Lawrence Chung

1490133

OMIS 105 FINAL

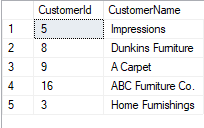
-- 1. List the customers (ID and name) who live in California or New York. Order them by zip code, from high to low. (1 point)

SELECT CustomerId, CustomerName

FROM Customer\_T

WHERE CustomerState='CA' OR CustomerState='NY'

ORDER BY CustomerPostalCode DESC



-- 2. Write an SQL query that will find any salesperson (ID) who have not facilitated any order in 2017. (2 points)

SELECT SalespersonID

FROM Salesperson\_T

WHERE SalesPersonID NOT IN

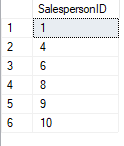
(

SELECT SalesPersonID

FROM Order\_T

WHERE YEAR(OrderDate) = '2017'

)



-- 3. Display the product line ID, product line name and the average standard price for Walnut made products in those product lines that have an average standard price of at least $500. (3 points)

SELECT Product\_T.ProductLineID, ProductLineName, AVG(ProductStandardPrice) AS AveragePrice

FROM Product\_T INNER JOIN ProductLine\_T ON Product\_T.ProductLineID = ProductLine\_T.ProductLineID

WHERE ProductFinish = 'Walnut'

GROUP BY Product\_T.ProductLineID, ProductLineName

HAVING AVG(ProductStandardPrice) >= 500



-- 4. List, in alphabetical order, the names of all employees (managers) who are managing people in the SM1 work center. (3 points)

SELECT EmployeeName

FROM Employee\_T INNER JOIN

(

SELECT WorksIn\_T.EmployeeID, WorkCenterID, EmployeeSupervisor

FROM WorksIn\_T INNER JOIN Employee\_T On WorksIn\_T.EmployeeID = Employee\_T.EmployeeID

WHERE WorkCenterID = 'SM1'

) Temp\_T ON Temp\_T.EmployeeSupervisor = Employee\_T.EmployeeID

ORDER BY EmployeeName

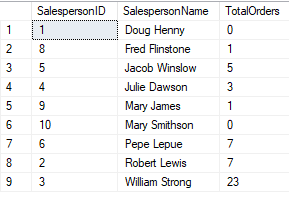


-- 5. Write a query to list the SalesPersonID, SalesPersonName and the number of orders facilitated (label as TotalOrders). If a salesperson does not facilitate any orders, display the result with a total of 0. (3 points)

SELECT Salesperson\_T.SalespersonID, SalespersonName, COUNT(OrderID) AS TotalOrders

FROM Salesperson\_T LEFT OUTER JOIN Order\_T ON Salesperson\_T.SalespersonID = Order\_T.SalesPersonID

GROUP BY Salesperson\_T.SalespersonID, SalespersonName



-- 6. Retrieve a list of the customer states and the total profit earned from all of their orders on the Cherry and Birch products. Profit earned on a given order is the order total minus the total cost to produce the products sold. The cost of production for a given product is the total cost of all raw materials consumed in production. The query should output the state, total amount of all orders placed by its customers, total cost of production of all products sold in these orders, and total dollars profit earned. The output should be limited to the states with at least $2000 profit. Sort the results in descending order of profit. (3 points)

SELECT CustomerState, SUM(OrderedQuantity \* ProductStandardPrice) AS TotalOfAllOrders, SUM(TotalCost \* OrderedQuantity) AS TotalCost, (SUM(OrderedQuantity \* ProductStandardPrice) - SUM(TotalCost \* OrderedQuantity)) AS TotalProfit

FROM Customer\_T INNER JOIN Order\_T ON Customer\_T.CustomerId = Order\_T.CustomerID INNER JOIN OrderLine\_T ON Order\_T.OrderID = OrderLine\_T.OrderID INNER JOIN Product\_T ON OrderLine\_T.ProductID = Product\_T.ProductID INNER JOIN

(

SELECT Product\_T.ProductID, SUM(MaterialStandardPrice\*QuantityRequired) AS TotalCost

FROM Product\_T INNER JOIN Uses\_T ON Product\_T.ProductID = Uses\_T.ProductID INNER JOIN RawMaterial\_T ON Uses\_T.MaterialID = RawMaterial\_T.MaterialID

WHERE ProductFinish IN ('Cherry', 'Birch')

GROUP BY Product\_T.ProductID

) TotalCosts\_T

ON Product\_T.ProductID = TotalCosts\_T.ProductID

GROUP BY CustomerState

HAVING (SUM(OrderedQuantity \* ProductStandardPrice) - SUM(TotalCost \* OrderedQuantity)) >= 2000

ORDER BY (SUM(OrderedQuantity \* ProductStandardPrice) - SUM(TotalCost \* OrderedQuantity)) DESC



-- 7. Write an SQL query to list the order ID, product ID, and ordered quantity for all ordered products for which the ordered quantity is at least three times the average ordered quantity for that product. Also, the average ordered quantity for each product should appear at the end in a separate column. Limit the output to the products with the average ordered quantity of at least 2 . (3 points)

SELECT OrderID, OrderLine\_T.ProductID, OrderedQuantity, ProductAverage

FROM OrderLine\_T INNER JOIN

(

SELECT ProductID, AVG(OrderedQuantity) AS ProductAverage

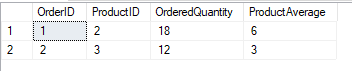
FROM OrderLine\_T

GROUP BY ProductID

)Temp\_T ON OrderLine\_T.ProductID = Temp\_T.ProductID

WHERE OrderedQuantity >= ProductAverage \* 3 AND ProductAverage >= 2

ORDER BY OrderID



-- 8. For the those Ash materials that are provided by at least two vendors, write a query that lists the vendor ID, vendor name, material ID, material name, and size. (4 points)

SELECT Vendor\_T.VendorID, VendorName, Supplies\_T.MaterialID, MaterialName, Size

FROM Vendor\_T INNER JOIN Supplies\_T ON Vendor\_T.VendorID = Supplies\_T.VendorID INNER JOIN RawMaterial\_T ON Supplies\_T.MaterialID = RawMaterial\_T.MaterialID INNER JOIN

(

SELECT Supplies\_T.MaterialID, COUNT(Vendor\_T.VendorID) AS VendorCount

FROM Vendor\_T INNER JOIN Supplies\_T ON Vendor\_T.VendorID = Supplies\_T.VendorID INNER JOIN RawMaterial\_T ON Supplies\_T.MaterialID = RawMaterial\_T.MaterialID

GROUP BY Supplies\_T.MaterialID

)Temp\_T ON RawMaterial\_T.MaterialID = Temp\_T.MaterialID

WHERE VendorCount >= 2 AND Material = 'Ash'



-- 9. Display the salesperson names of all salespersons who have ordered at least one Cherry product and at least one Walnut product on the same or different orders. (4 points)

SELECT DISTINCT Salesperson\_T.SalespersonName

FROM Salesperson\_T INNER JOIN Order\_T ON Salesperson\_T.SalespersonID = Order\_T.SalesPersonID

WHERE

(Salesperson\_T.SalespersonName IN

(

SELECT Salesperson\_T.SalespersonName

FROM Salesperson\_T INNER JOIN Order\_T ON Salesperson\_T.SalespersonID = Order\_T.SalesPersonID INNER JOIN OrderLine\_T ON Order\_T.OrderID = OrderLine\_T.OrderID INNER JOIN Product\_T ON OrderLine\_T.ProductID = Product\_T.ProductID

WHERE ProductFinish = 'Cherry')) AND

(Salesperson\_T.SalespersonName IN

(

SELECT Salesperson\_T.SalespersonName

FROM Salesperson\_T INNER JOIN Order\_T ON Salesperson\_T.SalespersonID = Order\_T.SalesPersonID INNER JOIN OrderLine\_T ON Order\_T.OrderID = OrderLine\_T.OrderID INNER JOIN Product\_T ON OrderLine\_T.ProductID = Product\_T.ProductID

WHERE ProductFinish = 'Walnut'

))



-- 10. Retrieve a list of salespersons and the orders they have each facilitated, including order ID, order date, and order total price. Each order facilitated by a salesperson contributes to their total sales. Also provide the contribution of each order to the salesperson's total sales, as a percentage of total sales. The query should output the salesperson name, order ID, order date, order total price, distinct number of orders facilitated by salesperson, and percentage of salesperson total sales. Exclude orders with more than 30% contribution. Sort the list first in alphabetical order of salesperson name, then by ascending order date, and then with larger order totals listed first. (4 points)

SELECT SalespersonName, Order\_T.OrderID, OrderDate, SUM(OrderedQuantity \* ProductStandardPrice) AS OrderTotalPrice, DistinctOrders, SUM((OrderedQuantity \* ProductStandardPrice) / TotalSales \* 100) AS PercentageofSales

FROM Salesperson\_T INNER JOIN Order\_T ON Salesperson\_T.SalespersonID = Order\_T.SalesPersonID INNER JOIN OrderLine\_T ON Order\_T.OrderID = OrderLine\_T.OrderID INNER JOIN Product\_T ON OrderLine\_T.ProductID = Product\_T.ProductID INNER JOIN

(

SELECT SalesPersonID, SUM(OrderedQuantity \* ProductStandardPrice) AS TotalSales, COUNT (DISTINCT Order\_T.OrderID) AS DistinctOrders

FROM Order\_T INNER JOIN OrderLine\_T ON Order\_T.OrderID = OrderLine\_T.OrderID INNER JOIN Product\_T ON OrderLine\_T.ProductID = Product\_T.ProductID

GROUP BY SalesPersonID

) TotalSales\_T

ON Order\_T.SalespersonID = TotalSales\_T.SalespersonID

GROUP BY SalespersonName, Order\_T.OrderID, OrderDate, DistinctOrders

HAVING SUM(((OrderedQuantity \* ProductStandardPrice) / TotalSales) \* 100) <= 30

ORDER BY SalespersonName ASC, OrderDate ASC, OrderTotalPrice DESC

