МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

КРЕМЕНЧУЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ІМЕНІ МИХАЙЛА ОСТРОГРАДСЬКОГО

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ЕЛЕКТРИЧНОЇ ІНЖЕНЕРІЇ

ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра автоматизації та інформаційних систем

Звіт з лабораторної роботи №2

З навчальної дисципліни «Організація баз даних та знань»

Виконав: студент групи КН-19-1

Данілочкін Данііл

Перевірив: старший викладач

Васильєв Денис Олегович

КРЕМЕНЧУК 2022

**Тема:** **LINQ – язык интегрированных запросов**

Завдання: создать запросы, где необходимо выполнить операции объединения, конкатенации, преобразования, группировки, упорядочивания, разделения на части. Для выполнения операции запроса требуется создать источник данных, определить выражение запроса и выполнить запрос в операторе foreach.

**Хід роботи**

Код програми:

static void Main(string[] args)

{

var employees = new[] {

new {surname = "Іванець", name = "Микола", secondname = "Сергійович", position = "бригадир маляр", salary = 12000, birthday = new DateTime(1970, 3, 28) },

new {surname = "Коношевич", name = "Ольга", secondname = "Володимирівна", position = "маляр", salary = 8000, birthday = new DateTime(1983, 6, 1) },

new {surname = "Коваль", name = "Мирослав", secondname = "Всеволодович", position = "бригадир покрівельників", salary = 15000, birthday = new DateTime(1978, 8, 14) },

new {surname = "Шмигаль", name = "Юрій", secondname = "Юрійович", position = "покрівельник", salary = 8000, birthday = new DateTime(1997, 10, 15) },

new {surname = "Заєць", name = "Микита", secondname = "Васильович", position = "покрівельник", salary = 8000, birthday = new DateTime(1985, 4, 30) },

new {surname = "Куцихвіст", name = "Данило", secondname = "Андрійович", position = "сантехнік", salary = 15000, birthday = new DateTime(1976, 7, 3) },

new {surname = "Пальчун", name = "Назар", secondname = "Миколаєвич", position = "електрик", salary = 12000, birthday = new DateTime(1980, 9, 20) },

new {surname = "Почечун", name = "Павло", secondname = "Некіфорович", position = "директор фірми", salary = 30000, birthday = new DateTime(1975, 5, 17) },

new {surname = "Некрасова", name = "Любов", secondname = "Іванівна", position = "бухгалтер", salary = 15000, birthday = new DateTime(1989, 11, 12) },

new {surname = "Підгірна", name = "Людмила", secondname = "Миколаєвна", position ="юрист", salary = 13000, birthday = new DateTime(1973, 5, 18) },

new {surname = "Непийпиво", name = "Наталія", secondname = "Петрівна", position = "спеціаліст по роботі з клієнтами", salary = 20000, birthday = new DateTime(1992, 7, 29) }

};

var emplSalary =

from e in employees

select e.salary;

Console.Write("Зартплати співробітників: ");

foreach (var t in emplSalary)

{

Console.Write(t + " ");

}

double averageSalary = emplSalary.Average();

int countEmployees = employees.Count();

int minSalary = emplSalary.Min();

int maxSalary = emplSalary.Max();

int summarySalary = emplSalary.Sum();

Console.WriteLine($"\nСередня зарплата {averageSalary}. Кількість працівників {countEmployees}. Мінімальна зарплата {minSalary}. Максимальна зарплата {maxSalary}. Сума всіх зарплат {summarySalary}");

var bregadirs =

from e in employees

where e.position.Contains("бригадир")

select e;

var directors =

from e in employees

where e.position.Contains("директор")

select e;

var BD = bregadirs.Concat(directors);

Console.WriteLine("\n Керівники: ");

foreach (var t in BD)

{

Console.WriteLine($"{t.surname} {t.name} {t.secondname} {t.position}");

}

IEnumerable<string> birthdays = employees.Cast<string>();

var arrayOfEmployees = employees.ToArray();

var dictionaryOfEmployees = employees.ToDictionary(e => e.surname + ' ' + e.name + ' ' + e.secondname);

var listOfEmployees = employees.ToList();

var employeesGroups = employees.GroupBy(e => (int)e.salary / 5000);

Console.WriteLine("\nПрацівники згруповані по зарплаті: ");

foreach (var t in employeesGroups)

{

Console.WriteLine($"\n{t.Key\*5000}-{(t.Key + 1)\*5000} грн");

foreach (var tt in t)

{

Console.WriteLine($"{tt.surname} {tt.name} {tt.secondname} {tt.position}");

}

}

var sortedEmployees =

from e in employees

orderby e.salary descending

select e;

Console.WriteLine("\nПрацівники відсортовані по зарплаті по зменшенню");

foreach (var t in sortedEmployees)

{

Console.WriteLine($"{t.surname} {t.name} {t.secondname} {t.position} {t.salary}");

}

var reverse = sortedEmployees.Reverse();

Console.WriteLine("\nПрацівники відсортовані по зарплаті по зростанню");

foreach (var t in sortedEmployees)

{

Console.WriteLine($"{t.surname} {t.name} {t.secondname} {t.position} {t.salary}");

}

var skip = employees.Skip(3);

Console.WriteLine("\nПрацівники без перших 3");

foreach (var t in skip)

{

Console.WriteLine($"{t.surname} {t.name}");

}

var skipWhile = employees.SkipWhile(e => e.salary > averageSalary);

Console.WriteLine("\nПрацівники з першого в кого зарплата вище за середню");

foreach (var t in skipWhile)

{

Console.WriteLine($"{t.surname } {t.name} {t.salary}");

}

var take = employees.Take(4);

Console.WriteLine("\nПерші 4 працівника");

foreach (var t in take)

{

Console.WriteLine($"{t.surname} {t.name}");

}

var takeWhile = employees.TakeWhile(e => e.salary < averageSalary);

Console.WriteLine("\nПрацівники до першого в кого зарплата більша за середню");

foreach (var t in takeWhile)

{

Console.WriteLine($"{t.surname} {t.name} {t.salary}");

}

Console.ReadKey();

}

Зображення, що містить текст

Автоматично згенерований опис

Рисунок 1 – результат роботи коду

Зображення, що містить текст

Автоматично згенерований опис

Рисунок 2 – результат роботи коду

**Висновок:** в результаті виконання лабораторної роботи було вивчено і протестовано на практиці знання з LINQ.

**Контрольні питання**

1. Что представляет собой LINQ?

Набор функций Visual Studio 2008, расширяющих мощные возможности запроса в синтаксисе языка C# и Visual Basic. LINQ представляет стандартные, легко изучаемые шаблоны для создания запросов и обновления данных.

1. Перечислите действия необходимые для выполнения запроса LINQ

- Получение источника данных.

- Создание запроса.

- Выполнение запроса.

1. Какие типы запросов можно сформировать с помощью LINQ.

Объединения, конкатенации, преобразования, группировки, упорядочивания, разделения на части.

1. Что представляет собой оператор запроса LINQ?

Метод, который преобразует последовательность. Оператор запроса никогда не изменяет входную последовательность, вместо этого он возвращает новую.

1. Какие есть формы синтаксиса LINQ?

Запросов и методов.

1. Ключевые слова синтаксиса запросов LINQ.

From, where, select, group, into, orderby, join, let, in, on, equals, by, ascending, descending

1. Что представляет собой переменная запроса LINQ?

Переменная, сохраняющая запрос вместо результатов запроса. Говоря точнее, переменная запроса всегда является перечислимым типом и производит последовательность элементов, когда она используется в итерации оператора foreach или прямом вызове ее методаIEnumerator.MoveNext.