上面我们说了单向关联关系，那么双向关联关系应该怎么配置呢？我们保持 UserInfo 不变，在 User 实体对象里面添加一段代码即可。

```
@OneToOne(mappedBy = "user")
private UserInfo userInfo;
```

完整的 User 实体对象就会变成如下模样。

```
@Entity
@Data
@Builder
@AllArgsConstructor
@NoArgsConstructor
public class User {
    @Id
    @GeneratedValue(strategy= GenerationType.AUTO)
    private Long id;
    private String name;
    private String email;
    @OneToOne(mappedBy = "user")
    private UserInfo userInfo;// 变化之处
    private String sex;
    private String address;
}
```

我们运行任何一个测试用例，都会看到运行结果是一样的，还是上面三条 SQL。那么我们再查看一下 @OneToOne 源码，看看其支持的配置都有哪些。

7.1.1 @OneToOne 的源码解读

下面我列举了 @OneToOne 的源码，并加以解读。通过这些你可以了解 @OneToOne 的用法。

```
public @interface OneToOne {
    // 表示关系目标实体，默认该注解标识的返回值的类型的类
    Class targetEntity() default void.class;
    // cascade 级联操作策略，就是我们常说的级联操作
    CascadeType[] cascade() default {};
    // 数据获取方式 EAGER( 立即加载 )/LAZY( 延迟加载 )
    FetchType fetch() default EAGER;
    // 是否允许为空，默认是可选的，也就表示可以为空
    boolean optional() default true;
    // 关联关系被谁维护的一方对象里面的属性名字。双向关联的时候必填
    String mappedBy() default "";
    // 当被标识的字段发生删除或者置空操作之后，是否同步到关联关系的一方，即进行级联删除操作
    // 默认 false，注意与 CascadeType.REMOVE 级联删除的区别
    boolean orphanRemoval() default false;
}
```