ADO.NET 4.5
With
LINQ AND Entity Framework
Lab Book

Document Revision History

Date	Revision No.	Author	Summary of Changes
22-June-2011	1	Ajit Jog	Content Creation
06-February-2015	2	Nachiket Inamdar	Addition of ADO.NET 4.5 features. Pending Approval.
14-Apr-2018	3	Shital Patil	Revamped as per new curriculum

Table of Contents

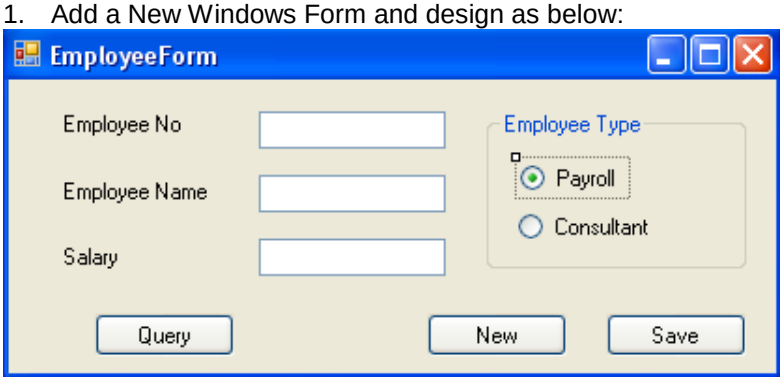
Lab 1. Using SqlCommand and SqlDataReader Classes

Description	In this Lab we will be retrieving employee information based on employee no and will be able to save new employee details
Goals	To Learn - • How to use sqlcommand object to execute a database stored procedure • How to use sqldatareader to read employee data • How to use sqlcommand to execute a DML statement
Time	180 Mins

The Table and SQL Server Stored Procedure used for this Lab:

```
create table employee
( empno int primary key,
  empname varchar(50) not null,
  empsal numeric(10,2) check(empsal >= 25000) ,
  emptype varchar(1) check(emptype in('C','P'))
)
go
create proc GetEmployeeById
(
        @eno int
)
as
        select * from employee where empno = @eno
go
```

1. Add a New Windows Form and design as below:



2. Define a connection object as form level member and write the following form load:

```
private void EmployeeForm_Load(object sender, EventArgs e)
{
        con = new SqlConnection(@"server=atrgsql\sql2005;database=labdemos;" +
        "user id=sqluser;password=sqluser");
        con.Open();
}
```


3. Add the following codes to the command buttons as below:

```
private void btnquery_Click(object sender, EventArgs e)
{
        try
        {
                SqlDataReader dreader=null;
                //The Procedure to execute
                SqlCommand cmd = new SqlCommand("GetEmployeeById", con);
                cmd.CommandType = CommandType.StoredProcedure;

                //define procedure parameter
                SqlParameter prm;
                prm = new SqlParameter();
                prm.SqlDbType = SqlDbType.Int;
                prm.Direction = ParameterDirection.Input;
                prm.ParameterName = "@eno";
                cmd.Parameters.Add(prm);

                //assign parameter value
                cmd.Parameters["@eno"].Value = int.Parse(txtempno.Text);

                //execute
                dreader = cmd.ExecuteReader();

                //if employee record found
                if (dreader.Read())
                {
                        txtempname.Text = dreader["empname"].ToString();
                        txtsalary.Text = dreader["empsal"].ToString();
                        if (dreader["emptype"].ToString() == "P")
                                rdpayroll.Checked = true;
                        else
                                rdconsultant.Checked = true;
                }
                else
                {
                        btnnew_Click(btnnew, e);
                        MessageBox.Show("No such employee");
                }
                dreader.Close();
        }
        catch (SqlException sqlex)
        {
                MessageBox.Show(sqlex.Message);
        }
}

private void btnsave_Click(object sender, EventArgs e)
{
        try
        {
```



```
                //The Insert DML to add employee record
                SqlCommand cmd = new SqlCommand ("insert into employee
                values(@eno,@enm,@esal,@etyp)", con);

                //The Parameters
                cmd.Parameters.Add("@eno", SqlDbType.Int);
                cmd.Parameters.Add("@enm", SqlDbType.VarChar, 50);
                cmd.Parameters.Add("@esal", SqlDbType.Decimal);
                cmd.Parameters.Add("@etyp", SqlDbType.VarChar, 1);

                //Assigning Values to parameters
                cmd.Parameters["@eno"].Value = txtempno.Text;
                cmd.Parameters["@enm"].Value = txtempname.Text;
                cmd.Parameters["@esal"].Value = txtsalary.Text;
                cmd.Parameters["@etyp"].Value = rdpayroll.Checked == true ? "P" :
                "C";

                //Execute Insert ....
                cmd.ExecuteNonQuery();
                MessageBox.Show("Employee Details Saved");
        }
        catch (SqlException sqlex)
        {
                MessageBox.Show(sqlex.Message);
        }
}

private void btnnew_Click(object sender, EventArgs e)
{
        txtempno.Text = "";
        txtempname.Text = "";
        txtsalary.Text = "";
        txtempno.Focus();
}
```

4. Run the Application
   a. Click New to clear the form if textboxes are already populated
   b. Fill the Employee Form and click Save to add new employee record.
   c. Click New to clear the form and type in an Employee No and click Query.

Assignment 1

To Do:
Add the delete button on the form. If the user clicks delete button ask for confirmation and if user confirms then delete the employee record.

Assignment 2

To Do:
Create a SQL Server stored procedure to add a new employee record. The procedure should accept all the employee details as parameter except empno. Procedure should auto generate next sequential empno and return that as well to the caller. [Hint: Use Output parameter]. Rewrite the btnsave click code so as to call this stored procedure while adding the new employee record.

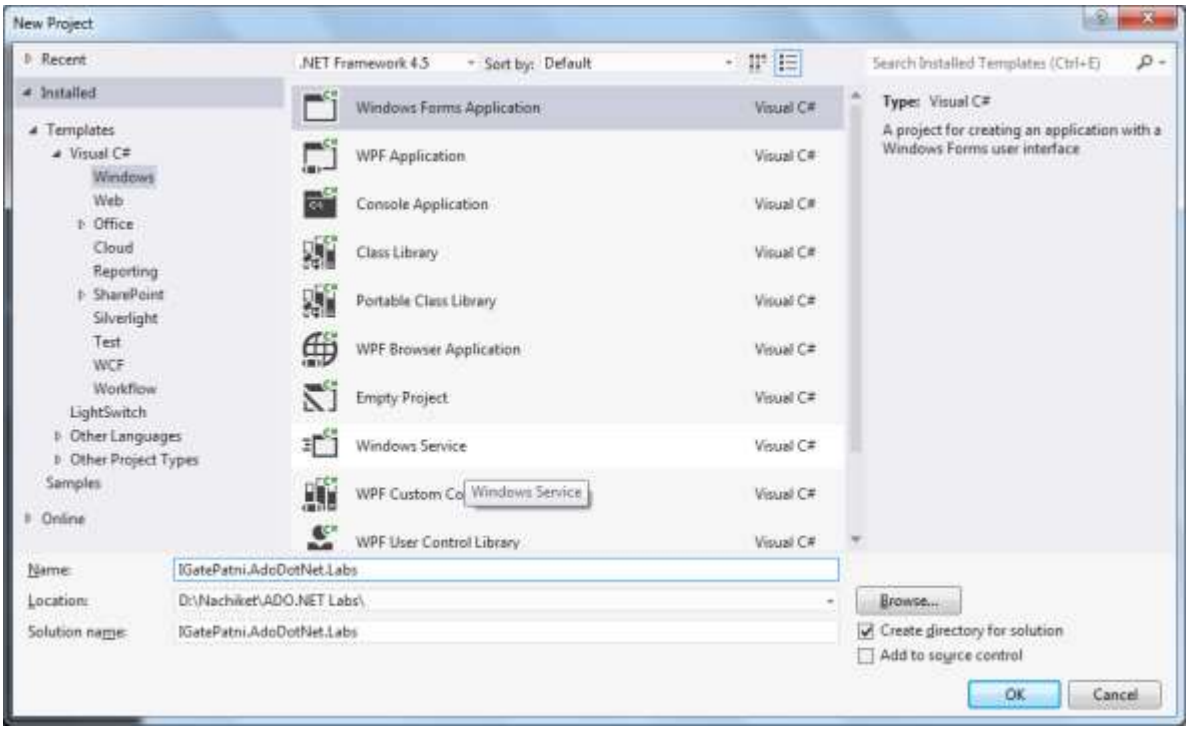
Lab 2. DataSet and DataAdapter

Description	We will be using DataAdapter and DataSet classes to retrieve information about application users from database. User can scroll as well edit the details and save the changes back.
Goals	To Learn - • How to use DataAdapter to retrieve relational data • Use DataSet to store and display data • Save the changes back to database
Time	60 Mins

The Database Table used in this Lab:

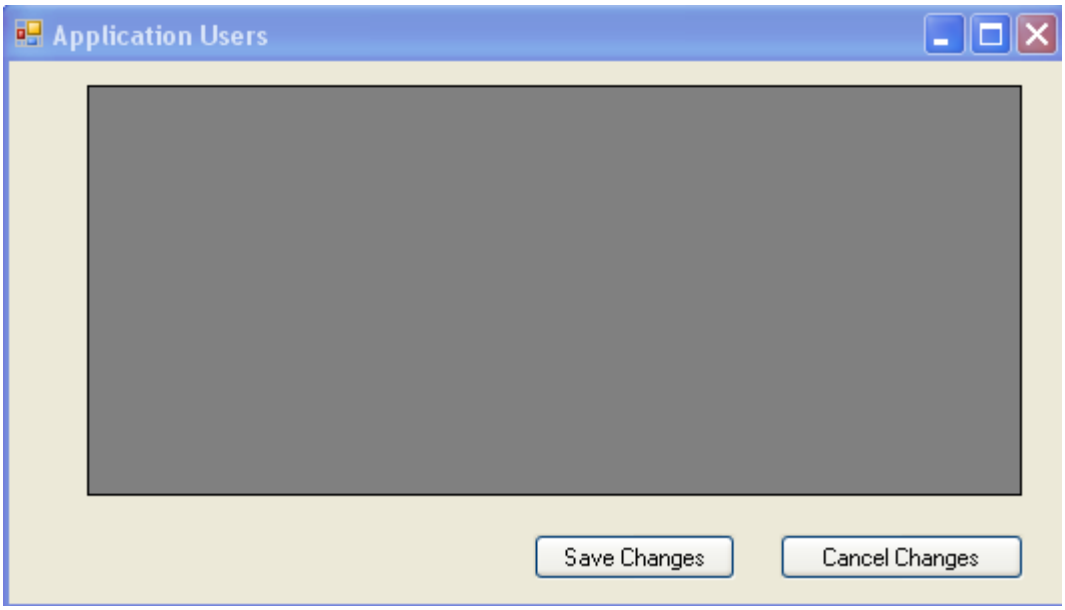
```
create table applicationusers
(
        userid varchar(10) primary key,
        username varchar(30) not null,
        city varchar(30) not null,
        password varchar(30) check(len(password) >5)
)
```

1. Create a new Windows Application Project , Name it IGatePatni.AdoDotNet.Labs



2. Design the Form as below:
   a. Drag GridView (Name: grdUsers) and 2 Buttons

b. Rename the Form1 to DataSetAdapterDemo



3. Include the following namespace in code behind of the form:

```
using System.Data.SqlClient;
```

4. Add the following as Form level members (inside form class below constructor definition)

```
SqlConnection con;
SqlDataAdapter da;
DataSet ds;
```

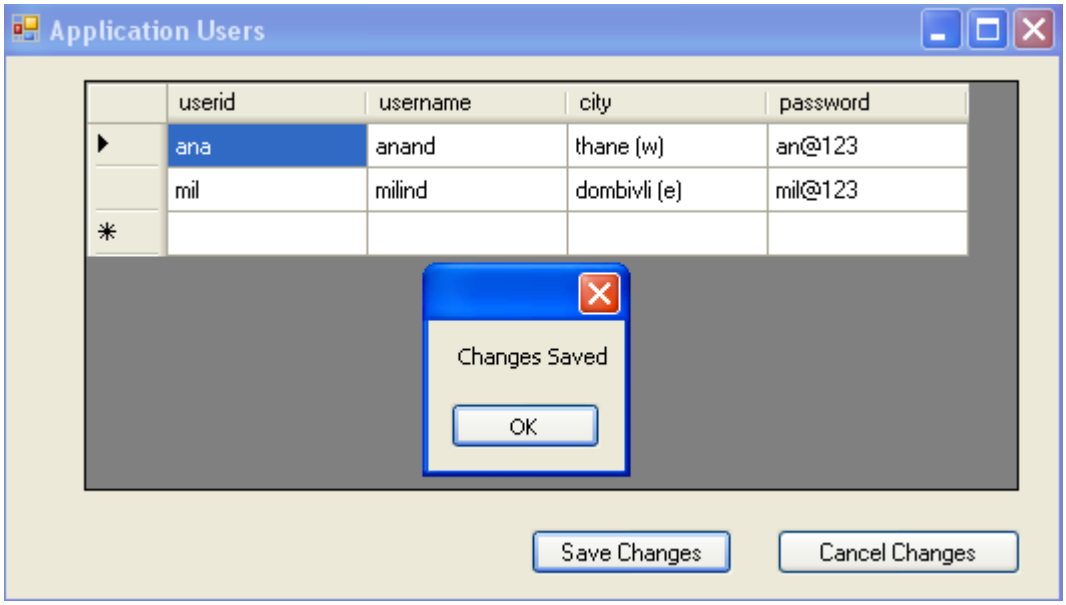
5. The Form Load code:

```
private void DataSetAdapterDemo_Load(object sender, EventArgs e)
{
        con = new SqlConnection (@"server=atrgsql\sql2005;database=labdemos;" +
        "user id=sqluser;password=sqluser");

        con.Open();
        ds = new DataSet();
        //select - For Data Retrieval
        da = new SqlDataAdapter("select * from applicationusers", con);

        //So that we should be able to save changes back to database....
        SqlCommandBuilder bld = new SqlCommandBuilder(da);

        da.Fill(ds, "appusers");

        grdUsers.DataSource = ds.Tables["appusers"];
}
```


6. The Cancel and Save Button Code:

```
private void btncancel_Click(object sender, EventArgs e)
{
        ds.Tables["appusers"].RejectChanges();
}

private void btnsave_Click(object sender, EventArgs e)
{
        try
        {
                //Save Changes to Database
                da.Update(ds.Tables["appusers"]);
                MessageBox.Show("Changes Saved");
        }
        catch (SqlException sqlex)
        {
                MessageBox.Show(sqlex.Message);
        }
}
```

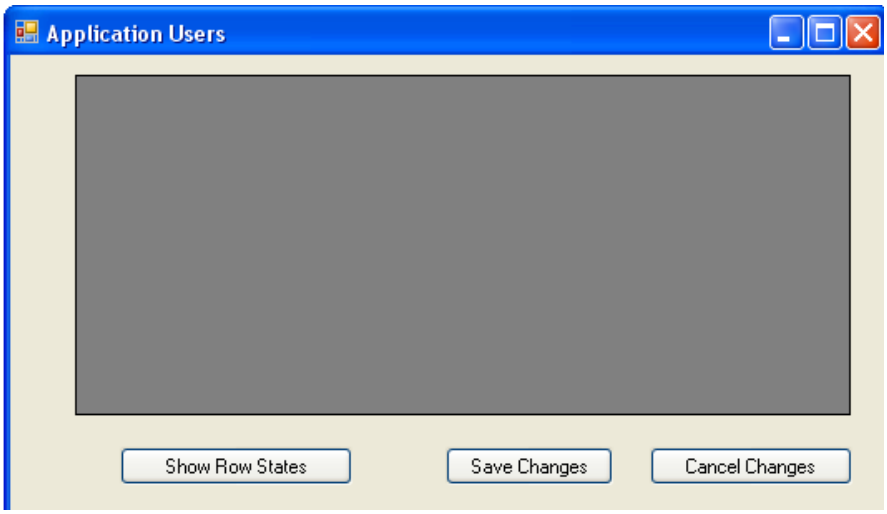
7. Run the application
   a. makes changes in the Grid and click cancel
   b. again make modification and click save button
8. Close and rerun the application to ensure that the changes are saved.



Lab 3. Understand RowState Concept for Data Rows of DataTable

Description	In the previous Lab we will be adding extra code to iterate Data Rows and check their row states.
Goals	To Learn - • Understand RowState Concept • See how the row states changes according to the changes made by the end user; to track it.
Time	30 Mins

1. In the previous lab drag one more button (btnstate) "Show Row States" on the form as below:



2. Write the following code:

```
private void btnstates_Click(object sender, EventArgs e)
{
    //Iterate through Rows of DataTable and display RowStates .....

    foreach (DataRow drow in ds.Tables["appusers"].Rows)
    {
        if (drow.RowState == DataRowState.Deleted)
        {
                MessageBox.Show(drow["username",DataRowVersion.Original].ToString() +
                " deleted ");
        }
        else
                MessageBox.Show(drow["username"].ToString()+ "  " +
drow.RowState.ToString() );
    }
}
```


3. Run the Program check it following test cases:
    a. make changes to the first row and click the above button
    b. click cancel button and recheck the rowstates
    c. make changes to 2nd row and delete 1st row and check their rows states, and then click cancel changes button the deleted row should come back.
    d. Now make changes to both rows and click save changes and check the row states.

Lab 4. Using DataView Object to Filter DataTable

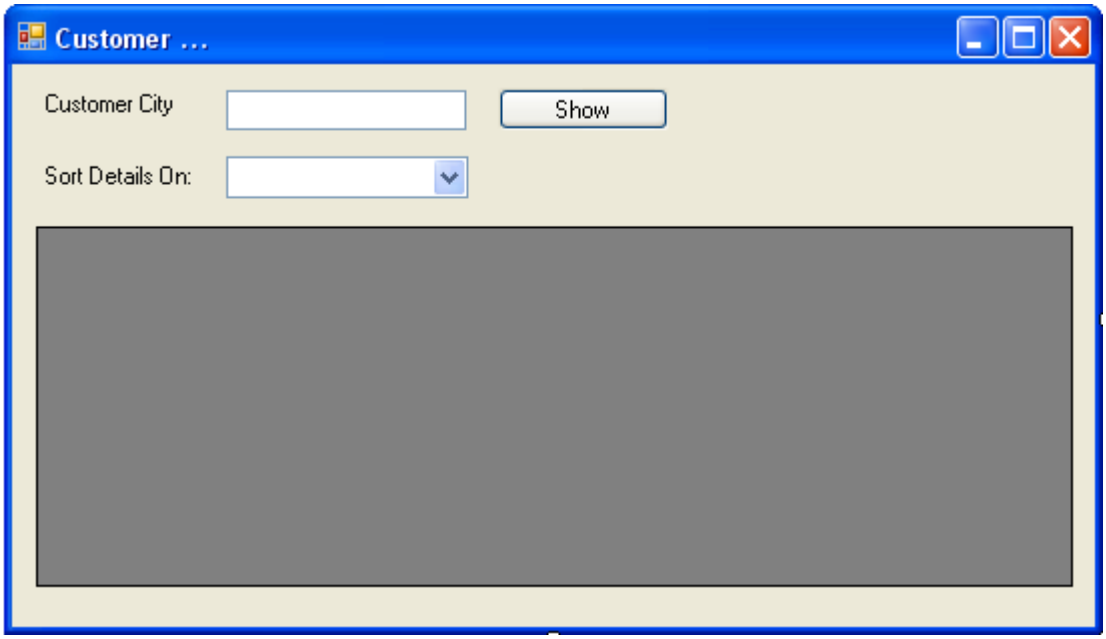
Description	In this Lab we will be displaying customer details and allow the user to filter the customer information on city.
Goals	To Learn - • Understand how to use DataView to do client side filtering and sorting.
Time	90 Mins

The Database Table Structure used in this Lab:

```
create table customer
(
        customerid      int identity primary key,
        customername varchar(50),
        city      varchar(30),
        creditlimit       numeric(10,2)
)
```

1. Add a new blank windows form in the project
   a. Project => Add Windows Form, Name it CustomerForm

2. Drag the controls and design the form as below:
   a. TextBox (txtcity), Button (btnshow), ComboBox (cmbcolumnlist) and grid grdCustomers



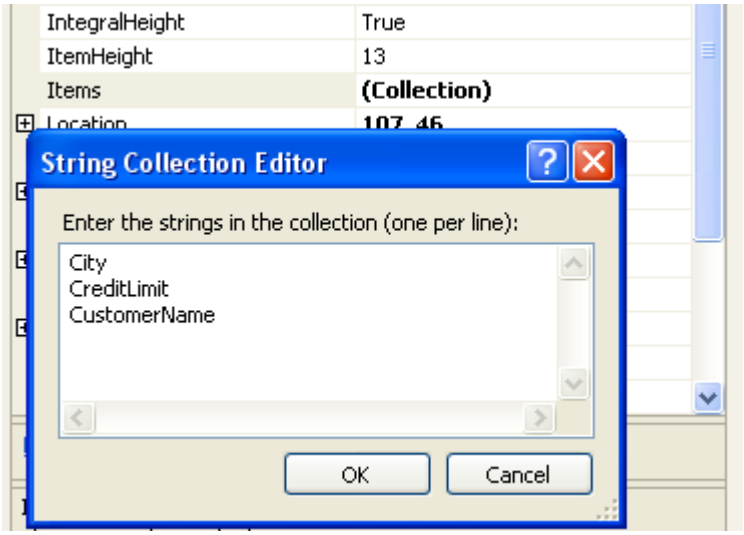
3. In the code behind include using **System.Data.SqlClient;** at the top
4. Define the following Ado.net objects at the form level below:


```
SqlConnection con;
SqlDataAdapter da;
DataSet ds;
```

5. Form Load event code is given

```
private void CustomerForm_Load(object sender, EventArgs e)
{
        con = new SqlConnection (@"server=atrgsql\sql2005;database=labdemos;" +
                "user id=sqluser;password=sqluser");

        con.Open();
        ds = new DataSet();
        //select - For Data Retrieval
        da = new SqlDataAdapter("select * from customer", con);

        da.Fill(ds, "cust");

        grdCustomers.DataSource = ds.Tables["cust"];
}
```

6. Go to Items property of ComboBox and add the following options



7. Set DropDownStyle property of ComboBox to “DropDownList”
8. The ComboBox “SelectedIndexChanged” event code

```
private void cmbcolumnlist_SelectedIndexChanged(object sender, EventArgs e)
{
        ds.Tables["cust"].DefaultView.Sort = cmbcolumnlist.Text;
}
```

9. The Show Button Code:

```
private void btnshow_Click(object sender, EventArgs e)
{
```



```
            ds.Tables["cust"].DefaultView.RowFilter ="City like '" + txtcity.Text + "'";
    }
```

10. Make this form as startup in Program.cs
11. Run the Program
    a. Type “dom*” in city textbox and click show
    b. Type “*n*” in city textbox and click show
    c. Select different column names from combo and see the sorted data in grid.

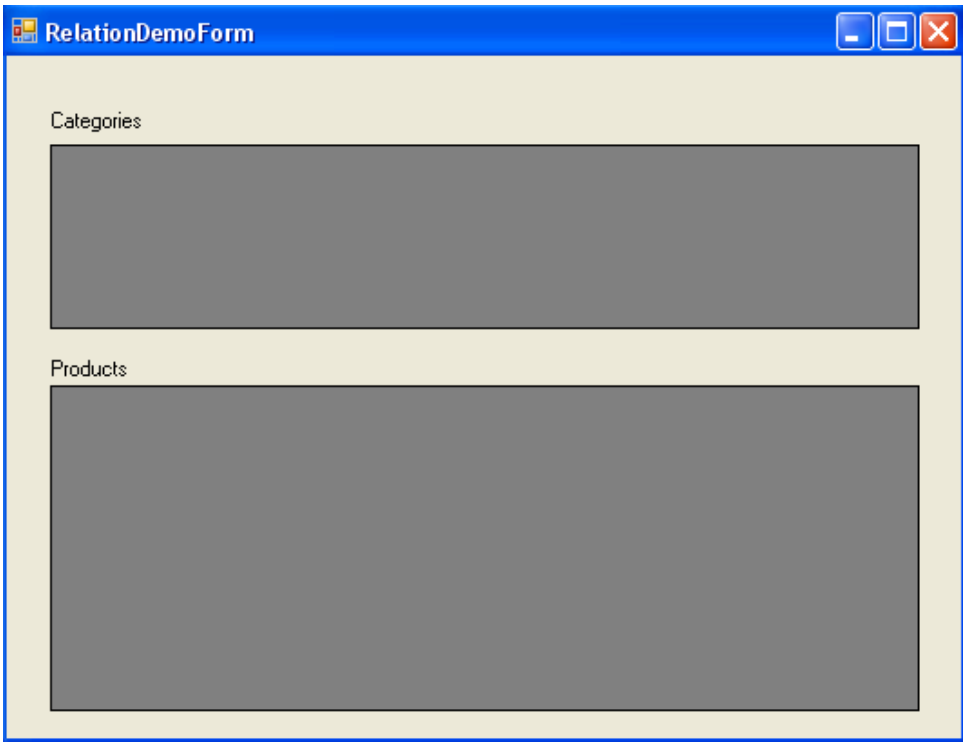
Assignment 3

To Do:
Create a table Supplier with coulmns SupplierId (primary key), Suppliername, City, ContactNo, CreditBalance. Create a Windows Form which will display the all the suppier details in Grid. Allow user to filter the suppliers based on City or Name. The user should also be able to make changes to the supplier details and save the changes back to database.

Lab 5. Using DataRelation Class

Description	In this Lab we will be filling a dataset with categories and products information from database and making a parent-child relationship within dataset.
Goals	To Learn - • Understand how to use establish a master-detail relationship between 2 Data Tables in a DataSet
Time	60 Mins

1. Add a New Windows Form and design as below:
   a. DataGridViews: grdCategories, grdProducts



2. Add the following in the Form Class Code
3. This Code
   a. Fills the DataSet with Categories, Products Details.
   b. Creates a DataRelation based on common column (Categoryid in this case) and adds it to Relations Collection of Dataset.
   c. Then sets the DataSource of both the grids. (Note the way the datasource is set for grid “grdProducts”)
   d. Sets the Update and Delete Cascade Rules.


```
//FORM LEVEL MEMBERS
SqlConnection con = new SqlConnection(@"server=atrgsql\sql2005;database=labdemos;" +
            "user id=sqluser;password=sqluser");
SqlDataAdapter dacat, daprod;
DataSet ds_cat_pro;

private void RelationDemoForm_Load(object sender, EventArgs e)
{
        dacat = new SqlDataAdapter("select * from category", con);
        daprod = new SqlDataAdapter("select * from product", con);
        con.Open();
        ds_cat_pro   = new DataSet();
        dacat.Fill(ds_cat_pro, "cat");
        daprod.Fill(ds_cat_pro, "pro");
        con.Close();
        //Setting Default Constraint

        ds_cat_pro.Tables["pro"].Columns["categoryid"].DefaultValue = 1;

        //CREATING RELATION BETWEEN THE TWO DATA TABLES
        DataRelation dre1 = new DataRelation("catpro_relation",

        ds_cat_pro.Tables["cat"].Columns["CategoryId"],

        ds_cat_pro.Tables["pro"].Columns["CategoryId"]);
        ds_cat_pro.Relations.Add(dre1);

        ds_cat_pro.Relations["catpro_relation"].ChildKeyConstraint.DeleteRule
                                                        = Rule.None;
        ds_cat_pro.Relations["catpro_relation"].ChildKeyConstraint.UpdateRule
                                                        = Rule.None;

        //DIFFERENT CONSTRAINT RULE OPTIONS:
        //NONE          WILL NOT ALLOW DELETING MASTER RECORD
        //CASCADE       WILL DELETE MASTER AS WELL AS CHILD RECORDS
        //SETDEFAULT:   WILL ALLOW DELETION OF MASTER RECORD AND SET THE
                        VALUE  IN CHILD WHICH IS DEFAULT
       //SETNULL:       WILL SET VALUE OF COLUMN TO NULL IN CHILD TABLE AND
                        DELETE RECORD FROM MASTER TABLE

       grdCategories.DataSource = ds_cat_pro.Tables["cat"];

        grdProducts.DataSource = ds_cat_pro.Tables["cat"];
        grdProducts.DataMember = "catpro_relation";
}
```

4. Run the Application
   a. The Grid will be loaded with category and product details
   b. If you navigate category records the corresponding product details will be automatically displayed.
   c. Try deleting a category by selecting the whole Grid Row and pressing delete key, it will fail because of delete rule set to none.
   d. Close the application
5. Change the delete rule to cascade and repeat the above step.

Lab 6. LINQ Basics

Description	In this Lab we will be implementing LINQ with collection and LINQ operators
Goals	To Learn • Understand the process of Implementing LINQ to a Collection • Learn to use LINQ • Learn to use LINQ Operators
Time	60 Mins

1. Create a console application and add class named Employee with following field.
   Employee Class
   EmployeeID (Integer)
   FirstName (String)
   LastName (String)
   Title (String)
   DOB (Date)
   DOJ (Date)
   City(String)

2. Create a Generic List Collection empList and populate it with the following records.

EmployeeID	FirstName	LastName	Title	DOB	DOJ	City
1001	Malcolm	Daruwalla	Manager	16/11/1984	8/6/2011	Mumbai
1002	Asdin	Dhalla	AsstManager	20/08/1984	7/7/2012	Mumbai
1003	Madhavi	Oza	Consultant	14/11/1987	12/4/2015	Pune
1004	Saba	Shaikh	SE	3/6/1990	2/2/2016	Pune
1005	Nazia	Shaikh	SE	8/3/1991	2/2/2016	Mumbai
1006	Amit	Pathak	Consultant	7/11/1989	8/8/2014	Chennai
1007	Vijay	Natrajan	Consultant	2/12/1989	1/6/2015	Mumbai
1008	Rahul	Dubey	Associate	11/11/1993	6/11/2014	Chennai
1009	Suresh	Mistry	Associate	12/8/1992	3/12/2014	Chennai
1010	Sumit	Shah	Manager	12/4/1991	2/1/2016	Pune

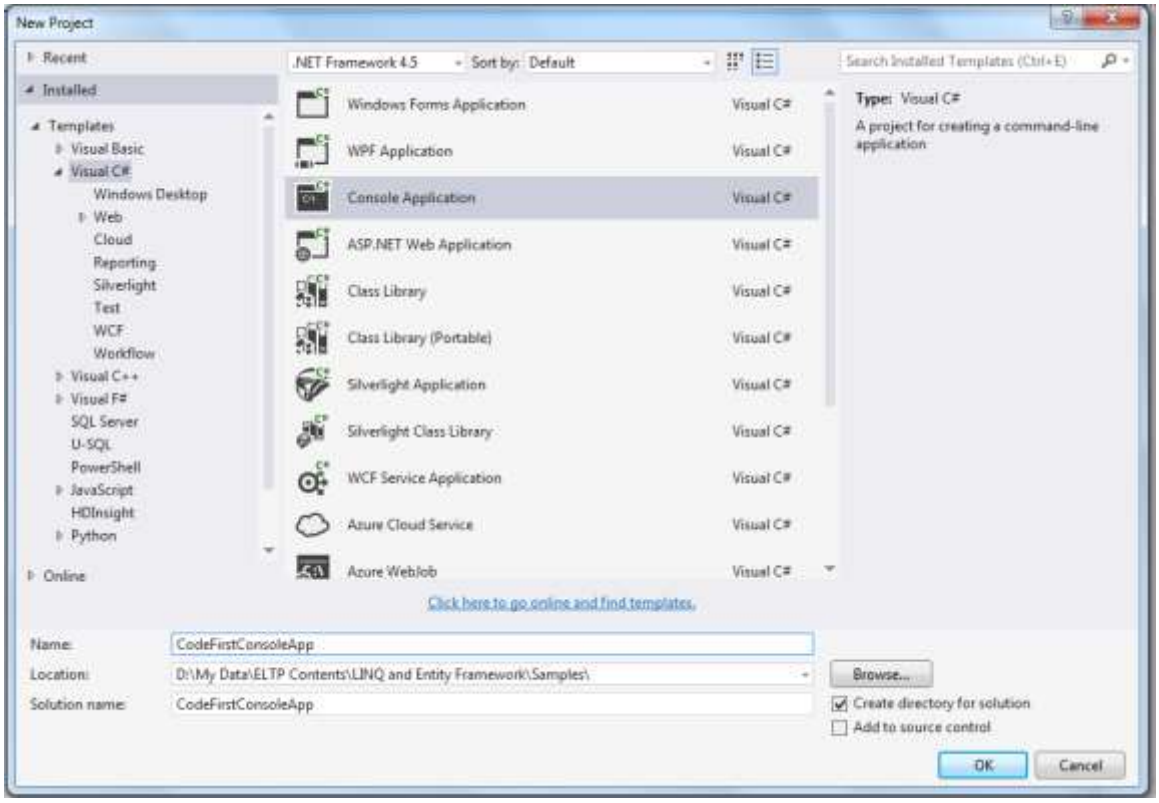
3. Now once the collection created write down and execute the LINQ queries for collection as follows :
   i) Display detail of all the employee
   ii) Display details of all the employee whose location is not Mumbai
   iii) Display details of all the employee whose title is AsstManager
   iv) Display details of all the employee whose Last Name start with S
   v) Display a list of all the employee who have joined before 1/1/2015
   vi) Display a list of all the employee whose date of birth is after 1/1/1990
   vii) Display a list of all the employee whose designation is Consultant and Associate
   viii) Display total number of employees
   ix) Display total number of employees belonging to “Chennai”
   x) Display highest employee id from the list
   xi) Display total number of employee who have joined after 1/1/2015
   xii) Display total number of employee whose designation is not “Associate”
   xiii) Display total number of employee based on City
   xiv) Display total number of employee based on city and title
   xv) Display total number of employee who is youngest in the list

Lab 7. Creating Entity Data Model

Description	In this Lab we will be filling a dataset with categories and products information from database and making a parent-child relationship within dataset.
Goals	To Learn - • Understand the process of Creating Entity Data Model • Learn to use Code First Approach • Learn to use Database First Approach
Time	60 Mins

Part 1:-
Using Code-First approach

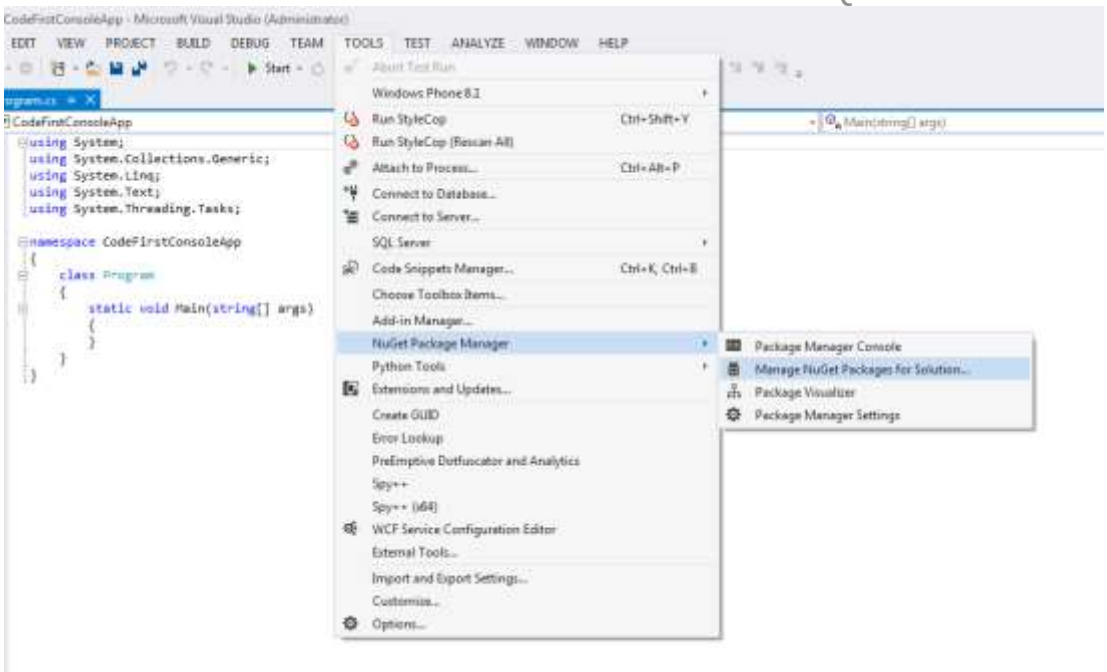
Solution:-
Create a Console application and name the application as **CodeFirstConsoleApp**



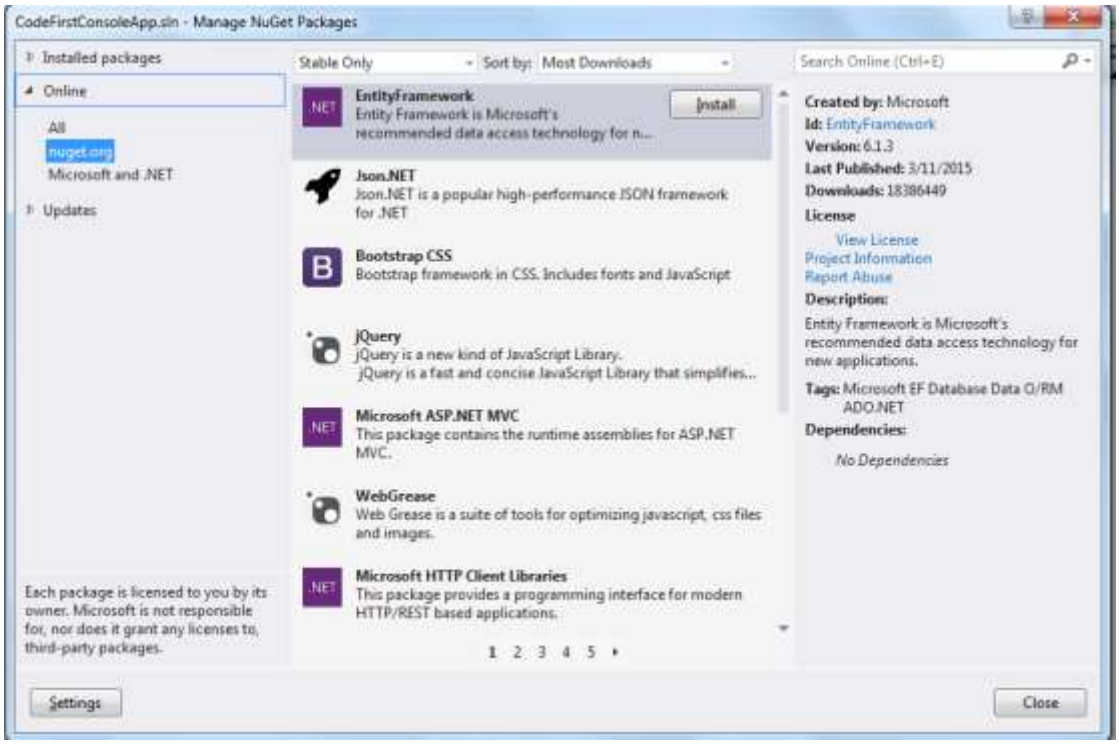
After the project is created Now we have to add the Entity Framework Library to Project. For that we will Nuget Package Manager Dialog

To Open Nuget Package Manager Dialog we need to follow the following step

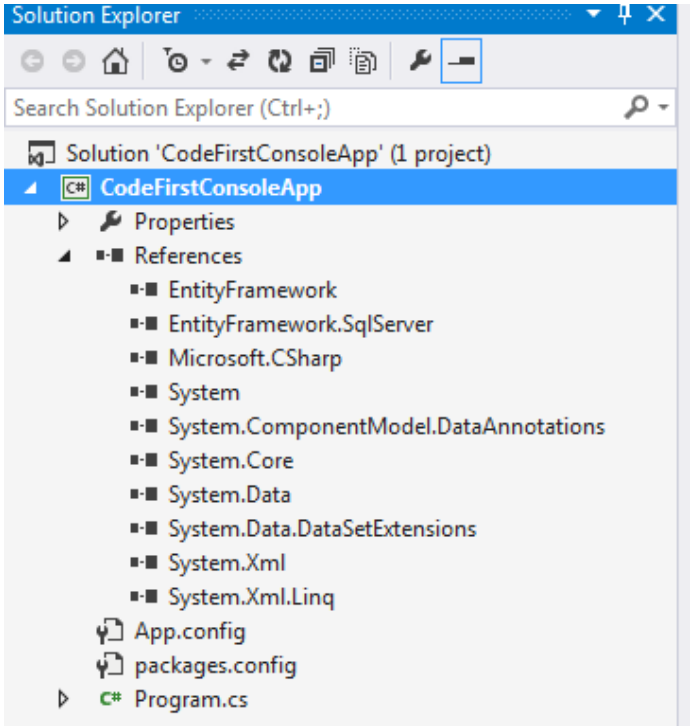
Tools -> Nuget Package Manager -> Manage Nuget Package for the Solution



In the dialog box for Nuget Package Manager select entity framework and click on install this will install EntityFramework to the project

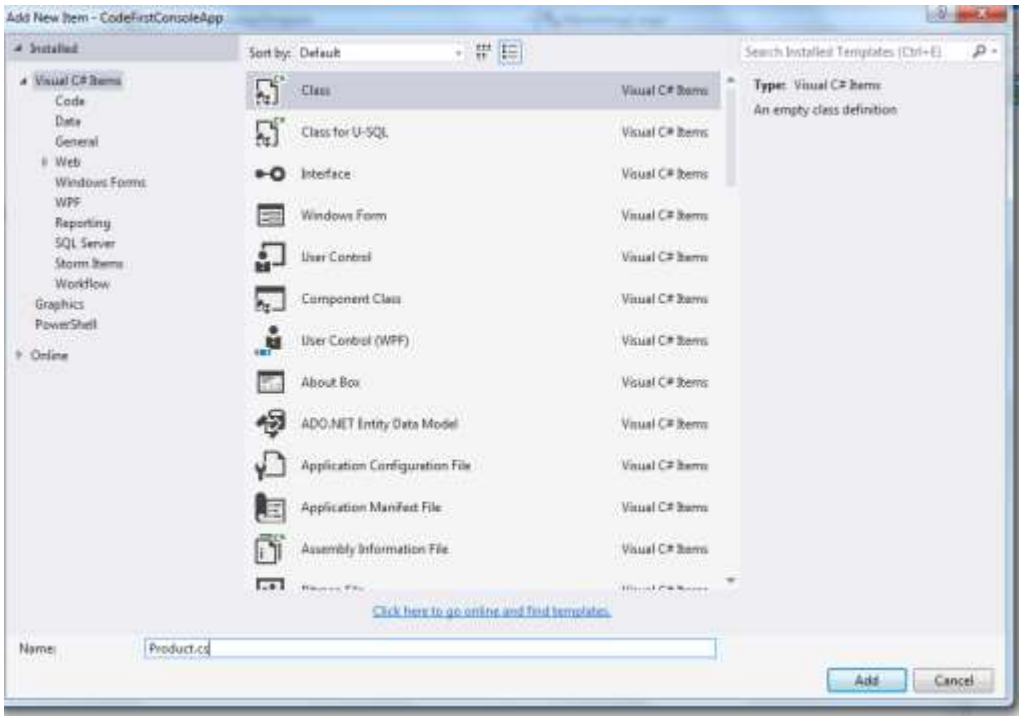


After installing the EntityFramework to the project we can see that EntityFrame dll file has been added to the References folder in Solution Explorer



Now as we have added the EntityFramework dll to the project now we have to add the Entity and Context to the project.

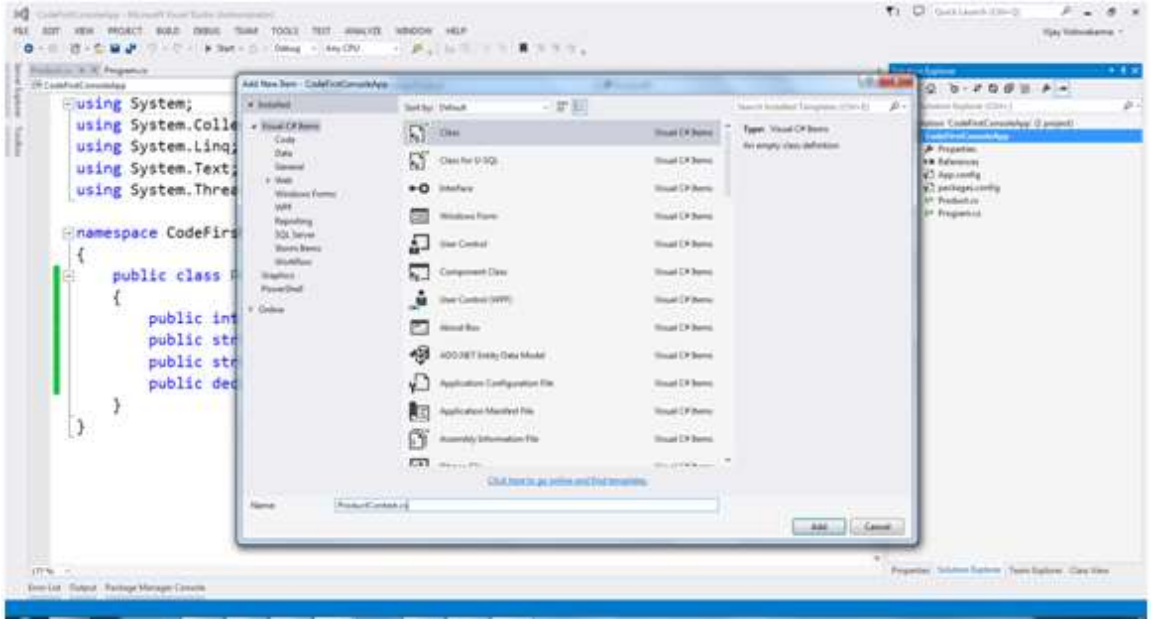
To add a entity class In the Solution Explorer right click on the project name than Add □ Class and name the class as Product.cs



Once the class has been created add the following code to the class

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace CodeFirstConsoleApp
{
    public class Product
    {
        public int ProductID { get; set; }
        public string Name { get; set; }
        public string Category { get; set; }
        public decimal Price { get; set; }
    }
}
```

After adding the class now we have to add a Context Class to the project .Context class will allow to perform database operation like add ,delete etc.
Context Class will always inherit from DbContext class available in System.Data.Entity namespace



After adding the context class add the following code to the class

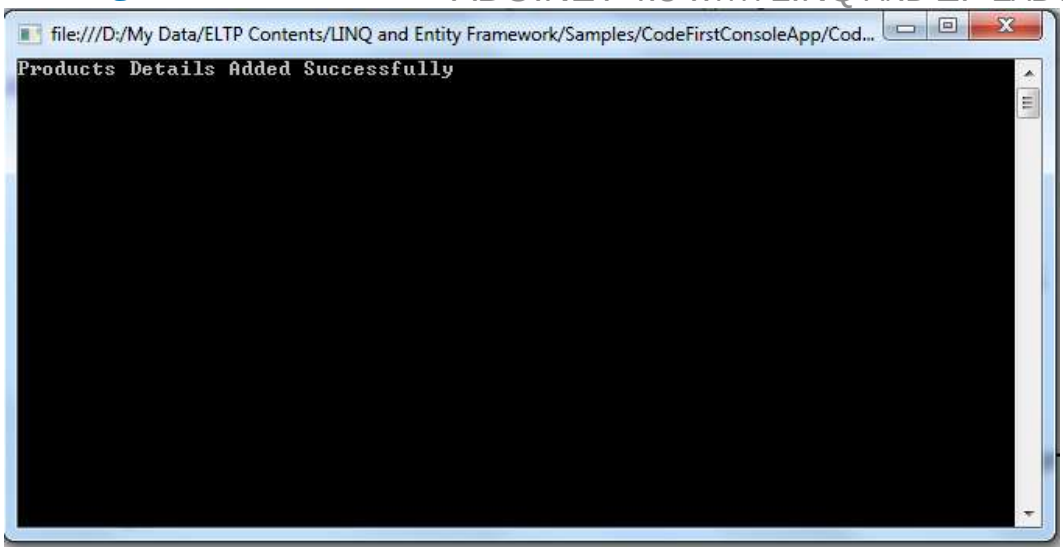

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Data.Entity;

namespace CodeFirstConsoleApp
{
    class ProductContext : DbContext
    {
        public ProductContext() : base()
        {
        }

        public DbSet<Product> Products { get; set; }
    }
}
```

In the Program.cs class file add the following code

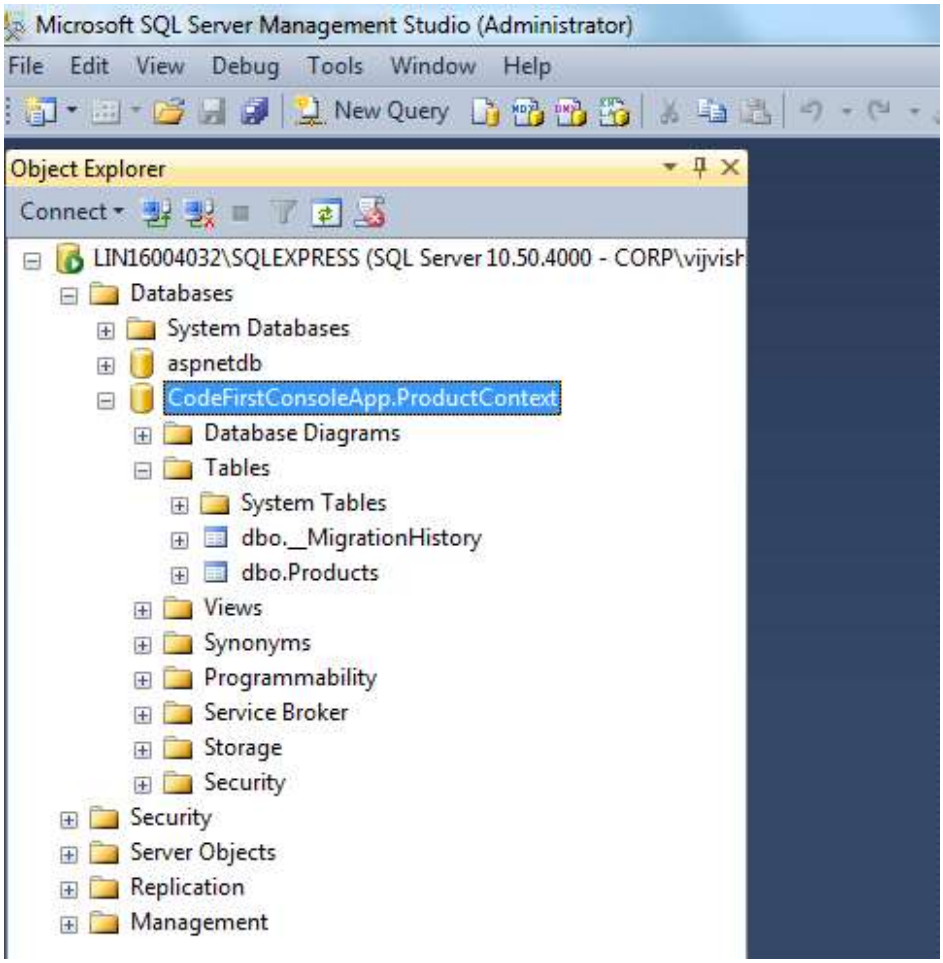
```
{
    class Program
    {
        static void Main(string[] args)
        {
            ProductContext pcontext = new ProductContext();

            Product p1 = new Product()
            {
                ProductID = 101,Name = "Barbie Doll",Category = "Toys",Price = 199.99M
            };

            Product p2 = new Product()
            {
                ProductID = 102,Name = "Montex Pen",Category = "Stationary",Price = 10.99M
            };

            pcontext.Products.Add(p1);
            pcontext.Products.Add(p2);
            pcontext.SaveChanges();

            Console.Write("Products Details Added Successfully");

            Console.ReadLine();
        }
```

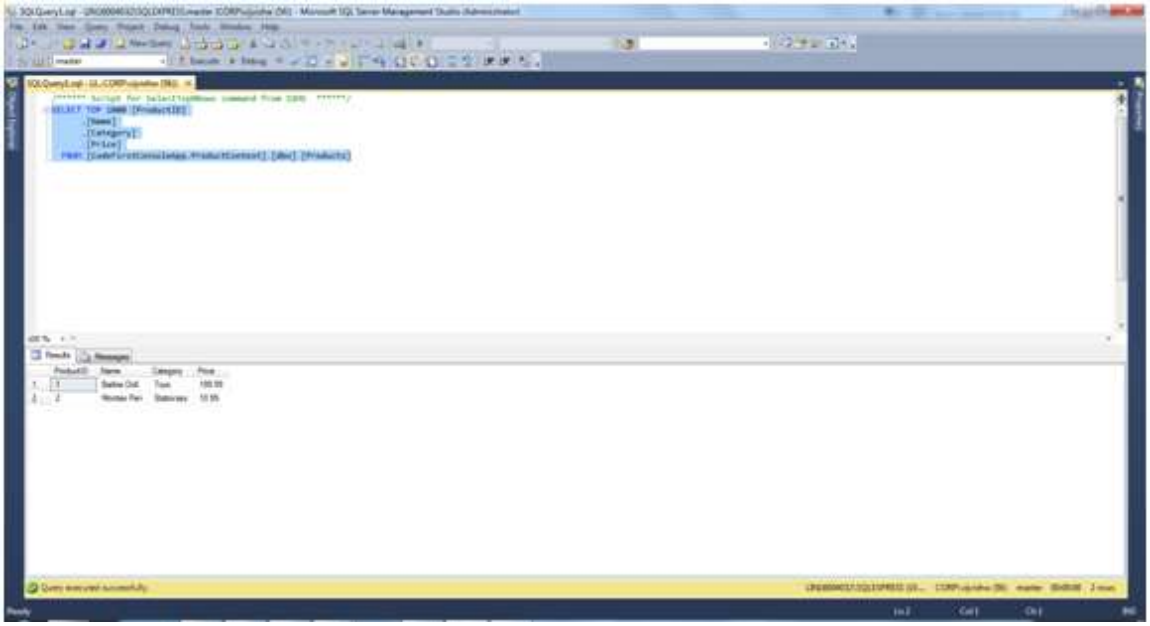
When the above code is executed it will create the database and table based on the Entity and the above record into the table

After executing the application we will get the following output



Now we will check database which is created for that open Sql Server Management Studio and connect to the default instance. In the object explorer you can see the database and table being created





Part 2:-
Using Database First Approach

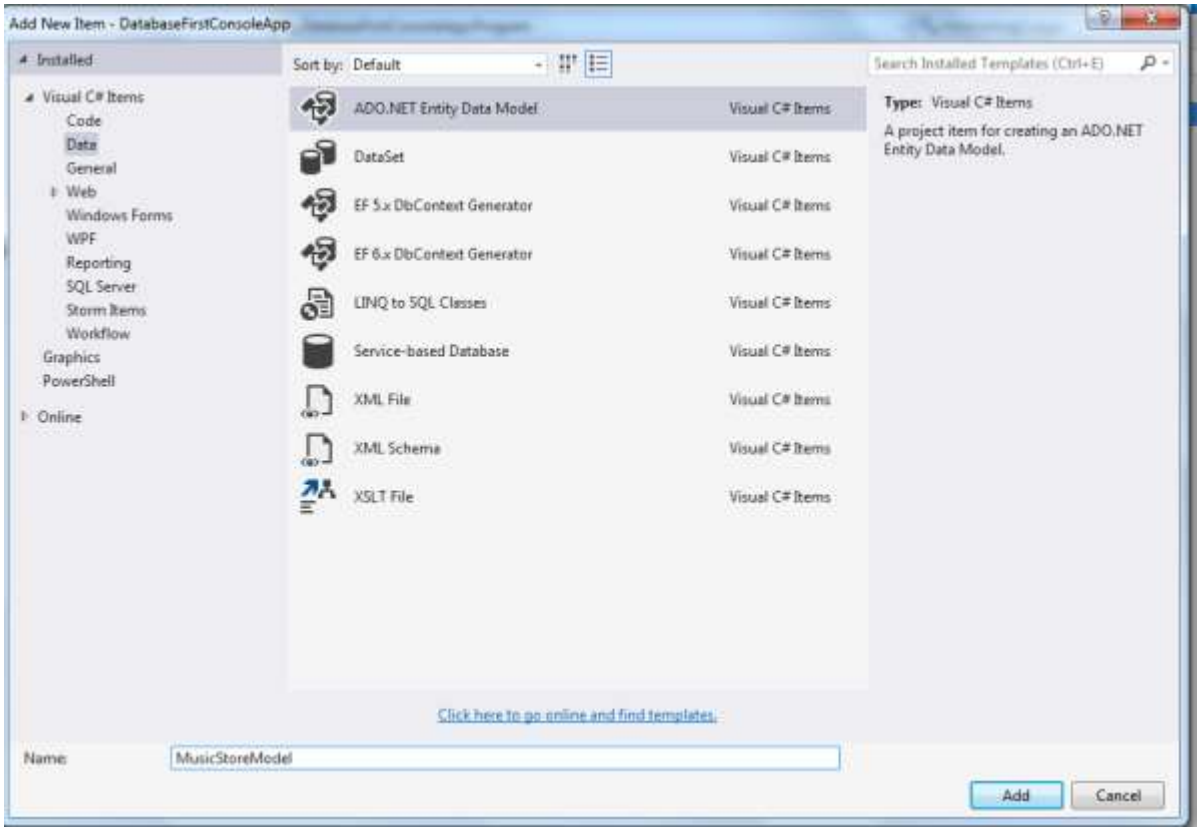
Solution:-
Open Sql Server Management Studio and Create a database name MusicStore and add aTable named Album with the following fields

Album
- AlbumID
- Name
- Genre
- Year
- Price

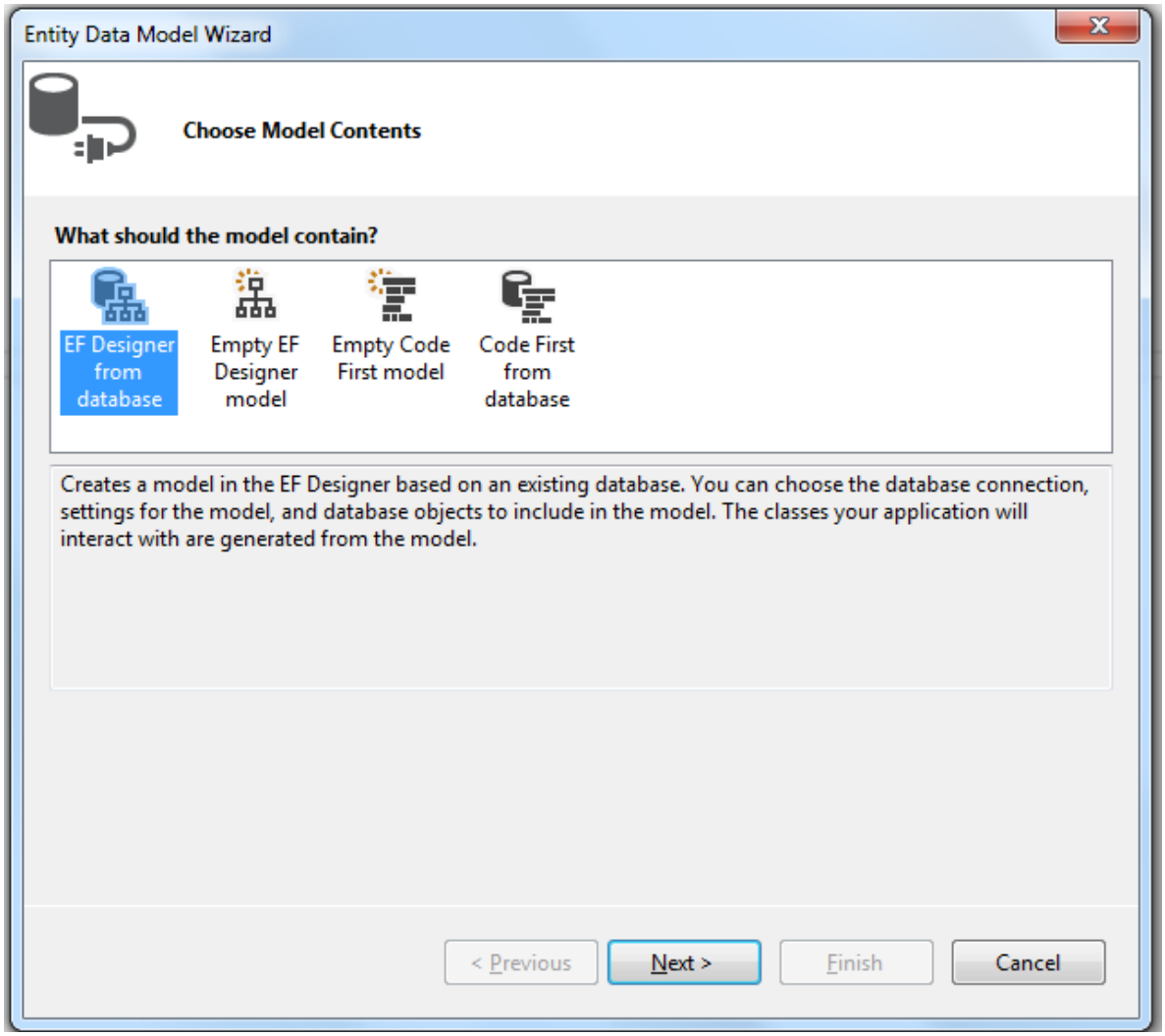
Add some dummy record into the table.

Now create a console application name DatabaseFirstConsoleApp and add the entityframework as done in the previous example

Once the project in create and entity framework library is added . Now we have a Entity data Model to the project .
To add a Entity data Model to the Project in the solution explorer right click in the project
Add -> New Item. Under the New Item dialog box select Ado.Net Entity Data Model and give name as MusicStoreModel and click on Add

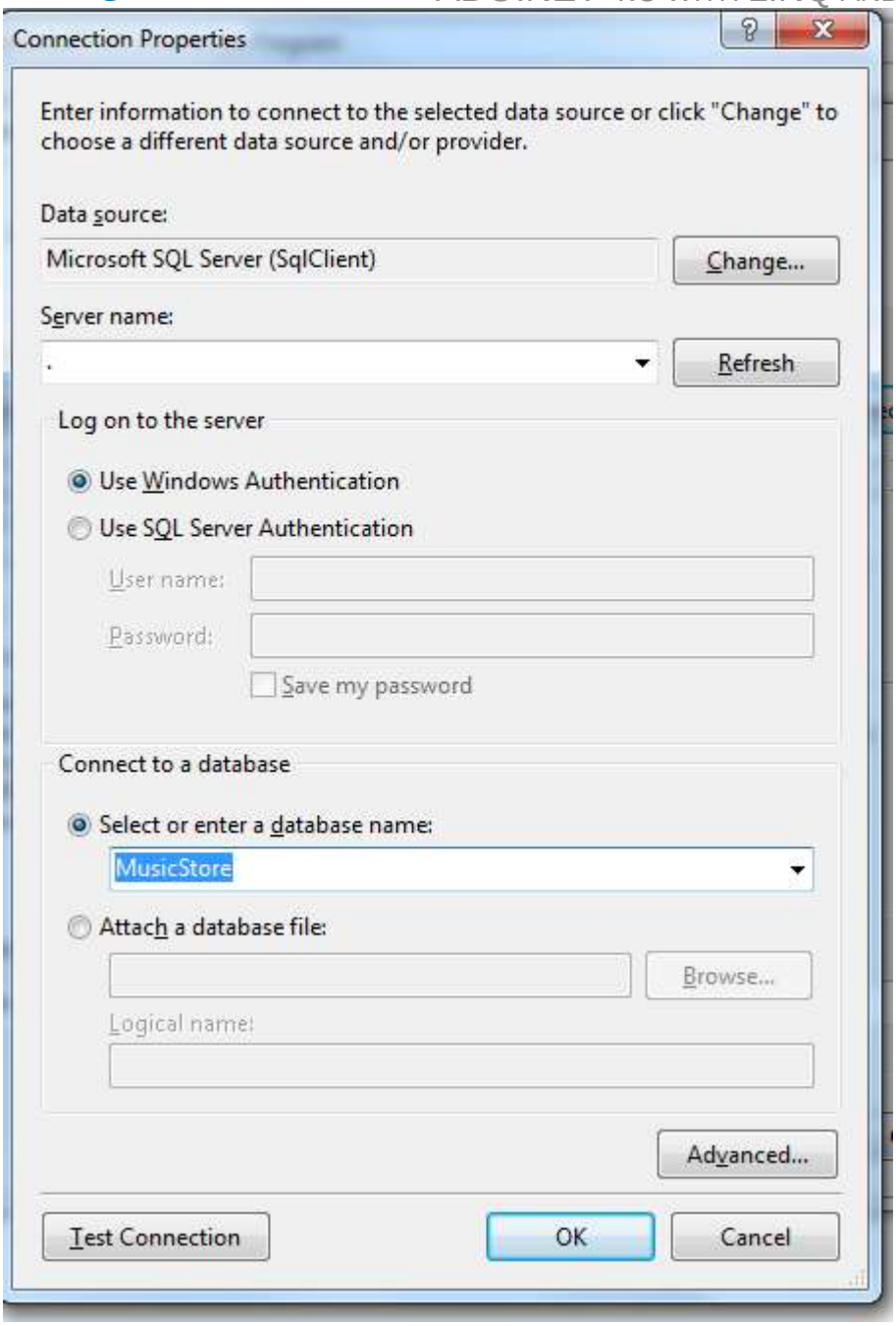


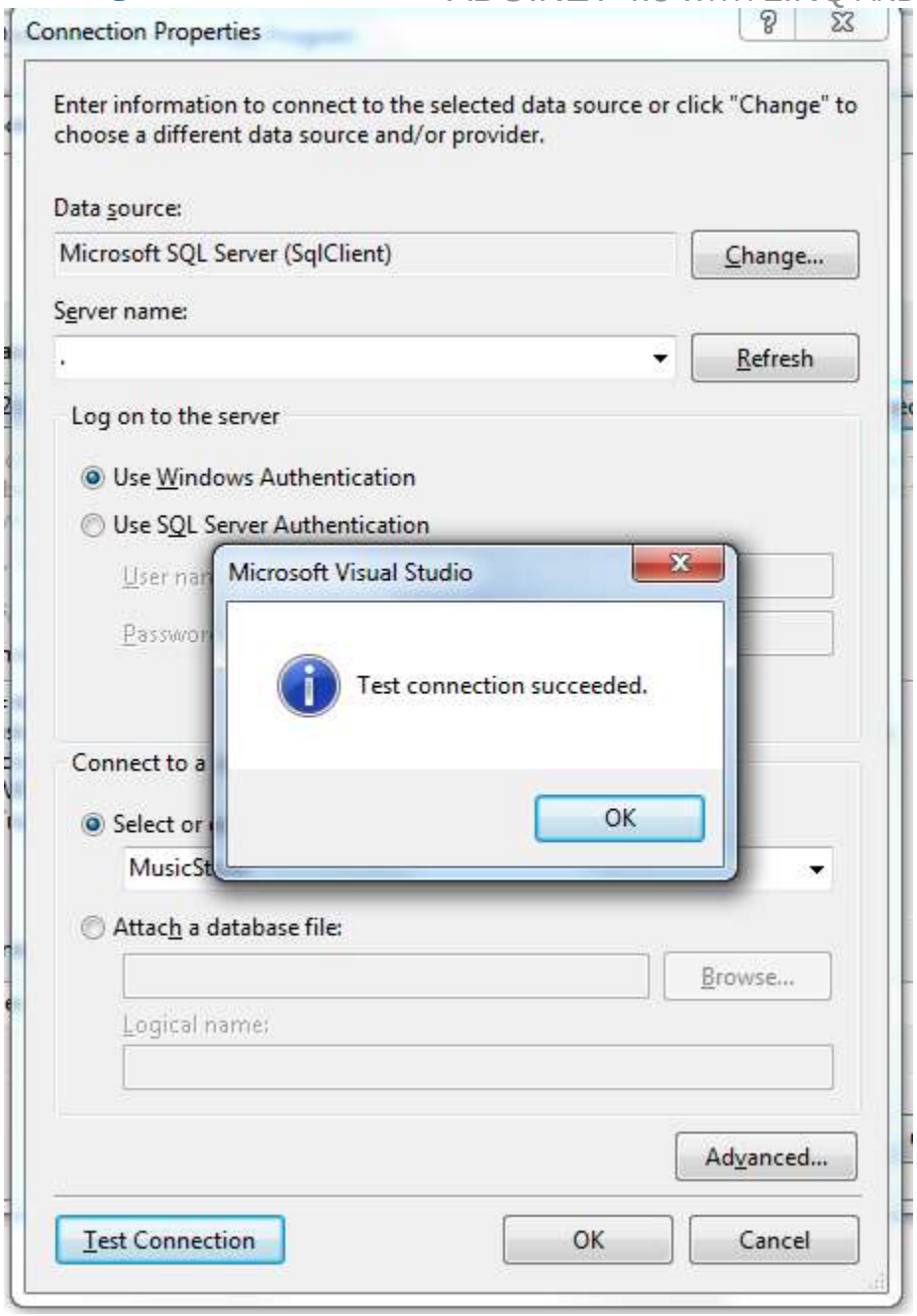
As we add the Entity Data model to the project . Entity Data model wizard popup in which we have different option for initializing the entity data model
In that dialog box select EF Designer from database and click on Next



Now on the next window we have select database for Model creation so now click on New Connection button in Choose Your Data Connection

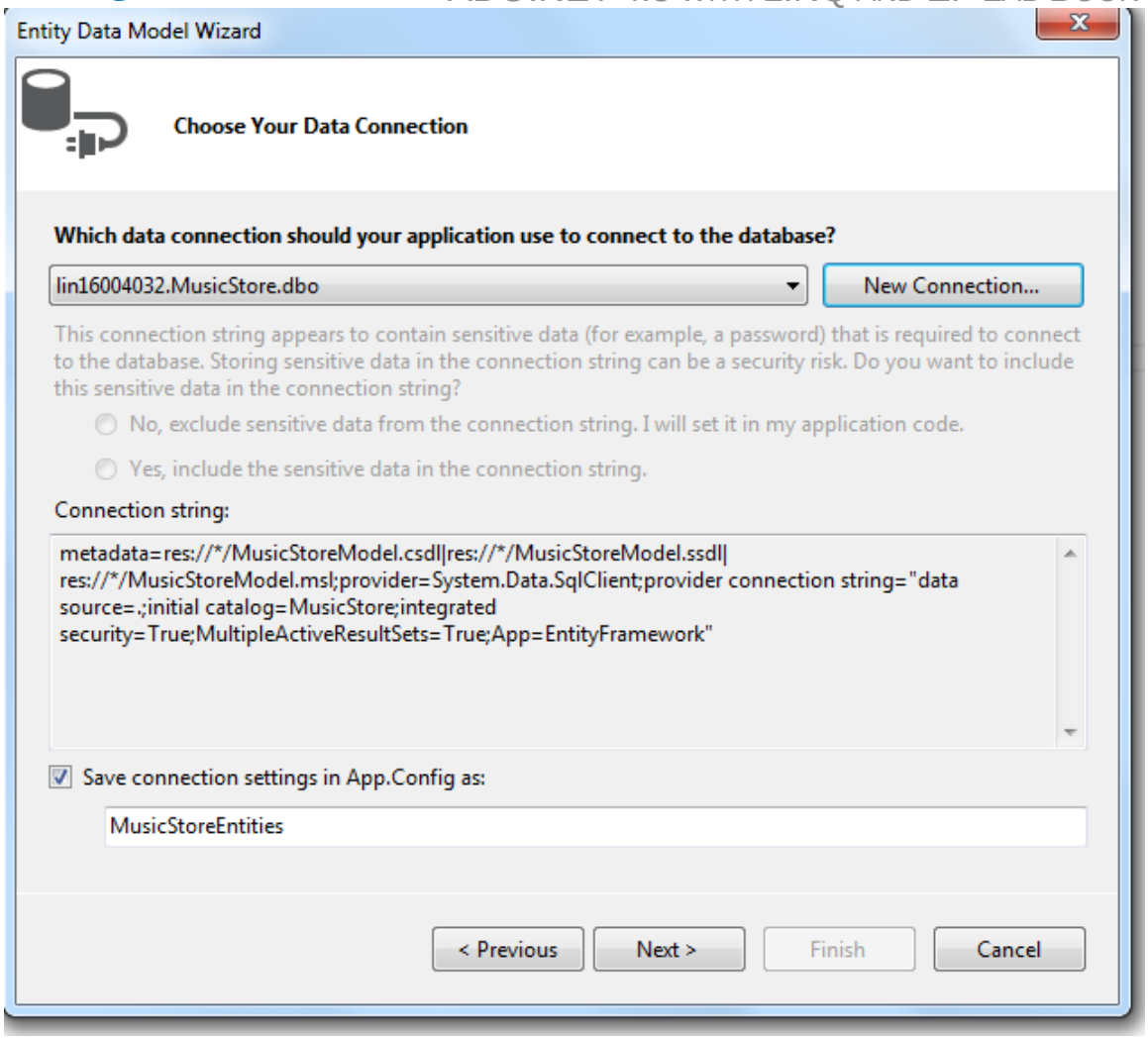
In the Connection properties dialog box provide the Database server name and select the database you want to use. Click on Test Connection to test the connection and then click on OK



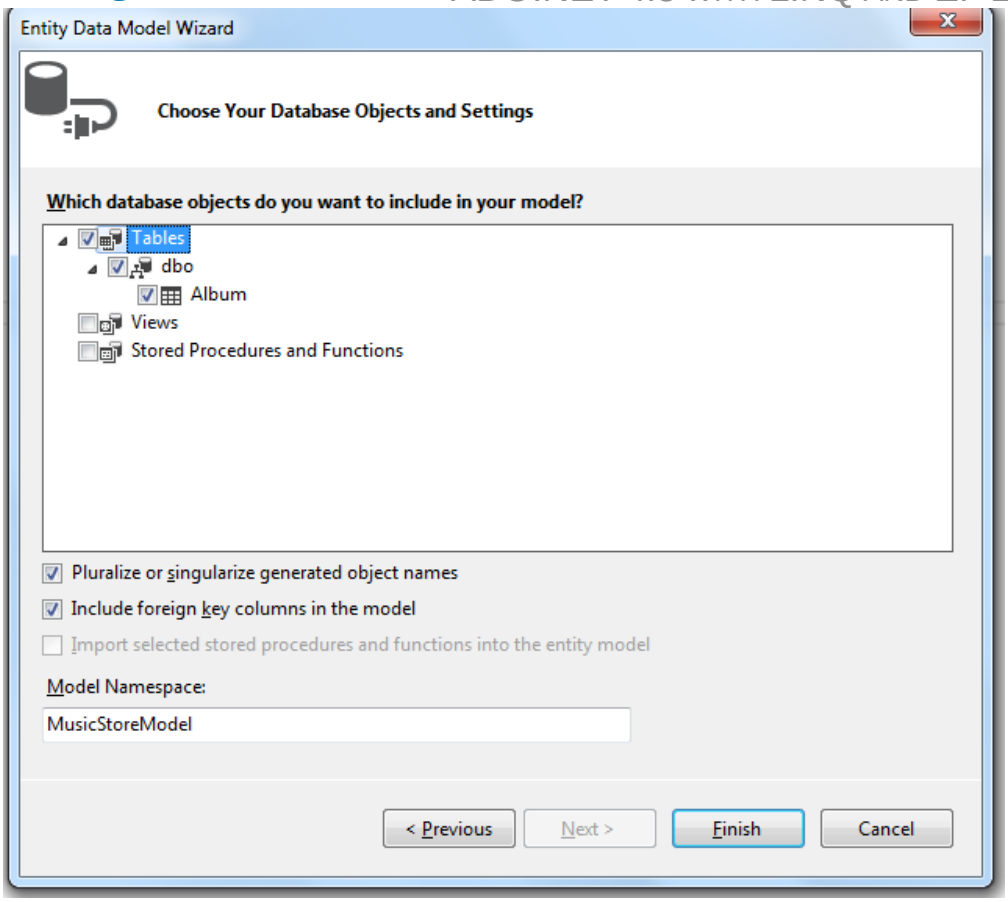


Once the connection test is passed then click on OK

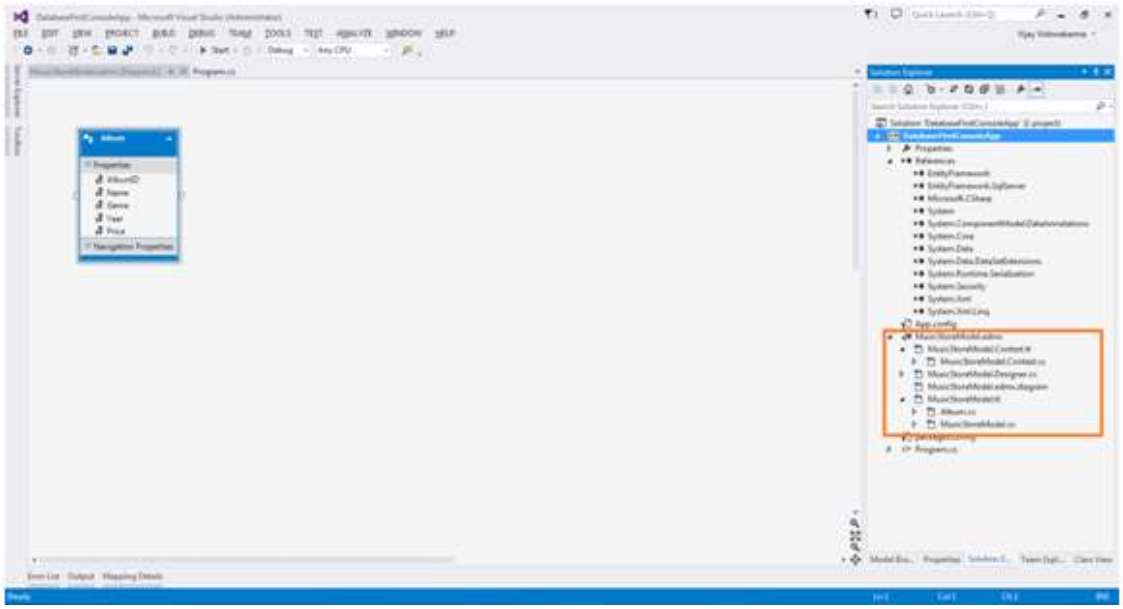
Now we can see the new connection string which we have created and a option to save the connection string in App.config or web.config file



Now Click on Next and Choose your Database Object and Settings option will be prompted . In that we have select all the database object which we need to add to our Model.



We have select the Album table as we have only one table in the database
Now click on Finish this will add the EDM to the project and create all the required code.



In the above image we can see the model name MusicStoreModel.edmx containing Album Entity and the highlighted region show files generated for MusicStoreModel.edmx

Now as the model is created we can write code to interact with database and perform read/write operations.

To display details of the Albums write down the following code in Program.cs File

```
namespace DatabaseFirstConsoleApp
{
    class Program
    {
        static void Main(string[] args)
        {
            MusicStoreEntities musicEntities = new MusicStoreEntities();

            var albums = musicEntities.Albums;

            foreach (Album a in albums)
            {
                Console.WriteLine(a.AlbumID+"\t"+a.Name+"\t"+a.Price);
            }

            Console.ReadLine();
        }
    }
}
```

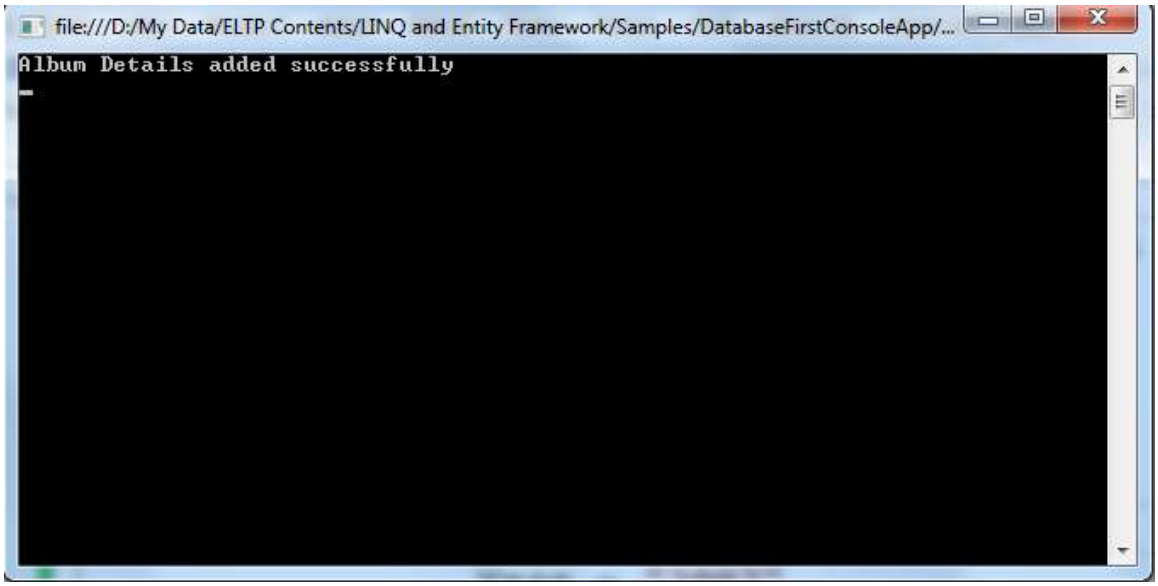
Output:-

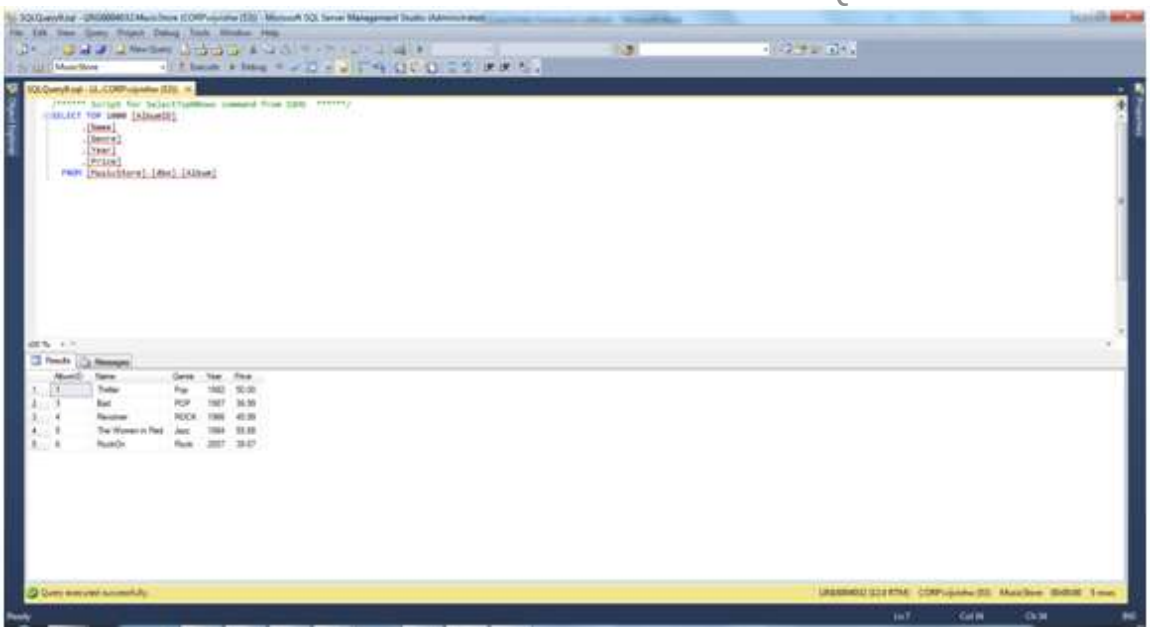
```
file:///D:/My Data/ELTP Contents/LINQ and Entity Framework/Samples/DatabaseFirstConsoleApp/...
1       Thriller        50.00
3       Bad     36.99
4       Revolver        45.99
5       The Women in Red        55.88
```


To add an Album into the database table write the following code in Program.cs File

```
MusicStoreEntities musicEntities = new MusicStoreEntities();

Album album = new Album()
{
    Name="RockOn",
    Genre="Rock",
    Year="2007",
    Price="39.87",
};

musicEntities.Albums.Add(album);
musicEntities.SaveChanges();

Console.WriteLine("Album Details added successfully");
Console.ReadLine();
}
}
}
```

Output:-

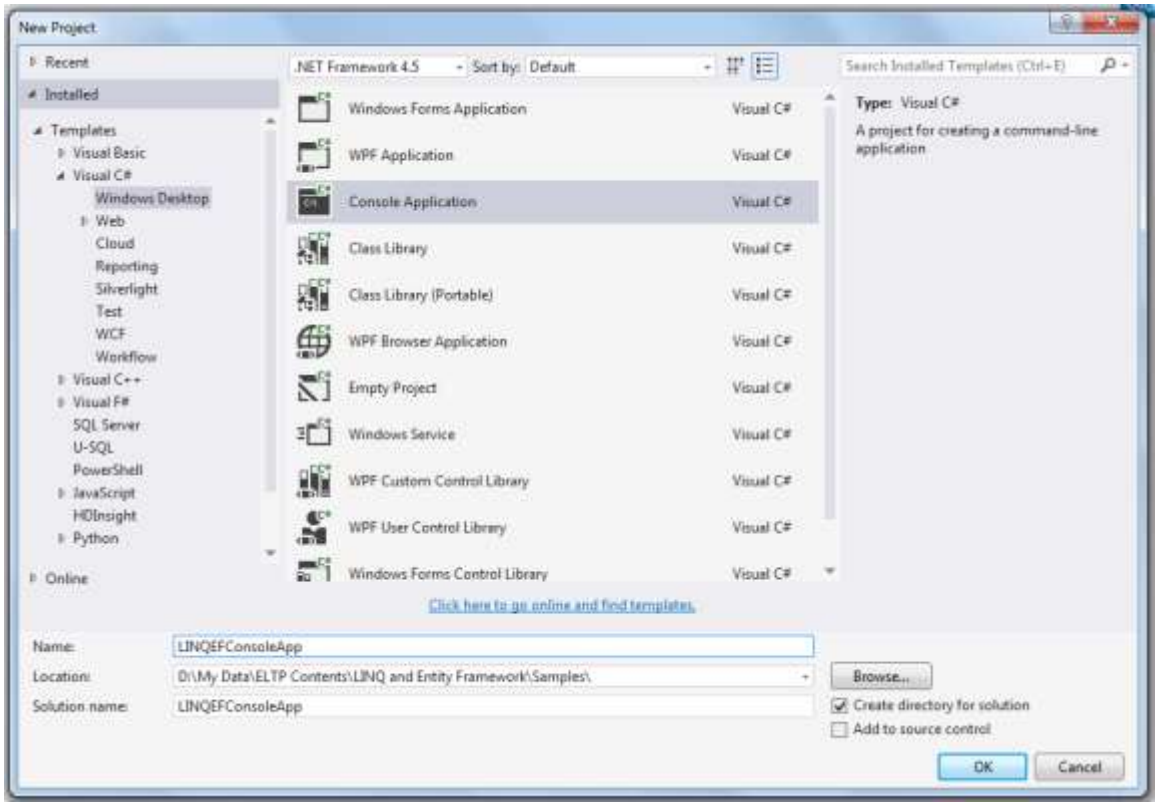


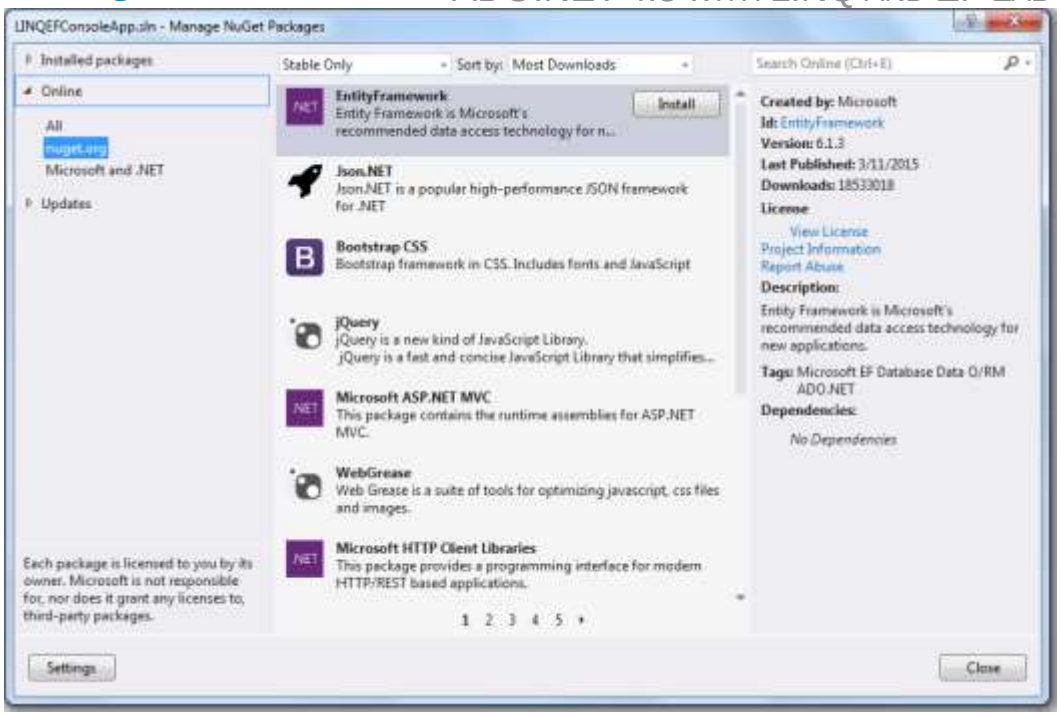


Lab 8. Basic Query Operations using LINQ to Entities

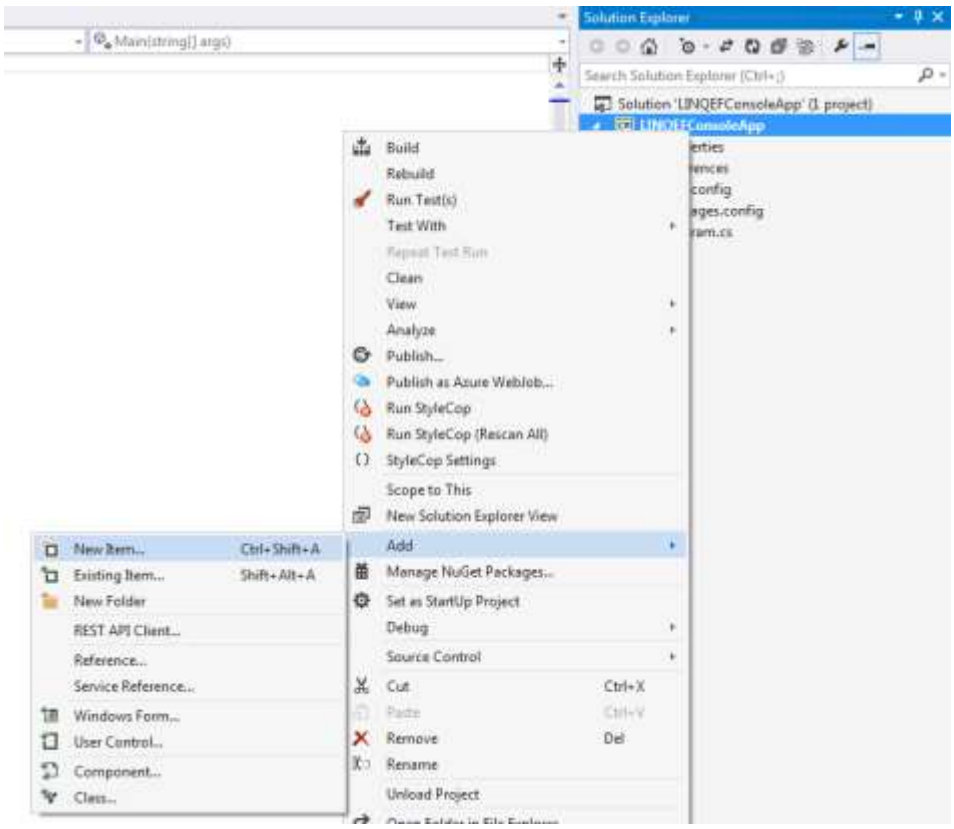
Description	In this Lab we will be filling a dataset with categories and products information from database and making a parent-child relationship within dataset.
Goals	To Learn - • Understand how to use establish a master-detail relationship between 2 Data Tables in a DataSet
Time	60 Mins

Solution:-
Open Visual studio and create a new console application named as LINQEFConsoleApp and the Entity Framework Library using the Nuget Package Manager to the application.

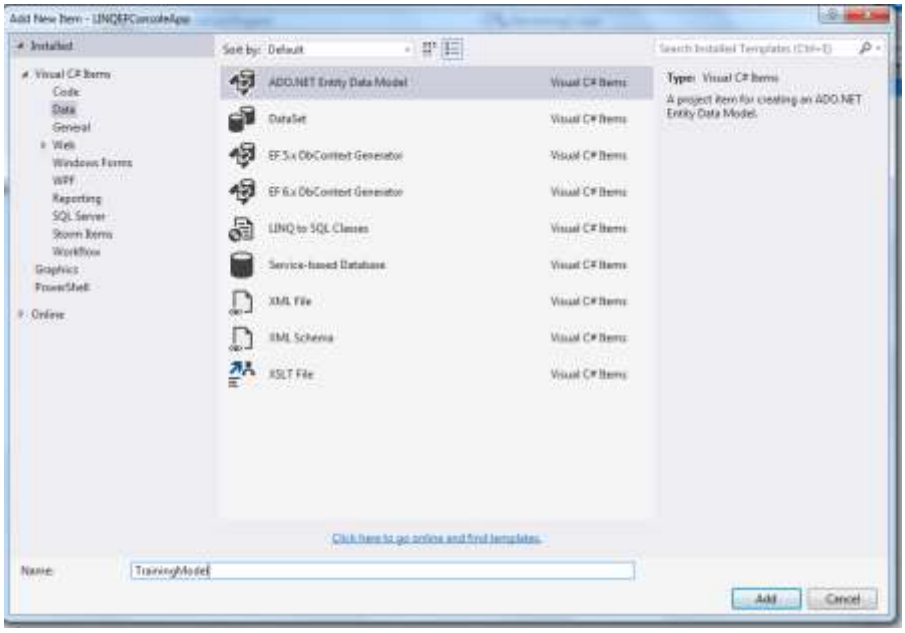




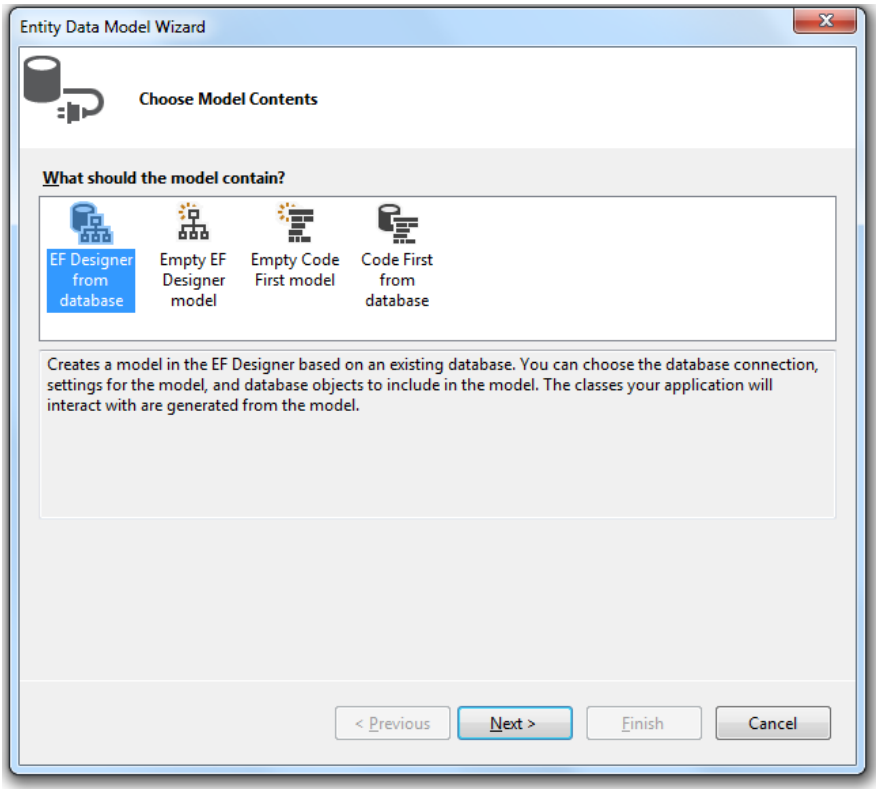
Now we have to create a Entity Data Model from an existing database.
Right click on the project in the solution explorer and select Add -> New Item



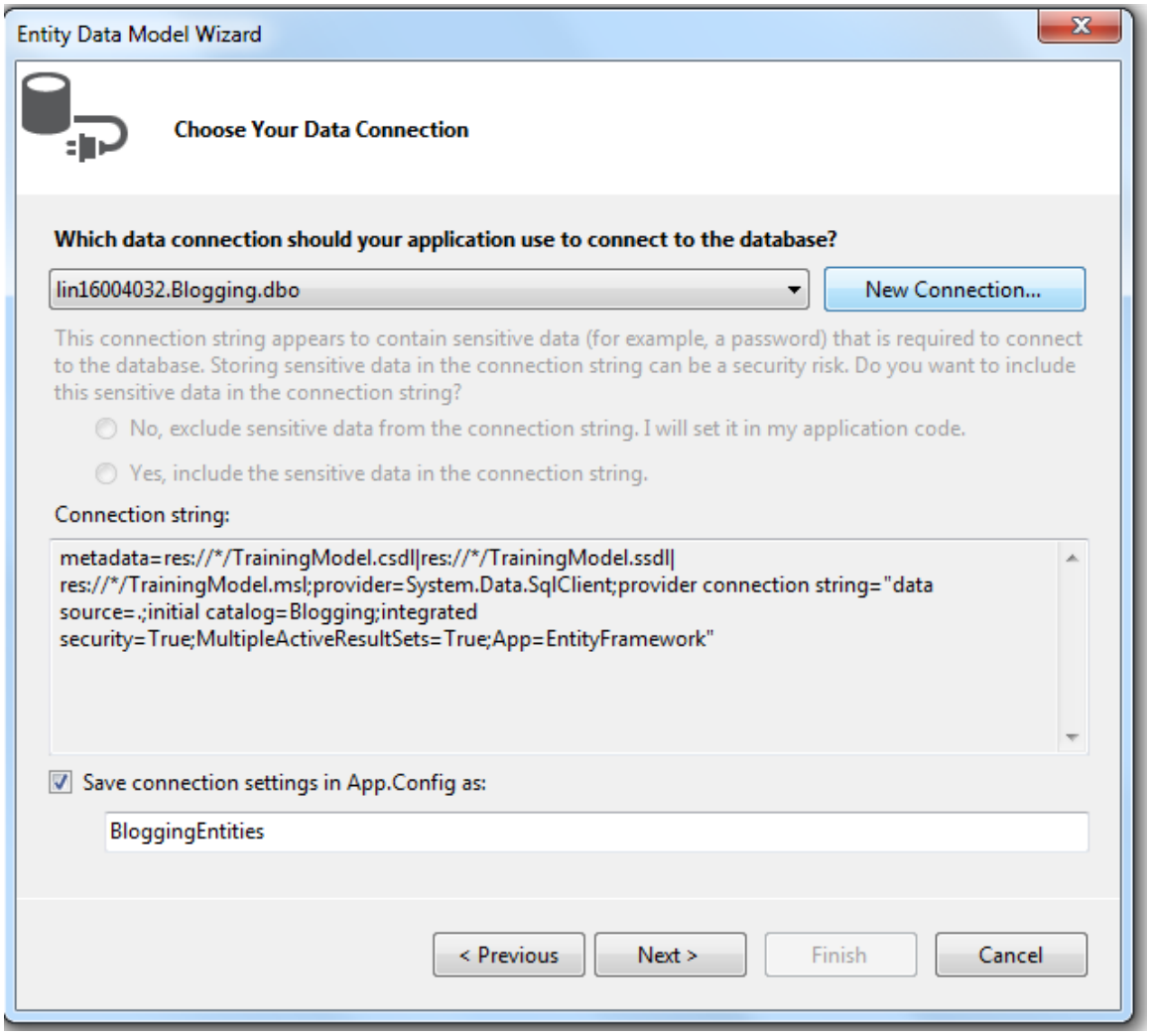
In the new Item Dialog Box select ADO.Net Entity Data Model and name it as Training Model



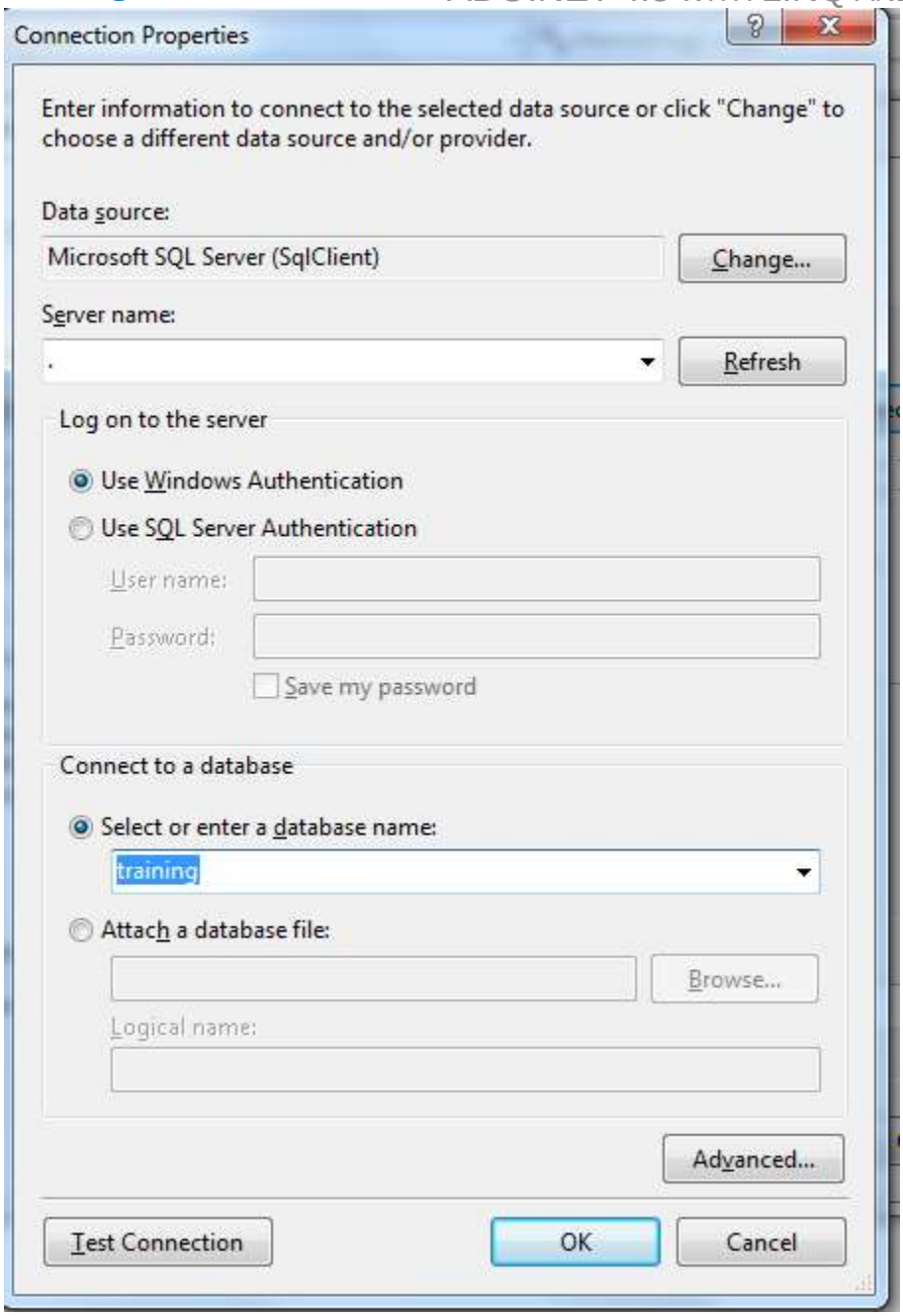
In the Entity Data Model Wizard select EF Designer from database and click on Next and configure the datasource.

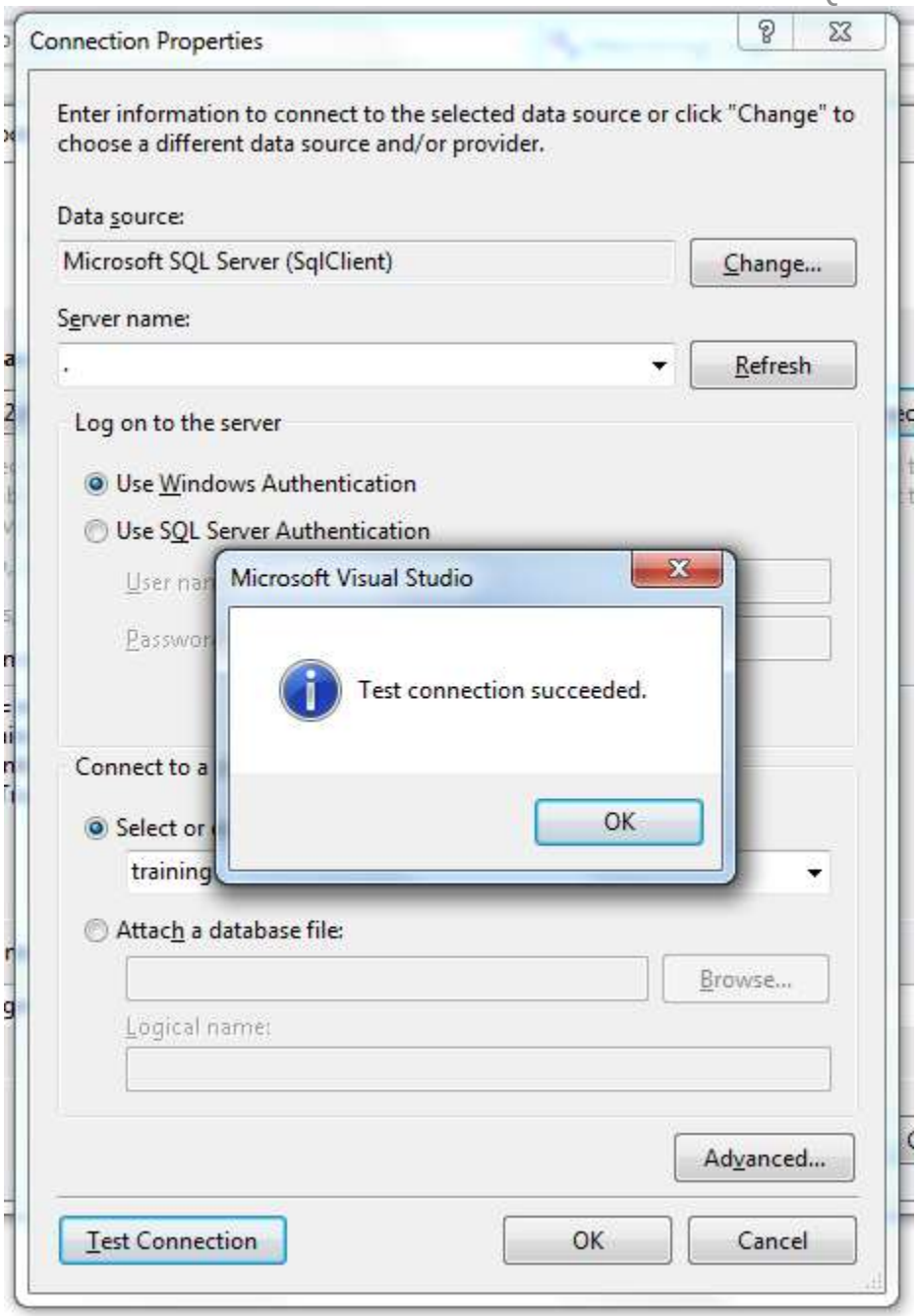


Now Click on New Connection in the Choose Your Data source option

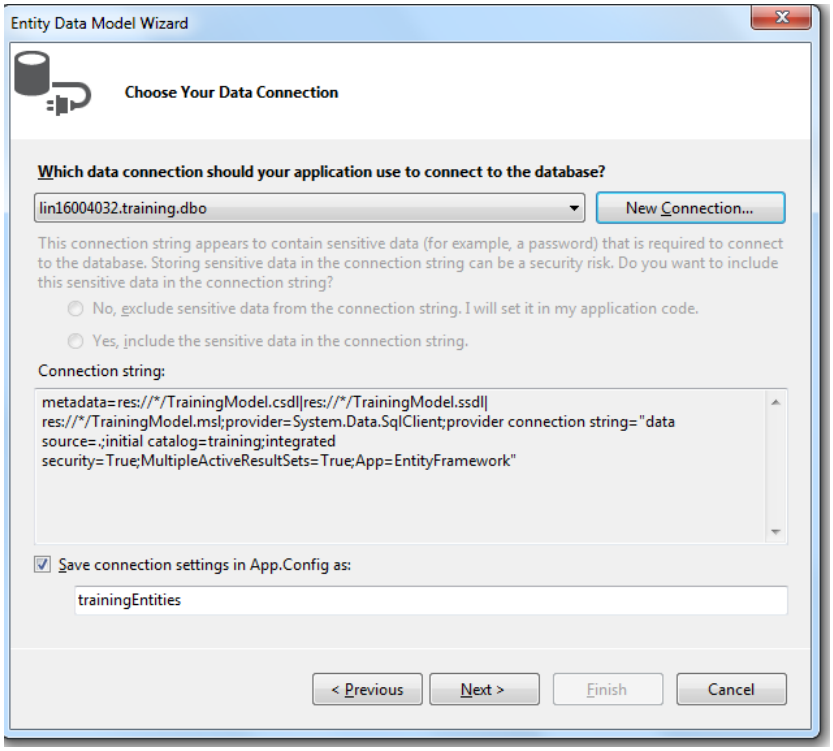


In the connection properties windows Provide the sql server name , authentication type and database name click on Test Connection to test the connection

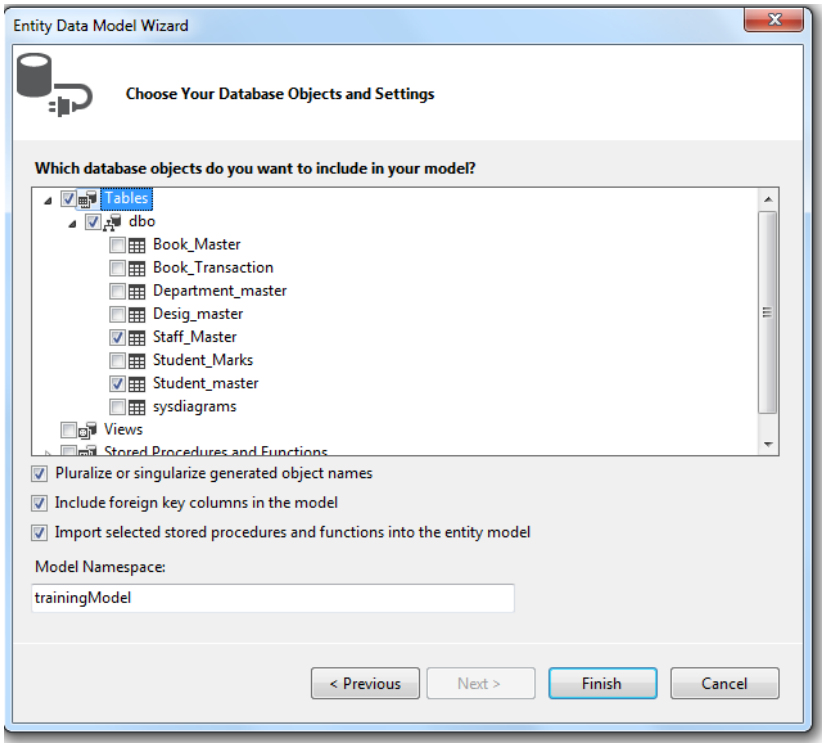




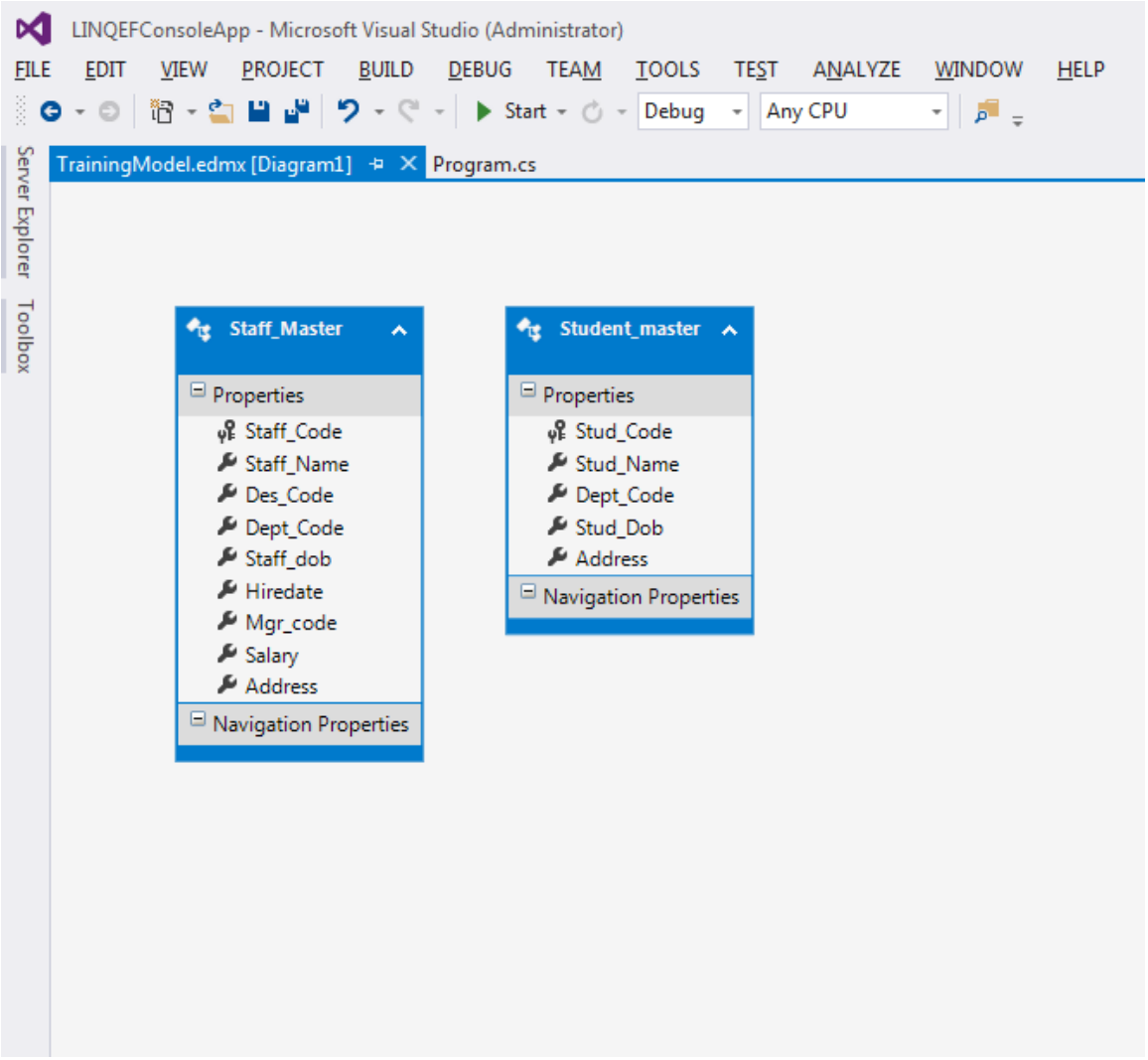
Click on OK



Now click on next to select the database objects which will be part of Entity Data Model.



Tick the Staff_Master and Student_Master Table and click on Finish this will add the entities to the model



Now we have to write LINQ query against the model for reading the data

Write Linq queries for the following.

1) To display staff details write the following query

```
static void Main(string[] args)
{
    trainingEntities context = new trainingEntities();

    var query = from staff in context.Staff_Master
                select staff;

    //Displaying details from Staff_Master Entity
    foreach (Staff_Master s in query)
    {
        Console.WriteLine("Staff Code= {0},Name = {1},HireDate = {2}",
                          s.Staff_Code,s.Staff_Name,s.Hiredate);
    }
}
```

2) To display list of employee whose salary is more than 30000

```
static void Main(string[] args)
{
    trainingEntities context = new trainingEntities();

    var query = from staff in context.Staff_Master
                where staff.Salary >30000
                select staff;


    foreach (Staff_Master s in query)
    {
        Console.WriteLine("Staff Code= {0},Name = {1},Salary = {2}",
                          s.Staff_Code,s.Staff_Name,s.Salary);
    }
}
```

Perform the following query by yourself

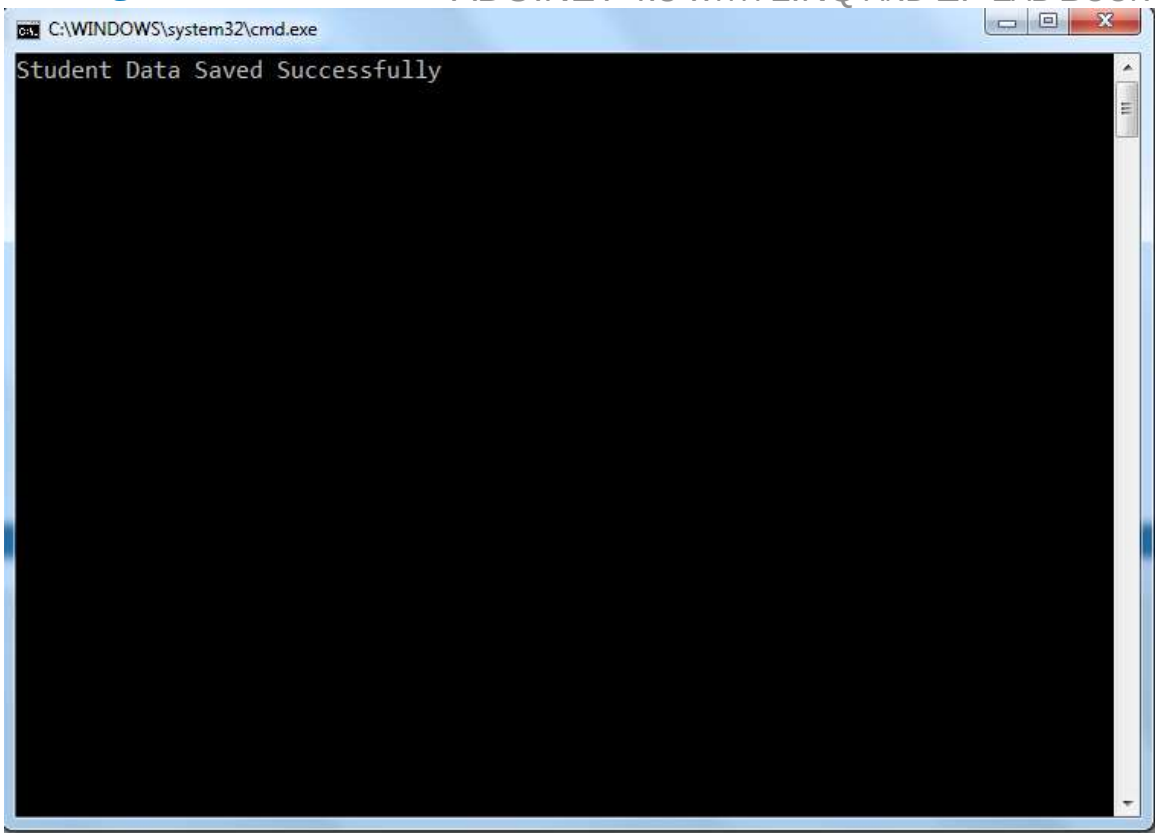
3) Display the list of student where city is not null
4) Display the list of student which includes Student name, department and date of birth
5) Display count of total student belonging to Bangalore
6) Display list of employees whose salary is more than the average salary of the employee.

Data Manipulation:-

Now we will CRUD operation on the model that we have create . we will use Student_Master Entity.
To ADD a record to the student master entity write the following code

```
static void Main(string[] args)
{
    //Intializing Object context
    trainingEntities context = new trainingEntities();

    //Initializing a student object
    Student_master student = new Student_master
    {
        Stud_Code=1055,
        Stud_Name="Suresh M",
        Dept_Code=10,
        Stud_Dob=Convert.ToDateTime("08/08/1985"),
        Address="Mumbai"
    };

    //Adding the student object to EntitySet
    context.Student_master.Add(student);

    //Saving Chnages to Database
    context.SaveChanges();
    Console.WriteLine("Student Data Saved Successfully");
    Console.ReadLine();
}
```

Output :-



SQLQuery11.sql - L...CORP\vijvishw (52))

```
/****** Script for SelectTopNRows command from SSMS  ******/
SELECT TOP 1000 [Stud_Code]
      ,[Stud_Name]
      ,[Dept_Code]
      ,[Stud_Dob]
      ,[Address]
  FROM [training].[dbo].[Student_master]
```

100 %

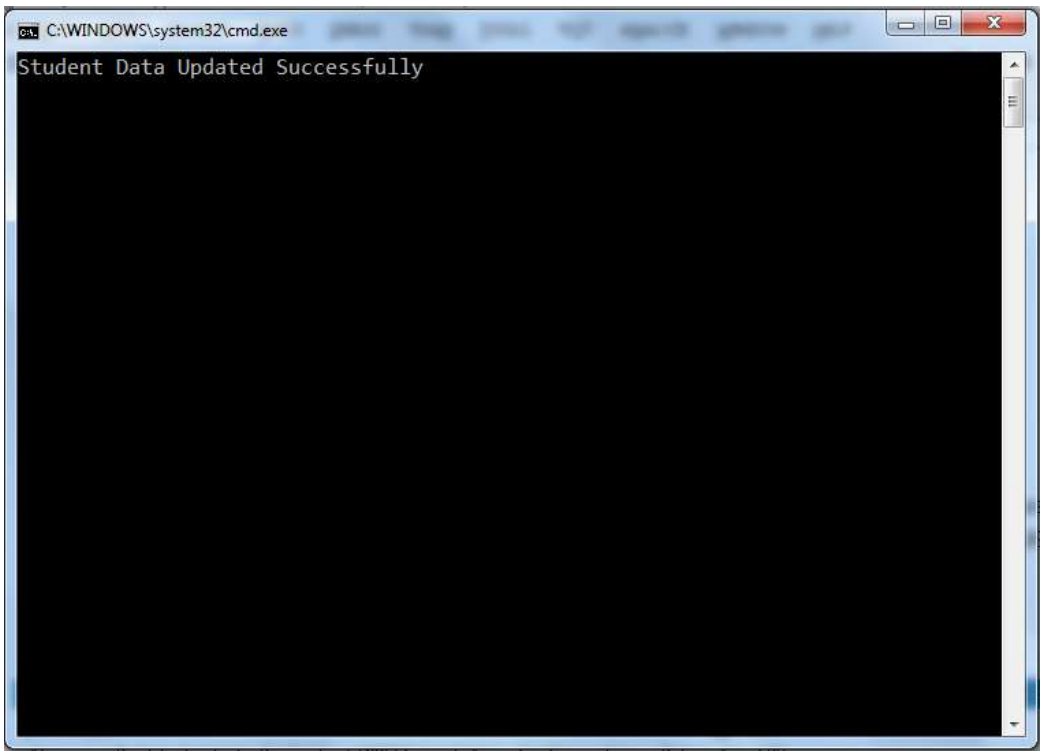
Results | Messages

	Stud_Code	Stud_Name	Dept_Code	Stud_Dob	Address
1	1001	Amit	10	1980-01-11 00:00:00.000	chennai
2	1002	Ravi	10	1981-11-01 00:00:00.000	New Delhi
3	1003	Ajay	20	1982-01-13 00:00:00.000	NULL
4	1004	Raj	30	1979-01-14 00:00:00.000	Mumbai
5	1005	Arvind	40	1983-01-15 00:00:00.000	Bangalore
6	1006	Rahul	50	1981-01-16 00:00:00.000	Delhi
7	1007	Mehul	20	1982-01-17 00:00:00.000	NULL
8	1008	Dev	10	1981-03-11 00:00:00.000	NULL
9	1009	Vijay	30	1980-01-19 00:00:00.000	NULL
10	1010	Rajat	40	1980-01-20 00:00:00.000	Bangalore
11	1011	Sunder	50	1980-01-21 00:00:00.000	NULL
12	1012	Rajesh	30	1980-01-22 00:00:00.000	NULL
13	1013	Anil	20	1980-01-23 00:00:00.000	Chennai
14	1014	Sunil	10	1985-02-15 00:00:00.000	NULL
15	1015	Kapil	40	1981-03-18 00:00:00.000	NULL
16	1016	Ashok	40	1980-11-26 00:00:00.000	NULL
17	1017	Ramesh	30	1980-12-27 00:00:00.000	NULL
18	1018	Amit Raj	50	1980-09-28 00:00:00.000	New Delhi
19	1019	Ravi Raj	50	1981-05-29 00:00:00.000	New Delhi
20	1020	Amrit	10	1980-11-11 00:00:00.000	NULL
21	1021	Sumit	20	1980-01-01 00:00:00.000	Chennai
22	1055	Suresh M	10	1985-08-08 00:00:00.000	Mumbai

To Update a Record Add the following code


```
static void Main(string[] args)
{
    //Intializing Object context
    trainingEntities context = new trainingEntities();

    //Acquiring the Object which need to be updated
    Student_master student = (from s in context.Student_master.Where
                                  (s => s.Stud_Code == 1020)
                              select s).FirstOrDefault();

    if (student != null)
    {
        student.Address = "Bangalore";
        context.SaveChanges();
        Console.WriteLine("Student Data Updated Successfully");
    }
    else
    {
        Console.WriteLine("Cannot Update Student\nStudent Not available");
    }
}
```

Output :-



SQLQuery11.sql - L...CORP\vijvishw (52))*

```
/****** Script for SelectTopNRows command from SSMS  ******/
SELECT TOP 1000 [Stud_Code]
      ,[Stud_Name]
      ,[Dept_Code]
      ,[Stud_Dob]
      ,[Address]
  FROM [training].[dbo].[Student_master]
  Where [Stud_Code]=1020
```

100 %

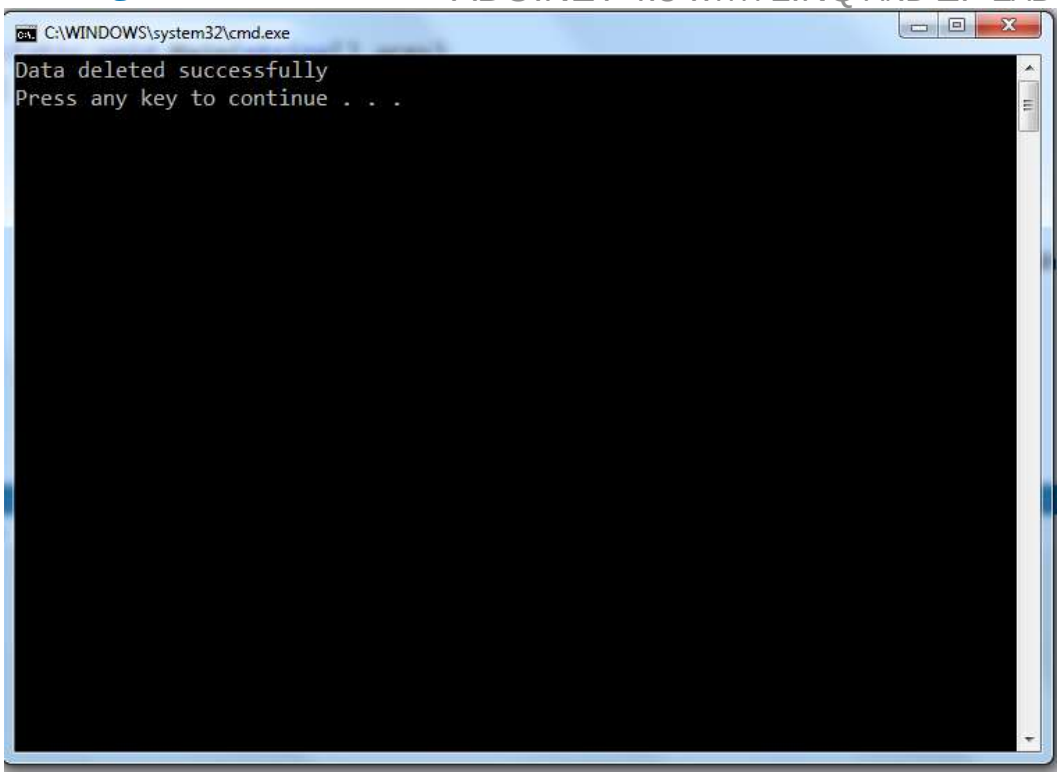
Results | Messages

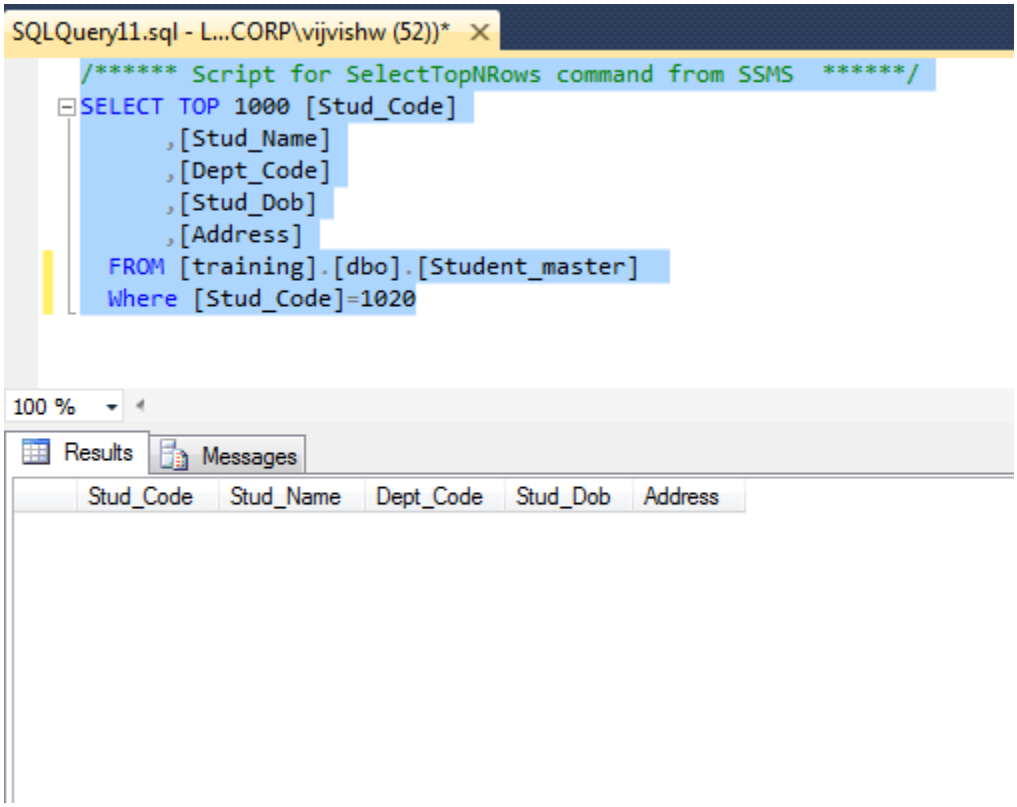
	Stud_Code	Stud_Name	Dept_Code	Stud_Dob	Address
1	1020	Amrit	10	1980-11-11 00:00:00.000	Bangalore

To Delete a Record Add the following code

```
static void Main(string[] args)
{
    //Intializing Object context
    trainingEntities context = new trainingEntities();

    //Acquiring the Object which need to be Deleted
    Student_master studentToDelete = (from s in context.Student_master.Where
                                      (s => s.Stud_Code == 1020)
                                    select s).FirstOrDefault();

    if (studentToDelete != null)
    {
        //Removing the record from the entity set
        context.Student_master.Remove(studentToDelete);
        context.SaveChanges();
        Console.WriteLine("Data deleted successfully");
    }
    else
    {
        Console.WriteLine("Cannot delete Student\nStudent Not available");
    }
}
```

Output:-





To-Do Assignments

1) Create a Console application to perform CRUD operation . You have to perform following step.
    a) Create a table named Employee which will have the following fields
        ID
        Name
        DOB
        DOJ
        Designation
        Salary
    b) Add a Entity Data Model to project which will include above mentioned Entity.
    c) Using LINQ to Entities write the following functionality and execute them
        i. Add a Employee details
        ii. Updating a Employee details
        iii. Searching for an Employee based on It ID
        iv. Deleteing an employee.

