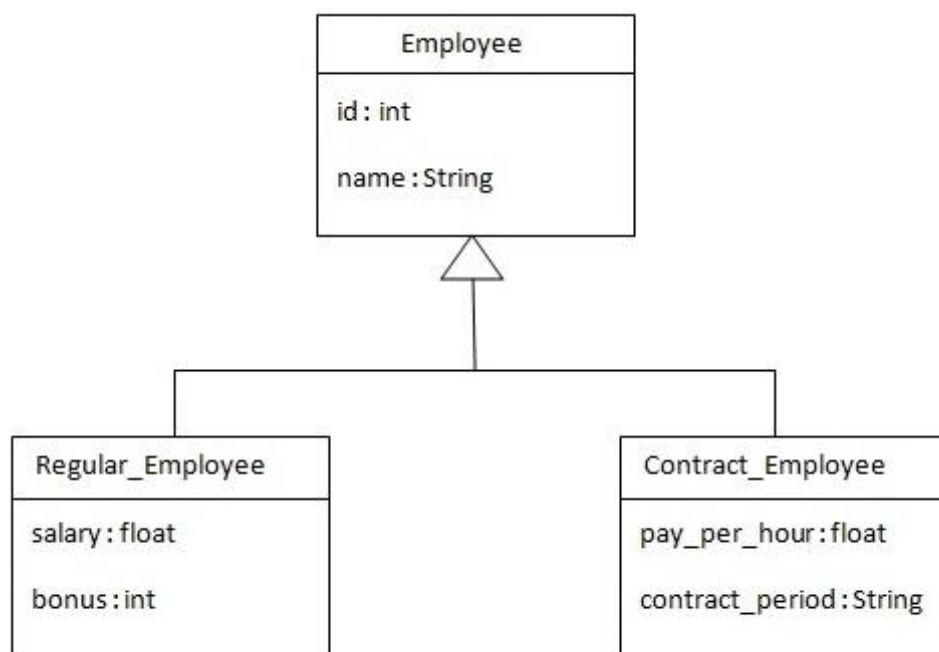


# Hibernate Table Per Hierarchy using xml file

By this inheritance strategy, we can map the whole hierarchy by single table only. Here, an extra column (also known as **discriminator column**) is created in the table to identify the class.

Let's understand the problem first. I want to map the whole hierarchy given below into one table of the database.



There are three classes in this hierarchy. Employee is the super class for Regular\_Employee and Contract\_Employee classes. Let's see the mapping file for this hierarchy.

```
<?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.javatpoint.mypackage.Employee" table="emp121" discriminator-
value="emp">
<id name="id">
<generator class="increment"></generator>
</id>
```

```

<discriminator column="type" type="string"></discriminator>
<property name="name"></property>

<subclass      name="com.javatpoint.mypackage.Regular_Employee"      discriminator-
value="reg_emp">
<property name="salary"></property>
<property name="bonus"></property>
</subclass>

<subclass      name="com.javatpoint.mypackage.Contract_Employee"      discriminator-
value="con_emp">
<property name="pay_per_hour"></property>
<property name="contract_duration"></property>
</subclass>

</class>

</hibernate-mapping>

```

In case of table per class hierarchy an discriminator column is added by the hibernate framework that specifies the type of the record. It is mainly used to distinguish the record. To specify this, **discriminator** subelement of class must be specified.

The **subclass** subelement of class, specifies the subclass. In this case, Regular\_Employee and Contract\_Employee are the subclasses of Employee class.

The table structure for this hierarchy is as shown below:

Column Name	Data Type	Nullable	Default	Primary Key
ID	NUMBER(10,0)	No	-	1
TYPE	VARCHAR2(255)	No	-	-
NAME	VARCHAR2(255)	Yes	-	-
SALARY	FLOAT	Yes	-	-
BONUS	NUMBER(10,0)	Yes	-	-
PAY_PER_HOUR	FLOAT	Yes	-	-
CONTRACT_DURATION	VARCHAR2(255)	Yes	-	-
1 - 7				

## Example of Table per class hierarchy

In this example we are creating the three classes and provide mapping of these classes in the employee.hbm.xml file.

## 1) Create the Persistent classes

You need to create the persistent classes representing the inheritance. Let's create the three classes for the above hierarchy:

*File: Employee.java*

```
package com.javatpoint.mypackage;

public class Employee {
    private int id;
    private String name;

    //getters and setters
}
```

*File: Regular\_Employee.java*

```
package com.javatpoint.mypackage;

public class Regular_Employee extends Employee{
    private float salary;
    private int bonus;

    //getters and setters
}
```

*File: Contract\_Employee.java*

```
package com.javatpoint.mypackage;

public class Contract_Employee extends Employee{
    private float pay_per_hour;
    private String contract_duration;

    //getters and setters
}
```

## 2) Create the mapping file for Persistent class

The mapping has been discussed above for the hierarchy.

*File: employee.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.javatpoint.mypackage.Employee"    table="emp121"    discriminator-
value="emp">
<id name="id">
<generator class="increment"></generator>
</id>

<discriminator column="type" type="string"></discriminator>
<property name="name"></property>

<subclass      name="com.javatpoint.mypackage.Regular_Employee"      discriminator-
value="reg_emp">
<property name="salary"></property>
<property name="bonus"></property>
</subclass>

<subclass      name="com.javatpoint.mypackage.Contract_Employee"      discriminator-
value="con_emp">
<property name="pay_per_hour"></property>
<property name="contract_duration"></property>
</subclass>

</class>

</hibernate-mapping>
```

## 3) Add mapping of hbm file in configuration file

Open the hibernate.cfg.xml file, and add an entry of mapping resource like this:

```
<mapping resource="employee.hbm.xml"/>
```

Now the configuration file will look like this:

*File: hibernate.cfg.xml*

```
<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE hibernate-configuration PUBLIC
    "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
    "http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">

<hibernate-configuration>

    <session-factory>
        <property name="hbm2ddl.auto">update</property>
        <property name="dialect">org.hibernate.dialect.Oracle9Dialect</property>
        <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
        <property name="connection.username">system</property>
        <property name="connection.password">oracle</property>
        <property name="connection.driver_class">oracle.jdbc.driver.OracleDriver</property>
        <mapping resource="employee.hbm.xml"/>
    </session-factory>

</hibernate-configuration>
```

The hbm2ddl.auto property is defined for creating automatic table in the database.

## 4) Create the class that stores the persistent object

In this class, we are simply storing the employee objects in the database.

*File: StoreData.java*




```
package com.javatpoint.mypackage;

import org.hibernate.*;
import org.hibernate.cfg.*;

public class StoreData {
```

```
public static void main(String[] args) {  
    Session session=new Configuration().configure("hibernate.cfg.xml")  
        .buildSessionFactory().openSession();  
  
    Transaction t=session.beginTransaction();  
  
    Employee e1=new Employee();  
    e1.setName("sonoo");  
  
    Regular_Employee e2=new Regular_Employee();  
    e2.setName("Vivek Kumar");  
    e2.setSalary(50000);  
    e2.setBonus(5);  
  
    Contract_Employee e3=new Contract_Employee();  
    e3.setName("Arjun Kumar");  
    e3.setPay_per_hour(1000);  
    e3.setContract_duration("15 hours");  
  
    session.persist(e1);  
    session.persist(e2);  
    session.persist(e3);  
  
    t.commit();  
    session.close();  
    System.out.println("success");  
}  
}
```

## Output:

EDIT	ID	TYPE	NAME	SALARY	BONUS	PAY_PER_HOUR	CONTRACT_DURATION
	1	emp	sonoo	-	-	-	-
	2	reg_emp	Vivek Kumar	50000	5	-	-
	3	con_emp	Arjun Kumar	-	-	1000	15 hours
							row(s) 1 - 3 of 3

[download this inheritance mapping example \(developed using Myeclipse IDE\)](#)  
[download this inheritance mapping example \(developed using Eclipse IDE\)](#)

## Topics in Hibernate Inheritance Mapping

[Table Per Hierarchy using xml file](#)

[Table Per Hierarchy using Annotation](#)

[Table Per Concrete class using xml file](#)

[Table Per Concrete class using Annotation](#)

[Table Per Subclass using xml file](#)

[Table Per Subclass using Annotation](#)

[← prev](#)

[next →](#)

Share  0