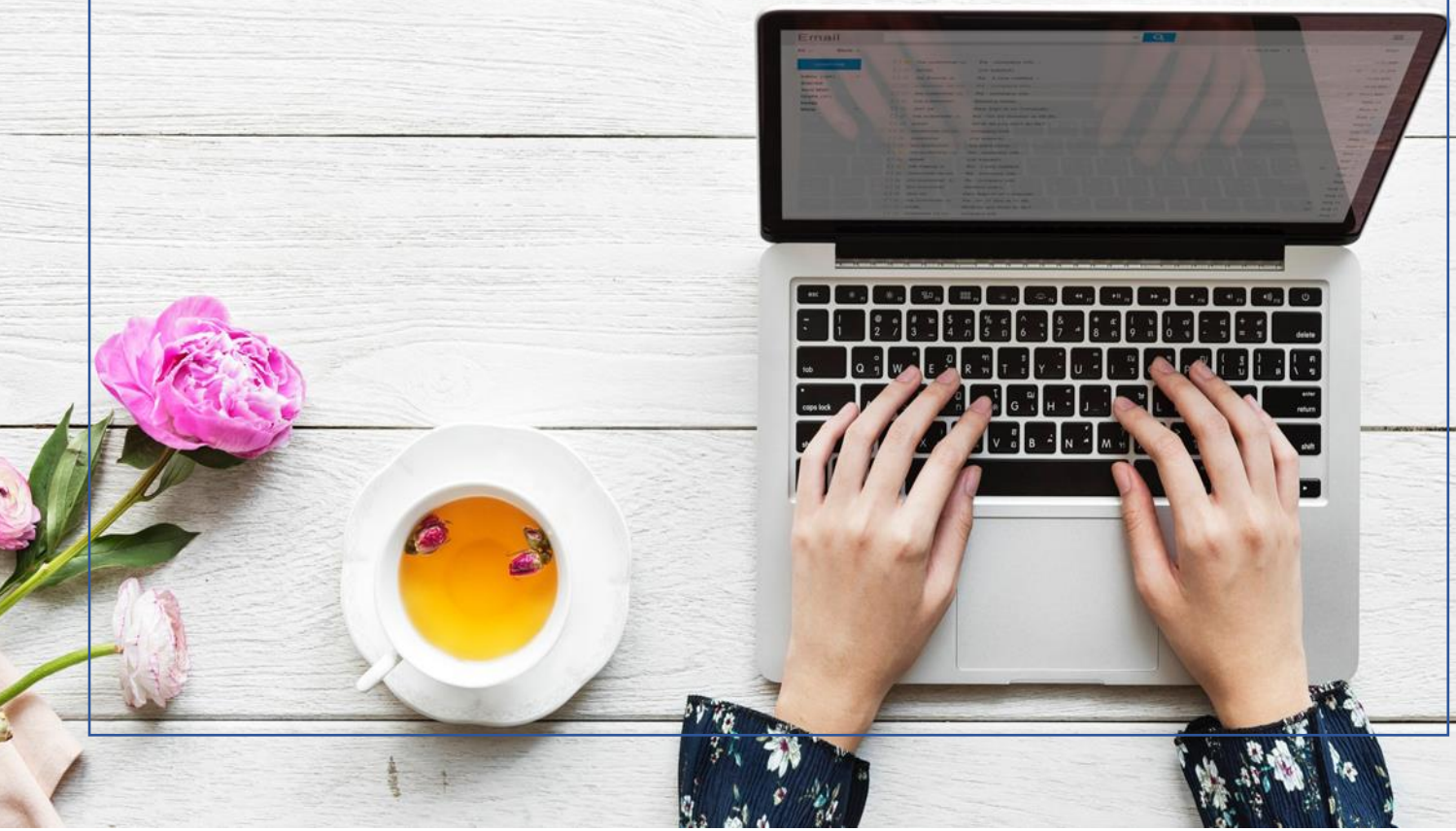
TEORİK ÖDEV SORULARI(HAFTA-3,4)

Rana Yılmaz



CEVAPLAR

1.Resful vs. Soap?

APIs yani Application Programming Interfaces yani Uygulama Programlama Arayüzleri yani farklı yazılımların birbiriyle iletişim kurması yoludur. Restful ve Soap bu iletişimin kurulmasıyla ilgili iki farklı yaklaşımdır.

Rest (Representational state transfer) network üzerinden durum paylaşımıdır,bir mimaridir. HTTP protokolü üzerinden çalışır. Bir istek(request) alındığında işlem Rest için tasarlanmış API'ler yani Restful API'lerin çeşitli biçimlerine döndürülür. Bunlar HTML, XML, text, JSON olabilir. Daha okunaklı olması nedeniyle genellikle JSON formatı tercih edilir.

Bir API'nin Restful olması için;

1. İstemci(client)-sunucu(server) mimarisi.
2. Stateless client-server iletişimi yani sunucuda hiçbir istemci içeriği depolamaz. Oturum durumu bilgilerini client tutar.
3. Client-server etkileşimlerine olan ihtiyacı kaldırmak için önbelleğe alınabilir veriler.
4. Standart bilgi aktarımı için tek tip arayüz.
5. Katmanlı sistem kısıtlaması.
6. Talep üzerine kodlama. Bununla server executable code aktararak client işlevselliğini artarır.

Soap(Simple Object Access Protocol) bir protokoldür.Transaction yani veri transferi işlemi için ACID(Atomicity:bütün olarak işlemlerin hepsinin başarılı olması, consistency:tutarlılık, isolaiton: transaction işlemlerinin tek tek yapılması, durability: hata üzerine sistemin önceki geçerli duruma dönebilmesi.) kurallarına olanak sağlar. WS-security, WS-ReliableMessaging, WS-adressing, WSDL gibi gereksinimler içerir. HTTP, SMTP, TCP gibi protokoller aracılığıyla işlenen veriyi sadece XML formatında çalıştırır. SOAP API'sine yapılan tamamlanmış bir request tarayıcı tarafından önbelleğe alınamadığı için isteğin API'ye tekrar gönderilmeden erişilmesi mümkün değildir.

Rest esnek ve hızlı olması, format kısıtı olmamasıyla web tabanlı uygulamalarda SOAP'ın yerini almıştır.

2.Acceptance test ve functional test arasındaki farklar?

Functional test, ürünün doğru çalışıp çalışmadığını kontrol eden bir doğrulama testidir.

Acceptance test, çalışır ürünün müşteri ihityacına uygun senaryolarda çalışıp çalışmadığını kontrol eder. Reliability(güvenilirlik), availability(uygunluk), sclability(ölçeklenebilirlik), usability(kullanılabilirlik), security(güvenlik), maintainability(sürdürülebilirlik), configurability(yapılandırılabilirlik) faktörleri kontrol edilir.

3.Mocking nedir?

Mocking, yazılan kodu test ederken bağımlı objelerin sahtelerinin oluşturulmasıdır. Sebebi ise tek bir senaryonun test edilmesi gerektiği ve bu senaryonun bağımlılıklarının elimine edilmesi gerektiğidir. Mock objeleri bu bağımlılıkların yerini alarak işlemleri daha kolay ve hızlı halletmemizi sağlar.

4.Unit test için uygun makul kod yüzde kaçtır ve neden?

Kod kapsamı, kaynağınızın ne kadarının test edildiğini anlamanıza yardımcı olabilecek bir ölçümdür. Test paketinizin kalitesini değerlendirmenize yardımcı olabilecek çok kullanışlı bir ölçümdür ve burada projelerinize nasıl başlayabileceğinizi göreceğiz.

5.HTTP/POST vs. HTTP/PUT?

POST: HTTP tarafından desteklenen ve bir web sunucusunun, istenen mesajın gövdesinde yer alan verileri kabul ettiğini gösteren bir yöntemdir. Bir dosya yüklediğinizde veya kullanıcı verilerini göndermek için kullanılır. Syntax'ı POST/articles.

PUT: Sunucudaki kaynağı güncellemek için kullanılır. Hedef URL'de var olanı başka bir şeyle değiştirir. Entity'nin URI(Uniform Resource Identifier) altında saklanmasını ister. Syntax'ı PUT/article/{article-id}

6.HTTP için güvenli ve güvenli olmayan metotlar nedir?

Güvenli metotlar herhangi bir kaynağı değiştirmez. GET, HEAD, TRACE, OPTIONS.

Güvenli olmayan metotlar sunucudaki bazı kaynakları değiştirebilir. PUT, DELETE, POST, PATCH.

7.HTTP'nin basic authentication'ı nasıl çalışır?

Authentication yani kimlik doğrulaması/yetkilendirme, HTTP için bir request(istek)te bulunurken kullanıcı adı ve parola sağlama yöntemidir. Kullanıcı adı/şifre Basic-Auth'da request body yerine request headerlarında iletilir.

1. HTTP client, web server'a request atar.
2. Web server, yetkilendirmeye gereksinim duyarsa 401 döner.
3. Client kullanıcı adı- şifre girmek için dialog box görüntüler.
4. Bilgiler girildikten sonra, client bunu auth header kullanarak gönderir. Bilgiler doğruysa 200; yanlışsa 401 döner.

8.RestTemplate'i Spring'te tanımla.

RestTemplate, Spring uygulaması içinde REST servislerini çağırmak için tasarlanmış bir sınıftır. Bu sınıfı kullanarak verileri tüketmek ve daha fazla işlemek için API'yi ondan çağıran yöntemler yazabiliriz, ayrıca herhangi bir bağımlılığımız varsa üçüncü taraf veya başka bir uygulamadan veri alabiliriz.

Web servislerini tüketmek(consume) için exchange() metodu kullanılır. spring-boot-starter-web dependency içinde bulunur.

getForObject(url, classType)

getForEntity(url, responseType)

exchange(url, httpMethod, requestEntity, responseType)

execute(url, httpMethod, requestCallback, responseExtractor)

postForObject(url, request, classType)

postForEntity(url, request, responseType)

postForLocation(url, request, responseType)

exchange(url, requestEntity, responseType)

execute(url, httpMethod, requestCallback, responseExtractor)

put(url, request)

delete(url)

headForHeaders()

optionsForAllow()

9. @Controller ve @RestController arasındaki fark?

@Controller annotation, @Component annotation'dan özelleşmeştir. Bir classın controller yani web request işleiycisi olduğunu belirtir. @RequestMapping ile birlikte kullanılır.

@Restcontroller annotation, Restful web servisleri oluşturmak için kullanılır. Bir classın, client tarafından yapılan requestleri işlemesini sağlar. GET, POST, DELETE, PUT gibi istekleri tüm Rest API'lerin işlemesine izin verir.

10.DNS Spoofing nedir? Nasıl engellenir?

DNS(Domain Name Service) Spoofing, hedeflenen bir kullanıcıyı saldırgan kontrolü altında kötü niyetli bir web sitesine yönlendirmek için bir DNS sunucusundaki girişleri zehirleme işlemidir. İnternete genel Wi-Fi'den erişen herhangi bir kullanıcı, DNS Spoofing'e karşı savunmasızdır. DNS Spoofing'den korunmak için internet sağlayıcıları DNSSEC (DNS güvenliği) kullanabilir. DNSSEC, DNS aramalarını gerçek kabul etmeden önce şifreleme yapar.

11.Content negotation nedir?

REST kaynaklarnda farklı temsilleri bekleyen farklı clientlar olabilir. Bunu düzenlemek için kullanılan content negotation, client ve server arasında yapılan bir içerik anlaşmasıdır. Amacı, aynı URI ile farklı döküman türlerinde içerik sunabilmektir. Yani daha genel bir ifadeyle kaynak gösterim şeklinin kullanıcılar tarafından belirlenmesi diyebiliriz.

12.RestFul içindeki statelessness(uygunsuzluk) nedir?

RESTful Web Servislerinde, istemciden sunucuya yapılan her istek, isteği anlamak için gerekli tüm bilgileri içermelidir. Sunucu, sunucuda saklanan herhangi bir bağlamdan yararlanamaz. Bu nedenle uygulamanın oturum durumu tamamen istemcide tutulur. İstemci, oturumla ilgili bilgileri kendi tarafında depolamaktan ve yönetmekten ve ayrıca ihtiyaç duyulduğunda sunucuya herhangi bir durum bilgisini göndermekten sorumludur. İstemci ve sunucu arasında herhangi bir oturum yakınlığı veya yapışkan oturum olmamalıdır.

Sunucu tarafında istemci oturumu ile ilgili herhangi bir durumu depolayamayan sunucu ile ilgili bu kısıtlamaya statelessness denir. Başka bir deyişle, her HTTP isteği tam bir izolasyon içinde gerçekleşir. İstemcilerin bu durum bilgisi olmayan API'lere erişmesini sağlamak için, sunucuların da istemcinin durumu kendi tarafında oluşturmak/sürdürmek için ihtiyaç duyabileceği her türlü bilgiyi içermesi gerekir. Durumsuz hale gelmek için sunucular, istemcinin kimlik doğrulama/yetkilendirme ayrıntılarını bile saklayamaz. İstemci, her istekle kimlik doğrulama bilgileri sağlar. Bu nedenle, her istek tek başına olmalı ve geçmişte aynı müşteriyle yapılan önceki konuşmadan etkilenmemelidir.

Statelessness, API'leri birden çok sunucuya dağıtarak milyonlarca eşzamanlı kullanıcıya ölçeklendirmeye yardımcı olur. Oturumla ilgili bir bağımlılık olmadığından herhangi bir sunucu herhangi bir isteği işleyebilir.

13.CSRF saldırısı nedir? Nasıl engellenir?

CSRF (Cross Site Request Forgery) genel yapı olarak sitenin açığından faydalanarak siteye sanki o kullanıcıymış gibi erişerek işlem yapmasını sağlar. Genellikle GET requesleri ve SESSION işlemlerinin doğru kontrol edilememesi durumlarındaki açıklardan saldırganların faydalanmasını sağlamaktadır.

Bu tür açıkları kapatmak için en pratik yol ise token kullanımıdır.

14.HTTP request ve response componentleri nelerdir?

Bir HTTP request için 3 component vardır:

- request line: İstek satırında kullanılacak HTTP yönetimi,isteğin URI'sini ve kullanılacak HTTP protokol versiyonunu yerleştirir,GET/api/yazar http/1.1.
- request header: İsteğin başlığının nerede olduğunu belirtir, en.wikipedia.org.
- request body(opsiyonel): Sunucuya göndereceğimiz ek bilgileri nereye koyduğumuzu gösterir.

Bir HTTP response için de 3 tane component vardır:

- Status line : Üç önemli bileşen içerir.
  1.HTTP version: HTTP sürüm numarası, sunucunun yanıt mesajını uyumlu hale getirmeye çalıştığı HTTP belirtimini gösterir.
  2.HTTP response code : Bu, istenen kaynak için Sunucu durumunu gösterir. Talebin sonucunu gösteren 3 haneli bir sayıdır. Örneğin, 404, kaynak bulunamadı ve 200, yanıtın tamam olduğu anlamına gelir.
  3.reason-phrase : Durum Kodunu insan tarafından okunabilir biçimde özetlediği için Durum Metni olarak da bilinir.
- Response header : Yanıt Başlığı, yanıt olarak döndürülen içerikle ilgili bilgileri, meta verileri, onu gönderen Sunucu hakkındaki verilerle birlikte anahtar/değer çiftleri olarak içerir. Bu bilgi, Müşterinin/Tarayıcının yanıt verilerinin ne şekilde kullanılacağına karar vermesine yardımcı olur. En popüler yanıt başlıkları Content-Length, Content-Type, Date, Server, Set-Cookie vs.'dir. Sunucu gerektiği kadar başlık gönderebilir. Her başlık, iki nokta üst üste (:) ile ayrılmış bir anahtar/değer çifti olarak gönderilir.
- Body (Opsiyonel) : Başarılı bir yanıt durumunda, Müşteriye/Kullanıcıya istekte istenen kaynak ile hizmet vermek için Kaynak gösterimi kullanılır. Gövde isteğe bağlı olmasına rağmen, İstemci ve Sunucu arasındaki iletişimin en temel parçalarından biridir ve çoğu zaman gönderilir. Gövde verileri taşır ve başlıklarda buna göre belirtilen json, html, image, vb. gibi birçok formattan birinde olabilir.

……

1.JPA nedir?

OOP bir dille veritabanı işlemleri yapan bir uygulama geliştirirken ilk olarak verilerin veritabanından alınarak programlama diline ait veri yapısı ile ifade edilmesi/dönüştürülmesi gerekir. Bu dönüşüm

sırasında veritabanındaki sütun adlarının değişmesi, veri tipinin değişmesi, uygun veri türüne dönüşüm yapılamaması gibi çeşitli hatalar ile karşılaşılır.

ORM(Object Relational Mapping) yani veritabanı ve programama dili arasında ilişki kurarak her bir alanı programlama diline uygun veri yapısıyla eşleştiren yapının araçları genellikle veritabanında yer alan tabloları sınıflarla ifade edererek eşlemeyi yapar.

JPA(Java persistance API), temel objectlerin korunması için kalıcılık sağlayan bir ORM frameworktür. Bu relational objectler tablo biçiminde tutulur. Javax.persistence paketi ile import edilir.

2.Spring data repository interface içindeki finder metotları için kurallar nelerdir?

Keyword	**Sample**
And	findByLastnameAndFirstname
Or	findByLastnameOrFirstname
Is , Equals	findByFirstname , findByFirstnameIs ,findByFirstnameEquals
Between	findByStartDateBetween
LessThan	findByAgeLessThan
LessThanEqual	findByAgeLessThanEqual
GreaterThan	findByAgeGreaterThan
GreaterThanEqual	findByAgeGreaterThanEqual
After	findByStartDateAfter
Before	findByStartDateBefore
IsNull	findByAgeIsNull
IsNotNull,NotNull	findByAge(Is)NotNull
Like	findByFirstnameLike
NotLike	findByFirstnameNotLike
StartingWith	findByFirstnameStartingWith
EndingWith	findByFirstnameEndingWith

Keyword	Sample
Containing	findByFirstnameContaining
OrderBy	findByAgeOrderByLastnameDesc
Not	findByLastnameNot
In	findByAgeIn(Collection<Age> ages)
NotIn	findByAgeNotIn(Collection<Age> ages)
True	findByActiveTrue()
False	findByActiveFalse()
IgnoreCase	findByFirstnameIgnoreCase

3.PagingAndSortingRepository nedir?

Pagigng, tek seferde küçük bir veri bölümü görmeyi sağlar. Sorting, kullanıcıların verileri düzenli görmesini sağlar. Paging iki alandan oluşur – sayfa boyutu ve sayfa numarası. Sorting, tablodaki birden çok alandan biri üzerinde yapılır.

PagingAndSortingRepository, CrudRepository'nin bir uzantısı olarak Spring Data JPA tarafından packing ve sorting API'lerini kullanmak için sağlanır.

Page findAll(Pageable pageable), Pageable nesnesinde sağlanan disk belleği kısıtlamasını karşılayan varlıkların bir Sayfasını döndürür.

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

4.findBvId() ve getOne() arasındaki farklar nelerdir?

İki metot da temel alınan veri deposundan bir nesneyi almak için kullanılır. Farkları;

- findById() EAGER loaded, getOne() lazy loaded.
- findById() Database üzeirnden, getOne() entity üzerinden veri getirir.
- findById() CrudRepository ile, getOne() JpaRepository ile çalışır.
- findById() obje yoksa null, getOne() EntityNotFoundException döner.
- findById() opsiyonel bir obje, getOne() entityden verilen tanımlayıcıya bağlı bir referans döndürür.
- findById() veri tabanına ek gidiş-dönüş gerekir, getOne() daha iyi performanlıdır.
- findById() EntityManager.find(), getOne() EntitManager.getReference().
- findById() obje eager loaded olduğundan tüm attribute'lere erişilebilir , getOne() yalnızca nesnenin özelliklerine erişim gerekli olmadığında kullanışlıdır

5.@Query ne için kullanılır?

Veri tabanından bilgi almak için yapılan isteğe query(sorgu) denir.

6.Hibernate'te lazy loading nedir?

Lazy loading, Hibernate'teki tüm entityleri almak için kullanılan bir tekniktir. Temel amacı objeleri veri tabanına getirmektir.

Hibernate, inheritance'ı destekler. Bir parent, birden fazla child record'a sahipse lazy loading kullanılarak gerekli bütün classların yüklenmesi sağlanır.

Gereksiz yüklenmeleri önleyerek performansı artırır ve bellek gereksinimleirni azaltır.

fetch= FetchType.LAZY kullanımıyla etkinleşir. Syntax'ı @OneToOne(fetch= FetchType.LAZY).

7.SQL injection saldırısı nedir? Hibernate buna açık mıdır?

SQL injection, kötü niyetli SQL koduna sahip kullanıcıların veritabanlarının arka ucunu manipüle ederek gizli bilgilere erişmesine izin veren yaygın bir saldırı vektörüdür. SQL enjeksiyonu, birinin bilginiz olmadan veritabanınızda çalıştırılmak üzere MySQL deyimi eklemesi eylemini ifade eder. Enjeksiyon genellikle bir kullanıcıdan adı gibi bir girdi istediğinizde gerçekleşir ve bir ad yerine size bilmeden veritabanınızda çalıştıracağınız bir MySQL ifadesi verirler. Bu veriler hassas iş bilgilerini, özel müşteri ayrıntılarını veya kullanıcı listelerini içerebilir. Başarılı bir SQL enjeksiyonu, tüm veritabanlarının silinmesine, hassas verilerin yetkisiz kullanımına ve bir veritabanına istenmeyen yönetici haklarının verilmesine neden olabilir.

Hibernate, elle kodlanmış SQL deyimleri oluşturmamızı engeller, ancak savunmasız kod yazmamızı engellemez. Hibernate, native SQL kullanımına izin verir ve HQL adlı, birincisi SQL injectiona, sonraki ise HQL injectiona yatkın olan özel bir query dili tanımlar. Çünkü JPA sorgusu oluşturmak için doğrulanmamış girdi kullandığımızdan, Hibernate SQL Injection saldırısına da açıktır.

8.Hibernate criteria API nedir?

HQL ile veri tabanından veri okumak için queryler elle hazırlanır. Daha iyi performans için ayarlanmış queryler kullanılır. Hibernate sadece HQL'den SQL'e dönüşüm yapar, queryleri ayarlamaz. Queryleri yazmak ve açıkça ayarlamak için Hibernate Criteria API'si kullanılır. Bu API, sorgu(query) oluşturmaya gerek olmadan, hibernate tuned query hazırlar.

Object class oluşturmak için Criteria criteria = session.thecreateCriteria(x.class); kullanılır.

9.Erlang nedir? Neden RabbitMQ için required?

Erlang, eşzamanlı ve fonksiyonel bir programlama dili ve garbage-collected runtime sistemidir. Erlang hata toleransı, dağıtım, dinamik yazma, tekli atama için yerleşik desteğe sahiptir. OTP(Open Telecom Platform) geniş bir kütüphane sağlar. Erlang, dağıtılmış, hata toleranslı, soft rea-time, sistem durmadan kodun değiştirilebileceği uygulamalar için tasarlanmıştır.

Erlang OTP, RabbitMQ'da olduğu gibi, çok sayıda eşzamanlı işlemi işleme konusunda yerel yeteneklere sahip, kararlı, güvenilir, hataya dayanıklı ve yüksek düzeyde ölçeklenebilir sistemler oluşturmak için uyarlanmış bir teknolojidir.

Erlang programlama dilinde yazılan RabbitMQ sunucusu, kümeleme ve yük devretme için Erlang/OTP çerçevesi üzerine kurulmuştur. Aracıyla arabirim oluşturmak için istemci kitaplıkları, tüm ana

programlama dilleri için mevcuttur. RabbitMQ'yu kurmadan önce Erlang'ı kurmamız gerekiyor, çünkü RabbitMQ, Erlang üzerine kurulu.

10.JPQL nedir?

Persistence Query Language (JPQL), özellikle ilişkisel bir veritabanındaki kalıcı varlıklar üzerinde çeşitli veritabanı işlemlerini gerçekleştiren, nesne yönelimli bir sorgu dilidir. JPQL, varlık nesne modeli kullanılarak SQL sözdizimine dayalı olarak geliştirilmiştir. Bir JPQL sorgusu, tablolardan alan değerlerinden daha fazla nesneleri alabilir ve döndürebilir. JPA'nın asıl amacının JPQL'yi SQL'e dönüştürmek olduğunu söyleyebiliriz. Bu, SQL sözdizimine dayalı olarak kurulmuş olsa da, veritabanı doğrudan etkilenmeyecektir.

JPQL, Varlık JavaBeans Sorgu Dili'nin (EJBQL) bir uzantısıdır ve ona bazı önemli özellikler ekler: - Birleştirme işlemlerini gerçekleştirme. Verileri toplu olarak güncelleme ve silme. Sıralama ve gruplandırma yan tümceleri ile toplama işlevi gerçekleştirme. Tek ve çoklu değer sonuç türleri.

11.Bir entity nesnenin oluşum adımları nelerdir?

1. Entity üreten Factory Object oluşturulur.
2. Factory'den entity manager oluşturulur.
3. Entity manager başlatılır.
4. Bir veri, veri tabanında persistence yapılır.
5. Transactain metodu kapatılır.
6. Factory kaynakları bırakılır.

```
EntityManagerFactory emf = Persistence.createEntityManagerFactory("bla bla bla");
EntityManager em = emf.createEntityManager();
Em.getTransaction().begin();
em.persist(bla1);
em.getTransaction().commit();
emf.close();
em.close();
```

12.Entity mapping çeşitleri?

One-to-one : Bir varlığın örneğinin başka bir varlığın örneğiyle ilişkilendirildiği tek değerli bir ilişkiyi temsil eder. Bu tür ilişkilendirmede, kaynak varlığın bir örneği, hedef varlığın en fazla bir örneği ile eşlenebilir. Örneğin, bir kişinin yalnızca bir pasaportu olabilir ve bir kişiye bir pasaport atanır.

One-to-many : Bu tür eşleme, bir varlığın diğer varlıkların bir koleksiyonuyla ilişkilendirildiği koleksiyon değerli ilişki kategorisine girer. Bu tür ilişkilendirmede, bir varlığın örneği, başka bir varlığın herhangi bir sayıda örneği ile eşlenebilir. Örneğin, bir Kişi örneğinin birden fazla Banka Hesabı örneği olabilir, ancak bir banka hesabı örneğinin hesap sahibi olarak en fazla bir kişi örneği olabilir. Bir takımı yöneten tek bir yönetici varken, birden fazla takımı yöneten çalışanlar var".

Many-to-one : Bu eşleme türü, bir varlık koleksiyonunun benzer varlıkla ilişkilendirilebileceği tek değerli bir ilişkiyi temsil eder. İlişkisel veritabanında, bir varlığın birden fazla satırı, başka bir varlığın aynı satırına başvurabilir. Bire çok eşleme ile ilgilidir, ancak fark perspektiften kaynaklanmaktadır. Örnek olarak, herhangi bir sayıda kredi kartı bir müşteriye ait olabilir ve herhangi bir kredi kartı

olmayan bazı müşteriler olabilir, ancak bir sistemdeki her kredi kartının bir çalışanla ilişkilendirilmesi gerekir (toplam katılım gibi). Tek bir kredi kartı birden fazla müşteriye ait olamazken.

Many-to-many : Çoktan Çoka eşleme, herhangi bir sayıda varlığın diğer varlıkların bir koleksiyonuyla ilişkilendirilebileceği, koleksiyon değerli bir ilişkiyi temsil eder. İlişkisel veritabanında, bir varlığın birden fazla satırı, başka bir varlığın birden fazla satırına başvurabilir. Bir kişinin birden fazla yeteneği olabilir. Bir beceriyi birden fazla kişi elde edebilir. Bir müşteri herhangi bir sayıda ürünü satın alabilir ve bir ürün birçok müşteri tarafından satın alınabilir.

13.Bir entity'nin propertileri nelerdir?

Entity, Java Persistence kütüphanesindeki uygulama tanımlı bir nesnedir. Bu nedenle uygulama tanımlı(application-defined) nesnenin sahip olması gereken özelliklerini taşır.

Persistability : Bir nesne, veritabanında depolanıyorsa ve herhangi bir zamanda erişilebilirse kalıcı olarak adlandırılır.

Persistability Identity : Java'da her entity benzersizdir ve bir nesne kimliğini temsil eder. Benzer şekilde, nesne kimliği bir veritabanında depolandığında, kalıcı kimlik olarak temsil edilir. Bu nesne kimliği, veritabanındaki birincil anahtarla aynıdır.

Transactionality : Entity oluşturma, silme, güncelleme gibi çeşitli işlemleri gerçekleştirebilir. Her işlem veritabanında bazı değişiklikler yapar. Veritabanında yapılan her türlü değişikliğin atomik olarak başarılı veya başarısız olmasını sağlar.

Granularity : Entityler, ilkel, ilkel sarmalayıcılar veya tek boyutlu duruma sahip yerleşik nesneler olmamalıdır.

14.Spring Data JPA'daki CrudRepository ve JpaRepository arasındaki fark?

CrudRepository	**JpaRepository**
Spring Data Repository interface'den extend edilir.	PagingAndSortingRepository 'den extend edilir.
CRUD operasyonları için metotları vardır.save(), saveAll(), findById(), findAll()	CrudRepository ve PagingAndSortingRepository içinden tüm APIyi sağlar.Forinstance, flush(), saveAndFlush(), saveAllAndFlush(), deleteInBatch()
Pacing ve sorting yok.	Packing ve sorting sağlar. Yeni metotlar da ekleneblir.
The saveAll(Iterable<S> entities) metodu Iterable döner.	The saveAll(Iterable<S> entities) metodu List döner.