CSCI 5020 Assignment 2

Part 1

Chapter 3 Exercises 1, 3, and 6 on page 123.

Chapter 4 Exercises 2, 4, 6, and 7 on page 157 - 158.

Chapter 5 Exercises 2, 4, 6, 7, and 8 on page 180 - 181.

The solutions to the chapter exercises in the book are provided through the student download and posted in D2L as well. For learning purpose, make sure you complete each exercise before checking the solution.

Part 2

Chapter 3 How to retrieve data from a single table

Enter and run your own SELECT statements

In these exercises, you'll enter and run your own SELECT statements against MyGuitarShop database.

1. Write a SELECT statement that returns four columns from theProducts table: ProductCode, ProductName, ListPrice, and DiscountPercent. Then, run this statement to make sure it works correctly.

   Add an ORDER BY clause to this statement that sorts the result set by list price in descending sequence. Then, run this statement again to make sure it works correctly. This is a good way to build and test a statement, one clause at a time.

2. Write a SELECT statement that returns these column names and data from the Products table:

ProductName	The ProductName column
ListPrice	The ListPricecolumn
DateAdded	The DateAdded column

   Return only the rows with a list price that's greater than 500 and less than 2000.

   Sort the result set in descending sequence by the DateAdded column.

Work with nulls and test expressions

3. Write a SELECT statement that returns these columns from the Orders table:

OrderID	The OrderID column
OrderDate	The OrderDate column
ShipDate	The ShipDate column

   Return only the rows where the ShipDate column contains a null value.

4. Write a SELECT statement without a FROM clause that creates a row with these columns:

Price	100 (dollars)
TaxRate	.07 (7 percent)
TaxAmount	The price multiplied by the tax
Total	The price plus tax

   To calculate the fourth column, add the expressions you used for the first and third columns.

Chapter 4 How to retrieve data from two or more tables

1. Write a SELECT statement that joins the Customers table to the Addresses table and returns these columns: FirstName, LastName, Line1, City, State, ZipCode.

   Return one row for each customer, but only return addresses that are the shipping address for a customer.

2. Write a SELECT statement that joins the Customers, Orders, OrderItems, and Products tables. This statement should return these columns:LastName, FirstName, OrderDate, ProductName, ItemPrice, DiscountAmount, and Quantity.

   Use aliases for the tables.

   Sort the final result set by LastName, OrderDate, and ProductName.

3. Write a SELECT statement that returns theProductName and ListPrice columns from the Products table.

   Return one row for each product that has the same list price as another product.*Hint: Use a self-join to check that the ProductID columns aren't equal but the ListPrice columnis equal.*

   Sort the result set by ProductName.

4. Write a SELECT statement that returns these two columns:

CategoryName	The CategoryName column from the Categories table
ProductID	The ProductID column from the Products table

   Return one row for each category that has never been used. *Hint: Use an outer join and only return rows where the ProductID column contains a null value.*

5. Use the UNION operator to generate a result set consisting of three columns from the Orders table:

ShipStatus	A calculated column that contains a value of SHIPPED or NOT SHIPPED
OrderID	The OrderID column
OrderDate	The OrderDate column

   If the order has a value in the ShipDate column, the ShipStatuscolumn should contain a value of SHIPPED. Otherwise, it should contain a value of NOT SHIPPED.

   Sort the final result set by OrderDate.

Chapter 5 How to code summary queries

1. Write a SELECT statement that returns these columns:

   The count of the number of orders in the Orders table

   The sum of the TaxAmountcolumns in the Orders table

2. Write a SELECT statement that returns one row for each category that has products with these columns:

   The CategoryName column from the Categories table

   The count of the products in the Products table

   The list price of the most expensive product in the Products table

   Sort the result set so the category with the most products appears first.

3. Write a SELECT statement that returns one row for each customer that has orders with these columns:

   The EmailAddress column from the Customers table

   The sum of the item price in the OrderItems table multiplied by the quantiy in the OrderItems table

   The sum of the discount amount column in the OrderItems table multiplied by the quantiy in the OrderItems table

   Sort the result set in descending sequence by the item price total for each customer.

4. Write a SELECT statement that returns one row for each customer that has orders with these columns:

   The EmailAddresscolumn from the Customers table

   A count of the number of orders

   The total amount for each order (*Hint: First, subtract the discount amount from the price. Then, multiply by the quantity.*)

   Return only those rows where the customer has more than 1 order.

   Sort the result set in descending sequence by the sum of the line item amounts.

5. Write a SELECT statement that answers this question: What is the total amount ordered for each product? Return these columns:

   The product name from the Products table

   The total amount for each product in the OrderItemstable (*Hint: You can calculate the total amount by subtracting the discount amount from the item price and then multiplying it by the quantity*)

   Use the WITH ROLLUP operator to include a row that gives the grand total.

Submit each statement in separate script files (*.sql) to the Dropbox, for example, ch4ex3.sql.