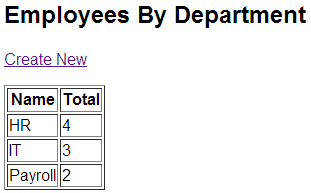
In this video we will discuss, using data transfer object as the model in mvc. Please [watch Part 28](http://csharp-video-tutorials.blogspot.com/2013/05/part-28-customizing-autogenerated-edit.html), before proceeding.   
  
Let's say the business requirement is such that, we want to display**total number of employees by department** as shown below. At the moment, either the **Employee** or **Department** class does not have Total property. This is one example, where a **Data Transfer Object** can be used as a model.  


Right click on the **"Models"** folder and add a class with name=**"DepartmentTotals.cs"**. Copy and paste the following code.  
public class DepartmentTotals  
{  
    public string Name { get; set; }  
    public int Total { get; set; }  
}  
  
Now add the following **"EmployeesByDepartment"** controller action method to **EmployeeController** class.  
public ActionResult EmployeesByDepartment()  
{  
    var departmentTotals = db.Employees.Include("Department")  
                                .GroupBy(x => x.Department.Name)  
                                .Select(y => new DepartmentTotals   
                                {   
                                    Name = y.Key, Total = y.Count()   
                                }).ToList();  
    return View(departmentTotals);  
}  
  
At this point, build the solution, so that the newly added **DepartmentTotals**class is compiled.  
  
Now right click on **"EmployeesByDepartment"** action method in **"EmployeeController"** and select **"Add View"** from the context menu.  
**View name** = EmployeesByDepartment  
**View engine** = Razor  
Select **"Create a strongly-typed view"** checkbox  
**Model class** = DepartmentTotals  
**Model class** = DepartmentTotals  
  
To list the employees in **ascending**order of total employee, use **OrderBy**() LINQ method as shown below.  
var departmentTotals = db.Employees.Include("Department")  
                            .GroupBy(x => x.Department.Name)  
                            .Select(y => new DepartmentTotals   
                            {   
                                Name = y.Key, Total = y.Count()   
                            }).ToList().OrderBy(y => y.Total);  
  
To sort the list in **descending**order use, **OrderByDescending**() LINQ method.  
var departmentTotals = db.Employees.Include("Department")  
                            .GroupBy(x => x.Department.Name)  
                            .Select(y => new DepartmentTotals   
                            {   
                                Name = y.Key, Total = y.Count()   
                            }).ToList().OrderByDescending(y => y.Total);  
            return View(departmentTotals);