The following 2 standard LINQ query operators belong to **Projection Operators**category.   
Select  
SelectMany   
  
   
  
**Projection Operators (Select & SelectMany)** are used to transform the results of a query. In this video we will discuss **Select**operator and in a later video session we will discuss **SelectMany**operator.   
  
**Select clause**in SQL allows to specify what columns we want to retrieve. In a similar fashion LINQ SELECT standard query operator allows us to specify what properties we want to retrieve. It also allows us to perform calculations.  
  
**For example**, you may have a collection of Employee objects. The following are the properties of the **Employee**class.  
EmployeeID  
FirstName  
LastName  
AnnualSalay  
Gender  
  
**Now using the SELECT projection operator**  
**1.** We can select just **EmployeeID**property OR  
**2.** We can select multiple properties (**FirstName & Gender**) into an anonymous type OR  
**3.** Perform calculations   
    **a)** MonthlySalary = AnnualSalay/12  
    **b)** FullName = FirstName + " " + LastName  
  
We will be using the following **Employee**class for this demo.

public class Employee

{

    public int EmployeeID { get; set; }

    public string FirstName { get; set; }

    public string LastName { get; set; }

    public string Gender { get; set; }

    public int AnnualSalary { get; set; }

    public static List<Employee> GetAllEmployees()

    {

        List<Employee> listEmployees = new List<Employee>

        {

            new Employee

            {

                EmployeeID = 101,

                FirstName = "Tom",

                LastName = "Daely",

                Gender = "Male",

                AnnualSalary = 60000

            },

            new Employee

            {

                EmployeeID = 102,

                FirstName = "Mike",

                LastName = "Mist",

                Gender = "Male",

                AnnualSalary = 72000

            },

            new Employee

            {

                EmployeeID = 103,

                FirstName = "Mary",

                LastName = "Lambeth",

                Gender = "Female",

                AnnualSalary = 48000

            },

            new Employee

            {

                EmployeeID = 104,

                FirstName = "Pam",

                LastName = "Penny",

                Gender = "Female",

                AnnualSalary = 84000

            },

        };

        return listEmployees;

    }

}

**Example 1:** Retrieves just the **EmployeeID**property of all employees

IEnumerable<int> employeeIds = Employee.GetAllEmployees()

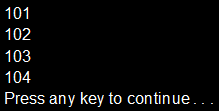
    .Select(emp => emp.EmployeeID);

foreach (int id in employeeIds)

{

    Console.WriteLine(id);

}

**Output:**   
   
  
**Example 2:** Projects **FirstName & Gender**properties of all employees into **anonymous type**.

var result = Employee.GetAllEmployees().Select(emp => new

                    {

                        FirstName = emp.FirstName,

                        Gender = emp.Gender

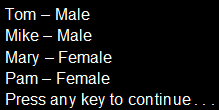
                    });

foreach (var v in result)

{

    Console.WriteLine(v.FirstName + " - " + v.Gender);

}

**Output:**   
   
  
**Example 3:** Computes **FullName and MonthlySalay**of all employees and projects these 2 new computed properties into anonymous type.

var result = Employee.GetAllEmployees().Select(emp => new

{

    FullName = emp.FirstName + " " + emp.LastName,

    MonthlySalary = emp.AnnualSalary / 12

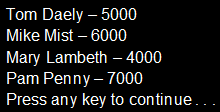
});

foreach (var v in result)

{

    Console.WriteLine(v.FullName + " - " + v.MonthlySalary);

}

**Output:**   
   
  
**Example 4:** Give **10% bonus**to all employees whose annual salary is greater than **50000**and project all such employee's **FirstName, AnnualSalay and Bonus**into anonymous type.

var result = Employee.GetAllEmployees()

                .Where(emp => emp.AnnualSalary > 50000)

                .Select(emp => new

                 {

                    Name = emp.FirstName,

                    Salary = emp.AnnualSalary,

                    Bonus = emp.AnnualSalary \* .1

                 });

foreach (var v in result)

{

    Console.WriteLine(v.Name + " : " + v.Salary + " - " + v.Bonus);

}

**Output:**   
