Test Questions

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.

Aşağıdaki kodun çıktısı nedir?

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

B) Hi-Hi-Hi

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();
}
```

C) İlk Where SQL'de, ikinci Where belleğe alındıktan sonra çalışır.

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();
        }
}
```

B) Department ve Employee verileri iki ayrı SQL sorgusu ile getirilir, EF Core değişiklik izleme yapmaz.

Aşağıdaki kodun çıktısı nedir?

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

C) "World Hello"

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

B) Enumerable metodları IEnumerable üzerinde çalışır, Queryable metodları Expression Tree ile sorgu üretir.

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));
}
```

B) Mehmet,Ali

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

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

A) Listeyi serialize et → JSON string oluştur → Deserialize → liste

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

Hangisi doğrudur?

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

B) Skip/Take sadece ana tabloya uygulanır, ilişkilerde tüm kayıtlar gelir.

Bu kodun sonucu ile ilgili doğru ifade hangisidir?

```
{
        var query = context.Customers
          .GroupJoin(
            context.Orders,
            c => c.Id,
            o => o.CustomerId,
            (c, orders) => new { Customer = c, Orders = orders }
          )
          .SelectMany(co => co.Orders.DefaultIfEmpty(),
            (co, order) => new
            {
              CustomerName = co.Customer.Name,
              OrderId = order != null ? order.Id : (int?)null
            })
          .ToList();
}
```

B) Siparişi olmayan müşteriler de listelenir, OrderId null olur.

Bu kodun SQL karşılığı ile ilgili hangisi doğrudur?

```
{
        var names = context.Employees
            .Where(e => EF.Functions.Like(e.Name, "A%"))
            .Select(e => e.Name)
            .Distinct()
            .Count();

}
```

A) EF.Functions.Like SQL tarafında çalışır, Distinct ve Count SQL tarafında yapılır.

Hangisi doğrudur?

```
{
        var result = context.Orders
            .Include(o => o.Customer)
            .Select(o => new { o.Id, o.Customer.Name })
            .ToList();
}
```

A) Include bu senaryoda gereksizdir, EF Core sadece Select ile ilgili alanları çeker.

Hangisi doğrudur?

```
{
        var query = context.Employees
            .Join(context.Departments,
                e => e.DepartmentId,
                d => d.Id,
                (e, d) => new { e, d })
            .AsEnumerable()
            .Where(x => x.e.Name.Length > 5)
            .ToList();
}
```

B) Join SQL'de yapılır, Name.Length kontrolü belleğe alındıktan sonra yapılır.