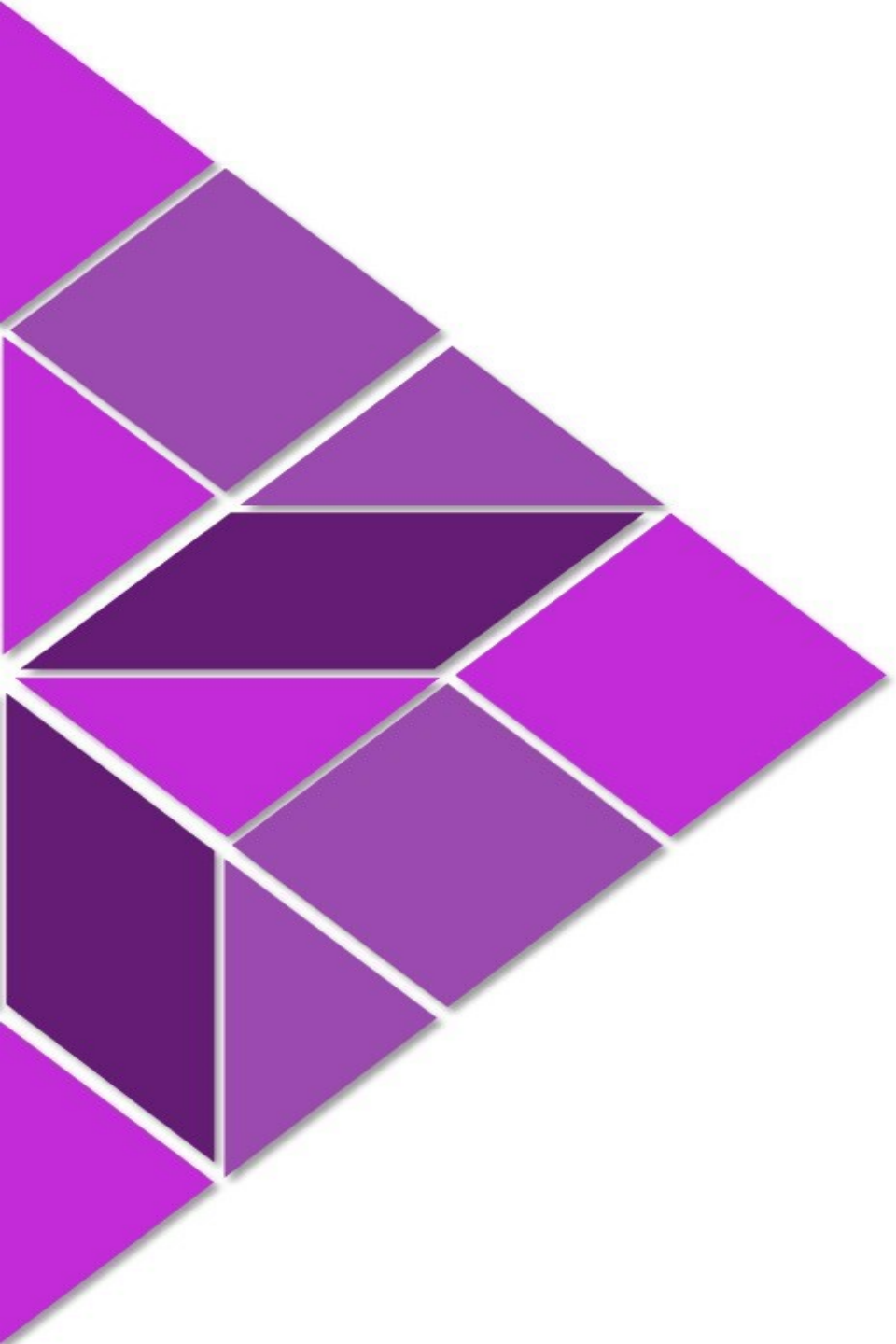


Hibernate Inheritance

Presented by Hosein Zare
Twitter: @zare88r

Inheritance

- با اینکه جداول رابطه ای در پایگاه داده ها از ارث بری پشتیبانی نمیکنند ولی hibernate استراتژی هایی برای استفاده از این قابلیت در نظر گرفته است:

Mapped Super Class –

Single Table –

Joined Table –

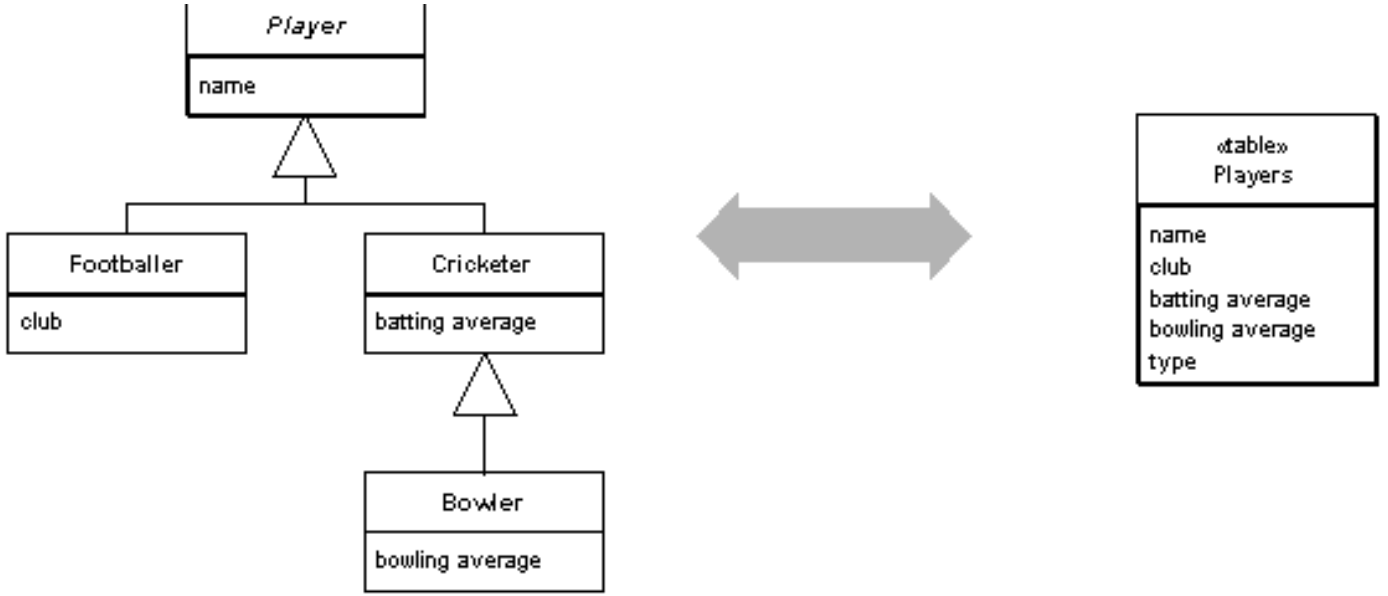
Table per class –

Mapped Super Class

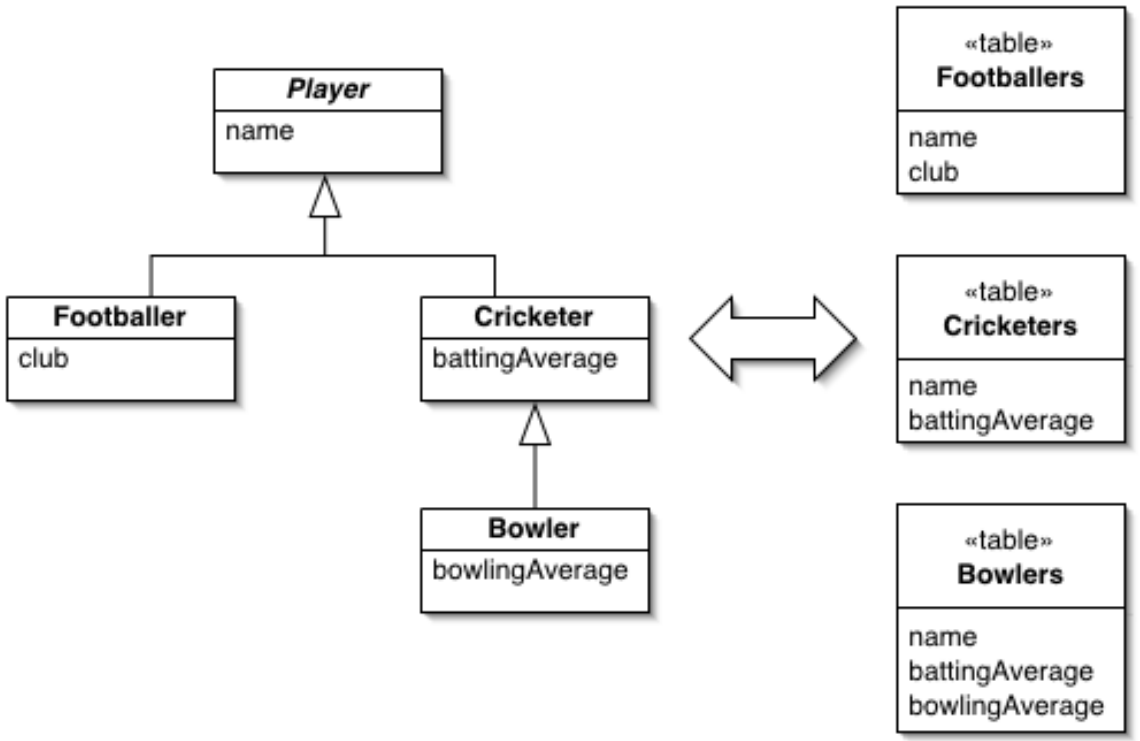
```java
@MappedSuperclass
public class Account {
    @Id
    private Long id;
    private String owner;
    private BigDecimal balance;
    private BigDecimal interestRate;
    //GETTER AND SETTER
}

@Entity
public class DebitAccount extends Account {
    private BigDecimal overdraftFee;
    // ...
}

@Entity
public class CreditAccount extends Account {
    private BigDecimal creditLimit;
    // ...
}
```

- در این حالت اینگونه فرض میشود که تمامی ستون های نگاشت شده در entity والد در تمامی جداول entity فرزند کپی شده اند

Single Table Model



Multi Table Model



Single Table Inheritance

```java
@Entity
@Inheritance(strategy = InheritanceType.SINGLE_TABLE)
public static class Account {
    @Id
    private Long id;
}
```

- در این مدل به ستونی اضافی به عنوان جدا ساز (discriminator) نیازداریم
- اگر ستونی معرفی نشود پیش فرض از اسم ستونی به نام DTYPE استفاده میشود و برای مقدار نام entity را ذخیره میکند.

Single Table Inheritance

```
DebitAccount debitAccount = new DebitAccount();
debitAccount.setId( 1L );
debitAccount.setOwner( "John Doe" );
debitAccount.setBalance( BigDecimal.valueOf( 100 ) );
debitAccount.setInterestRate( BigDecimal.valueOf( 1.5d ) );
debitAccount.setOverdraftFee( BigDecimal.valueOf( 25 ) );

CreditAccount creditAccount = new CreditAccount();
creditAccount.setId( 2L );
creditAccount.setOwner( "John Doe" );
creditAccount.setBalance( BigDecimal.valueOf( 1000 ) );
creditAccount.setInterestRate( BigDecimal.valueOf( 1.9d ) );
creditAccount.setCreditLimit( BigDecimal.valueOf( 5000 ) );

session.save( debitAccount );
session.save ( creditAccount );
```

```
INSERT INTO Account (balance, interestRate, owner, overdraftFee, DTYPE, id)
VALUES (100, 1.5, 'John Doe', 25, 'DebitAccount', 1)

INSERT INTO Account (balance, interestRate, owner, creditLimit, DTYPE, id)
VALUES (1000, 1.9, 'John Doe', 5000, 'CreditAccount', 2)
```

Discriminator

- یک discriminator column ستونی است که در یک سطر مشخص میکند از چه نوع کلاس فرزندی برای ایجاد آن باید استفاده شود. در کلاس های فرزند از @DiscriminatorValue برای مشخص کردن مقدار استفاده میشود
  - @DiscriminatorColumn(“TYPE_COLUMN”)
- همچنین اگر ستونی در نظر گرفته نشده باشد میتوان از یک فرمول sql برای تشخیص آن استفاده کرد
  - ```
    @DiscriminatorFormula(
            "case when debitKey is not null " +
                    "then 'Debit' " +
                    "else ( case when creditKey is not null " +
                    "   then 'Credit' else 'Unknown' end ) " +
                    "end ")
    ```
    ```

Joined table

- هر entity فرزند خود به یک جدول مستقل نگاشت پیدا کرده اند.
- در این نمونه وجود discriminator column الزامی نیست.
- کلید های اصلی در جداول فرزند کلید خارجی به جدول والد هستند
- در صورتی که نام کلید اصلی در جدول فرزند متفاوت از کلید والد بود از PrimaryKeyJoinColumn برای توضیح نام آن استفاده میشود

Joined table

```
List<Account> accounts = entityManager
        .createQuery("select a from Account a").getResultList();
```

```
SELECT jointablet0_.id AS id1_0_ ,
       jointablet0_.balance AS balance2_0_ ,
       jointablet0_.interestRate AS interest3_0_ ,
       jointablet0_.owner AS owner4_0_ ,
       jointablet0_1_.overdraftFee AS overdraf1_2_ ,
       jointablet0_2_.creditLimit AS creditLi1_1_ ,
       CASE WHEN jointablet0_1_.id IS NOT NULL THEN 1
            WHEN jointablet0_2_.id IS NOT NULL THEN 2
            WHEN jointablet0_.id IS NOT NULL THEN 0
       END AS clazz_
FROM   Account jointablet0_
       LEFT OUTER JOIN DebitAccount jointablet0_1_ ON jointablet0_.id = jointablet0_1_.id
       LEFT OUTER JOIN CreditAccount jointablet0_2_ ON jointablet0_.id = jointablet0_2_.id
```

Table Per Class

- هر entity فرزند خود به یک جدول مستقل نگاشت پیدا کرده اند. با این تفاوت که هر کلاس تمامی ستون ها را در خود کپی کرده است.
- زمانی که نیاز به واکشی سطر خاصی داشته باشید hibernate از طریق union subclass ها را بدست می آورد

Table Per Class

```
CREATE TABLE Account (
    id BIGINT NOT NULL ,
    balance NUMERIC(19, 2) ,
    interestRate NUMERIC(19, 2) ,
    owner VARCHAR(255) ,
    PRIMARY KEY ( id )
)

CREATE TABLE CreditAccount (
    id BIGINT NOT NULL ,
    balance NUMERIC(19, 2) ,
    interestRate NUMERIC(19, 2) ,
    owner VARCHAR(255) ,
    creditLimit NUMERIC(19, 2) ,
    PRIMARY KEY ( id )
)

CREATE TABLE DebitAccount (
    id BIGINT NOT NULL ,
    balance NUMERIC(19, 2) ,
    interestRate NUMERIC(19, 2) ,
    owner VARCHAR(255) ,
    overdraftFee NUMERIC(19, 2) ,
    PRIMARY KEY ( id )
)
```

```
SELECT tablepercl0_.id AS id1_0_ ,
       tablepercl0_.balance AS balance2_0_ ,
       tablepercl0_.interestRate AS interest3_0_ ,
       tablepercl0_.owner AS owner4_0_ ,
       tablepercl0_.overdraftFee AS overdraf1_2_ ,
       tablepercl0_.creditLimit AS creditLi1_1_ ,
       tablepercl0_.clazz_ AS clazz_
FROM (
    SELECT    id ,
              balance , interestRate , owner ,
              CAST(NULL AS INT) AS overdraftFee ,
              CAST(NULL AS INT) AS creditLimit ,
              0 AS clazz_
    FROM      Account
    UNION ALL
    SELECT    id , balance , interestRate ,
              owner , overdraftFee ,
              CAST(NULL AS INT) AS creditLimit ,
              1 AS clazz_
    FROM      DebitAccount
    UNION ALL
    SELECT    id , balance , interestRate ,
              owner ,
              CAST(NULL AS INT) AS overdraftFee ,
              creditLimit ,
              2 AS clazz_
    FROM      CreditAccount
) tablepercl0_
```