

Cardinalidad Hibernate

Adjunto el proyecto con el código.

Alta de cantantes

```
private static void altaCantante(Scanner scanner) {
    Session session = sessionFactory.openSession();
    Transaction transaction = session.beginTransaction();

    System.out.println("\nALTA DE CANTANTE");
    Cantante cantante = new Cantante();

    System.out.print("Ingrese el nombre del cantante: ");
    cantante.setNombre(scanner.nextLine());

    System.out.print("Ingrese el año: ");
    cantante.setAnio(scanner.nextInt());
    scanner.nextLine();

    System.out.print("Ingrese el nombre del álbum: ");
    cantante.setAlbum(scanner.nextLine());

    session.save(cantante);
    transaction.commit();
    session.close();
    System.out.println("Cantante añadido con éxito.");
}
```

Listado de cantantes

```
private static void listadoDeCantantes() {
    Session session = sessionFactory.openSession();
    List<Cantante> cantantes = session.createQuery("from Cantante",
        Cantante.class).list();

    System.out.println("Listado de todos los cantantes:");
    for (Cantante cantante : cantantes) {
        System.out.println("Código: " + cantante.getCodigo() +
            ", Nombre: " + cantante.getNombre() +
            ", Género: " + cantante.getGenero().getNombre());
    }

    session.close();
}
```

Consulta por género

```
private static void consultaPorGenero(Scanner input)
{
    Session session = sessionFactory.openSession();

    System.out.print("ID del género musical: ");
    int generoId = input.nextInt();

    GeneroMusical genero = session.get(GeneroMusical.class, generoId);
    if (genero == null)
    {
        System.out.println("Género no encontrado.");
        session.close();
        return;
    }

    List<Cantante> cantantes = genero.getCantantes();
    System.out.println("Lista de cantantes para el género " +
genero.getNombre() + ":");
    for (Cantante cantante : cantantes)
    {
        System.out.println(
            "Código: " + cantante.getCodigo() +
            ", Nombre: " + cantante.getNombre() +
            ", Año: " + cantante.getAnio() +
            ", Álbum: " + cantante.getAlbum());
    }
    session.close();
}
```

hibernate.cfg.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC "-//Hibernate/Hibernate Configuration DTD
3.0//EN" "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">

<hibernate-configuration>
    <session-factory>
        <property name="hibernate.connection.driver_class">com.mysql.cj.jdbc.Driver</
property>
        <property name="hibernate.connection.url">jdbc:mysql://localhost:3306/musica</
property>
        <property name="hibernate.connection.username">examen</property>
        <property name="hibernate.connection.password">Examen2024</property>
        <property name="hibernate.show_sql">>false</property>
        <property name="hibernate.format_sql">>true</property>
        <property name="hibernate.hbm2ddl.auto">none</property>
        <property name="hibernate.connection.pool_size">10</property>
        <mapping class="hibernate.GeneroMusical" />
        <mapping class="hibernate.Cantante" />
    </session-factory>
</hibernate-configuration>
```

```
</session-factory>
</hibernate-configuration>
```

Clase Cantante

```
import jakarta.persistence.*;

@Entity
@Table(name = "cantantes")
public class Cantante {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int codigo;

    @Column(name = "nombre", nullable = false)
    private String nombre;

    @Column(name = "anio", nullable = false)
    private int anio;

    @Column(name = "album", nullable = false)
    private String album;

    @ManyToOne
    @JoinColumn(name = "genero_id", nullable = false)
    private GeneroMusical genero;

    // Getters y Setters
    public int getCodigo() {
        return codigo;
    }

    public void setCodigo(int codigo) {
        this.codigo = codigo;
    }

    public String getNombre() {
        return nombre;
    }

    public void setNombre(String nombre) {
        this.nombre = nombre;
    }

    public int getAnio() {
        return anio;
    }

    public void setAnio(int anio) {
        this.anio = anio;
    }
}
```

```

    public String getAlbum() {
        return album;
    }

    public void setAlbum(String album) {
        this.album = album;
    }

    public GeneroMusical getGenero() {
        return genero;
    }

    public void setGenero(GeneroMusical genero) {
        this.genero = genero;
    }
}

```

Capturas de consola y BD

BD al inicio:

Cantantes

	codigo	nombre	anio	album	genero_id
*	NULL	NULL	NULL	NULL	NULL

Géneros musicales

	id	nombre
▶	1	Pop
	2	Rock
	3	Flamenco
	4	Clasica
*	NULL	NULL

Registro de un cantante:



```

=== MENÚ PRINCIPAL ===
1. Alta de cantantes
2. Consulta por género musical
3. Listado de todos los cantantes
4. Salir
Elija una opción: 1

ALTA DE CANTANTE
Ingrese el nombre del cantante: Madonna
Ingrese el año: 1989
Ingrese el nombre del álbum: Like a Prayer
ID del género musical: 1
Cantante añadido con éxito.

```

BD actualizada:

Result Grid			Filter Rows:	<input type="text"/>	Edit:
	codigo	nombre	anio	album	genero_id
▶	1	Madonna	1989	Like a Prayer	1
*	NULL	NULL	NULL	NULL	NULL

Consulta por género musical:

```
=== MENÚ PRINCIPAL ===
1. Alta de cantantes
2. Consulta por género musical
3. Listado de todos los cantantes
4. Salir
Elija una opción: 2
ID del género musical: 1
Lista de cantantes para el género Pop:
Código: 1, Nombre: Madonna, Año: 1989, Álbum: Like a Prayer
```

Listado de los cantantes:

```
=== MENÚ PRINCIPAL ===
1. Alta de cantantes
2. Consulta por género musical
3. Listado de todos los cantantes
4. Salir
Elija una opción: 3
Listado de todos los cantantes:
Código: 1, Nombre: Madonna, Género: Pop
```