Project 1

I. My Guitar Shop Database

In part I, you will use SQL Server Management Studio to create the MyGuitarShop database, to review the tables in the MyGuitarShop database, and to enter SQL statements and run them against this database.

A Database Setup [2 pts.]

1. (1) Download CreateMyGuitarShop.sql from Project 1 directory on Blackboard and open it in SQL server management studio. Execute the entire script and show the message in the Message tab, indicating the script is executed successfully. A complete screenshot of execution result is required.

   (2) Navigate through the database objects and view the column definitions for each table. Open a new Query Editor window. Show details in Customers table and Orders table using SELECT statement. Full screenshots of execution results are required.

B An introduction to SQL [6 pts.]

1. [3] Write a SELECT statement that returns one column from the Customers table named FullName that joins the FirstName and LastName columns.
   Format this column with the first name, a comma, a space, and the last name like this:
   John, Doe
   Add an ORDER BY clause to this statement that sorts the result set by first name in ascending sequence. Return only the contacts whose last name begins with a letter from A to C.

2. [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 does not contain a null value.

C The essential SQL skills [44 pts.]

1. [4] 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.

2. [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.)

3. [4] Write a SELECT statement that returns one row for each customer that has orders with these columns:
   a) The EmailAddress column from the Customers table
   b) The sum of the item price in the OrderItems table multiplied by the quantity in the OrderItems table
   c) The sum of the discount amount column in the OrderItems table multiplied by the quantity in the OrderItems table.
   Sort the result set in ascending sequence by the item price total for each customer.

4. [4] Write a SELECT statement that returns one row for each customer that has orders with these columns:
   a) The EmailAdress column from the Customers table
   b) A count of the number of orders
   c) 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 items have a more than 600 ItemPrice value.
   Sort the result set in descending sequence by the sum of the line item amounts.

5. [4] (1) Write a SELECT statement that returns three columns: EmailAddress, OrderID, and the order total for each customer. To do this, you can group the result set by the EmailAddress and OrderID columns. In addition, you must calculate the order total from the columns in the OrderItems table. (2) Write a second SELECT statement that uses the first SELECT statement in its FROM clause. The main query should return two columns: the customer's email address and the least order for that customer. To do this, you can group the result set by the EmailAddress column.

6. [4] Use a correlated subquery to return one row per customer, representing the customer's newest order (the one with the latest date). Each row should include these three columns: EmailAddress, OrderID, and OrderDate.

7. [4] Write a SELECT statement that returns these columns from the Products table:
   a) The ListPrice column
   b) A column that uses the CAST function to return the ListPrice column with 1 digit to the right

of the decimal point
c) A column that uses the **CONVERT** function to return the ListPrice column as an integer
d) A column that uses the **CAST** function to return the ListPrice column as an integer.

8. [4] Write a **SELECT** statement that returns these columns from the Orders table:
   a) The CardNumber column
   b) The length of the CardNumber column
   c) The last four digits of the CardNumber column
   d) A column that displays the last four digits of the CardNumber column in this format: XXXX-XXXX-XXXX-1234. In other words, use Xs for the first 12 digits of the card number and actual numbers for the last four digits of the number.

9. [4] Write a **SELECT** statement that returns these columns from the Orders table:
   a) The OrderID column
   b) The OrderDate column
   c) A column named ApproxShipDate that's calculated by adding 2 days to the OrderDate column
   d) The ShipDate column
   e) A column named DaysToShip that shows the number of days between the order date and the ship date

   When you have this working, add a **WHERE** clause that retrieves just the orders for May 2016.

For question 10-11: To test whether a table has been modified correctly as you do these questions, please write and run an appropriate SELECT statement and take full screenshots of the verification.

10. [4] Write an **INSERT** statement that adds this row to the Customers table:

EmailAddress:	ellie@Krieger.com
Password:	(empty string)
FirstName:	Ellie
LastName:	Krieger

    Use a column list for this statement.

11. [4] Write an **UPDATE** statement that modifies the Customers table. Change the password column to "secret" for the customer with an email address of erinv@gmail.com.

D Advanced SQL skills (views/stored procedures/functions/scripts) [24 pts.]

Open the script named CreateMyGuitarShop.sql and run this script. That should restore the data that's in the database. Then complete questions in Section D. Screenshot of execution is required for each question. Please also use SELECT statement to verify results of your codes (Screenshots of verification are required).

1. [4] Create a view named OrderItemProducts that returns columns from the Orders, OrderItems, and Products tables.
   a) This view should return these columns from the Orders table: OrderID, OrderDate, TaxAmount, and ShipDate.
   b) This view should return these columns from the OrderItems table: ItemPrice, DiscountAmount, FinalPrice (the discount amount subtracted from the item price), Quantity, and ItemTotal (the calculated total for the item).
   c) This view should return the ProductName column from the Products table.

2. [5] Create a view named Top5BestSelling that uses the view you created in Section D Question 1. This view should return some summary information about five best selling products. Each row should include these columns: ProductName, OrderTotal (the total sales for the product) and OrderCount (the number of times the product has been ordered).

3. [5] Write a script that creates and calls a stored procedure named spUpdateProductDiscount that updates the DiscountPercent column in the Products table. This procedure should have one parameter for the product ID and another for the discount percent.
   If the value for the DiscountPercent column is a negative number, the stored procedure should raise an error that indicates that the value for this column must be a positive number.
   Code at least two EXEC statements that test this procedure.

4. [5] Write a script that calculates the common factors between 10 and 20. To find a common factor, you can use the modulo operator (%) to check whether a number can be evenly divided into both numbers. Then, this script should print lines that display the common factors like this:
   Common factors of 10 and 20
   1
   2
   5

5. [5] (1) Write a script that creates and calls a function named fnDiscountPrice that calculates the discount price of an item in the OrderItems table (discount amount subtracted from item price). To do that, this function should accept one parameter for the item ID, and it should return the value of the discount price for that item. (2) Write a script that creates and calls a function named fnItemTotal that calculates the total amount of an item in the OrderItems table (discount price multiplied by quantity). To do that, this function should accept one parameter for the item ID, it should use the DiscountPrice function that you created in (1), and it should return the value of the total for that item.

II. Database Design [20 pts.]

Create a sample database design for a library and draw one database model for it. The library wants to keep track of books (regular books/electronic/audio); book authors; book genres; book location (area/shelves); customer information; books that the customers borrowed, when they are due, employee that checked them out; employee information; and employee working schedules.

1. [3] Design the database that makes sense for the problem and select fields that make the most sense. A complete screenshot of your final design model is required.

2. [10] Determine the tables, columns, primary keys, nullabilities and show relationships between tables (one -one/one-many/many-many).

3. [5] Normalize your design into 3rd Normal Form. Please use MS Visio or any similar database design tool.

4. [2] Explain your design, including relationship between tables.

In your typed report (pdf or docx only), please include the following items:

1. Complete SQL source codes with your comments.
2. Full screenshots of the requested actions or each question including the total number of rows in result set and your name, which are listed above with your comments on them.
3. Your remarks on the project [4 pts.]. The remarks should be specific thoughts and references you learned that are related with knowledge points.

Submit your report on Blackboard.