Outcomes addressed in this activity:

Unit Outcomes
1. Use SQL to Insert new rows into a table
2. Use SQL to update columns in a row or in multiple rows.
3. USE SQL to delete 1 or more rows in a table.
4. Determine when any DML statement may cause a problem with related rows in other tables.

Course Outcomes
IT350-5: Create SQL insert, update and delete statements to maintain data within a database.

[CLA]

Directions:
Using the Adventure Works Database

Create a separate screenshot for each question and place each under the question in a Word document. Submit to the dropbox for the Unit Assignment.

Answer all of the following questions using SQL select statements executed in SQL Server Management Studio.

Review The Adventure Works data base schema diagram found in doc sharing.

Remember that you can expand the tables in the Object Explorer to see the list of table names and then expand the table to see the list of column names.

Run the following code to create required tables.

1. Create a screenshot of the new tables Object Explorer. Only some of the questions below will use the new tables.

------------------------------------------------------------------------------
USE AdventureWorks2012;
GO
IF EXISTS (SELECT * FROM sys.objects
    WHERE object_id = OBJECT_ID(N'[dbo].[demoProduct]')
    AND type in (N'U'))
------------------------------------------------------------------------------
DROP TABLE [dbo].[demoProduct]
GO

CREATE TABLE [dbo].[demoProduct](
    [ProductID] [INT] NOT NULL PRIMARY KEY,
    [Name] [dbo].[Name] NOT NULL,
    [Color] [NVARCHAR](15) NULL,
    [StandardCost] [MONEY] NOT NULL,
    [ListPrice] [MONEY] NOT NULL,
    [Size] [NVARCHAR](5) NULL,
    [Weight] [DECIMAL](8, 2) NULL,
);  
IF EXISTS (SELECT * FROM sys.objects
    WHERE object_id = OBJECT_ID(N'[dbo].[demoSalesOrderHeader]')
    AND type in (N'U'))
DROP TABLE [dbo].[demoSalesOrderHeader]
GO

CREATE TABLE [dbo].[demoSalesOrderHeader](
    [SalesOrderID] [INT] NOT NULL PRIMARY KEY,
    [SalesID] [INT] NOT NULL IDENTITY,
    [OrderDate] [DATETIME] NOT NULL,
    [CustomerID] [INT] NOT NULL,
    [SubTotal] [MONEY] NOT NULL,
    [TaxAmt] [MONEY] NOT NULL,
    [Freight] [MONEY] NOT NULL,
    [DateEntered] [DATETIME],
    [SalesNumber] [INT] NOT NULL,
    [TotalDue] AS (ISNULL(([SubTotal]+[TaxAmt])+[Freight],(0))),
    [RV] ROWVERSION NOT NULL);

ALTER TABLE [dbo].[demoSalesOrderHeader] ADD CONSTRAINT
    [DF_demoSalesOrderHeader_DateEntered]
DEFAULT (getdate()) FOR [DateEntered];

IF EXISTS (SELECT * FROM sys.objects
    WHERE object_id = OBJECT_ID(N'[dbo].[demoSalesSequence]'))
DROP SEQUENCE [dbo].[demoSalesSequence]
GO

CREATE SEQUENCE demoSalesSequence
AS INT
START WITH 1
INCREMENT BY 1;

GO

IF EXISTS (SELECT * FROM sys.objects
    WHERE object_id = OBJECT_ID(N'[dbo].[demoAddress]')
    AND type in (N'U'))
DROP TABLE [dbo].[demoAddress]

GO

CREATE TABLE [dbo].[demoAddress](
    [AddressID] [INT] NOT NULL IDENTITY PRIMARY KEY,
    [AddressLine1] [NVARCHAR](60) NOT NULL,
    [AddressLine2] [NVARCHAR](60) NULL,
    [City] [NVARCHAR](30) NOT NULL,
    [PostalCode] [NVARCHAR](15) NOT NULL
);

USE AdventureWorks2012;
GO

IF OBJECT_ID('demoCustomer') IS NOT NULL BEGIN
    DROP TABLE demoCustomer;
END;
CREATE TABLE demoCustomer(CustomerID INT NOT NULL PRIMARY KEY,
    FirstName NVARCHAR(50) NOT NULL, MiddleName NVARCHAR(50) NOT NULL, LastName
    NVARCHAR(50) NOT NULL);

USE AdventureWorks2012;
GO

IF OBJECT_ID('dbo.Demo') IS NOT NULL BEGIN
    DROP TABLE dbo.Demo;
END;
GO
CREATE TABLE dbo.Demo(ID INT PRIMARY KEY, Name VARCHAR(25));

---------------------------------------------------------------------------------------------

1. Write a SELECT statement to retrieve data from the Production.Product table. Use these values to insert
   five rows into the dbo.demoProduct table using literal values. Write five individual INSERT statements.
2. Insert five more rows into the dbo.demoProduct table. This time write one INSERT statement.
3. Write an INSERT statement that inserts all the rows into the dbo.demoSalesOrderHeader table from
   the Sales.SalesOrderHeader table. Hint: Pay close attention to the properties of the columns in the
   dbo.demoSalesOrderHeader table.
4. Write a SELECT INTO statement that creates a table, dbo.tempCustomerSales, showing every CustomerID from the Sales.Customer along with a count of the orders placed and the total amount due for each customer.

5. Write an INSERT statement that inserts all the products into the dbo.demoProduct table from the Production.Product table that have not already been inserted. Don’t specify literal ProductID values in the statement.

6. Write an INSERT statement that inserts all the addresses into the dbo.demoAddress table from the Person(Address table. Before running the INSERT statement, type in and run the following command so that you can insert values into the AddressID column: SET IDENTITY_INSERT dbo.demoAddress ON;

7. Write a query that deletes the rows from the dbo.demoCustomer table where the LastName values begin with the letter S.

8. Delete the rows from the dbo.demoCustomer table if the customer has not placed an order or if the sum of the TotalDue from the dbo.demoSalesOrderHeader table for the customer is less than $1,000.

9. Delete the rows from the dbo.demoProduct table that have never been ordered. Write an UPDATE statement that changes all NULL values of the AddressLine2 column in the dbo.demoAddress table to N/A.

10. Write an UPDATE statement that increases the ListPrice of every product in the dbo.demoProduct table by 10 percent.

11. Write an UPDATE statement that corrects the UnitPrice and LineTotal of each row of the dbo.demoSalesOrderDetail table by joining the table on the dbo.demoProduct table.

12. Write an UPDATE statement that updates the SubTotal column of each row of the dbo.demoSalesOrderHeader table with the sum of the LineTotal column of the dbo.demoSalesOrderDetail table.

13. Write a transaction that includes two INSERT statements to add two rows to the dbo.Demo table.

14. Write a transaction that includes two INSERT statements to add two more rows to the dbo.Demo table. Attempt to insert a letter instead of a number into the ID column in one of the statements. Select the data from the dbo.Demo table to see which rows made it to the table.

Review the grading rubric below before beginning this activity.

Assignment grading rubric = 40 points

<table>
<thead>
<tr>
<th>Assignment Criteria</th>
<th>Points</th>
<th>Points</th>
</tr>
</thead>
</table>
1. Write a SELECT statement to retrieve data from the Production.Product table. Use these values to insert five rows into the dbo.demoProduct table using literal values. Write five individual INSERT statements.

2. Insert five more rows into the dbo.demoProduct table. This time write one INSERT statement.

3. Write an INSERT statement that inserts all the rows into the dbo.demoSalesOrderHeader table from the Sales.SalesOrderHeader table. Hint: Pay close attention to the properties of the columns in the dbo.demoSalesOrderHeader table.

4. Write a SELECT INTO statement that creates a table, dbo.tempCustomerSales, showing every CustomerID from the Sales.Customer along with a count of the orders placed and the total amount due for each customer.

5. Write an INSERT statement that inserts all the products into the dbo.demoProduct table from the Production.Product table that have not already been inserted. Don’t specify literal ProductID values in the
<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>6.</td>
<td>Write an INSERT statement that inserts all the addresses into the dbo.demoAddress table from the Person.Address table. Before running the INSERT statement, type in and run the following command so that you can insert values into the AddressID column: SET IDENTITY_INSERT dbo.demoAddress ON;</td>
</tr>
<tr>
<td></td>
<td>0 - 4</td>
</tr>
<tr>
<td>7.</td>
<td>Write a query that deletes the rows from the dbo.demoCustomer table where the LastName values begin with the letter S.</td>
</tr>
<tr>
<td></td>
<td>0 - 4</td>
</tr>
<tr>
<td>8.</td>
<td>Delete the rows from the dbo.demoCustomer table if the customer has not placed an order or if the sum of the TotalDue from the dbo.demoSalesOrderHeader table for the customer is less than $1,000.</td>
</tr>
<tr>
<td></td>
<td>0 - 4</td>
</tr>
<tr>
<td>9.</td>
<td>Delete the rows from the dbo.demoProduct table that have never been ordered. Write an UPDATE statement that changes all NULL values of the AddressLine2 column in the dbo.demoAddress table to N/A.</td>
</tr>
<tr>
<td></td>
<td>0 - 4</td>
</tr>
<tr>
<td>Question</td>
<td>Points</td>
</tr>
<tr>
<td>-------------------------------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>10. Write an <code>UPDATE</code> statement that increases the <code>ListPrice</code> of every product in the <code>dbo.demoProduct</code> table by 10 percent.</td>
<td>0-4</td>
</tr>
<tr>
<td>11. Write an <code>UPDATE</code> statement that corrects the <code>UnitPrice</code> and <code>LineTotal</code> of each row of the <code>dbo.demoSalesOrderDetail</code> table by joining the table on the <code>dbo.demoProduct</code> table.</td>
<td>0-2</td>
</tr>
<tr>
<td>12. Write an <code>UPDATE</code> statement that updates the <code>SubTotal</code> column of each row of the <code>dbo.demoSalesOrderHeader</code> table with the sum of the <code>LineTotal</code> column of the <code>dbo.demoSalesOrderDetail</code> table.</td>
<td>0-2</td>
</tr>
<tr>
<td>13. Write a transaction that includes two <code>INSERT</code> statements to add two rows to the <code>dbo.Demo</code> table.</td>
<td>0-2</td>
</tr>
</tbody>
</table>

Column Total | 0-40 | 40 |