МГТУ им. Баумана  
Факультет «Информатика и системы управления»  
Кафедра «Автоматизированные системы обработки информации и управления»  
Дисциплина «Базовые компоненты интернет-технологий»

Отчёт по лабораторной работе №7



Описание задания:

Разработать программу, реализующую работу с LINQ to Objects. В качестве примера используйте проект «SimpleLINQ» из примера «Введение в LINQ».

1. Программа должна быть разработана в виде консольного приложения на языке C#.
2. Создайте класс «Сотрудник», содержащий поля:
   * ID записи о сотруднике;
   * Фамилия сотрудника;
   * ID записи об отделе.
3. Создайте класс «Отдел», содержащий поля:
   * ID записи об отделе;
   * Наименование отдела.
4. Предполагая, что «Отдел» и «Сотрудник» связаны соотношением один-ко-многим разработайте следующие запросы:
   * Выведите список всех сотрудников и отделов, отсортированный по отделам.
   * Выведите список всех сотрудников, у которых фамилия начинается с буквы «А».
   * Выведите список всех отделов и количество сотрудников в каждом отделе.
   * Выведите список отделов, в которых у всех сотрудников фамилия начинается с буквы «А».
   * Выведите список отделов, в которых хотя бы у одного сотрудника фамилия начинается с буквы «А».
5. Создайте класс «Сотрудники отдела», содержащий поля:
   * ID записи о сотруднике;
   * ID записи об отделе.
6. Предполагая, что «Отдел» и «Сотрудник» связаны соотношением много-ко-многим с использованием класса «Сотрудники отдела» разработайте следующие запросы:
   * Выведите список всех отделов и список сотрудников в каждом отделе.
   * Выведите список всех отделов и количество сотрудников в каждом отделе.

Текст программы:

Класс Program:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LINQ\_Basics {

class Program {

/// <summary>

/// Тип сущности "Сотрудник"

/// </summary>

public class Employee {

public int EmployeeID; //Primary key

public string EmployeeLastName;

public int DepartmentID; //Foreign key

public Employee(int eID, string name, int dID) {

EmployeeID = eID;

EmployeeLastName = name;

DepartmentID = dID;

}

public override string ToString() {

return "EmployeeID: " + EmployeeID.ToString() + "; EmployeeLastName: " + EmployeeLastName + "; DepartmentID: " + DepartmentID.ToString();

}

}

/// <summary>

/// Тип сущности "Отдел"

/// </summary>

public class Department {

public int DepartmentID; //Primary key

public string DepartmentName;

public Department(int dID, string name) {

DepartmentID = dID;

DepartmentName = name;

}

public override string ToString() {

return "DepartmentID: " + DepartmentID.ToString() + "; DepartmentName: " + DepartmentName;

}

}

/// <summary>

/// Имплементация связи многие-ко-многим между типами сущностей "Сотрудник" и "Отдел"

/// </summary>

public class DepartmentEmployee {

public int EmployeeID; //Foreign key

public int DepartmentID; // Foreign key

public DepartmentEmployee(int eID, int dID) {

EmployeeID = eID;

DepartmentID = dID;

}

}

/// <summary>

/// Эзкемпляры сущности "Сотрудник"

/// </summary>

static List<Employee> Employees = new List<Employee>() {

new Employee(1, "Абрамова", 3),

new Employee(2, "Александров", 3),

new Employee(3, "Зяблицева", 2),

new Employee(4, "Ахатова", 1),

new Employee(5, "Архипова", 4),

new Employee(6, "Длютров", 3),

new Employee(7, "Просветов", 2),

new Employee(8, "Безверхний", 1),

new Employee(9, "Агапов", 3),

new Employee(10, "Андреева", 4)

};

/// <summary>

/// Экземпляры сущности "Отдел"

/// </summary>

static List<Department> Departments = new List<Department> {

new Department(1, "Отдел кадров"),

new Department(2, "Бухгалтерия"),

new Department(3, "Администрация"),

new Department(4, "Маркетинг"),

};

/// <summary>

/// Эзкемпляры связи "Отдел"-"Сотрудник" вида многие-ко-многим

/// </summary>

static List<DepartmentEmployee> DepartmentEmployees = new List<DepartmentEmployee> {

new DepartmentEmployee(1, 3),

new DepartmentEmployee(2, 3),

new DepartmentEmployee(3, 2),

new DepartmentEmployee(4, 1),

new DepartmentEmployee(5, 4),

new DepartmentEmployee(6, 3),

new DepartmentEmployee(7, 2),

new DepartmentEmployee(8, 1),

new DepartmentEmployee(9, 3),

new DepartmentEmployee(10, 4),

new DepartmentEmployee(1, 2),

new DepartmentEmployee(2, 2),

new DepartmentEmployee(1, 4),

new DepartmentEmployee(4, 2),

new DepartmentEmployee(4, 3),

new DepartmentEmployee(5, 2),

new DepartmentEmployee(6, 1),

new DepartmentEmployee(10, 2),

new DepartmentEmployee(8, 2),

new DepartmentEmployee(8, 4),

new DepartmentEmployee(3, 3)

};

static void Main(string[] args) {

Console.WriteLine("ОТДЕЛ - СОТРУДНИК: 1 - М\n");

Console.WriteLine("\nСотрудники по отделам: \n");

var q1 = from x in Departments

join y in Employees on x.DepartmentID equals y.DepartmentID into g

orderby x.DepartmentID ascending

select new { Dep = x, Emps = g.OrderBy(g => g.EmployeeLastName) };

foreach (var x in q1) {

Console.WriteLine(x.Dep + ":");

foreach (var y in x.Emps) {

Console.WriteLine(" " + y);

}

}

Console.WriteLine("\nЧисло сотрудников в отделах:\n");

var q2 = from x in Departments

join y in Employees on x.DepartmentID equals y.DepartmentID into g

select new { Dep = x.DepartmentName, Emps = g.Count() };

foreach (var x in q2) {

Console.WriteLine(x.Dep + ": {0}", x.Emps);

}

Console.WriteLine("\nСотрудники с фамилией на А:\n");

var q3 = from x in Employees

where x.EmployeeLastName[0] == 'А' //Условие

orderby x.EmployeeLastName ascending //Сортировка

select x;

foreach (var x in q3) {

Console.WriteLine(x);

}

Console.WriteLine("\nОтделы, в которых фамилии всех сотрудников начинаются на \"А\":\n");

var q4 = from x in Departments

join y in Employees on x.DepartmentID equals y.DepartmentID into g

where g.All(g => g.EmployeeLastName[0] == 'А')

select new { Dep = x.DepartmentName, Emps = g };

foreach (var x in q4) {

Console.WriteLine(x.Dep + ":");

foreach (var y in x.Emps) {

Console.WriteLine(" " + y);

}

}

Console.WriteLine("\nОтделы, в которых есть сотрудники с фамилией на \"А\":\n");

var q5 = from x in Departments

join y in Employees on x.DepartmentID equals y.DepartmentID into g

where g.Any(g => g.EmployeeLastName[0] == 'А')

select new { Dep = x.DepartmentName, Emps = g };

foreach (var x in q5) {

Console.WriteLine(x.Dep + ":");

foreach (var y in x.Emps) {

Console.WriteLine(" " + y);

}

}

Console.WriteLine("\n\nОТДЕЛ - СОТРУДНИК: М - М\n");

Console.WriteLine("\nСотрудники по отделам:\n");

var q6 = from x in Departments

join y in DepartmentEmployees on x.DepartmentID equals y.DepartmentID into temp1

from t1 in temp1

join z in Employees on t1.EmployeeID equals z.EmployeeID into temp2

from t2 in temp2

orderby t1.DepartmentID, t2.EmployeeLastName ascending

group t2 by x into g

select g;

foreach (var x in q6) {

Console.WriteLine(x.Key + ":");

foreach (var y in x) {

Console.WriteLine(" " + y);

}

}

Console.WriteLine("\nЧисло сотрудников в отделах:\n");

var q7 = from x in Departments

join y in DepartmentEmployees on x.DepartmentID equals y.DepartmentID into temp1

from t1 in temp1

join z in Employees on t1.EmployeeID equals z.EmployeeID into temp2

from t2 in temp2

group t2 by x.DepartmentName into g

select new { Dep = g.Key, Emps = g.Count() };

foreach (var x in q7) {

Console.WriteLine(x.Dep + ": {0}", x.Emps);

}

Console.ReadLine();

}

}

}

Экранные формы:



