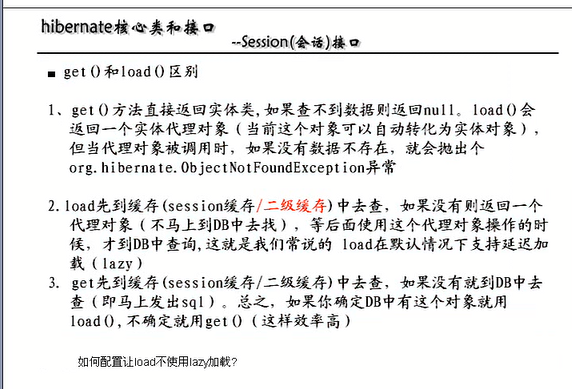
1.

# 1.我们先做一个测试

## 1.1测试load

**private** **static** **void** testgetAndload(){

Session session = MySessionFactory.*getSessionFactory*().openSession();

//这里强调 100 是不存在

Employee employee = (Employee)session.load(Employee.**class**,100);

System.*out*.println(employee);

}

No row with the given identifier exists: [com.hlj.domain.Employee#100]

## 1.2测试get

**private** **static** **void** testgetAndload(){

Session session = MySessionFactory.*getSessionFactory*().openSession();

//这里强调 100 是不存在

Employee employee = (Employee)session.load(Employee.**class**,100);

Employee employee2 = (Employee)session.get(Employee.**class**,1000);

System.*out*.println(employee2);

}

这样不会报错，第一个employee 在使用的时候才会报错，不使用 是不会报错的

输出 null’

# 3. 懒加载对于load中的现象就是，

等到用的时候才会去执行他默认是true

<class name="Employee" lazy="false" table="employee">

但是如果我们设置为 false 时候 上面的 1.2测试get就会报错，因为只要有了load就会直接进去查询，报错

# 4.我们测试load的 缓存机制，减少了对数据库的重复调用

**private** **static** **void** testLoad(){

Session session = MySessionFactory.*getSessionFactory*().openSession();

//下面这个执行之后会 将这个缓存放到session 缓存/二级缓存中

Employee employee1 = (Employee)session.load(Employee.**class**,5);

System.*out*.println(employee1);

//下面这个不会执行db 在二级缓存中查找到的，下面这个缓存之后 如果长时间使用只会，会放到一级缓存中去

Employee employee2 = (Employee)session.load(Employee.**class**,5);

System.*out*.println(employee2);

Employee employee3 = (Employee)session.get(Employee.**class**,5);

System.*out*.println(employee3);

}

上面的东西 只是执行一个sql语句