



Chương 4: Kết nối CSDL - Entity Framework

Giảng viên: Vũ Minh Sang
Phòng: E9.4
E-mail: sangvm@uit.edu.vn



Entity Framework

Object Relational Mapping

- 📖 Object Relational Mapping (ORM) là kỹ thuật cho phép ứng dụng xây dựng theo hướng đối tượng làm việc với các hệ quản trị CSDL quan hệ (RDBMS)
- 📖 Cho phép ánh xạ hai chiều: class với cấu trúc bảng, object với dòng trong bảng, thuộc tính với các cột của bảng và tham chiếu sang object với quan hệ của các bảng.
- 📖 Truy vấn SQL được ORM tự động sinh và thực thi.
- 📖 ORM xử lý việc không phù hợp (khác biệt) giữa lập trình hướng đối tượng và CSDL quan hệ:
 - Kiểu dữ liệu của .NET và RDBMS.
 - Cách thức lưu trữ dữ liệu (đối tượng với bảng) trong bộ nhớ.
 - Sự khác biệt về mối quan hệ giữa các bảng và giữa các đối tượng

Entity Framework

- 📖 Là một ORM.
- 📖 Thể hiện CSDL dưới dạng tập hợp các object.
 - Cả CSDL được thể hiện bằng lớp con của lớp `DbContext`.
 - Mỗi bảng dữ liệu là một object của lớp `DbSet<T>`.
 - Mỗi hàng trong bảng là một object con của object bảng dữ liệu
 - Mỗi cột của bảng là thuộc tính của object.
- 📖 Có thể truy vấn dữ liệu bằng LINQ; thực hiện các truy vấn CRUD bằng code C# không cần dùng câu lệnh SQL.
- 📖 Thực hiện các thao tác với cấu trúc của CSDL (tạo CSDL, tạo bảng, thay đổi cấu trúc của bảng...) dễ dàng và ko mất dữ liệu

Entity Framework



Ưu điểm:

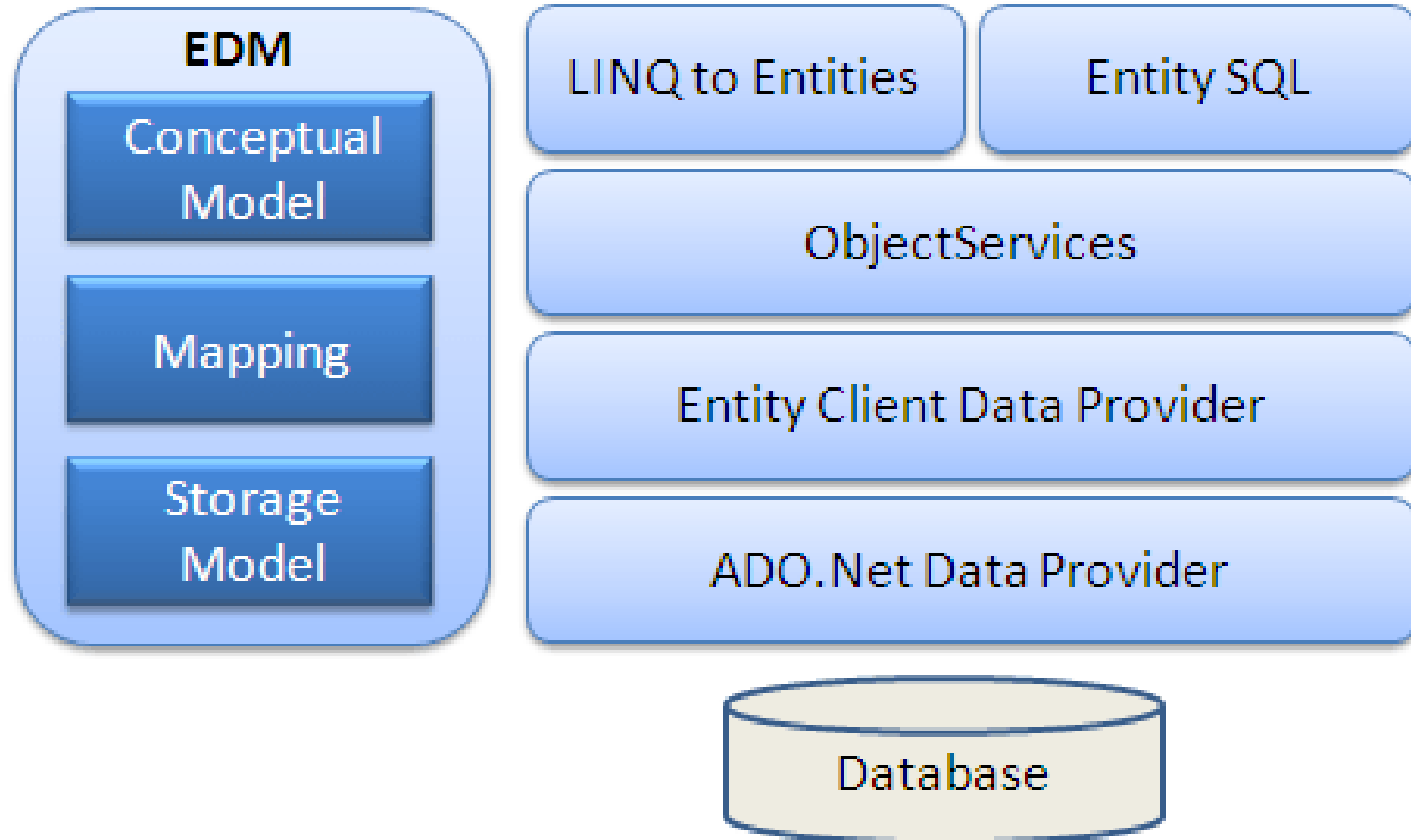
- Tăng năng suất hoạt động của ứng dụng (do giảm số lượng code).
- Dễ dàng bảo trì (ở cả phía ứng dụng và CSDL).



Nhược điểm:

- Thường phức tạp trong việc thiết kế.
- Giảm hiệu suất hoạt động khi so sử dụng trực tiếp bằng ADO.NET.

Kiến trúc Entity Framework



Kiến trúc Entity Framework

📖 Được chia là 2 giai đoạn:

- Xây dựng khối EDM (Entity Data Model)
- Truy vấn và xử lý dữ liệu.

📖 Thành phần của EDM gồm:

- *Conceptual Model*: chứa model class và quan hệ của class; độc lập với thiết kế bảng CSDL (là thành phần sử dụng trực tiếp trong ứng dụng)
- *Storage Model*: là cấu trúc của CSDL gồm: bảng, views, stored procedures, mối quan hệ và khóa.
- *Mapping*: thông tin về cách để Conceptual kết nối với Storage Model

📖 Xây dựng EDM, có 4 cách tiếp cận:

- Database-first
- Model-first
- Code-first
- Code-first from Database

Kiến trúc Entity Framework



Truy vấn và xử lý dữ liệu:

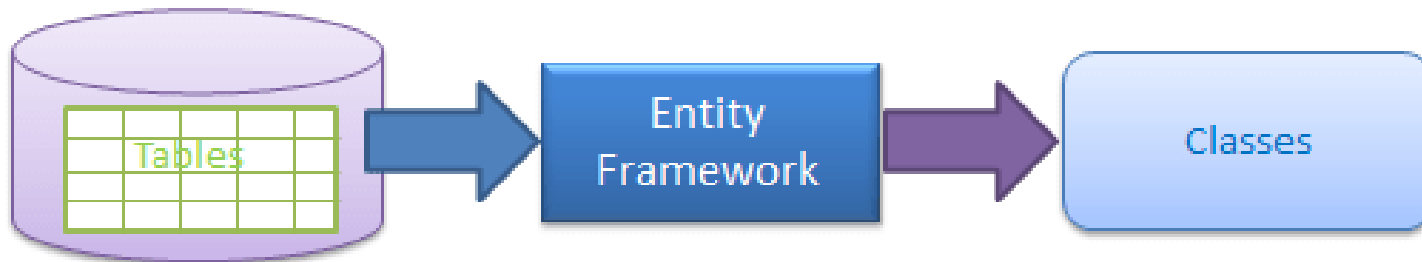
- *LINQ to Entities*: là ngôn ngữ để truy vấn tới object model; kết quả trả về: các đối tượng được định nghĩa ở Conceptual model.
- *Object Services*: theo dõi và ghi dữ liệu.
- *ADO.Net Data Provider*: giao tiếp với CSDL sử dụng chuẩn ADO.Net.

Xây dựng EDM



Database-First:

- Xuất hiện đầu tiên, trong EF.
- EF Wizard sẽ tạo ra Model và Code từ CSDL, stored procedures hoặc views có sẵn.
- Chỉnh sửa lại model (domain classes) theo nhu cầu thông qua giao diện đồ họa (visual design).
- Như vậy: từ Storage Model có sẵn, EF sẽ tạo ra Conceptual Model và Mapping.
- Nên dùng khi có sẵn CSDL.



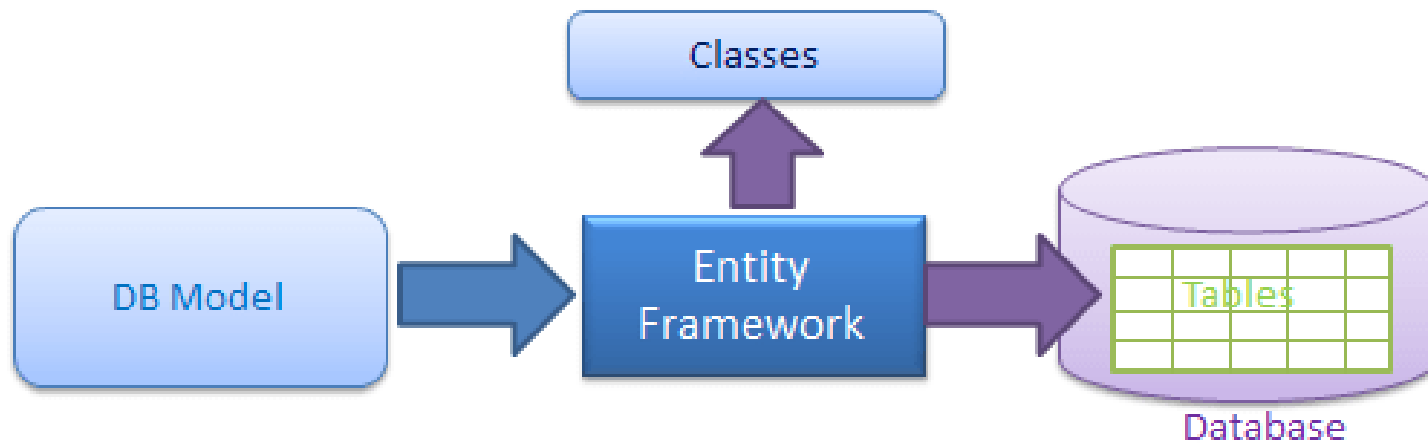
Generate Data Access Classes for Existing Database

Xây dựng EDM



Model-First:

- Xuất hiện từ EF4.
- Thiết kế mô hình CSDL (model) trước bằng giao diện có sẵn.
- EF sẽ tạo ra các lớp model tương ứng và mã SQL để tạo CSDL.
- Như vậy: EF sẽ tạo ra cả ba thành phần của EDM



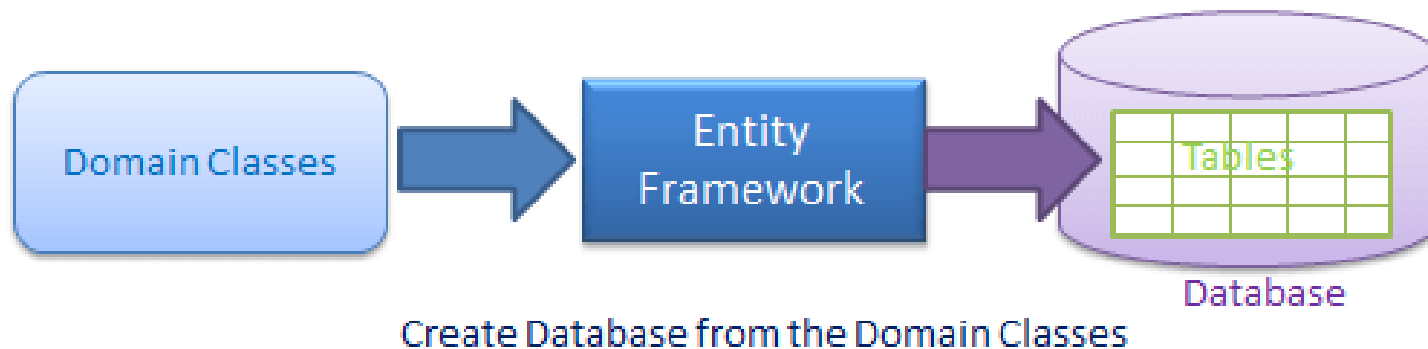
Create Database and Classes from the DB Model design

Xây dựng EDM



Code-First:

- Xuất hiện từ EF4.1.
- Xây dựng các lớp model (domain classes) theo hướng đối tượng.
- EF sẽ tạo ra một CSDL phù hợp
- Như vậy: cần phải viết code Conceptual Model, và EF sẽ tạo ra Mapping và Storage Model.







Xây dựng EDM

Code-First from Database

- Từ CSDL có sẵn, EF sẽ tạo ra Conceptual Model và Mapping.
- Kết quả nhận được là code của các domain class; không có giao diện thiết kế như database first.

Xây dựng EDM – Code-First

-  Cài đặt Entity Framework cho project
-  Tạo CSDL từ class trong C#
-  Tạo bản ghi mới vào bảng
-  Truy vấn dữ liệu với LINQ

Xây dựng EDM – Code-First



Cài đặt Entity Framework cho project

- Cài đặt Entity Framework bằng NuGet Package Manager
- B1: Click chuột phải vào tên Project => Manage NuGet Packages
- B2: Ở cửa sổ Manage NuGet Packages chọn tab Browse.
- B3: Gõ từ khóa Entity Framework ở ô tìm kiếm.
- B4: Chọn thư viện Entity Framework và Install

Xây dựng EDM – Code-First

The screenshot shows the NuGet Package Manager interface for a project named 'EF_HelloCodeFirst'. At the top, there are tabs for 'Browse', 'Installed', and 'Updates'. The 'Browse' tab is active, showing a search bar with 'entity framework' and a search icon. To the right of the search bar is a checkbox for 'Include prerelease' and a 'Package source' dropdown set to 'nuget.org'. Below the search bar, a list of packages is displayed. The first package is 'EntityFramework' by Microsoft, with 85.7M downloads and version 6.4.0. The description states: 'Entity Framework 6 (EF6) is a tried and tested object-relational mapper for .NET with many years of feature development and stabilization.' Below this are two other packages: 'Microsoft.EntityFrameworkCore.Abstractions' and 'Microsoft.EntityFrameworkCore.Analyzers', both by Microsoft, with 60 downloads and version 3.1.3. The descriptions for these are: 'Provides abstractions and attributes that are used to configure Entity Framework Core' and 'CSharp Analyzers for Entity Framework Core.' On the right side of the interface, the details for the 'EntityFramework' package are shown. It includes the package icon, name, version (6.4.0), and an 'Install' button. Below this is an 'Options' section and a 'Description' section, which repeats the description of Entity Framework 6 (EF6) as a tried and tested object-relational mapper for .NET.

Browse Installed Updates

entity framework ☐ Include prerelease

Package source: nuget.org

EntityFramework by Microsoft, 85.7M downloads v6.4.0
Entity Framework 6 (EF6) is a tried and tested object-relational mapper for .NET with many years of feature development and stabilization.

Microsoft.EntityFrameworkCore.Abstractions by Microsoft, v3.1.3
Provides abstractions and attributes that are used to configure Entity Framework Core

Microsoft.EntityFrameworkCore.Analyzers by Microsoft, 60. v3.1.3
CSharp Analyzers for Entity Framework Core.


EntityFramework by Microsoft, 85.7M downloads v6.4.0
Entity Framework 6 (EF6) is a tried and tested object-relational mapper for .NET with many years of feature development and stabilization.

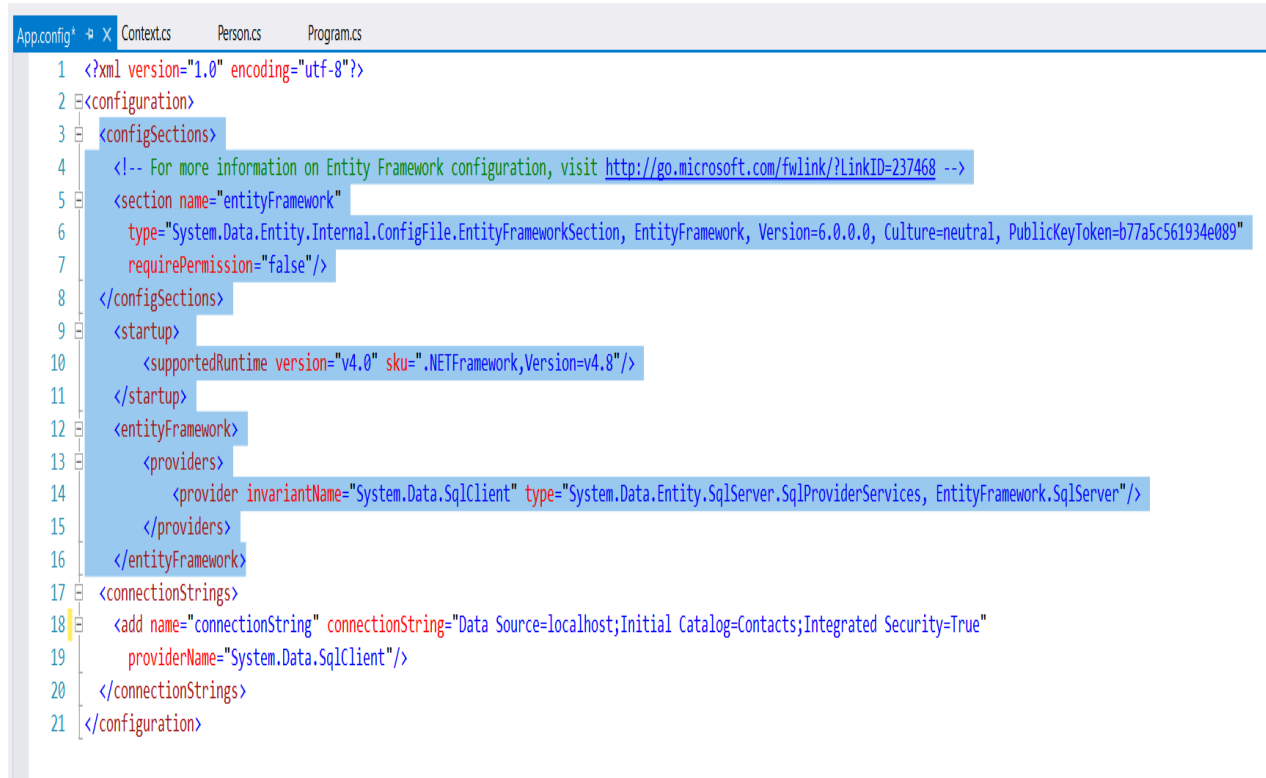
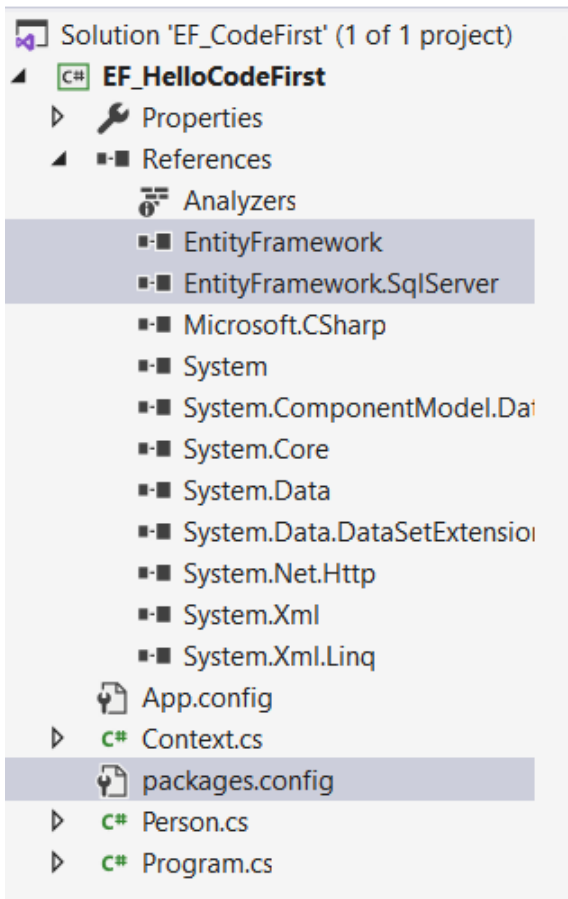
Version: 6.4.0 Install

Options

Description
Entity Framework 6 (EF6) is a tried and tested object-relational mapper for .NET with many years of feature development and stabilization.

Xây dựng EDM – Code-First

 Cài đặt thành công sẽ có thêm các thư viện của Entity Framework.



Xây dựng EDM – Code-First



Tạo CSDL từ class trong C#

- Tạo các class mô tả thông tin cần quản lý:
 - Mỗi class sẽ ánh xạ sang một bảng trong CSDL.
 - Class này sẽ có các tên gọi: model, domain, business hoặc entity class.
 - Class chỉ chứa dữ liệu trong các property
- Tạo một class là class con của lớp DbContext, ánh xạ chính của CSDL. Có tên gọi là context hoặc entity container.

Xây dựng EDM – Code-First



Tạo entity class ánh xạ CSDL:

- Sử dụng property để nhập xuất dữ liệu.
- Sử dụng các kiểu dữ liệu cơ sở của C#
- Là các class kiểu POCO – Plain Old Class Object.
- Mỗi entity class phải chứa một property thuộc kiểu int và có tên Id hoặc <tên_class>Id. EF sẽ tự động lấy property này làm khóa chính cho bảng dữ liệu.

```
class Person
{
    0 references
    public int PersonID { get; set; }
    0 references
    public string FullName { get; set; }
    0 references
    public string Address { get; set; }
}
```

Xây dựng EDM – Code-First



Tạo class Context là lớp con của DbContext:

- Được xây dựng để ánh xạ sang một CSDL
- Thông số để truy cập CSDL được lưu trong file App.Config.
- Lớp DbContext và DbSet<T> thuộc thư viện System.Data.Entity.

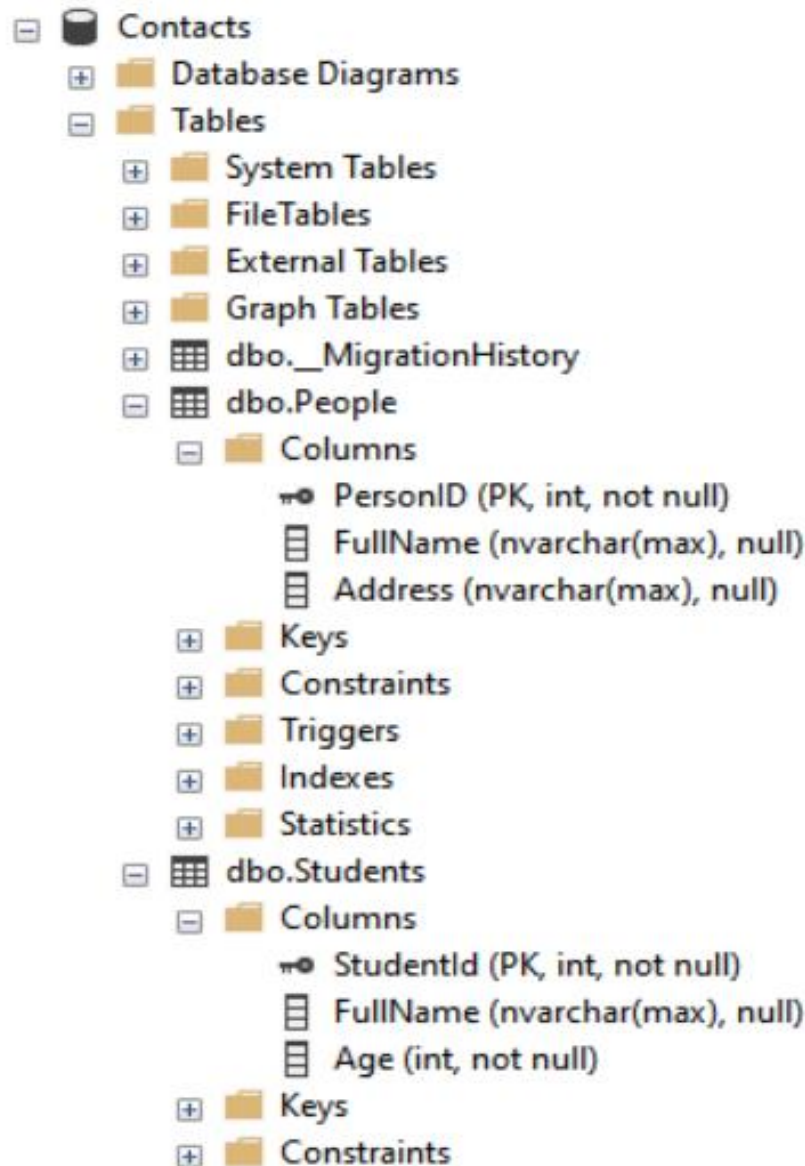
```
class Context: DbContext
{
    1 reference
    public Context() : base("name=connectionString"){
    }
    0 references
    public DbSet<Person> Person { get; set; }
    0 references
    public DbSet<Student> Student { get; set; }
}
```

Xây dựng EDM – Code-First

- 📖 Sử dụng code-first để tạo CSDL trong hệ quản trị từ các class đã xây dựng.
- 📖 Phương thức `CreateIfNotExists()` để kiểm tra CSDL được chỉ định trong Connection String đã có hay chưa.
- 📖 Nếu chưa sẽ tạo ra CSDL cùng tên đã chỉ định và các bảng tương ứng với các class.

```
static void Main(string[] args)
{
    using (var context = new Context())
    {
        context.Database.CreateIfNotExists();
    }
}
```

Xây dựng EDM – Code-First



Xây dựng EDM – Code-First



Tạo bản ghi mới và lưu xuống CSDL:

- Sử dụng lớp Context để thêm dữ liệu vào bảng CSDL.
- Tất cả bản ghi sẽ tương đương với một object của class mà nó tham chiếu.
- Class ánh xạ sẽ là thuộc tính của object lớp Context kiểu `DbSet<T>`.
- Object của lớp Context tương đương một CSDL.
- Sử dụng phương thức `Add(entity)` để thêm một bản ghi vào context
- Phương thức `SaveChanges()` để lưu xuống CSDL

Xây dựng EDM – Code-First

 Tạo giá trị để thêm vào CSDL:

```
using (var context = new Context())
{
    context.Database.CreateIfNotExists();

    var person = new Person
    {
        FullName = "ABC",
        Address = "US"
    };
    context.Person.Add(person);
    context.SaveChanges();
}
```

 Kết quả trong CSDL:

Results		Messages	
	PersonID	FullName	Address
1	1	ABC	US

Xây dựng EDM – Code-First



Truy vấn dữ liệu:

- Lấy dữ liệu theo kiểu `Select *` trong SQL bằng cách gọi thuộc tính của lớp ánh xạ trong object context
`var people = context.Person;`
- EF thực hiện cơ chế thực thi trễ (delayed execution) cho các truy vấn dữ liệu bằng LINQ.
- Dữ liệu lấy từ CSDL sẽ đưa vào biến thuộc kiểu danh sách `DbSet<T>`.
- Thao tác sửa: lấy đối tượng muốn sửa trong danh sách `DbSet<T>` trên ra, và chỉnh sửa theo yêu cầu.
- Thao tác xóa: cũng chọn đối tượng muốn xóa trong danh sách, và gọi phương thức `Remove(entity)`.
- Cuối cùng: gọi phương thức `SaveChanges()` để lưu thay đổi xuống CSDL

Xây dựng EDM – Code-First

Truy vấn dữ liệu Select:

```
var people = context.Person;
foreach (var person in people)
{
    Console.WriteLine($"ID: {person.PersonId}\tFullName: {person.FullName}\t" +
        $"Address: {person.Address}");
}
```

Cập nhật dữ liệu:

```
var people = context.Person;
foreach (var person in people)
{
    if (person.PersonId == 2)
    {
        person.Address = "AU";
        break;
    }
}
context.SaveChanges();
```

Xóa dữ liệu:

```
var person = context.Person.Find(1);
context.Person.Remove(person);
context.SaveChanges();
```

Xây dựng EDM – Code-First

- 📖 Thêm, xóa, sửa dữ liệu với code-first giống với các phương thức trong `List<T>` với LINQ.
- 📖 Chỉ yêu cầu sau khi chỉnh sửa trên danh sách cần gọi phương thức `SaveChanges()` để lưu xuống CSDL.
- 📖 EF theo dõi trạng thái của object và sinh ra câu truy vấn phù hợp để thực hiện các thay đổi (nếu có).
- 📖 Cơ chế theo dõi và transaction giúp EF hoạt động hiệu quả và đơn giản hóa việc xử lý dữ liệu

Truy vấn và xử lý dữ liệu



Các thao tác xử lý dữ liệu của EF đều dựa trên DbContext API bao gồm các nhóm class: **DbContext**, **DbSet**, **DbQuery**

- Lớp **DbContext**: là lớp cầu nối giữa lớp entity và CSDL, các thao tác truy vấn và lưu dữ liệu.
- Lớp **DbSet**: cung cấp các phép toán trên các kiểu entity (Add, Remove, Attach...); các khả năng truy vấn trên tập hợp.
- Lớp **DbQuery**: viết truy vấn trên tập hợp entity. Là lớp cha của **DbSet**.

Truy vấn và xử lý dữ liệu

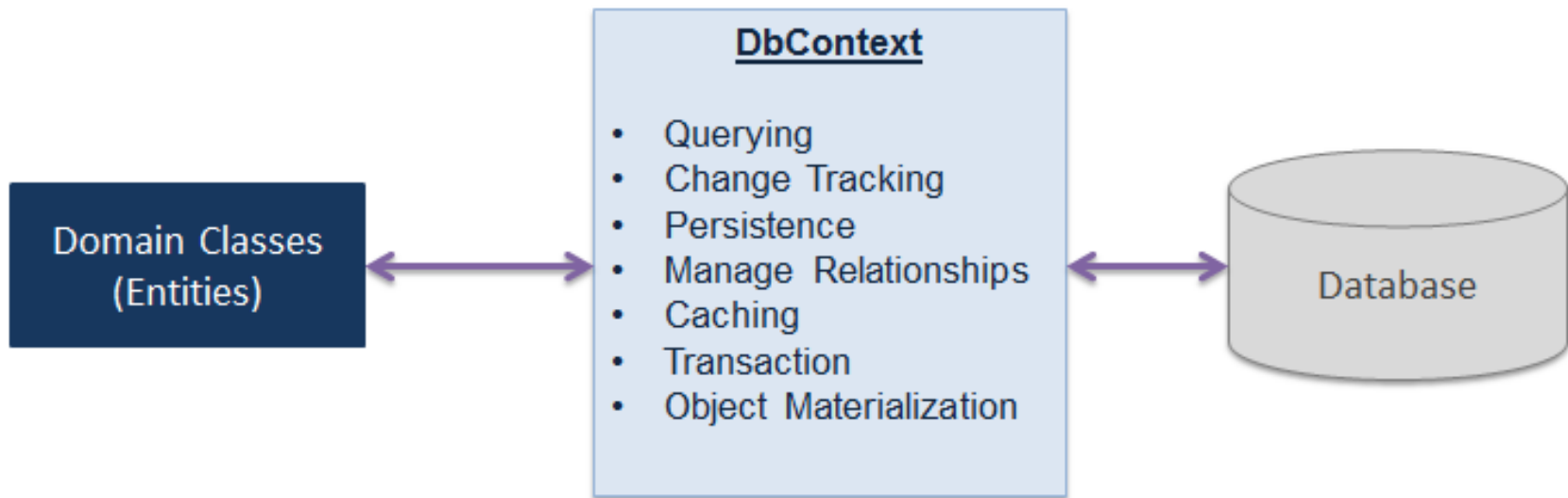


DbContext API còn có thêm một số tính năng quan trọng khác

- *Change Tracker API*: truy xuất tới các thông tin và phép toán của hệ thống theo dõi object của context. VD: khi object thay đổi giá trị, có thể xem giá trị gốc hoặc giá trị hiện tại của nó.
- *Validation API*: kiểm tra dữ liệu tự động.
- *Code First Model Building*: đọc các class và file cấu hình để tạo model class, metadata và CSDL.

Class DbContext

- 📖 Là lớp quan trọng trong EF.
- 📖 Là cầu nối giữa lớp entity và CSDL
- 📖 Phối hợp hoạt động của DbSet.
- 📖 DbContext chịu trách nhiệm cho các hoạt động sau:



Class DbContext

- *Query (truy vấn dữ liệu)*: chuyển đổi truy vấn LINQ to Entities thành truy vấn SQL và gửi tới CSDL
- *Change Tracking*: theo dõi những thay đổi trong object trong khi chương trình hoạt động (entity được tạo mới, chỉnh sửa, bị xóa) ; giúp cho DbContext chỉ sinh ra các truy vấn SQL cần thiết cho mỗi thay đổi.
- *Persisting Data*: thực hiện các truy vấn thêm, xóa và sửa tới CSDL dựa trên những thể hiện của thực thể.
- *Caching*: lưu tạm các entity đã được nhận từ CSDL, để hạn chế truy vấn và tăng hiệu suất.
- *Transaction*: Xử lý giao dịch

Class DbContext

- *Manage Relationship*: quản lý các quan hệ sử dụng CSDL; MSL và SSDL trong DB-First hoặc Model-First hoặc fluent API trong Code-First.
- *Object Materialization*: chuyên đổi bảng dữ liệu thô thành các object tương ứng của entity class.

Class DbContext



Một vài phương thức và thuộc tính của DbContext

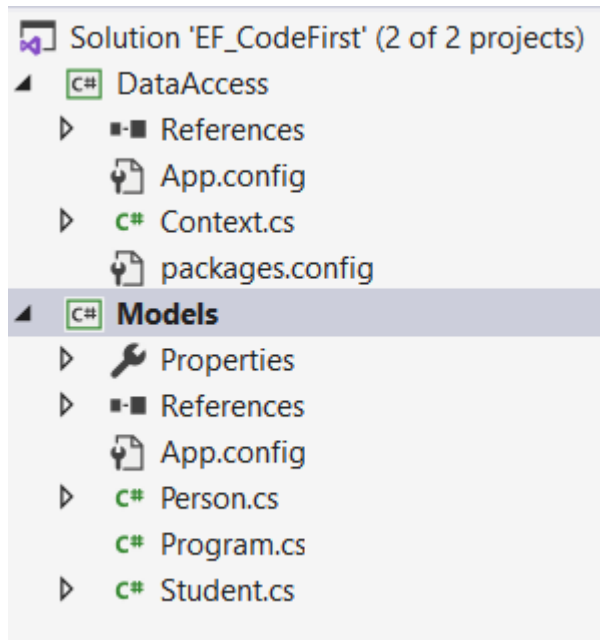
Thuộc tính/ Phương thức	Chức năng
<code>ChangeTracker</code>	Truy xuất thông tin và hoạt động của entity được context theo dõi
<code>Configuration</code>	Truy xuất cấu hình của context
<code>Database</code>	Truy xuất thông tin và hoạt động của CSDL
<code>Entry(Object)</code>	Lấy thông tin entity để truy xuất thông tin về sự thay đổi hoặc các thao tác thay đổi trên entity
<code>SaveChanges()</code>	Lưu tất cả các thay đổi trong context xuống CSDL
<code>SaveChangesAsync()</code>	Là dạng bất đồng bộ của <code>SaveChange()</code>
<code>Set<TEntity>()</code>	Tạo object của <code>DbSet<TEntity></code> để truy vấn và lưu.

Class DbContext



Một vài lưu ý khi dùng DbContext:

- Vì DbContext tạo sự kết nối đến CSDL nên có thể tách DbContext ra một project riêng, các entity class một project riêng.
- Trong project chứa context class tham chiếu đến các thư viện của EF.
- Sau khi không sử dụng context nữa nên giải phóng kết nối bằng phương thức `Dispose()` hoặc dùng cấu trúc `using`



Class DbSet

- 📖 Là tập hợp các entity object.
- 📖 Thực hiện các thao tác tạo mới, đọc, cập nhật và xóa bỏ các entity.
- 📖 Các phương thức của DbSet chỉ tác dụng với các entity nó đang chứa, không tác động ngay đến bảng dữ liệu.
- 📖 Mỗi đối tượng lưu trong DbSet sẽ được theo dõi đồng thời bởi DbContext.
- 📖 Mỗi đối tượng trong DbSet được gắn một trong các trạng thái: Unchanged (không thay đổi), Modified (cập nhật), Deleted (xóa) và Added (thêm).
- 📖 Khi thực hiện các phương thức Add, Remove hoặc thay đổi giá trị của đối tượng sẽ làm thay đổi trạng thái.
- 📖 Khi thực hiện SaveChange(), DbContext sẽ căn cứ vào trạng thái của đối tượng để tạo truy vấn phù hợp.

Class DbSet



Một vài phương thức và thuộc tính của DbSet

Thuộc tính/ Phương thức	Chức năng
Add	Thêm một entity vào danh sách
AsNoTracking()	Trả về danh sách entity nhưng chỉ đọc không được theo dõi bởi Context.
Create()	Tạo entity mới nhưng không thêm vào DbSet
Remove()	Đánh dấu là xóa một entity
Find(int)	Tìm entity theo khóa chính.
SqlQuery()	Thực thi truy vấn SQL
Include(String)	Chỉ định đối tượng có quan hệ (1-n, n-n, 1-1) vào câu truy vấn

Class DbSet



Một vài phương thức và thuộc tính của DbSet

Thuộc tính/ Phương thức	Chức năng
<code>Attach(Object)</code>	Thêm một entity đã có vào trong danh sách và báo cho EF biết đây là entity đã có, không cần phải tải lên từ CSDL