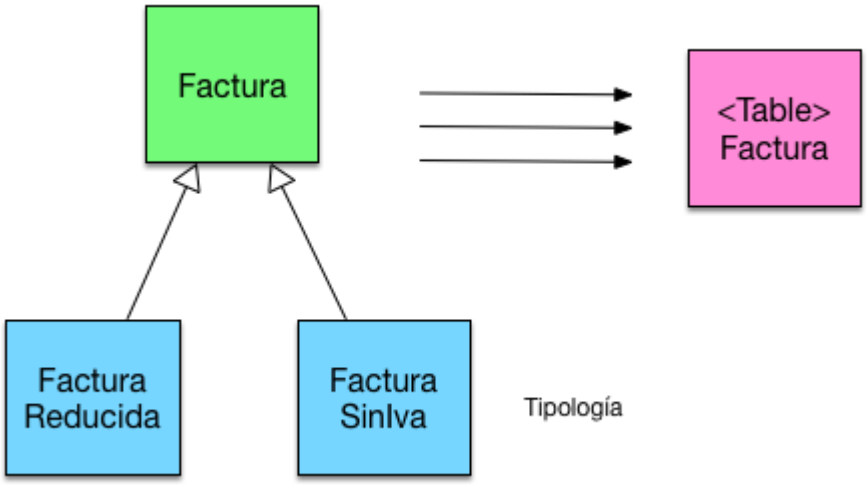
Vamos a construir un ejemplo de JPA Single Table inheritance. La herencia es una de las características que menos se comprende cuando hablamos de frameworks de persistencia. En este caso vamos a ver la casuística más sencilla, una jerarquía de clases que se almacenan en una única Tabla.



JPA Single Table Inheritance implementación

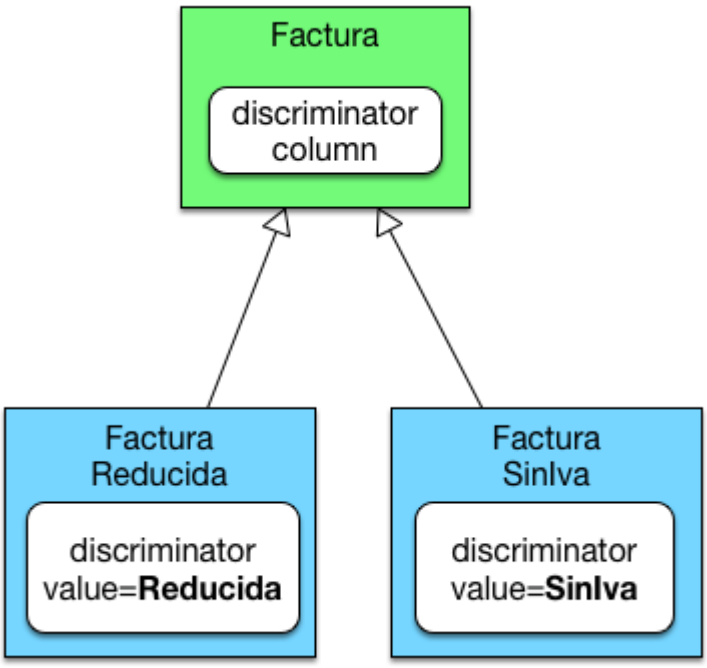
Vamos a construir las clases necesarias de la jerarquía , comenzando con la clase Factura.

```
package com.arquitecturajava.bo;

import javax.persistence.DiscriminatorColumn;
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.Inheritance;
import javax.persistence.InheritanceType;

@Entity
```

```
@DiscriminatorColumn(name="tipo")
@Inheritance(strategy = InheritanceType.SINGLE_TABLE)
public abstract class Factura {
 @Id
 private String numero;
 private String concepto;
 private double importe;

 public String getNumero() {
 return numero;
 }
 public void setNumero(String numero) {
 this.numero = numero;
 }
 public String getConcepto() {
 return concepto;
 }
 public void setConcepto(String concepto) {
 this.concepto = concepto;
 }
 public double getImporte() {
 return importe;
 }
 public void setImporte(double importe) {
 this.importe = importe;
 }

 public Factura(String numero, String concepto, double importe) {
 super();
 this.numero = numero;
 this.concepto = concepto;
```

```
 this.importe = importe;
 }
 public abstract double importeConIva();

}
```

En este caso estamos ante una clase abstracta que hace factor común de propiedades y métodos. Todas las facturas comparten , numero , concepto e importe . JPA añade el concepto de discriminador que define un campo de la tabla que identifica en que tipo de categoría se encuentra cada objeto como registro.



En este caso existen dos clases hijas y sobreescriben el método importeConIva() .Cada una de ellas usa el discriminatorvalue para definir su tipo a nivel de base de datos.

```
package com.arquitecturajava.bo;

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

@Entity
@DiscriminatorValue(value="Reducida")
public class FacturaReducida extends Factura{

 public FacturaReducida(String numero, String concepto, double
importe) {
 super(numero, concepto, importe);
 // TODO Auto-generated constructor stub
 }

 @Override
 public double importeConIva() {
 // TODO Auto-generated method stub
 return this.getImporte()*1.05;
 }


}
```

```
package com.arquitecturajava.bo;

import javax.persistence.DiscriminatorValue;
import javax.persistence.Entity;
```

```
@Entity
@DiscriminatorValue(value="SinIva")
public class FacturaSinIva extends Factura {


 public FacturaSinIva(String numero, String concepto, double importe)
{
 super(numero, concepto, importe);
 // TODO Auto-generated constructor stub
 }

 @Override
 public double importeConIva() {
 // TODO Auto-generated method stub
 return this.getImporte();
 }


}
```

JPA Single Table Inheritance persist

Vamos a usar JPA para persistir que ambas clases en la base de datos utilizando el diagrama de herencia.

```
package main;

import javax.persistence.EntityManager;
import javax.persistence.EntityManagerFactory;
import javax.persistence.EntityTransaction;
```

```
import javax.persistence.Persistence;

import com.arquitecturajava.bo.Factura;
import com.arquitecturajava.bo.FacturaReducida;
import com.arquitecturajava.bo.FacturaSinIva;
import com.arquitecturajava.bo.Persona;

public class Principal2 {

public static void main(String[] args) {

EntityManagerFactory emf =
Persistence.createEntityManagerFactory("CursoJPA");
EntityManager em = emf.createEntityManager();
EntityTransaction transaccion = em.getTransaction();

transaccion.begin();
Factura f = new FacturaReducida("1", "tableta", 100);
System.out.println(f.importeConIva());
Factura f2 = new FacturaSinIva("2", "tableta", 100);
System.out.println(f2.importeConIva());

em.persist(f);
em.persist(f2);
transaccion.commit();

}

}
```

Ambas facturas serán almacenadas en la la base de datos , dentro de la misma tabla

especificando el tipo de Factura usando el discriminador.

```
+--------+----------+---------+----------+
| numero | concepto | importe | tipo     |
+--------+----------+---------+----------+
| 1      | tableta  |     100 | Reducida |
| 2      | tableta  |     100 | SinIva   |
```

Acabamos de usar JPA Single Table Inheritance para almacenar una jerarquía de clases en la base de datos.

Otros artículos relacionados : JPA OneToMany , JPA First Level Cache , JPA Named Query Organización