A Dataset offers an extremely useful data representation in memory and is used for a diverse range of data based applications. LINQ to Dataset as one of the technology of LINQ to ADO.NET facilatitates performing queries on the data of a Dataset in a hassle-free manner and enhance productivity.

Introduction of LINQ To Dataset

LINQ to Dataset has made the task of querying simple for the developers. They don't need to write queries in a specific query language instead the same can be written in programming language. LINQ to Dataset is also usable for querying where data is consolidated from multiple data sources. This also does not need any LINQ provider just like LINQ to SQL and LINQ to XML for accessing data from in memory collections.

Below is a simple example of a LINQ to Dataset query in which a data source is first obtained and then the dataset is filled with two data tables. A relationship is established between both the tables and a LINQ query is created against both tables by the means of join clause. Finally, foreach loop is used to display the desired results.

C#
```vbnet
Imports System.Data.SqlClient
Imports System.Linq
Module LinqToDataSet
    Sub Main()
        Dim connectString As String = System.Configuration.ConfigurationManager.ConnectionStrings("LinqToSQLDBConnectionString").ToString()

        Dim sqlSelect As String = "SELECT * FROM Department;" + "SELECT * FROM Employee;"
        Dim sqlCnn As SqlConnection = New SqlConnection(connectString)
        sqlCnn.Open()

        Dim da As New SqlDataAdapter
        da.SelectCommand = New SqlCommand(sqlSelect, sqlCnn)
        da.TableMappings.Add("Table", "Department")
        da.TableMappings.Add("Table1", "Employee")

        Dim ds As New DataSet()
        da.Fill(ds)

        Dim dr As DataRelation = ds.Relations.Add("FK_Employee_Department", ds.Tables("Department").Columns("DepartmentId"), ds.Tables("Employee").Columns("DepartmentId"))

        Dim department As DataTable = ds.Tables("Department")
        Dim employee As DataTable = ds.Tables("Employee")

        Dim query = From d In department.AsEnumerable() Join e In employee.AsEnumerable() On d.Field(Of Integer)("DepartmentId") Equals e.Field(Of Integer)("DepartmentId")
        Select New Person With{
            EmployeeId = e.Field(Of Integer)("EmployeeId"),
            EmployeeName = e.Field(Of String)("Name"),
            DepartmentId = d.Field(Of Integer)("DepartmentId"),
            DepartmentName = d.Field(Of String)("Name")
        }

        For Each e In query
            Console.WriteLine("Employee Id = {0} , Name = {1} , Department Name = {2}", e.EmployeeId, e.EmployeeName, e.DepartmentName)
        Next
    End Sub
End Module
```
When the above code of C# or VB is compiled and executed, it produces the following result:

Employee Id = 1, Name = William, Department Name = Account
Employee Id = 2, Name = Benjamin, Department Name = Account
Employee Id = 3, Name = Miley, Department Name = Sales

Press any key to continue.

Querying Dataset using LINQ to Dataset

Before beginning querying a Dataset using LINQ to Dataset, it is vital to load data to a Dataset and this is done by either using DataAdapter class or by LINQ to SQL. Formulation of queries using LINQ to Dataset is quite similar to formulating queries by using LINQ alongside other LINQ enabled data sources.

Single-Table Query

In the following single-table query, all online orders are collected from the SalesOrderHeader table and then order ID, Order date as well as order number are displayed as output.

C#
DataTable department = ds.Tables["Department"];

var query = from d in department.AsEnumerable()
            select new
            {
                DepartmentId = d.Field<int>("DepartmentId"),
                DepartmentName = d.Field<string>("Name")
            };

foreach (var q in query)
{
    Console.WriteLine("Department Id = {0} , Name = {1}",
                      q.DepartmentId, q.DepartmentName);
}

Console.WriteLine("\nPress any key to continue.");
Console.ReadKey();
When the above code of C# or VB is compiled and executed, it produces the following result:

Department Id = 1, Name = Account
Department Id = 2, Name = Sales
Department Id = 3, Name = Pre-Sales
Department Id = 4, Name = Marketing

Press any key to continue.