BÀI 11: TRUY VẤN VÀ XỬ LÝ DỮ LIỆU VỚI ENTITY FRAMEWORK CORE

- ✓ LINQ
- ✓ Truy vấn, cập nhật dữ liệu sử dụng EF Core

1. LINQ

- **LINQ** (Language Integrated Query) - ngôn ngữ truy vấn tích hợp) là tên của một tập hợp các công nghệ dựa trên việc tích hợp các khả năng truy vấn trực tiếp vào ngôn ngữ C #.
- LINQ có thể dùng cho tất cả mọi nguồn dữ liệu.
- Trong LINQ, một hoạt động truy vấn hoàn chỉnh bao gồm: tạo nguồn dữ liệu, định nghĩa biểu thức truy vấn và thực hiện truy vấn trong câu lệnh foreach.

1. LINQ

- **Nguồn dữ liệu** được tổ chức như một chuỗi các phần tử cùng loại, dưới dạng tập hợp **IEnumerable** hoặc **IEnumerable<T>** như Array, List<T>, Stack . . .
- **Cú pháp truy vấn**

  Kiểu-biến-truy-vấn Tên-biến-truy-vấn = biểu-thức-truy-vấn;

- **Biến truy vấn**: là biến lưu trữ truy vấn, không lưu kết quả của truy vấn.

1. LINQ

- **Kiểu của biến truy vấn**:
  + Sử dụng kiểu ngầm định **var** để trình biên dịch suy luận ra kiểu của biến truy vấn (hoặc bất kỳ biến cục bộ nào khác) tại thời điểm biên dịch.
  + Hoặc khai báo tường minh kiểu của biến truy vấn để thể hiện mối quan hệ kiểu giữa biến truy vấn và mệnh đề select.

1. LINQ

- **Biểu thức truy vấn:**
  - Phải bắt đầu bằng mệnh đề **from** và phải kết thúc bằng mệnh đề **select** hoặc **group**.
  - Giữa mệnh đề from đầu tiên và mệnh đề select hoặc group cuối cùng, có thể chứa một hoặc nhiều mệnh đề tùy chọn sau: **where**, **orderby**, **join**, **let**
  - Có thể sử dụng into để cho phép kết quả của mệnh đề join hoặc group trở thành nguồn cho các mệnh đề truy vấn bổ sung trong cùng một biểu thức truy vấn.

1. LINQ

- **Mệnh đề from:** xác định nguồn dữ liệu mà truy vấn sẽ thực hiện. Nguồn dữ liệu là những phần tử trong lớp triển khai giao diện IEnumerable như: mảng, List . . .

from <biến-phạm-vi> in <nguồn-dữ-liệu>

Ví dụ, giả sử nguồn dữ liệu scores được khai báo như sau:

```
int[] scores = new int[] { 97, 92, 81, 60 };
```

Để truy vấn dữ liệu từ scores, mệnh đề from được viết như sau:

```
from score in scores
```

biến phạm vi

nguồn dữ liệu

1. LINQ

- **Mệnh đề select:** chỉ định các dữ liệu được lấy ra của câu lệnh LINQ

Ví dụ:

```
var scoreQuery = from score in scores
                 select score;
```

Câu lệnh trên có nghĩa: với mỗi phần tử (được đại diện bởi biến phạm vi score) trong nguồn dữ liệu scores, lấy ra phần tử đó

1. LINQ

- **Mệnh đề where:** được sử dụng để lọc dữ liệu

  where biểu-thức-logic

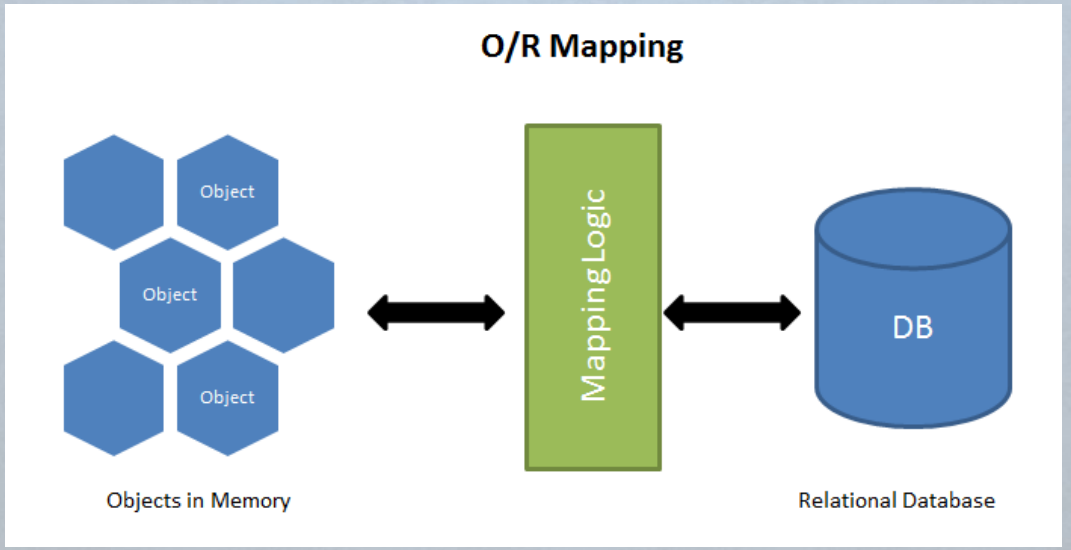
Ví dụ:

```
var scoreQuery = from score in scores
                 where score > 80
                 select score;
```

Câu lệnh trên chỉ lấy ra những phần tử >80 trong nguồn dữ liệu scores.

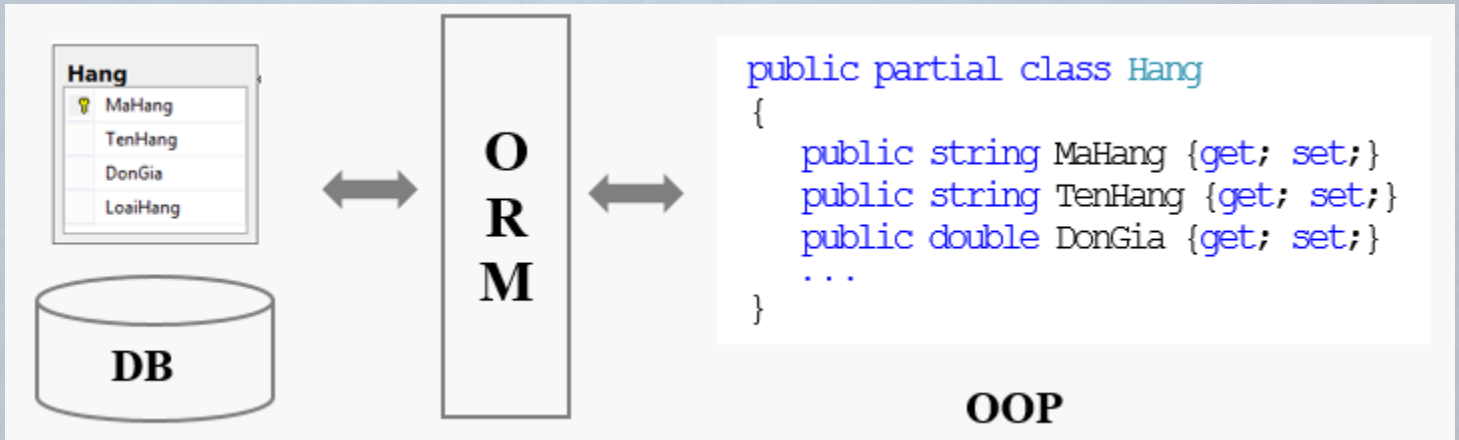
2. ORM VÀ EF

ORM (**O**bject **R**elational **M**apping) là một kỹ thuật thực hiện ánh xạ CSDL sang các đối tượng trong các ngôn ngữ lập trình hướng đối tượng



2. ORM VÀ EF

ORM . . .



- Các bảng tương ứng các class, mối ràng buộc giữa các bảng tương ứng quan hệ giữa các class.
- Được cài đặt trong tất cả các ngôn ngữ hiện đại như: c#, java, php, node.js, …

2. ORM VÀ EF

EF(Entity Framework) là cách thực hiện của ORM trong .NET

- **Các tính năng:**
  - Cung cấp cơ chế để truy cập và lưu trữ dữ liệu vào CSDL
  - Cung cấp các dịch vụ như theo dõi sự thay đổi của dữ liệu, dịch truy vấn . . .
  - Sử dụng LINQ để truy vấn, thêm, sửa, xóa dữ liệu
  - Tăng khả năng bảo trì và mở rộng cho ứng dụng

3. EF CORE

- **EF Core** (Entity Framework Core) là phiên bản nhẹ, có thể mở rộng, mã nguồn mở và đa nền tảng của EF.
- Cho phép làm việc với cơ sở dữ liệu sử dụng các đối tượng .NET
- Loại bỏ hầu hết các lệnh truy cập dữ liệu thường phải viết.
- Có thể truy cập nhiều csdl khác nhau như SQL Server, Sqllite, MySQL … thông qua các thư viện plug-in được gọi là **Database Providers**

3. EF CORE

- EF Core sử dụng **Model** (mô hình) để thực hiện việc truy cập dữ liệu
- Một model được tạo thành từ các lớp thực thể (**Entities**) và một đối tượng context đại diện cho một phiên làm việc với cơ sở dữ liệu (**DbContext**)
- DbContext cho phép truy vấn và cập nhật dữ liệu.

3. EF CORE

- Các cách tiếp cận phát triển model:
  - + Tạo một model từ cơ sở dữ liệu hiện có.
  - + Viết code model để tạo cơ sở dữ liệu.
- **Truy vấn dữ liệu:** các instance của các lớp thực thể được truy xuất từ cơ sở dữ liệu bằng cách sử dụng truy vấn LINQ
- **Cập nhật dữ liệu**: dữ liệu được tạo, xóa và sửa đổi trong cơ sở dữ liệu bằng cách sử dụng các instances của các lớp thực thể.

4. TRUY VẤN DỮ LIỆU SỬ DỤNG EF CORE

Các bước để truy vấn dữ liệu sử dụng EF Core

B1. Thêm thư viện tùy mục đích sử dụng (Microsoft.EntityFrameworkCore.SqlServer và Microsoft.EntityFrameworkCore.Tools)

B2. Tạo model

B3. Truy vấn dữ liệu sử dụng LINQ

4. TRUY VẤN DỮ LIỆU SỬ DỤNG EF CORE

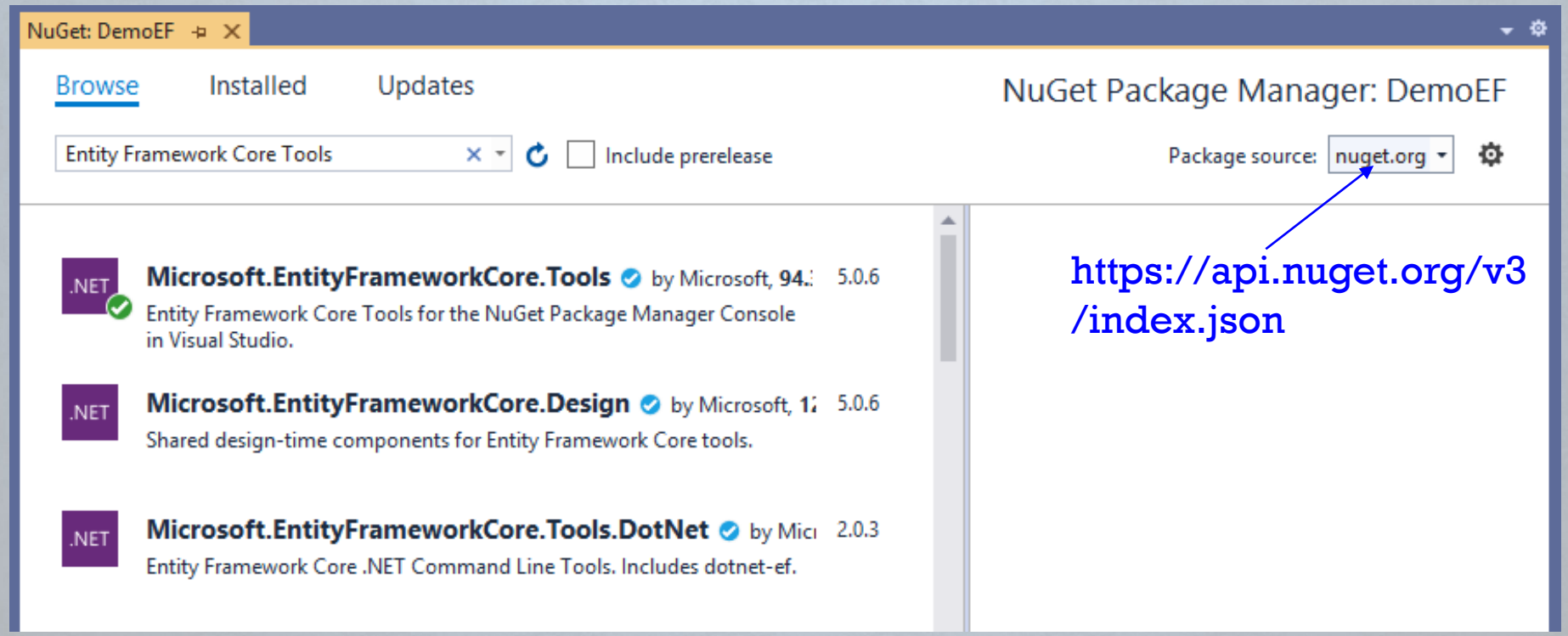
4.1. Thêm các package cần thiết

Thêm package bằng Manage NuGet packages for Solution

- Chọn menu Tools → NuGet Package Manager → Manage NuGet packages for Solution . . .
- Trong cửa sổ NuGet –Solution → chọn tab Browse → nhập tên Package cần cài đặt → trong danh sách hiển thị chọn Package muốn cài → trong phần bên phải của cửa sổ NuGet –Solution → chọn tên Project → nhấn Install để cài đặt

4. TRUY VẤN DỮ LIỆU SỬ DỤNG EF CORE

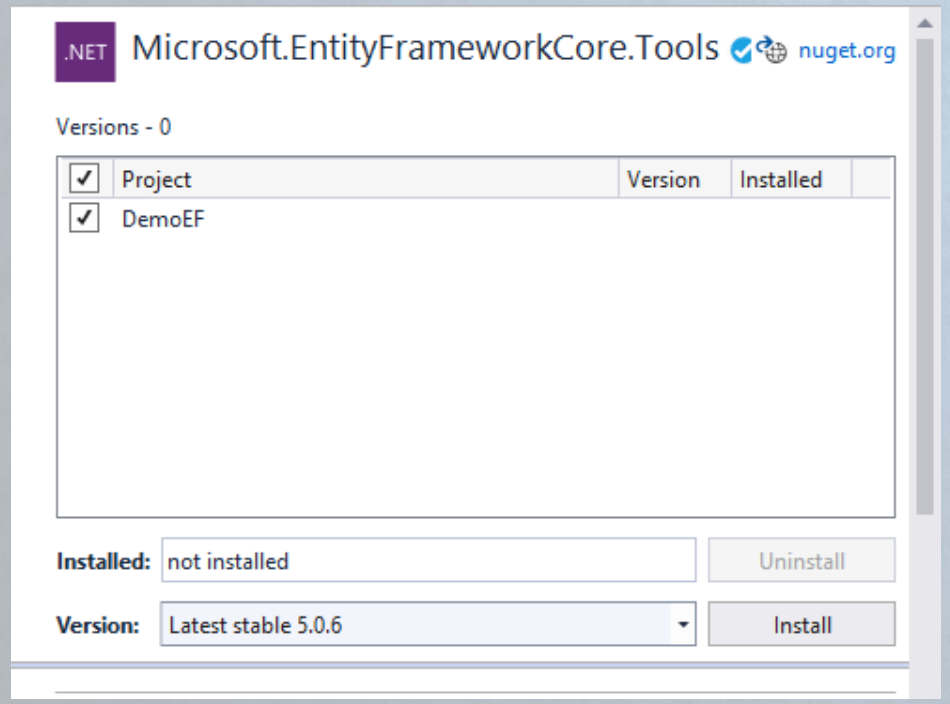
Thêm package bằng Manage NuGet packages for Solution . . .



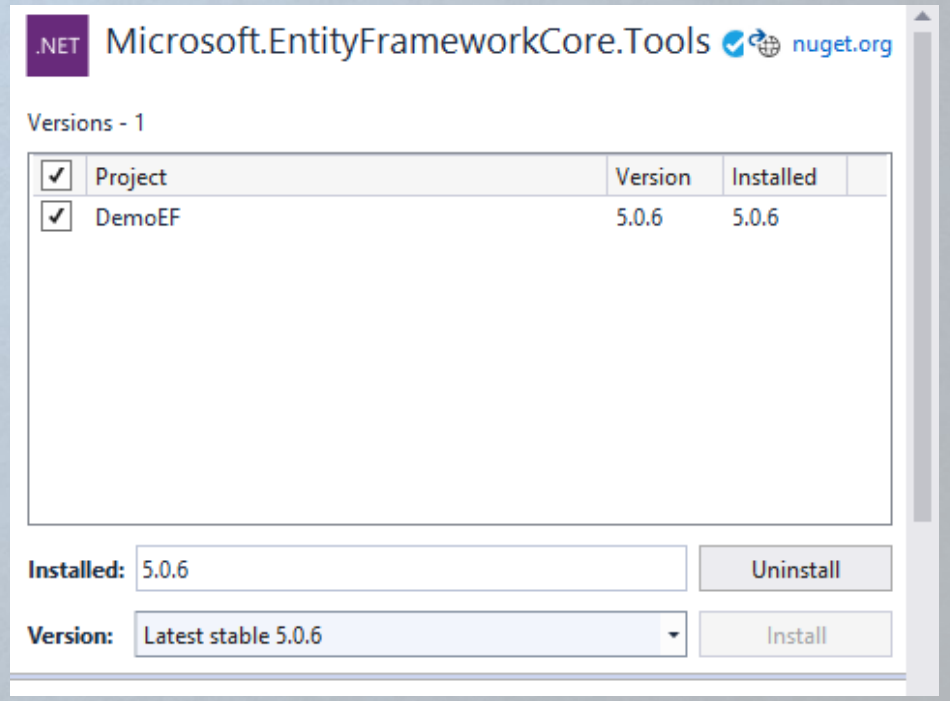
Tìm Package muốn cài đặt

4. TRUY VẤN DỮ LIỆU SỬ DỤNG EF CORE

Thêm package bằng Manage NuGet packages for Solution . . .



Chọn Project muốn thêm thư viện



Thêm thư viện thành công

4. TRUY VẤN DỮ LIỆU SỬ DỤNG EF CORE

Thêm package bằng Package Manager Console

- Chọn menu Tools → NuGet Package Manager → Package Manager Console
- Nhập lệnh theo cú pháp sau vào cửa sổ Package Manager Console

  PM> Install-Package tên-package –Version tên-phiên-bản

  + Nếu không có tùy chọn –Version, mặc định cài phiên bản mới nhất

Ví dụ:

PM> Install-Package Microsoft.EntityFrameworkCore.SqlServer -Version 5.0.6

4. TRUY VẤN DỮ LIỆU SỬ DỤNG EF CORE

4.2. Tạo model từ cơ sở dữ liệu hiện có

Thực hiện các bước sau:

B1. Chuẩn bị cơ sở dữ liệu

B2. Trong cửa sổ Package Manager Console thực hiện lệnh

```
Scaffold-DbContext "chuỗi-kết-nối" tên-DataProvider [–OutputDir tên-thư-mục-chứa-các-lớp-ánh-xạ]
```

Sau khi thực thi thành công ta sẽ có lớp DbContext và các lớp thực thể ánh xạ từ cơ sở dữ liệu trong thư mục chỉ định

4. TRUY VẤN DỮ LIỆU SỬ DỤNG EF CORE

4.2. Tạo model từ cơ sở dữ liệu hiện có . . .

Chuỗi kết nối có cấu trúc như sau:

Tên Server CSDL

Tên cơ sở dữ liệu

```
"Data Source=MyPC;Initial Catalog=QLBanHang;  User
ID=sa;Password=123456
```

Tên tài khoản và mật khẩu đăng nhập

4. TRUY VẤN DỮ LIỆU SỬ DỤNG EF CORE

4.2. Tạo model từ cơ sở dữ liệu hiện có . . .

Ví dụ: lệnh sau tạo model cho csdl QLBanHang trên SQL Server, đăng nhập vào SQL Server bằng tài khoản Window (Windows Authentication), thư mục để lưu các lớp ánh xạ là Models

```
PM> Scaffold-DbContext "Data Source=MyPC;Initial Catalog
=QLBanHang;Integrated Security=True"
Microsoft.EntityFrameworkCore.SqlServer –OutputDir Models
```

4. TRUY VẤN DỮ LIỆU SỬ DỤNG EF CORE

Lớp DbContext:

- Là một phần không thể thiếu của EF, là lớp chịu trách nhiệm tương tác với dữ liệu dưới dạng các đối tượng.
- Một thể hiện của DbContext đại diện cho một phiên làm việc với csdl, nó được sử dụng để truy vấn dữ liệu và cập nhật các thể hiện của các lớp thực thể vào csdl
- Lớp DataContext có các thuộc tính để biểu diễn mỗi lớp thực thể được mô hình hóa từ csdl

4. TRUY VẤN DỮ LIỆU SỬ DỤNG EF CORE

Lớp DbContext . . .

- Ví dụ: giả sử ta có csdl QLBanHang, sau khi thực hiện lệnh tạo model ta có lớp QLBanHangContext. Khai báo thể hiện của lớp QLBanHangContext như sau:

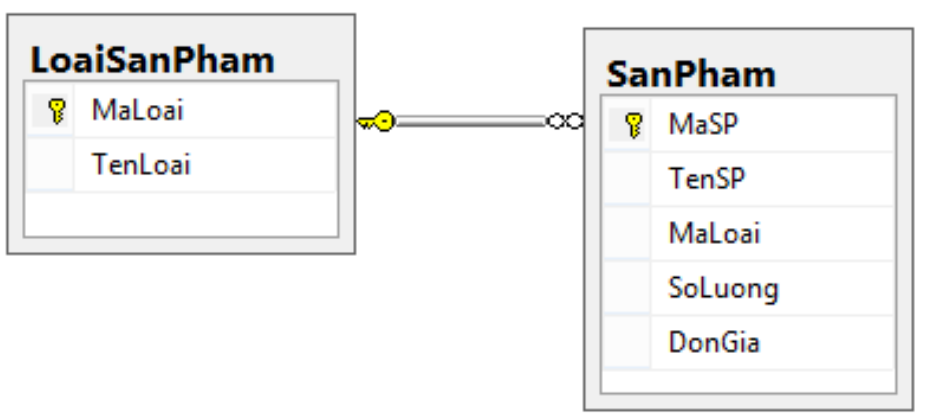
```
QLBanHangContext db = new QLBanHangContext();
```

db sẽ đại diện cho 1 phiên làm việc với csdl QLBanHang

4. TRUY VẤN DỮ LIỆU SỬ DỤNG EF CORE

Lớp thực thể (Entity class):

- Là các lớp ánh xạ vào các bảng bên trong một csdl. Các thuộc tính của lớp ánh xạ vào các cột trong bảng. Mỗi thể hiện của một lớp thực thể biểu diễn 1 dòng trong bảng



Lược đồ csdl QLBanHang

- Ví dụ: Lớp thực thể SanPham ánh xạ vào bảng SanPham trong csdl. Các thuộc tính như MaSP, TenSP … ánh xạ các cột MaSP, TenSP… trong bảng sản phẩm. Mỗi một instance của lớp SanPham biểu diễn 1 dòng trong bảng SanPham

4. TRUY VẤN DỮ LIỆU SỬ DỤNG EF CORE

Quan hệ giữa các lớp thực thể: được tự động tạo ra dựa trên các mối quan hệ primary key/foreign key trong CSDL, khi đó các thuộc tính tương ứng sẽ được thêm vào các lớp thực thể.

- Ví dụ: quan hệ giữa lớp thực thể LoaiSanPham và SanPham được tạo dựa trên quan hệ 1–nhiều giữa bảng LoaiSanPham và bảng SanPham. Mối quan hệ này làm lớp thực thể SanPham có thêm thuộc tính “LoaiSanPham”, dùng để truy cập vào thực thể LoaiSanPham của một SanPham. Lớp LoaiSanPham cũng có thêm thuộc tính “SanPhams”, đây là một tập hợp cho phép ta lấy ra tất cả các SanPham có trong LoaiSanPham đó.

4. TRUY VẤN DỮ LIỆU SỬ DỤNG EF CORE

Truy vấn dữ liệu sử dụng LINQ

Ví dụ: lấy về tất cả các dòng trong bảng LoaiSanPham

```
QLBanHangDataContext db=new QLBanHangDataContext();
...
var query=from lsp in db.LoaiSanPhams
          select lsp;
//Hiển thị dữ liệu lên data grid
dgvLoaiSanPham.ItemsSource = query.ToList();;
```

4. TRUY VẤN DỮ LIỆU SỬ DỤNG EF CORE

- Dùng mệnh đề where lấy về một tập các đối tượng thỏa mãn điều kiện

Ví dụ:

```
//lấy các loại sản phẩm có tên loại bắt đầu bằng Đ
var loaiSanPhamQuery=
                    from lsp in db.LoaiSanPhams
                    where lsp.TenLoai.StartsWith("Đ")
                    select lsp;
```

4. TRUY VẤN DỮ LIỆU SỬ DỤNG EF CORE

- Sử dụng mối quan hệ giữa các lớp thực thể trong truy vấn

Ví dụ:

```
//lấy ra loại sản phẩm có từ 2 sản phẩm trở lên
var loaiSanPhamQuery=from lsp in db.LoaiSanPhams
                     where lsp.SanPhams.Count>=2
                     select lsp;
```

5. CẬP NHẬT DỮ LIỆU

Change Tracking và DataContext.SaveChanges

- Ta đã dùng các truy vấn LINQ để lấy dữ liệu từ cơ sở dữ liệu bằng cách dùng lớp Context.
- Ví dụ: Biểu thức LINQ sau lấy về các đối tượng loại sản phẩm có nhiều hơn 2 sản phẩm

```
var loaiSanPhamQuery =
                    from lsp in db.LoaiSanPhams
                    where lsp.SanPhams.Count>=2
                    select lsp;
```

5. CẬP NHẬT DỮ LIỆU

Change Tracking và DataContext.SaveChanges . . .

- Khi ta thực hiện truy vấn và lấy về các đối tượng, LINQ sẽ lưu lại vết của tất cả các thay đổi mà ta thực hiện trên các đối tượng đó (gọi là change tracking).
- Sau khi đã cập nhật các đối tượng lấy từ LINQ, gọi phương thức ”SaveChanges()” của lớp Context để lưu các thay đổi lên CSDL.

5. CẬP NHẬT DỮ LIỆU

Thực hiện các bước sau để thêm bản ghi mới vào CSDL:

- Tạo (các) đối tượng thuộc lớp thực thể muốn thêm
- Gán giá trị cho thuộc tính của (các) đối tượng
- Thêm đối tượng vào tập hợp đối tượng tương ứng sử dụng phương thức **Add()** hoặc **AddRange()**của lớp Context
- Thực thi phương thức **SaveChanges()** của lớp Context để cập nhật các thay đổi vào csdl

5. CẬP NHẬT DỮ LIỆU

Ví dụ: thêm một loại sản phẩm mới vào csdl

```
//tạo đối tượng loại sản phẩm có thuộc tính do user nhập
LoaiSanPham lspMoi = new LoaiSanPham();
lspMoi.MaLoai = txtMa.Text;
lspMoi.TenLoai = txtTen.Text;
//2. Thêm vào tập hợp LoaiSanPhams
db.LoaiSanPhams.Add(lspMoi);
//3. Lưu thay đổi vào csdl
db.SaveChanges();
```

5. CẬP NHẬT DỮ LIỆU

Thực hiện các bước sau để sửa dữ liệu:

- Sử dụng LINQ lấy ra các đối tượng muốn sửa dữ liệu
- Sửa thông tin của các đối tượng
- Thực thi phương thức **SaveChanges()** của lớp Context để cập nhật các thay đổi vào csdl

5. CẬP NHẬT DỮ LIỆU

Ví dụ: sửa thông tin của loại sản phẩm có mã trong text box txtMa

```
//1. lấy ra sản phẩm muốn sửa
var spSua = db.LoaiSanPhams.SingleOrDefault(
                        lsp => lsp.MaLoai == txtMa.Text);
//2. cập nhật thông tin của sản phẩm
spSua.TenLoai = txtTen.Text;
//3. Lưu thay đổi vào csdl
db.SaveChanges();
```

5. CẬP NHẬT DỮ LIỆU

Thực hiện các bước sau để xóa dữ liệu:

- Sử dụng LINQ lấy ra (các) đối tượng muốn xóa
- Xóa (các) đối tượng khỏi tập hợp tương ứng của lớp Context sử dụng phương thức **Remove()** hoặc **RemoveRange()**
- Thực thi phương thức **SubmitChanges()** của DataContext để cập nhật các thay đổi vào csdl

5. CẬP NHẬT DỮ LIỆU

Ví dụ: xóa loại sản phẩm có mã trong text box txtMa

```
//1. lấy ra sản phẩm muốn xóa
var spSua = db.LoaiSanPhams.SingleOrDefault(lsp =>
lsp.MaLoai == txtMa.Text);
//2. xóa đối tượng khỏi tập hợp
db.LoaiSanPhams.Remove(spSua);
//3. lưu thay đổi vào csdl
db.SaveChanges();
```