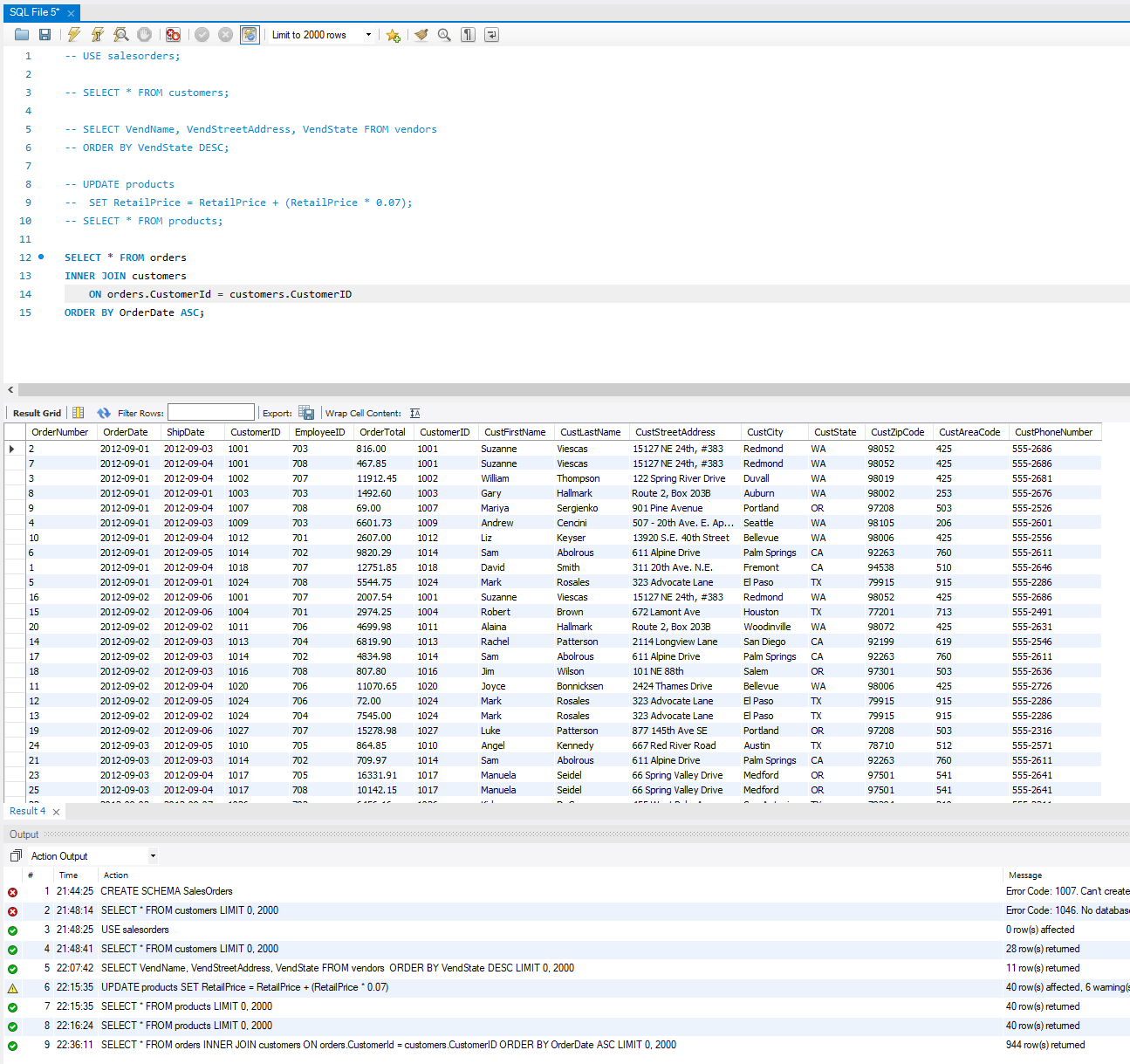
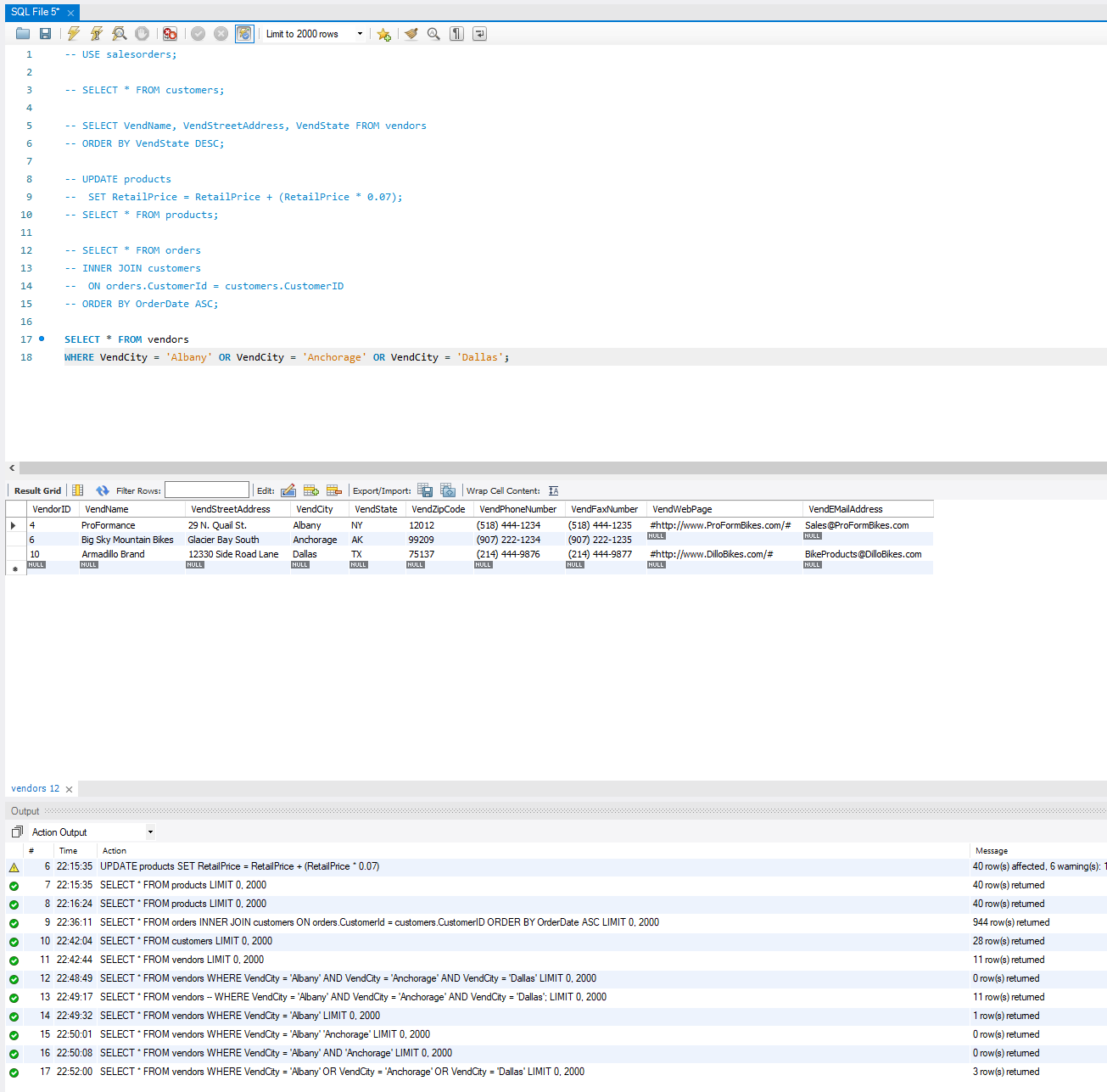
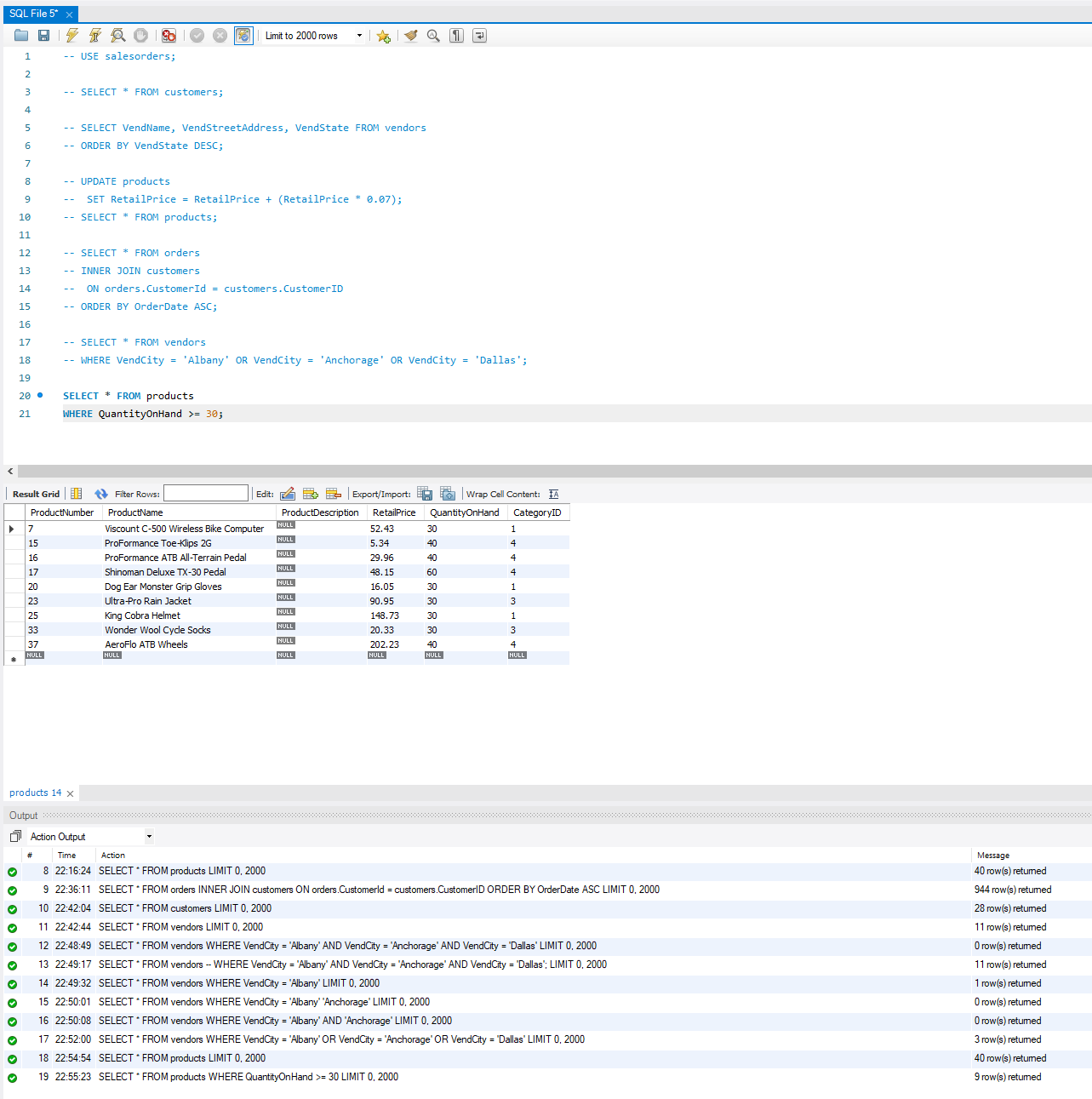
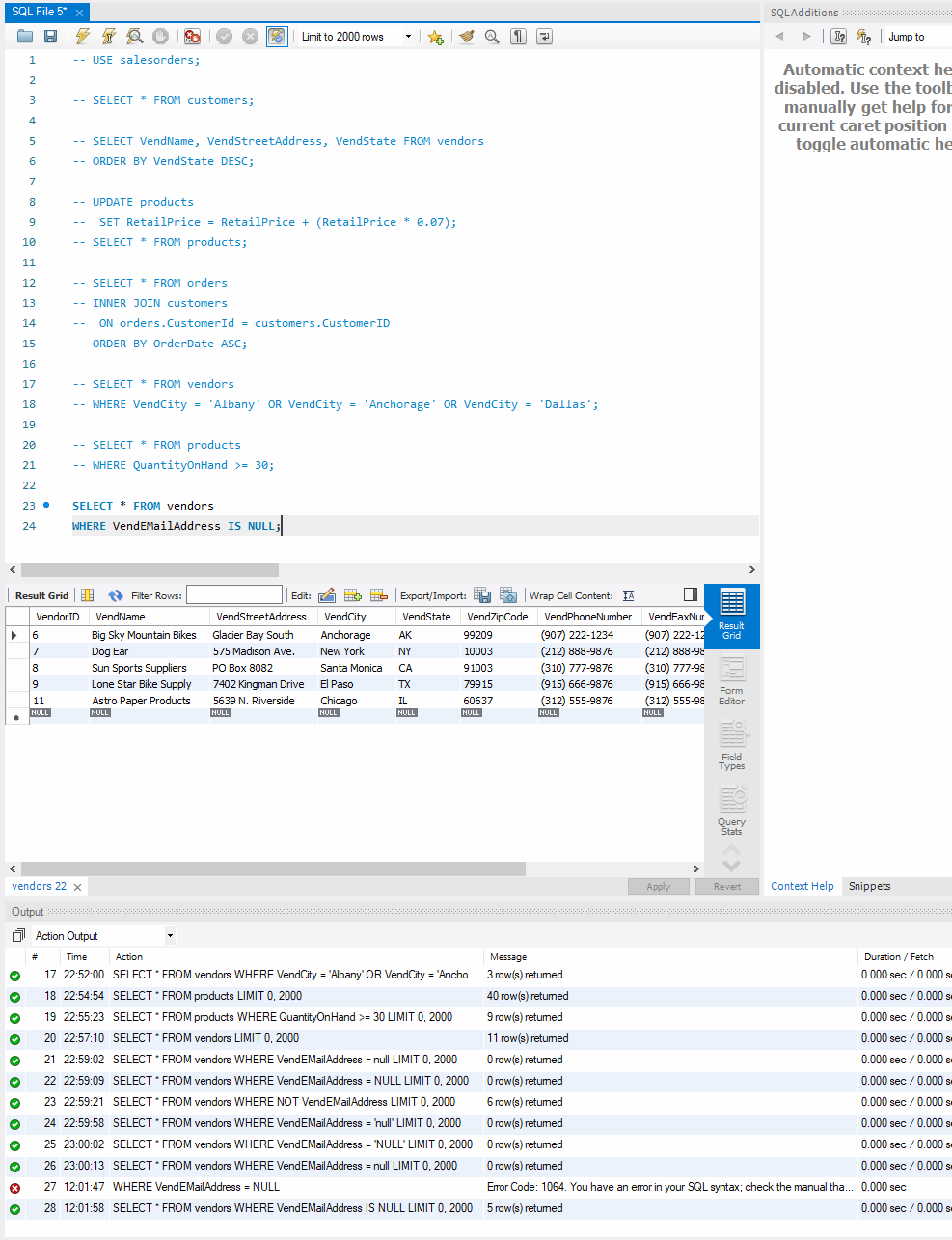
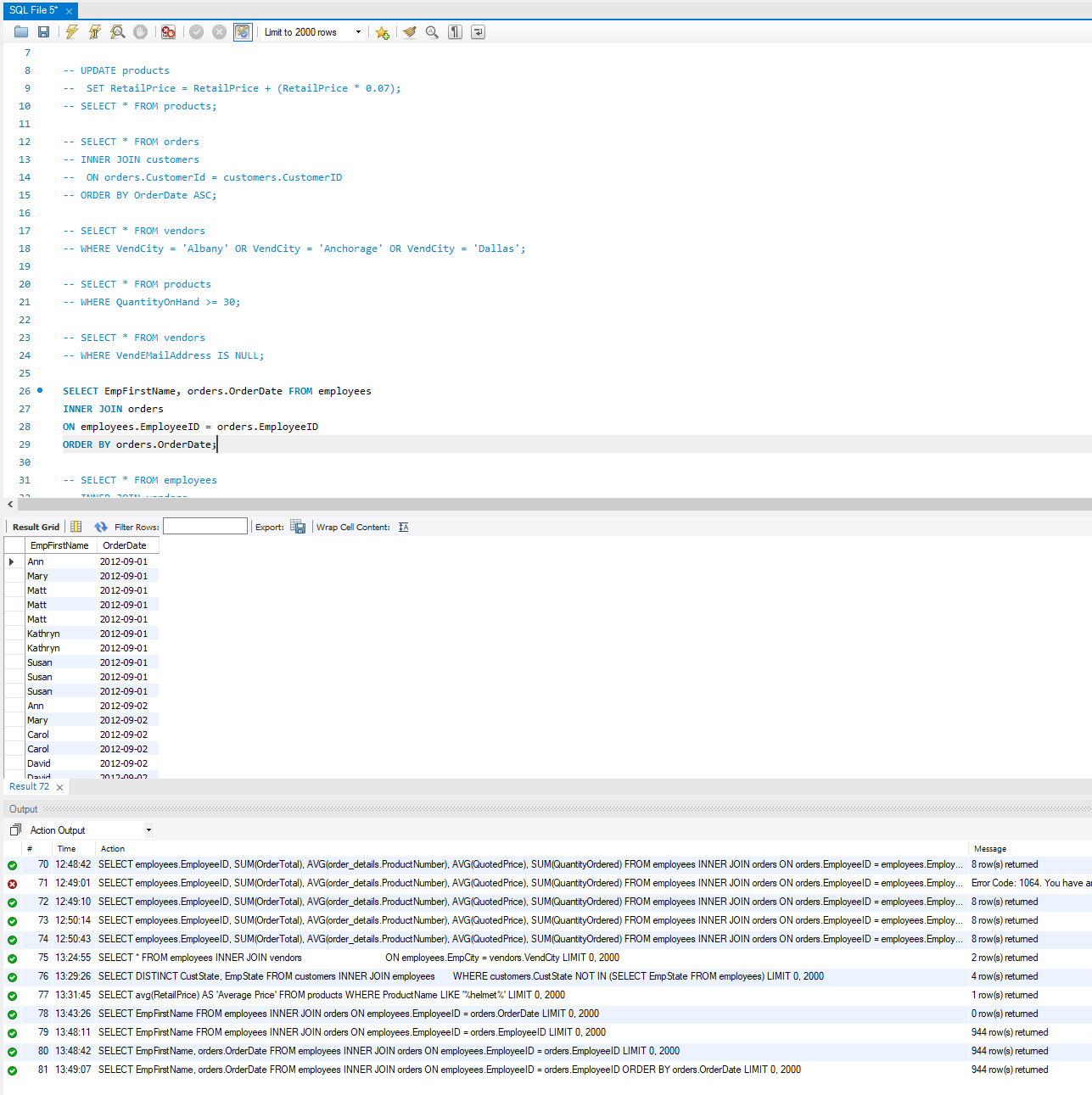
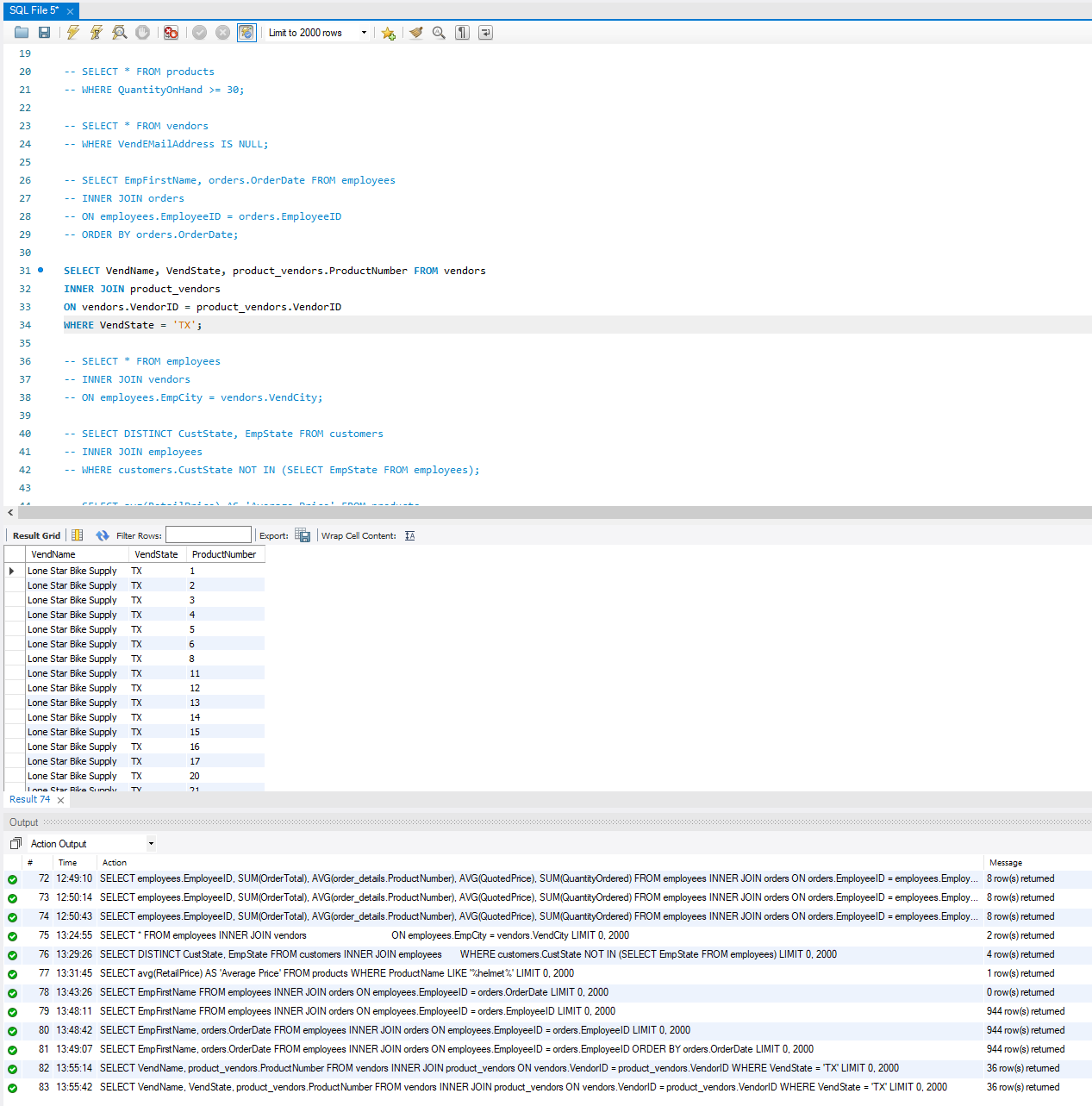
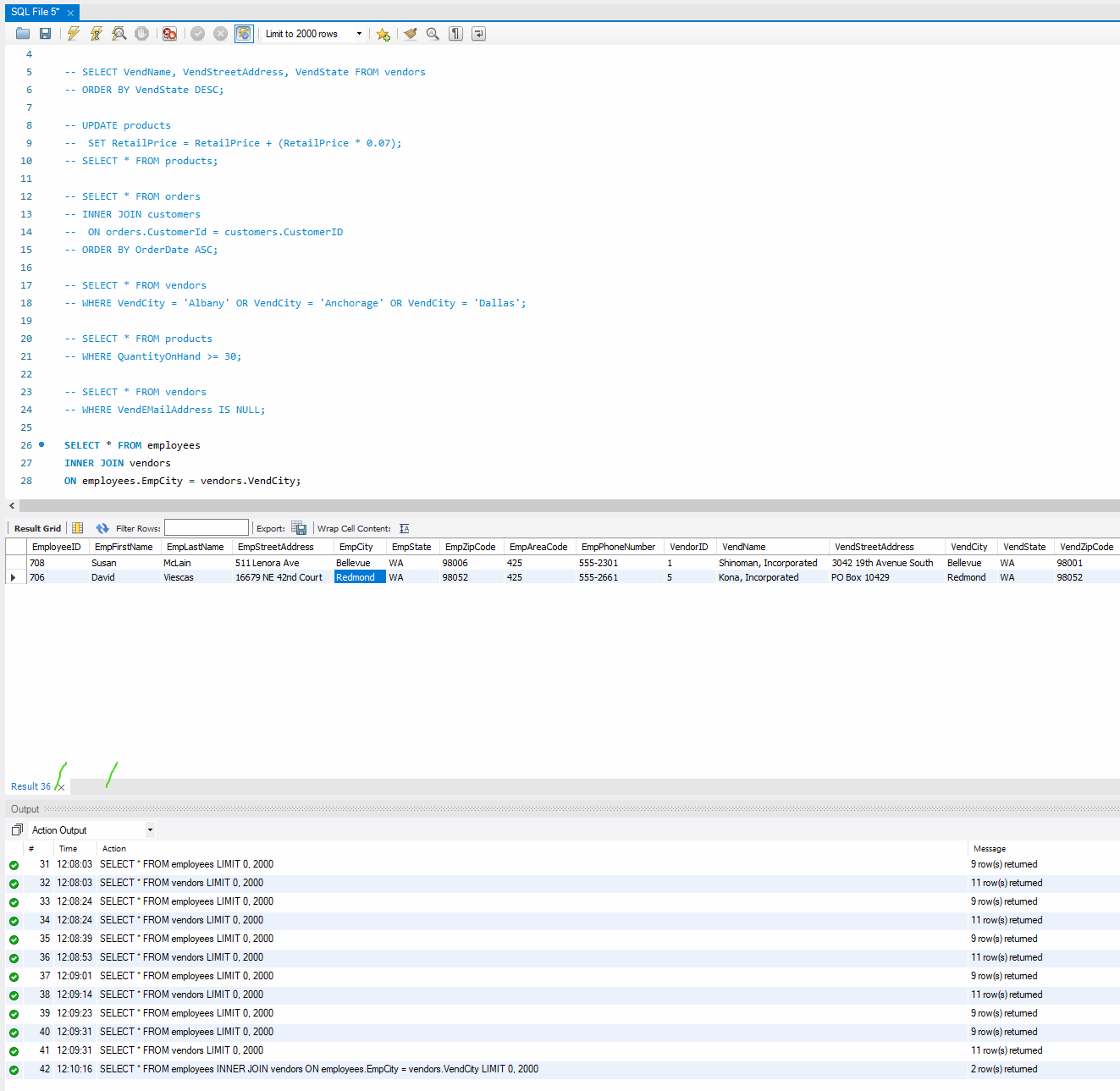
For this assignment, write queries using SQL to acquire data about customers, vendors, products, and employees in a fictitious sales database. These queries will cover many of the core aspects of writing SQL to produce data for reporting and analyzing information. There may be multiple ways to produce the same results, but ensure you are returning the requested fields.

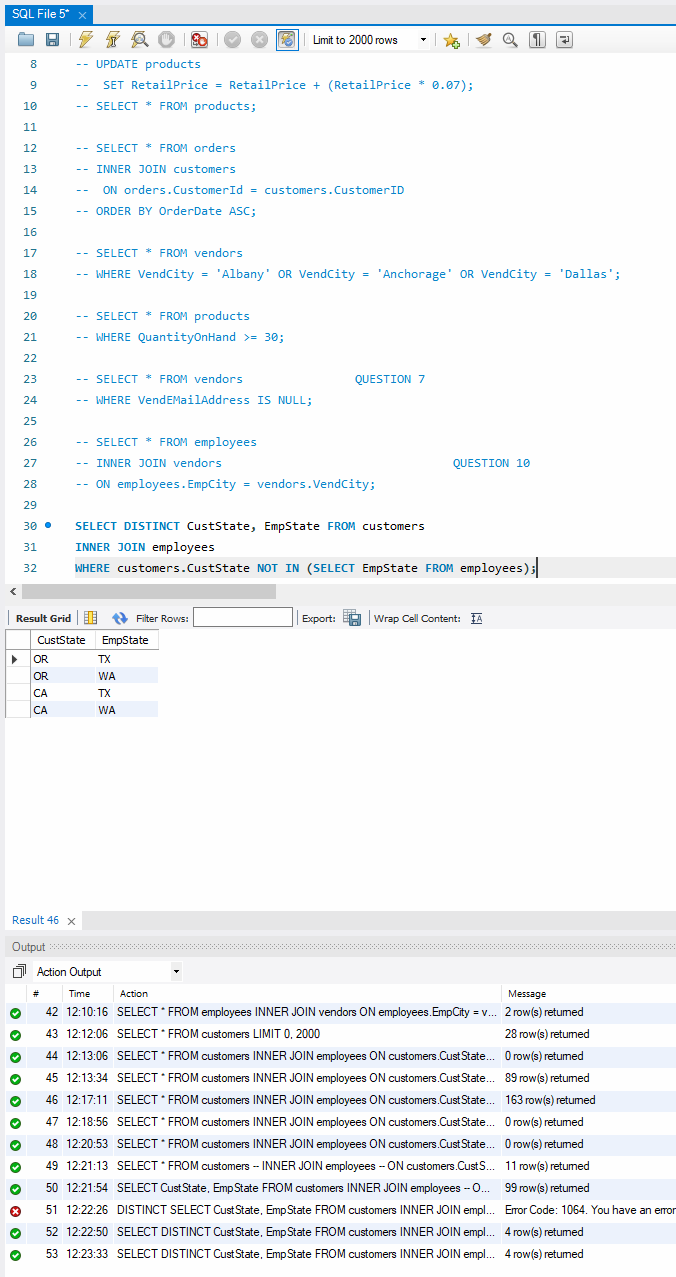
Using the Sales Orders database, complete the queries below.

1. **Show all the information on our customers**.
2. Query: SELECT \* FROM customers;
3. Columns: 9
4. Expected Row Count: 28
5. Screenshot: 
6. **Show a list of states, in reverse alphabetical order, where our vendors are located, and include the names of the vendor.**
7. Query: SELECT VendName, VendStreetAddress, VendState FROM vendors ORDER BY VendState DESC;
8. Columns: 3
9. Expected Row Count: 11
10. Screenshot: 
11. **What if we adjusted the retail price of each product by increasing it 7 percent?**
12. Query: UPDATE products SET RetailPrice = RetailPrice + (RetailPrice \* 0.07);
13. Columns: 6
14. Expected Row Count: 40
15. Screenshot: 
16. **Show a list of orders made by each customer in ascending date order.**
17. Query: SELECT \* FROM orders INNER JOIN customers ON orders.CustomerId customers.CustomerID ORDER BY OrderDate ASC;
18. Columns: 15
19. Expected Row Count: 944
20. Screenshot: 
21. **Give the names of all vendors based in Albany, Anchorage, and Dallas.**
22. Query: SELECT \* FROM vendors WHERE VendCity = 'Albany' OR VendCity = 'Anchorage' OR VendCity = 'Dallas';
23. Columns: 10
24. Expected Row Count: 3
25. Screenshot: 
26. **Show an alphabetized list of products with a quantity on hand greater than or equal to 30.**
27. Query: SELECT \* FROM products WHERE QuantityOnHand >= 30;
28. Columns: 6
29. Expected Row Count: 9
30. Screenshot: 
31. **What vendors do we work with that don’t have an email address?**
32. Query: SELECT \* FROM vendors WHERE VendEMailAddress IS NULL;

1. Columns: 10
2. Expected Row Count: 5

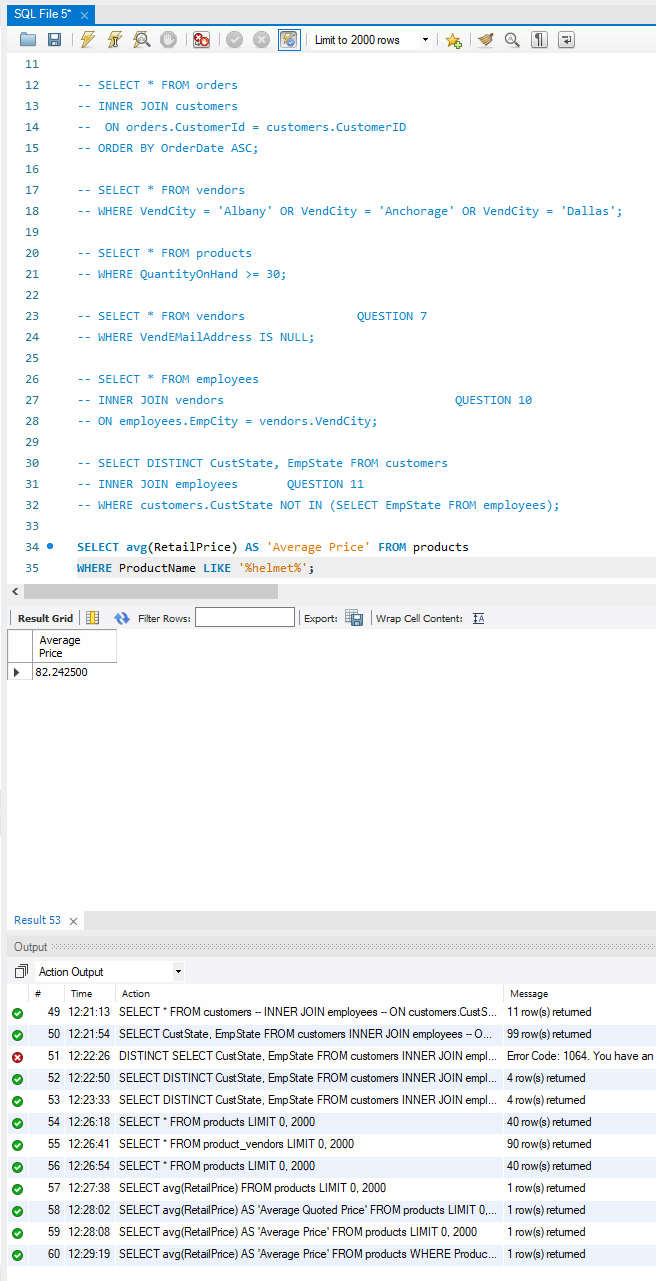


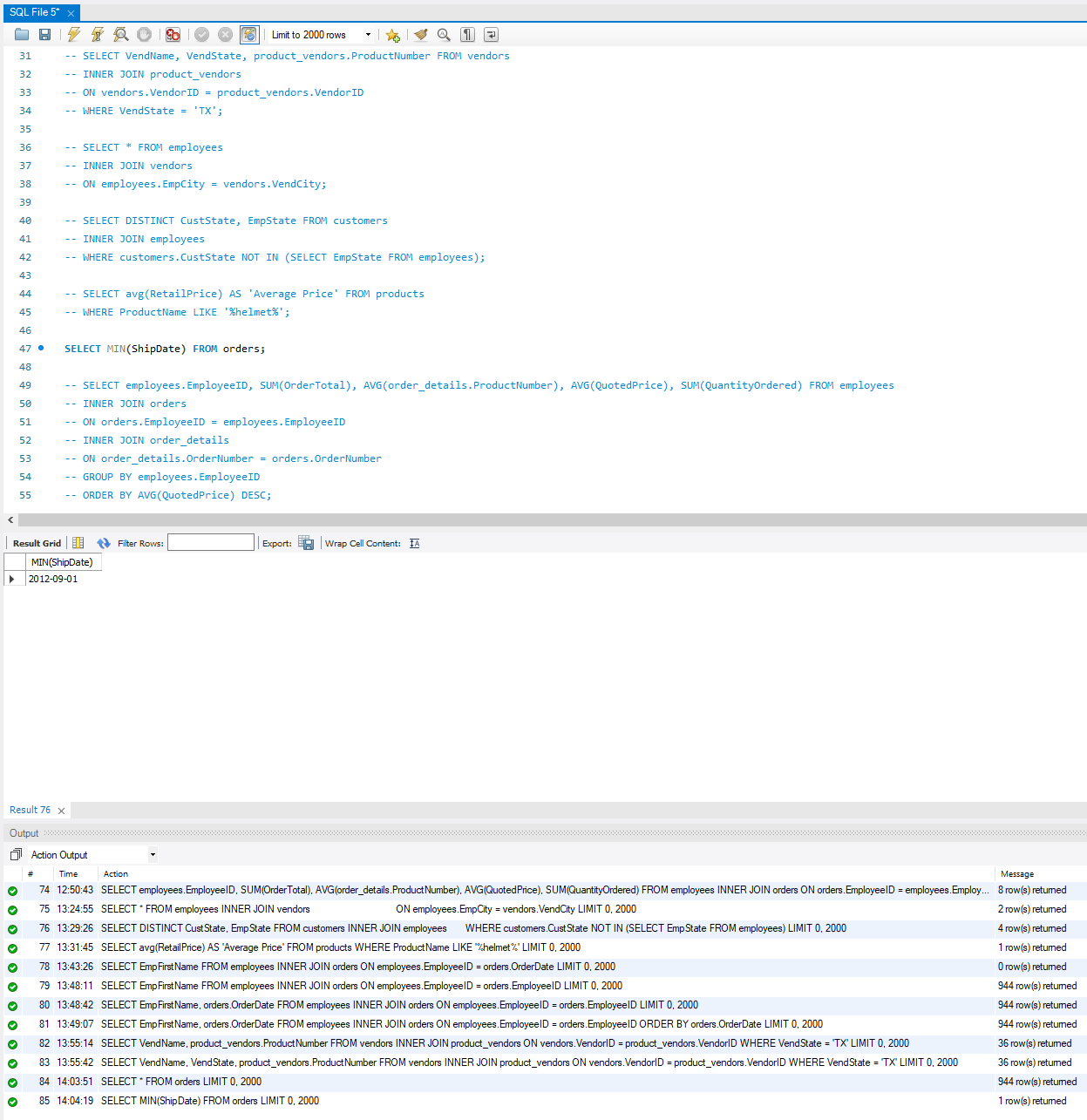
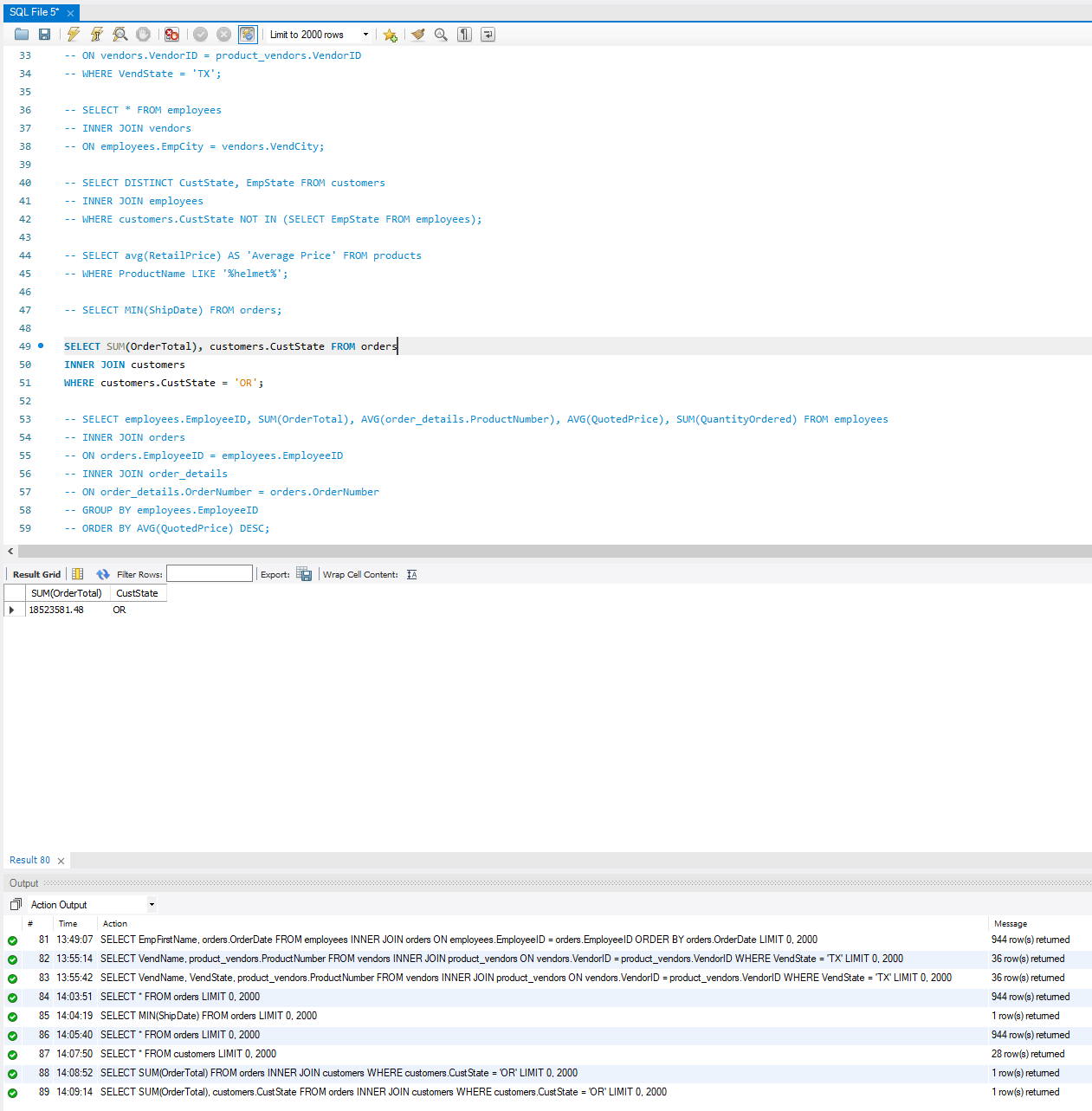
1. Screenshot:
2. **List employees and the dates their orders shipped sorted by order date.**
3. Query: SELECT EmpFirstName, orders.OrderDate FROM employee INNER JOIN orders ON employees.EmployeeID = orders.EmployeeID ORDER BY orders.OrderDate;
4. Columns: 2
5. Expected Row Count: 944
6. Screenshot:
7. **Show the vendors and products they supply to us for products over $75 for vendors in Texas.**
8. Query: SELECT VendName, VendState, product\_vendors.ProductNumber FROM vendors INNER JOIN product\_vendors ON vendors.VendorID product\_vendors.VendorID WHERE VendState = 'TX';
9. Columns: 3
10. Expected Row Count: 36
11. Screenshot: 
12. **Show employees who live in the same city and state as our vendors.**
13. Query: SELECT \* FROM employees INNER JOIN vendors ON employees.EmpCity = vendors.VendCity;
14. Columns: 19
15. Expected Row Count: 2
16. Screenshot: 
17. **Display customers who have no sales rep (employees) in the same state.**
18. Query: SELECT DISTINCT CustState, EmpState FROM customers INNER JOIN employees WHERE customers.CustState NOT IN (SELECT EmpState FROM employees);
19. Columns: 2
20. Expected Row Count: 4



1. Screenshot:
2. **What is the average quoted price of a helmet?**

1. Query: SELECT avg(RetailPrice) AS 'Average Price' FROM product WHERE ProductName LIKE '%helmet%';
2. Columns: 1



1. Expected Row Count: 1
2. Screenshot:
3. **What was the date of the earliest ship date?**
4. Query: SELECT MIN(ShipDate) FROM orders;
5. Columns: 1
6. Expected Row Count: 1
7. Screenshot: 
8. **What is the total amount (in dollars) of orders from the state of Oregon?**
9. Query: SELECT SUM(OrderTotal), customers.CustState FROM orders INNER JOIN customers WHERE customers.CustState = 'OR';
10. Columns: 2
11. Expected Row Count: 1
12. Screenshot: 
13. **Show each employee, the employee’s total sales (in dollars), the employee’s total sales item quantity, and the average item sales price ordered by the employee’s average item sales price highest to lowest.**
14. Query: SELECT employees.EmployeeID, SUM(OrderTotal), AVG(order\_details.ProductNumber), AVG(QuotedPrice), SUM(QuantityOrdered) FROM employees INNER JOIN orders ON orders.EmployeeID = employees.EmployeeID INNER JOIN order\_details ON order\_details.OrderNumber = orders.OrderNumber GROUP BY employees.EmployeeID ORDER BY AVG(QuotedPrice) DESC;
15. Columns: 5
16. Expected Row Count: 8
17. Screenshot:

