1) Imperative Programming ve Declarative Programming kavramlarını kısaca açıklayıp farklarını belirtiniz.
   İmperative: Neyin nasıl yapılacağını akışı detaylı bir şekilde yazdığımız programlama stilidir. Örnek vermek gerekirse bir widget yaratan fonksiyonda imperative yükseklik genişlik arkaplan gibi özelliklerini belirtip daha sonra widgeti return ederiz.
   Declarative: Ne yapılacağını içeren, nasıl yapılacağının mantığını içermeyen bir programlama. Yine widget örneğinden gidecek olursak burada ise sadece return new widget diyerek widget yaratırız fakat nasıl yaratıldığıyla alakalı pek fikrimiz yoktur.

2) Veri tabanlarının sorgu optimizasyonlarında index oluşturmanın avantajı nedir? Sık index kullanmak bir probleme yol açar mı?
   Avantajlari: SQL querry'lerini optimize etmek amaci ile indexleme kullanabiliriz. Binlerce kayit arasindan arama yapmadan kayıta ulaşabilmemizi sağlıyor.
   Dezavantajları: index'ler ekstra yer kaplayan objelerdir. Bu sebeple filtreleme yapmayacağımız anahtar değer'ler için index oluşturmak, disk üzerinde gereksiz yer kaplayacaktır.

3) İlişkisel veritabanları için normalizasyon kavramı neyi ifade etmektedir? İlk 3 normal formu örnek üzerinden açıklayınız.
   Normalizasyon: Tasarım aşamasında veri tekrarını, veri kaybını veya veri yetersizliğini önlemek için gerçekleştirilen işlemlerdir.
   1NF:
   - Aynı tablo içinde tekrarlayan kolonlar bulunmaz
   - Her kolonda yalnızda bir değer bulunabilir.

   Örnek: 2. Ödevde geliştirdiğimiz Movie yapısında, Movie tablosu içinde Oyuncu1, Oyuncu2 gibi tekrarlayan kolon yazmamız gerekir veya Oyuncular diye kolon oluşturup hepsini bir değere yazmamalıyız.
   2NF:
   - 1NF'ye uygun olmalıdır.
   - Her satır eşsiz bir anahtarla tanınmalıdır.(Primary Key — Unique Key)
   - Herhangi bir veri alt kümesi birden çok satırda tekrarlanmamalıdır. Bu tür veri alt kümeleri için yeni tablolar oluşturulmalıdır.
   - Ana tablolar ile yeni tablolar arasında dış anahtarlar(foeign key) kullanılarak ilişkiler tanımlanmalıdır.

   Örnek: Her Movie için MovieId olmalıdır, Oyuncuları yazmak için birden fazla row insert etmemeliyiz oyuncuları farklı tablolarda tutmalıyız. Memberların MemberId si ile foreign key ilişkisi kurabiliriz.

   3NF:

   - Veri tabanı 2NF olmalıdır
   - Anahtar olmayan hiçbir kolon bir diğerine bağlı olmamalı ya da geçişken fonksiyonel bir bağımlılığı(trabsitional functional depency) olmamalıdır. Diğer bir deyişle her kolon anahtara tam bağlı olmak zorundadır.

   Örnek: MovieId veya MemberId dışında hiçbir column'un bağı olmaması gerekir. Örneğin oyuncular column'u izleme listesi tablosuyla bağlı olmamalıdır

4) ORM kütüphaneleri kullanmak her zaman avantajlı mıdır? ORM kütüphanelerinin ne gibi dezavantajları olabilir?
   Veri modellerimizi bir kere yazarız ve güncellemek,yeniden kullanmak daha kolaydır.
   Çoğu veritabanı işlemi arkada otomatik olarak yapılabilir.
   Karmaşık SQL yapılarını yazmak zorunda kalmayız.
   Kodumuzu temiz ve okunabilir hale getirir.
   Kullandığımız dil ile uyumlu çalışır.
   Dezavantajları için otomatik oluşturulan Querry belki en optimize olanı olmayabilir.

5) Domain Specific Language (DSL) kavramını açıklayınız.
   Domain Specific Language'ler, belirli bir domain(alan)'deki problemleri soyutlamak icin gelistirilen dillerdir. DSL diller, kendi domain'indeki konsept ve kurallari kullanir.

6) Long lived transaction kavramı hangi tip transactionları ifade etmektedir? Dezavantajları var mıdır? Varsa nelerdir ?
   Uzun süren transactionlar, databasede arama bulma gibi İşlemlerde kilitlenirse diğer operasyonlara devam edemeyeceği için Long lived transaction deriz. Lock, timeout gibi sorunlara yol açabilir.

7) Thread Pool nedir ? Nerelerde kullanılır ?
   Thread Pool: Threadlerin manage edilmesi için kullanılan bir yöntemdir. Threadleri işi bitince sonlandırmak yerine gelecekteki işler için bekletmektedir. Thread pool yaratıldığında belirli sayıda Thread yaratılır ve yapılacak işler gelene kadar bu Threadleri bekletir. Yapılacak işler sıraya girdikçe boşta bulunan Threadler uyanır işleri yapar ve tekrar bekleme durumuna geçer. Springde de Thread Pool yapısı bulunuyor. Yapılacak çok fazla task varsa bunları threadlere bölüp hem zaman hemde performans kazanabiliriz.

8) Scalability nedir ? Horizontal ve Vertical Scalability kavramlarını açıklayınız.
   Scalability: Bir sistemin, ağın veya sürecin artan iş yükünü yönetebilmek ve büyümeyi karşılamak için kaynakların esnetilerek/arttırılarak kullanılması.
   Horizotal Scalability: Uygulamamızın çalıştığı sanal veya fiziksel makine sayısını arttırırız buna yatay ölçekleme denir
   Vertical Scalability: Uygulamamızın çalıştığı sanal veya fiziksel makineye daha fazla kaynak eklememiz(ram,disk,cpu vs) dikey ölçeklemedir.

9) Data replication ve sharding nedir ? Aralarında nasıl bir fark bulunmaktadır ?
   Sharding: Database'de tutulan verinin daha küçük parçalara ayrılarak query performanslarını arttırmaya yönelik yapılan bir işlemdir.
   Data replication: Database'de tutulan tüm verinin başka bir Database'ye aktarılması işlemidir. Yedekleme veya performans amacıyla yapılabilir.
   Aralarındaki fark replication veriyi bütün olarak kopyalarken sharding veriyi hepsi birbirinden farklı parçalara ayırıyor.