Group member names: Aaron Henry, Michael Ly, Max Phillips, Dhara Patel, Ryan Bell

**Exercise 9: Problem Solving** (10 points)

**Instructions:**

1. This is a group assignment. You only need to submit one solved exercise per group.
2. You would only submit your SQL statements one single PDF file. If there are any notes you would like me to look at, please include those with the SQL statement.
3. Every group member needs to run the SQL statements in their account. I have access to your database and will examine the individual work done.
4. Once finished, upload the file to the appropriate assignment response in blackboard.
5. You may upload as many attempts as you may like. Please note that only the most recent file uploaded file would be graded.
6. Do not handwrite any responses.
7. If you have any additional information you would like me to know about this assignment, you may provide that to me at the end of this document as a note.

**Use the data model shown at the bottom of this document for this assignment:**

What to do:

1. This assignment must be completed in SQL Server. No other DBMS is allowed.
2. All SQL statements written by different team members should be collated in this one single document in its appropriate area. No exceptions.
3. **Write a query for each of the following questions. The tables are in a database called PET. You can access the tables by qualifying the tables as in the following example: SELECT \* FROM PET..CUSTOMER**
4. Copy paste these SQL statements in appropriate space in one single document.
5. ERD for the database is provided at the end of this document.
6. You would not create/copy this database into your account. It is included in your schema and you would access it following instructions in #3 above.

**Assignment:**

1. Which merchandise items have an average sale price more than 50 percent higher than their average purchase cost?



SELECT M.ItemID, M.Description, AVG(O.Cost) as AvgOfCost, AVG(SalePrice) AS AvgOfSalePrice

FROM PET..Merchandise M

INNER JOIN PET..OrderItem O ON M.ItemID = O.ItemID

INNER JOIN PET..SaleItem S ON M.ItemID = S.ItemID

GROUP BY M.ItemID, M.Description

HAVING AVG(SalePrice) > AVG(Cost) \* 1.5

ORDER BY M.ItemID;

1. On average, which supplier charges the highest shipping cost as a percent of the merchandise order total. OrderTotal is Sum(Quantity\*Cost) and PctShipCost is Avg(ShippingCost/OrderTotal)?



SELECT TOP 1 MO.SupplierID, S.Name, ROUND(AVG(MO.ShippingCost/O.OrderTotal), 2) AS PctShipCost

FROM PET..MerchandiseOrder MO

INNER JOIN PET..Supplier S ON MO.SupplierID = S.SupplierID

INNER JOIN (SELECT O.PONumber, SUM(O.Quantity \* O.Cost) AS OrderTotal

FROM PET..OrderItem O

WHERE O.PONumber IN (SELECT MO.PONumber

FROM PET..MerchandiseOrder MO)

GROUP BY O.PONumber) O ON MO.PONumber = O.PONumber

GROUP BY MO.SupplierID, S.Name

ORDER BY PctShipCost DESC;

1. Which customer has given us the most total money for animals and merchandise? SumofSalePrice is total money from animal sales given by sum(SalePrice). MerchTotal is total money from Merchandise sale given by Sum(Quantity\*SalePrice). GrandTotal is SumofSalePrice added with MerchTotal. You may want to use ISNULL function to convert null values to zero. This is important because if a customer has SumofSalePrice as null while MerchTotal is not null, the addition of the two would be null. After all, any number + NULL = NULL. (ISNULL(MerchTotal,0)+ISNULL(SumOfSalePrice, 0)) AS GrandTotal



SELECT TOP 1 C.CustomerID, C.LastName, C.FirstName, SUM(ISNULL(SI.MerchTotal, 0)) AS MerchTotal, SUM(ISNULL(SA.SumOfSalePrice, 0)) AS SumOfSalePrice, SUM(ISNULL(SI.MerchTotal, 0)) + SUM(ISNULL(SA.SumOfSalePrice, 0)) AS GrandTotal

FROM PET..Sale S

INNER JOIN PET..Customer C ON S.CustomerID = C.CustomerID

LEFT JOIN (SELECT SI.SaleID, SUM(SI.Quantity \* SI.SalePrice) AS MerchTotal

FROM PET..SaleItem SI

GROUP BY SI.SaleID) SI ON S.SaleID = SI.SaleID

LEFT JOIN (SELECT SA.SaleID, SUM(ISNULL(SA.SalePrice, 0)) AS SumOfSalePrice

FROM PET..SaleAnimal SA

GROUP BY SA.SaleID) SA ON S.SaleID = SA.SaleID

GROUP BY C.CustomerID, C.LastName, C.FirstName

ORDER BY GrandTotal DESC;

