1 - What is JPA ?

Java Persistence API, ilişkisel verilerin Java sınıfları ile tutulmasına ve uygulamanın sonlanmasından sonra verinin kalıcı olmasını sağlayan teknolojidir.

JPA ile geliştiriciler SQL komutları yerine direk olarak nesneler üzerinde çalışabilir. SQL sorgularını JPA kendi içerisinde barındırır. Veritabanında verileri saklayabilir, güncelleyebilir veya veritabanından verileri çekebilir veya map edebilirsiniz. JPA, verilerin veritabanı üzerinde nasıl eşleşeceği, verinin güvenliği ve performansı ile ilgilenir.

JPA'da önemli kavramlardan birisi de Entity dir. Entity veritabanımız ile yazılımımız arasında ilişki kurmasını sağlayan nesnelerdir. Bu nesneler kalıcı olmak zorundadır.
Java Persistence API'nin çalıştırdığı nesne Entity'dir. Tüm işleri yapan bu nesnemizdir. Arama, silme işlemlerini de yöneterek yapabilmekteyiz.

JPA implementasyonu yapan başlıca araçlar olarak Hibernate, TopLink, EclipseLink ve OpenJPA sxayılabilir.

2 - What is the naming convention for finder methods in the Spring data repository interface ?

Bunu örnek vererek açıklamak gerekirse, repositoryde Id,Name,Status,Price gibi değişkenlerimiz olsun. Bunlar için find methodları yazmak istediğimizde findBy ön eki ile birlikte oluşturduğumuz method'a verilecek parametre ile değişkeniyle isimlendirmeye devam ederiz. Yani kısaca, id ile arama yapacaksak "findById" , name ile arama yapacaksak findByName şeklinde kullanırız.

3 - What is PagingAndSortingRepository ?

PagingAndSortingRepository, sayfalandırma ve sıralama soyutlamayı kullanarak varlıkları almak için ek yöntemler sağlamak için CrudRepository'nin bir uzantısıdır.

Page findAll(Pageable pageable)Page – nesnede sağlanan disk belleği kısıtlamasını karşılayan varlıkların a'sını döndürür Pageable.

Iterable findAll(Sort sort) – verilen seçeneklere göre sıralanmış tüm varlıkları döndürür. Burada sayfalama uygulanmaz.

4 - Differentiate between findById() and getOne() ?

getOne() metodu verilen id'ye ait objenin referansını döner. Bu metod içeride EntityManager.getReference() metodunu çağırır. Bu metod her zaman database'e gitmeden (lazy fetch) bir proxy döner. İstenen entity'nin database'de bulunmaması durumunda EntityNotFoundException fırlatır.

findById() metodu ise her çağırıldığında gerçekten database'e gider ve objeyi oradan getirir. Bu EAGER yüklenen işlemdir bu sebeple eğer getirmeye çalıştığımız obje DB'de yoksa null dönecektir.

Bir tablo ile kıyaslamak gerekirse

getOne()	findById()
Lazy Load ile hedefin referansını getirir	Verilen id'yi gerçekten getirir (referans değil)
Objenin propert'ilerine erişmek gerekmediğinde oldukça kullanışlı	Obje tamamen getirildiği için tüm özelliklerine erişilebilir
Objeye erişmek gerektiğinde obje DB'de yoksa EntityNotFoundException hatası fırlatır	Eğer obje DB'de kayıtlı değilse null döner
Daha iyi performans	İlave bir DB sorgusu gereklidir

5 - What is @Query used for ?

Query : Veritabanı tablolarıyla eşleştirilen ve Entity adı verilen sınıflardan yaratılan objeler üzerinden sorgu yaratmak için kullanılır. Entity üzerinden yaratılan objeler üzerinde @Query anatasyonu ile SQL sorgusu yazabiliriz. Örnek kullanım şekli olarak aşağıdaki gibi kullanabiliriz.

```
@Query("SELECT b FROM Book b WHERE b.title = ?2 and b.author = ?1")
Book findBookByTitleAndAuthorIndexJpql( String authorName, String title);
```

6 - What is lazy loading in hibernate ?

Çocuk entitymizin oldugunu düşünelim. Çocuk entitysi ile ilişkili oyuncak entitysi olsun ve aralarındaki ilişki OneToMany olsun. Yani bir çocugun birden fazla oyuncagı olabilsin. Veritabanından çocugu çagırdıgımızda sahip oldugu oyuncakları getirmesin sadece kendisi gelsin istiyorsak burada lazy loading yapmalıyız. Burada sadece kendisi gelsin ben gerektiginde oyuncaklarını da çagırırım demiş oluyoruz. Bu lazy loading için güzel bir örnek olur

7 – What is SQL injection attack ? Is Hibernate open to SQL injection attack ?

SQL enjeksiyonu, bir saldırganın bir uygulamanın veritabanına yaptığı sorgulara müdahale etmesine olanak tanıyan bir web güvenlik açığıdır. Genellikle bir saldırganın normalde alamadıkları verileri görüntülemesine izin verir. Bu, diğer kullanıcılara ait verileri veya uygulamanın kendisinin erişebildiği diğer verileri içerebilir. Çoğu durumda, bir saldırgan bu verileri değiştirebilir veya silebilir ve uygulamanın içeriğinde veya davranışında kalıcı değişikliklere neden olabilir.

Bazı durumlarda, bir saldırgan, temel alınan sunucunun veya diğer arka uç altyapısının güvenliğini aşmak veya bir hizmet reddi saldırısı gerçekleştirmek için bir SQL enjeksiyon saldırısını yükseltebilir.

Örnek vermek gerekirse, Bir kullanıcı giriş formunda kullanıcı ve şifrenin doğruluğunu şu şekilde kontrol ederiz:

select * from users where uname='"uname"' and pass='"pass"'

Biz eğer sql injeciton kontrolü yapmadıysak username ve pass alanına ' OR '1'='1 sql sorgusunu şu hale getirilerek yani %100 doğru olarak kabul ettirerek kolayca sistemdeki kullanıcı adı ve şifre bilgileri çalınabilir

Yaptığım araştırmlar sonucunda da, Hibernate'de güvenlik önlemleri almasına karşın büyük bir ek güvenlik katmanı sağlamıyor.

8 - What is criteria API in hibernate ?

Criteria Queries adından da anlaşıldığı gibi sorgularımızı kriterler ekleyerek bir sorgulama yöntemidir. Criteria API'si ile SQL'de kullandığımız =, <, >, <=, >= ifadeleri, bunun yanında or, and, like bağlama işlemleri gerçekleştiriliyor. VEYA ile şart bağlamanın bir diğer yolu da Disjunction'dır. Restrictions.or() methodu iki farklı şartı bağlamamızı sağlar. Disjunction ile ise ikiden fazla şartı bağlayabilirsiniz.

Ham SQL yapmadan sorgu yazmamızı sağlar ve ayrıca Hibernate'in ana özelliklerinden biri olan sorgular üzerinde nesne yönelimli kontrol sağlar. Criteria API, farklı türde filtreleme kuralları ve mantıksal koşullar uygulayabileceğimiz, programlı olarak bir ölçüt sorgu nesnesi oluşturmamıza olanak tanır.

Hibernate 5.2'den bu yana, Hibenate Criteria API'si kullanımdan kaldırılmıştır ve yeni geliştirmeler JPA Criteria API'sine odaklanmıştır.

9 - What Is Erlang? Why Is It Required For Rabbitmq ?

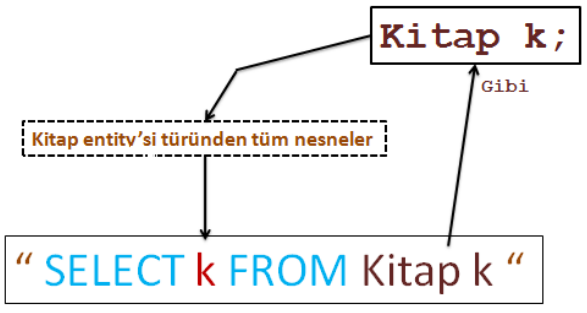
Erlang, büyük endüstriyel ve gerçek zamanlı sistemlerin programlanması için tasarlanmıs eszamanli(concurent) ve fonksiyonel bir programlama dilidir.

Erlang dinamik olarak yazilmistir ve bir desen esleme (pattern matching) sözdizimine sahiptir.Fonksiyonlar bu dilde ozyinelemeli olarak kullanilir ve tanimlanirlar. Erlang acik es zamanlilik saglar, ve yan etkilerden(side effect) aritilmis bir dildir.

RabbitMQ, Unix üzerine Erlang ile yazılmış mesaj kuyruğu sistemidir. Bu yüzden sistemimizde Erlang kurulmadığı taktirde RabbitMQ çalışmayacaktır.

10 – What is the JPQL ?

JPQL (Java Persistence Query Language) , JPA standardının Entity nesnelerini sorgulamak üzerine tanımladığı bir dil. JPQL, HQL 'e fazlasıyla benzeşir. Bu diller SQL diline hemen hemen benzemelerine karşın, kullandığı argümanlar veritabanı tabloları yerine Entity nesneleridir.



Bu sorgu, FROM tümcesinde belirtilen kaynağı ("k" karakteriyle simgelenmiş Kitap türündeki tüm entity nesneleri) tanımlamakta ve SELECT tümcesinden sonraki "k" karakteriyle tüm entity nesnelerini veritabanından çekmeyi sağlamaktadır.

11 – What are the steps to persist an entity object ?

- Entity Manager Factory nesnesi oluşturulur. Java.persistence paketinde bulunan EntityManagerFactory interface'i, bir entity yöneticisi sağlamak için kullanılır.

- Factoryden bir entity nesnesi alınır

- Bu entity manageri başlatılır.

- Bir data ilişkisel veri tabanında kalıcı hale getirilir.

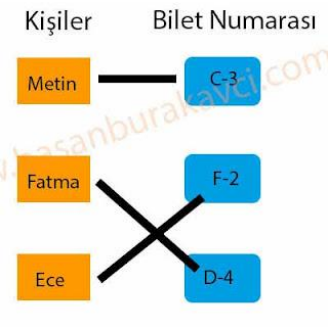
- Transaction kapatılır.

- Factory kaynakları kapatılır ve serbest bırakılır.

12 – What are the different types of entity mapping ?

@OneToOne

Bire bir ilişki için kullanılır. Örnek üzerinden işleyelim.



Bir tiyatro salonunda kişilerin sadece bir bilet alma hakkı vardır. Çünkü bir kişi bir koltuğa oturabilir. Birden fazla koltuğa oturması mümkün değildir. Aynı zamanda bir koltuğa sadece bir kişi oturur. Yani bir kişi bir koltuğa sahip olur. Bir koltuk bir kişi tarafından kullanılır.

@OneToMany

Bire-çok ilişkisi veritabanımızdaki bir tablodaki bir kaydın diğer tablomuzda bulunan kayıtlardan birden fazlasıyla ilişki kurmasına denir. Örnek verecek olursak bir personelimiz olsun. Bu personelimizde ait birden fazla adres olabilir

@ManyToOne

Bir Kullanıcının ve bir Mesajın olduğunu, bir kullanıcının binlerce mesaja sahip olabileceğini varsayalım, bunu sadece Mesajdan Kullanıcıya bir ManyToOne olarak modellemek mantıklı olabilir, çünkü zaten bir kullanıcının tüm mesajlarını nadiren isteyeceksiniz.

@ManyToMany

Bunu da örnek vererek açıklamak gerekirse, kitaplarımız ve yazarlarımız olsun. Kitapların da bir çok yazarı olabileceği gibi yazarlarında birden çok kitabı olabilir. Bu gibi ilişkilerde ManyToMany anotasyonunu kullanırız.

13 - What are the properties of an entity ?

Persistable(Kalıcılık): Kaydedilebilir ve ulaşılabilir olmalı. Public no-arg constructor, getter/setter metodları kullanılarak yapılabilir.

Idetity(Kimlik) : Her Entity'nin bir unique(eşsiz) kimlik bilgisi olmalı. Primary Key olarak ifade edilebilir

Transactionality(İşlem) : Veritabanına ekleme / silme / güncelleme bir transaction'dır. İşlem tam anlamıyla tamamlanmazsa rollback yapılmalı yani en başa dönülmelidir.

14 - Difference between CrudRepository and JpaRepository in Spring Data JPA?

CrudRepository base interface'dir ve Repository Interface'i bundan extends edilirken JPARepository,CrudRepository'den extend edilmiş olan PagingAndSortingRepository'sinden extend edilir.

CrudRepository, save(), saveAll,findByID,findAll gibi CRUD işlemleri için metotlar içerirken JPARepository hem Crud hem de PagingAndSorting Repository içerisindeki metotları içerir.

CrudRepository sayfalandırma ve sıralama işlemleri için metotlar içermezken JPARepository bunları içerisinde barındırır.