Sample 1. Simple one

var empQuery = from emp in employees

where emp.Department == "IT Dept"

&& emp.FName != “John”

orderby emp.FName

select emp;

int cnt = empQuery.Count();

foreach (var emp in empQuery)

{

Console.WriteLine(emp.Department);

}

Sample 2. From an array

string[] employees = {"Michael", "Hank", "Benjamin"}

var empQuery = from emp in employees

where emp.Length > 5

select emp;

Sample 3. From an XML

XElement empXml = new XElement("Employees",

new XElement("Employee",

new XElement("ID", "111"),

new XElement("FirstName", "Michael"),

new XElement("Department", "IT Department"),

new XElement("City", "Pittsburgh")

),

new XElement("Employee",

new XElement("ID", "112"),

new XElement("FirstName", "Hank"),

new XElement("Department", "IT Department"),

new XElement("City", "Redmond")

),

new XElement("Employee",

new XElement("ID", "113"),

new XElement("FirstName", "Benjamin"),

new XElement("Department", "Human Resources"),

new XElement("City", "Chicago"),

)

);

IEnumerable<XElement> empQuery =

from emp in empXml.Descendants("Employee")

where emp.Element("FirstName").Value.Length > 5

select emp;

Sample 4. Join

var empQuery = from emp in employees

join ts in timesheets

on emp.ID equals ts.EmployeeId

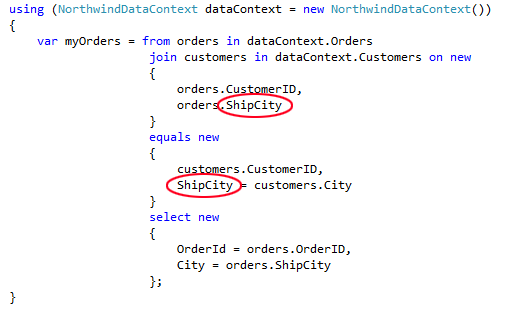
select new {

EmployeeId=emp.ID,

EmployeeName =emp.First + " " + emp.Last,

VacationHours = ts.HoursVacation

};



var query = (from person in context.People   
             join pet in context.Pets

on person equals pet.Owner   into tempPets   
              from pets in tempPets.DefaultIfEmpty()   
             select new {

OwnerName = person.Name,

Pet = pet.Name })   
            .ToList();

decimal minimo = 3.5**M**;

var StudentInfo =

from dptos in context.Department

join courses in context.Course on dptos.DepartmentID equals courses.DepartmentID

join notas in context.StudentGrade on notas.CourseID equals courses.CourseID

join people in context.Person on people.PersonID equals notas.PersonID

where notas.Grade > minimo

select new

{

Nombre = people.FirstMidName,

Dpto = dptos.Name,

Curso = courses.Title,

Nota = notas.Grade

};

Sample 5. Concat

var employeesAndConsul = (from emp in employees

where emp.City == "Redmond"

select emp.First + " " + emp.Last).Concat(

from cn in consultants

where cn.Location == "Redmond"

select cn.Name);

Sample 6. To a DataView

ConnectionStringSettings pubsCnn = ConfigurationManager.ConnectionStrings["PubsData"];

PubsDataSet pubs = new PubsDataSet();

SqlDataAdapter adp = new SqlDataAdapter("select \* from publishers;", pubsCnn.ConnectionString);

adp.Fill(pubs, "publishers");

IEnumerable<DataRow> pubsQuery = from p in pubs.publishers

where p.country == "USA"

select p;

GridView1.DataSource = pubsQuery;

GridView1.DataBind();

Sample 7. For Each

DataTable employees = **MyDataProvider**.GetEmployeeData();

EnumerableRowCollection<DataRow> query = from employee in employees.AsEnumerable()

where employee.Field<Decimal>("salary") > 20

orderby employee.Field<Decimal>("salary")

select employee;

foreach (DataRow emp in query)

{

Response.Write(emp.Field<String>("LastName") + ": ");

Response.Write(emp.Field<Decimal>("salary") + "<br />");

}

Sample 8. 2 Data Tables

DatabaseSchema dbSchema = new DatabaseSchema();

DataSet pubs = dbSchema.GetPubsData();

// 2 data tables

DataTable publishers = pubs.Tables["publishers"];

DataTable titles = pubs.Tables["titles"];

var pubQuery = from title in **titles**.AsEnumerable()

join pub in **publishers**.AsEnumerable()

on title.Field<string>("pub\_id") equals pub.Field<string>("pub\_id")

where title.Field<decimal>("price") < 10

orderby title.Field<string>("title")

select new

{

Publisher = pub.Field<string>("pub\_name"),

Title = title.Field<string>("title"),

Price = title.Field<decimal>("price")

};

GridView1.DataSource = pubQuery;

GridView1.DataBind();

Sample 9. From a DataSet, select from a condition, then Distinct and then to a DataView

In the following example:

- A DataSet is created from the titles table in the pubs database.

- A LINQ query then runs to get all price values for books that have sold more than 1000 copies.

- The Distinct method is then used to show a list of distinct prices across this list of books.

- Assign result to a DataView

DatabaseSchema dbSchema = new DatabaseSchema();

DataSet titlesDs = dbSchema.GetTitles();

var titleQuery = from title in titlesDs.Tables["titles"].AsEnumerable()

where title.Field<int>("ytd\_sales") > 1000

orderby title.Field<decimal>("price")

select title.Field<decimal>("price");

var prices = titleQuery.Distinct();

GridView1.DataSource = prices;

GridView1.DataBind();

Sample 10. Calculations

var queryAvg = from employee in employees.AsEnumerable()

group employee by "" into g

select new

{

AvgSalary = g.Average(employee => employee.Field<Decimal>("Salary"))

};

foreach (var emp in queryAvg)

{

Response.Write(emp.AvgSalary.ToString() + "<br />");

}

Sample 11. Into DataTables, and using Intersect

As another example, the following code

* Executes two queries against the titles DataSet.
* The **first query** returns all titles that were written with an advance of $5,000 or more.
* The **second query** returns all titles with year-to-date sales greater than $3,000.
* These queries are each converted into separate DataTable objects by using the **CopyToDataTable** method.
* Finally, the Intersect method is called to return all titles that match between the two DataSet objects.

DatabaseSchema dbSchema = new DatabaseSchema();

DataSet titlesDs = dbSchema.GetTitles();

// First query

var titleAdvQuery = from title in titlesDs.Tables["titles"].AsEnumerable()

where title.Field<decimal>("advance") > 5000

select title;

// Second query

var titleSalesQuery = from title in titlesDs.Tables["titles"].AsEnumerable()

where title.Field<int>("ytd\_sales") > 3000

select title;

// Each query into a DataTable

DataTable dtAdvance = titleAdvQuery.CopyToDataTable();

DataTable dtSales = titleSalesQuery.CopyToDataTable();

// Intersect, titles common in both DataTables

IEnumerable<DataRow> advanceSales =

dtAdvance.AsEnumerable().Intersect(dtSales.AsEnumerable(), DataRowComparer.Default);

GridView1.DataSource = advanceSales.CopyToDataTable();

GridView1.DataBind();

Sample 12. Nested LINQ

private void PopulateDropDownLists()

{

using (var context = new SchoolModel.SchoolEntities())

{

// GET ALL COURSES

var allCourses = (from c in context.Course

select c).ToList();

// FILTER FOR ASSIGNED COURSES

var instructorID = Convert.ToInt32(InstructorDropDown.SelectedValue);

var instructor = (from p in context.Person.Include("Course")

where p.PersonID == instructorID

select p).First();

var assignedCourses = instructor.Course.ToList();

DropDownList1.DataSource = assignedCourses;

DropDownList1.DataBind();

//FILTER FOR UNASSIGNED COURSES

var unassignedCourses =

allCourses.Except(assignedCourses.AsEnumerable()).ToList();

DropDownList2.DataSource = unassignedCourses;

DropDownList2.DataBind();

}

}

|  |  |
| --- | --- |
| Without Store Procedure | With Store Procedure |
| var allCourses = (from c in context.Courses                    select c).ToList(); | var allCourses = context.sp\_GetCourses(); |

//Missing agents

var query2 = from predialer in dtPredialer.AsEnumerable()

join roster in dtRoster.AsEnumerable()

on predialer.Field<Int64>("EmpNumber") equals roster.Field<Int64>("UNIQUE EMPLOYEE INDENTIFIER")

into RightTableResults

from roster1 in RightTableResults.DefaultIfEmpty()

where (roster1 == null ? 0 : roster1.Field<Int64>("UNIQUE EMPLOYEE INDENTIFIER")) == 0

orderby predialer.Field<String>("Location"), predialer.Field<Int64>("EmpNumber")

select new

{

EmpNumber1 = predialer.Field<Int64>("EmpNumber"),

Agent = predialer.Field<String>("Agent"),

};