PATİKA TÖDEB
JAVA SPRİNG BOOTCAMP

ÖDEV 4

BATUHAN AYYILDIZ

1-

JPA(Java Persistence API) birbiriyle ilişkili verileri idare etmek için kullanılan bir java yönergesidir. Framework değildir. Konsept veya rehber gibi düşünülebilir. Bu yüzden herhangi bir framework tarafından uygulanabilir. Java obje/sınıfları ile ilişkisel database'ler arasındaki verilere ulaşmamızı ve onları sürdürülebilir ve kalıcı kılmamızı sağlar.

2-

İsim düzeni aşağıdaki gibidir.

find[limit]By[property/properties expression][comparision][ordering operator]

limit---> Bu kısım aramayı sınırlandırmak için yazılır. first/top kelimeleri ile bu sınırladnırma yapılır.

Örnek : findTop3ByLastname, findFirstByOrderByLastnameAsc

property/properties expression---> Bu kısım filtrelenecek özelliği veya entity'i belirtmek için yazılır. Birden fazla özellik için And yada Or kullanılabilir.

Örnek : findByLastnameOrFirstname, findByFirstname

comparison---> Özellik belirtildikten sonra karşılaştırma durumu belirtilir.

Örnek : findByAgeGreaterThan, findByAgeIsNull

ordering operator---> İsteğe bağlı olarak bu kısımda sıralama belirtilir.(Azalan veya artan şekilde)

Örnek : findByLastnameOrderByFirstnameDesc, findByLastnameOrderByFirstnameAsc

3-

CrudRepository'e bir eklentidir. Entity'leri, pagination 'ı ve sorting abstraction'ı kullanarak elde eden ek methodları barındırır. CrudRepository'ları methodları miras almıştır.

pagination--> Dijital içeriklerin ayrı sayfalara dağıtılması.

Method : Page<T> findAll(Pageable pageable)

sorting abstraction --> Belirli bir standarta göre olan verileri sıralanmış olarak elde eder.

Method : Iterable<T> findAll(Sort sort)

4-

findById(), CrudRepository'nin içinde bulunan bir methodtur. getOne() ise JpaRepostiory'nin içinde bulunur. findById() aranan şey bulamadığında null döndürür. getOne() ise exception fırlatır. findById() çağrıldığından database'e gider ve objeyi oradan getirir. Eager işleme sahiptir yani değişkene ifade verildiğinde anında hesaplanır ve değişkenin değeri atanır. getOne() ise objenin referansını döner ve lazy fetch'tir. Yani değere gerçekten ihtiyaç olana kadar değer hesaplanmaz. eğer değere erişilmezse, değeri belirleyen kod çalışmaz. getOne() database'e gitmediği için daha hızlıdır ve performansı daha iyidir. Ama obje içinde bulunan başka bir objeye erişmek gerekince hata verir.

5-

@Query anotasyonu, spring boot'ta sql query'leri yazmak için kullanılır. Böylelikle database'den istenen filtrelere göre veriler çekilebilir.

Örnek:

@Query("SELECT e FROM EMP e WHERE e.city = 2")

List<EMP> getAll();

6-

Lazy loading bir design pattern'dır. Obje'nin initialize edilmesini olabildiğince geciktirmek için kullanılır. Hibernatede de one to one, many to one gibi ilişkiler içeren veri tablolarını fetchleme tiplerinden biridir. Mesela, A ve B lazy fetch'e sahipse A'yı çağırdığımızda B açıkça çağırılmadan belirtilmeden initialize olmaz ve belleğe yüklenmez.

7-

Zararlı SQL kodlar kullanarak backend database'ini manipüle edip gösterilmek istemeyen bilgilere ulaşılmasıdır. (Kullancı listesi , kredi kartı detayları vb.) . Hibernate, ORM(Object relational Mapping) framework olduğu için SQL Injection'a karşı daha korunaklıdır ama yine de %100 koruma sağlamaz.

8-

Criteria API, mantıksal koşullar veya filtre kuralları kullanılıp programlı olarak bir ölçüt sorgu (criteria query) nesnesi oluşturmak için kullanılan bir yoldur.

9-

Erlang, genel amaçlı, eş zamanlı, dinamik , fonksiyonel ve aynı zamanda atık toplama özelliğine sahip bir programlama dili ve çalışma ortamıdır. Erlang'ın sağladığı OTP(Open Telecom Platform) geniş çaplı kütüphanelerin birleşiminden oluşuyor ve ASN.1 derlemeleri yapmaktan WWW server sağlamaya kadar benzeri alanlarda kullanılıyor. RabbitMQ da kurumsal bir mesajlaşma aracı olduğu için telekomünikasyon konusunda büyük avantajlar sağlayan Erlang bu araç için gerekli bir hale geliyor.

10-

JPQL(Java Persistence Quey Language), Entity nesneleri sorgulamak için JPA tarafından tanımlanan bir dildir. HQL(Hibernate Query Language) 'e benzerdir. Bu iki dilinde SQL'e benzerlikleri vardır ama kullanılan argümanlar veritabanı tabloları değil de Entity nesneleridir.

11-

1- Entity manager factory nesnesi oluşturulur

EntityManagerFactory emf=Persistence.createEntityManagerFactory("Student_details");

2- Factory'den entity manager edinilir

EntityManager em=emf.createEntityManager();

3- Entity manager initialize edilir

em.getTransaction().begin();

4- Veri, ilişkili database'e yollanır.

em.persist(s1);

5- Etkileşim kapatılır

em.getTransaction().commit();

6- Factory kaynakları serbest bırakılır.

emf.close();

em.close();

12-

4 tane çeşit vardır.

one to one ---> Bir entity beanle diğer entity bean'in ilişkilendirilmesidir

one to many ---> Bir nesne birden fazla nesnenin instance'na referans verebilir

many to one ---> Birden fazla nesne tek nesneye referans verebilir.

many to many ---> Birden fazla nesne birden fazla nesneye bakılan açıdan bağımsız referans verebilir. Örneği bir kişi birden fazla projede çalışabilir ve bir projede birden fazla kişi çalışıyor olabilir.

13-

Entity, veritabanı ile yazılım arasında bağlantı, ilişki kurmamızı sağlayan kalıcı nesnelerdir. Entity'nin 3 özelliği vardır

1- Kalıcılık : Ulaşılabilir ve kaydedilebilir olmalıdır.
2- Kimlik: Sadece kendine ait bir kimlik bilgisine sahip olmalıdır.
3- İşlem (Transactionality): İşlem uygulanabilmelidir. Veritabanına ekleme yapmak güncellemek veya silmek gibi. Eğer işlem herhangi bir sebepten tamamlanmazsa da geri dönüş yapılıp işleme baştan başlanmalıdır.

14-

JpaRepository, PagingAndSortingRepository'e extend eder. PagingAndSortingRepository de CrudRepository'e extend etmiştir. Bu yüzde JpaRepository hem kendine ait özellikler hem PagingAndSortingRepository'e ait özellikleri hem de CrudRepository'e ait özellikleri barındırır. Bu yüzden daha kapsamlıdır ve daha fazla metod barındırır. Ek gelen methodlar uygulama için gerekli görülmediği zaman CrudRepository'nin kullanılması daha uygundur.