Hibernate – V

Inheritance Mapping

Table of Contents

Introduction

- In hibernate there is one more important advantage over JDBC, **i.e. inheritance** mapping.
- Suppose if we have base and derived classes, and we want to store base and derived class object then ,using inheritance mapping we can save derived class object, base class object will also be stored into the database.

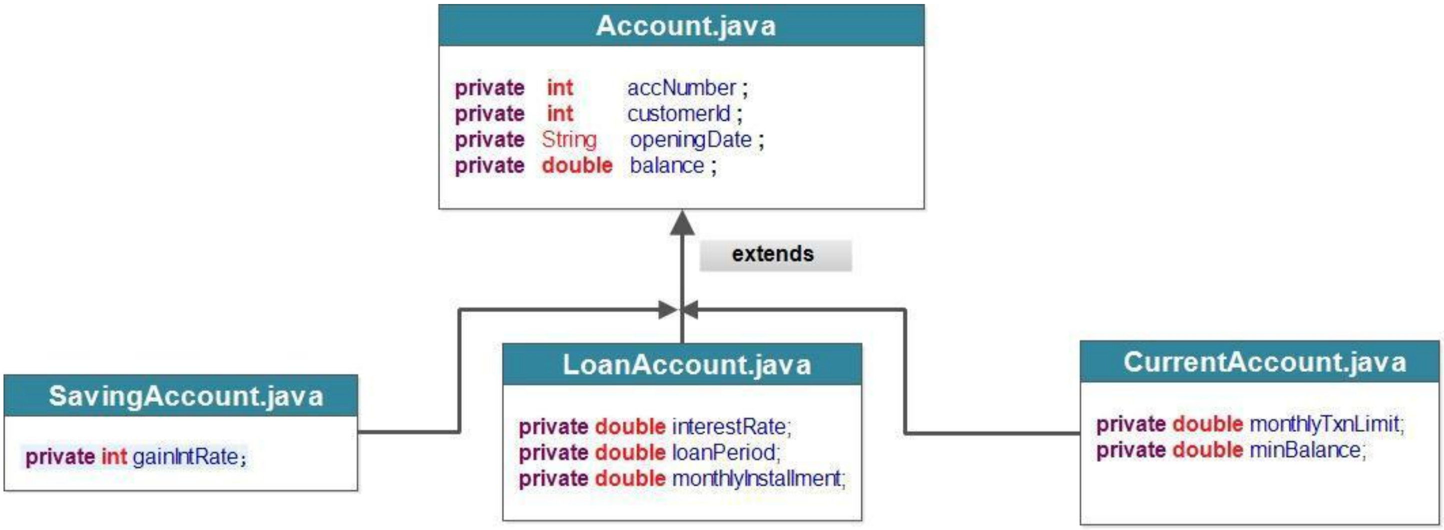
Hibernate supports **3 types of Inheritance** Mappings:

- Table **per class hierarchy**
- Table **per sub-class**
- Table **per concrete class**

It is possible to define **subclass**, **joined-subclass** and **joined-subclass** mappings in separate mapping documents

Here we take **example so,** how inheritance mapping can be done**:**

Base class is **Account** and derived class is **SavingAccount**, **CurrentAccount** and **LoanAccount.**



```
private   int      accNumber ;
private   int      customerId ;
private   String   openingDate ;
private   double   balance ;
```

```
private int gainIntRate;
```

```
private double interestRate;
private double loanPeriod;
private double monthlyInstallment;
```

```
private double monthlyTxnLimit;
private double minBalance;
```

Table per Class Hierarchy Inheritance Mapping

- Here is the explanation and one example on hibernate table per class hierarchy, consider we have **Base** class as **Account** and **derived** classes like **CurrentAccount**, **SavingAccount** and **LoanAccount**
- If we **save** the **derived class** object like **SavingAccount** or **CurrentAccount** and **LoanAccount** then automatically **Account class object will also be saved** into the database, and in the database all the data will be **stored into a single** table only.
- To differentiate between entities object, we required one discriminator column to store the discriminator value.
- Discriminator Column must be **first statement** after <id> element

Required Files

Account.java

```
package com.kalibermind.hibernate.entity;
public class Account
{
    private int accNumber;
    private int customerId;
    private String openingDate;
    private double balance;
    public Account() { }
 public Account(int accNumber, int customerId, String openingDate, double balance)
   {super();
            this.accNumber = accNumber;
            this.customerId = customerId;
            this.openingDate = openingDate;
            this.balance = balance;
  }
}//corresponding getter and setter
```

CurrentAccount.java

```
package com.kalibermind.hibernate.entity;

public class CurrentAccount extends Account
{
       private double monthlyTxnLimit;
       private double minBalance;

 public CurrentAccount (int accNumber, int customerId, String
 openingDate, double balance, double monthlyTxnLimit, double
minBalance)
   {
        super(accNumber, customerId, openingDate,
balance);
        this.monthlyTxnLimit = monthlyTxnLimit;
        this.minBalance = minBalance;
     }
        //corresponding getter and setter
   }
```

SavingAccount.java

```
package com kalibermind.hibernate.entity;

public class SavingAccount extends Account
{
     private int gainIntRate;
     public SavingAccount(int accNumber, int customerId, String
             openingDate, double balance, int gainIntRate)
    {
        super(accNumber, customerId, openingDate, balance);
        this.gainIntRate = gainIntRate;
    }
        //corresponding getter and setter
}
```

LoanAccount.java

```
package com.kalibermind.hibernate.entity;
public class LoanAccount extends Account
{
       private double interestRate;
       private double loanPeriod;
       private double monthlyInstallment;
      public LoanAccount (int accNumber, int customerId, String openingDate,
       double balance, double interestRate, double loanPeriod, double
                                                              monthlyInstallment)
      {
             super(accNumber, customerId, openingDate, balance);
             this.interestRate = interestRate;
             this.loanPeriod = loanPeriod;
             this.monthlyInstallment = monthlyInstallment;
      }  //corresponding getter and setter
}
```

hibernate.cfg.xml

```
<hibernate-configuration>
    <session-factory>
                        <property name="connection.driver_class">com.mysql.jdbc.Driver</property> <property
                        name="connection.url">jdbc:mysql://localhost:3306/hibernate</property> <property
                        name="connection.username">root</property> <property
                        name="connection.password">admin</property>
                        <property name="connection.pool_size">10</property>
                        <property name="dialect">org.hibernate.dialect.MySQLDialect</property>
                        <property name="hbm2ddl.auto">update</property> <property
                        name="show_sql">true</property>

                        <mapping resource="resources/Account.hbm.xml"/>
            </session-factory>
</hibernate-configuration>
```

Account.hbm.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-mapping PUBLIC
"-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping>
   <class name="com.kalibermind.hibernate.entity.Account” table="ACCOUNTS">
            <id name="accNumber" column="ACC_NUMBER">
                <generator class="native"></generator>
            </id>
           <discriminator column="ACC_TYPE" type="string" length="15"/>
           <property name="customerId" column="CUSTOMER_ID"/>
           <property name="openingDate" column="OPENING_DATE"/>
           <property name="balance" column="BALANCE" />

            <subclass name="com.kalibermind.hibernate.entity.SavingAccount” discriminator-
                  value="SAVING_ACCOUNT">
             <property name="gainIntRate" column="INTEREST_RATE"/>
            </subclass>

          <subclass name="com.kalibermind.hibernate.entity.CurrentAccount” discriminator-
                value="CURRENT_ACCOUNT">
                <property name="monthlyTxnLimit" column="MONTHLY_TXN_LIMIT"/>
                <property name="minBalance" column="MINIMUM_BALANCE"/>
          </subclass>
        <subclass name="com.kalibermind.hibernate.entity.LoanAccount" discriminator-
                value="LOAN_ACCOUNT">
                 <property name="interestRate" column="INTEREST_RATE"/>
                 <property name="loanPeriod" column="LOAN_PERIOD"/>
                <property name="monthlyInstallment" column="MONTHLY_INSTALLMENT" />
         </subclass>
     </class>
</hibernate-mapping>
```

Table per Class Hierarchy Inheritance Mapping Using Annotation

```
@Entity
@Table(name="ACCOUNT")
@Inheritance(strategy=InheritanceType.SINGLE_TABLE)
@DiscriminatorColumn( name="ACCOUNT_TYPE", discriminatorType=DiscriminatorType.STRING)
@DiscriminatorValue("ACCOUNT")

public class Account
{
   //Required property
 }

@Entity

@DiscriminatorValue("SVING_ACCOUNT")

public class SavingAccount extends Account
 {
        //Required property
}

@Entity

@DiscriminatorValue("CURRENT_ACCOUNT")

public class CurrentAccount extends Account
{
        //Required property

}
```

Table per Subclass Inheritance Mapping

This is same as previous example, but some changes are there, in table per class hierarchy all the data was saved in a single table but here,

N number of classes **= N** number of tables in the database

- ✓ In the mapping file, <key> element is because, once we save the derived class object, then hibernate will first save the base class object then derived class object , so at the time of saving the derived class object hibernate will copy the primary key value of the base class into the corresponding derived class.

- ✓ If we save the **CurrentAccount** class object, then first hibernate will saves the data related to super class(**Account**) object into the super class related table in the database and then **CurrentAccount** object data in **CurrentAccount** related table in the database, so first base class data will be saved**.**

We have to change only in Account.hbm.xml file

Account.hbm.xml

```
<!DOCTYPE hibernate-mapping PUBLIC "- //Hibernate/Hibernate
Mapping DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">

<hibernate-mapping>
    <class name="com.kalibermind.hibernate.entity.Account" table="ACCOUNTS">
        <id name="accNumber" column="ACCOUNT_NUMBER">
            <generator class="native"></generator>
        </id>
        <property name="customerId" column="CUSTOMER_ID"/>
        <property name="openingDate" column="OPENING_DATE"/>
        <property name="balance" column="BALANCE" />

        <joined-subclass name="com.kalibermind.hibernate.entity.SavingAccount"table="SAVING_ACCOUNT">
           <key column="ACCOUNT_NUMBER"/>
           <property name="gainIntRate"column="INTEREST_RATE"/>
        </joined-subclass>

        <joined-subclass name="com.kalibermind.hibernate.entity.CurrentAccount"
            table="CURRENT_ACCOUNT"> <key column="ACCOUNT_NUMBER"/>
            <property name="monthlyTxnLimit" column="MONTHLY_TXN_LIMIT"/>
            <property name="minBalance" column="MINIMUM_BALANCE"/>
        </joined-subclass>

        <joined-subclass name="com.kalibermind.hibernate.entity.LoanAccount" table="LOAN_ACCOUNT">
            <key column="ACCOUNT_NUMBER"/>
            <property name="interestRate" column="INTEREST_RATE"/>
            <property name="loanPeriod" column="LOAN_PERIOD"/>
            <property name="monthlyInstallment" column="MONTHLY_INSTALLMENT" />
       </joined-subclass>
    </class>
</hibernate-mapping>
```

Note:

- Table per subclass does not require any discriminator column.
- Other object/relational mappers use a different implementation of table per subclass that requires a type discriminator column in the super class table

Table per Subclass Inheritance mapping using annotation

```
@Entity
@Table(name="ACCOUNT")
@Inheritance(strategy=InheritanceType.JOINED)
public class Account implements Serializable { ... }

@Entity
@Table(name="SAVING_ACCOUNT")
@PrimaryKeyJoinColumn(name="ACC_NUMBER")
public class SavingAccount extends Account { ... }

@Entity
@Table(name="CURRENT_ACCOUNT")
@PrimaryKeyJoinColumn(name="ACC_NUMBER")
public class CurrentAccount extends Account { ... }
```

Table per Concrete Class Inheritance Mapping

Once we save the derived class object, then derived class data and base class data will be saved in the derived class related table in the database for this type we need the tables for derived classes,

But not for the base class in the mapping file we need to **use <union-subclass>** element inside <**class**> elements

N number of derived classes = **N** number of tables in the database

Account.hbm.xml

```
!DOCTYPE hibernate-mapping PUBLIC
"-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping>
        <class name="com.kalibermind.hibernate.entity.Account" table="ACCOUNTS">
          <id name="accNumber" column="ACC_NUMBER">
              <generator class="native"></generator>
           </id>
          <property name="customerId" column="CUSTOMER_ID"/>
          <property name="openingDate" column="OPENING_DATE"/>
          <property name="balance" column="BALANCE" />
<union-subclass name="com.kalibermind.hibernate.entity.SavingAccount"table="SAVING_ACCOUNT">
         <property name="gainIntRate" column="INTEREST_RATE"/>
</union-subclass>

< union-subclass name="com.kalibermind.hibernate.entity.CurrentAccount" table="CURRENT_ACCOUNT">
        <property name="monthlyTxnLimit" column="MONTHLY_TXN_LIMIT"/>
        <property name="minBalance" column="MINIMUM_BALANCE"/>
</union-subclass>

<union-subclass name="com.kalibermind.hibernate.entity.LoanAccount" table="LOAN_ACCOUNT">
         <property name="interestRate" column="INTEREST_RATE"/>
         <property name="loanPeriod" column="LOAN_PERIOD"/>
         <property name="monthlyInstallment" column="MONTHLY_INSTALLMENT" />
</union-subclass>

 </class>
</hibernate-mapping>
```

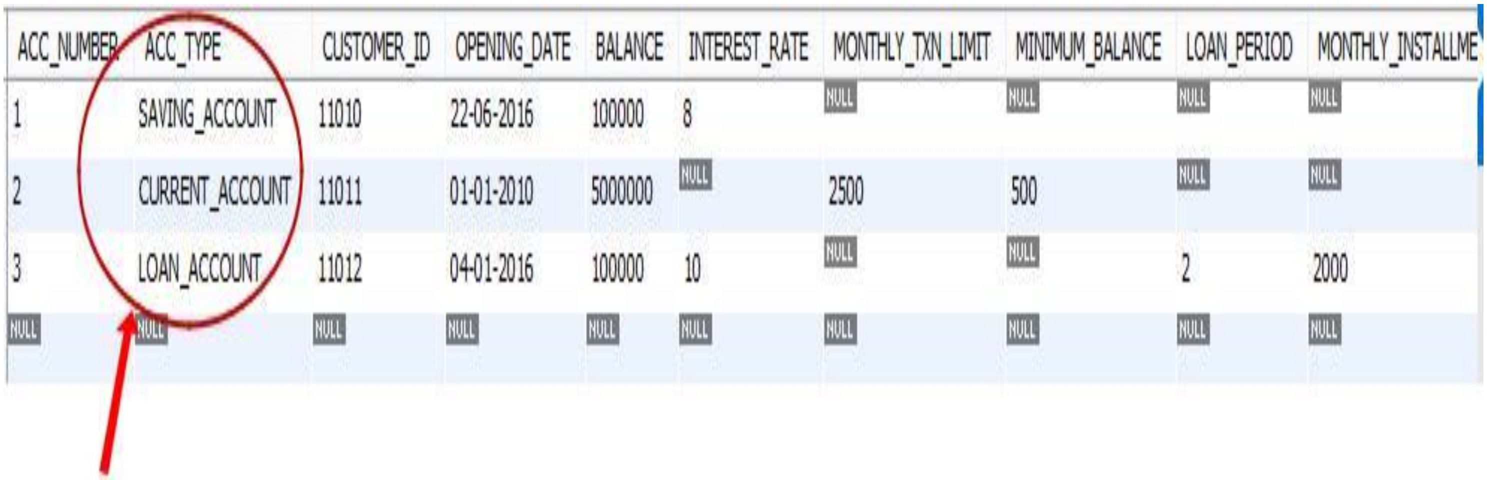
Table per Concrete Class Inheritance Mapping using Annotation

```
@Entity
@Table(name="ACCOUNT")
@Inheritance(strategy = InheritanceType.TABLE_PER_CLASS)
public class Account implements Serializable { ... }

@Entity
@Table(name="SAVING_ACCOUNT")
public class SavingAccount extends Account { ... }

@Entity
@Table(name="CURRENT_ACCOUNT")
public class CurrentAccount extends Account { ... }
```

The @**PrimaryKeyJoinColumn** and @**PrimaryKeyJoinColumns** annotations define the primary key(s) of the joined subclass table

.

Table Per-class Hierarchy (generated table)

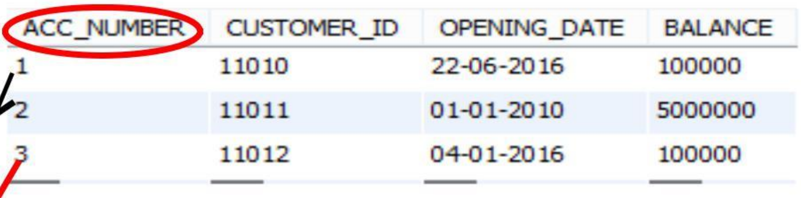


ACC_NUMBER	ACC_TYPE	CUSTOMER_ID	OPENING_DATE	BALANCE	INTEREST_RATE	MONTHLY_TXN_LIMIT	MINIMUM_BALANCE	LOAN_PERIOD	MONTHLY_INSTALLME
1	SAVING_ACCOUNT	11010	22-06-2016	100000	8	NULL	NULL	NULL	NULL
2	CURRENT_ACCOUNT	11011	01-01-2010	5000000	NULL	2500	500	NULL	NULL
3	LOAN_ACCOUNT	11012	04-01-2016	100000	10	NULL	NULL	2	2000
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

- ✓ In the above generated table you can see **ACC_TYPE** is **discriminator column** i.e. we pass inside the base entity mapping
- ✓ And **SAVING_ACCOUNT**, **CURRENT_ACCOUNT**, and **LOAN_ACCOUNT** is value that we have passed as discriminator value inside each subclass entity mapping.

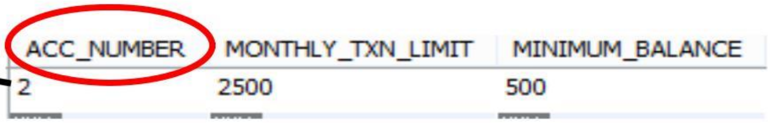
Table per Sub Class (Generated Table)

Select * from accounts;



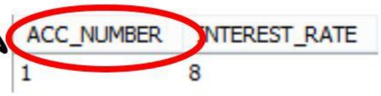
ACC_NUMBER	CUSTOMER_ID	OPENING_DATE	BALANCE
1	11010	22-06-2016	100000
2	11011	01-01-2010	5000000
3	11012	04-01-2016	100000

Select * from currunt_account;



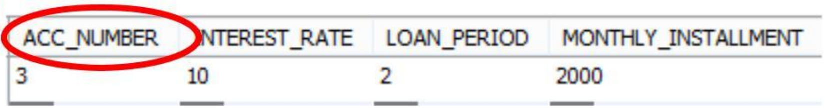
ACC_NUMBER	MONTHLY_TXN_LIMIT	MINIMUM_BALANCE
2	2500	500

Select * from saving_account;



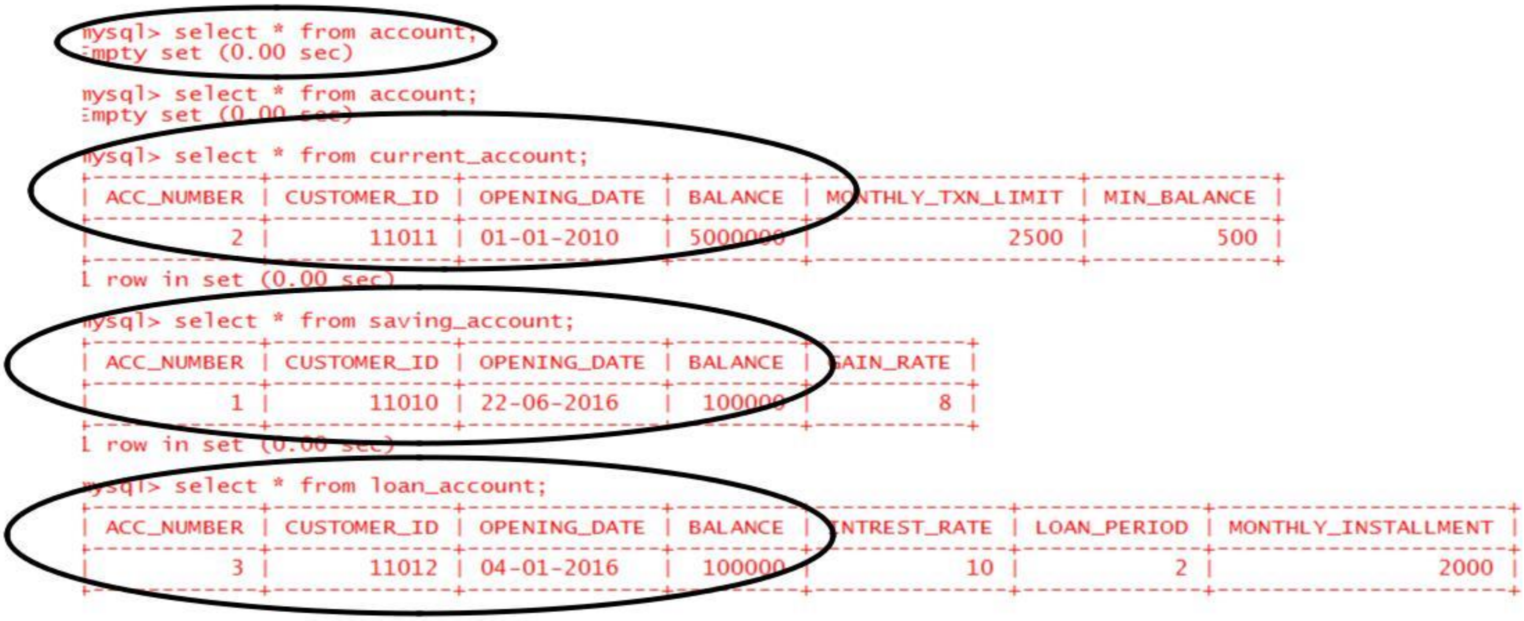
ACC_NUMBER	INTEREST_RATE
1	8

Select * from loan_account;



ACC_NUMBER	INTEREST_RATE	LOAN_PERIOD	MONTHLY_INSTALLMENT
3	10	2	2000

Table per concrete class



```
mysql> select * from account;
Empty set (0.00 sec)

mysql> select * from account;
Empty set (0.00 sec)

mysql> select * from current_account;
+------------+-------------+--------------+---------+-------------------+-------------+
| ACC_NUMBER | CUSTOMER_ID | OPENING_DATE | BALANCE | MONTHLY_TXN_LIMIT | MIN_BALANCE |
+------------+-------------+--------------+---------+-------------------+-------------+
|          2 |       11011 | 01-01-2010   | 5000000 |              2500 |         500 |
+------------+-------------+--------------+---------+-------------------+-------------+
1 row in set (0.00 sec)

mysql> select * from saving_account;
+------------+-------------+--------------+---------+-----------+
| ACC_NUMBER | CUSTOMER_ID | OPENING_DATE | BALANCE | GAIN_RATE |
+------------+-------------+--------------+---------+-----------+
|          1 |       11010 | 22-06-2016   |  100000 |         8 |
+------------+-------------+--------------+---------+-----------+
1 row in set (0.00 sec)

mysql> select * from loan_account;
+------------+-------------+--------------+---------+--------------+-------------+---------------------+
| ACC_NUMBER | CUSTOMER_ID | OPENING_DATE | BALANCE | INTREST_RATE | LOAN_PERIOD | MONTHLY_INSTALLMENT |
+------------+-------------+--------------+---------+--------------+-------------+---------------------+
|          3 |       11012 | 04-01-2016   |  100000 |           10 |           2 |                2000 |
+------------+-------------+--------------+---------+--------------+-------------+---------------------+
```

In table per concrete class it will generate table as per union concept. You can see in above table **ACC_NUMBER**, **CUSTOMER_ID**, **OPENING_DATE** and **BALANCE** is common because we pass these fields inside the base class **Account.** And other field will be added of subclass which we pass the property inside.