

## Consultas HQL Hibernate

Cuando creamos un proyecto maven nos crea en Project Files el archivo pom.xml donde tenemos que indicar las dependencias, estas se encuentran en la página <https://mvnrepository.com/>

Buscando hibernate, mysql, persistence, etc y en maven nos dá el código que tendríamos que poner

**Hibernate Core Relocation » 6.0.0.Alpha4**  
Hibernate's core ORM functionality

License	LGPL 2.1
Categories	Object/Relational Mapping
Tags	persistence mapping orm hibernate relational
Organization	Hibernate.org
HomePage	<a href="http://hibernate.org/orm">http://hibernate.org/orm</a>
Date	Dec 21, 2019
Files	<a href="#">View All</a>
Repositories	Central
Ranking	#115 in MvnRepository (See Top Artifacts) #1 in Object/Relational Mapping
Used By	4,198 artifacts

**Note:** This artifact was moved to:  
[org.hibernate.org](http://org.hibernate.org) » [hibernate-core](http://hibernate-core) » 6.0.0.Alpha4

**Note:** There is a **New Version** available: [6.4.1.Final](#)

Maven [Gradle](#) [Gradle \(Short\)](#) [Gradle \(Kotlin\)](#) [SBT](#) [Ivy](#) [Grape](#) [Leiningen](#) [Buildr](#)

```
<!-- https://mvnrepository.com/artifact/org.hibernate/hibernate-core -->
<dependency>
  <groupId>org.hibernate</groupId>
  <artifactId>hibernate-core</artifactId>
  <version>6.0.0.Alpha4</version>
  <type>pom</type>
</dependency>
```

**ProjectoPruebaHibernate - Apache NetBeans IDE 12.0**

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Design Source History

1 <?xml version="1.0" encoding="UTF-8"?>  
2 <persistence version="2.1" xmlns="http://xmlns.jcp.org/xmlns/persistence" xmlns:xs="http://www.w3.org/2001/XMLSchema-instance">  
3 <persistence-unit name="projectoPrueba" transaction-type="RESOURCE\_LOCAL">  
4 <provider>org.eclipse.persistence.jpa.PersistenceProvider</provider>  
5 <properties>  
6 <property name="javax.persistence.jdbc.url" value="jdbc:mysql://localhost:3307/usuarios?serverTimezone=UTC"/>  
7 <property name="javax.persistence.jdbc.user" value="root"/>  
8 <property name="javax.persistence.jdbc.driver" value="com.mysql.cj.jdbc.Driver"/>  
9 <property name="javax.persistence.jdbc.password" value="" />  
10 <property name="javax.persistence.schema-generation.database.action" value="create"/>  
11 </properties>  
12 </persistence-unit>  
13 </persistence>

**New File**

Steps  
1. Choose File Type  
2. ...

Choose File Type

Project: ProyectoPruebaHibernate

Q Filter:

Categories:

- Java
- Swing GUI Forms
- JavaBeans Objects
- AWT GUI Forms
- Unit Tests
- Selenium Tests
- Persistence
- Groovy
- Web Services
- ...

File Types:

- Entity Class
- Entity Classes from Database
- JPA Controller Classes from Entity Classes
- Persistence Unit
- DB Scripts from Entity Classes
- Database Schema

Description:  
Creates an JSR220/JSR317/JSR338 persistence unit. If persistence.xml doesn't exist, it is created.

< Back Next > Finish Cancel Help

org.hibernate.jpa.HibernatePersistenceProvider

tenemos que crear las entidades, una por cada tabla de la base de datos teniendo en cuenta los tipos de datos, las primary key y las relaciones si hay más de una tabla

```
@Entity
@Table (name="empleados")
public class Empleado implements Serializable {

    private static final long serialVersionUID = 1L;

    @Id
    @Column (name = "emp_no")
    private Short emp_no;

    @Column (name = "apellido")
    private String apellido;

    @Column (name = "comision")
    private Float comision;

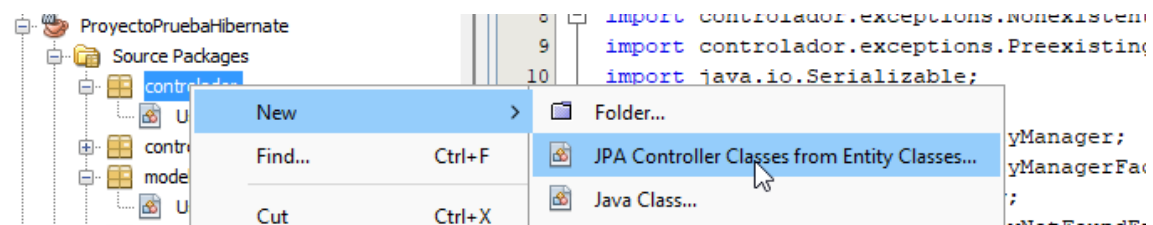
    @ManyToOne
    @JoinColumn (name = "dept_no")
    private Departamento dept_no;
```

Nombre de la tabla en la BD

Esto siempre hay que ponerlo

Luego creamos varios constructores, los getter y setter y también en insert nos da la opción de añadir el código de unos métodos que se llaman hashCode y equals. Los añadimos

Una vez terminada la Entidad podemos crear el controlador



Aquí añadiríamos los métodos con el código HQL con las consultas a la base de datos. Igualmente nos da algunos métodos creados con los que se pueden hacer casi todas las operaciones más usuales

```
public class UsuarioJpaController implements Serializable {

    public UsuarioJpaController(EntityManagerFactory emf) {
        this.emf = emf;
    }

    public UsuarioJpaController () {
        emf = Persistence.createEntityManagerFactory("usuarios");
    }
}
```

Tenemos que añadir este constructor sin parámetros, el segundo.

En las clases en las que vamos a necesitar acceso a la base de datos creamos un objeto del controlador con el constructor que hemos creado antes

```
public class ControladorMenu implements ActionListener{

    private MenuPrincipal vistaMenu;
    private UsuarioJpaController usuarioJpaController;

    public ControladorMenu(MenuPrincipal menu) {
        this.vistaMenu = menu;
        usuarioJpaController = new UsuarioJpaController();

        this.vistaMenu.btn_add.addActionListener(this);
    }
}
```