EF CORE ASSIGNMENTS Day02

Puneeth Puligundla 20249

1. Code First Approach

using CodeFirstSample.Data;

using CodeFirstSample.Models;

using Microsoft.EntityFrameworkCore;

using System;

using System.Linq;

using Microsoft.EntityFrameworkCore.Metadata;

namespace CodeFirstSample

{

class Program

{

static void Main(string[] args)

{

//AddNewBook();

//RemoveBook(4);

UpdateBookAuthor(1);

//ConflictResolve();

}

/// <summary>

/// Add Book Using CodeFirst

/// </summary>

private static void AddNewBook()

{

var ctx = new BookContext();

Book b1 = new Book();

b1.BookID = 4;

b1.BookName = "F#";

b1.Author = "R D Sharma";

b1.Price = 3000;

try

{

ctx.Books.Add(b1);

ctx.SaveChanges();

Console.WriteLine("Book added Successfully");

}

catch(Exception e)

{

Console.WriteLine(e.InnerException.ToString());

}

}

/// <summary>

/// Delete Using Code First

/// </summary>

///

private static void RemoveBook(int bid)

{

var ctx = new BookContext();

var book = ctx.Books.Where(b => b.BookID == bid).SingleOrDefault();

try

{

ctx.Books.Remove(book);

ctx.SaveChanges();

Console.WriteLine("Book with ID " + bid+" has been deleted From MyBook Table");

}

catch(Exception e)

{

Console.WriteLine(e.InnerException.ToString());

}

}

///<summary>

///Update Record

/// </summary>

private static void UpdateBookAuthor(int bid)

{

var ctx = new BookContext();

var book = ctx.Books.Where(b => b.BookID == bid).SingleOrDefault();

book.Author = "William Stalling";

try

{

ctx.Books.Update(book);

ctx.SaveChanges();

Console.WriteLine("Changes Saved");

}

catch(Exception e)

{

Console.WriteLine(e.InnerException.ToString());

}

}

}

}

2. Customer and Orders

using CodeFirstSample.Data;

using CodeFirstSample.Models;

using Microsoft.EntityFrameworkCore;

using System;

using System.Linq;

using Microsoft.EntityFrameworkCore.Metadata;

using System.Collections.Generic;

namespace CodeFirstSample

{

class Program

{

static void Main(string[] args)

{

//AddNewCustAndOrder();

//AddNewCustAndOrderViaCust();

//AddNewOrderForExistingCust(20249);

//GetAllCustomersWithOrder\_EagerLoading();

//GetAllCustomersWithOrder\_ExplicitLoading();

//RemoveOrderByOrderId(15);

//RemoveOrderByCustID(20249);

GetOrdersByLinq();

}

private static void GetOrdersByLinq()

{

var ctx = new BookContext();

var ordGrp = from ord in ctx.Orders

group ord by ord.cust.ID into g

select new { customer = g.Key, count = g.Count() };

foreach (var item in ordGrp)

{

Console.WriteLine(item.customer+" "+item.count.ToString());

}

}

/// <summary>

/// Add Customer And Order

/// </summary>

private static void AddNewCustAndOrder()

{

var ctx = new BookContext();

Customer newCust = new Customer();

newCust.ID = 20249;

newCust.Name = "Puneeth";

Order ord = new Order();

ord.Order\_ID = 10;

ord.Amount = 250;

ord.OrderDate = DateTime.Now;

ord.cust = newCust;

try

{

ctx.Orders.Add(ord);

ctx.SaveChanges();

Console.WriteLine("Order and Customer added Successfully");

}

catch(Exception e)

{

Console.WriteLine(e.InnerException.ToString());

}

}

/// <summary>

/// Add More than one Order Via Customer

/// </summary>

private static void AddNewCustAndOrderViaCust()

{

var ctx = new BookContext();

Customer newCust = new Customer();

newCust.ID = 20239;

newCust.Name = "Aditya";

Order ord1 = new Order();

ord1.Order\_ID = 12;

ord1.Amount = 250;

ord1.OrderDate = DateTime.Now;

Order ord2 = new Order();

ord2.Order\_ID = 15;

ord2.Amount = 700;

ord2.OrderDate = DateTime.Now;

List<Order> myOrds = new List<Order>();

myOrds.Add(ord1);

myOrds.Add(ord2);

newCust.Orders = myOrds;

try

{

ctx.Customers.Add(newCust);

ctx.SaveChanges();

Console.WriteLine("Order and Customer added Successfully");

}

catch (Exception e)

{

Console.WriteLine(e.InnerException.ToString());

}

}

///<summary>

/// Add Order For Existing Customers

/// </summary>

private static void AddNewOrderForExistingCust(int cid)

{

var ctx = new BookContext();

Order ord1 = new Order();

ord1.Order\_ID = 23;

ord1.Amount = 700;

ord1.OrderDate = DateTime.Now;

var eCust = ctx.Customers.Where(c => c.ID == cid).SingleOrDefault();

ord1.cust = eCust;

try

{

ctx.Orders.Add(ord1);

ctx.SaveChanges();

Console.WriteLine("Order assigned to Customer Successfully");

}

catch (Exception e)

{

Console.WriteLine(e.InnerException.ToString());

}

}

///<summary>

/// Remove Order For a Given Order Id

/// </summary>

private static void RemoveOrderByOrderId(int oid)

{

var ctx = new BookContext();

var order = ctx.Orders.Where(o => o.Order\_ID == oid).SingleOrDefault();

try

{

ctx.Orders.Remove(order);

ctx.SaveChanges();

Console.WriteLine("Order with oid "+oid+" is removed successfully");

}

catch(Exception ex)

{

Console.WriteLine(ex.InnerException.ToString());

}

}

///<summary>

/// Remove Order For a Given Cust Id

/// </summary>

private static void RemoveOrderByCustID(int cid)

{

var ctx = new BookContext();

var customerID = ctx.Customers.SingleOrDefault(c => c.ID == cid);

var ord = ctx.Orders;

try

{

foreach (var item in ord)

{

if (customerID == item.cust)

{

ctx.Orders.Remove(item);

}

}

ctx.SaveChanges();

Console.WriteLine("Deleted the orders for given " + cid);

}

catch(Exception ex)

{

Console.WriteLine(ex.InnerException.ToString());

}

}

///<summary>

/// Eager Loading

/// </summary>

///

private static void GetAllCustomersWithOrder\_EagerLoading()

{

//Eager Loading means that the related data is loaded from db

//as a part of the initial Query

var ctx = new BookContext();

try

{

//var customers=ctx.Customers.Include("Orders");

var customers = ctx.Customers.Include(o => o.Orders);

foreach (var item in customers)

{

Console.WriteLine(item.Name);

Console.WriteLine("================");

foreach (var item1 in item.Orders)

{

Console.WriteLine(item1.Amount.ToString());

}

Console.WriteLine("================");

}

}

catch(Exception e)

{

Console.WriteLine(e.InnerException.ToString());

}

}

/// <summary>

/// Explicit Loading

/// </summary>

private static void GetAllCustomersWithOrder\_ExplicitLoading()

{

//Explicit Loading Means that the related data is

//explicitly loaded from the database at a later time.

//Needs MultipleActiveResultSet to be set to True in the connect string in the context class

var ctx = new BookContext();

try

{

//var customers=ctx.Customers.Include("Orders");

var customers = ctx.Customers;

foreach (var customer in customers)

{

Console.WriteLine(customer.Name);

Console.WriteLine("================");

ctx.Entry(customer).Collection(o => o.Orders).Load();

foreach (var order in customer.Orders)

{

Console.WriteLine(order.Amount.ToString()+" "+order.OrderDate.ToString());

}

Console.WriteLine("================");

}

}

catch (Exception e)

{

Console.WriteLine(e.InnerException.ToString());

}

}

}

}