Práctica No. 2. Operaciones Básicas con Hibernate - Inserciones

Preparación del Entorno

- Abrir el entorno de desarrollo *Eclipse*
- Crear un proyecto Java llamado ***OperacionesBasicasInsertar***
- Se creará un nuevo directorio llamado ***lib*** (*Clic derecho sobre el proyecto → New → Folder*)
- Copiar a este directorio las bibliotecas necesarias de *Hibernate* y de la conexión para *MySQL*
- Agregar las bibliotecas al proyecto para que las reconozca
- Crear los siguientes paquetes:
  - ***hibernate.principal***
  - ***hibernate.conexion***
  - ***hibernate.operaciones***
  - ***hibernate.beans***
- Crear y configurar el archivo ***hibernate.cfg.xml*** de acuerdo a lo realizado en la práctica anterior
- Crear la clase ***CrearConexion*** en el paquete ***hibernate.conexion***
- Crear la clase ***Principal*** en el paquete ***hibernate.principal***
- Crear la clase ***OperacionesAlumno*** en el paquete ***hibernate.operaciones***
- Crear la clase ***OperacionesLicenciatura*** en el paquete ***hibernate.operaciones***
- Crear la clase ***OperacionesUea*** en el paquete ***hibernate.operaciones***
- Crear la clase ***OperacionesAlumnoUea*** en el paquete ***hibernate.operaciones***

Preparación de la Base de Datos

- Descargar de la página web http://academicos.azc.uam.mx/jfg/pags/tarea_taller_web.html el archivo ***script_01_generacion_clase.sql*** el cuál contiene la base de datos y las tablas a utilizar.
- Abrir el entorno MySQL Workbench y ejecutar el *script* para crear la base de datos y las tablas

Clases Java Beans

Se crearán las clases *POJO* con sus métodos *get/set* para que cumplan los requisitos de un *Java Bean*, no es necesario que los nombres de los atributos sean iguales a los nombres de las columnas en la tabla.

- Se crearán las siguientes clases en el paquete ***hibernate.beans***
  - **Licenciatura**
  - **Alumno**
  - **Uea**
  - **AlumnoUea**

Licenciatura.java

```
package hibernate.beans;

public class Licenciatura {

	private String clave;
	private String nombre;

	public String getClave() {
		return clave;
	}
	public void setClave(String clave) {
		this.clave = clave;
	}
	public String getNombre() {
		return nombre;
	}
	public void setNombre(String nombre) {
		this.nombre = nombre;
	}

}
```

Alumno.java

```
package hibernate.beans;

public class Alumno {

	private String matricula;
	private String nombre;
	private String primerApellido;
	private String segundoApellido;
	private String licenciatura;


	public String getMatricula() {
		return matricula;
	}
	public void setMatricula(String matricula) {
		this.matricula = matricula;
	}
	public String getNombre() {
		return nombre;
	}
	public void setNombre(String nombre) {
		this.nombre = nombre;
	}
```

```
	public String getPrimerApellido() {
		return primerApellido;
	}
	public void setPrimerApellido(String primerApellido) {
		this.primerApellido = primerApellido;
	}
	public String getSegundoApellido() {
		return segundoApellido;
	}
	public void setSegundoApellido(String segundoApellido) {
		this.segundoApellido = segundoApellido;
	}
	public String getLicenciatura() {
		return licenciatura;
	}
	public void setLicenciatura(String licenciatura) {
		this.licenciatura = licenciatura;
	}

}
```

Uea.java

```
package hibernate.beans;

public class Uea {

	private String claveUea;
	private String nombre;
	private int creditos;

	public String getClaveUea() {
		return claveUea;
	}
	public void setClaveUea(String claveUea) {
		this.claveUea = claveUea;
	}
	public String getNombre() {
		return nombre;
	}
	public void setNombre(String nombre) {
		this.nombre = nombre;
	}
	public int getCreditos() {
		return creditos;
	}
	public void setCreditos(int creditos) {
		this.creditos = creditos;
	}
}
```

AlumnoUea.java

```
package hibernate.beans;

public class AlumnoUea {

	private int idAlumnoUea;
	private String trimestre;
	private String calificacion;
	private float calificacionNumerica;
	private String alumnoMatricula;
	private String ueaClaveUea;

	public int getIdAlumnoUea() {
		return idAlumnoUea;
	}
	public void setIdAlumnoUea(int idAlumnoUea) {
		this.idAlumnoUea = idAlumnoUea;
	}
	public String getTrimestre() {
		return trimestre;
	}
	public void setTrimestre(String trimestre) {
		this.trimestre = trimestre;
	}
	public String getCalificacion() {
		return calificacion;
	}
	public void setCalificacion(String calificacion) {
		this.calificacion = calificacion;
	}
	public float getCalificacionNumerica() {
		return calificacionNumerica;
	}
	public void setCalificacionNumerica(float calificacionNumerica) {
		this.calificacionNumerica = calificacionNumerica;
	}
	public String getAlumnoMatricula() {
		return alumnoMatricula;
	}
	public void setAlumnoMatricula(String alumnoMatricula) {
		this.alumnoMatricula = alumnoMatricula;
	}
	public String getUeaClaveUea() {
		return ueaClaveUea;
	}
	public void setUeaClaveUea(String ueaClaveUea) {
		this.ueaClaveUea = ueaClaveUea;
	}
}
```

Mapeos de las Clases (Objetos) con las Tablas

Es necesario crear archivos que contengan la relación entre las columnas de las tablas y los atributos de las clases a utilizar en la aplicación. El nombre del archivo puede o no ser el mismo que la clase o tabla, pero si debe incluir la extensión ***hbm.xml***.

Lo común es crear un archivo *XML* por clase, aunque se pueden realizar todos los mapeos en uno solo. La ubicación del o los archivos de mapeo es indistinta pudiendo colocarse en su propio paquete. En esta práctica, se creará un archivo XML por clase y el nombre será el mismo que el de la clase correspondiente.

Cada mapeo tendrá un indicador de qué atributo corresponde a la llave primaria **<*id*>**, además de los nombres de columnas, nombres de atributos y tipo de mapeo, esto con la etiqueta **<property>** y las propiedades ***name*** (para el atributo), ***column*** (para la columna) y ***type*** (para el tipo del mapeo).

```
<id name="matricula" column="matricula" type="string"/>
<property name="nombre" column="nombre" type="string"/>
<property name="primerApellido" column="primer_apellido" type="string"/>
```

Primero se creará el paquete ***hibernate.mapeos*** en donde se colocarán los archivos *XML* de mapeo.

Alumno.hbm.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<hibernate-mapping>
   <class name="hibernate.beans.Alumno" table="alumno">
      <id name="matricula" column="matricula" type="string"/>
      <property name="nombre" column="nombre" type="string"/>
      <property name="primerApellido" column="primer_apellido"
type="string"/>
      <property name="segundoApellido" column="segundo_apellido"
type="string"/>
      <property name="licenciatura" column="licenciatura_clave"
type="string"/>
   </class>
</hibernate-mapping>
```

AlumnoUea.hbm.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<hibernate-mapping>
 <class name="hibernate.beans.AlumnoUea" table="alumno_uea">
      <id name="idAlumnoUea" column="id_alumno_uea" type="int">
           <generator class="native"/>
      </id>
      <property name="trimestre" column="trimestre" type="string"/>
      <property name="calificacion" column="calificacion" type="string"/>
      <property name="calificacionNumerica" column="calificacion_numerica"
type="float"/>
```

```
      <property name="alumnoMatricula" column="alumno_matricula"
type="string"/>
      <property name="ueaClaveUea" column="uea_clave_uea" type="string"/>
   </class>
</hibernate-mapping>
```

Licenciatura.hbm.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<hibernate-mapping>
   <class name="hibernate.beans.Licenciatura" table="licenciatura">
      <id name="clave" column="clave" type="string"/>
      <property name="nombre" column="nombre" type="string"/>
   </class>
</hibernate-mapping>
```

Uea.hbm.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<hibernate-mapping>
   <class name="hibernate.beans.Uea" table="uea">
      <id name="claveUea" column="clave_uea" type="string"/>
      <property name="nombre" column="nombre" type="string"/>
      <property name="creditos" column="creditos" type="int"/>
   </class>
</hibernate-mapping>
```

La ubicación y mapeo de estos archivos se especifica en el archivo de configuración ***hibernate.cfg.xml***

hibernate.cfg.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<hibernate-configuration>
  <session-factory>
    <property name="hibernate.connection.driver_class">
    com.mysql.jdbc.Driver
    </property>
    <property name="hibernate.connection.url">
    jdbc:mysql://localhost:3306/practicas_taller_web
    </property>
    <property name="hibernate.connection.username">
    root
    </property>
    <property name="hibernate.connection.password">
    root
    </property>
    <property name="hibernate.dialect">
    org.hibernate.dialect.MySQLDialect
    </property>
    <mapping resource="hibernate/mapeos/Alumno.hbm.xml"/>
```

```
    <mapping resource="hibernate/mapeos/Uea.hbm.xml"/>
    <mapping resource="hibernate/mapeos/Licenciatura.hbm.xml"/>
    <mapping resource="hibernate/mapeos/AlumnoUea.hbm.xml"/>
  </session-factory>
</hibernate-configuration>
```

Conexión a la base de datos

Completar la clase ***CrearConexión***

CrearConexion.java

```
package hibernate.conexion;

import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

public class CrearConexion {

     private static final SessionFactory sessionFactory =
buildSessionFactory();

    private static SessionFactory buildSessionFactory() {
        try {
           return new Configuration().configure().buildSessionFactory();
            }
            catch (Throwable ex) {
             System.err.println("Error al crear sessionFactory " + ex);
             throw new ExceptionInInitializerError(ex);
              }
         }


         public static SessionFactory getSessionFactory() {
             return sessionFactory;
         }

         public static void closeSessionFactory(){
           sessionFactory.close();
         }

}
```

Se agrega el método para permitir que se cierre el objeto ***SessionFactory*** utilizado para la creación de las ***Session***

Operaciones de Inserción

Se completarán las clases ***OperacionesAlumno***, ***OperacionesLicenciatura***, ***OperacionesUea*** y ***OperacionesAlumnoUea*** para realizar las inserciones correspondientes. Se deben considerar las restricciones de cada una de las tablas.

Para realizar una inserción (y muchas otras operaciones) es necesario comenzar una Transacción (***Transaction***), esto es con el método ***beginTransaction()*** del objeto ***Session***.

Una vez creada, se utiliza el método ***save()*** al cuál se le pasa el objeto a guardar, el cuál ya habrá sido mapeado en los archivos **XML** correspondientes.

Posteriormente se debe realizar un *commit* para asegurar que se reflejan los cambios en la Base de Datos.

OperacionesLicenciatura.java

```
package hibernate.operaciones;

import hibernate.beans.Licenciatura;
import hibernate.conexion.CrearConexion;
import org.hibernate.Session;

public class OperacionesLicenciatura {

     public void insertarLicenciatura(Licenciatura licenciatura){

          try{
               Session sesion =
CrearConexion.getSessionFactory().openSession();
               sesion.beginTransaction();


               sesion.save(licenciatura);
            System.out.println("Licenciatura Registrada");
            sesion.getTransaction().commit();
            sesion.close();

          }catch (Throwable ex) {
             System.err.println("No se pudo crear la sesion " + ex);
             throw new ExceptionInInitializerError(ex);
          }
     }
}
```

Con base en esto, se completan el resto de las clases y métodos para insertar cada uno de los elementos en la Base de Datos.

OperacionesAlumno.java

```
package hibernate.operaciones;

import org.hibernate.Session;
import hibernate.beans.Alumno;
import hibernate.conexion.CrearConexion;

public class OperacionesAlumno {

	public void insertarAlumno(Alumno alumno){


		try{
			Session sesion =
CrearConexion.getSessionFactory().openSession();
			sesion.beginTransaction();


			sesion.save(alumno);
		  System.out.println("Alumno Registrado");
		  sesion.getTransaction().commit();
		  sesion.close();

		 }catch (Throwable ex) {
		    System.err.println("No se pudo crear la sesion " + ex);
		    throw new ExceptionInInitializerError(ex);
		 }

	}

}
```

OperacionesUea.java

```
package hibernate.operaciones;

import org.hibernate.Session;
import hibernate.beans.Uea;
import hibernate.conexion.CrearConexion;

public class OperacionesUea {

	public void insertarUea(Uea uea){

		try{
```

```
                Session sesion =
CrearConexion.getSessionFactory().openSession();
                sesion.beginTransaction();
                sesion.save(uea);
              System.out.println("Uea Registrada");
              sesion.getTransaction().commit();
              sesion.close();

            }catch (Throwable ex) {
               System.err.println("No se pudo crear la sesion " + ex);
               throw new ExceptionInInitializerError(ex);
            }

      }
}
```

OperacionesAlumnoUea.java

```
package hibernate.operaciones;

import hibernate.beans.AlumnoUea;
import hibernate.conexion.CrearConexion;
import org.hibernate.Session;

public class OperacionesAlumnoUea {

      public void insertarAlumnoUea(AlumnoUea alumnoUea){

            try{
                Session sesion =
CrearConexion.getSessionFactory().openSession();
                sesion.beginTransaction();

                sesion.save(alumnoUea);
              System.out.println("Relación Alumno Uea Registrada");
              sesion.getTransaction().commit();
              sesion.close();

            }catch (Throwable ex) {
               System.err.println("No se pudo crear la sesion " + ex);
               throw new ExceptionInInitializerError(ex);
            }

      }
}
```

Aunque no es común, es posible insertar los elementos de una colección, en este caso se utilizará una lista ligada (***LinkedList***).

OperacionesLicenciatura.java

```
	public void insertarListaLicenciatura(LinkedList<Licenciatura>lista)
{
		try{
			Session sesion =
CrearConexion.getSessionFactory().openSession();
			sesion.beginTransaction();
			for(int i=0;i<lista.size();i++){
				sesion.save(lista.get(i));
				}
			sesion.getTransaction().commit();
			sesion.close();
		}catch (Throwable ex) {
		    System.err.println("No se pudo crear la sesion " + ex);
		    throw new ExceptionInInitializerError(ex);
		 }
	}
```

Finalmente se crea la clase principal que invocará a varios métodos a manera de prueba, recodando que para cada prueba se deben modificar algunos valores que sirvan como llave primaria

Principal.java

```
package hibernate.principal;

import hibernate.beans.Alumno;
import hibernate.beans.AlumnoUea;
import hibernate.beans.Licenciatura;
import hibernate.beans.Uea;
import hibernate.conexion.CrearConexion;
import hibernate.operaciones.OperacionesAlumno;
import hibernate.operaciones.OperacionesAlumnoUea;
import hibernate.operaciones.OperacionesLicenciatura;
import hibernate.operaciones.OperacionesUea;

public class Principal {
	public static void main(String[] args) {

		insertarLicenciatura();
		insertarAlumno();
		insertarUea();
		relacionAlumnoUea();
		insertarLista();
		CrearConexion.closeSessionFactory();
	}
```

```
	public static void insertarLicenciatura(){

		OperacionesLicenciatura operaciones = new
OperacionesLicenciatura();
		Licenciatura licenciatura = new Licenciatura();

		licenciatura.setClave("110090");
		licenciatura.setNombre("Nombre de la Licenciatura");

		operaciones.insertarLicenciatura(licenciatura);

	}

	public static void insertarAlumno(){
		OperacionesAlumno operaciones = new OperacionesAlumno();
		Alumno alumno = new Alumno();

		alumno.setLicenciatura("110014");
		alumno.setMatricula("23456");
		alumno.setNombre("Nombre Alumno");
		alumno.setPrimerApellido("Primer Apellido");
		alumno.setSegundoApellido("Segundo Apellido");

		operaciones.insertarAlumno(alumno);
	}

	public static void insertarUea(){
		OperacionesUea operaciones = new OperacionesUea();
		Uea uea = new Uea();

		uea.setClaveUea("1112");
		uea.setNombre("Nombre de la UEA");
		uea.setCreditos(9);

		operaciones.insertarUea(uea);
	}
	public static void relacionAlumnoUea(){

		OperacionesAlumnoUea operaciones = new OperacionesAlumnoUea();
		AlumnoUea alumnoUea = new AlumnoUea();

		alumnoUea.setAlumnoMatricula("23456");
		alumnoUea.setCalificacion("MB");
		alumnoUea.setCalificacionNumerica(9.5F);
		alumnoUea.setTrimestre("16-P");
		alumnoUea.setUeaClaveUea("1111");

		operaciones.insertarAlumnoUea(alumnoUea);
	}
```

```
	public static void insertarLista(){
			OperacionesLicenciatura operaciones = new
	OperacionesLicenciatura();

			LinkedList <Licenciatura>lista = new
	LinkedList<Licenciatura>();
		Licenciatura licenciaturaLista1 = new Licenciatura();
		Licenciatura licenciaturaLista2 = new Licenciatura();
		Licenciatura licenciaturaLista3 = new Licenciatura();
		Licenciatura licenciaturaLista4 = new Licenciatura();

		licenciaturaLista1.setClave("L001");
		licenciaturaLista1.setNombre("Licenciatura 001");

		licenciaturaLista2.setClave("L002");
		licenciaturaLista2.setNombre("Licenciatura 002");

		licenciaturaLista3.setClave("L003");
		licenciaturaLista3.setNombre("Licenciatura 003");

		licenciaturaLista4.setClave("L004");
		licenciaturaLista4.setNombre("Licenciatura 004");

		lista.add(licenciaturaLista1);
		lista.add(licenciaturaLista2);
		lista.add(licenciaturaLista3);
		lista.add(licenciaturaLista4);

		operaciones.insertarListaLicenciatura(lista);

	}
}
```

Con esto se han insertado elementos, sin embargo, las bases de datos suelen tener restricciones, por lo que se pueden generar varios errores al momento de insertar elementos nuevos. A continuación se manejarán estos posibles errores.

Excepciones y Logs

Es importante tener una manera de revisar los eventos que pudieran aparecer en la aplicación. Si bien, se tiene disponible la Consola, en algún momento, al ser una aplicación web, no será posible utilizarla, por lo que se configurará un manejo utilizando archivos de log (bitácoras).

Se comenzará descargando la biblioteca necesaria de:

https://logging.apache.org/log4j/1.2/download.html

Se seleccionará el archivo ZIP o TAR.GZ dependiendo lo que se desee y descomprimirlo. De todos los archivos disponibles, se utilizará: ***log4j-1.2.17.jar*** el cuál se copiará al directorio de bibliotecas del proyecto y se agregará de la misma forma que las de *Hibernate* y *JDBC*.

Lo primero es crear un archivo de configuración en el directorio base del proyecto (**NO** el directorio de fuentes), el nombre del archivo puede ser el que se desee y para esta práctica se utilizará ***logger.properties*** (un nombre común es ***log4j.properties***).

El contenido del archivo es el siguiente:

```
log4j.rootCategory=INFO,ARCHIVO
log4j.appender.ARCHIVO=org.apache.log4j.FileAppender
log4j.appender.ARCHIVO.file=logging.log
log4j.appender.ARCHIVO.layout=org.apache.log4j.PatternLayout
log4j.appender.ARCHIVO.append=FALSE
log4j.appender.ARCHIVO.layout.ConversionPattern=%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}::%M::%L - %m%n
```

Lo primero es crear un objeto de tipo Logger que se encargará de almacenar los mensajes deseados.

```
private final static Logger log = Logger.getLogger(OperacionesAlumno.class);
```

También se debe configurar de acuerdo a los parámetros especificados en el archivo de propiedades

```
PropertyConfigurator.configure("logger.properties");
```

Error de Llave Primaria

Uno de los errores más comunes es insertar un elemento cuya llave primaria (***PK***) ya se encuentra registrada. Para manejar este error, primero se separará el bloque ***try/catch*** de la conexión, no es necesario hacer esto, pero se recomienda para un manejo de errores más claro. Se creará un método ***try/catch*** para realizar el proceso de inserción.

OperacionesLicenciatura.java

```
private final static Logger log = Logger.getLogger(OperacionesLicenciatura.class);

	public boolean insertarLicenciatura(Licenciatura licenciatura){

		PropertyConfigurator.configure("logger.properties");


		Session sesion = null;
		boolean resultado = false;
		try{

			sesion = CrearConexion.getSessionFactory().openSession();

		}
			catch (Throwable ex) {
		       System.err.println("No se pudo crear la sesion " + ex);
		       throw new ExceptionInInitializerError(ex);
		}

			try{
				log.info("INSERTANDO LICENCIATURA");
				sesion.beginTransaction();
				sesion.save(licenciatura);
				sesion.getTransaction().commit();
				resultado = true;
				log.info("SE INSERTO LA LICENCIATURA");

			}catch(PersistenceException e){
			  log.error("ERROR AL INSERTAR LICENCIATURA");

			 }finally{
			   sesion.close();
		}
		   return resultado;
	}
```

Adicionalmente se regresa un valor *booleano* que servirá como indicador para saber si se realizó la operación de manera correcta o no.

Principal.java

```
public static void insertarLicenciatura(){
		OperacionesLicenciatura operaciones = new OperacionesLicenciatura();
		Licenciatura licenciatura = new Licenciatura();

		licenciatura.setClave("210091");
		licenciatura.setNombre("Nombre de la Licenciatura");
```

```
            boolean resultado =
operaciones.insertarLicenciatura(licenciatura);
            if(resultado)
                  System.out.println("Se insertó la licenciatura");
            else
                  System.out.println("No fue posible insertar la
licenciatura");
      }
```

Se realizará algo similar con la inserción de una UEA

OperacionesUea.java

```
private final static Logger log = Logger.getLogger(OperacionesUea.class);

      public boolean insertarUea(Uea uea){

            PropertyConfigurator.configure("logger.properties");

            Session sesion = null;
            boolean resultado = false;

            try{
                  sesion = CrearConexion.getSessionFactory().openSession();


             }catch (Throwable ex) {
                System.err.println("No se pudo crear la sesion " + ex);
                throw new ExceptionInInitializerError(ex);
             }
            try{
                  log.info("INSERTANDO UEA");
                  sesion.beginTransaction();
                  sesion.save(uea);
                  sesion.getTransaction().commit();
                  resultado = true;
                  log.info("SE INSERTO LA UEA");

            }catch(PersistenceException e){
              log.error("ERROR AL INSERTAR LA UEA");

             }finally{
               sesion.close();
    }
      return resultado;

      }
```

Principal.java

```
public static void insertarUea(){
            OperacionesUea operaciones = new OperacionesUea();
            Uea uea = new Uea();

            uea.setClaveUea("9113");
            uea.setNombre("Nombre de la UEA");
            uea.setCreditos(9);

            boolean resultado = operaciones.insertarUea(uea);

            if(resultado)
                  System.out.println("Se insertó la UEA");
            else
                  System.out.println("No se pudo insertar la UEA");
      }
```

Finalmente se agrega el manejo de *logs* a las clases y métodos faltantes.

OperacionesAlumno.java

```
private final static Logger log =
Logger.getLogger(OperacionesAlumno.class);

      public boolean insertarAlumno(Alumno alumno){

            PropertyConfigurator.configure("logger.properties");

            Session sesion = null;
            boolean resultado = false;

            try{
                  sesion = CrearConexion.getSessionFactory().openSession();

             }catch (Throwable ex) {
                System.err.println("No se pudo crear la sesion " + ex);
                throw new ExceptionInInitializerError(ex);
             }
            try{
                  log.info("INSERTANDO ALUMNO");
                  sesion.beginTransaction();
                  sesion.save(alumno);
               sesion.getTransaction().commit();
            }catch(PersistenceException e){
                  log.error("ERROR AL INSERTAR ALUMNO");

             }
```

```
            finally{
               sesion.close();
            }
            return resultado;

      }
```

Principal.java

```
public static void insertarAlumno(){
            OperacionesAlumno operaciones = new OperacionesAlumno();
            Alumno alumno = new Alumno();

            alumno.setLicenciatura("1100141");
            alumno.setMatricula("345671");
            alumno.setNombre("Nombre Alumno");
            alumno.setPrimerApellido("Primer Apellido");
            alumno.setSegundoApellido("Segundo Apellido");

            boolean resultado = operaciones.insertarAlumno(alumno);
            if(resultado)
                  System.out.println("Se insertó el alumno");
            else
                  System.out.println("No se pudo insertar el alumno");
      }
```

OperacionesAlumnoUea.java

```
private          final          static          Logger          log          =
Logger.getLogger(OperacionesAlumnoUea.class);

      public boolean insertarAlumnoUea(AlumnoUea alumnoUea){

            PropertyConfigurator.configure("logger.properties");

            Session sesion = null;
            boolean resultado = false;

            try{
                  sesion = CrearConexion.getSessionFactory().openSession();

             }catch (Throwable ex) {
                System.err.println("No se pudo crear la sesion " + ex);
                throw new ExceptionInInitializerError(ex);
             }

            try{
                  log.info("INSERTANDO RELACION ALUMNO UEA");
```

```
                sesion.beginTransaction();
                sesion.save(alumnoUea);
                 sesion.getTransaction().commit();
               sesion.close();
                log.info("SE INSERTO LA RELACION ALUMNO UEA");
                resultado = true;

            }catch(PersistenceException e){
                log.error("ERROR AL INSERTAR LA RELACION ALUMNO UEA");

            }

            finally{
               sesion.close();
            }
            return resultado;

    }
```

Principal.java

```
public static void relacionAlumnoUea(){

            OperacionesAlumnoUea operaciones = new OperacionesAlumnoUea();
            AlumnoUea alumnoUea = new AlumnoUea();

            alumnoUea.setAlumnoMatricula("23456");
            alumnoUea.setCalificacion("MB");
            alumnoUea.setCalificacionNumerica(9.5F);
            alumnoUea.setTrimestre("16-P");
            alumnoUea.setUeaClaveUea("1111");

            boolean resultado = operaciones.insertarAlumnoUea(alumnoUea);

            if(resultado)
                  System.out.println("Se registró la relación");
            else
                  System.out.println("No se pudo registrar la relación");


    }
```