河北师范大学软件学院
Software College of Hebei Normal University

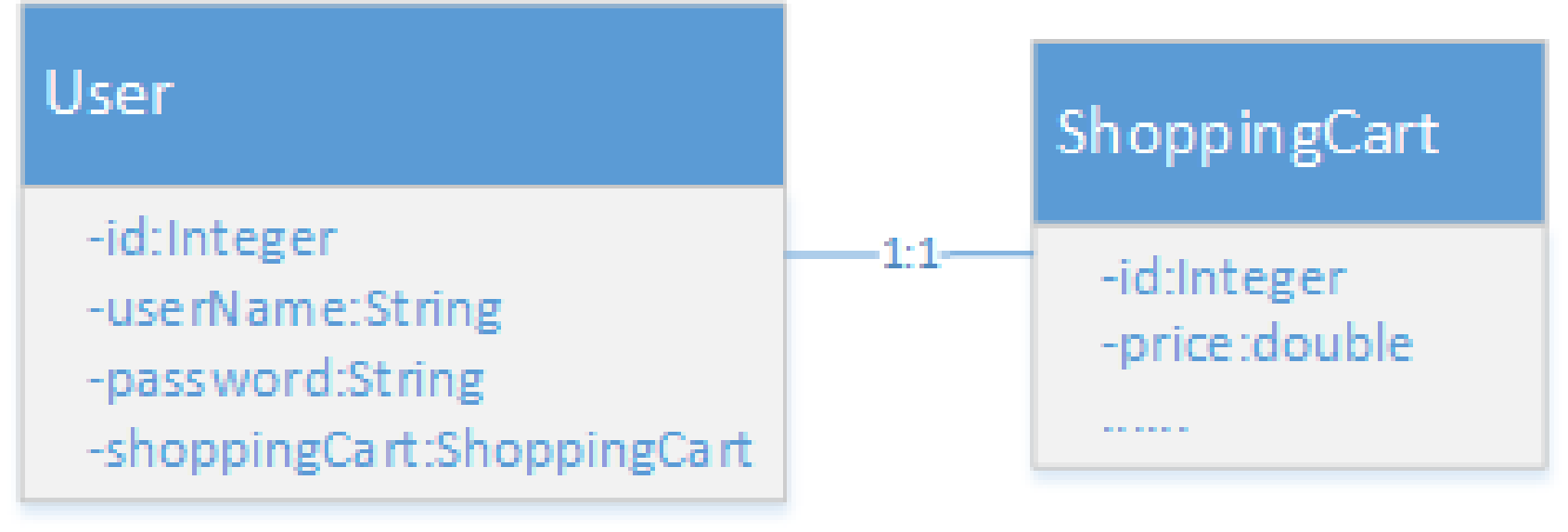
MyBatis

第三讲
MyBatis关联映射

目录

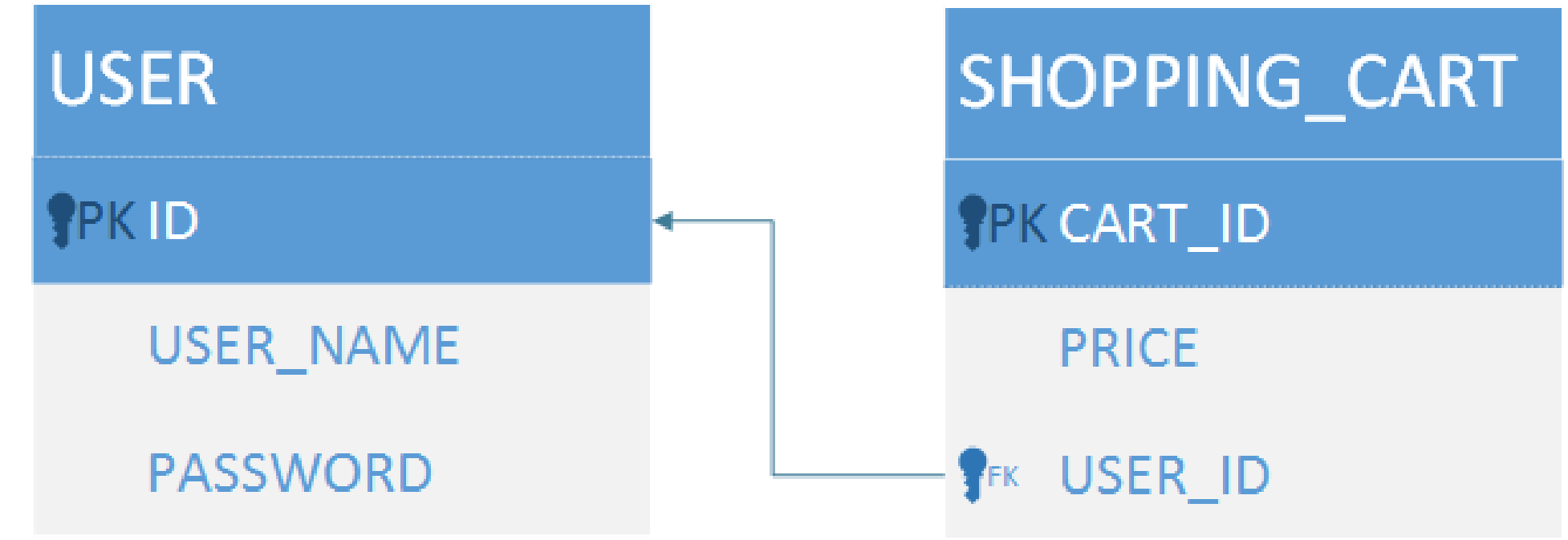
一对一映射

- 在某网络购物系统中，一个用户只能拥有一个购物车，用户与购物车的关系可以设计为一对一关系
- 域模型



一对一映射

- 数据库表结构（唯一外键关联）



一对一关联映射

- 创建两个实体类和映射接口

```
public class User {
    private Integer id;
    private String userName;
    private String password;
    private ShoppingCart
            shoppingCart;
    ……
}
```

```
public class ShoppingCart {

    private String id;
    private double price;
    ……
}
```

一对一关联映射

- 实现根据用户id查询出所有用户信息，包括该用户的购物车信息

```
public interface UserMapper {
    public User findUserAndShoppingCartById(Integer id);
}
```

一对一关联映射

- MyBatis中处理一对一关联关系的方法有四种
  - 使用自动映射处理一对一关系
  - 使用resultMap配置一对一映射
  - 使用association元素配置一对一映射
  - association元素的嵌套查询

一对一关联映射

- 方式一：使用自动映射处理一对一关系，也就是通过别名自动将值匹配到对应的字段上

```
<select id="findUserAndShoopingCartById"
        resultType="com.mybatis.entity.User">
    select u.id, u.user_name userName, u.password,
        s.cart_id "shoppingCart.id",
        s.price "shoppingCart.price"
    from User u
    left join shopping_cart s on u.id = s.user_id
    where u.id = #{id}
</select>
```

一对一关联映射

- 复杂的属性映射时，可以多层嵌套，比如将shopping_cart表中的cart_id字段映射到shoppingCart.id属性上

一对一关联映射

- 方式二：在XML映射文件中配置结果映射

```
<resultMap type="com.mybatis.entity.User"
           id="userShoppingCartMap">
    <id property="id" column="id"/>
    <result property="userName" column="user_name"/>
    <result property="password" column="password"/>
    <result property="shoppingCart.id" column="cart_id"/>
    <result property="shoppingCart.price"
            column="price"/>
</resultMap>
```

一对一关联映射

```
<select id="findUserAndShoopingCartById"
        resultMap="userShoppingCartMap">
    select u.id, u.user_name, u.password,
        s.cart_id, s.price
    from user u
    left join shopping_cart s on u.id = s.user_id
    where u.id = #{id}
</select>
```

- 使用这种方式同自动映射方式相似之处为，ShoppingCart中的属性配置部分使用了”shoppingCart.”前缀

一对一关联映射

- 方式三：association元素用于和一个复杂的类型进行关联

```
<resultMap type="com.mybatis.entity.User"
           id="userShoppingCartMap">
    <id property="id" column="id"/>
    <result property="userName" column="user_name"/>
    <result property="password" column="password"/>
    <association property="shoppingCart" >
        <id property="id" column="cart_id"/>
        <result property="price" column="price"/>
    </association>
</resultMap>
```

一对一关联映射

- association元素包含以下属性
  - property：对应实体类中的属性名，必填项
  - javaType：属性对应的java类型，可选项
  - resultMap：可以直接使用现有的resultMap，而不需要在这里配置，可选项

一对一关联映射

- 方式四：association元素的嵌套查询

```
<resultMap type="com.mybatis.entity.User"
           id="userShoppingCartMap">
    <id property="id" column="id"/>
    <result property="userName" column="user_name"/>
    <result property="password" column="password"/>
    <association property="shoppingCart"
        column="{user_id=id}" select="com.mybatis.mapper.
        ShoppingCartMapper.findShoppingCartById"/>
</resultMap>
```

一对一关联映射

- association元素的嵌套查询常用的属性如下
  - select：另一个查询映射的statement id，MyBatis会额外执行这个查询获取嵌套对象
  - column：列名，将主查询中列的结果作为嵌套查询的参数，如column=“{prop1=col1,prop2=col2}”，其中prop1和prop2将作为嵌套查询的参数
  - fetchType：数据加载方式，可选值为lazy和eager，分别为延迟加载和积极加载，会覆盖全局的lazyLoadingEnable配置

一对一关联映射

```
<select id="findUserAndShoopingCartById"
        resultMap="userShoppingCartMap">
    select u.id, u.user_name, u.password
    from user u
    where u.id = #{id}
</select>
```

- 注意：此处查询语句只包含user表，购物车信息需要在

  ShoppingCartMapper.xml中配置

  findShoppingCartById方法进行查询

一对一关联映射

```
<resultMap type="com.mybatis.entity.ShoppingCart"
           id="shoppingCartMap">
    <id property="id" column="cart_id"/>
    <result property="price" column="price"/>
</resultMap>
<select id="findShoppingCartById"
        resultMap="shoppingCartMap"
    select *
    from shopping_cart
    where user_id = #{user_id}
</select>
```

一对一关联映射

- 嵌套查询会多执行SQL，当查询N条数据时，就会出现N+1次查询问题
- 为了解决N+1次查询问题，可以设置延迟加载策略，将association元素的fetchType属性设置为lazy

一对一关联映射

- 在MyBatis主配置文件中有两个延时加载相关settings设置

设置项	描述	有效值	默认值
lazyLoadingEnabled	延迟加载的全局开关。当开启时，所有关联对象都会延迟加载。 可通过设置fetchType属性来覆盖该项的开关状态	true | false	false
aggressiveLazyLoading	当开启时，任何方法的调用都会加载该对象的所有属性。否则，每个属性会按需加载	true | false	false V3.4.5以上

一对一关联映射

```
<settings>
    <setting name="lazyLoadingEnabled" value="true"/>
    <setting name="aggressiveLazyLoading" value="false"/>
</settings>
```

- 设置为延迟加载的对象，默认情况下，当调用该对象的equals、hashCode、toString、clone方法时，就会加载该对象的全部数据

一对一关联映射

- 对比四种方式
  - 前面的三种方式都属于“关联的嵌套结果映射“,即通过一次SQL查询根据表或指定的属性映射到不同的对象中
  - 最后一种方式属于“关联的嵌套查询”，利用简单的SQL语句，通过多次查询得到想要的结果，可实现延迟加载效果

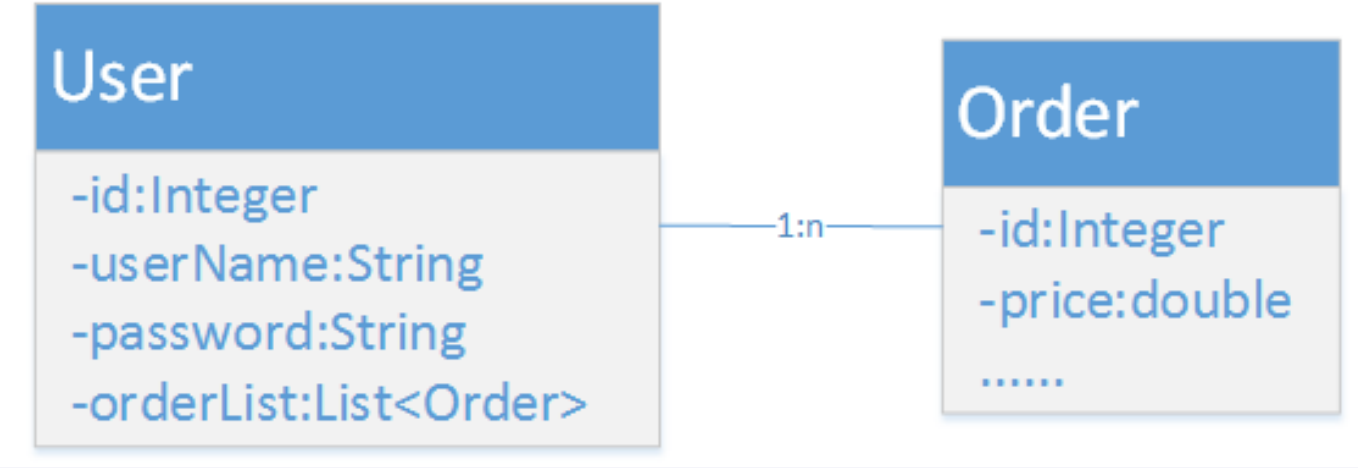
练习

- 1、使用主键关联方式设计数据库完成一对一关联映射
- 2、实现给用户创建购物车操作

目录

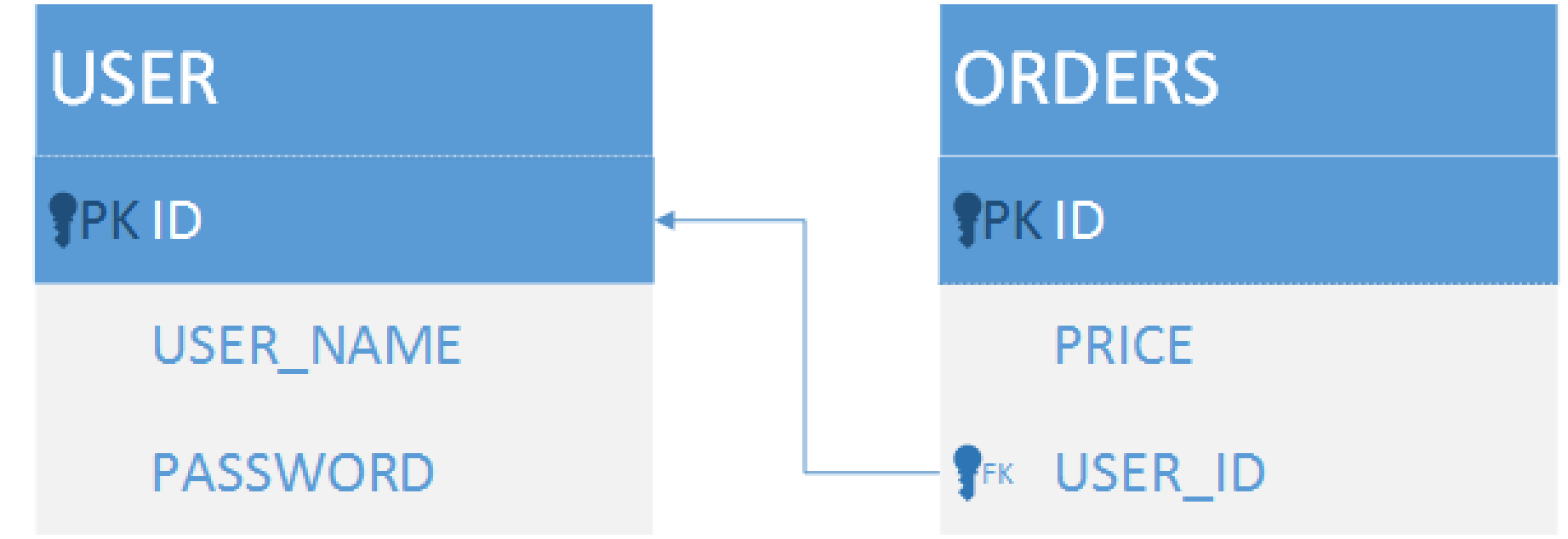
一对多关联映射

- 一对多关联映射有两种方式，都用到了collection元素
  - collection集合的嵌套结果映射
  - collection集合的嵌套查询
- 以购物网站中用户和订单之间的一对多关系为例



一对多关联映射

- 数据库表结构



一对多关联映射

- 创建两个实体类和映射接口

```
public class User {
    private Integer id;
    private String userName;
    private String password;
    private List<Order>
            orderList;
    ……
}
```

```
public class Order {

    private String id;
    private double price;
    ……
}
```

一对多关联映射

- 实现根据用户id查询出所有用户信息，包括该用户的所有订单信息

```
public interface UserMapper {
    public User findUserAndOrderListById(Integer id);
}
```

一对多关联映射

- 方式一：与association类似，集合的嵌套结果映射就是指通过一次SQL查询得到所有的结果

```
<resultMap type="com.mybatis.entity.User"
        id="userMap">
        <id property="id" column="id"/>
    <result property="userName" column="user_name"/>
    <result property="password" column="password"/>
</resultMap>
```

- 注：由于此映射会频繁用到，可以将其单独配置，后面可以直接通过id引用

一对多关联映射

```
<resultMap type="com.mybatis.entity.User"
    id="userAndOrderListMap" extends="userMap">
    <collection property="orderList"
        ofType="com.mybatis.entity.Order">
        <id property="id" column="order_id"/>
        <result property="price" column="price"/>
    </collection>
</resultMap>
```

- resultMap元素中的extends属性可以实现结果映射的继承
- collection的ofType属性指定集合中元素的类型，必选项

一对多关联映射

- 查询映射配置如下

```
<select id="findUserAndOrderListById"
    resultMap="userAndOrderListMap">
    select u.id, u.user_name, u.password,
    o.order_id, o.price
    from user u
    left join orders o on u.id = o.user_id
    where u.id = #{id}
</select>
```

一对多关联映射

- 方式二：集合的嵌套查询同样会执行额外的SQL查询

```
<resultMap type="com.mybatis.entity.User"
    id="userAndOrderListMap" extends="userMap">
    <collection property="orderList" column="{uid=id}"
        ofType="com.mybatis.entity.Order"
select="com.mybatis.mapper.OrderMapper.findOrdersByUserId">
    </collection>
</resultMap>
<select id="findUserAndOrderListById"
    resultMap="userAndOrderListMap">
    select * from user where id = #{id}
</select>
```

一对多关联映射

- OrderMapper.xml

```
<resultMap type="com.mybatis.entity.Order"
    id="orderMap">
    <id property="id" column="order_id"/>
    <result property="price" column="price"/>
</resultMap>
<select id="findOrdersByUserId"
    resultMap="orderMap">
    select * from orders where user_id = #{uid}
</select>
```

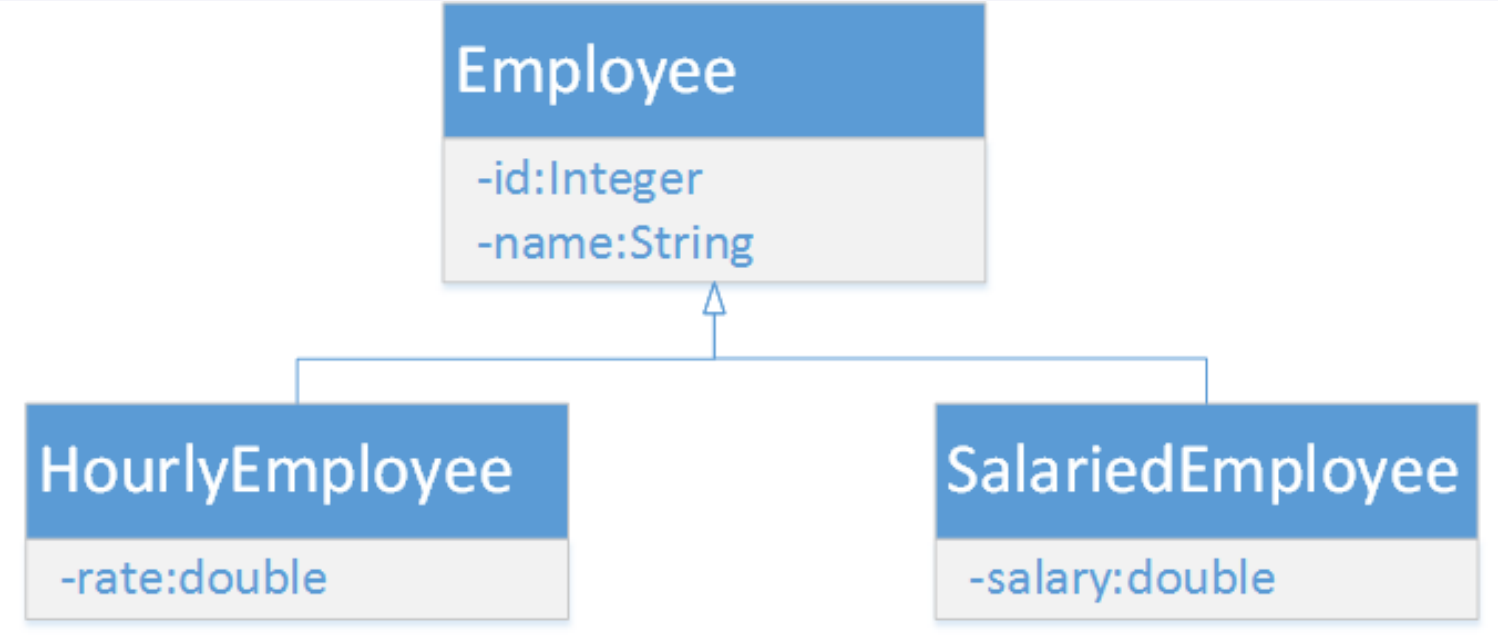
一对多关联映射

- 对比两种方式
  - 第一种方式属于“关联的嵌套结果映射“,即通过一次SQL查询根据表或指定的属性映射到不同的对象中
  - 第二种方式属于“关联的嵌套查询”，利用简单的SQL语句，通过多次查询得到想要的结果，也可以实现延迟加载效果

目录

鉴别器映射

- 有时一个单独的数据库查询会返回很多种不同数据类型的结果集。discriminator鉴别器元素可以用来处理这种情况
- 以Hibernate中继承映射为例



鉴别器映射

- 数据库表结构

鉴别器映射

- 创建实体类（省略）、映射接口和映射文件

```
public interface EmployeeMapper {
    public Employee findEmployeeById(Integer id);
}
```

```
<resultMap type="com.mybatis.entity.Employee"
        id="EmployeeMap">
    <id property="id" column="id"/>
    <result property="name" column="name"/>
</resultMap>
```

鉴别器映射

- 使用resultMap的extends属性来映射两个子类的属性

```
<resultMap type="com.mybatis.entity.HourlyEmployee"
    id="HourlyMap" extends="EmployeeMap">
    <result property="rate" column="rate"/>
</resultMap>
<resultMap type="com.mybatis.entity.SalariedEmployee"
    id="SalaryMap" extends="EmployeeMap">
    <result property="salary" column="salary"/>
</resultMap>
```

鉴别器映射

- 使用discriminator元素映射employee_type字段

```
<resultMap type="com.mybatis.entity.Employee"
        id="selectMap">
    <discriminator column="employee_type"
        javaType="String">
        <case value="HE" resultMap="HourlyMap"/>
        <case value="SE" resultMap="SalaryMap"/>
    </discriminator>
</resultMap>
```

鉴别器映射

- discriminator元素常用的两个属性如下：
  - column：该属性用于设置需要进行鉴别比较值的列
  - javaType：该属性用于指定列的类型

鉴别器映射

- discriminator元素可以有1个或者多个case子元素，case元素包含三个属性
  - value：该值用来匹配column指定字段的值
  - resultMap：当value值和column的值匹配时的结果映射，优先级高于resultType
  - resultType：当value值和column的值匹配时的结果类型

练习

- 完成从订单到用户的多对一映射
- 完成订单和商品之间的多对多映射

本章小结

- 一对一关联映射
  - association元素
- 一对多关联映射
  - resultMap的继承、collection元素
- 鉴别器映射
  - discriminator元素

THANK YOU