Cac thể loại LINQ

Query Syntax

Đây là cách viết câu lệnh LINQ giống với câu lệnh SQL truyền thống, sử dụng các từ khóa như
`from`, `join`, `where`, `select`, v.v.

```csharp
var result = from ms in context.MedicalServices
             join rms in context.RoomMedicalServices on ms.Id equals rms.MedicalServiceId
             join r in context.Rooms on rms.RoomId equals r.Id
             where ms.Id == "service-tim-mach--04"
             select r;
```

Query Syntax Method Chaining (Chuỗi phương thức cú pháp truy vấn).

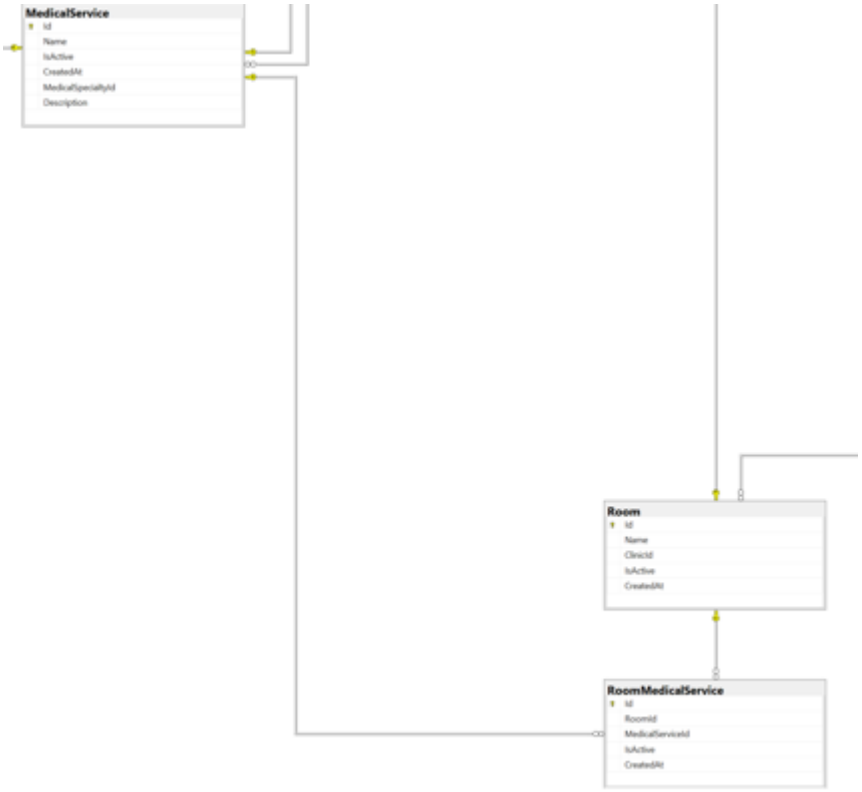
```
return await dbContext.MedicalServices
    .Where(ms => ms.MedicalServiceId.ToString() == medicalServiceId)
    .SelectMany(ms => ms.RoomMedicalServices)
    .Select(rms => new Room
    {
        Name = rms.Room.Name,
        IsActive = rms.Room.IsActive,
        CreatedAt = rms.Room.CreatedAt
    })
    .ToListAsync();
```

Method Syntax

phương thức LINQ một cách liên tiếp, sử dụng các phương thức như `Join`, `Where`, `Select`, v.v.

```
var result = context.MedicalServices
                    .Join(context.RoomMedicalServices,
                          ms => ms.Id,
                          rms => rms.MedicalServiceId,
                          (ms, rms) => new { ms, rms })
                    .Join(context.Rooms,
                          msrms => msrms.rms.RoomId,
                          r => r.Id,
                          (msrms, r) => new { msrms.ms, msrms.rms, r })
                    .Where(x => x.ms.Id == "service-tim-mach--04")
                    .Select(x => x.r);
```

Trường hợp khi join với các bảng Nhiều - nhiều



có 2 cách là

1. sẽ lấy luôn nếu có cột null

```
result = await dbContext.Room
                        .Include(x => x.Clinic)
                        .Include(x => x.RoomMedicalServices.Where(x => x.IsActive))
                            .ThenInclude(x => x.MedicalService)
                                .ThenInclude(x => x.MedicalSpecialty)
                                    .ThenInclude(x => x.Employees.Where(x => x.ClinicId == clinicId))
                        .Where(x => x.ClinicId == clinicId)
                        .Select(x => new RoomMedicalSpecialtyResponseModel
                        {
                            RoomId = x.Id,
                            RoomName = x.Name,
                            SpecialtyName = x.RoomMedicalServices.FirstOrDefault().MedicalService.MedicalSpecialty.Name,
                            Status = x.IsActive,
                            Doctor = x.RoomMedicalServices.FirstOrDefault().MedicalService.MedicalSpecialty.Employees
                .Where(x => x.ClinicId == clinicId
                       && x.RoleId == RoleStatic.Doctor
                       && x.IsActive)
                .Select(y => new DoctorBasicInfo
                {
                    Id = y.Id,
                    Name = y.Fullname,
                }).ToList()
                        })
                        .ToListAsync();
```

2 sẽ lấy hàng nếu ko có cột null

```
var â = await (from r in dbContext.Room
               join c in dbContext.Clinic on r.ClinicId equals c.Id
               join cm in dbContext.RoomMedicalService on r.Id equals cm.RoomId
               join meds in dbContext.MedicalService on cm.MedicalServiceId equals meds.Id
               join ms in dbContext.MedicalSpecialty on meds.MedicalSpecialtyId equals ms.Id
               join e in dbContext.Employee on ms.Id equals e.MedicalSpecialtyId
               where r.ClinicId == clinicId && e.ClinicId == clinicId
               group new { r, cm, meds, ms, e } by r.Id into grouped
              select new RoomMedicalSpecialtyResponseModel
               {
                   RoomId = grouped.Key,
                   RoomName = grouped.FirstOrDefault().r.Name,
                  SpecialtyName =  grouped.FirstOrDefault().ms.Name,
                  Status = grouped.First().e.IsActive,
                   Doctor = grouped.First().r.RoomMedicalServices.FirstOrDefault().MedicalService.MedicalSpecialty.Employees
                       .Where(x => x.ClinicId == clinicId
                                   && x.RoleId == RoleStatic.Doctor
                                   && x.IsActive)
                       .Select(y => new DoctorBasicInfo
                       {
                           Id = y.Id,
                           Name = y.Fullname,
                       }).ToList()
               }).ToListAsync();

return result;
```

Eager loading vs Explicit loading vs Lazy loading

- Lazy loading hoạt động bằng cách tạo ra các proxy class cho các entity của bạn. Khi bạn truy cập một thuộc tính navigation chưa được tải, proxy sẽ tự động kích hoạt một truy vấn đến cơ sở dữ liệu để tải dữ liệu cần thiết.

- Eager loading: Tải tất cả dữ liệu liên quan cùng một lúc bằng cách sử dụng Include(). Phù hợp khi bạn chắc chắn cần dữ liệu liên quan.

- Explicit loading: Tải dữ liệu liên quan một cách rõ ràng khi cần bằng cách sử dụng Load(). Cho phép kiểm soát chính xác khi nào dữ liệu được tải.

Cách 1: `foreach` Loop

Cách này sử dụng `foreach` loop để xử lý từng hóa đơn một cách tuần tự và truy vấn các mục dòng tương ứng cho từng hóa đơn trong mỗi lần lặp.

Cách 2: Batch Processing

Cách này sử dụng batch processing để truy vấn tất cả các mục dòng một lần và sau đó ánh xạ chúng trở lại từng hóa đơn bằng cách sử dụng từ điển.