**Include**

SELECT sorğusu yaradılan zaman Relational Data-ların da əldə olunması üçün Include method-dan istifadə edilir. Ehtiyac duyulan Navigation Property-lər nəzərə alınaraq tətbiq edilir beləliklə, sorğu nəticəsində Relation Data-lar da əldə edilir. Eager Loading-i həyata keçirmək üçün istifadə olunan bir method-dur. Include method nəticəsində SELECT sorğusuna JOIN əlavə edilir.

Aşağıda yerləşən query execute olunduqda, Navigation property null gəlmək əvəzinə, datalar yüklənmiş olaraq qayıdacaq.

List<TEntity> entityList = await \_context.Set<TEntity>().Include(e => e.NavigationProperty).ToListAsync(); //type safety version

//və ya

List<TEntity> entityList = await \_context.Set<TEntity>().Include(navigationPropertyName).ToListAsync();

Yuxarıda yerləşən LINQ Method Syntax ilə yaradılan sorğunun nəticəsi aşağıdaki kimi olacaq

SELECT \*

FROM Entities e

LEFT JOIN RelationTable rt ON e.RelationalTableId = rt.Id

**Filtered Include**

Biz Include method-u tətbiq edən zaman, gələn datalar üzərində filterasiya həyata keçirə bilərik. Filterasiya yalnız ICollection tipində olan Navigation property-lərə şamil edilir. Filterasiyaya, Where, OrderBy, OrderByDescending, ThenBy, ThenByDescending, Skip, Take method-lar aiddir.

Change Tracker-in aktiv olduğu situasiyalarda Filtered Include tətbiq edilən sorğu nəticəsində filterasiya olduğu halda gözlənilməyən data-lar gələ bilər. Bu daha əvvəl SELECT sorğusu nəticəsində gəlmiş və Change Tracker tərəfindən izlənilən və filterasiyadan keçməmiş datalara aiddir. Düzgün şəkildə Filtered Include prosesi üçün Change Tracker tərəfindən izlənməyən data-ların SELECT sorğusuna tətbiq edilməlidir.

**ThenInclude**

SELECT sorğusu yaradılan zaman Relational Data-ları əldə etmək üçün Include method-u istifadə etməli olduğumuzu bilirik. Include edilən table daxilində də Relational data-lar yer alarsa həmçinin onları əldə etməyə ehtiyac duyula bilər.

Biz Include vasitəsi ilə bu funskiyasını həyata keçirə bilərik, lakin sorğu zamanı Include edilən Navigation property daxilində yer alan Navigation property ICollection tipində olarsa, Include tətbiq edərək istədiyimiz nəticəni əldə edə bilməyəcəyik.

List<TEntity> entityList = await \_context.Set<TEntity>()

.Include(e => e.NavigationProperty)

.Include(e => e.NavigationProperty.InnerNavigationProperty)

.ToListAsync();

Bunun yerinə ThenInclude method istifadə edilir və biz Inlclude method tətbiq etdikdən sonra göndərdiyimiz NavigationProperty-ni ThenInclude ilə əldə edib SELECT sorğusuna onun Relational Data-larını əldə etmək üçün JOIN edə bilərik.

List<TEntity> entityList = await \_context.Set<TEntity>()

.Include(e => e.NavigationProperty)

.ThenInclude(rP => rP.InnerNavigationProperty)

.ToListAsync();

Yuxarıda yerləşən LINQ Method Syntax ilə yaradılan sorğunun nəticəsi aşağıdaki kimi olacaq

SELECT \*

FROM Entities e

LEFT JOIN RelationTable rt ON e.RelationalTableId = rt.Id

LEFT JOIN InnerRelationalTable irt ON rt.InnerRalationalTableId = irt.Id

**Auto Include**

Application daxilində bir Entity ilə bağlı yaranacaq bütün SELECT sorğuları zamanı mütləq bir şəkildə Include edilən Relational Table(lar) mövcuddursa bu zaman Include prosesini manual olaraq hər bir sorğu üçün tək-tək etmək əvəzinə, mərkəzləşdirilmiş bir şəkildə edərək təkrarın qarşısını almış olarıq və ən əsas bu prosesin idarəsini asanlaşdırmış olarıq.

Bunun üçün Auto Include adlanan anlayışdan istifadə edilir.

Context class daxilində override edilmiş OnModelCreating method daxilində aşağıdaki method-u yazaraq artıq Navigation property-ə aid olan Relational data-ların gətirilməsi üçün Include etməyə ehtiyac yoxdur. Özü avtomatik olaraq bunu edəcək.

modelBuilder.Entity<TEntity>().Navigation(e=>e.NavigationProperty).AutoInclude();

**Ignore Auto Includes**

Ehtiyac doğrultusunda bizim Auto Include ilə baş verən Include prosesini deaktiv etməyimiz lazım gələ bilər. Bunun üçün də SELECT sorğusu generate olan zaman IgnoreAutoIncludes method tətbiq edilir.

**Hierarchy Include**

Müvəqqəti olaraq gözləmədədir.