

Hibernate

Burak Bayramlı

Bu belge, \LaTeX ile üretilmiştir

Nesneler Arası İlişkiler

Hibernate tablolar arası ilişkileri nesnelere yansıtılabilir (ya da tam tersi). İlişki türleri:

- Bire bir (one to one)
- Bire çok (one to many)
- Çoka çok (many to many)

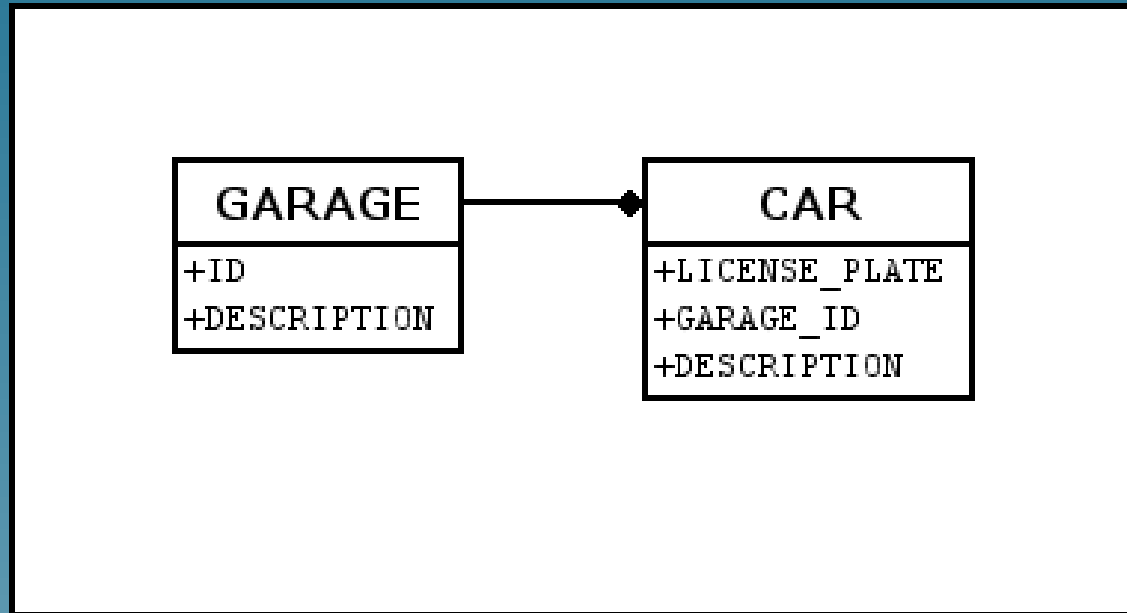
Bire Çok

- Bir nesne birden fazla nesne ile alâkalı olduğu zaman
- Arada ki bağlantı veri tabanından bir foreign key
- Bire çok ilişkisi, iki taraflı olduğu zaman, karşıda muhakkak çoka bir (many to one) ilişkisi olmalıdır.

Annotation

- Hibernate'de ilişki türlerinin kendi annotation'ı var.
- Bire bir: @OneToOne
- Bire çok: @OneToMany
- Çoka çok: @ManyToOne

Bire Çok



Metot

- İlişkide bağlantı (foreign key) kolonunu deklare edecek taraf, @JoinColumn ibaresini içerecek taraftır.
- Diğer taraf, mappedBy="relation" gibi bir ibareyi kullanır, ve diğer herşeyi karşı tarafa bırakır.
- Bahsedilen relation, diğer nesnede bize işaret eden relation ismidir.

Metot

```
@Entity
public class A {
    @Id
    int aId;
    ...
    @OneToMany(mappedBy='a')
    List<B> blist;
    ...
}
```

Metot

```
@Entity
public class B {
    @Id
    ...
    @ManyToOne()
    @JoinColumn(name="a_id")
    A a;
    ...
}
```

Veri İşlemleri

- Daha önce `EntityManager` nesnesini kullandık.
- Bu class üzerinde `persist` gibi daha birçok metot var.
- Meselâ `find`, `flush` gibi.

find

- Bir class ve id verilerek tek bir nesneyi bulmamızı sağlar
- Id için geçilen parametre, Pojo üzerinde @Id işaretlenmiş olan öge tipinde bir nesne olmalıdır.
- Meselâ: `A a = entityManager.find(A.class, 2)` gibi.

Görevler

- Car ve Garage class'ları kodlanacak.
- Car üzerinde `description` ve `licensePlate` olmalı ve `licensePlate` `license_plate` kolonuna map etmeli.
- Garage üzerinde ise `int` tipinde `garageId` ve `description` olacak. `garageId` ögesi `id` kolonuna map etmeli.

- Garage üzerinde `getCars` bir `List` döndürmeli.
- Aradaki bağlantı `Car` tarafında tanımlanmalı, ve `garage_id` foreign key'i olmalı.
- Daha önceden yazdığımız `CarBean`'inin içine `getCars` → `(int id)` adlı bir metot ekleyeceğiz. Bu metot gönderilen `garageId`'ye göre `find` ile o garajı bulup, `getCars` ile listesini alıp, client'a döndürecek. Gerekli test verileri `import.sql` içinde yaratıldı. Bu veriler Hibernate tarafından otomatik olarak yüklenecektir.

- Bunları yaptıktan sonra `LazyInitializationException` alacaksınız. Neden? Bunu konuşacağız.
- Çözümü listeyi döndürmeden önce liste üzerinde `toArray` çağrısını yapmaktır.

Car Import

```
import javax.persistence.Entity;  
import javax.persistence.Column;  
import javax.persistence.Id;  
import javax.persistence.ManyToOne;  
import javax.persistence.JoinColumn;
```

Garage Import

```
import java.util.List;  
import javax.persistence.Entity;  
import javax.persistence.Column;  
import javax.persistence.Id;  
import javax.persistence.OneToMany;  
import java.io.Serializable;
```

Interface Import

```
import javax.ejb.Remote;  
import java.util.List;
```

Bean Import

```
import javax.ejb.Stateless;
import javax.ejb.Stateful;
import java.io.Serializable;
import javax.ejb.TransactionAttribute;
import javax.ejb.TransactionAttributeType;
import javax.persistence.PersistenceContext;
import javax.persistence.EntityManager;
import org.bilgidata.kitapdemo.pojo.*;
import java.util.ArrayList;
import java.util.List;
```