**The following are the different types of joins in LINQ**  
Group Join - We will discuss in this video  
Inner Join - Discussed in [Part 22](http://csharp-video-tutorials.blogspot.com/2014/08/part-22-inner-join-in-linq.html)  
Left Outer Join  
Cross Join   
  
   
  
In this video, we will discuss **Group Join**. Group Join produces hierarchical data structures. Each element from the first collection is paired with a set of correlated elements from the second collection.   
  
Let us understand **Group Join**with an **example.**Consider the following **Department**and **Employee**classes. A Department may have ZERO or MORE employees. 

public class Department

{

    public int ID { get; set; }

    public string Name { get; set; }

    public static List<Department> GetAllDepartments()

    {

        return new List<Department>()

        {

            new Department { ID = 1, Name = "IT"},

            new Department { ID = 2, Name = "HR"},

            new Department { ID = 3, Name = "Payroll"},

        };

    }

}

public class Employee

{

    public int ID { get; set; }

    public string Name { get; set; }

    public int DepartmentID { get; set; }

    public static List<Employee> GetAllEmployees()

    {

        return new List<Employee>()

        {

            new Employee { ID = 1, Name = "Mark", DepartmentID = 1 },

            new Employee { ID = 2, Name = "Steve", DepartmentID = 2 },

            new Employee { ID = 3, Name = "Ben", DepartmentID = 1 },

            new Employee { ID = 4, Name = "Philip", DepartmentID = 1 },

            new Employee { ID = 5, Name = "Mary", DepartmentID = 2 },

            new Employee { ID = 6, Name = "Valarie", DepartmentID = 2 },

            new Employee { ID = 7, Name = "John", DepartmentID = 1 },

            new Employee { ID = 8, Name = "Pam", DepartmentID = 1 },

            new Employee { ID = 9, Name = "Stacey", DepartmentID = 2 },

            new Employee { ID = 10, Name = "Andy", DepartmentID = 1}

        };

    }

}

**Example 1:** Group **employees**by **Department**.

var employeesByDepartment = Department.GetAllDepartments()

                                                                           .GroupJoin(Employee.GetAllEmployees(),

                                                                             d => d.ID,

                                                                             e => e.DepartmentID,

                                                                             (department, employees) => new

                                                                             {

                                                                                 Department = department,

                                                                                 Employees = employees

                                                                             });

foreach (var department in employeesByDepartment)

{

    Console.WriteLine(department.Department.Name);

    foreach (var employee in department.Employees)

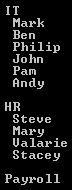
    {

        Console.WriteLine(" " + employee.Name);

    }

    Console.WriteLine();

}

**Output:**   
   
  
**Example 2:** Rewrite **Example 1**using SQL like syntax.

var employeesByDepartment = from d in Department.GetAllDepartments()

                                                       join e in Employee.GetAllEmployees()

                                                       on d.ID equals e.DepartmentID into eGroup

                                                       select new

                                                       {

                                                          Department = d,

                                                          Employees = eGroup

                                                       };

**Please note:**Group Join uses the **join**operator and the **into**keyword to group the results of the join.