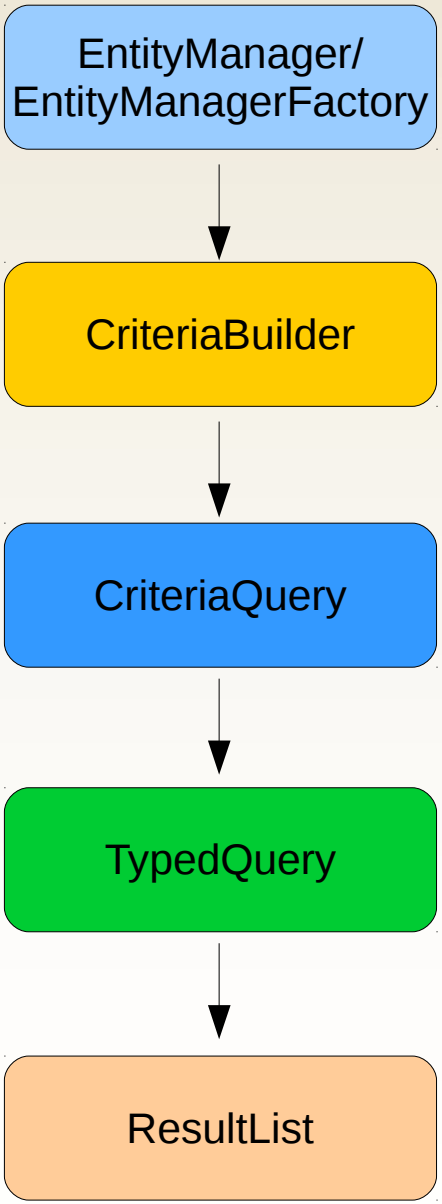
JPA Criteria API ile Sorgulama

JPA ve Criteria API

- Criteria API **programatik** olarak sorgu hazırlamak için tasarlanmıştır
- Genellikle **parametrik sorgu ekranları**nda kullanılır
- Bu tür ekranlarda kullanıcının seçtiği kriterlere göre **sorgu dinamik olarak** oluşturulur
- Hibernate implemantasyonu **Serializable**'dır, dolayısı ile **Criteria sorguları** DB veya dosya sisteminde saklanıp tekrar kullanılabilir

JPA Criteria API ile Çalışma Adımları



JPA Criteria API ile Çalışma Adımları: CriteriaBuilder

- Öncelikle **EntityManagerFactory** veya **EntityManager** kullanılarak bir **CriteriaBuilder** nesnesi elde edilmelidir
- **CriteriaBuilder** nesnesi **sorgunun değişik bölümleri**ni (selection, expression, predicate, ordering) oluşturmak için kullanılır

```
CriteriaBuilder criteriaBuilder =
                        entityManager.getCriteriaBuilder();
```

EntityManagerFactory üzerinde de benzer bir metot vardır

JPA Criteria API ile Çalışma Adımları: CriteriaQuery

- CriteriaBuilder kullanılarak da **bir veya daha fazla entity üzerinde navigate etme**yi sağlayacak bir **CriteriaQuery** nesnesi oluşturulur
- **CriteriaQuery** üzerinde **projection, join, restriction gibi bölümler** eklenecektir

```
CriteriaQuery<Pet> criteriaQuery =
                        criteriaBuilder.createQuery(Pet.class);
```

JPA Criteria API ile Çalışma Adımları: Query Root

- Sorgunun **root entity**'si, join'ler, restriction'lar, projection'lar vb belirtilir

```
Root<Pet> pet = criteriaQuery.from(Pet.class);

...
```

JPA Criteria API ile Çalışma Adımları: TypedQuery

- **EntityManager** ile **CriteriaQuery** nesnesinden bir de **TypedQuery** nesnesi elde edilir
- Sorgu **parametreleri, hint bilgisi, flush modu** vs hepsi **TypedQuery** üzerinden yönetilir

```
TypedQuery<Pet> typedQuery =
                        entityManager.createQuery(criteriaQuery);
```

JPA Criteria API ile Çalışma Adımları: Result List

- Son adımda **TypedQuery getResultList**, **getSingleResult** veya **getFirstResult** gibi metotlarla çalıştırılarak **sorgu sonuçları** elde edilebilir

```
List<Pet> result = typedQuery.getResultList();
```

JPA Criteria API ve Sorgu Kısıtları: Where Clause

```
CriteriaQuery<Pet> criteriaQuery =
                            cb.createQuery(Pet.class);

Root<Pet> pet = criteriaQuery.from(Pet.class);

criteriaQuery.where(pet.get("name").isNull());
```

CriteriaQuery.where() metodu Expression veya Predicate nesneleri kabul ederek sorgu ifadesindeki kısıtları tanımlamaya imkan verir

```
CriteriaQuery<Pet> criteriaQuery =
                            cb.createQuery(Pet.class);

Root<Pet> pet = criteriaQuery.from(Pet.class);

criteriaQuery.where(pet.get("name").in("Maviş", "Tarçın"),
                    pet.get("birthDate").isNotNull());
```

Where ifadesindeki her bir Predicate AND ile eklenir

JPA Criteria API ve Sorgu Kısıtları: Where Caluse ve OR

```
CriteriaBuilder cb = em.getCriteriaBuilder();

CriteriaQuery<Pet> cq = cb.createQuery(Pet.class);

Root<Pet> pet = cq.from(Pet.class);

Predicate predicate1 = cb.like(pet.get("name"), "F%");
Predicate predicate2 = cb.between(pet.get("birthDate"),
                       new GregorianCalendar(2000, 1, 1).getTime(),
                       new GregorianCalendar(2010, 12, 31).getTime());


cq.where(cb.or(predicate1,predicate2));

TypedQuery tq = em.createQuery(cq);
List<Pet> resultList = tq.getResultList();
```

CriteriaBuilder.or() ve and() metotları ile input argüman olarak verilen Predicate nesneleri arasında OR veya AND ilişkisi kurulabilir

JPA Criteria API ve İlişkilerin JOIN Edilmesi: Cartesian

```
CriteriaBuilder cb = em.getCriteriaBuilder();

CriteriaQuery cq = cb.createQuery();

Root<Pet> pet = cq.from(Pet.class);
Root<Visit> visit = cq.from(Visit.class);

cq.where(cb.equal(pet, visit.get("pet")));

cq.select(visit);

TypedQuery tq = em.createQuery(cq);
List<Visit> resultList = tq.getResultList();
```

cq.from() metodu ile birden fazla root entity eklenebilir eklenen query root'lar kartezyen çarpım oluştururlar

where clause'una eklenecek bir restriction ile de kartezyen çarpım üzerinde sınırlama yapılabilir

Son olarak TypedQuery Oluşturmadan evvel CriteriaQuery select() metodu ile hangi Entity'nin dönüleceği belirtilmelidir

JPA Criteria API ve İlişkilerin JOIN Edilmesi: INNER

```
CriteriaBuilder cb = em.getCriteriaBuilder();

CriteriaQuery<Pet> cq = cb.createQuery(Pet.class);

Root<Pet> pet = cq.from(Pet.class);

Join<Pet, Owner> owner = pet.join("owner");

TypedQuery tq = em.createQuery(cq);
List<Pet> resultList = tq.getResultList();
```

Root query üzerinde join metodu ile inner join yapmak mümkündür

Join() metotları her zaman Join<X,Y> şeklinde bir nesne dönerler
X source entity, Y ise join'deki target entity nesnesine karşılık gelir

JPA Criteria API ve İlişkilerin JOIN Edilmesi: JOIN Con.

```
CriteriaBuilder cb = em.getCriteriaBuilder();

CriteriaQuery<Pet> cq = cb.createQuery(Pet.class);

Root<Pet> pet = cq.from(Pet.class);

Join<Pet, Owner> owner = pet.join("owner");
owner.on(owner.get("id").in(7,8,9,16))
Join<Owner, Address> address = owner.join("address");
```

İstenirse join condition değiştirilebilir, yada ilave join'ler japılabilir

```
cq.where(cb.like(address.get("street"), "1%"));

TypedQuery tq = em.createQuery(cq);
List<Pet> resultList = tq.getResultList();
```

Where clause'unda da join'ler ile ilgili ilave kısıtlamalara gidilebilir

JPA Criteria API ve İlişkilerin JOIN Edilmesi: OUTER

```
CriteriaBuilder cb = em.getCriteriaBuilder();

CriteriaQuery<Pet> cq = cb.createQuery(Pet.class);

Root<Pet> pet = cq.from(Pet.class);

Join<Pet, Visit> visits = pet.join("visits",JoinType.LEFT);

cq.where(visits.isNull(),cb.like(pet.get("name"), "M%"));

TypedQuery tq = em.createQuery(cq);
List<Pet> resultList = tq.getResultList();
```

Inner join yerine LEFT outer join Yapılabilir

Where clause'unda Join üzerinden de Kısıtlamalara gidilebilir

JPA Criteria API ve Sorgu Parametreleri

```
CriteriaBuilder cb = em.getCriteriaBuilder();

CriteriaQuery<Pet> cq = cb.createQuery(Pet.class);

Root<Pet> pet = cq.from(Pet.class);

ParameterExpression<List> namesParameter =
                    cb.parameter(List.class,"names");

cq.where(pet.get("name").in(namesParameter));


TypedQuery<Pet> tq = em.createQuery(cq);

tq.setParameter("names",Arrays.asList("Maviş","Tarçın","Sarı"));

List<Pet> resultList = tq.getResultList();
```

CriteriaBuilder ile bir ParameterExpression nesnesi oluşturulur ve CriteriaQuery'nin where caluse'unda Predicate içerisinde kullanılır

TypedQuery.setParameter metodu ile parametre değerleri set edilir ve ardından sorgu çalıştırılır

JPA Criteria API ve Projection

```
CriteriaBuilder cb = em.getCriteriaBuilder();

CriteriaQuery<Tuple> cq = cb.createQuery(Tuple.class);

Root<Pet> pet = cq.from(Pet.class);

cq.multiselect(pet.get("name").alias("name"),
               pet.get("birthDate").alias("birthDate")
              );

TypedQuery<Tuple> tq = em.createQuery(cq);

List<Tuple> resultList = tq.getResultList();

for(Tuple t:resultList) {
    System.out.println(t.get("name") + " - " + t.get("birthDate"));
}
```

Multiselect() metoduna project edilecek property path'leri belirtilir
alias belirtilmesi daha sonra bu property değerlerine Tuple üzerinden erişmek için önemlidir

JPA Criteria API ve Sorgu Sonuçlarının Sıralanması

```
CriteriaBuilder cb = em.getCriteriaBuilder();

CriteriaQuery<Pet> cq = cb.createQuery(Pet.class);

Root<Pet> pet = cq.from(Pet.class);

cq.orderBy(
            cb.asc(pet.get("name")),
            cb.desc(pet.get("birthDate"))
          );

TypedQuery<Pet> tq = em.createQuery(cq);

List<Pet> resultList = tq.getResultList();
```

CriteriaBuilder'ın asc() veya desc() metotları ile elde edilen Order nesneleri CriteriaQuery'nin orderBy metodu ile eklenir

JPA Criteria API ve Sorgu Sonuçlarının Gruplanması

```
CriteriaQuery<Date> cq = cb.createQuery(Date.class);

Root<Pet> pet = cq.from(Pet.class);

cq.groupBy(pet.get("birthDate"),pet.get("name"));

cq.having(pet.get("name").in("Maviş","Tarçın"));

cq.select(pet.get("birthDate"));

TypedQuery<Date> typedQuery = em.createQuery(cq);

List<Date> resultList = typedQuery.getResultList();
```

CriteriaQuery'nin groupBy() ve having() metotları ile sorgu sonuçları üzerinde gruplama ve grup içerisinde ilave kısıtlamalar yapmak mümkündür. Select() metodu ile de sorgudan dönecek kısım belirtilebilir

JPA Criteria API ve Toplu İşlemler

JAVA Eğitimleri

- CriteriaUpdate ve CriteriaDelete arayüzleri üzerinden **toplu güncelleme ve silme işlemleri** de yapılabilir
- Ancak bu işlemler **persistence context ile senkron gerçekleşmez**
- Optimistic **versiyon bilgisi de güncellenmez**
- Update işleminde versiyon alanının da **explicit olarak** ele alınması gerekir

CriteriaUpdate ile Toplu Güncelleme

```
EntityManager em = emf.createEntityManager();
em.getTransaction().begin();

CriteriaBuilder cb = em.getCriteriaBuilder();

CriteriaUpdate<Visit> cu = cb.createCriteriaUpdate(Visit.class);

Root<Visit> root = cu.from(Visit.class);

cu.set("visitDate", new Date());

cu.where(cb.isNull(root.get("description")));

Query query = em.createQuery(cu);

int updateCount = query.executeUpdate();

em.getTransaction().commit();
```

Update/Delete işlemlerinde TX başlatılması şarttır

CriteriaBuilder'ın createCriteriaUpdate() metodu ile bir CriteriaUpdate nesnesi oluşturulur

Bu nesne üzerinde güncellenecek alanlar ve değerler belirtilir

ardından EntityManager.createQuery() metodu ile bir Query nesnesi elde edilerek executeUpdate() metodu ile çalıştırılır

CriteriaDelete ile Toplu Silme

```
EntityManager em = emf.createEntityManager();
em.getTransaction().begin();

CriteriaBuilder cb = em.getCriteriaBuilder();

CriteriaDelete<Visit> cd = cb.createCriteriaDelete(Visit.class);

Root<Visit> root = cd.from(Visit.class);

cd.where(cb.isNull(root.get("visitDate")));

Query query = em.createQuery(cd);
int deleteCount = query.executeUpdate();

em.getTransaction().commit();
```

JPA Criteria API ve Toplu İşlemler

- Criteria API ile gerçekleştirilen UPDATE/DELETE işlemlerindeki sorgularda **Join ifadeleri yer alamaz**

İletişim

- Harezmi Bilişim Çözümleri
- Kurumsal Java Eğitimleri
- http://www.java-egitimleri.com
- info@java-egitimleri.com