**MIDTERM 1 – Review Questions**

**A diagram of a product

Description automatically generated**

Use the above ERD to answer the following SQL problems:

1. Return the store with the most staffs.

SELECT TOP 1 StoreName, COUNT(\*)

FROM Stores JOIN Staff ON Stores.StoreID = Staff.StoreID

GROUP BY StoreName

ORDER BY COUNT(\*) DESC

1. Return the customers that have bought products with a list price of $2000 in the last month.

SELECT DISTINCT Customers.\*

FROM Customers JOIN Orders ON Customers.CustomerID = Orders.CustomerID

JOIN OrderItems ON OrderItems.OrderID = Orders.OrderID  
 JOIN Products ON Products.ProductID = OrderItems.ProductID

WHERE ListPrice = 2000 AND DATEDIFF(MONTH, OrderDate, GETDATE()) <=1

1. Return the number of times that products with model year of 2017 and price of $1000 or more, have been ordered.

SELECT Products.ProducID, ProductName, BrandID, CategoryID, COUNT(\*)

FROM Products JOIN OrderItems ON Products.ProductID = OrderItems.ProductID

WHERE ModelYear = 2017 AND ListPrice >=1000

GROUP BY Products.ProducID, ProductName, BrandID, CategoryID,

1. Return the CA customers that have purchased from an out of state store.

SELECT Customers.\*

FROM Customers JOIN Orders ON Customers.CustomerID = Orders.CustomerID

JON Stores ON Stores.StoreID = Orders.StoreID

WHERE Customers.State = ‘CA’ AND Orders.State <> ‘CA’

1. List the out of stock products for Store ID 1.

SELECT Products.\*

FROM Products JOIN Stocks ON Products.ProductID = Stocks.ProductID

WHERE StoreID =1 AND Quantity =0

1. Return the customers that their number of orders is above average.

**WITH NumberOfOrders AS** (

SELECT CustomerID, COUNT(\*) AS ‘NumOrders’

FROM Orders

GROUP BY CustomerID)

SELECT CustomerID, COUNT(\*) AS ‘NumOrders’ **INTO #NumberOfOrders**

FROM Orders

GROUP BY CustomerID

**---- USE THE TEMP TABLE IN THE SUB QUERY**

SELECT CustomerID, COUNT(\*) AS ‘NumOrders’

FROM Orders

GROUP BY CustomerID

HAVING COUNT(\*) > (*SELECT AVG(NumOrders) FROM #NumberOfOrders*)

1. Return the name of all the managers.

**--USE SELF JOIN**

SELECT DISTINCT Manager. FirstName, Manager.LastName

FROM Staff JOIN Staff AS Manger ON Staff.ManagerID = Manager. StaffID

**--USE SUB QUERY**

SELECT Fristname, Lastname

FROM Staff

WHERE StaffID IN (SELECT DISTINT ManagerID FROM Staff)

1. For 2016, return the store that has the most orders in Spring season (March, April, May).

SELECT TOP 1 StoreName, COUNT(\*)

FROM Stores JOIN Orders ON Stores.StoreID = Orders.StoreID

WHERE YEAR(OrderDate) = 2016 AND (MONTH(OrderDate) =3 OR MONTH(OrderDate)=4 OR MONTH(OrderDate)=5)

-- WHERE YEAR(OrderDate) = 2016 AND MONTH(OrderDate) IN (3,4,5)

GROUP BY StoreName

ORDER BY COUNT(\*) DESC

A screenshot of a computer

Description automatically generated

Use the above ERD to answer the following SQL problems:

1. Return the products and total quantity that was ordered in the last month. Sort the result with the most popular product at the top.

SELECT ProductName, SUM(Quantity)

FROM Products JOIN OrderItems ON Products.ProductID = OrderItems. ProductID

JOIN Orders ON Orders.OrderID = OrderItems.OrderID

WHERE DATEDIFF(MONTH,OrderDate,GETDATE()) <=1

GROUP BY ProductName

ORDER BY SUM(Quantity) DESC

1. Return the categories, number of products in each category and the list price of the more expensive product in that category. Sort the result set so the category with the most products appears first.

SELECT CategoryName, COUNT(\*), MAX(ListPrice)

FROM Categories JOIN Products ON Categories.CategoryID = Products.CategoryID

GROUP BY CategoryName

ORDER BY COUNT(\*) DESC

1. Return the customer information along with total shipping and the date of their most recent order.

SELECT Customers.CustomerID, Firstname, Lastname, SUM(ShipAmount), MAX(OrderDate)

FROM Customers JOIN Orders ON Customers.CustomerID = Orders.CustomerID

GROUP BY Customers.CustomerID, Firstname, Lastname

1. Return the customers with the count of their orders and the total amount for those orders. Order by the customer with the highest orders. SKIP
2. Modify Q3 to only count the line items that have an item price value that is greater than $400. SKIP
3. Write a query that answers this question: What is the total amount ordered for

each product? SKIP

1. Write a query that answers this question: Which customers have ordered more than one product? SKIP
2. What is the average number of days from the time the order is placed till the shipping date.

SELECT AVG(DATEDIFF(DAY, OrderDate,ShipDate))

FROM Orders

Average number of days for each category

SELECT CategoryName, AVG(DATEDIFF(DAY, OrderDate,ShipDate))

FROM Orders JOIN OrderItems ON Orders.OrderID = OrderItems.OrderID

JOIN Products ON Products.ProductID = OrderItems.ProductID

JOIN Categories ON Categories.CategoryID = Products.CategoryID

GROUP BY CategoryName

1. Assuming each credit card company will receive 2% of each order, return the amount each credit card company has made based on the orders placed so far.

SELECT CardType, SUM ((ItemPrice – DiscountAmount)\* Quantity) \*0.02

FROM FROM Orders JOIN OrderItems ON Orders.OrderID = OrderItems.OrderID

GROUP BY CardType

1. Return the year 2024 credit cards that have expired and those that haven’t. Add a column card status that will either say “Expired” or “Expires this year”.

The card expires column in not a date: 06/2024

SELECT CardType, CardNumber, CardExpires, “Expired” AS Status

FROM Orders

WHERE RIGHT(CardExpires,4) =2024 AND LEFT(CardExpires, 2) < MONTH(GETDATE())

UNION

SELECT CardType, CardNumber, CardExpires, “Expire this year” AS Status

FROM Orders

WHERE RIGHT(CardExpires,4) =2024 AND LEFT(CardExpires, 2) >= MONTH(GETDATE())

Return all the products that haven’t been ordered yet.

SELECT Products.\*

FROM Products LEFT JOIN OrderItems ON Products.ProductID = OrderItems.ProducID

WHERE OrderID IS NULL

SELECT \*

FROM Products

WHERE ProductID NOT IN (SELECT DISTINCT ProductID FROM OrderItems)

Return all the customers that have put at least 5 orders in the last 6 months. Sort the result with the customer with the most order at the top.

SELECT Customers.CustomerID, Firstname, Lastname, COUNT(\*)

FROM Customers JOIN Orders ON Customers.CustomerID = Orders.CustomerID

WHERE DATEDIFF(MONTH, OrderDate, GETDATE()) <=6

GROUP BY Customers.CustomerID, Firstname, Lastname

HAVING COUNT(\*) >=5

ORDER BY COUNT(\*) DESC

Action Queries:

1. The company shipped all the orders that has been placed in the week, today. Update their shipping date (Update the shipping date of all the orders that have been placed in the last week to today.)

UPDATE Orders

SET ShipDate = GETDATE()

WHERE DATEDIFF(DAY, OrderDate, GETDATE()) <=7

1. Add a new address for customerID 1: 3601 PacificAve, Stockton, CA 95211

INSERT INTO Addresses

VALUES (1, ‘3601 Pacific Ave’, NULL, ‘Stockton’, ‘CA’, 9511, NULL, NULL)

INSERT INTO Addresses (CustomerID, Line1, City, State,Zipcode)

VALUES (1, ‘3601 Pacific Ave’, ‘Stockton’, ‘CA’, 9511)

1. Delete all the CA Addresses.

DELETE \* FROM Addresses

WHERE State = ‘CA’