Lab 4

Connection values:

Server Type = Database Engine
Server Name = boyce.coe.neu.edu
Authentication = SQL Server Authentication
Login = INFO6210
Password = NEUHusky!

-- Create a database and some tables in the new database.

```sql
CREATE DATABASE "Use your name for the database name";
GO


USE "Use your name for the database name";


CREATE TABLE dbo.Customers
    (
    CustomerID varchar(5) NOT NULL PRIMARY KEY ,
    Name varchar(40) NOT NULL
    );

CREATE TABLE dbo.Orders
    (
    OrderID int IDENTITY NOT NULL PRIMARY KEY,
    CustomerID varchar(5) NOT NULL
        REFERENCES Customers(CustomerID),
    OrderDate datetime DEFAULT Current_Timestamp
    );

CREATE TABLE dbo.Products
    (
    ProductID int IDENTITY NOT NULL PRIMARY KEY,
    Name varchar(40) NOT NULL,
    UnitPrice money NOT NULL
    );

CREATE TABLE dbo.OrderItems
    (
    OrderID int NOT NULL
        REFERENCES dbo.Orders(OrderID),
    ProductID int NOT NULL
        REFERENCES dbo.Products(ProductID),
    UnitPrice money NOT NULL,
    Quantity int NOT NULL
        CONSTRAINT PKOrderItem PRIMARY KEY CLUSTERED
            (OrderID, ProductID)
    );
```

```sql
-- Put some data in the database

-- INSERT sample records

INSERT dbo.Customers
    VALUES ('ABC', 'Bob''s Pretty Good Garage');

INSERT dbo.Orders (CustomerID)
    VALUES ('ABC');

INSERT dbo.Products
    VALUES ('Widget', 5.55),
           ('Thingamajig', 8.88)

INSERT dbo.OrderItems
    VALUES (1, 1, 5.55, 3);
```

```
/*
    If you create a table without specifying constraints,
    You can use ALTER TABLE to add a constraint
*/


-- Create a table without specifying constraints.

CREATE TABLE TBL3 (pk3 int);


-- Add the NOT NULL constraint

ALTER TABLE tbl3 ALTER COLUMN pk3 int not null;


-- Add the Primary Key constraint.

ALTER TABLE tbl3 ADD CONSTRAINT key3 PRIMARY KEY (pk3);


-- Add the Foreign Key constraint.

-- Create the parent table first.

CREATE TABLE TBL1 (pk1 int PRIMARY KEY);


ALTER TABLE tbl3 ADD CONSTRAINT R3 FOREIGN KEY (pk3)
      REFERENCES tbl1(pk1)


-- Must DROP the child table before dropping the parent table.

DROP TABLE TBL3;
DROP TABLE TBL1;
```

-- A simple example of WHILE Statement

```
/*
    SQL variables start with either @ or @@.
    @ indicates a local variable, which is in effect in the current
    scope.
    @@ indicates a global variable, which is in effect for all
    scopes of the current connection.
*/

/*
    We need to make sure that we have a way to stop the WHILE loop.
    Otherwise, we'll have an endless WHILE loop which may run forever.
    We use the variable @counter to determine when to terminate
    the WHILE loop.
    We use CAST to convert an integer to character(s) so that we
    can concatenate the integer with other characters.
*/

DECLARE @counter INT
SET @counter = 0
WHILE @counter <> 5
    BEGIN
        SET @counter = @counter + 1
        PRINT 'The counter : ' + CAST(@counter AS CHAR)
    END
```

-- Use a Nested Loop to populate your table.

-- Create a test table.

```sql
CREATE TABLE PART (Part_Id int, Category_Id int,
    Description varchar(50));
```

-- The statements highlighted in yellow must be executed together

-- Declare SQL variables.

```sql
    DECLARE @Part_Id int;
    DECLARE @Category_Id int;
    DECLARE @Desc varchar(50);
```

-- Initilize SQL variables.

```sql
    SET @Part_Id = 0;
    SET @Category_Id = 0;
```

-- Populate the test table.

```sql
    WHILE @Part_Id < 10
    BEGIN
      SET @Part_Id = @Part_Id + 1;
      WHILE @Category_Id < 3
      BEGIN
        SET @Category_Id = @Category_Id + 1;
        SET @Desc = 'Part_Id is ' + cast(@Part_Id as char(1)) +
                    ' Category_Id ' + cast(@Category_Id as char(1));
        INSERT INTO PART VALUES (@Part_Id,
                                 @Category_Id,
                                 @Desc );
      END;
      SET @Category_Id = 0;
    END;
```

-- Retrieve the test data.

```sql
    SELECT * FROM PART;
```

-- Drop the test table.

```sql
    DROP TABLE PART;
```

-- SQL View

```sql
USE AdventureWorks2008R2;

-- CREATE VIEW Command

-- You need to execute these statements on your own computer

CREATE VIEW vwEmployeeContactInfo
    AS
    SELECT e.[BusinessEntityID] as [ContactID], FirstName,
        MiddleName, LastName, JobTitle
    FROM Person.Person c
    INNER JOIN HumanResources.Employee e
        ON c.BusinessEntityID = e.BusinessEntityID;

-- Select from the view

SELECT *
FROM vwEmployeeContactInfo;

-- See the script that generated the view

EXEC sp_helptext vwEmployeeContactInfo;

-- Delete the view from the database

DROP VIEW vwEmployeeContactInfo;
```

```
/*
  Create a view to include the encryption and
  schemabinding options. Encryption protects the
  view query definition. Schemabinding means the
  definition of the database object(s) on which
  the view is defined can not be changed without
  first dropping the view.
*/

CREATE VIEW vwEmployeeContactInfo
      WITH ENCRYPTION, SCHEMABINDING
      AS
      SELECT e.[BusinessEntityID] as [ContactID], FirstName,
             MiddleName, LastName, JobTitle
      FROM Person.Person c
      INNER JOIN HumanResources.Employee e
             ON c.BusinessEntityID = e.BusinessEntityID;




/*
   Alter the view to remove schemabinding - must
   restate everything, including changes.
*/

ALTER VIEW vwEmployeeContactInfo
      WITH ENCRYPTION
      AS
      SELECT e.[BusinessEntityID] as [ContactID], FirstName,
             MiddleName, LastName, JobTitle
      FROM Person.Person c
      INNER JOIN HumanResources.Employee e
             ON c.BusinessEntityID = e.BusinessEntityID;
```
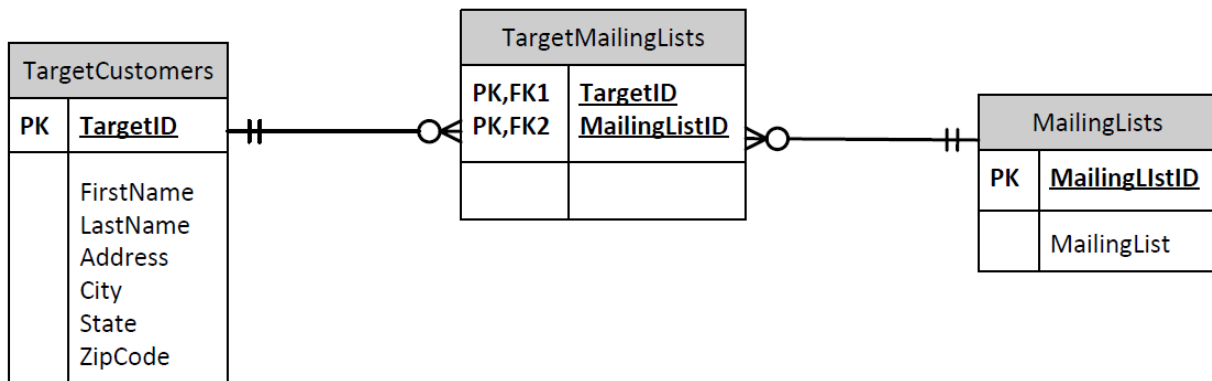
Lab 4 Questions

Part A (2 points)

Create 3 tables and the corresponding relationships to implement the ERD below in your own database.

TargetCustomers	
PK	**TargetID**
	FirstName
	LastName
	Address
	City
	State
	ZipCode

TargetMailingLists	
PK,FK1	**TargetID**
PK,FK2	**MailingListID**

MailingLists	
PK	**MailingLIstID**
	MailingList

Part B (2 point)

```
/* Using the content of AdventureWorks, write a query to retrieve
   all unique customers with all salespeople they have dealt with.
   If a customer has never worked with a salesperson, make the
   'Salesperson ID' column blank instead of displaying NULL.
   Sort the returned data by CustomerID in the descending order.
   The result should have the following format.

   Hint: Use the SalesOrderHeadrer table.
```

CustomerID	SalesPerson ID
30118	275, 277
30117	275, 277
30116	276
30115	289
30114	290
30113	282
30112	280, 284

```
*/
```

Part C (2 points)

```
/* Bill of Materials - Recursive */

/* The following code retrieves the components required for manufacturing
   the "Mountain-500 Black, 48" (Product 992). Use it as the starter code
   for calculating the material cost reduction if the component 815
   is manufactured internally at the level 1 instead of purchasing it
   for use at the level 0. Use the list price of a component as
   the material cost for the component. */


-- Starter code
WITH Parts(AssemblyID, ComponentID, PerAssemblyQty, EndDate, ComponentLevel) AS
(
    SELECT b.ProductAssemblyID, b.ComponentID, b.PerAssemblyQty,
           b.EndDate, 0 AS ComponentLevel
    FROM Production.BillOfMaterials AS b
    WHERE b.ProductAssemblyID = 992 AND b.EndDate IS NULL

    UNION ALL

    SELECT bom.ProductAssemblyID, bom.ComponentID, p.PerAssemblyQty,
           bom.EndDate, ComponentLevel + 1
    FROM Production.BillOfMaterials AS bom
    INNER JOIN Parts AS p
    ON bom.ProductAssemblyID = p.ComponentID AND bom.EndDate IS NULL
)
SELECT AssemblyID, ComponentID, Name, PerAssemblyQty, ComponentLevel
FROM Parts AS p
INNER JOIN Production.Product AS pr
ON p.ComponentID = pr.ProductID
ORDER BY ComponentLevel, AssemblyID, ComponentID;
```

Useful Links

Some great discussions about naming conventions

http://social.msdn.microsoft.com/Forums/sqlserver/en-US/fc76df37-f0ba-4cae-81eb-d73639254821/sql-server-naming-convention?forum=databasedesign

Create Database Using SQL Server Management Studio

http://www.youtube.com/watch?v=J59MGbQ_Shc

Create Tables Using SQL Server Management Studio

http://technet.microsoft.com/en-us/library/ms188264.aspx

Create Tables Using SQL Server Management Studio

http://www.youtube.com/watch?v=8l5Hw4kQE8o

Data Types

http://msdn.microsoft.com/en-us/library/ms187752.aspx

Create View

http://technet.microsoft.com/en-us/library/ms187956.aspx

How to Create a View

http://www.youtube.com/watch?v=MK_dWEcltWY