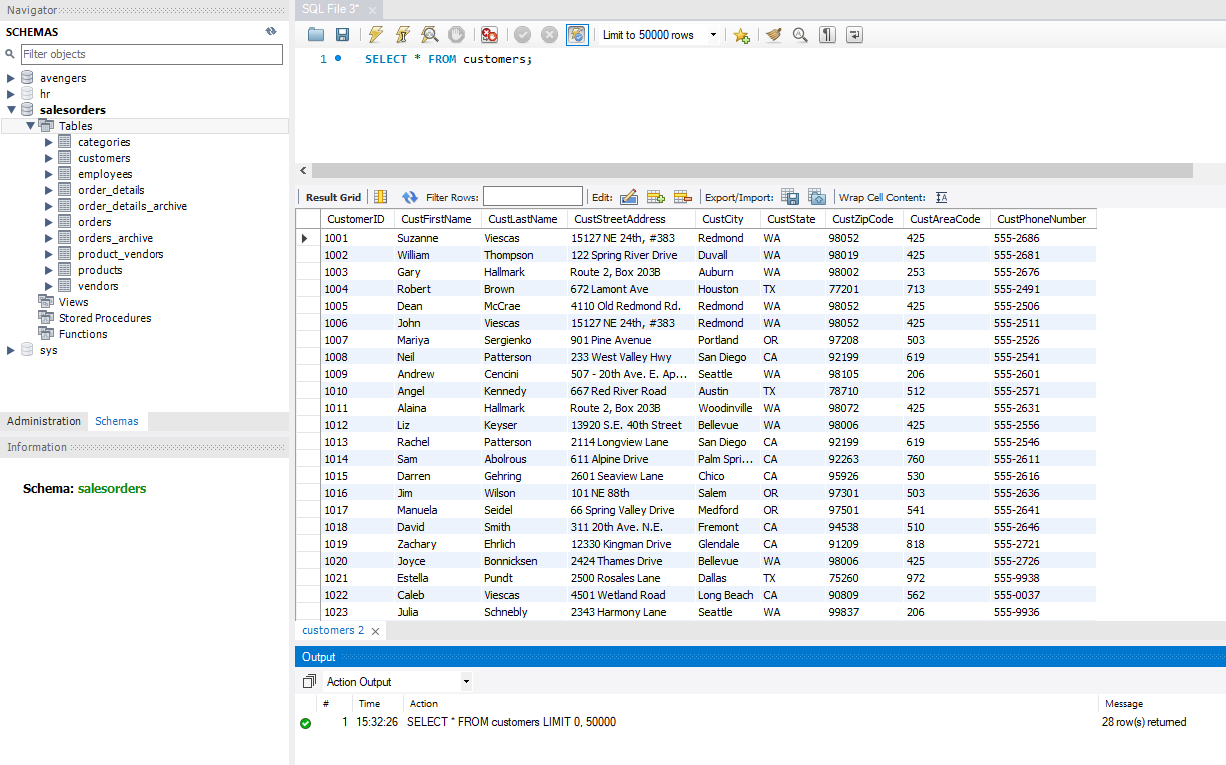
For this assignment, write queries using SQL to acquire data about customers, vendors, products, and employees in a fictitious sales database. These queries will cover many of the core aspects of writing SQL to produce data for reporting and analyzing information. There may be multiple ways to produce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

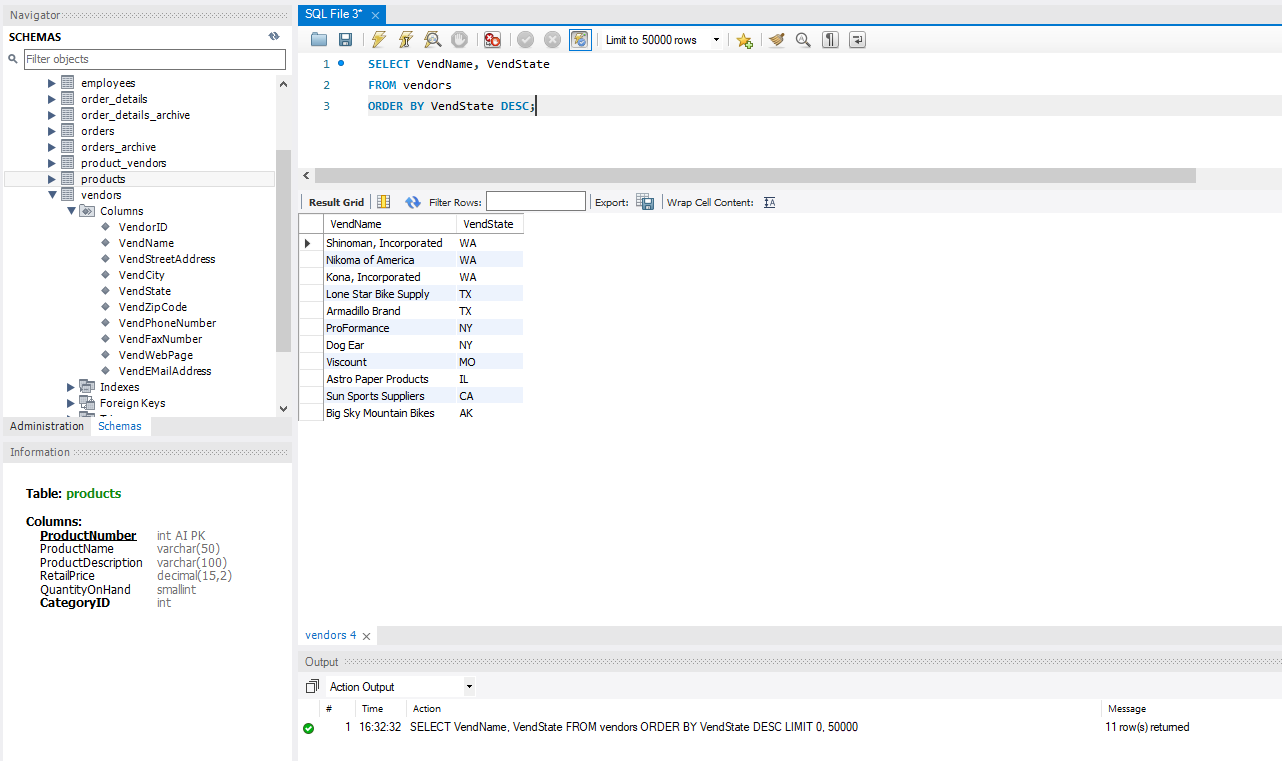
Using the Sales Orders database, complete the queries below.

1. **Show all the information on our customers**.
2. Query: SELECT \* FROM customers;
3. Columns: CustomerID, CustFirstName, CustLastName, CustStreetAddress, CustCity, CustState, CustZipCode, CustAreaCode, CustPhoneNumber
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, VendState

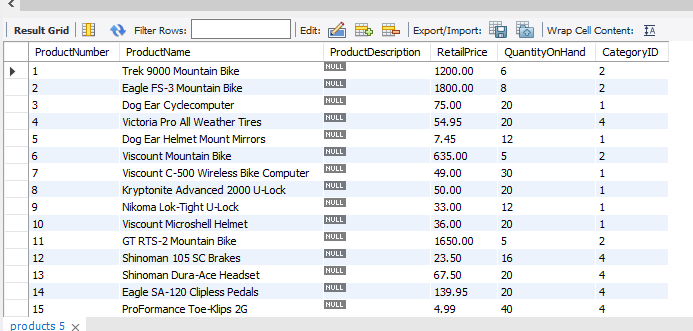
FROM vendors

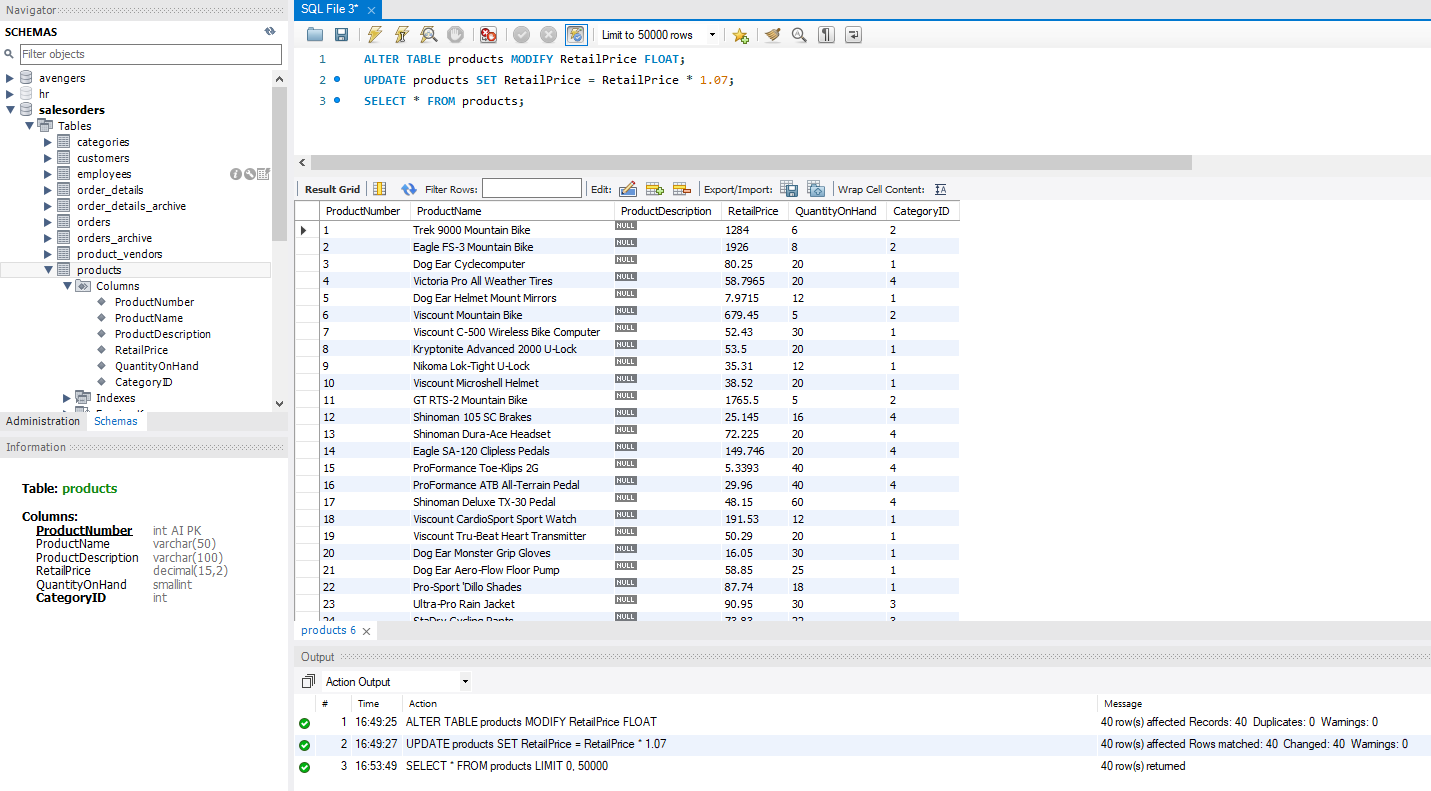
ORDER BY VendState DESC;

1. Columns: VendName, VendState
2. Expected Row Count: 11
3. Screenshot: 
4. **What if we adjusted the retail price of each product by increasing it 7 percent?**
5. Query:

ALTER TABLE products MODIFY RetailPrice FLOAT;

UPDATE products SET RetailPrice = RetailPrice \* 1.07;

1. Columns: ProductNumber, ProductName, ProductDescription, RetailPrice, QuantityOnHand, CategoryID
2. Expected Row Count: 40
3. Screenshot: 

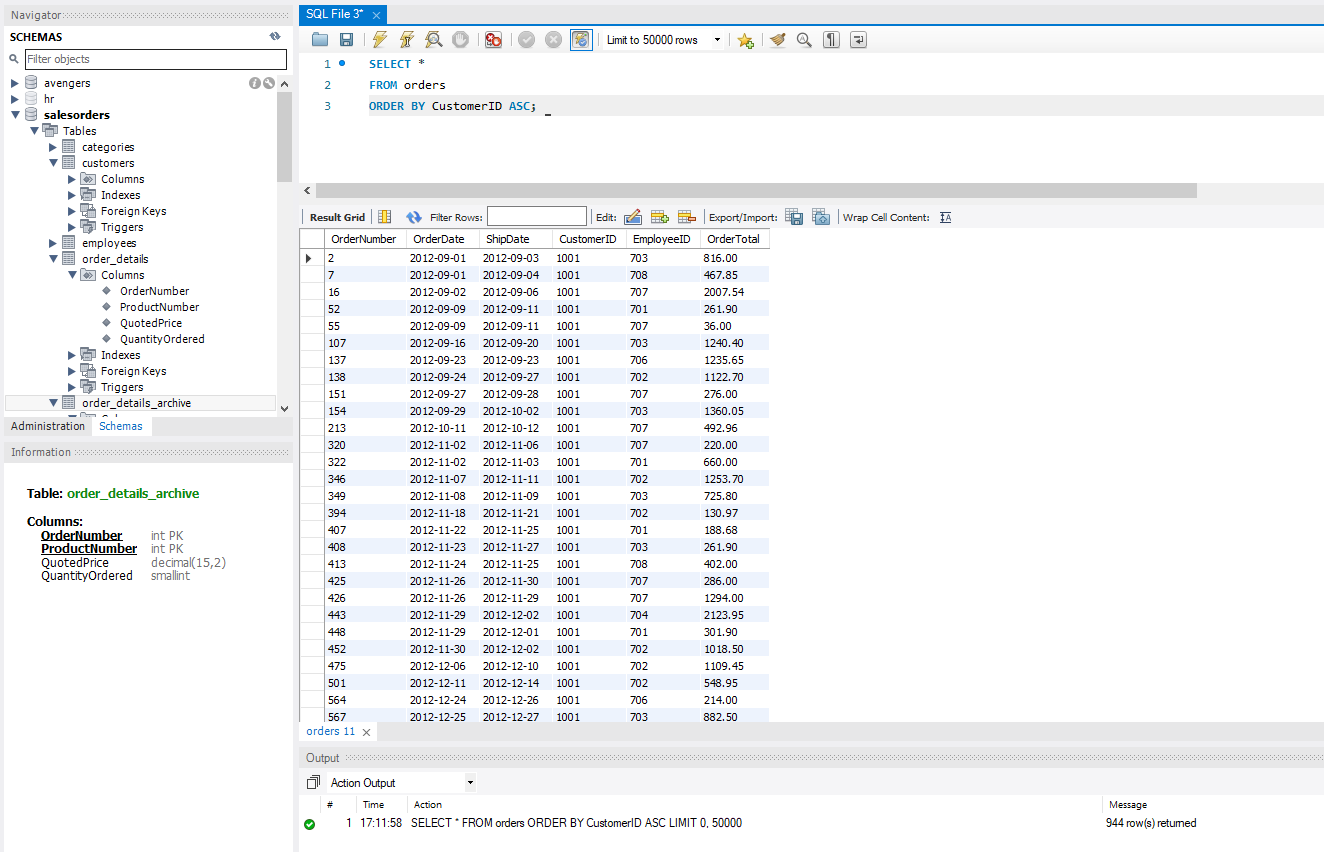


1. **Show a list of orders made by each customer in ascending date order.**
2. Query:

SELECT \*

FROM orders

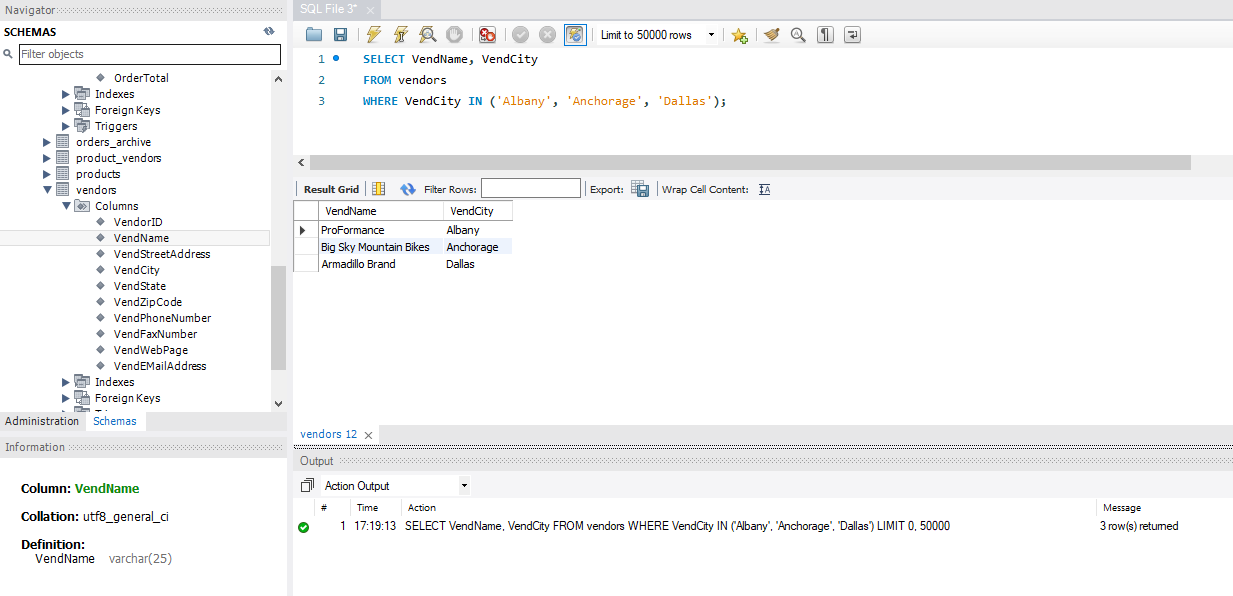
ORDER BY CustomerID ASC;

1. Columns: OrderNumber, OrderDate, ShipDate, CustomerID, EmployeeID, OrderTotal
2. Expected Row Count: 944
3. Screenshot: 
4. **Give the names of all vendors based in Albany, Anchorage, and Dallas.**
5. Query:

SELECT VendName, VendCity

FROM vendors

WHERE VendCity IN ('Albany', 'Anchorage', 'Dallas');

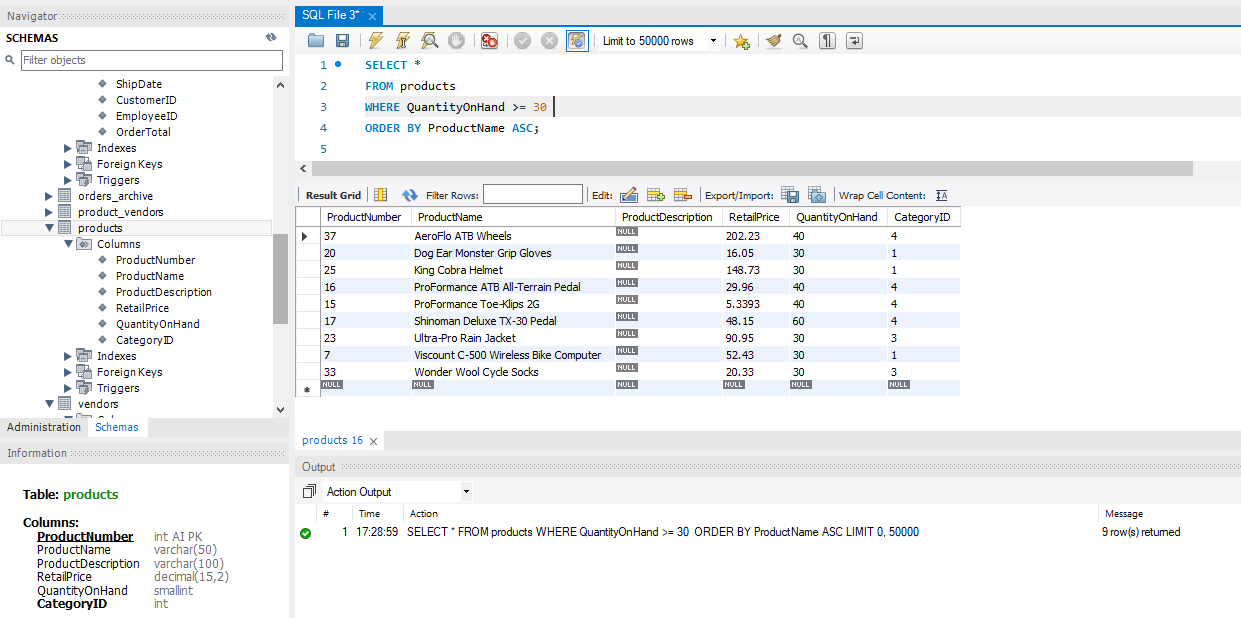
1. Columns: VendName, VendCity
2. Expected Row Count: 3
3. Screenshot: 
4. **Show an alphabetized list of products with a quantity on hand greater than or equal to 30.**
5. Query:

SELECT \*

FROM products

WHERE QuantityOnHand >= 30

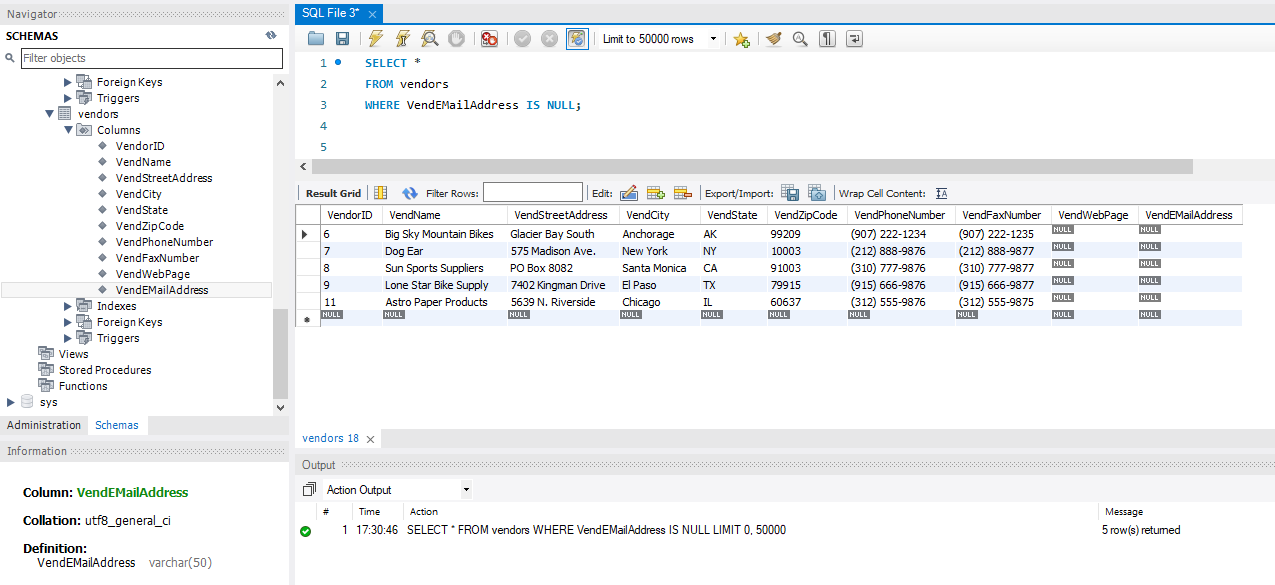
ORDER BY ProductName ASC;

1. Columns: ProductNumber, ProductName, ProductDescription, RetailPrice, QuantityOnHand, CategoryID
2. Expected Row Count: 9
3. Screenshot: 
4. **What vendors do we work with that don’t have an email address?**
5. Query:

SELECT \*

FROM vendors

WHERE VendEMailAddress IS NULL;

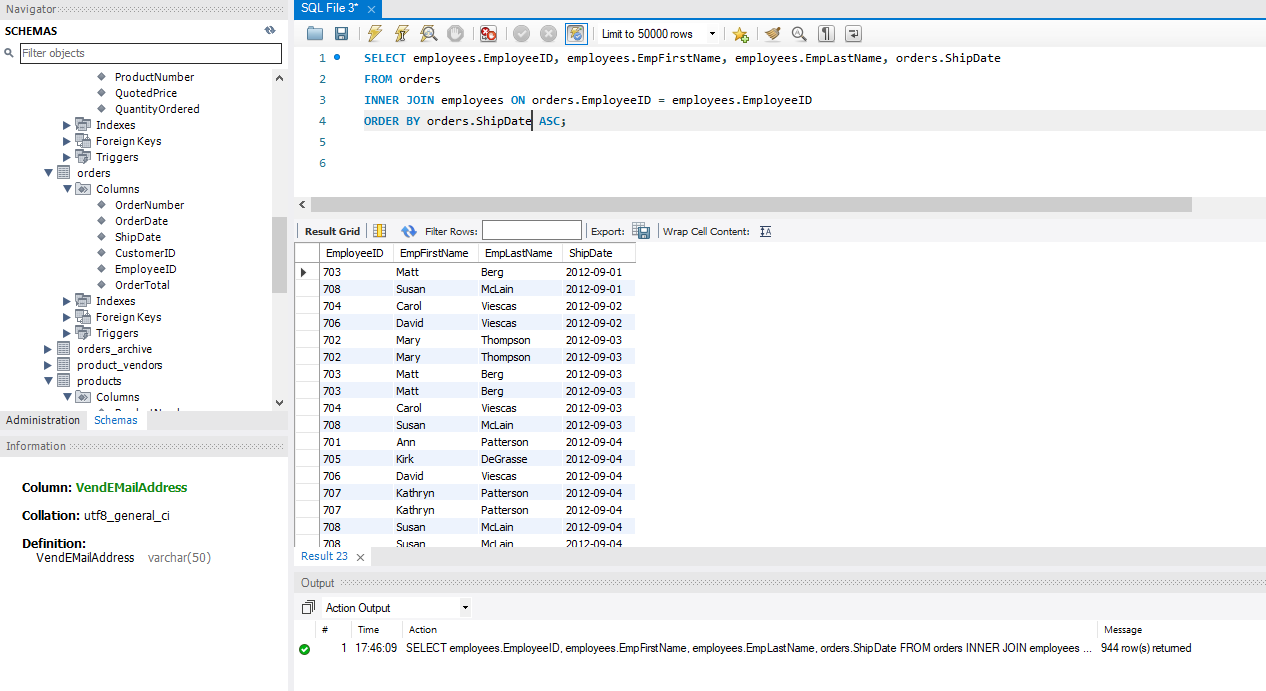
1. Columns: VendorID, VendName, VendStreetAddress, VendCity, VendState, VendZipCode, VendPhoneNumber, VendFaxNumber, VendWebPage, VendEMailAddress
2. Expected Row Count: 5
3. Screenshot: 
4. **List employees and the dates their orders shipped sorted by order date.**
5. Query:

SELECT employees.EmployeeID, employees.EmpFirstName, employees.EmpLastName, orders.ShipDate

FROM orders

INNER JOIN employees ON orders.EmployeeID = employees.EmployeeID

ORDER BY orders.ShipDate ASC;

1. Columns: EmployeeID, EmpFirstName, EmpLastName, ShipDate
2. Expected Row Count: 944
3. Screenshot: 
4. **Show the vendors and products they supply to us for products over $75 for vendors in Texas.**
5. Query:

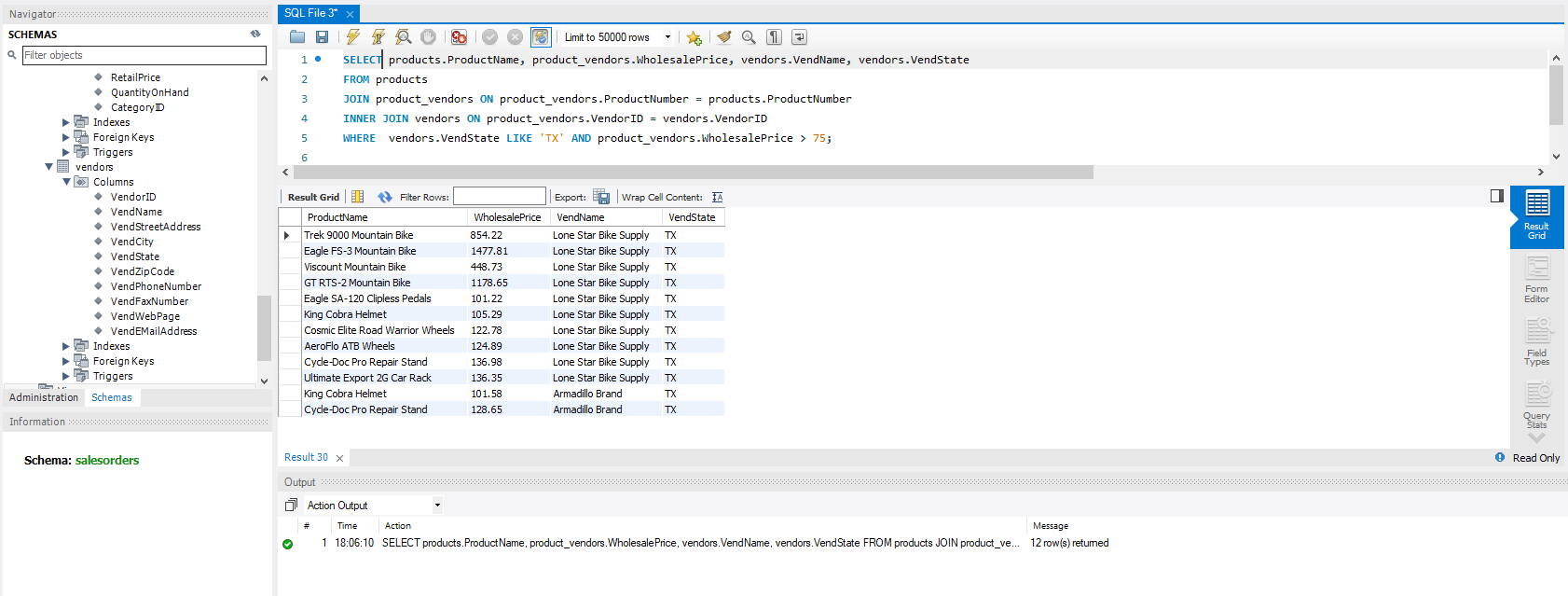
SELECT products.ProductName, product\_vendors.WholesalePrice, vendors.VendName, vendors.VendState

FROM products

JOIN product\_vendors ON product\_vendors.ProductNumber = products.ProductNumber

INNER JOIN vendors ON product\_vendors.VendorID = vendors.VendorID

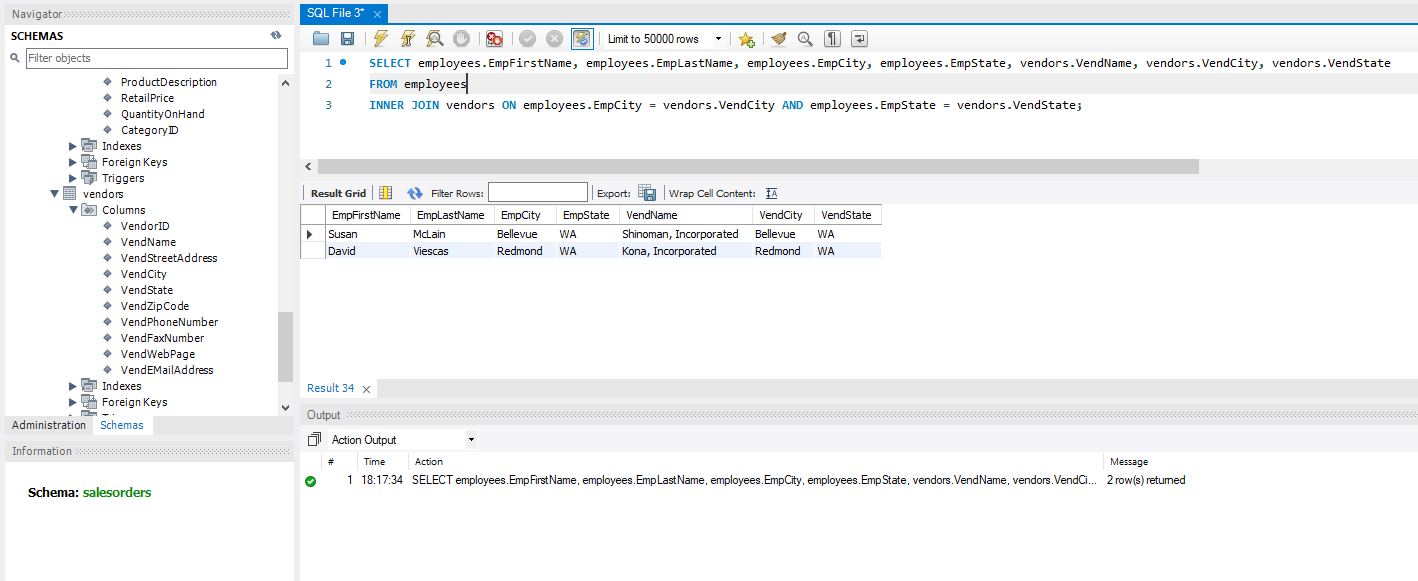
WHERE vendors.VendState LIKE 'TX' AND product\_vendors.WholesalePrice > 75;

1. Columns: ProductName, WholesalePrice, VendName, VendState
2. Expected Row Count: 12
3. Screenshot: 
4. **Show employees who live in the same city and state as our vendors.**
5. Query:

SELECT employees.EmpFirstName, employees.EmpLastName, employees.EmpCity, employees.EmpState, vendors.VendName, vendors.VendCity, vendors.VendState

FROM employees

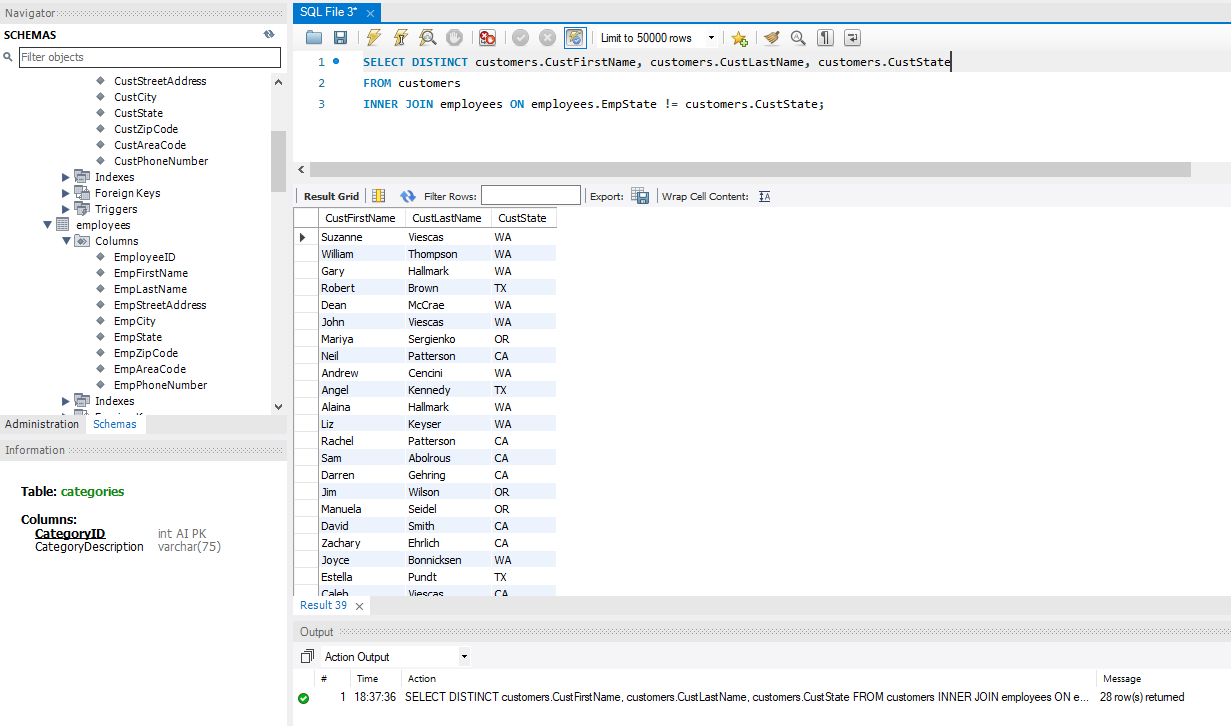
INNER JOIN vendors ON employees.EmpCity = vendors.VendCity AND employees.EmpState = vendors.VendState;

1. Columns: EmpFirstName, EmpLastName, EmpCity, EmpState, VendName, VendCity, VendState
2. Expected Row Count: 2
3. Screenshot: 
4. **Display customers who have no sales rep (employees) in the same state.**
5. Query:

SELECT DISTINCT customers.CustFirstName, customers.CustLastName, customers.CustState

FROM customers

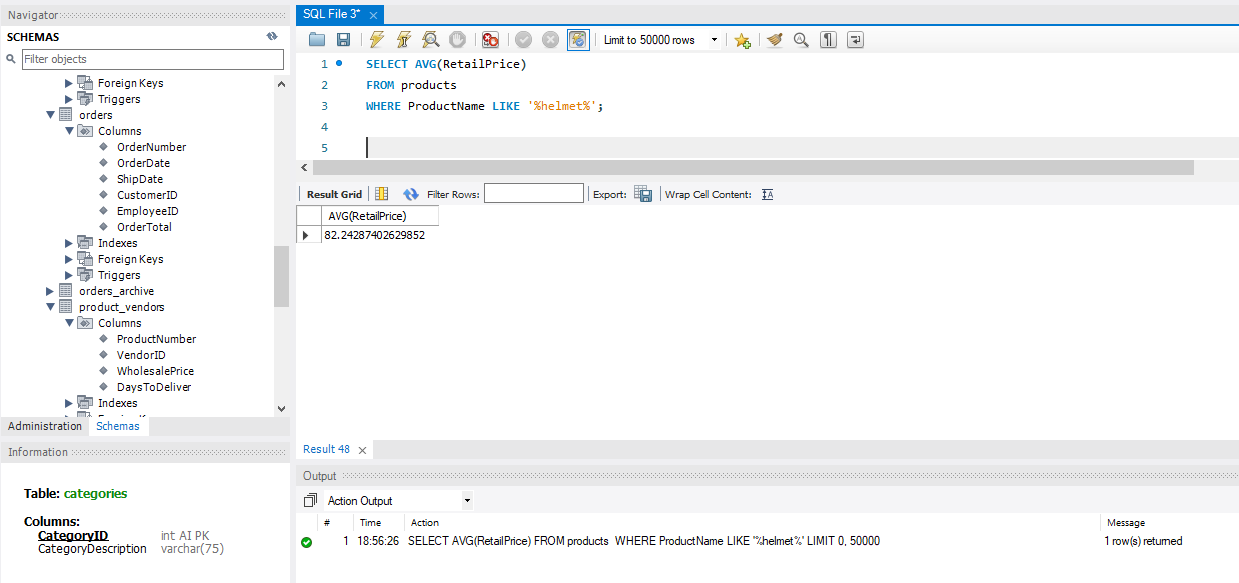
INNER JOIN employees ON employees.EmpState != customers.CustState;

1. Columns: CustFirstName, CustLastName, CustState
2. Expected Row Count: 28
3. Screenshot: 
4. **What is the average quoted price of a helmet?**
5. Query:

SELECT AVG(RetailPrice)

FROM products

WHERE ProductName LIKE '%helmet%';

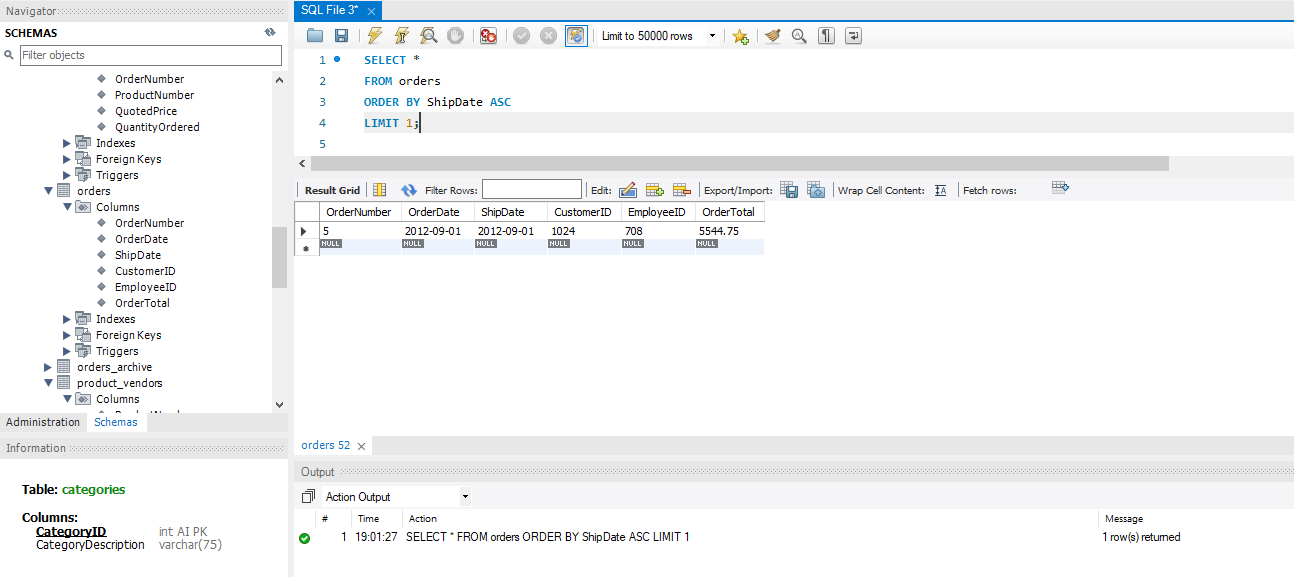
1. Columns: AVG(RetailPrice)
2. Expected Row Count: 1
3. Screenshot: 
4. **What was the date of the earliest ship date?**
5. Query:

SELECT \*

FROM orders

ORDER BY ShipDate ASC

LIMIT 1;

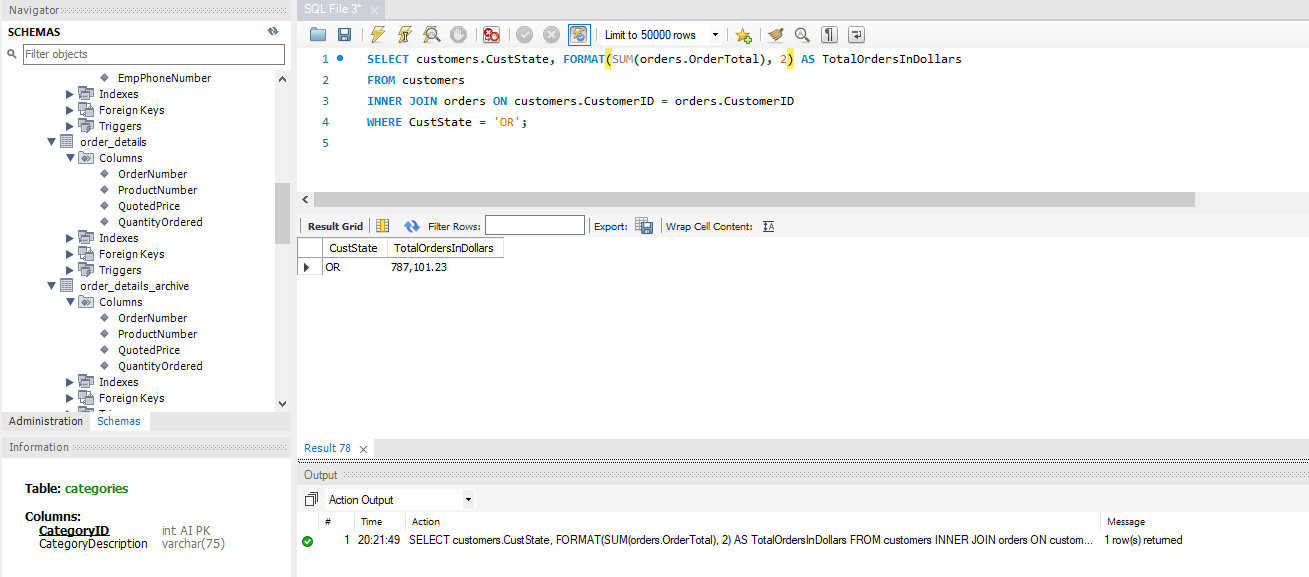
1. Columns: OrderNumber, OrderDate, ShipDate, CustomerID, EmployeeID, OrderTotal
2. Expected Row Count: 1
3. Screenshot: 
4. **What is the total amount (in dollars) of orders from the state of Oregon?**
5. Query:

SELECT customers.CustState, SUM(orders.OrderTotal)

FROM customers

INNER JOIN orders ON customers.CustomerID = orders.CustomerID

WHERE CustState = 'OR';

1. Columns: CustState, TotalOrdersInDollars
2. Expected Row Count: 1
3. Screenshot: 
4. **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.**
5. Query:

SELECT employees.EmpFirstName, employees.EmpLastName, FORMAT(SUM(orders.OrderTotal), 2) AS TotalSales, SUM(order\_details.QuantityOrdered) AS TotalItemQuantity, AVG(order\_details.QuotedPrice) AS AverageSalesPrice

FROM employees

INNER JOIN orders ON employees.EmployeeID = orders.EmployeeID

INNER JOIN order\_details on orders.OrderNumber = order\_details.OrderNumber

GROUP BY employees.EmployeeID

ORDER BY AverageSalesPrice DESC;

1. Columns: EmpFirstName, EmpLastName, TotalSales, TotalItemQuantity, AverageSalesPrice
2. Expected Row Count: 8
3. Screenshot: 