# 实体类操作

## 准备

**public** **class** JpaTest {

**private** EntityManagerFactory entityManagerFactory;//会话工厂

**private** EntityManager entityManager;//会话

**private** EntityTransaction entityTransaction;//事务

@Before

**public** **void** Init() {

entityManagerFactory = Persistence.*createEntityManagerFactory*("jpa\_test1");

entityManager = entityManagerFactory.createEntityManager();

entityTransaction = entityManager.getTransaction();

entityTransaction.begin();

}

@After

**public** **void** destory() {

entityTransaction.commit();

entityManager.close();

entityManagerFactory.close();

}

}

## Find和getReference（查）

### Find

find (Class<T> entityClass,Object primaryKey)：返回指定的 OID 对应的实体类对象，如果这个实体存在于当前的持久化环境，则返回一个被缓存的对象；否则会创建一个新的 Entity, 并加载数据库中相关信息；若 OID 不存在于数据库中，则返回一个 null。第一个参数为被查询的实体类类型，第二个参数为待查找实体的主键值。

//类似于 hibernate 中 Session 的 get 方法.

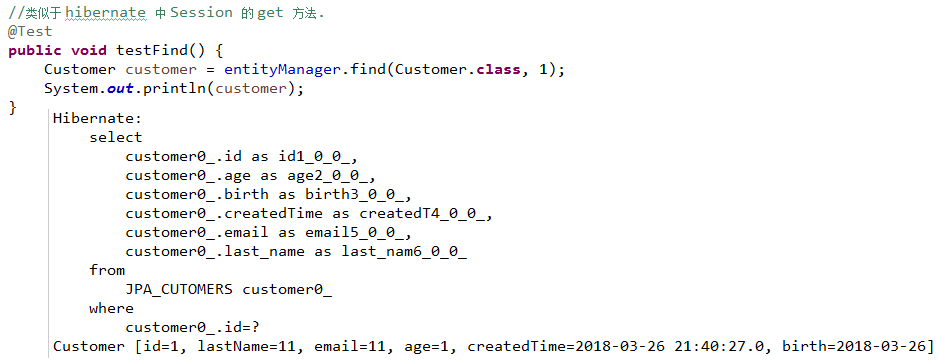
@Test

**public** **void** testFind() {

Customer customer = entityManager.find(Customer.**class**, 1);

System.***out***.println(customer);

}



### getReference

getReference (Class<T> entityClass,Object primaryKey)：与find()方法类似，不同的是：如果缓存中不存在指定的 Entity, EntityManager 会创建一个 Entity 类的代理，但是不会立即加载数据库中的信息，只有第一次真正使用此 Entity 的属性才加载，所以如果此 OID 在数据库不存在，getReference() 不会返回 null 值, 而是抛出EntityNotFoundException

//类似于 hibernate 中 Session 的 load 方法(懒加载)

@Test

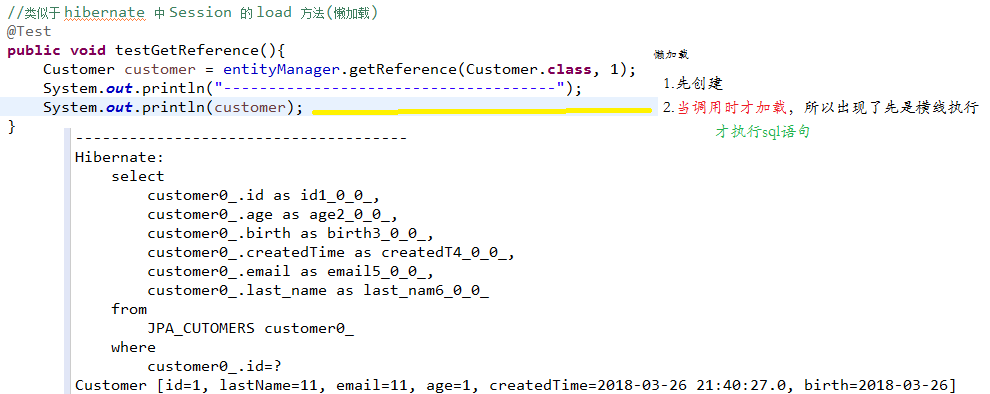
**public** **void** testGetReference(){

Customer customer = entityManager.getReference(Customer.**class**, 1);

System.***out***.println("-------------------------------------");

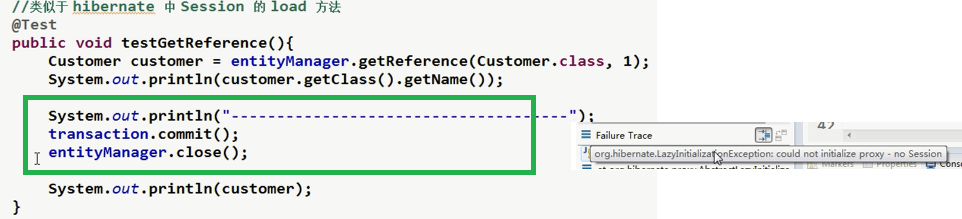
System.***out***.println(customer);

}



懒加载异常

这里未调用实例，就提前关闭会话

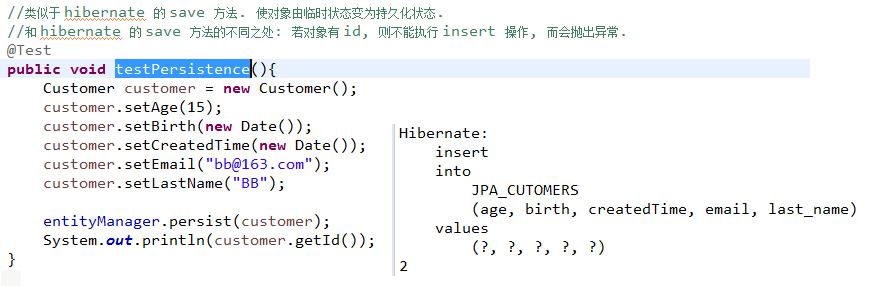


## persist（增）

persist (Object entity)：用于将新创建的 Entity 纳入到 EntityManager 的管理。该方法执行后，传入 persist() 方法的 Entity 对象转换成持久化状态。

1. 如果传入 persist() 方法的 Entity 对象已经处于持久化状态，则 persist() 方法什么都不做。

2.如果对删除状态的 Entity 进行 persist() 操作，会转换为持久化状态。



//类似于 hibernate 的 save 方法. 使对象由临时状态变为持久化状态.

@Test

**public** **void** testPersistence(){

Customer customer = **new** Customer();

customer.setAge(15);

customer.setBirth(**new** Date());

customer.setCreatedTime(**new** Date());

customer.setEmail("bb@163.com");

customer.setLastName("BB");

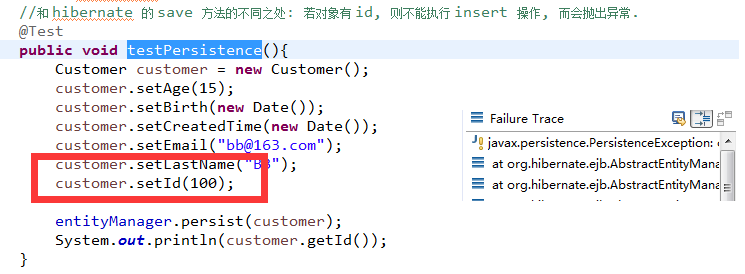
entityManager.persist(customer);

System.***out***.println(customer.getId());

}

3.如果对游离状态的实体执行 persist() 操作，可能会在 persist() 方法抛出 EntityExistException(也有可能是在flush或事务提交后抛出)。

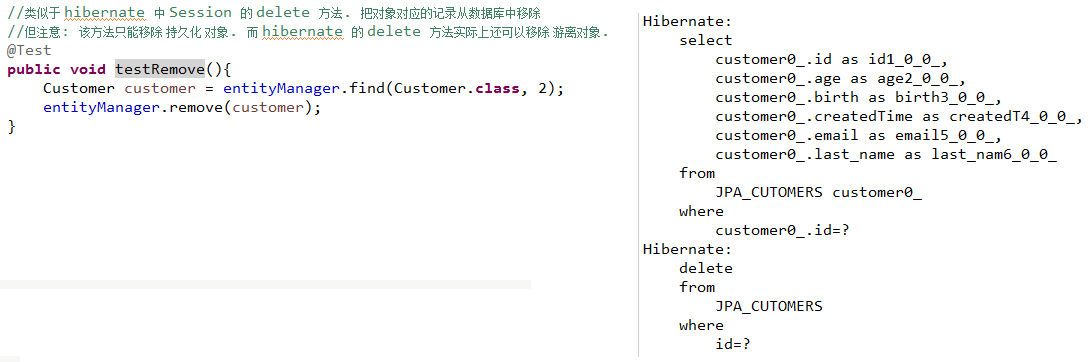
//和 hibernate 的 save 方法的不同之处: 若对象有 id, 则不能执行 insert 操作, 而会抛出异常.



## Remove（移）

remove (Object entity)：删除实例。如果实例是被管理的，即与数据库实体记录关联，则同时会删除关联的数据库记录。

1. 该方法只能移除 持久化 对象.



//类似于 hibernate 中 Session 的 delete 方法. 把对象对应的记录从数据库中移除

//但注意: 该方法只能移除 持久化 对象. 而 hibernate 的 delete 方法实际上还可以移除 游离对象.

@Test

**public** **void** testRemove(){

Customer customer = entityManager.find(Customer.**class**, 2);

entityManager.remove(customer);

}

2.而 hibernate 的 delete 方法实际上还可以移除 游离对象.

