<https://www.entityframeworktutorial.net/what-is-entityframework.aspx>

Batch.cs

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace CodeFiratApproach.Models

{

//[Table("tbl\_batches")]

class Batch

{

//[Key]

public int Code { get; set; }

public string Name { get; set; }

public int Strength { get; set; }

public DateTime StartDate { get; set; }

//Foreign key for Course

public int CourseId { get; set; }

public Course Course{ get; set; }

}

}

BatchDbContext.cs

using System;

using System.Collections.Generic;

using System.Data.Entity;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace CodeFiratApproach.Models

{

class BatchDbContext : DbContext

{

public BatchDbContext() { }

public DbSet<Batch> Batches { get; set; }

protected override void OnModelCreating(DbModelBuilder modelBuilder)

{

modelBuilder.Entity<Batch>()

.ToTable("tblBatch");

modelBuilder.Entity<Batch>().HasKey<int>(s => s.Code);

modelBuilder.Entity<Batch>()

.HasRequired<Course>(s => s.Course)

.WithMany(g => g.Batches)

.HasForeignKey<int>(s => s.CourseId);

}

}

}  
App.config.file

<connectionStrings>

<add name="BatchDbContext" connectionString="data source=admivm\SQLEXPRESS;initial catalog=StudentDb;user id=sa;password=pass@123" providerName="System.Data.SqlClient"/>

</connectionStrings>

Program.cs file

using CodeFiratApproach.Models;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace CodeFiratApproach

{

class Program

{

static void Main(string[] args)

{

string choice = "y";

while (choice == "y")

{

Console.WriteLine("1. List all Records");

Console.WriteLine("2. Insert Record");

Console.WriteLine("3. Update Record");

Console.WriteLine("4. Delete Records");

Console.WriteLine("5. Search Record");

Console.WriteLine("Enter Choice");

int ch = int.Parse(Console.ReadLine());

switch (ch)

{

case 1: GetStudents(); break;

case 2:

{

Console.WriteLine("Enter Code");

int Code = int.Parse(Console.ReadLine());

Console.WriteLine("Enter Name");

string name = Console.ReadLine();

Console.WriteLine("Enter Course");

string course = Console.ReadLine();

Console.WriteLine("Enter Strength");

int strength = int.Parse(Console.ReadLine());

InsertRecord(Code, name, course, strength); break;

}

case 3:

{

Console.WriteLine("Enter Code for which to edit Record");

int Code = int.Parse(Console.ReadLine());

Console.WriteLine("Enter Course");

string course = Console.ReadLine();

Console.WriteLine("Enter Strength");

int strength = int.Parse(Console.ReadLine());

UpdateRecord(Code, course, strength); break;

}

case 4:

{

Console.WriteLine("Enter Code for which to delete Record");

int Code = int.Parse(Console.ReadLine());

DeleteBatch(Code); break;

}

case 5:

{

Console.WriteLine("Enter Code for which to find Record");

int Code = int.Parse(Console.ReadLine());

GetBatch(Code); break;

}

}

Console.WriteLine("Do yu want to repeat tge process");

choice = Console.ReadLine();

}

}

public static void GetStudents()

{

BatchDbContext db = new BatchDbContext();

// Get Records

// LINQ

var batches = db.Batches.ToList();

if (batches.Count > 0)

{

foreach (var batch in batches)

{

Console.WriteLine(batch.Code + " " + batch.Name + " " + batch.Course);

}

}

else

{

Console.WriteLine("There are no records");

}

}

public static void InsertRecord(int Code, string name, string course, int strength)

{

BatchDbContext db = new BatchDbContext();

Batch batch = new Batch()

{

Code = Code,

Name = name,

// Course = course,

Strength = strength

};

db.Batches.Add(batch);

db.SaveChanges();

}

public static void UpdateRecord(int Code, string course, int strength)

{

BatchDbContext db = new BatchDbContext();

var batch = db.Batches.FirstOrDefault(x => x.Code == Code);

if (batch != null)

{

foreach (Batch temp in db.Batches)

{

if (temp.Code == Code)

{

// temp.Course = course;

temp.Strength = strength;

}

}

db.SaveChanges();

}

}

public static void DeleteBatch(int Code)

{

BatchDbContext db = new BatchDbContext();

var batch = db.Batches.FirstOrDefault(x => x.Code == Code);

if (batch != null)

{

db.Batches.Remove(batch);

db.SaveChanges();

}

}

public static void GetBatch(int Code)

{

BatchDbContext db = new BatchDbContext();

var batch = db.Batches.FirstOrDefault(x => x.Code == Code);

if (batch != null)

{

Console.WriteLine(batch.Code + " " + batch.Name + " " + batch.Course);

}

}

}

}