

Hibernate-3-Component and Inheritance Mappings

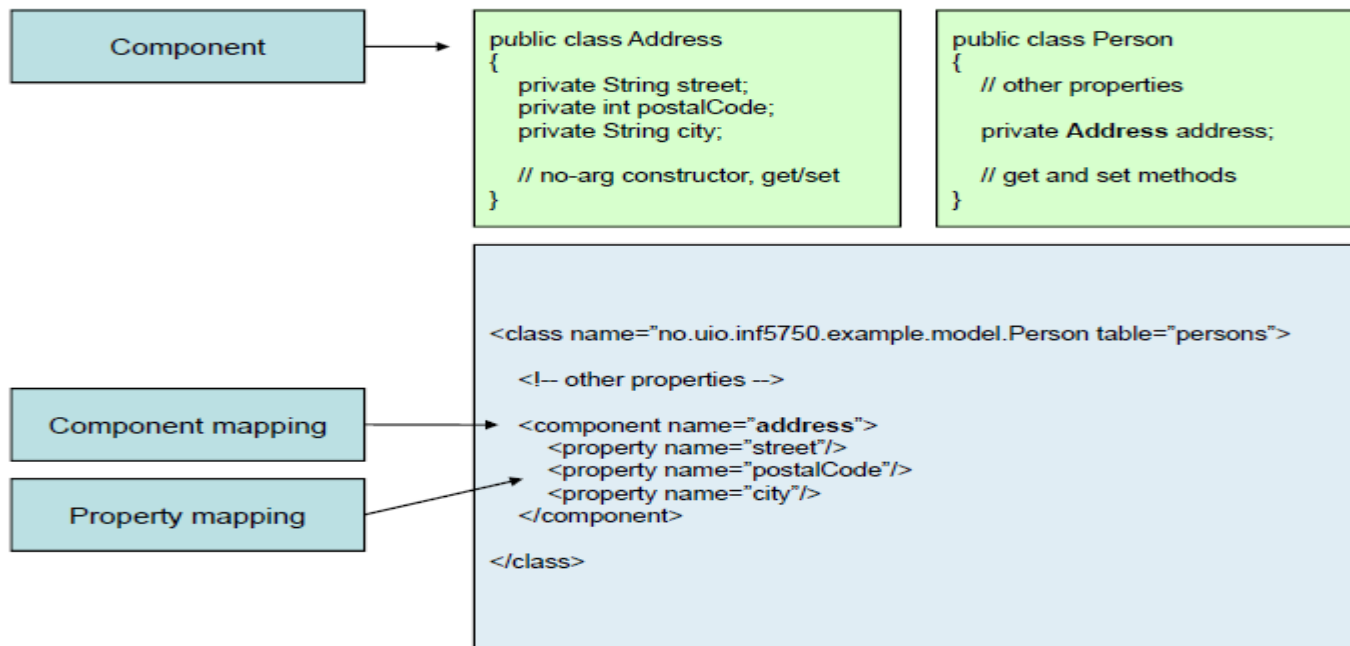
Presented by

Component Mapping

A component is an object saved as a value in another object

Resembles **HAS – A** relation

Ex : Person **HAS-A** Address



Component Mapping ...

```
public class Employee {  
    private int id;  
    private String name;  
    private Address address;//HAS-A relation  
  
    Employee(int id, String name, Address address ){  
        this.id = id;  
        this.name = name;  
        this.address = address;  
    }  
}
```



Component Mapping ...

```
Address preranaAddress = new Address();  
Address anirudhAddress = new Address();
```

```
preranaAddress.setCity("Bangalore");  
preranaAddress.setCountry("India");  
preranaAddress.setPinCode(560036);
```

```
anirudhAddress.setCity("Pune");  
anirudhAddress.setCountry("India");  
anirudhAddress.setPinCode(430098);
```

```
Employee prerana=new Employee(1003,"Prerana", preranaAddress);  
Employee anirudh=new Employee(1004,"Anirudh", anirudhAddress);
```

```
session.persist(prerana);  
session.persist(anirudh);
```

1 • `select * from employee;`

Filter:		↕	Edit			
	id	firstname	city	country	pincode	
▶	1001	Prerana	Bangalore	India	560036	
	1002	Anirudh	Pune	India	430098	
	1003	Prerana	Bangalore	India	560036	
	1004	Anirudh	Pune	India	430098	



Inheritance Mapping

- Mapping the inheritance hierarchy classes with the table of the database.
- There are three inheritance mapping strategies defined in the hibernate:
 - Table Per Hierarchy
 - Table Per Concrete class
 - Table Per Subclass

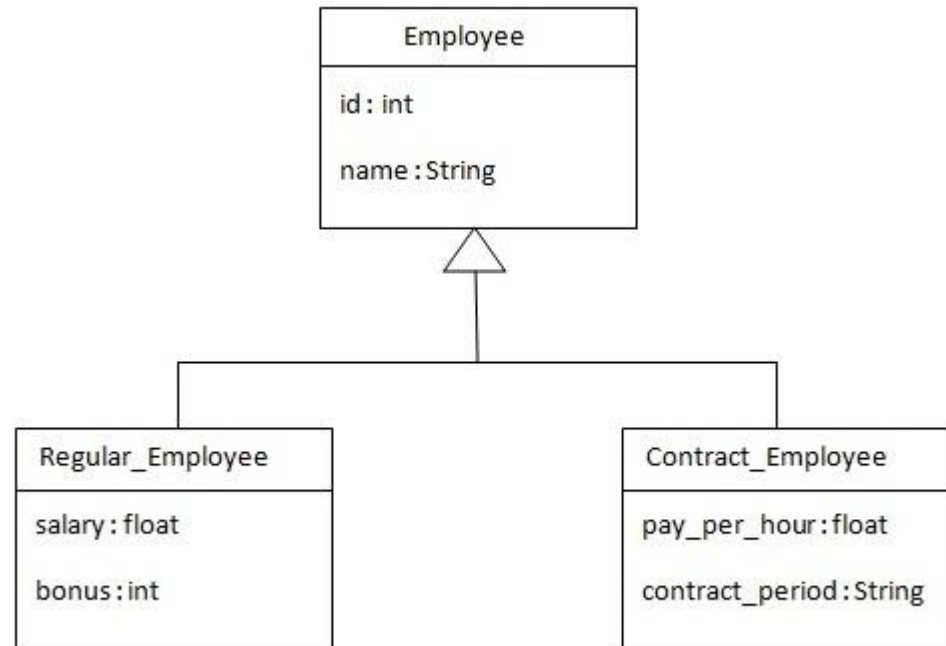
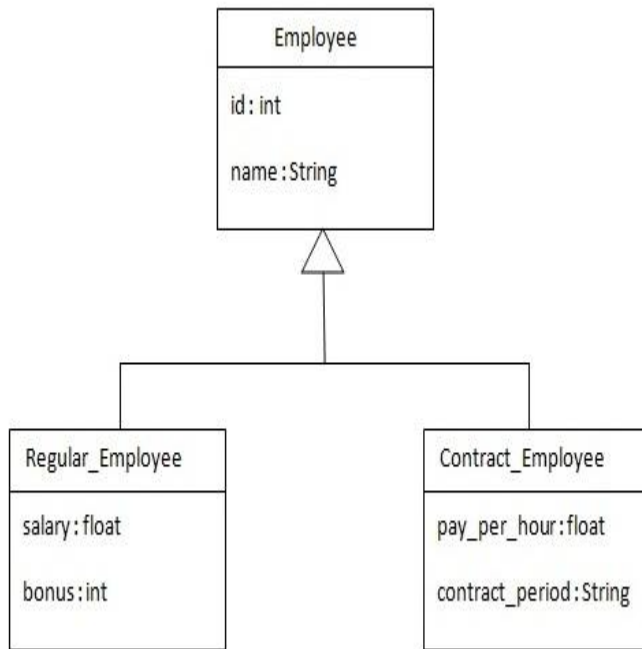


Table per Hierarchy

- Mapping the whole hierarchy by single table only.



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				



Table per Hierarchy ...

```
<class name="Employee" table="Employee" discriminator-value="emp">
  <id name="id">
    <generator class="increment"></generator>
  </id>
  <discriminator column="type" type="string"></discriminator>
  <property name="name"></property>
  <subclass name="RegularEmployee" discriminator-value="regularEmp">
    <property name="salary"></property>
    <property name="bonus"></property>
  </subclass>
  <subclass name="ContractEmployee" discriminator-value="contractEmp">
    <property name="payPerHour"></property>
    <property name="contractDuration"></property>
  </subclass>
</class>
```



Table per Hierarchy ...

An extra column (also known as discriminator column) is created in the table to identify the class.

```
RegularEmployee prerana = new RegularEmployee();
prerana.setName("Prerana");
prerana.setSalary(50000);
prerana.setBonus(5);
```

```
ContractEmployee karan = new ContractEmployee();
karan.setName("Karan");
karan.setPayPerHour(1000);
karan.setContractDuration("15 hours");
```

```
session.persist(prerana);
session.persist(karan);
```

```
1 • create database hibernatetableperhierarchy;
2 • use hibernatetableperhierarchy;
3 • select * from employee;
```



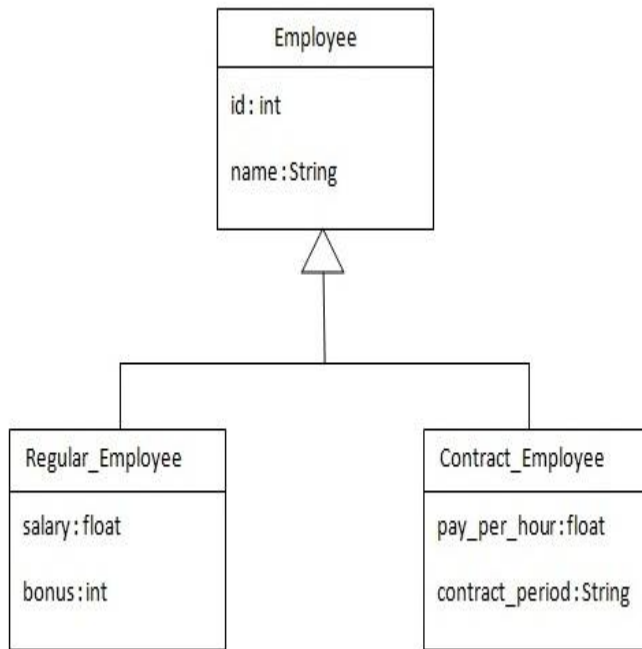
Filter: <input type="text"/> Export  Autosize: 							
	id	type	name	salary	bonus	payPerHour	contractDuration
▶	1	regularEmp	Prerana	50000	5	NULL	NULL
	2	contractEmp	Karan	NULL	NULL	1000	15 hours



Table Per Concrete Class

In case of Table Per Concrete class, there will be three tables in the database having no relations to each other.



Column Name	Data Type	Nullable	Default	Primary Key
ID	NUMBER(10,0)	No	-	1
NAME	VARCHAR2(255)	Yes	-	-
				1 - 2

Column Name	Data Type	Nullable	Default	Primary Key
ID	NUMBER(10,0)	No	-	1
NAME	VARCHAR2(255)	Yes	-	-
SALARY	FLOAT	Yes	-	-
BONUS	NUMBER(10,0)	Yes	-	-
				1 - 4

Column Name	Data Type	Nullable	Default	Primary Key
ID	NUMBER(10,0)	No	-	1
NAME	VARCHAR2(255)	Yes	-	-
PAY_PER_HOUR	FLOAT	Yes	-	-
CONTRACT_DURATION	VARCHAR2(255)	Yes	-	-
				1 - 4



Table Per Concrete Class ...

```
<class name="Employee" table="Employee">
  <id name="id">
    <generator class="increment"></generator>
  </id>
  <property name="name"></property>

  <union-subclass name="RegularEmployee" table="RegularEmployee">
    <property name="salary"></property>
    <property name="bonus"></property>
  </union-subclass>

  <union-subclass name="ContractEmployee" table="ContractEmployee">
    <property name="payPerHour"></property>
    <property name="contractDuration"></property>
  </union-subclass>
</class>
```



Table Per Concrete Class ...

```
Employee kishore=new Employee();  
kishore.setName("Kishore");
```

```
RegularEmployee prerana =new RegularEmployee();  
prerana.setName("Prerana");  
prerana.setSalary(50000);  
prerana.setBonus(5);
```




```
ContractEmployee karan=new ContractEmployee();  
karan.setName("Karan");  
karan.setPayPerHour(1000);  
karan.setContractDuration("15 hours");
```

```
session.save(kishore);  
session.save(prerana);  
session.save(karan);
```

1 •	use hibernatetableperconcreteclass;
2 •	select * from employee;




Filter:		Export	Autosize
	id	name	
▶	1	Kishore	

```
1 • use hibernatetableperconcreteclass;  
2 • select * from contractemployee;
```

Filter:  Export:  Autosize: 

	id	name	payPerHour	contractDuration
▶	3	Karan	1000	15 hours

```
1 • use hibernatetableperconcreteclass;  
2 • select * from regularemployee;
```

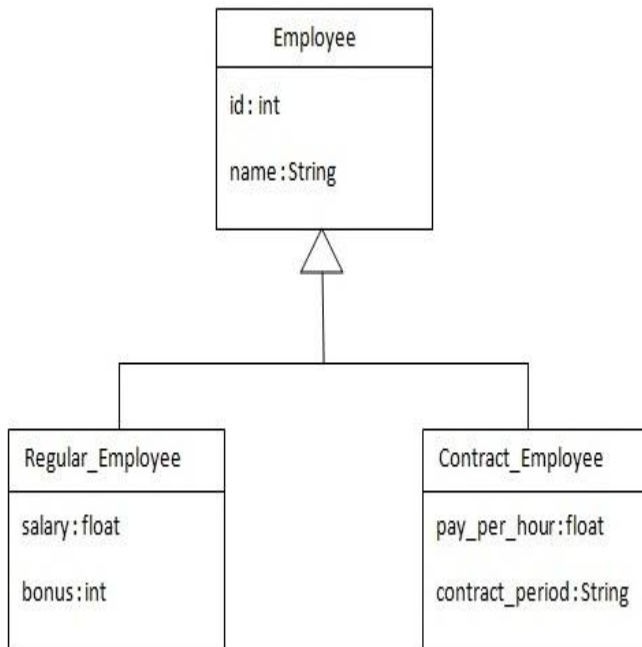
Filter:  Export:  Autosize: 

	id	name	salary	bonus
▶	2	Prerana	50000	5



Table Per Sub Class

In case of table per subclass strategy, tables are created as per persistent classes however, they are related using **primary and foreign key**



We need to specify –

-- In the **Super class**

@Inheritance(strategy=InheritanceType.JOINED)

-- in the **Sub class**

@PrimaryKeyJoinColumn



Table per sub class ...

Table structure for Employee class

Column Name	Data Type	Nullable	Default	Primary Key
ID	NUMBER(10,0)	No	-	1
NAME	VARCHAR2(255)	Yes	-	-
				1 - 2

Table structure for Regular_Employee class

Column Name	Data Type	Nullable	Default	Primary Key
EID	NUMBER(10,0)	No	-	1
SALARY	FLOAT	Yes	-	-
BONUS	NUMBER(10,0)	Yes	-	-
				1 - 3

Table structure for Contract_Employee class

Column Name	Data Type	Nullable	Default	Primary Key
EID	NUMBER(10,0)	No	-	1
PAY_PER_HOUR	FLOAT	Yes	-	-
CONTRACT_DURATION	VARCHAR2(255)	Yes	-	-
				1 - 3



No Mapping file - Annotations

The configuration file should have mapping tags as follows

```
<mapping class="Employee"/>
<mapping class="ContractEmployee"/>
<mapping class="RegularEmployee"/>
```

```
import javax.persistence.*;

@Entity
@Table(name = "Employee")
@Inheritance(strategy=InheritanceType.JOINED)
public class Employee {
    @Id
    @GeneratedValue(strategy=GenerationType.AUTO)

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

    @Column(name = "name")
    private String name;
    //getters and setters
    ...
}
```



Table per sub class ...Annotate

```
import javax.persistence.*;

@Entity
@Table(name = "Employee")
@Inheritance(strategy=InheritanceType.JOINED)
public class Employee {
    @Id
    @GeneratedValue(strategy=GenerationType.AUTO)

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

    @Column(name = "name")
    private String name;
    //getters and setters
    ...
}
```

Column Name	
ID	NAME
NAME	...



Table per sub class ...

```
1 • use hibernatetablepersubclass;  
2 • show tables;
```

Filter: Export: Autosize:

Tables_in_hibernatetablepersubclass	
▶	contractemployee
	employee
	regularemployee

```
1 • select * from employee;  
2
```

Filter: Export: Autosize:

	id	name
▶	1	Prerana
	2	Karan

```
1 • select * from contractemployee;  
2
```

Filter: Edit: Export: Autosize:

	ID	payPerHour	contractDuration
▶	2	1000	15 hours

```
1 • select * from regularemployee;  
2
```

Filter: Edit: Export: Autosize:

	ID	salary	bonus
▶	1	50000	5



