LINQ

Language-Integrated Query (LINQ) is a powerful query language introduced with .Net 3.5 & Visual Studio 2008. LINQ can be used with C# or Visual Basic to query different data sources.

LINQ is nothing but the collection of extension methods for classes that implements IEnumerable and IQueryable interface.

Advantages of LINQ:

- **Familiar language:** Developers don't have to learn a new query language for each type of data source or data format.
- **Less coding:** It reduces the amount of code to be written as compared with a more traditional approach.
- **Readable code:** LINQ makes the code more readable so other developers can easily understand and maintain it.
- **Standardized way of querying multiple data sources:** The same LINQ syntax can be used to query multiple data sources.

LINQ Query Syntax

There are two basic ways to write a LINQ query to IEnumerable collection or IQueryable data sources.

1. Query Syntax or Query Expression Syntax
2. Method Syntax or Method extension syntax or Fluent

For all of Examples except join we would be using the following collection

```
// Student collection
IList<Student> studentList = new List<Student>>() {
        new Student() { StudentID = 1, StudentName = "John", Age = 13} ,
        new Student() { StudentID = 2, StudentName = "Moin",  Age = 21 } ,
        new Student() { StudentID = 3, StudentName = "Bill",  Age = 18 } ,
        new Student() { StudentID = 4, StudentName = "Ram" , Age = 20} ,
        new Student() { StudentID = 5, StudentName = "Ron" , Age = 15 }
    };
```

Query Syntax:

Query syntax is similar to SQL (Structured Query Language) for the database. It is defined within the C# or VB code.

The LINQ query syntax starts with from keyword and ends with select keyword. The following is a sample LINQ query that returns a collection of strings which contains a word "Tutorials".

```
// LINQ Query Syntax to find out teenager students
var teenAgerStudent = from s in studentList
                      where s.Age > 12 && s.Age < 20
                      select s;
```

LINQ Method Syntax

Method syntax (also known as fluent syntax) uses extension methods included in the **Enumerable** or **Queryable** static class, similar to how you would call the extension method of any class.

The following is a sample LINQ method syntax query that returns a collection of strings which contains a word "Tutorials".

Example: LINQ Method Syntax in C#

```
// LINQ Method Syntax to find out teenager students
var teenAgerStudents = studentList.Where(s => s.Age > 12 && s.Age < 20)
                                  .ToList<Student>();
```

OrderBy:

OrderBy sorts the values of a collection in ascending or descending order. It sorts the collection in ascending order by default because ascending keyword is optional here. Use descending keyword to sort collection in descending order.

Example: OrderBy in Query Syntax C#

```
var orderByResult = from s in studentList
                    orderby s.StudentName
                    select s;

var orderByDescendingResult = from s in studentList
                    orderby s.StudentName descending
                    select s;
```

Example: OrderBy in Method Syntax C#

```
var studentsInAscOrder = studentList.OrderBy(s => s.StudentName);

var studentsInDescOrder = studentList.OrderByDescending(s => s.StudentName);
```

JOIN Tables:

In our project we have used Query Syntax wherever we have to join tables. It is one of the common request where we have to fetch values from multiple table and in that case we would JOIN those tables.

Consider we have two tables Employee and Department and we have to join them to find employee and their department and the common key is departmentId

Example: JOIN in Query Syntax C#

```
var employees  = from e in employee                                    //Table employee
                 join d in department                                  //Table department
                 on e.departmentId equals d.departmentId               //Key to join tables
                 select { e.employeeName, d.departmentName };
```