Gülbahar Elmas

1)

Bu kodun SQL karşılığıyla ilgili doğru ifade nedir?

```
{
        var result = context.Employees
          .GroupBy(e => e.Department)
          .Select(g => new
          {
            Department = g.Key,
            MaxSalary = g.Max(e => e.Salary),
            AvgSalary = g.Average(e => e.Salary),
            TotalSalary = g.Sum(e => e.Salary),
            Count = g.Count()
          })
          .ToList();
}
```

A) GroupBy işlemi SQL tarafında yapılır.
B) GroupBy bellekte yapılır, tüm veriler önce çekilir.
C) Average ve Sum C# içinde hesaplanır.
D) MaxSalary C# içinde hesaplanır.

Cevap: Doğru seçenek **A**'dır. Çünkü context.Employees bir IQueryable olduğundan GroupBy, Max, Average, Sum, Count gibi işlemler SQL'e çevrilip veritabanında yapılır, belleğe sadece sonuçlar gelir.

2)

Aşağıdaki kodun çıktısı nedir?

```
{
        var result = string.Join("-", Enumerable.Repeat("Hi", 3));
        Console.WriteLine(result);
}
```

A) HiHiHi
B) Hi-Hi-Hi
C) Hi Hi Hi
D) Hi,Hi,Hi

Cevap: Doğru seçenek **B'**dir. Çünkü "Hi" üç kez üretilir ve "-" ile birleştirilir.

3)

Bu kodda IsPrime metodu C# içinde yazılmış özel bir metot. Kodun çalışmasıyla ilgili doğru ifade nedir?

```
{
        var query = context.Orders
          .Where(o => o.TotalAmount > 1000)
          .AsEnumerable()
          .Where(o => IsPrime(o.Id))
          .ToList();
}
```

A) Tüm filtreler SQL tarafında çalışır, performans çok yüksektir.
B) İlk Where SQL'de, ikinci Where belleğe alındıktan sonra çalışır.
C) Tüm Where filtreleri bellekte çalışır.
D) AsEnumerable sorguyu hızlandırır, hepsi SQL tarafında çalış

Cevap: Doğru seçenek **B'**dir. İlk Where SQL tarafında çalışır, sonrasında AsEnumerable ile veri belleğe alınır, bu nedenle IsPrime filtresi C# tarafında uygulanır.

4)

Kod çalıştırıldığında hangi durum/sonuç gerçekleşir?

```
{
   using (var context = new AppDbContext())
        {
          var departments = context.Departments
            .Include(d => d.Employees)
            .AsSplitQuery()
            .AsNoTracking()
            .Where(d => d.Employees.Count > 5)
            .ToList();
        }
}
```

A) Tüm Department kayıtları tek bir SQL sorgusu ile, JOIN kullanılarak getirilir. EF Core değişiklik izleme yapar.
B) Department ve Employee verileri iki ayrı SQL sorgusu ile getirilir, EF Core değişiklik izleme yapmaz.
C) Department ve Employee verileri ayrı sorgularla getirilir, ancak EF Core değişiklik izleme yapar.
D) Tüm veriler tek sorguda getirilir ve değişiklik izleme yapılmaz.

Cevap: Doğru seçenek **B'**dir. Çünkü AsSplitQuery verileri iki ayrı SQL sorgusu ile getirir ve AsNoTracking nedeniyle EF Core değişiklik izleme yapmaz.

5)

Aşağıdaki kodun çıktısı nedir?

```
{
        var result = string.Format("{1} {0}", "Hello", "World");
        Console.WriteLine(result);
}
```

A) "{0} {1} "
B) "Hello World"
C) "World Hello"
D) "HelloWorld"

Cevap: Doğru seçenek **C**'dir. Çünkü string.Format içindeki {0} ve {1} sayıları parametrelerin sırasını gösterir. {1} ikinci parametreyi (World), {0} birinci parametreyi (Hello) alır. Sonuç: "World Hello".

6)

Aşağıdakilerden hangisi System.Linq.Enumerable ve System.Linq.Queryable arasındaki farktır?

A) Enumerable metodları yalnızca IQueryable üzerinde çalışır
B) Enumerable metodları IEnumerable üzerinde çalışır, Queryable metodları Expression Tree ile sorgu üretir
C) Enumerable metodları SQL veritabanına sorgu gönderir
D) Queryable metodları yalnızca string koleksiyonları üzerinde çalışır

Cevap: Doğru seçenek **B**'dir. Çünkü Enumerable metotları IEnumerable üzerinde bellekte çalışır, Queryable metotları ise IQueryable üzerinde expression tree kullanarak sorguları SQL'e çevirir ve veritabanında çalıştırır.

7)

Aşağıdaki kodun çıktısı nedir?

```
{
        var people = new List<Person>{
          new Person("Ali", 35),
          new Person("Ayşe", 25),
          new Person("Mehmet", 40)
        };
        var names = people.Where(p => p.Age > 30)
                .Select(p => p.Name)
                .OrderByDescending(n => n);

        Console.WriteLine(string.Join(",", names));
}
```

A) Ali,Mehmet
B) Mehmet,Ali
C) Ayşe,Ali,Mehmet
D) Ali

Cevap: Doğru seçenek **B**'dir. Çünkü yaşı 30'dan büyük olan kişiler Ali ve Mehmet'tir. OrderByDescending kullanıldığı için alfabetiğe göre ters sıralama olur ve sonuç "Mehmet,Ali" çıkar.

8)

Aşağıdaki kodun çıktısı nedir?

```
{
        var numbers = new List<int>{1,2,3,4,5,6};
        var sb = new StringBuilder();
        numbers.Where(n => n % 2 == 0)
                .Select(n => n * n)
                .ToList()
                .ForEach(n => sb.Append(n + "-"));

        Console.WriteLine(sb.ToString().TrimEnd('-'));
}
```

A) 4-16-36
B) 2-4-6
C) 1-4-9-16-25-36
D) 4-16-36-

Cevap: Doğru seçenek **A**'dır. Çünkü çift sayılar seçilip kareleri alınır ve "4-16-36-" elde edilir. TrimEnd sayesinde "4-16-36-" ifadesinin sonundaki "-" silinir ve sonuç "4-16-36" olur.

9)

System.Text.Json ve System.Collections.Generic kullanılarak bir listeyi JSON'a dönüştürmek ve ardından deserialize etmek için doğru işlem sırası nedir?

A) Listeyi serialize et → JSON string oluştur → Deserialize → liste
B) Listeyi deserialize et → JSON string oluştur → liste
C) JSON string oluştur → liste → serialize
D) JSON string parse → ToString()

Cevap: Doğru seçenek **A**'dır. Çünkü önce liste serialize edilerek JSON string oluşturulur. Sonra bu JSON string deserialize edilerek tekrar listeye dönüştürülür.

10)

Aşağıdaki kodda trackedEntitites değeri kaç olur?

```
{
        var products = context.Products
           .AsNoTracking()
           .Where(p => p.Price > 100)
           .Select(p => new { p.Id, p.Name, p.Price })
           .ToList();

        products[0].Name = "Updated Name";

        var trackedEntities = context.ChangeTracker.Entries().Count();
}
```

A) 0
B) 1
C) Ürün sayısı kadar
D) EF Core hata fırlatır

Cevap: Doğru seçenek **A**'dır. Çünkü kodda AsNoTracking() kullanıldığı için EF Core entity'leri ChangeTracker'a eklemez. Bu nedenle trackedEntities sayısı 0 olur. Ayrıca Select ifadesinde herhangi bir class ismi belirtilmeden doğrudan new { ... } kullanıldığı için sonuç anonymous type olur. Anonymous type bir entity olmadığından dolayı AsNoTracking kullanılmasaydı bile EF Core bu nesneleri takip edemezdi ve trackedEntities değeri yine 0 olurdu.

11)

Hangisi doğrudur?

```
{
        var departments = context.Departments
           .Include(d => d.Employees)
              .ThenInclude(e => e.Projects)
           .AsSplitQuery()
           .OrderBy(d => d.Name)
           .Skip(2)
           .Take(3)
           .ToList();
}
```

A) Her include ilişkisi ayrı sorgu olarak çalışır, Skip/Take her sorguya uygulanır.
B) Skip/Take sadece ana tabloya uygulanır, ilişkilerde tüm kayıtlar gelir.
C) Skip/Take hem ana tablo hem ilişkili tablolara uygulanır.
D) AsSplitQuery performansı düşürür, tek sorgu ile çalışır

Cevap: Doğru seçenek **B**'dir. Çünkü Skip/Take sadece ana tabloya uygulanır, ilişkilerdeki tüm kayıtlar eksiksiz getirilir.