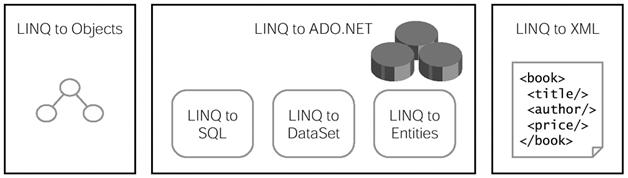
**LINQ**

**1. What is mean by LINQ?**

LINQ stands for Language Integrated Query. It is a uniform platform for any kind data. It allows us to write a query against Object Collection, Database, XML or other Data source in a standard form without having a separate Query Language.



**2. How LINQ works?**

Consider the sample customer class with two public members Name and City.

public class Customer

{

public string Name;

public string City;

}

In a console application, customer list with different Name and city are added. Using LINQ retrieve all the customers located in "Coimbatore" city and displays the name of the customer.

//Added a list of customers to the Customers collection list

List Customers = new List(){

new Customer(){Name="Sam",City="Coimbatore"}

,new Customer(){Name="Ram",City="Chennai"}

,new Customer(){Name="Sasi",City="Coimbatore"}};

//Using LINQ to retrive all the customers located in Coimbatore city

var query = from c in Customers

where c.City == "Coimbatore"

select c;

//Displaying all the customer located in Coimbatore city

foreach (Customer c in query)

{

Console.WriteLine(c.Name);

}

Console.ReadLine();

}

The compiler generates the following query for the LINQ

IEnumerable query = Customers.Where(c => c.City == "Coimbatore");

In this compiler generated code, "Where" is a extension method features introduced in C# 3.0 to filter the data based on condition. LINQ make use of these features to query data from object collection. Similar to the Where, we have Select, OrderBy, etc extension methods supported for object collection.

This query will be executed to retrieve data from the collection

**3. How will you work with relative model using LINQ?**

Using "Join" keyword we can describe relation between two entities as like SQL join statement. E.g In the below query we are joining the Customer collection with Orders collection using CustomerID members. End result we are returning new anonymous type with CustomerID, customer name, orderID as member.

var query = from c in Customers

join o in Orders

on c.CustomerID equals o.Customer.CustomerID

select new { c.CustomerID, c.Name, o.OrderID };

**4. In given line of LINQ, when actual query will be executed and why?**

In general, a LINQ query is not really executed until there is access the query result, because it describes set of operations that will be performed when necessary. In the below example actual query is executed only in the *foreach*loop

//Added a list of customers to the Customers collection list

List Customers = new List(){

new Customer(){Name="Sam",City="Coimbatore"}

,new Customer(){Name="Ram",City="Chennai"}

,new Customer(){Name="Sasi",City="Coimbatore"}};

//Using LINQ to retrive all the customers located in Coimbatore city

var query = from c in Customers

where c.City == "Coimbatore"

select c;

//Displaying all the customer located in Coimbatore city

foreach (Customer c in query)

{

Console.WriteLine(c.Name);

}

**5. What is mean by declarative programming?**

**Declarative programming** is a programming paradigm that expresses the logic of a computation without describing its control flow. There is main difference between filtering data using SQL query and C# (2.0) language such as table for SQL or an array for C# or VB.net. SQL describes what you want, where as in C# describe how to obtain the expected result. In SQL the selection of the best algorithm to implement the "How" is the responsibility of the query engine. It shows that SQL query engine has more freedom to apply optimization than c#.

**6. Is LINQ is Type safe or not?**

Data used in the LINQ area always strongly typed, including both the queried collections and the singe entities that are read and returned.

**7. What is the use of "Yield" type?**

Yeild key word is used to iterate through objects returned by a method. Method with IEnumerable return type can be used with Yeild type.

**Example:**

public static IEnumerable Power(int num, int exponent)

{

int counter = 1;

int result = 1;

//Loop till the power count reach exponent value

while (counter < exponent)

{

result = result \* num;

counter++;

//Returns power of num at each level e.g num, num^2, num^3....

yield return result;

}

}

**8. How will you map class with database table in LINQ? Or How will you create Entity class in LINQ?**

Class and Database table can be mapped using **Table** and **Column** attribute. **Table** Attribute has **Name** property to specify the name of the database table. If Name property is not supplied then LINQ-> SQL will assume database table has the same name as the class. **Column** attribute has few properties which is used to customize the exact mapping between entity and database.

[Table(Name = "Customers")]

public class Customer

{

[Column(IsPrimaryKey = true)]

public string CustomerID;

[Column]

public string Name;

[Column]

public string City;

}

**9. Why "from" clause is stated before "select" clause in LINQ which is not available in SQL?**

In LINQ, all the query starts with a "from" clause and ends with "Select" clause because Microsoft want to provide IntelliSense capabilities within the remaining part of the query which make writing rest part of the query easier.

**10. What is difference between LINQ->SQL and LINQ->Entity?**

* **Linq-to-SQL** is for direct queries using SQL Server. It's a mapping technology to map SQL Server database tables to .NET objects.
* **Linq-to-Entities** is Linq used while using Microsoft's Entity Framework, regardless of db server

**11. What are different operators used in LINQ?**

1. Where Operators - Where
2. Projection Operators - Select, SelectMany
3. Ordering Operators - OrderBy , OrderByDescending, ThenBy, ThenByDescending, Reverse
4. Grouping Operators - Group By
5. Join Operators - Join, GroupJoin
6. Set Operators - Distinct, Union, Intersect, Except
7. Aggregate Operators - count, LongCount, Sum,Min,Max, Avagrage, aggregate
8. Generation Operators - Range, Repeat, Empty
9. Qualifiers Operators -Any, All, Contains,
10. Partitioning Operators -Take, TakeWhile, Skip, SkipWhile
11. Element Operators - First, FirstOrDefault,Single, SingleOrDefault, ElementAt, ElementAtOrDefault, DefaultEmpty
12. Other Operators - Concat, SequenceEqual

**12. What are two main concepts in LINQ?**

Two main concepts which is used to understand the LINQ are Deferred query evaluation and Extension method

* Deferred Query Evaluation - LINQ queries are not evaluated when it is defined but when it is used. It is useful by defining once and used many times when it is required. Result of the query expression will always be based on latest updated content.
* Extension Method - This allows developer to add functionality in existing class without modifying the existing class or recompiling the existing class or extending the existing class. LINQ make use of this extension method for manipulating data e.g Where, Select...etc

**13. What is mean by DLINQ?**

DLINQ stands for Data Language Integrated Query, it is also referred to as LINQ to SQL. It is specifically the version of LINQ that focuses on querying data from relational data sources. In Dlinq, DataContext is the main object through which we will be retrieve objects from the database and submit changes back to DB.

**14. How SQL query generated from Linq query?**

In LINQ, DataContext class is used to communicate between LINQ and SQL. It accept and IConnection class to establish connection with SQL. It uses metadata information to map the Entity with physical structure of database.

[Table(Name = "Customers")]

public class Customer

{

[Column(IsPrimaryKey = true)]

public string CustomerID;

[Column]

public string Name;

[Column]

public string City;

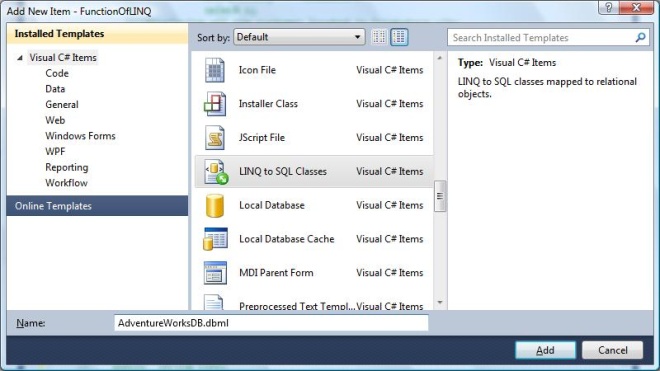
}

**15. How will you read data from SQL using Dlinq?**

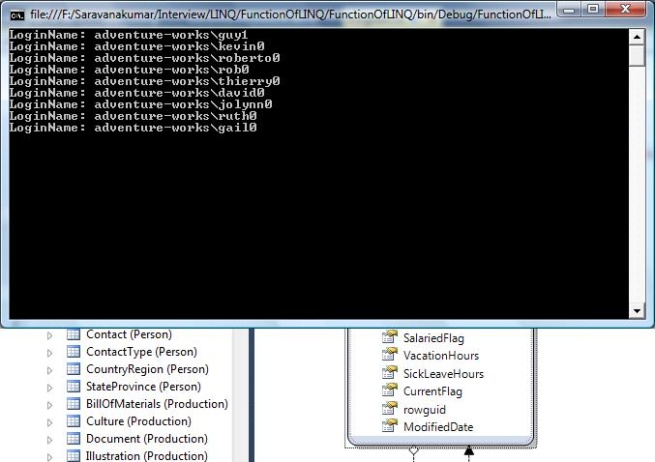
Using DLINQ user can easily read the data from SQL. Visual studio provides inbuilt template to create link between LINQ and SQL.

Example:

* Create a sample console application and add new item to the project
* In the shown template select "LINQ to SQL Classes". A new dbml file will be added to your project.



* Connect to the SQL server using Server Explorer and drag and drop any folder that we need to read the data.
* This ".dbml" file create a "Employee" Entity mapping to database table. Database column will be mapped to property of the Employee class.



* Once Table are mapped to Entity, automatically connection string will be added to your application as shown below.

<connectionStrings>

<add name="AdventureWorksConnectionString"

connectionString="Data Source=SARAVANAK-6930\SQLEXPRESS;

* + Initial Catalog=AdventureWorks;Integrated Security=True"

 providerName="System.Data.SqlClient" />

    </connectionStrings>

* As we discuss DataContext instance is root, through which we can access the corresponding table in database. In the below example I try to read the Employee table with employeeId 1 to 10.

//Creating connection string for DataContext object

SqlConnection conn = new SqlConnection();

string connString = System.Configuration.ConfigurationManager

.ConnectionStrings["AdventureWorksConnectionString"].ToString();

conn.ConnectionString = connString;

//Get the object of the Datacontext

AdventureWorksDBDataContext adWorksDC = new AdventureWorksDBDataContext(conn);

//Query from table

//Read the employee with ID between 1 and 10

var employees = from emp in adWorksDC.Employees

where emp.EmployeeID >=1 && emp.EmployeeID<10

select emp;

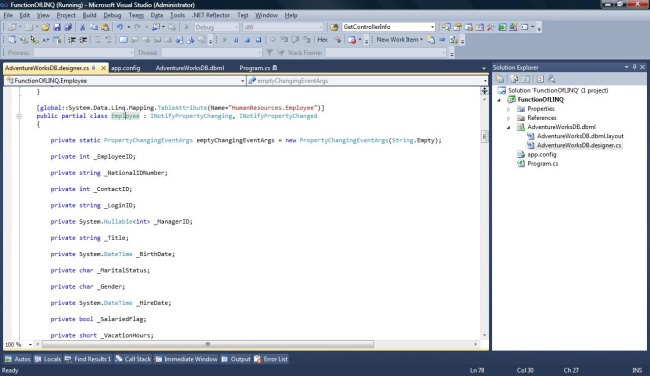
foreach (Employee e in employees )

{

Console.WriteLine(String.Format("LoginName: {0}", e.LoginID) );

}

Output:



**16. How will you Insert record to DB table using LINQ?**

A new data record can be inserted to Database table using *InsertOnSubmit()* method available in DataContext instance. Developer should call the SubmitChanges() method to commit the database transaction else data will not be insert in database table.

Example:

Below sample code shows to insert new Product Category record to database using LINQ.

//Inserting the Product category into table

ProductCategory cate = new ProductCategory();

cate.ProductCategoryID = 5;

cate.Name = "New Category";

cate.rowguid = System.Guid.NewGuid();

cate.ModifiedDate = DateTime.Now;

adWorksDC.ProductCategories.InsertOnSubmit (cate);

adWorksDC.SubmitChanges();

**17. How to update data using linq?**

Database update can be done using *"SubmitChanges()"* method in DataContext instance.

**18. How will you call Store procedure using LINQ?**

We can call the store procedure define in DB using *"Function"* and *"Parameter"* attribute. Consider the below example where user need to call the store procedure *"uspGetEmployeeManagers"* using LINQ->SQL, for which method*"GetEmployeeManagers"* will be decorated with Function attribute which requires "Name" property to mention to which SP need to call. If SP requires parameter value, these values can be passed using "Parameter" attribute. SP can return single or multiple result set. For Single result set we will be mapping with *ISingleResult* return type for multiple result set method it will return *IMultipleResults* return type of collection. Objects return by the SP will be mapped to new class with columns are mapped to property of the class.

**SP mapped to Method**

[System.Data.Linq.Mapping.Function(Name="dbo.uspGetEmployeeManagers")]

public ISingleResult GetEmployeeManagers([System.Data.Linq.Mapping.

Parameter(Name="EmployeeID", DbType="Int")] System.Nullable employeeID)

{

IExecuteResult result = this.ExecuteMethodCall(this,

((MethodInfo)(MethodInfo.GetCurrentMethod())), employeeID);

return ((ISingleResult)(result.ReturnValue));

}

**Calling SP using method**

////Creating connection string for DataContext object

SqlConnection conn = new SqlConnection();

string connString = System.Configuration.ConfigurationManager

.ConnectionStrings["AdventureWorksConnectionString"].ToString();

conn.ConnectionString = connString;

//Get the object of the Datacontext

AdventureWorksDBDataContext adWorksDC = new AdventureWorksDBDataContext(conn);

//Calling SP using method and iterating through values

foreach (uspGetEmployeeManagersResult e in adWorksDC.GetEmployeeManagers(1))

{

Console.WriteLine(String.Format("EmpID: {0} Name: {1}", e.EmployeeID, e.FirstName));

}

Console.ReadLine();

**19. How will you call User Define Function using LINQ?**

Similar to calling store procedure, functions also called using "Function" and "Parameter" attribute.

Please refer "How will you call Store procedure using LINQ?" for more information

Example:**Function mapped to a method**

[System.Data.Linq.Mapping.Function(Name="dbo.ufnGetContactInformation", IsComposable=true)]

public IQueryable GetContactInformation

([global::System.Data.Linq.Mapping.Parameter(Name="ContactID", DbType="Int")] System

.Nullable contactID)

{

return this.CreateMethodCallQuery(

this, ((MethodInfo)(MethodInfo.GetCurrentMethod())), contactID);

}

**Calling function using LINQ**

//Creating connection string for DataContext object

SqlConnection conn = new SqlConnection();

string connString = System.Configuration.ConfigurationManager

.ConnectionStrings["AdventureWorksConnectionString"].ToString();

conn.ConnectionString = connString;

//Get the object of the Datacontext

AdventureWorksDBDataContext adWorksDC = new AdventureWorksDBDataContext(conn);

//Calling Function by passing parameter as '1'

foreach (ufnGetContactInformationResult c in adWorksDC.GetContactInformation (1))

{

Console.WriteLine(String.Format("ContactID: {0} Name: {1}", c.ContactID , c.FirstName));

}

**20. How to provide Ready only access to DB?**

Suppose if we need to access data only in a read-only way and if we need to improve performance by disabling*DataContext* class to modify by using *ObjectTrackingEnabled* property of the *Datacontext* class

AdventureWorksDBDataContext adWorksDC = new AdventureWorksDBDataContext(conn);

adWorksDC.ObjectTrackingEnabled = false;

**21. How will you enable transaction in LINQ->SQL?**

A SubmitChanges() in DataContext call automatically starts a database explicit transaction. Using the*TransactionScope* class contained in the System.Transaction library will be transparently promoted to a distributed transaction.

Example:

using (TransactionScope ts = new TransactionScope())

{

//Inserting the Product category into table

ProductCategory cate = new ProductCategory();

cate.ProductCategoryID = 5;

cate.Name = "New Category";

cate.rowguid = System.Guid.NewGuid();

cate.ModifiedDate = DateTime.Now;

adWorksDC.ProductCategories.InsertOnSubmit(cate);

adWorksDC.SubmitChanges();

ts.Complete();

}

**22. How to create and delete DB from LINQ?**

We can create database using DataContext class, if we have a class derived from DataContext that contains entity definition decorated with Table and Column attributes, we can create the corresponding batabase by calling the "CreateDatabase" method.

Example:

MyDatabaseContext myDBContext = new MyDatabaseContext("Data Source=SARAVANAK-6930;

Initial Catalog=MyNewDatabase;Integrated Security=True");

myDBContext.CreateDatabase();

**23. What is the LINQ file extension that interacts with Code Behind's objects?**

\*.dbml

**24. What are the error reporting options during concurrency conflicts?**

There are two ways to report conflicts while summiting changes to database using linq.

ContinueOnConflict :- Specifies that all the update to the database to be tried and finally return all conflicts at the end of the process.

FailOnFirstConflict :- Specifies that attempts to update database should stop immediately when first concurrency conflicts are detected and return all the conflicts at that moment. In other words LINQ engine does not continue ahead executing the code.

Example:

adWorksDC.SubmitChanges(ConflictMode.FailOnFirstConflict);

**25. What are advantages of LINQ over Store procedure?**

* Debugging SP is very difficult, where as LINQ is simple by using visual studio
* Compile time safety when your schema changes
* Deployment is easier in linq everything is compiled to dll, where as in DB we need to manage deployment script.

**26. What are disadvantage of LINQ over Store procedure?**

* Store procedure can take full advantage of SQL features where as there will be some features missing in LINQ
* Store process need only serialize SP name and argument data over the wire while LINQ sends the entire query.
* Redeployment, if we need any modification, we need to recompile assembly in LINQ, where as in SP it is not the case.

**27. What is XLINQ?**

XLINQ = LINQ->XML, it is XML programming API to query XML data using LINQ. XLINQ provides power of both DOM and XQuery/XPath through LINQ extension methods to manage and query in-memory XML nodes.

**28. How will you create xml document object using XLINQ?**

XML Document can be created using *XDocument, XDeclaration , XElement, XAttribute* instance

Example:

XDocument xDocEmployee = new XDocument(

new XDeclaration("1.0", "utf-8", "yes"),

new XElement("Employee",

new XAttribute("ID", "2012"),

new XElement("FirstName", "Ram"),

new XElement("LastName", "Kumar"),

new XElement("Salary", 10000)));

XML document

<?xml version="1.0" encoding="utf-8" standalone="yes" ?>

<Employee ID="2012">

  <FirstName>Ram</FirstName>

  <LastName>Kumar</LastName>

  <Salary>10000</Salary>

</Employee>

**29. How will you read xml content using LINQ?**

Consider the xml document with list of employee. Below example gives filtering and reading the employee details based on designation.

XML Document

<?xml version="1.0" encoding="utf-8" ?>

<Employees>

  <Employee ID="101">

    <FirstName>Ram</FirstName>

    <LastName>Kumar</LastName>

    <Designation>SE</Designation>

  </Employee>

  <Employee ID="102">

    <FirstName>Sasi</FirstName>

    <LastName>Kumar</LastName>

    <Designation>SSE</Designation>

  </Employee>

  <Employee ID="103">

    <FirstName>Praveen</FirstName>

    <LastName>prakash</LastName>

    <Designation>SSE</Designation>

  </Employee>

</Employees>

LINQ Query

var employees =

from e in xDocEmployee.Elements("Employee")

where e.Element("Designation").Value == "SSE"

select new { Name = e.Element("FirstName").Value, ID = e.Attribute("ID").Value };

**30. How can we do an order by using LINQ query?**

Order by in LINQ is pretty simple. We just need to insert order by before the 'Select' query.

var employees = from emp in adWorksDC.Employees

where emp.EmployeeID >=1 && emp.EmployeeID<10

orderby emp.EmployeeID

select emp;

**31. What is the purpose of LINQ Providers in LINQ?**

LINQ Providers are a set of classes that takes a LINQ query and dynamically generates a method that executes an equivalent query against a specific data source.