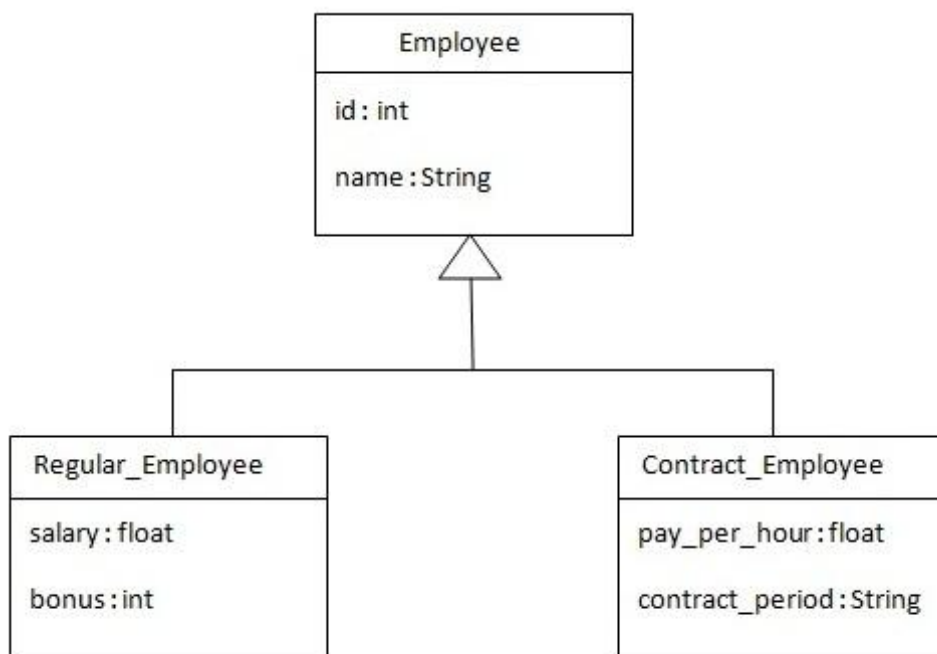


Hibernate Table Per Hierarchy using Annotation

In the previous page, we have mapped the inheritance hierarchy with one table only using xml file. Here, we are going to perform this task using annotation. You need to use `@Inheritance(strategy=InheritanceType.SINGLE_TABLE)`, `@DiscriminatorColumn` and `@DiscriminatorValue` annotations for mapping table per hierarchy strategy.

In case of table per hierarchy, only one table is required to map the inheritance hierarchy. Here, an extra column (also known as **discriminator column**) is created in the table to identify the class.

Let's see the inheritance hierarchy:



There are three classes in this hierarchy. Employee is the super class for Regular_Employee and Contract_Employee classes.

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 Hibernate Table Per Hierarchy using Annotation

You need to follow following steps to create simple example:

- Create the persistent classes
- Create the configuration file
- Create the class to store the fetch the data

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;
import javax.persistence.*;

@Entity
@Table(name = "employee101")
@Inheritance(strategy=InheritanceType.SINGLE_TABLE)
@DiscriminatorColumn(name="type",discriminatorType=DiscriminatorType.STRING)
@DiscriminatorValue(value="employee")

public class Employee {
    @Id
    @GeneratedValue(strategy=GenerationType.AUTO)

    @Column(name = "id")
    private int id;
```

```
@Column(name = "name")
private String name;

//setters and getters
}
```

File: Regular_Employee.java

```
package com.javatpoint.mypackage;

import javax.persistence.*;

@Entity
@DiscriminatorValue("regularemployee")
public class Regular_Employee extends Employee{

@Column(name="salary")
private float salary;

@Column(name="bonus")
private int bonus;

//setters and getters
}
```

File: Contract_Employee.java

```
package com.javatpoint.mypackage;

import javax.persistence.Column;
import javax.persistence.DiscriminatorValue;
import javax.persistence.Entity;

@Entity
@DiscriminatorValue("contractemployee")
public class Contract_Employee extends Employee{
```

```

@Column(name="pay_per_hour")
private float pay_per_hour;

@Column(name="contract_duration")
private String contract_duration;

//setters and getters
}

```

2) Add the persistent classes in configuration file

Open the hibernate.cfg.xml file, and add entries of entity classes like this:

```

<mapping class="com.javatpoint.mypackage.Employee"/>
<mapping class="com.javatpoint.mypackage.Contract_Employee"/>
<mapping class="com.javatpoint.mypackage.Regular_Employee"/>
</pre></div>
<table >
<tr><td>Now the configuration file will look like this:
</td></tr>
</table>
<span id="filename">File: hibernate.cfg.xml</span>
<div class="codeblock"><pre name="code" class="java" >
<?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">

<!-- Generated by MyEclipse Hibernate Tools.           -->
<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 class="com.javatpoint.mypackage.Employee"/>
<mapping class="com.javatpoint.mypackage.Contract_Employee"/>
<mapping class="com.javatpoint.mypackage.Regular_Employee"/>
</session-factory>

</hibernate-configuration>
```

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

3) Create the class that stores the persistent object

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

File: StoreTest.java

```
package com.javatpoint.mypackage;

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

public class StoreData {
public static void main(String[] args) {
    AnnotationConfiguration cfg=new AnnotationConfiguration();
    Session session=cfg.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)

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 18