Solutions for the Extra Problems in Module 5

The extra problems use the Order Entry Database as described in the Order Entry Database Background document. The course website also contains CREATE TABLE and INSERT statements for Oracle, MySQL, and PostgreSQL.

1. List the order number, order date, customer number, customer name (first and last), employee number, and employee name (first and last) of January 2017 orders placed by Colorado customers.
2. List the customer number, name (first and last), order number, order date, employee number, employee name (first and last), product number, product name, and order cost `(OrdLine.Qty * ProdPrice)` for products ordered on January 23, 2017, in which the order cost exceeds $150.
3. List the order number and total amount for orders placed on January 23, 2017. The total amount of an order is the sum of the quantity times the product price of each product on the order.
4. List the order number, order date, customer name (first and last), and total amount for orders placed on January 23, 2017. The total amount of an order is the sum of the quantity times the product price of each product on the order.
5. Insert yourself as a new row in the *Customer* table.
6. Insert an imaginary friend as a new row in the *Employee* table.
7. Increase the price by 10 percent of products containing the words Ink Jet.
8. Delete the new row added to the *Customer* table.

1.

Oracle SQL solutions

```
SELECT OrdNo, OrdDate, Customer.CustNo, CustFirstName, CustLastName,
        Employee.EmpNo, EmpFirstName, EmpLastName
  FROM OrderTbl, Customer, Employee
  WHERE CustState = 'CO' AND OrdDate BETWEEN '1-Jan-2017' AND '31-Jan-2017'
     AND OrderTbl.CustNo = Customer.CustNo
     AND OrderTbl.EmpNo = Employee.EmpNo;


SELECT OrdNo, OrdDate, Customer.CustNo, CustFirstName, CustLastName,
        Employee.EmpNo, EmpFirstName, EmpLastName
  FROM OrderTbl INNER JOIN Customer ON OrderTbl.CustNo = Customer.CustNo
        INNER JOIN Employee ON OrderTbl.EmpNo = Employee.EmpNo
  WHERE CustState = 'CO' AND OrdDate BETWEEN '1-Jan-2017' AND '31-Jan-2017';
```

MySQL and PostgreSQL solutions

```
SELECT OrdNo, OrdDate, Customer.CustNo, CustFirstName, CustLastName,
        Employee.EmpNo, EmpFirstName, EmpLastName
  FROM OrderTbl, Customer, Employee
  WHERE CustState = 'CO' AND OrdDate BETWEEN '2017-01-01' AND '2017-01-31'
     AND OrderTbl.CustNo = Customer.CustNo
     AND OrderTbl.EmpNo = Employee.EmpNo;


SELECT OrdNo, OrdDate, Customer.CustNo, CustFirstName, CustLastName,
        Employee.EmpNo, EmpFirstName, EmpLastName
  FROM OrderTbl INNER JOIN Customer ON OrderTbl.CustNo = Customer.CustNo
        INNER JOIN Employee ON OrderTbl.EmpNo = Employee.EmpNo
  WHERE CustState = 'CO' AND OrdDate BETWEEN '2017-01-01' AND '2017-01-31';
```

2.

Oracle solutions

```
SELECT Customer.CustNo, CustFirstName, CustLastName, OrderTbl.OrdNo,
        OrdDate, Employee.EmpNo, EmpFirstName, EmpLastName,
        Product.ProdNo, ProdName, ProdPrice*Qty AS OrderCost
  FROM OrderTbl, OrdLine, Product, Customer, Employee
  WHERE OrdDate = '23-Jan-2017' AND ProdPrice*Qty > 150
     AND OrderTbl.OrdNo = OrdLine.OrdNo
     AND OrdLine.ProdNo = Product.ProdNo
     AND OrderTbl.CustNo = Customer.CustNo
     AND Employee.EmpNo = OrderTbl.EmpNo;

SELECT Customer.CustNo, CustFirstName, CustLastName, OrderTbl.OrdNo,
```

```
        OrdDate, Employee.EmpNo, EmpFirstName, EmpLastName,
        Product.ProdNo, ProdName, ProdPrice*Qty AS OrderCost
  FROM OrderTbl INNER JOIN Customer ON OrderTbl.CustNo = Customer.CustNo
       INNER JOIN Employee ON OrderTbl.EmpNo = Employee.EmpNo
       INNER JOIN OrdLine ON OrderTbl.OrdNo = OrdLine.OrdNo
       INNER JOIN Product ON OrdLine.ProdNo = Product.ProdNo
  WHERE OrdDate = '23-Jan-2017' AND ProdPrice*Qty > 150;
```

MySQL and PostgreSQL solutions

```
SELECT Customer.CustNo, CustFirstName, CustLastName, OrderTbl.OrdNo,
        OrdDate, Employee.EmpNo, EmpFirstName, EmpLastName,
        Product.ProdNo, ProdName, ProdPrice*Qty AS OrderCost
  FROM OrderTbl, OrdLine, Product, Customer, Employee
  WHERE OrdDate = '2017-01-23' AND ProdPrice*Qty > 150
     AND OrderTbl.OrdNo = OrdLine.OrdNo
     AND OrdLine.ProdNo = Product.ProdNo
     AND OrderTbl.CustNo = Customer.CustNo
     AND Employee.EmpNo = OrderTbl.EmpNo;

SELECT Customer.CustNo, CustFirstName, CustLastName, OrderTbl.OrdNo,
        OrdDate, Employee.EmpNo, EmpFirstName, EmpLastName,
        Product.ProdNo, ProdName, ProdPrice*Qty AS OrderCost
  FROM OrderTbl INNER JOIN Customer ON OrderTbl.CustNo = Customer.CustNo
       INNER JOIN Employee ON OrderTbl.EmpNo = Employee.EmpNo
       INNER JOIN OrdLine ON OrderTbl.OrdNo = OrdLine.OrdNo
       INNER JOIN Product ON OrdLine.ProdNo = Product.ProdNo
  WHERE OrdDate = '2017-01-23' AND ProdPrice*Qty > 150;
```

3.

Oracle solutions

```
SELECT OrderTbl.OrdNo, SUM(Qty*ProdPrice) AS TotOrdAmt
  FROM OrderTbl, OrdLine, Product
  WHERE OrdDate = '23-Jan-2017'
     AND OrderTbl.OrdNo = OrdLine.OrdNo
     AND OrdLine.ProdNo = Product.ProdNo
  GROUP BY OrderTbl.OrdNo;

SELECT OrderTbl.OrdNo, SUM(Qty*ProdPrice) AS TotOrdAmt
  FROM OrderTbl INNER JOIN OrdLine ON OrderTbl.OrdNo = OrdLine.OrdNo
       INNER JOIN Product ON OrdLine.ProdNo = Product.ProdNo
  WHERE OrdDate = '23-Jan-2017'
  GROUP BY OrderTbl.OrdNo;
```

MySQL and PostgreSQL solutions

```
SELECT OrderTbl.OrdNo, SUM(Qty*ProdPrice) AS TotOrdAmt
  FROM OrderTbl, OrdLine, Product
  WHERE OrdDate = '2017-01-23'
       AND OrderTbl.OrdNo = OrdLine.OrdNo
       AND OrdLine.ProdNo = Product.ProdNo
  GROUP BY OrderTbl.OrdNo;

SELECT OrderTbl.OrdNo, SUM(Qty*ProdPrice) AS TotOrdAmt
  FROM OrderTbl INNER JOIN OrdLine ON OrderTbl.OrdNo = OrdLine.OrdNo
            INNER JOIN Product ON OrdLine.ProdNo = Product.ProdNo
  WHERE OrdDate = '2017-01-23'
  GROUP BY OrderTbl.OrdNo;
```

4.

Oracle solutions

```
SELECT OrderTbl.OrdNo, OrdDate, CustFirstName, CustLastName,
              SUM(Qty*ProdPrice) AS TotOrdAmt
  FROM OrderTbl, OrdLine, Product, Customer
  WHERE OrdDate = '23-Jan-2017'
       AND OrderTbl.OrdNo = OrdLine.OrdNo
       AND OrdLine.ProdNo = Product.ProdNo
       AND Customer.CustNo = OrderTbl.CustNo
  GROUP BY OrderTbl.OrdNo, OrdDate, CustFirstName, CustLastName;

SELECT OrderTbl.OrdNo, OrdDate, CustFirstName, CustLastName,
              SUM(Qty*ProdPrice) AS TotOrdAmt
  FROM OrderTbl INNER JOIN OrdLine ON OrderTbl.OrdNo = OrdLine.OrdNo
             INNER JOIN Product ON OrdLine.ProdNo = Product.ProdNo
             INNER JOIN Customer ON Customer.CustNo = OrderTbl.CustNo
   WHERE OrdDate = '23-Jan-2017'
   GROUP BY OrderTbl.OrdNo, OrdDate, CustFirstName, CustLastName;
```

MySQL and PostgreSQL solutions

```
SELECT OrderTbl.OrdNo, OrdDate, CustFirstName, CustLastName,
              SUM(Qty*ProdPrice) AS TotOrdAmt
  FROM OrderTbl, OrdLine, Product, Customer
 WHERE OrdDate = '2017-01-23'
       AND OrderTbl.OrdNo = OrdLine.OrdNo
       AND OrdLine.ProdNo = Product.ProdNo
       AND Customer.CustNo = OrderTbl.CustNo
  GROUP BY OrderTbl.OrdNo, OrdDate, CustFirstName, CustLastName;
```

```
SELECT OrderTbl.OrdNo, OrdDate, CustFirstName, CustLastName,
            SUM(Qty*ProdPrice) AS TotOrdAmt
  FROM OrderTbl INNER JOIN OrdLine ON OrderTbl.OrdNo = OrdLine.OrdNo
          INNER JOIN Product ON OrdLine.ProdNo = Product.ProdNo
          INNER JOIN Customer ON Customer.CustNo = OrderTbl.CustNo
  WHERE OrdDate = '2017-01-23'
  GROUP BY OrderTbl.OrdNo, OrdDate, CustFirstName, CustLastName;
```

5.

```
INSERT INTO Customer
  (CustNo, CustFirstName, CustLastName, CustStreet, CustCity, CustState,
   CustZip, CustBal)
VALUES ('C9999999', 'Michael', 'Mannino', '123 Any Street', 'MyTown', 'CO',
           '80217-0211', 500);
```

6.

```
INSERT INTO Employee
  ( EmpNo, EmpFirstName, EmpLastName, EmpPhone, EmpCommRate, EmpEmail)
VALUES ('E9999999', 'Mary', 'Mannino', '(720)543-1234', 0.04,
'Mary.Mannino@abc.com');
```

7.

Oracle and PostgreSQL solutions

```
UPDATE Product
  SET ProdPrice = ProdPrice * 1.1
  WHERE ProdName LIKE '%Ink Jet%';
```

MySQL solution

UPDATE and DELETE statements will not execute if WHERE conditions do not reference the primary key. The SQL_SAFE_UPDATES option set to 0 allows UPDATE and DELETE statements without a WHERE condition on the primary key.

```
SET SQL_SAFE_UPDATES = 0;

UPDATE Product
  SET ProdPrice = ProdPrice * 1.1
  WHERE ProdName LIKE '%Ink Jet%';

SET SQL_SAFE_UPDATES = 1;
```

8.

```
DELETE FROM Customer
  WHERE CustNo = 'C9999999';
```

```
DELETE FROM Employee
  WHERE EmpNo = 'E9999999';
```