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

1. Expected Row Count: 28
2. Screenshot: Screenshot #1
3. **Show a list of states, in reverse alphabetical order, where our vendors are located, and include the names of the vendor.**
4. Query:

SELECT vendname, vendstate FROM vendors

ORDER BY(vendstate) DESC

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

UPDATE products SET retailprice = retailprice \* 1.07

SELECT productname, retailprice FROM products

1. Columns: productname, retailprice
2. Expected Row Count: 40
3. Screenshot: Screenshot #3
4. **Show a list of orders made by each customer in ascending date order.**
5. Query:

SELECT CONCAT(customers.custfirstname, ' ', customers.custlastname) AS customer,

orders.orderdate, orders.customerid, orders.ordertotal FROM customers

JOIN orders ON orders.customerid = customers.customerid

ORDER BY(orderdate) ASC

1. Columns: customer, orderdate, customerid, ordertotal
2. Expected Row Count: 944
3. Screenshot: Screenshot #4
4. **Give the names of all vendors based in Albany, Anchorage, and Dallas.**
5. Query:

SELECT vendname, vendcity FROM vendors

WHERE vendcity LIKE '%Albany%'

OR vendcity LIKE '%Anchorage%'

OR vendcity LIKE '%Dallas%'

1. Columns: vendname, vendcity
2. Expected Row Count: 3
3. Screenshot: Screenshot #5
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

1. Expected Row Count: 9
2. Screenshot: Screenshot #6
3. **What vendors do we work with that don’t have an email address?**
4. Query:

SELECT vendname, vendemailaddress FROM vendors

WHERE vendemailaddress IS NULL

1. Columns: vendname, vendemailaddress
2. Expected Row Count: 5
3. Screenshot: Screenshot #7
4. **List employees and the dates their orders shipped sorted by order date.**
5. Query:

SELECT CONCAT(employees.empfirstname, ' ', employees.emplastname) AS employee,

orders.orderdate, orders.shipdate FROM employees

JOIN orders ON employees.employeeid = orders.employeeid

ORDER BY(orders.orderdate) ASC

1. Columns: employee, orderdate, shipdate
2. Expected Row Count: 944
3. Screenshot: Screenshot #8
4. **Show the vendors and products they supply to us for products over $75 for vendors in Texas.**
5. Query:

SELECT vendors.vendname, vendors.vendstate, products.productname, products.retailprice FROM vendors, products

WHERE products.retailprice > 75 AND vendors.vendstate = 'TX'

1. Columns: vendname, vendstate, productname, retailprice
2. Expected Row Count: 32
3. Screenshot: Screenshot #9
4. **Show employees who live in the same city and state as our vendors.**
5. Query:

SELECT CONCAT(employees.empfirstname, ' ', employees.emplastname) AS employee, vendors.vendname,

CONCAT(vendors.vendcity, ', ', vendors.vendstate) AS city\_state FROM employees JOIN vendors

ON CONCAT(vendors.vendcity, ', ', vendors.vendstate) = CONCAT(employees.empcity, ', ', employees.empstate);

1. Columns: employee, vendname, city\_state
2. Expected Row Count: 2
3. Screenshot: Screenshot #10
4. **Display customers who have no sales rep (employees) in the same state.**
5. Query:

SELECT CONCAT(customers.custfirstname, ' ', customers.custlastname) AS customer, customers.custstate

FROM customers LEFT JOIN employees ON customers.custstate = employees.empstate

WHERE employees.empstate IS NULL

1. Columns: customer, custstate
2. Expected Row Count: 11
3. Screenshot: Screenshot #11
4. **What is the average quoted price of a helmet?**
5. Query:

SELECT AVG(retailprice) AS average\_price FROM products

WHERE categoryid = 1 AND productname LIKE '%Helmet%'

1. Columns: average\_price
2. Expected Row Count: 1
3. Screenshot: Screenshot #12
4. **What was the date of the earliest ship date?**
5. Query: SELECT MIN(shipdate) AS earliest\_ship FROM orders
6. Columns: earliest\_ship
7. Expected Row Count: 1
8. Screenshot: Screenshot #13
9. **What is the total amount (in dollars) of orders from the state of Oregon?**
10. Query:

SELECT SUM(ordertotal) AS oregon\_total FROM orders

LEFT JOIN customers ON customers.customerid = orders.customerid

WHERE customers.custstate = 'OR'

1. Columns: oregon\_total
2. Expected Row Count: 1
3. Screenshot: Screenshot #14
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 CONCAT(employees.empfirstname, ' ', employees.emplastname) AS employee, SUM(orders.ordertotal) AS total\_sales,

COUNT(employees.employeeid = orders.employeeid) AS item\_quantity, AVG(orders.ordertotal) AS average\_price

FROM employees LEFT JOIN orders ON employees.employeeid = orders.employeeid GROUP BY(employee)

ORDER BY(average\_price) DESC

1. Columns: employee, total\_sales, item\_quantity, average\_price
2. Expected Row Count: 9
3. Screenshot: Screenshot #15