Lab 4

# Use the following script to create database and work on assignments

*Create database AdventureWorksLT*

*Go*

*USE AdventureWorksLT;*

*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, );*

*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], [TotalDue] AS (ISNULL(([SubTotal]+[TaxAmt])+[Freight],(0))), [RV] ROWVERSION NOT NULL);*

*GO*

*ALTER TABLE [dbo].[demoSalesOrderHeader] ADD CONSTRAINT [DF\_demoSalesOrderHeader\_DateEntered] DEFAULT (GETDATE()) FOR [DateEntered]; 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, [StateProvince] [dbo].[Name] NOT NULL, [CountryRegion] [dbo].[Name] NOT NULL, [PostalCode] [NVARCHAR](15) NOT NULL );*

1. **Write a SELECT statement to retrieve data from the SalesLT.Product table. Use these values to insert five rows into the dbo.demoProduct table using literal values. Write five individual INSERTstatements. The rows you choose to insert may vary.**
   1. SELECT ProductID, Name, Color, StandardCost, ListPrice, Size, Weight

FROM SalesLT.Product;

INSERT INTO dbo.demoProduct(ProductID, Name, Color, StandardCost, ListPrice, Size, Weight) VALUES (680,'HL Road Frame - Black,58', 'Black', 1059.31,1431.50,'58',1016.04);

INSERT INTO dbo.demoProduct(ProductID, Name, Color, StandardCost, ListPrice, Size, Weight) VALUES (706,'HL Road Frame - Red, 58','Red',1059.31, 1431.50,'58',1016.04);

INSERT INTO dbo.demoProduct(ProductID, Name, Color, StandardCost, ListPrice, Size, Weight) VALUES (707,'Sport-100 Helmet, Red','Red',13.0863,34.99,NULL,NULL);

INSERT INTO dbo.demoProduct(ProductID, Name, Color, StandardCost, ListPrice, Size, Weight) VALUES (708,'Sport-100 Helmet, Black','Black',13.0863,34.99,NULL,NULL);

INSERT INTO dbo.demoProduct(ProductID, Name, Color, StandardCost, ListPrice, Size, Weight) VALUES (709,'Mountain Bike Socks, M','White',3.3963,9.50,'M',NULL);

1. **Insert five more rows into the dbo.demoProduct table. This time write one INSERT statement. The rows you choose to insert may vary.**
   1. INSERT INTO dbo.demoProduct(ProductID, Name, Color,StandardCost, ListPrice, Size, Weight) VALUES

(711,'Sport-100 Helmet, Blue', 'Blue', 13.0863, 34.99, NULL, NULL),

(712, 'AWC Logo Cap', 'Multi', 6.9223, 8.99, NULL, NULL),

(713,'Long-Sleeve Logo Jersey,S', 'Multi', 38.4923, 49.99, 'S' ,NULL),

(714, 'Long-Sleeve Logo Jersey,M', 'Multi', 38.4923, 49.99, 'M', NULL),

(715, 'Long-Sleeve Logo Jersey,L', 'Multi', 38.4923, 49.99, 'L' ,NULL);

1. **Write an INSERT statement that inserts all the rows into the dbo.demoSalesOrderHeader table from the SalesLT.SalesOrderHeader table. Hint: Pay close attention to the properties of the columns in the dbo.demoSalesOrderHeader table. Don’t insert a value into the SalesID, DateEntered, and RV columns.**
   1. INSERT INTO dbo.demoSalesOrderHeader(SalesOrderID, OrderDate, CustomerID,

SubTotal, TaxAmt, Freight) SELECT SalesOrderID, OrderDate, CustomerID,

SubTotal, TaxAmt, Freight FROM SalesLT.SalesOrderHeader;

1. **Write a SELECT INTO statement that creates a table, dbo.tempCustomerSales, showing every CustomerID from the SalesLT.Customer along with a count of the orders placed and the total amount due for each customer.**
   1. SELECT COUNT(ISNULL(SalesOrderID,0)) AS CountOfORders, c.CustomerID, SUM(TotalDue) AS TotalDue INTO dbo.tempCustomerSales FROM SalesLT.Customer AS c LEFT JOIN SalesLT.SalesOrderHeader AS soh ON c.CustomerID = soh.CustomerID GROUP BY c.CustomerID**;**
2. **Write an INSERT statement that inserts all the products into the dbo.demoProduct table from the SalesLT.Product table that have not already been inserted. Do not specify literal ProductID values in the statement.**
   1. INSERT INTO dbo.demoProduct (ProductID, Name, Color, StandardCost, ListPrice, Size, Weight) SELECT p.ProductID, p.Name, p.Color, p.StandardCost, p.ListPrice, p.Size, p.Weight FROM SalesLT.Product AS p LEFT OUTER JOIN dbo.demoProduct AS dp ON p.ProductID = dp.ProductID WHERE dp.ProductID IS NULL;

INSERT INTO dbo.demoProduct (ProductID, Name, Color, StandardCost, ListPrice, Size, Weight) SELECT ProductID, Name, Color, StandardCost, ListPrice, Size, Weight FROM SalesLT.Product WHERE ProductID NOT IN (SELECT ProductID FROM dbo.demoProduct WHERE ProductID IS NOT NULL);

1. **Write an INSERT statement that inserts all the addresses into the dbo.demoAddress table from the SalesLT.Address table. Before running the INSERT statement, type and run the command so that you can insert values into the AddressID column.**
   1. SET IDENTITY\_INSERT dbo.demoAddress ON;

INSERT INTO dbo.demoAddress(AddressID,AddressLine1,AddressLine2,City, StateProvince, CountryRegion, PostalCode)

SELECT AddressID, AddressLine1, AddressLine2,City, StateProvince, CountryRegion, PostalCode FROM SalesLT.Address;

SET IDENTITY\_INSERT dbo.demoAddress OFF;

1. **Write a query that deletes the rows from the dbo.demoCustomer table where the LastName values begin with the letter S.**
   1. DELETE FROM dbo.demoCustomer WHERE LastName LIKE 'S%'
2. **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**
   1. DELETE FROM dbo.demoCustomer WHERE CustomerID IN (

SELECT C.CustomerID FROM dbo.demoCustomer AS C

LEFT OUTER JOIN dbo.demoSalesOrderHeader AS SOH

ON C.CustomerID = SOH.CustomerID

GROUP BY c.CustomerID

HAVING SUM(ISNULL(TotalDue,0)) < 1000);

WITH Sales AS (

SELECT C.CustomerID

FROM dbo.demoCustomer AS C

LEFT OUTER JOIN dbo.demoSalesOrderHeader AS SOH

ON C.CustomerID = SOH.CustomerID

GROUP BY c.CustomerID

HAVING SUM(ISNULL(TotalDue,0)) < 1000)

DELETE C FROM dbo.demoCustomer AS C

INNER JOIN Sales ON C.CustomerID = Sales.CustomerID;

1. **Delete the rows from the dbo.demoProduct table that have never been ordered.**
   1. DELETE P FROM dbo.demoProduct AS P

LEFT OUTER JOIN dbo.demoSalesOrderDetail AS SOD ON P.ProductID = SOD.ProductID WHERE SOD.ProductID IS NULL;

DELETE FROM dbo.demoProduct WHERE ProductID NOT IN

(SELECT ProductID FROM dbo.demoSalesOrderDetail

WHERE ProductID IS NOT NULL);

1. **Write an UPDATE statement that changes all NULL values of the AddressLine2 column in the dbo.demoAddress table to N/A.**
   1. UPDATE dbo.demoAddress SET AddressLine2 = 'N/A' WHERE AddressLine2 IS NULL;
2. **Write an UPDATE statement that increases the ListPrice of every product in the dbo.demoProducttable by 10 percent.**
   1. UPDATE dbo.demoProduct SET ListPrice \*= 1.1;
3. **Write an UPDATE statement that corrects the UnitPrice with the ListPrice of each row of the dbo.demoSalesOrderDetail table by joining the table on the dbo.demoProduct table.**
   1. UPDATE SOD SET UnitPrice = P.ListPrice

FROM SalesLT.SalesOrderDetail AS SOD

INNER JOIN dbo.demoProduct AS P ON SOD.ProductID = P.ProductID;

1. **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.demoSalesOrderDemo table.**
   1. WITH SOD AS(

SELECT SUM(LineTotal) AS TotalSum, SalesOrderID

FROM dbo.demoSalesOrderDetail GROUP BY SalesOrderID)

UPDATE SOH Set SubTotal = TotalSum

FROM dbo.demoSalesOrderHeader AS SOH

INNER JOIN SOD ON SOH.SalesOrderID = SOD.SalesOrderID;

# Use following script to create a table

CREATE TABLE dbo.Demo(ID INT PRIMARY KEY, Name VARCHAR(25));

1. **Write a transaction that includes two insert statements to add two rows to the dbo.Demo table.**
   1. BEGIN TRAN

INSERT INTO dbo.Demo(ID,Name) VALUES (1,'Test1');

INSERT INTO dbo.Demo(ID,Name) VALUES(2,'Test2');

COMMIT TRAN;

1. **Write a transaction that includes two insert statements to add two more rows to the dbo.Demotable. 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 into the table.**
   1. BEGIN TRAN

INSERT INTO dbo.Demo(ID,Name) VALUES(3,'Test3');

INSERT INTO dbo.Demo(ID,Name) VALUES('a','Test4');

COMMIT TRAN;

GO

SELECT ID,Name FROM dbo.Demo;