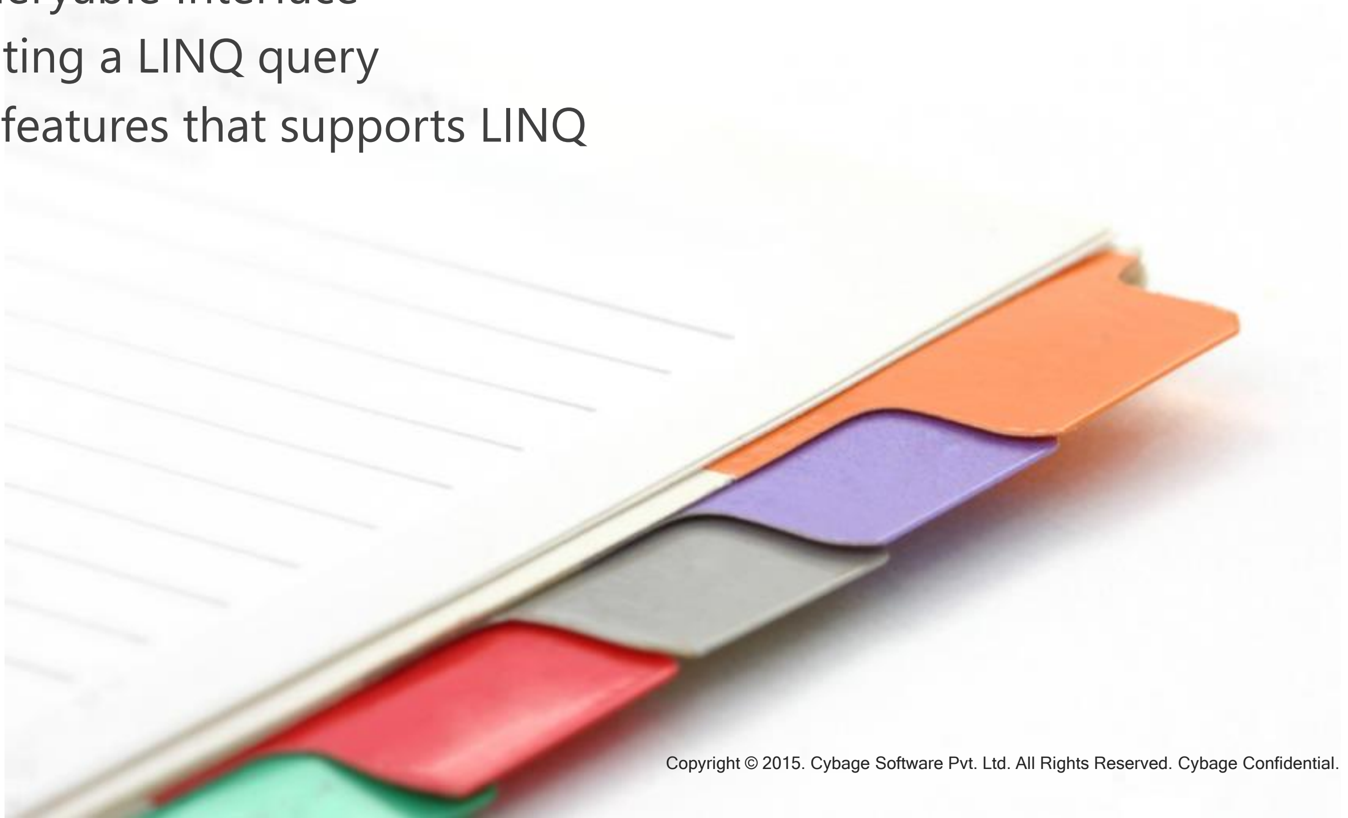
LINQ

Sushant Banerjee | sushantba@cybage.com | 7221

Agenda

- Introduction
- LINQ Architecture
- IEnumerable Interface
- IQueryable Interface
- Writing a LINQ query
- C# features that supports LINQ



The Problems

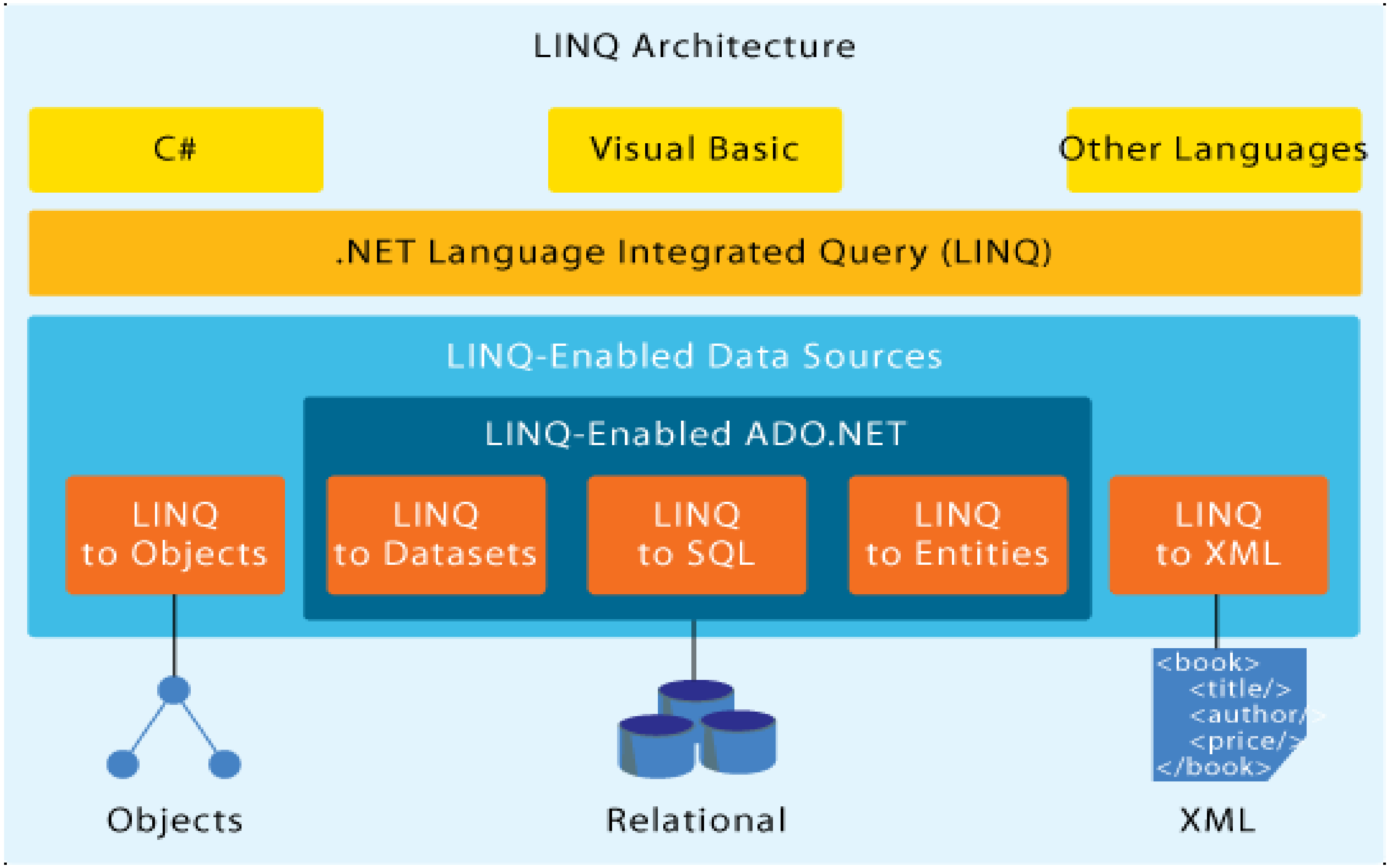
- While moving data between database and client application
  - Different data types in both locations
  - Handling null values in both locations
- While passing commands to the database
  - Commands are written in string format
  - Compiler can’t detect syntax errors
  - Compiler even can’t detect if the command is querying an incorrect object
- While querying different data sources
  - Learn C# to query in-memory collections
  - Learn SQL to query RDBMS
  - Lear XQuery to query XML.
- How to solve the above problems?

What is LINQ

- Stands for Language-Integrated Query
- Introduced with .NET Framework 3.5 and VS 2008
- Strongly typed queries with intelliSense support
- Supports any kind of data source
    - .NET Framework Collections
    - SQL Server databases
    - ADO.NET datasets
    - XML documents

- Yes! Now C# can query all the above data sources.

LINQ Architecture

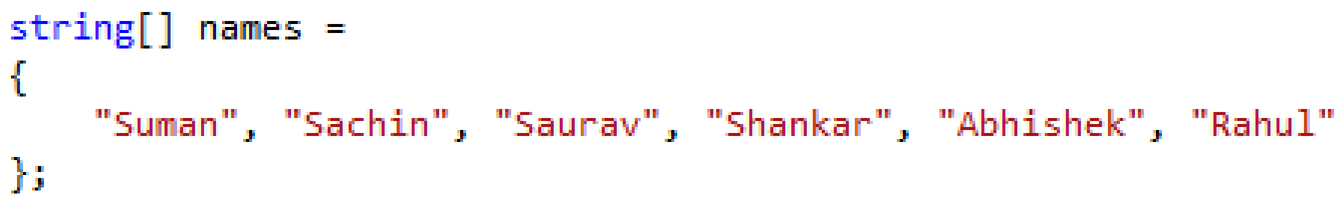


Writing LINQ Query – LINQ to Objects

- Creating a data source
  - In-memory collection objects
- Writing a query
  - The query returns IEnumerable object
- Executing a query
  - Iterate over items of the collection

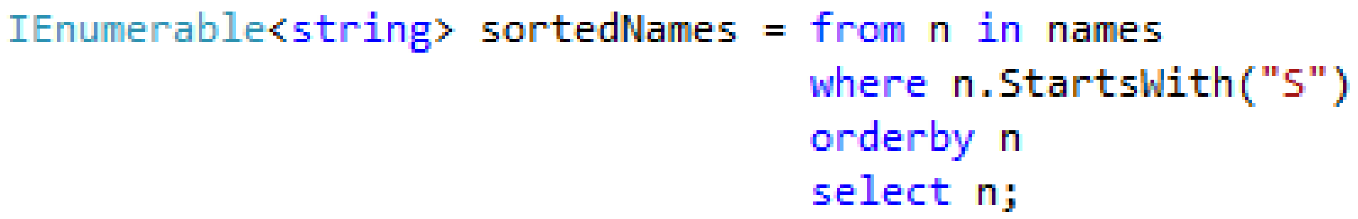
A Simple LINQ Query

- Creating a data source



```
string[] names =
{
     "Suman", "Sachin", "Saurav", "Shankar", "Abhishek", "Rahul"
};
```

- Writing a query



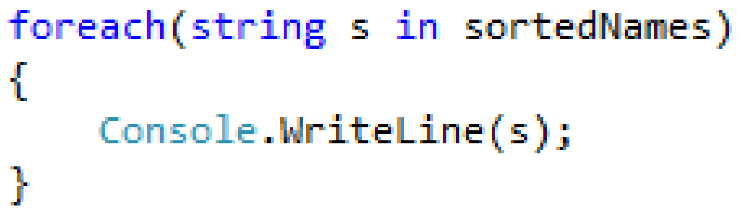
```
IEnumerable<string> sortedNames = from n in names
                                  where n.StartsWith("S")
                                  orderby n
                                  select n;
```

- Where is data? In sortedNames???

Executing Query

- Data is accessed only when you
  - Use foreach loop
  - Call ToList or ToArray methods
  - Call aggregation functions such as Count, Max, Average, First etc.

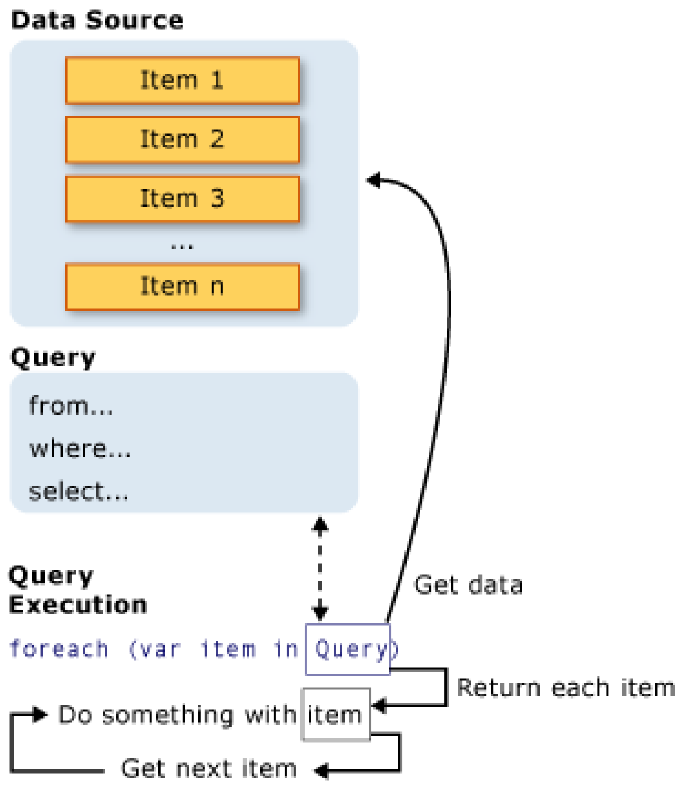
- This feature is called Deferred execution
  - Execution is delayed till the time you access data



```
foreach(string s in sortedNames)
{
    Console.WriteLine(s);
}
```

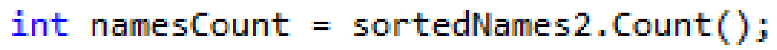
Execution of the Query

- LINQ query returns generic IEnumerable object
- Stores query commands instead of data
- A query variable doesn’t store any data



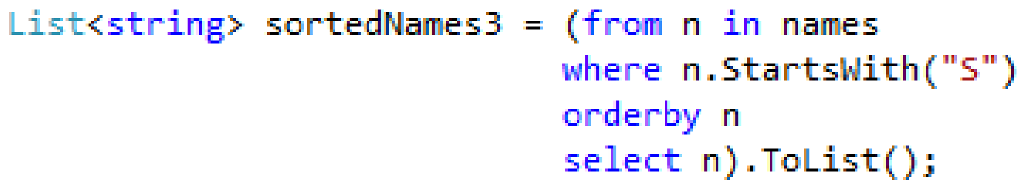
Forcing Immediate Execution

- Doing aggregation

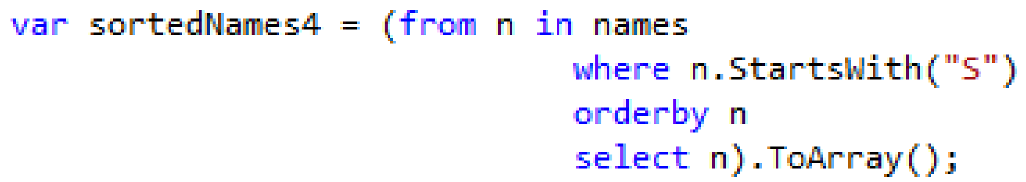


```
int namesCount = sortedNames2.Count();
```

- Calling methods



```
List<string> sortedNames3 = (from n in names
                             where n.StartsWith("S")
                             orderby n
                             select n).ToList();
```



```
var sortedNames4 = (from n in names
                            where n.StartsWith("S")
                            orderby n
                            select n).ToArray();
```

Demo

Query Operations

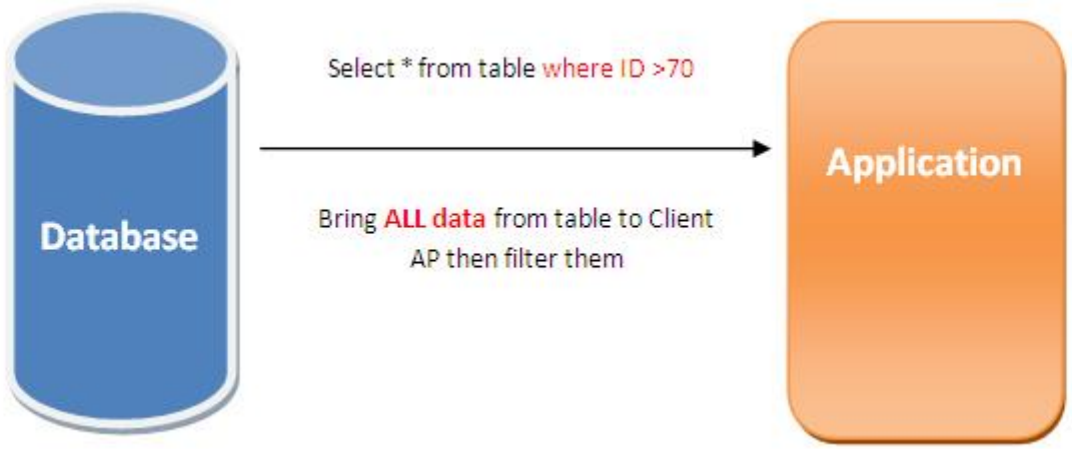
- Returning all data
- Filtering data
- Grouping data
- Ordering data
- Join operation
- Projection

A Queryable Type

- A Queryable type is a LINQ data source
- LINQ works with any data source that supports
  - IEnumerable or Generic IEnumerable<T> interface
  - IQueryable or Generic IQueryable<T> interface
- IEnumerable or IQueryable? Which one is better?

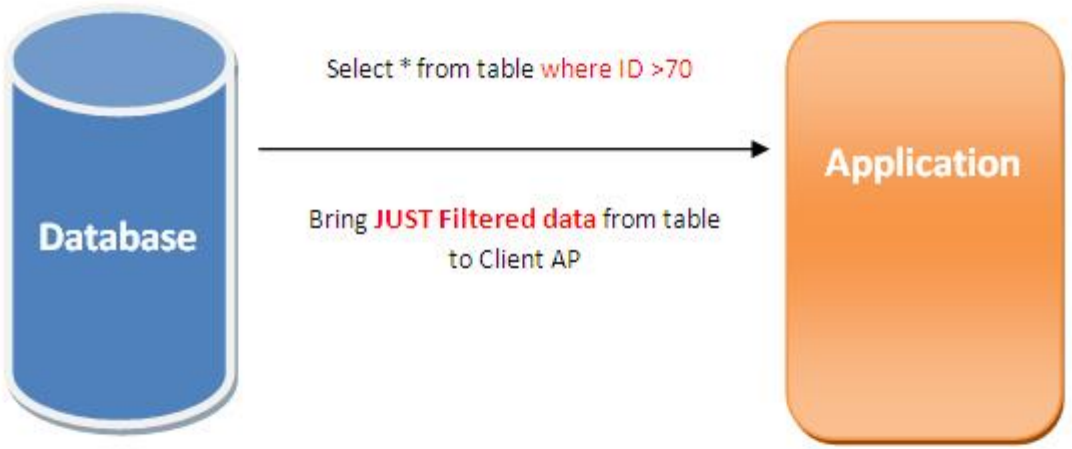
IEnumerable Interface

- Suitable for iterating through in-memory collections
- Creates performance issues


Database
Select * from table where ID >70
Bring ALL data from table to Client AP then filter them
Application

IQueryable Interface

- Suitable for iterating through out-memory collections
- Provides high performance
- Ideal for processing huge volume of data


Database
Select * from table where ID >70
Bring JUST Filtered data from table to Client AP
Application

IEnumerable Vs. IQueryable

IEnumerable	IQueryable
Exists in System.Collection namespace.	Exists in System.Linq namespace.
Has no base interface.	Derived from IEnumerable interface.
Best to query data from in-memory collection like Array, List etc.	Best to query data from out-memory collection like remote database, service etc.
Executes select query on server side, loads data in memory on client side and then filter data.	Executes select query on server side with all filters.
Does not support lazy loading.	Supports lazy loading.

IQueryable Extension Methods

Extension Method	Purpose	SQL
Select<T>()	Selects only specified properties	Result Projection
Where<T>()	Filters records	WHERE
Distinct<T>()	Distinct records	DISTINCT
Any<T>()	Checks if at least one record satisfies condition	WHERE EXISTS
All<T>()	Checks if ALL records satisfies condition	WHERE NOT EXISTS
Count<T>()	Count of elements satisfying given condition	Sub-Query

IQueryable Extension Methods contd..

Extension Method	**Purpose**	**SQL**
OrderBy<T>()	Sorting resultset in ascending order	ORDER BY
OrderByDescending<T>()	Sorting resultset in descending order	ORDER BY … DESC
First<T>()	Returns first record	TOP(1)
FirstOrDefault<T>()	Returns first record, if exists, else default value	TOP(1)
Single<T>()	Same as *First<T>()*, but WHERE clause MUST return 1 row. Throws Exception if 0 or 2 rows returned	TOP(2)
SingleOrDefault<T>()	Same as Single<T>() except it returns Default value if Collection is empty	TOP(2)

IQueryable Extension Methods contd..

- Aggregate Functions

Extension Method	**SQL**
Sum<T>()	SUM
Max<T>()	MAX
Min<T>()	MIN
Average<T>()	AVG

Required Features to Create LINQ

Features	Meaning
Object Initializers	A new syntax to create object and set properties in a single line of code.
Implicitly Typed Local Variables	A variable that uses "var" keyword instead of data type. The type is decided by compiler.
Anonymous Types	Creating classes on the fly without using type name.
Lambda Expressions	Shorter syntax to create anonymous method.
Extension Methods	Adding functionalities to inbuilt types without modifying or recompiling.

Cybage®
Delivering Value. Scientifically.

Demo

Summary

- Why LINQ
- What is LINQ
- IEnumerable Interface
- IQueryable Interface

Bibliography, Important Links

- https://msdn.microsoft.com/en-us/library/bb397933(v=vs.120).aspx

Any Questions?

Thank you!