Licenciatura en Sistemas - Orientación a Objetos II – 2024

Prof. Titular: Mg. María Alejandra Vranić

Prof. Ayudantes: Esp. Lic. Gustavo Siciliano
Lic. Ezequiel Scordamaglia
Lic. Oscar Ruina

DER y Diagrama de clases

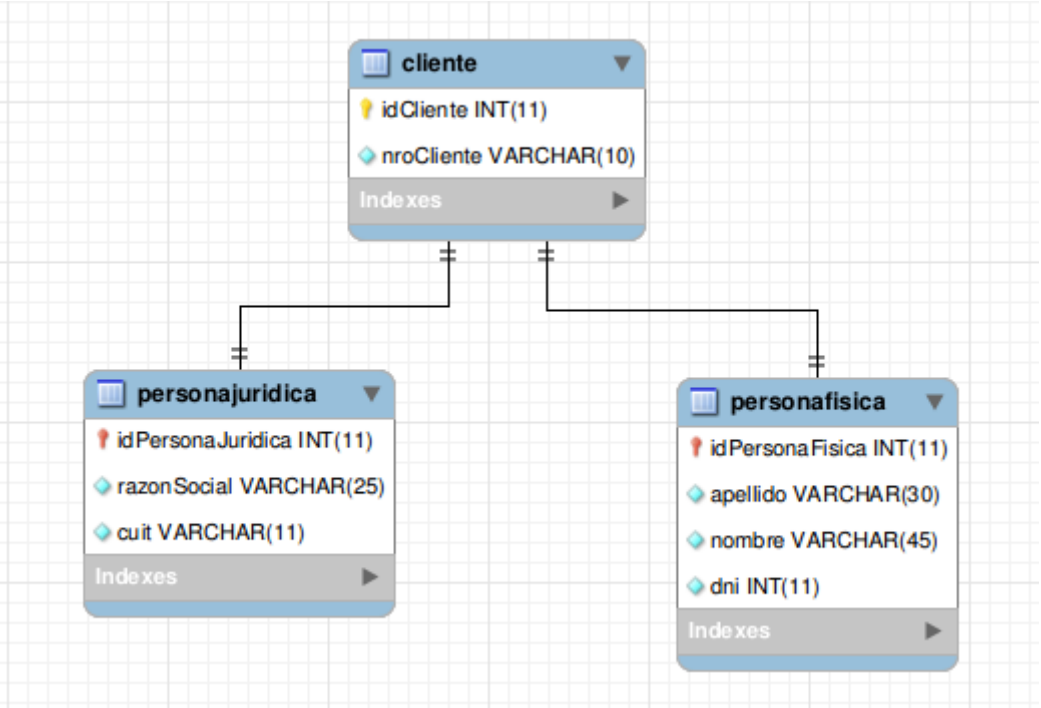

Diagrama entidad relación

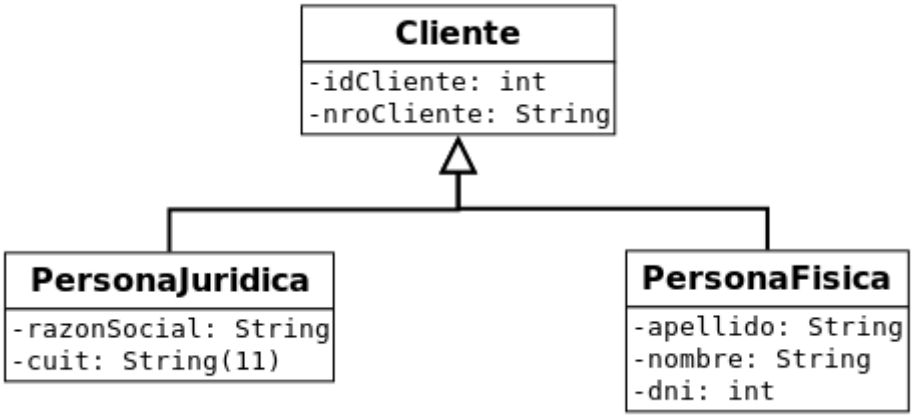

Diagrama de clases

Clases en la jerarquía de herencia

Cliente.java

```
package datos;
public abstract class Cliente {
	protected int idCliente;
	protected String nroCliente;

	public Cliente(){}
	public Cliente(String nroCliente) {
		super();
		this.nroCliente = nroCliente;
	}
	public int getIdCliente() {
		return idCliente;
	}
	protected void setIdCliente(int idCliente) {
		this.idCliente = idCliente;
	}
	public String getNroCliente() {
		return nroCliente;
	}
	public void setNroCliente(String nroCliente) {
		this.nroCliente = nroCliente;
	}
	@Override
	public String toString() {
		return "idCliente=" + idCliente + ", nroCliente=" + nroCliente;
	}
}
```

PersonaFisica.java

```
package datos;
public class PersonaFisica extends Cliente{
	private String apellido;
	private String nombre;
	private int dni;

	public PersonaFisica() {}
	public PersonaFisica(String nroCliente, String apellido, String nombre, int dni) {
		super(nroCliente);
		this.apellido = apellido;
		this.nombre = nombre;
		this.dni = dni;
	}

	public String getApellido() {
		return apellido;
	}
	public void setApellido(String apellido) {
		this.apellido = apellido;
	}
```

```
	public String getNombre() {
		return nombre;
	}
	public void setNombre(String nombre) {
		this.nombre = nombre;
	}
	public int getDni() {
		return dni;
	}
	public void setDni(int dni) {
		this.dni = dni;
	}
	@Override
	public String toString() {
		return "PersonaFisica [" + super.toString()+ ", apellido=" + apellido + ", nombre=" + nombre
				+ ", dni=" + dni + "]";
	}
}
```

PersonaJuridica.java

```
package datos;
public class PersonaJuridica extends Cliente{
	private String razonSocial;
	private String cuit;

	public PersonaJuridica() {}
	public PersonaJuridica(String nroCliente, String razonSocial, String cuit) {
		super(nroCliente);
		this.razonSocial = razonSocial;
		this.cuit = cuit;
	}

	public String getRazonSocial() {
		return razonSocial;
	}
	public void setRazonSocial(String razonSocial) {
		this.razonSocial = razonSocial;
	}
	public String getCuit() {
		return cuit;
	}
	public void setCuit(String cuit) {
		this.cuit = cuit;
	}
	@Override
	public String toString() {
		return "PersonaJuridica [" + super.toString() + ", razonSocial=" + razonSocial + ", cuit=" + cuit + "]";
	}

}
```

Mapeo con herencia

En la herencia únicamente se tiene que mapear a la superclase, en este caso: Cliente.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="datos.Cliente" table="cliente">
                <id column="idCliente" name="idCliente">
                        <generator class="identity" />
                </id>
                <property name="nroCliente" column="nroCliente" type="string" />

                <joined-subclass name="datos.PersonaJuridica" table="personajuridica">
                        <key column="idPersonaJuridica" />
                        <property column="razonSocial" name="razonSocial" type="string" />
                        <property column="cuit" name="cuit" type="string" />
                </joined-subclass>

                <joined-subclass name="datos.PersonaFisica" table="personafisica">
                        <key column="idPersonaFisica" />
                        <property column="apellido" name="apellido" type="string" />
                        <property column="nombre" name="nombre" type="string" />
                        <property column="dni" name="dni" type="int" />
                </joined-subclass>
        </class>
</hibernate-mapping>
```

```
public List<Cliente> traerClientes(LocalDate fechaNac) {
 List<Cliente> lista = null;
 try {
  iniciaOperacion();
  lista = session.createQuery("from PersonaFisica pf", Cliente.class).list();
 } finally {
  session.close();
 }
 return lista;
}
```

```
public List<Cliente> traerFechaDePrestamo(LocalDate fecha) {
 List<Cliente> lista = new ArrayList<Cliente>();
 try {
  iniciaOperacion();
  lista = session.createQuery(
    "select distinct c from Cliente c inner join fetch c.prestamos p where p.fecha = :fecha",
    Cliente.class).setParameter("fecha", fecha).list();
 } finally {
  // TODO: handle finally clause
  session.close();

 }
 return lista;
}
```

ClienteDAO.java

```
package dao;
import java.util.List;
import org.hibernate.HibernateException;
import org.hibernate.Session;
import org.hibernate.Transaction;
import datos.Cliente;

public class ClienteDao {

	private static Session session;
	private Transaction tx;
	private static ClienteDao instancia = null; // Patrón Singleton

	protected ClienteDao() {
	}
	public static ClienteDao getInstance() {
		if (instancia == null)
			instancia = new ClienteDao();
		return instancia;
	}
	protected void iniciaOperacion() throws HibernateException {
		session = HibernateUtil.getSessionFactory().openSession();
		tx = session.beginTransaction();
	}
	protected void manejaExcepcion(HibernateException he) throws HibernateException {
		tx.rollback();
		throw new HibernateException("ERROR en la capa de acceso a datos", he);
	}

	public Cliente traer(int idCliente) {
		Cliente objeto = null;
		try {
			iniciaOperacion();
			objeto = (Cliente) session.createQuery("from Cliente c where c.idCliente=:idCliente")
					.setParameter("idCliente", idCliente).uniqueResult();
		} finally {
			session.close();
		}
		return objeto;
	}

	public List<Cliente> traer() throws HibernateException {
		List<Cliente> lista = null;
		try {
			iniciaOperacion();
			lista = session.createQuery("from Cliente", Cliente.class).list();
		} finally {
			session.close();
		}
		return lista;
	}
}
   public List<Cliente> traerClientes(LocalDate fechaDesde,LocalDate fechaHasta) {
     List<Cliente> lista = null;
     try {
      iniciaOperacion();
      lista = session.createQuery("from PersonaFisica pf where pf.fechaNacimiento betweeen"
        + ":fechaDesde AND :fechaHasta", Cliente.class).
        setParameter("fechaDesde", fechaDesde).setParameter("fechaHasta", fechaHasta).list();
     } finally {
      session.close();
     }
```

ClienteABM.java

```
package negocio;
import java.util.List;
import dao.ClienteDao;
import datos.Cliente;
public class ClienteAbm {
	private static ClienteAbm instancia = null; // Patrón Singleton
	protected ClienteAbm() {
	}
	public static ClienteAbm getInstance() {
		if (instancia == null)
			instancia = new ClienteAbm();
		return instancia;
	}

	public Cliente traer(int idCliente) {
		return ClienteDao.getInstance().traer(idCliente);
	}

	public List<Cliente> traer() {
		return ClienteDao.getInstance().traer();
	}
}
```

Testeos

```
package test;
import datos.Cliente;
import negocio.ClienteAbm;
public class TestCliente {

	public static void main(String[] args) {
		int idCliente=1;
		System.out.printf("+ traer(%d)\n", idCliente);
		System.out.println(ClienteAbm.getInstance().traer(idCliente));
		idCliente=2;
		System.out.printf("\n+ traer(%d)\n", idCliente);
		System.out.println(ClienteAbm.getInstance().traer(idCliente));

		System.out.println("\n+ traer()");
		for (Cliente c: ClienteAbm.getInstance().traer()) {
			System.out.println(c);
		}
	}
}
```

```
	public List<Cliente>traerClientesFisicosConApellido(LocalDate desde, LocalDate hasta, String apellido){
	 List<Cliente> clientes= new ArrayList<Cliente>();
	 for(Cliente c:dao.traer(desde, hasta)) {
	  if((PersonaFisica) c).getApellido().equalsIgnoreCase(apellido)){
	   clientes.add(c);
	  }
	 }
	 return clientes;
	}
```