# eager load:

- Eager loading cho phép bạn tải trước tất cả các dữ liệu liên quan trong một truy vấn duy nhất, tránh được vấn đề "N+1" query problem. Bằng cách sử dụng phương thức Include, có thể chỉ định các navigation properties mà bạn muốn tải cùng với thực thể chính.

- VD:

// Sử dụng eager loading để tải trước dữ liệu liên quan

var employeesDepartments = context.EmployeesDepartments

.Include(ed => ed.Emp)

.Include(ed => ed.Dept)

.ToList();

// Dữ liệu của Employees và Departments đã được tải cùng với EmployeesDepartments trong một truy vấn

- Sử dụng include để left join với bảng chứa khóa ngoại để lấy đầy đủ thông tin

(mặc định ban đầu không tải các thực thể liên quan)

- Chỉ dùng 1 câu lệnh sql (left join khi trong code sử dụng include)

# 2. lazy loading:

+ Sẽ tải các dữ liệu liên quan khi được gọi tới (qua backend or frontend)

VD: var emps = \_context.Employees.ToList();

var e = emps.First();

var emp\_depts = e.EmployeesDepartments.ToList();

or

<tbody>

@foreach (var employee in Model)

{

@if (employee.EmployeesDepartments != null)

{

@foreach (var ed in employee.EmployeesDepartments)

{

<tr>

<td>@ed.EmpId</td>

<td>@ed.DeptId</td> <!-- Assuming DepartmentName is a property of the Department class -->

</tr>

}

}

}

</tbody>

+ Tất cả thuộc tính liên quan phải đánh dấu virtual để EF Core có thể tạo các lớp proxy ghi đè các thuộc tính này và thực hiện hành vi lazy loading.

VD:

public virtual Employee? Emp { get; set; }

public virtual ICollection<EmployeesDepartment> EmployeesDepartments { get; set; }

+ Bắt buộc phải tải nuget proxies và config optionsBuilder.UseLazyLoadingProxies(); trong dbContext or program

+ Chiến lược thực thi sql:

**- Tải thực thể chính khi truy cập**

VD: SELECT [e].[Id], [e].[EmpId], [e].[DeptId] FROM [Employees\_Departments] AS [e]

**- Sẽ thực thi câu lệnh sql khi mà ta gọi tới (không join như eager mà sẽ where )**

VD: SELECT [e].[EmpId], [e].[Dob], [e].[Email], [e].[EmpName], [e].[Phone]

FROM [Employees] AS [e]

WHERE [e].[EmpId] = @\_\_p\_0

**- Nhược điểm của lazy loading:**

- Khi bạn truy cập vào một thuộc tính điều hướng (navigation property) của một thực thể,

EF Core sẽ tự động thực hiện một truy vấn mới đến cơ sở dữ liệu để tải dữ liệu liên quan.

🡪 Điều này dễ dẫn đến hiện tượng "N+1" query problem, nơi mà mỗi lần truy cập navigation property sẽ gây ra một truy vấn mới.

(Vấn đề sẽ xảy ra khi có nhiều employee và mỗi employee có nhiều thuộc tính điều hướng, Khi dùng lazy loading, nếu ta truy cập thuộc tính điều hướng đó qua backend or front end sẽ thực hiện nhiều câu truy vấn)

# 3. Explicit loading

- Là cơ chế tải dữ liệu trong EF gần giống lazy loading, khác ở chỗ ta sẽ chỉ định load những thuộc tính điều hướng nào

- Thay vì tự động tải như lazy (lazy thì chỉ cần gọi tới thuộc tính điều hướng thông qua backend or frontend là sẽ tự động load)

VD:

var emp = \_context.Employees.ToList().First();

var emps\_depts = \_context.EmployeesDepartments.toList().First();

\_context.Entry(emp).Collection("EmployeesDepartments").Load(); // tải dữ liệu điều hướng dạng danh sách cách object

\_context.Entry(emps\_depts).References("Employee").Load(); // tải dữ liệu điều hướng dạng 1 object