Cách tạo Table liên kết với nhau trong Model (Visual)

**// note : khóa ngoại hay Class nối sau kiểu dữ liệu thêm dấu chấm ?**

**// trong Model hai class join vs nhau thì Class này chứa Navigation của class kia và ngược lại để có thể join table theo hai chiều.**

**// Navigation property = 1 thuộc tính trong model C# để EF Core biết và để bạn đi từ Class này sang Class kia để join vs nhau thông qua khóa chính và khóa ngoại**

**//Tên khóa ngoại : bằng tên biến Navigation property + Id**

**// Navigation Property : cho biết khóa ngoại tham chiếu đến bảng nào**

**Item.cs (1-1 với SerialNumber) // quan hệ 1- 1 chuẩn**

public class Item

{

[Key]

public int ItemId { get; set; } // Khóa chính

public string Name { get; set; }

public decimal Price { get; set; }

// Navigation property: 1 Item chỉ có 1 SerialNumber

public SerialNumber ? SerialNumber { get; set; }

}

**SerialNumber.cs**

public class SerialNumber

{

[Key]

public int SerialNumberId { get; set; } // Khóa chính

public string Code { get; set; } // Serial code

// Khóa ngoại

public int ItemId { get; set; }

// Navigation property

[ForeignKey("ItemId")]

public Item Item { get; set; }

}

**🔹 Giải thích**

* **Item chỉ có một SerialNumber** → mối quan hệ **1-1**
* **SerialNumber vẫn có ItemId** làm khóa ngoại → liên kết với Item

SQL (1-1)

CREATE TABLE Item (

ItemId INT PRIMARY KEY,

Name NVARCHAR(100),

Price DECIMAL(18,2)

);

CREATE TABLE SerialNumber (

SerialNumberId INT PRIMARY KEY,

Code NVARCHAR(50),

ItemId INT UNIQUE, -- UNIQUE để đảm bảo 1-1

FOREIGN KEY (ItemId) REFERENCES Item(ItemId)

);

A screenshot of a computer screen

AI-generated content may be incorrect.

Result join

**Var items = \_context .Item .Include (s=> s.SerialNumber)**

A screenshot of a number

AI-generated content may be incorrect.

-----------------------------------------------------------------------------------------------------------------

------------------------------------------------------------------------------------------------------------------

**Item.cs (1 - nhiều )**

public class Item

{

[Key]

public int ItemId { get; set; } // Khóa chính

public string Name { get; set; }

public decimal Price { get; set; }

// Navigation property: 1 Item có nhiều SerialNumbers

public List<SerialNumber> ? SerialNumbers { get; set; } = new List<SerialNumber> () ;

}

public class SerialNumber

{

[Key]

public int SerialNumberId { get; set; } // Khóa chính

public string Code { get; set; } // Serial code

// Khóa ngoại

public int ItemId { get; set; }

// Navigation property

[ForeignKey("ItemId")]

public Item Item { get; set; }

}

SQL (1-nhiều )

CREATE TABLE Item (

ItemId INT PRIMARY KEY,

Name NVARCHAR(100),

Price DECIMAL(18,2)

);

CREATE TABLE SerialNumber (

SerialNumberId INT PRIMARY KEY,

Code NVARCHAR(50),

ItemId INT,

FOREIGN KEY (ItemId) REFERENCES Item(ItemId)

);

A screenshot of a computer

AI-generated content may be incorrect.

Các quy tắc đặt khóa chính khóa ngoại trong Model

public class Item {

public int ItemId { get; set; } // PK quy tắc ClassChinh + Id = khóa chính

public string Name { get; set; }

public int SerialNumberId { get; set; } // FK (cột trong DB) ClassNavigationProperty + Id = khóa ngoại // có thể thêm : [ForeignKey (" SerialNumberId")] để chỉ rõ khóa ngoại

// 👉 Đây là Navigation Property : Navigation Property nói cho biết là khóa ngoại tham chiếu từ bảng nào

public SerialNumber SerialNumber { get; set; }

}

public class SerialNumber

{

public int serialNumberId { get; set; } // EF sẽ nhận đây là PK

public string Name { get; set; }

public int ItemId { get; set; } // FK

[ForeignKey ("ItemId")]

public Item Item { get; set; }

}

CREATE TABLE SerialNumber (

SerialNumberId INT PRIMARY KEY,

Code NVARCHAR(50),

ItemId INT,

FOREIGN KEY (ItemId) REFERENCES Item(ItemId) // ItemId trong Table SerialNumber được tham chiếu từ ItemId trong Table Item

);

A screenshot of a computer

AI-generated content may be incorrect.

SerialNumber.serialNumberId [ khóa chính ] = Item.serialNumberId [khóa ngoại]

( SerialNumber-> Item ).

Item.ItemId [khóa chính] = SerialNumber.ItemId [khóa ngoại]

( Item-> SerialNumber).

**Cách dùng lệnh Join trong LINQ để Join nhiều bảng lại với nhau**

**Note : cùng bậc thì ghi cái nào trước cũng được còn khác thì thì đi tuần tự**

**🗂️ Bảng trong database**

**Item**

| **ItemId** | **Name** | **Price** | **SerialNumberId** |
| --- | --- | --- | --- |
| 1 | Bút | 5000 | 10 |
| 2 | Thước | 3000 | NULL |
| 3 | Vở | 7000 | 12 |

**Class Item**

**Public int ItemId // khóa chính**

**Public string Name**

**Public Double Price**

**Public int SerialNumberId // khóa ngoại**

**Public SerialNumber SerialNumber // Navigation Property**

**SerialNumber**

| **SerialNumberId** | **Name** | **CategoryId** |
| --- | --- | --- |
| 10 | SN-BUT-001 | 100 |
| 12 | SN-VO-777 | 100 |

**Class SerialNumber**

**Public int SerialNumberId // khóa chính**

**Public string Name**

**Public int CategoryId // khóa ngoại**

**Public Category Category // Navigation Property**

**Category**

| **CategoryId** | **Name** |
| --- | --- |
| 100 | Học tập |

**Class Category**

**Public int CategoryId // khóa chính**

**Public string Name**

A piece of paper with writing on it

AI-generated content may be incorrect.

// xuất phát từ Items

var items = await \_context.Items

.Include(i => i.SerialNumber) // Item → SerialNumber

.ThenInclude(sn => sn.Category) // SerialNumber → Category

.ToListAsync();

//xuất phát từ serialNumber

var serialNumber = await \_context.SerialNumbers

.Include(sn => sn.Category) // SerialNumber → Category

.Include(sn => sn.Item) // SerialNumber → Item

.ToListAsync();

//xuất phát từ category

var categories = await \_context.Categories

.Include(c => c.SerialNumbers) // Category → SerialNumbers

.ThenInclude(sn => sn.Item) // SerialNumber → Item

.ToListAsync();