D:\Dropbox\SS-Slides\DeCuong-CDIO\TemplateCDIOv1\HinhAnh\LogoCDIO.pngD:\Dropbox\SS-Slides\DeCuong-CDIO\TemplateCDIOv1\HinhAnh\LogoTruong.png

**LẬP TRÌNH ỨNG DỤNG VỚI NGÔN NGỮ C#**

**Hướng dẫn sử dụng entity framework trong .NET**

Bộ môn **Hệ thống thông tin**

Khoa Công nghệ thông tin

ĐH Khoa học tự nhiên TP HCM

Tóm tắt nội dung bài hướng dẫn:

* Giới thiệu Entity Framework (EF)
* Kiến trúc EF
* Hướng dẫn sử dụng EF trong .NET 4
* Bài tập

**PHÁT TRIỂN ỨNG DỤNG CSDL 1**

**Tháng 9/2014**

**MỤC LỤC**

[1 Giới thiệu 1](#_Toc404412912)

[2 Kiến trúc 1](#_Toc404412913)

[3 Hướng dẫn sử dụng 2](#_Toc404412914)

[4 Ví dụ tổng hợp 11](#_Toc404412915)

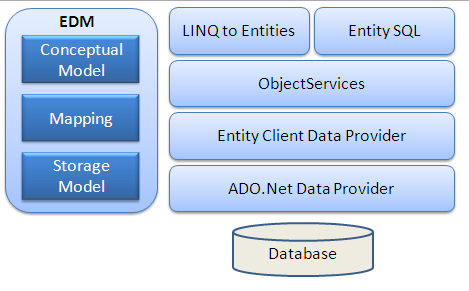
[5 Vấn đề hiệu suất cho mô hình EF 19](#_Toc404412916)

[6 Bài tập 19](#_Toc404412917)

# Giới thiệu

* Entity Framework (EF) là bộ ánh xạ các đối tượng quan hệ thành các đối tượng ứng dụng đặc trưng. Các nhà phát triển theo trường phái hướng đối tượng sẽ làm việc với những đối tượng EF mà không cần quan tâm dữ liệu quan hệ. ( <http://msdn.microsoft.com/en-us/data/ef.aspx> )
* EF là hiện thực hóa của framework Object/Relational Mapping (ORM), ORM thực hiện ánh xạ qua lại giữa dữ liệu quan hệ và các đối tượng lĩnh vực đặc trưng. Công việc này thực hiện tự động mà không cần sự can thiệp của lập trình viên. Hiện nay có nhiều framework hiện thực ORM như: DataObjects.Net, NHibernate, OpenAccess, SubSonic,….

# Kiến trúc

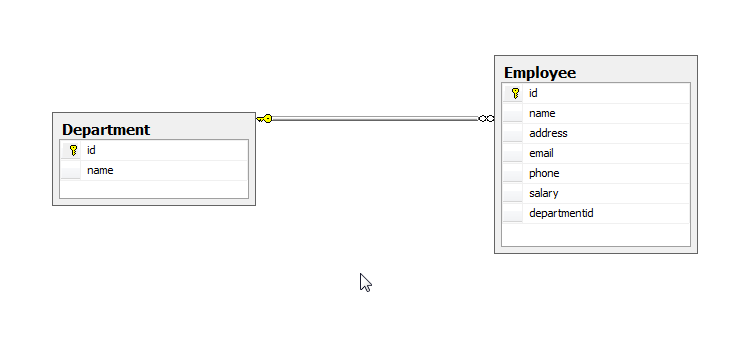


Giải thích các thành phần trong EF

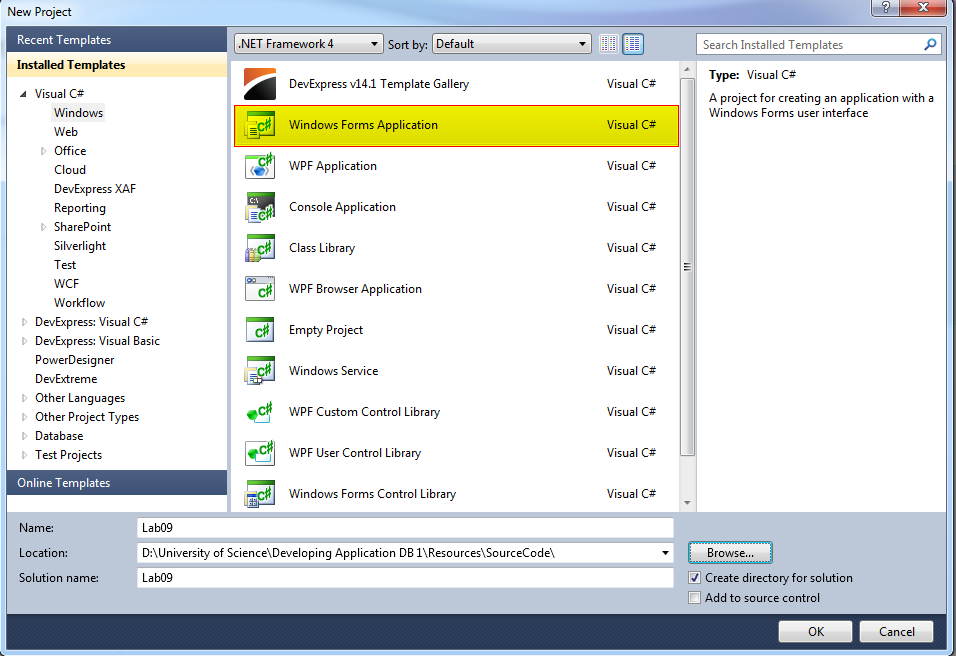
* **EDM (Entity Data Model)**: EDM gồm 3 phần chính- Conceptual model, Mapping and Storage model.
* **Conceptual Model**: Chứa các classes và quan hệ giữa các classes, độc lập với csdl quan hệ.
* **Storage Model**: Chứa mô hình thiết kế csdl như bảng, stored procedure, khóa chính, khóa ngoại
* **Mapping**: Ánh xạ thông tin nhất quán giữa Conceptual Model - Storage Model
* **LINQ to Entities**: Ngôn ngữ truy vấn đối tượng, các đối tượng trong Conceptual Model
* **Entity SQL**: Entity SQL là ngôn ngữ truy vấn giống như LINQ to Entities. Tuy nhiên, nó khó khăn để học hơn LINQ
* **Object Service**: Truy cập dữ liệu từ csdl.
* **Entity Client Data Provider**: Chuyển câu truy vấn LINQ hoặc E-SQL thành câu truy vấn thực thi trong môi trường csdl quan hệ.
* **ADO.Net Data Provider**: Giao tiếp với csdl theo chuẩn ADO.NET

# Hướng dẫn sử dụng

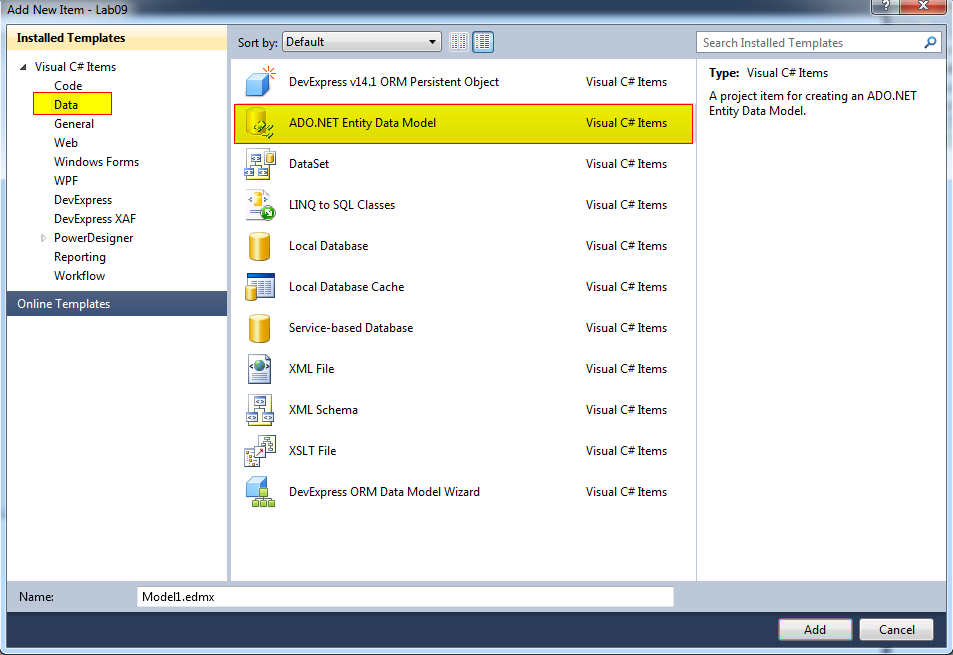
* Để sử dụng mô hình EF, đề nghị người đọc tìm hiểu ngôn ngữ LINQ với các link tham khảo:
  + <http://www.dotnetlearners.com/linq/introduction-to-linq.aspx>
  + <http://www.linq-training-guide.com/index.html>
  + ………….
* Các bước xây dựng ứng dụng áp dụng mô hình Entity Framework
  + Bước 1: Tạo csdl QuanLySinhVien có lược đồ như hình dưới đây:



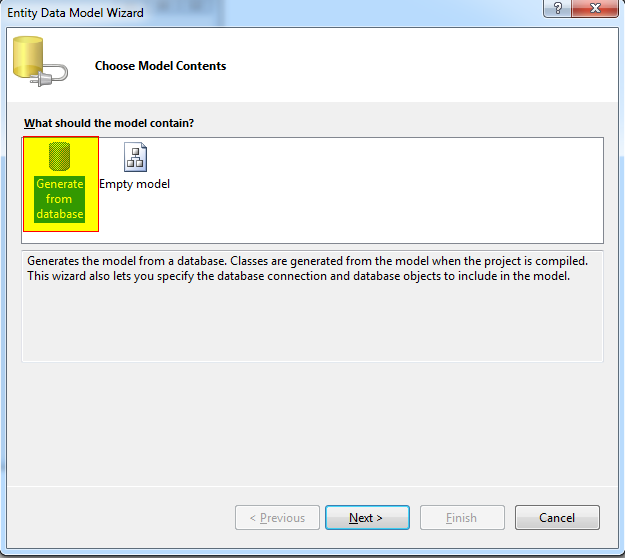
* Bước 2: Tạo project **Windows Forms Application**



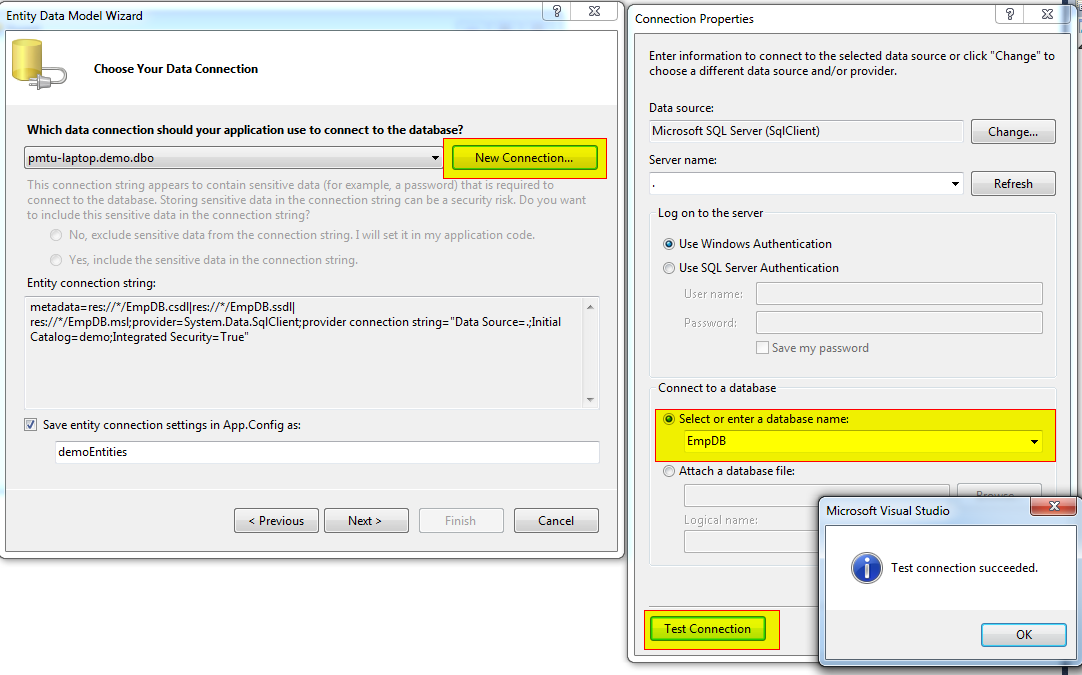
* Bước 3: Click phải project -> chọn Add New item và chọn hạng mục như hình dưới đây:



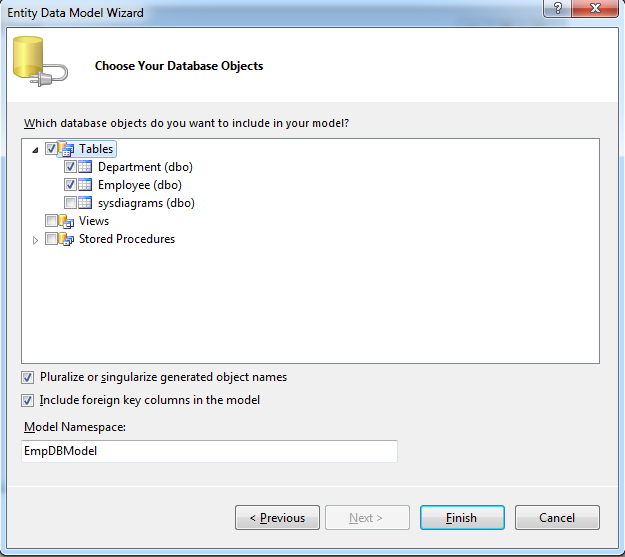
* Bước 4: Để phát sinh mô hình từ cơ sở dữ liệu chọn **Generate from database.** Các lớp trong mô hình này được phát sinh tương ứng với các quan hệ trong csdl.



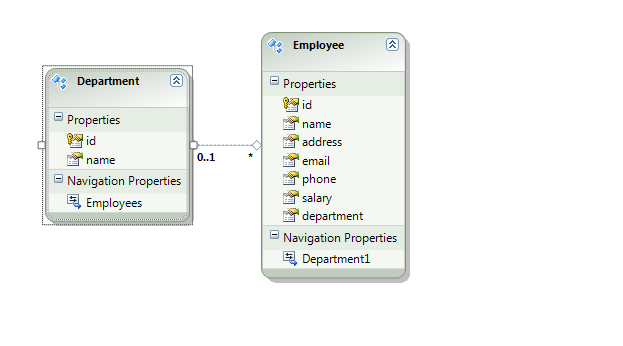
* Bước 5: Chọn cơ sở dữ liệu kết hợp với mô hình EF. Điền đầy đủ thông tin kết nối và kiểm tra kết nối csdl



* Bước 6: Chọn các đối tượng csdl mà chúng ta muốn gắn kết với mô hình EF



Sau đó mô hình EF sẽ phát sinh các lớp như sau:



Chúng ta đã tạo thành công mô hình EF, từ giờ chúng ta sẽ thao tác với các đối tượng csdl thông qua mô hình này. Ngôn ngữ mà mô hình này truy vấn chính là LinQ.

LINQ có thể truy vấn đến các đối tượng như: Strings, String Array, Files, List,…. ; tài liệu XML; hoặc **LinQ to SQL** sẽ phiên dịch sang câu truy vấn sql và gửi chúng đến csdl để thực thi. Trong phần này chủ yếu hướng dẫn các bạn dùng **LINQ to SQL.**

Để bắt đầu truy vấn, bạn phải khai báo một thể hiện của mô hình EF đã tạo trước đó. Sau đó xem đối tượng này như nguồn dữ liệu (data source) cần khai thác.

|  |
| --- |
| EmpDBEntities context = new EmpDBEntities(); |

* LINQ to SQL Writing Select Query

Sử dụng LINQ to SQL để truy vấn thông tin tương tự câu lệnh truy vấn **select** trong SQL. Ví dụ: Lấy thông tin id, name trong bảng Departments

|  |
| --- |
| var q = from p in context.Departments  select new  {  p.id,  p.name  }; |

Hoặc lấy tất cả các thông tin trong bảng Departments

|  |
| --- |
| var q = from p in context.Departments  select p; |

* LINQ to SQL Inner Join

Sử dụng phép kết nội bộ trong mô hình EF dùng join…….on………

Ví dụ: Kết bảng Department với bảng Employee

|  |
| --- |
| var q = from emp in context.Employees  join dep in context.Departments  on emp.department equals dep.id  select new  {  emp.id,  emp.name,  emp.address,  dep\_name=dep.name}; // *Đổi tên cột name bảng Department vì trùng tên cột với bảng Employee* |

Hoặc dùng left join như sau:

|  |
| --- |
| var q = from emp in context.Employees  join dep in context.Departments  on emp.department equals dep.id  into d from dept in d.DefaultIfEmpty()  select new  {  emp.id,  emp.name,  emp.address,  dep\_name=dept.name  }; |

Dòng lệnh được làm nổi bật trong ví dụ trên có ý nghĩa giữ thông tin bảng Employee, bảng Department nếu không có thông tin thì mặc định sẽ hiển thị rỗng (DefaultIfEmpty).

* LINQ to SQL LIKE Operator

Bạn có thể dùng các phương thức Contains(), StartsWith(), EndsWith() để thực thi phép toán like trong SQL

Ví dụ:

|  |
| --- |
| * like '%SearchString%' = Contains("SearchString") * like '%SearchString' = StartsWith("SearchString") * like 'SearchString%' = EndsWith("SearchString")   var q = from emp in context.Employees  where emp.address.Contains("hcm")  select new  {  emp.id,  emp.name,  emp.address  }; |

* LINQ to SQL Insert

Bạn có thể thêm một đối tượng tương ứng một record trong một quan hệ csdl dùng LINQ.

Ví dụ: Thêm một nhân viên mới ta dùng câu truy vấn SQL insert into ……values…

Với LINQ mọi thứ đơn giản như sau:

|  |
| --- |
| context.Employees.AddObject(e); |

Thêm một đối tượng nhân viên mới trên bộ nhớ tạm và

|  |
| --- |
| context.SaveChanges(); |

Phương thức SaveChanges() mới thực sự lưu trữ thông tin tại mức cơ sở dữ liệu.

Ví dụ

|  |
| --- |
| EmpDBEntities context = new EmpDBEntities();  Employee e = new Employee();  e.name = "nguyen van d";  e.address = "tp hcm";  e.email = "nvd@gmail.com";  e.phone = "0943345234";  e.salary = 450000;  e.department = 1;  context.Employees.AddObject(e);  context.SaveChanges(); |

* LINQ to SQL Update

Bạn có thể cập nhật thông tin của một đối tượng thông qua ví dụ sau:

Ví dụ: Cập nhật thông tin tên nhân viên cho nhân viên có mã số 1

* Đầu tiên tìm kiếm nhân viên có mã số 1

|  |
| --- |
| Employee e = (from p in context.Employees  where p.id == 1  select p).Single(); |

Phương thức Single() trả về **chỉ một** đối tượng

* Cập nhật thông tin tên nhân viên và cập nhật sự thay đổi trong csdl

|  |
| --- |
| e.name = "nguyen van a1";  context.SaveChanges(); |

* LINQ to SQL Delete

Bạn có thể xóa thông tin một đối tượng tương tự các bước như hành động cập nhật. Ví dụ: Xóa thông tin nhân viên có mã số 1

|  |
| --- |
| Employee e = (from p in context.Employees  where p.id == 1  select p).Single();  context.DeleteObject(e);  context.SaveChanges(); |

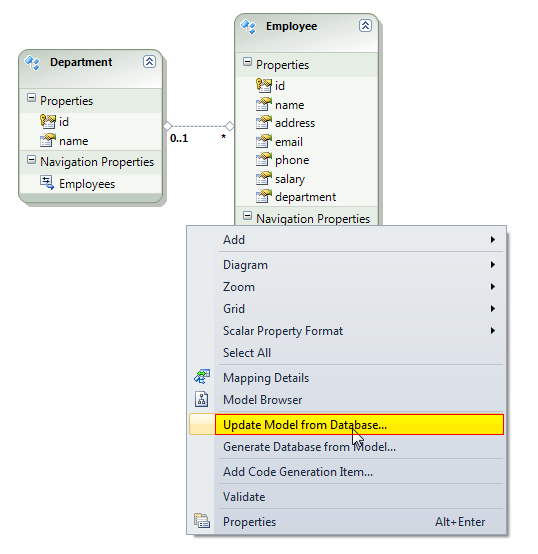
* LINQ to SQL Calling Stored Procedure

Đối tượng Stored Procedure được gắn kết với mô hình EF dưới dạng một phương thức. Ví dụ thêm một stored procedure dưới csdl và cập nhật trong mô hình EF như sau:

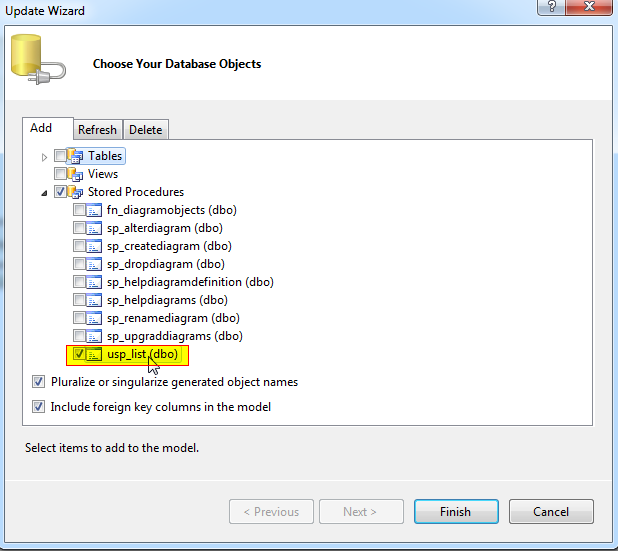
* Tạo stored procedure dưới csdl

|  |
| --- |
| create proc usp\_list  as begin  select \* from employee  end |

* Cập nhật đối tượng trong mô hình EF bằng cách click phải sơ đồ lớp và chọn **Update Model from Database**



* Chọn đối tượng cần gắn kết



* Bạn phải biên dịch lại project và bạn thấy chúng ta có một phương thức usp\_list() trong mô hình EL
* Gọi phương thức usp\_list()

|  |
| --- |
| EmpDBEntities context = new EmpDBEntities();  var q = context.usp\_list(); |

Ví dụ 2:

* Tạo SP có tham số

|  |
| --- |
| create proc usp\_insertDepartment  @name nvarchar(20)  as begin  insert into Department  values (@name)  end |

* Gọi SP từ mô hình EF

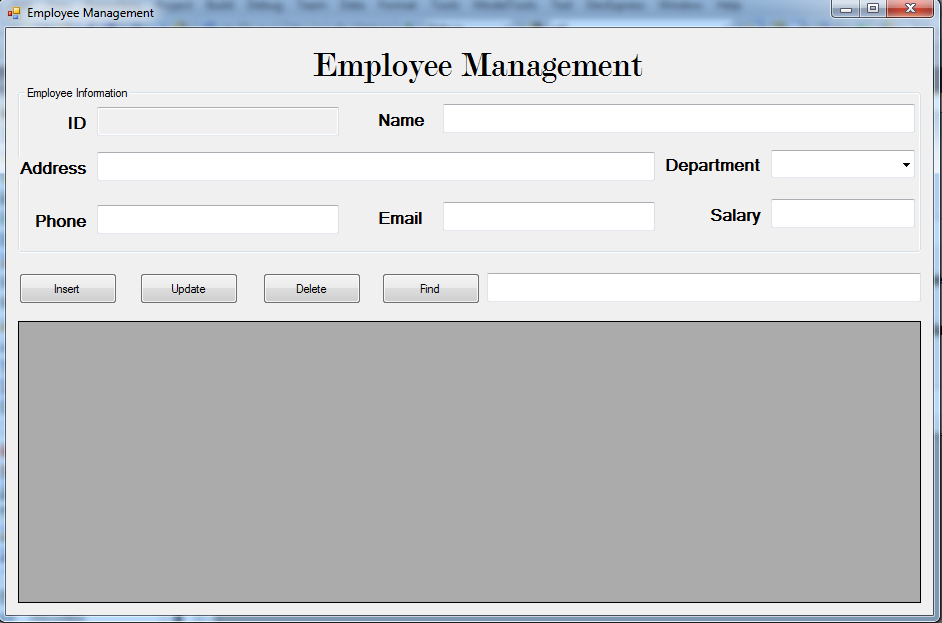
|  |
| --- |
| EmpDBEntities context = new EmpDBEntities();  context.usp\_insertDepartment("Hành chính"); |

# Ví dụ tổng hợp

Trong mục này sẽ hướng dẫn cho các bạn thực hiện các chức năng thêm, xóa, cập nhật và xem dữ liệu. Các chức năng tương ứng như: thêm một nhân viên, xóa một nhân viên dựa vào mã nhân viên, cập nhật nhân viên và xem danh sách nhân viên dùng gridview. Chức năng tìm kiếm nhân viên theo nhiều tiêu chí khác nhau.

Các bước thực hiện

* Bước 1: Tạp project Windows Forms Application -> Tạo giao diện như hình bên dưới



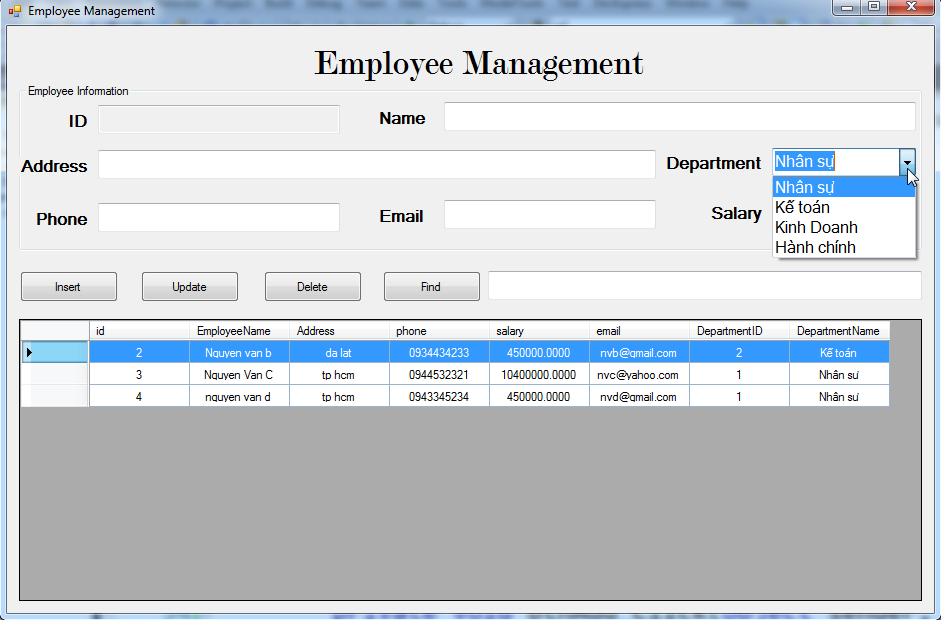
* Bước 2: Xây dựng 2 hàm: lấy danh sách các phòng ban hiển thị trên combobox **Department** ; lấydanh sách các nhân viên hiển thị trên GridView

|  |
| --- |
| public void loadDepartment()  {  EmpDBEntities context = new EmpDBEntities();  var q = from p in context.Departments  select p;  cboDepartment.DataSource = q;  cboDepartment.DisplayMember = "name";  cboDepartment.ValueMember = "id";  }  public void loadEmployee()  {  EmpDBEntities context = new EmpDBEntities();  var q = from em in context.Employees  join dep in context.Departments  on em.department equals dep.id  select new  {  id=em.id,  EmployeeName=em.name,  Address=em.address,  em.phone,  em.salary,  em.email,  DepartmentID=dep.id,  DepartmentName=dep.name  };  dgvEmployee.DataSource = q;  } |

Gọi 2 hàm này trong sự kiện này ngay khi form được load - form\_load()

|  |
| --- |
| private void Form1\_Load(object sender, EventArgs e1)  {  loadDepartment();  loadEmployee();    } |

Kết quả

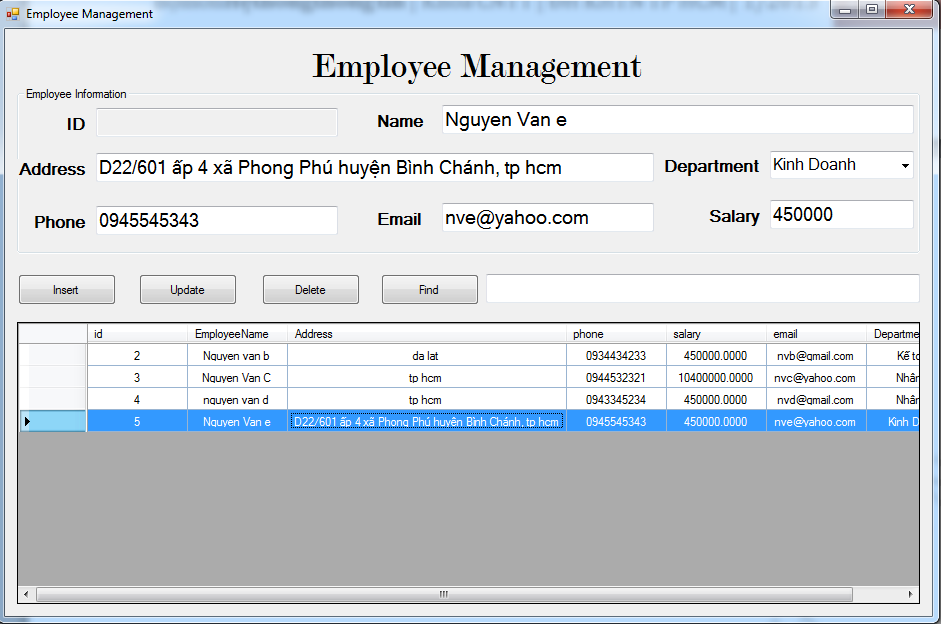


* Bước 3

Tạo sự kiện click trên nút nhấn **Insert** và viết mã lệnh thêm một nhân viên mới như sau:

|  |
| --- |
| private void btnAdd\_Click(object sender, EventArgs e)  {  /\*  \* Kiểm tra dữ liệu nhập hợp lệ từ người dùng  \* ............  \*/  EmpDBEntities context = new EmpDBEntities();  Employee emp = new Employee();  emp.name = txtName.Text;  emp.address = txtAddress.Text;  emp.email = txtEmail.Text;  emp.salary = decimal.Parse(txtSalary.Text);  emp.phone = txtPhone.Text;  emp.department = (int)cboDepartment.SelectedValue;  context.Employees.AddObject(emp);  context.SaveChanges();  MessageBox.Show("Insert Successfully!!!");  loadEmployee();  } |

Kết quả

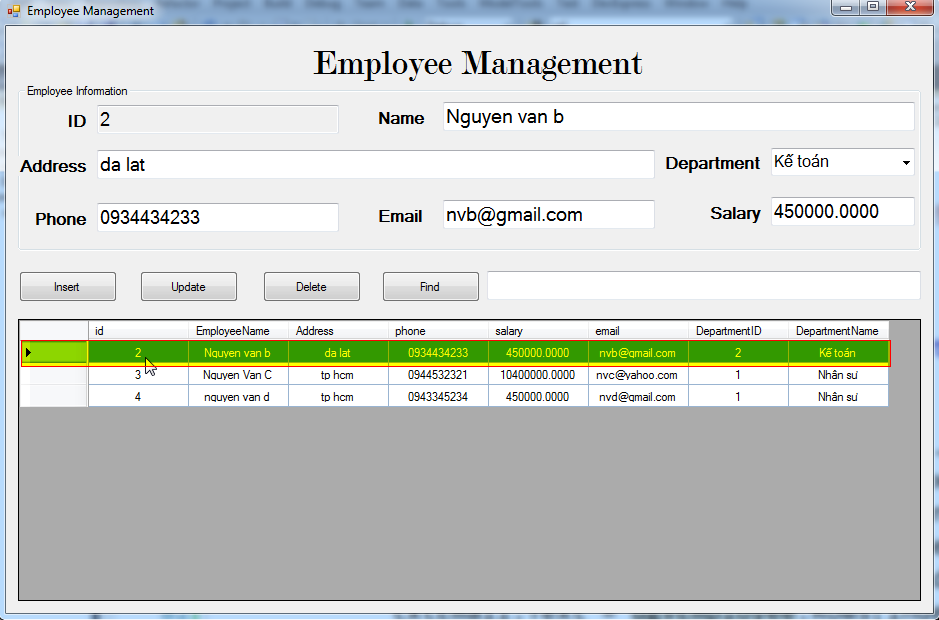


* Bước 4: Cập nhật một nhân viên

Tạo sự kiện cell click trên drigview , chọn nhân viên cần cập nhật thông tin, các thông tin này hiển thị trên các control tương ứng để người dùng chỉnh sửa.

|  |
| --- |
| private void dgvEmployee\_CellClick(object sender, DataGridViewCellEventArgs e)  {  int index = e.RowIndex;  txtID.Text = dgvEmployee.Rows[index].Cells[0].Value.ToString();  txtName.Text = dgvEmployee.Rows[index].Cells[1].Value.ToString();  txtAddress.Text = dgvEmployee.Rows[index].Cells[2].Value.ToString();  txtPhone.Text = dgvEmployee.Rows[index].Cells[3].Value.ToString();  txtSalary.Text = dgvEmployee.Rows[index].Cells[4].Value.ToString();  txtEmail.Text = dgvEmployee.Rows[index].Cells[5].Value.ToString();  cboDepartment.SelectedValue=int.Parse(dgvEmployee.Rows[index].Cells[6].Value.ToString()); } |

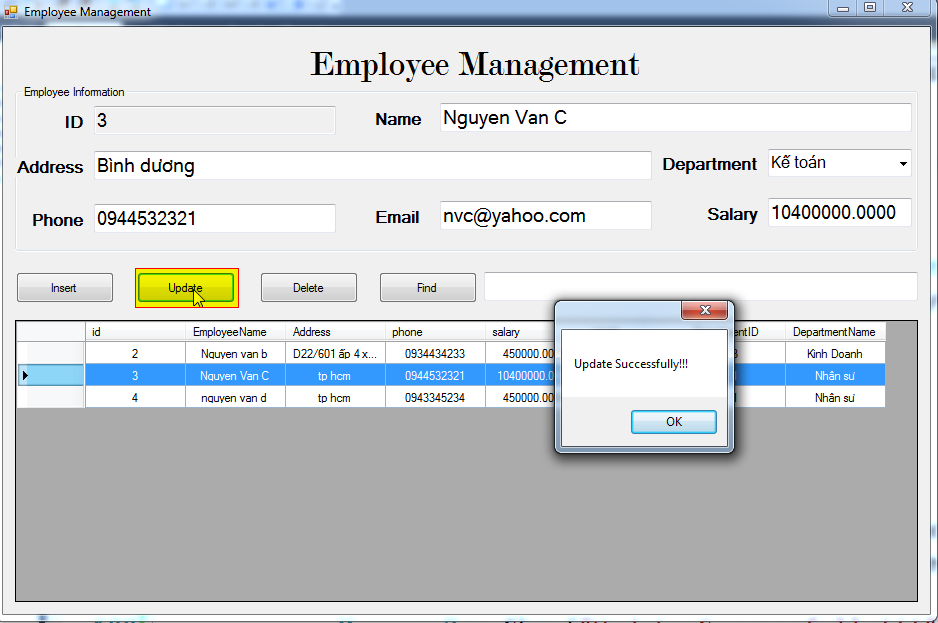
Kết quả



Tạo sự kiện click cho nút nhấn Update và viết mã lệnh cập nhật thông viên mới cho nhân viên.

|  |
| --- |
| private void btnUpdate\_Click(object sender, EventArgs e)  {  int id=int.Parse(txtID.Text);  EmpDBEntities context = new EmpDBEntities();  Employee emp = (from p in context.Employees  where p.id == id  select p).Single();  emp.name = txtName.Text;  emp.address = txtAddress.Text;  emp.email = txtEmail.Text;  emp.phone = txtPhone.Text;  emp.salary = decimal.Parse(txtSalary.Text);  emp.department = (int)cboDepartment.SelectedValue;  context.SaveChanges();  MessageBox.Show("Update Successfully!!!");  loadEmployee();  } |

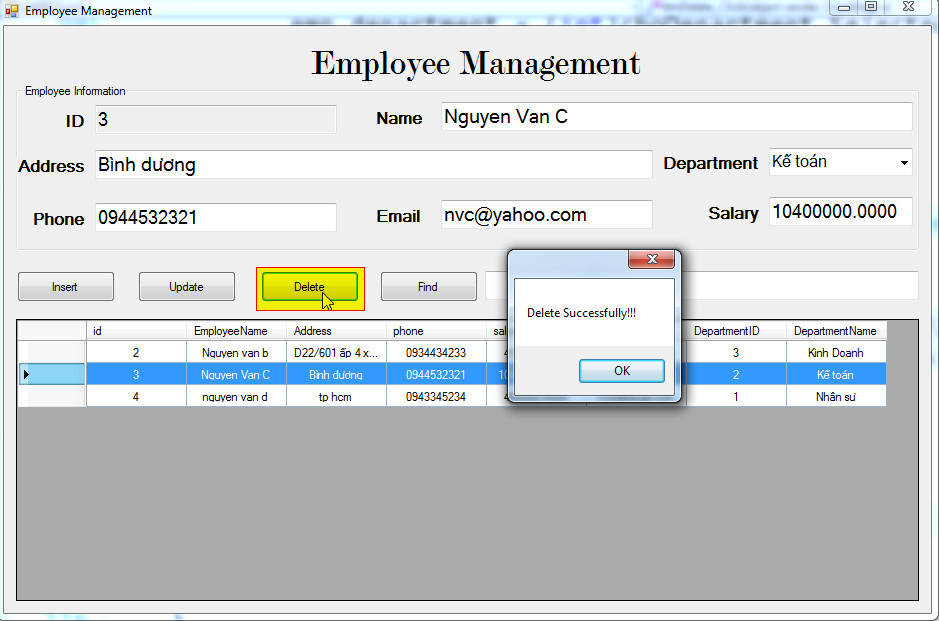
Kết quả



* Bước 5: Xóa một nhân viên dựa vào mã nhân viên, chức năng này tương tự như cập nhật một nhân viên, chúng ta bắt đầu tạo sự kiện click cho nút nhất delete và viết mã lệnh như sau:

|  |
| --- |
| private void btnDelete\_Click(object sender, EventArgs e)  {  int id = int.Parse(txtID.Text);  EmpDBEntities context = new EmpDBEntities();  Employee emp = (from p in context.Employees  where p.id == id  select p).Single();  context.Employees.DeleteObject(emp);  context.SaveChanges();  MessageBox.Show("Delete Successfully!!!");  loadEmployee();  } |

Kết quả

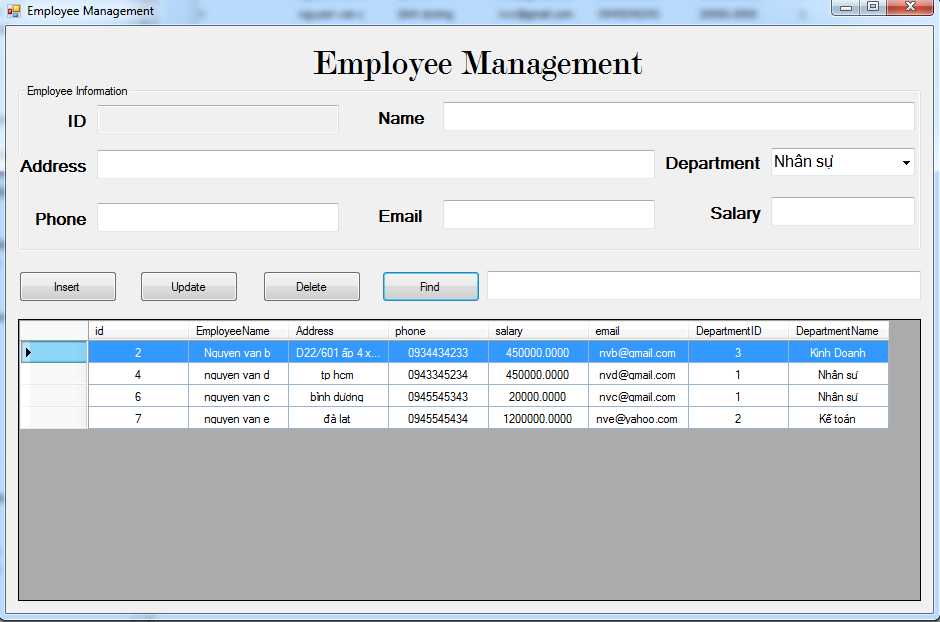


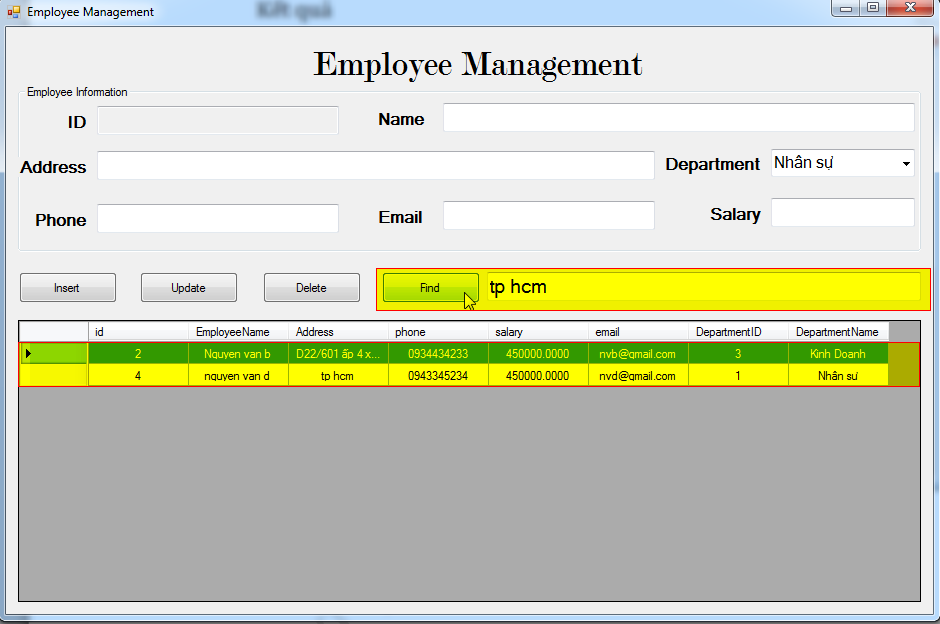
* Bước 6: Tìm kiếm nhân viên dựa vào các tiêu chí như: ID, name, address,email,phone,……

Tạo sự kiện click cho nút nhấn Find và viết mã lệnh tìm kiếm các nhân viên.

|  |
| --- |
| private void btnFind\_Click(object sender, EventArgs e)  {  string keyword = txtKeyWord.Text;  EmpDBEntities context = new EmpDBEntities();  var q = from em in context.Employees  join dep in context.Departments  on em.department equals dep.id  where em.name.Contains(keyword) ||  em.phone.Contains(keyword) ||  em.email.Contains(keyword) ||  em.address.Contains(keyword) ||  dep.name.Contains(keyword)  select new  {  id = em.id,  EmployeeName = em.name,  Address = em.address,  em.phone,  em.salary,  em.email,  DepartmentID = dep.id,  DepartmentName = dep.name  };  dgvEmployee.DataSource = q;  } |

Kết quả





# Vấn đề hiệu suất cho mô hình EF

Tìm hiểu thêm các chủ đề cải thiện hiệu suất cho mô hình EF, phần này chỉ cung cấp các link tham khảo

* <http://www.dotnet-tricks.com/Tutorial/entityframework/J8bO140912-Tips-to-improve-Entity-Framework-Performance.html>
* <http://www.farreachinc.com/blog/far-reach/2013/09/26/entity-framework-query-optimizations>
* <http://www.asp.net/web-forms/tutorials/continuing-with-ef/maximizing-performance-with-the-entity-framework-in-an-asp-net-web-application>
* <http://www.codeproject.com/Articles/247254/Improving-Entity-Framework-Query-Performance-Using>

…..

# Bài tập

Hãy làm lại bài ví dụ tổng hợp với mô hình 3 lớp và sử dụng mô hình Entity Framework, trong đó chỉnh sửa chức năng thêm và xóa theo cách dùng stored procedure kết hợp mô hình EF