Chapter 6 Part 2

How to code summary queries

Exercises

1. **(15 Points)** Write a SELECT query which returns the amount of orders each customer has placed, including those who have made zero orders. Use a correlated subquery in the SELECT clause to retrieve the amount of orders. Return the following columns:

   - The **CustomerID** column from the **Customers** table.
   - A calculated column with alias **Fullname** which returns the customer's last name and firstname with a comma and space between them, example: (Haley, Shane).
   - The count of orders made by the customer. Use alias **TotalOrders** for the aggregated column.

   Sort the result set in descending sequence by **TotalOrders** and ascending sequence by **Fullname**.

2. **(10 Points)** Alter exercise 1 so that it does not use any subqueries and returns the same results.

3. **(15 Points)** Write a query which selects data from two common table expressions (CTEs). The first CTE should be called **ShippingAddresses** and should return all of the columns from the **Addresses** table where the address is a customer shipping addresses. The second CTE should be called **CustomersSingleAddress** and should return all of the columns from the **Customers** table where the customer uses the same address for both shipping and billing purposes. Using both CTEs, write a select statement which returns the following:

   - A calculated column with alias **Fullname** which returns the customer's last name and firstname with a comma and space between them, example: (Haley, Shane). All customers should be returned from CTE **CustomersSingleAddress** regardless of the **ShippingAddresses** CTE.
   - The **Line1** column
   - The **City** column
   - The **State** column

   Sort the result set in ascending sequence by **State**, ascending sequence by **City,** and ascending sequence by **Fullname**.

   CONSTRAINTS: Do not use any joins in the CTEs. Use subqueries in the CTEs if necessary to compare data from one table to data in another table. Joins may be used in the final SELECT statement which queries the two CTEs.

4. **(15 Points)** Write a query which selects data from two common table expressions (CTEs). The first CTE should be called **CategoryTotals** and should return the following:

   - The **CategoryID** column from the **Categories** table.
   - The **CategoryName** column from the **Categories** table.
   - An aggregate column with alias **TotalCatOrders** which sums the **Quantity** from the **OrderItems** table

   The second CTE should be called **ProductTotals** and should return the following

- The **CategoryID** column from the **Products** table.
- The **ProductName** column from the **Products** table.
- An aggregate column with alias **TotalProdOrders** which sums the **Quantity** from the **OrderItems** table

Using both CTEs, write a select statement which returns the following:

- The **CategoryName** column
- The **TotalCatOrders** column
- The **ProductName** column
- The **TotalProdOrders** column

Sort the result set in descending sequence by **TotalCatOrders** and descending sequence by **TotalProdOrders**

CONSTRAINTS: None. With the exception of the usage of the CTEs, no subqueries are required to complete this exercise (everything can be accomplished using joins, even within the CTEs).