

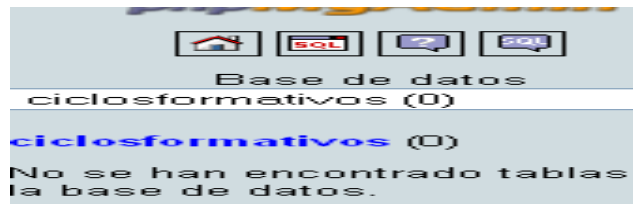
Práctica 3. Crear una Entidad persistente con MySql.

La práctica consiste en crear una entidad llamada módulo y persistirla en la base de datos como una tabla llamada modulo.

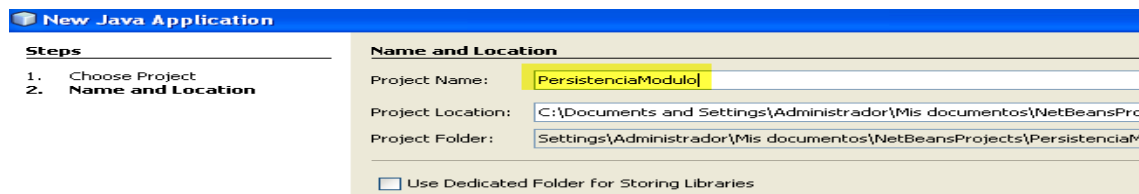
Pasos previos:

Creamos la base de datos ciclosformativos, de momento sin

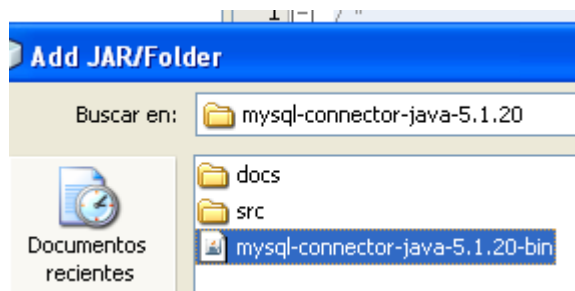
tablas, usuario root sin pw.



Crea un proyecto llamado PersistenciaModulo

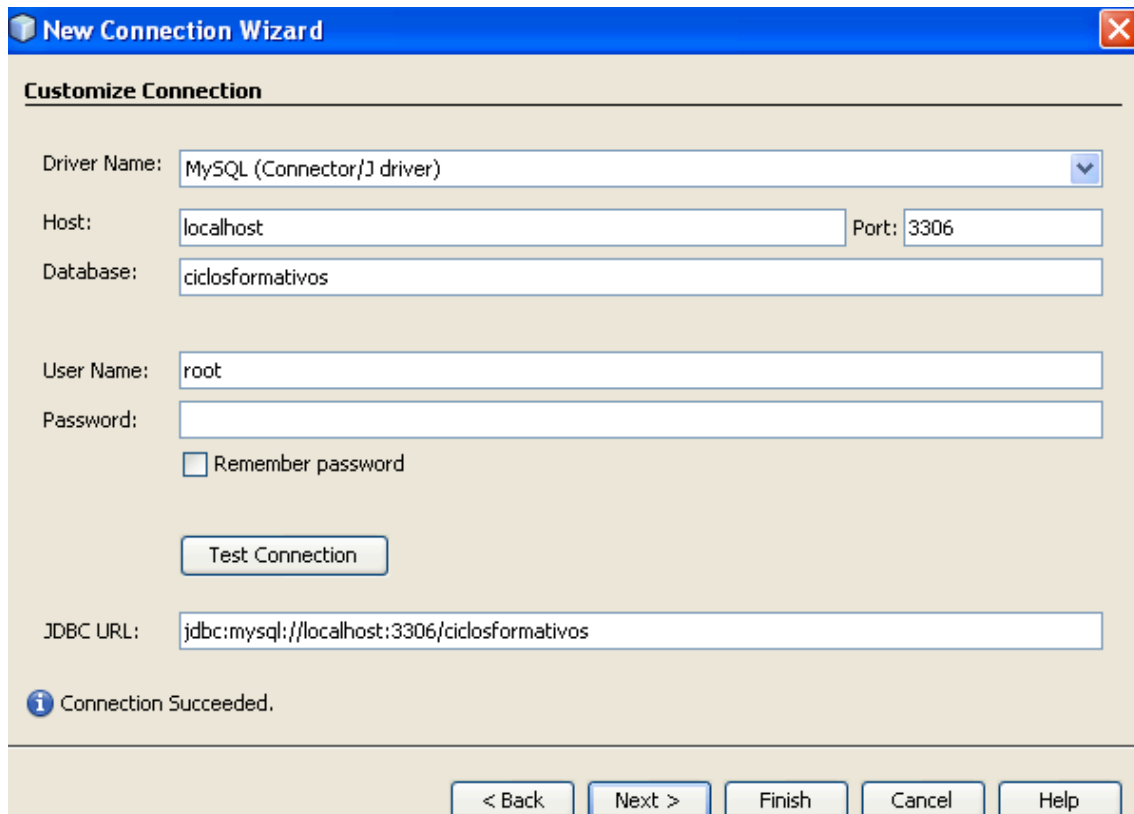


Añade el .jar de conexión a mysql



Conectate como root sin pw a mysql como esquema la base de datos ciclosformativos





New Connection Wizard

Customize Connection

Driver Name: MySQL (Connector/J driver)

Host: localhost Port: 3306

Database: ciclosformativos


User Name: root

Password:

☐ Remember password

Test Connection

JDBC URL: jdbc:mysql://localhost:3306/ciclosformativos

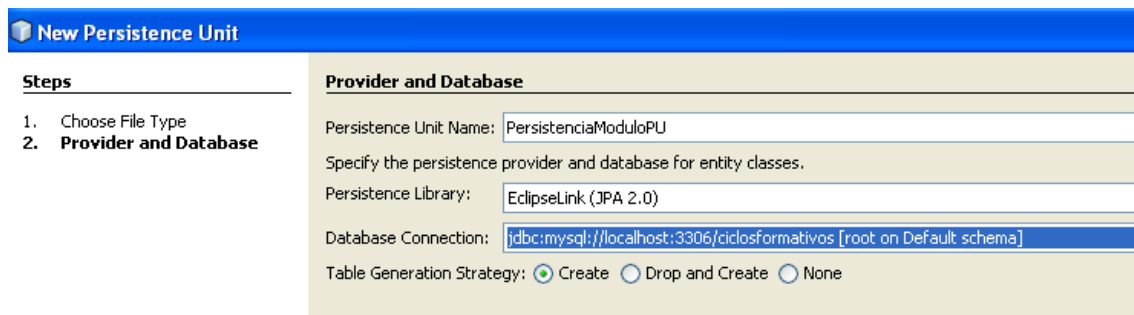
 Connection Succeeded.

< Back Next > Finish Cancel Help

PASOS:

1. Crear unidad de persistencia
 2. Crear una entidad persistente
 3. crear un jpa Controller
 4. crear una clase aplicación
-

1. Crear unidad de persistencia



New Persistence Unit

Steps

1. Choose File Type
2. **Provider and Database**

Provider and Database

Persistence Unit Name: PersistenciaModuloPU

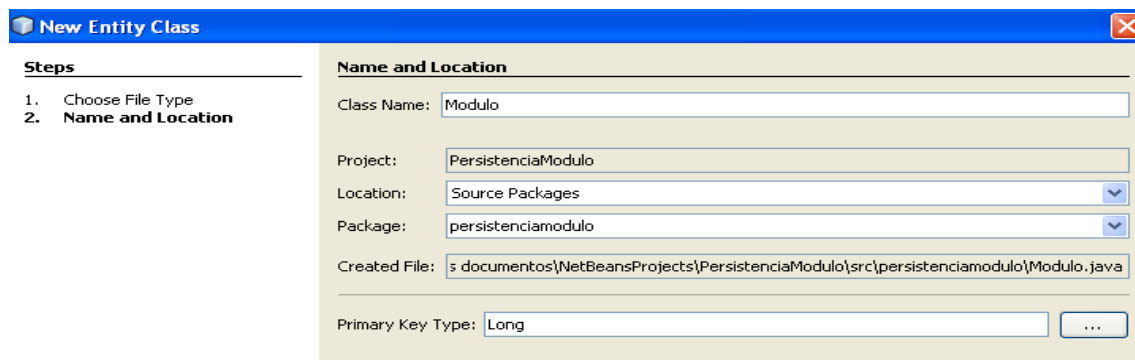
Specify the persistence provider and database for entity classes.

Persistence Library: EclipseLink (JPA 2.0)

Database Connection: jdbc:mysql://localhost:3306/ciclosformativos [root on Default schema]

Table Generation Strategy: ☒ Create ☐ Drop and Create ☐ None

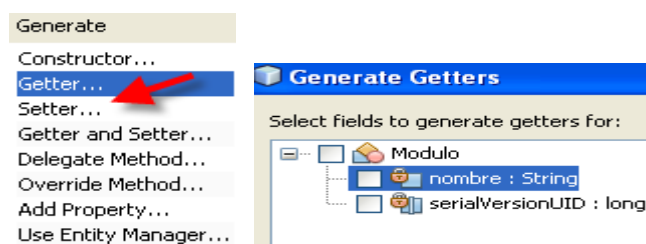
2. Crear una entidad persistente



Completo el código y añadido el atributo `private String nombre` y sus métodos get and set

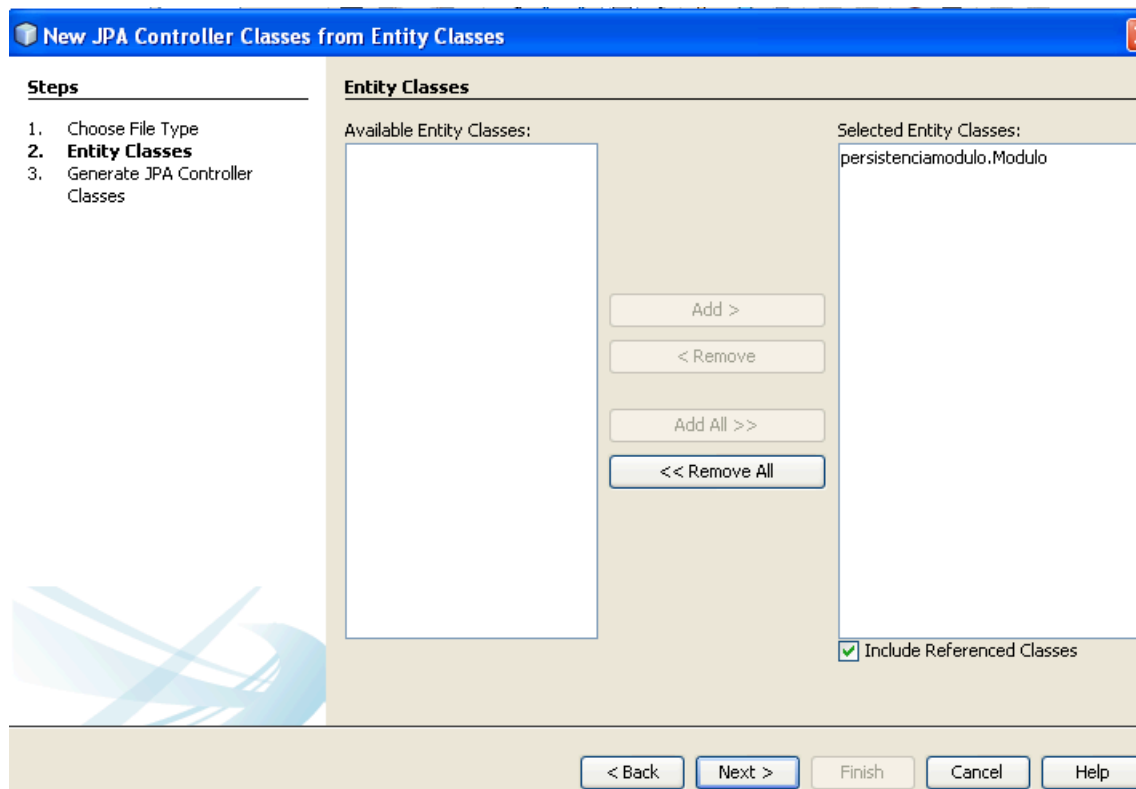
```
public String getNombre() {  
  
    return this.nombre;  
  
}  
  
public void setNombre(final String nombre) {  
  
    this.nombre = nombre;  
  
}
```

Lo podemos escribir manualmente o bien sólo crear el atributo nombre y con el botón derecho ratón seleccionar insert code ☐ getter and setter



```
@Entity  
public class Modulo implements Serializable {  
    private static final long serialVersionUID = 1L;  
    @Id  
    @GeneratedValue(strategy = GenerationType.AUTO)  
    private Long id;  
    private String nombre;  
  
    public Long getId() {  
        return id;  
    }  
  
    public void setId(Long id) {  
        this.id = id;  
    }  
  
    public String getNombre() {  
        return nombre;  
    }  
  
    public void setNombre(String nombre) {  
        this.nombre = nombre;  
    }  
}
```

3. crear un jpa Controller



4. crear un clase aplicación

Antes de crear la aplicación comprobar el fichero persistence.xml que contenga los nombres de los .java, en este caso modulo.java

```
<?xml version="1.0" encoding="UTF-8"?>
<persistence version="2.0" xmlns="http://java.sun.com/xml/ns/persistence" xmlns:xsi
  <persistence-unit name="PersistenciaModuloPU" transaction-type="RESOURCE_LOCAL">
    <class>Modulo.java</class>
    <provider>org.eclipse.persistence.jpa.PersistenceProvider</provider>
    <class>persistenciamodulo.Modulo</class>
    <properties>
      <property name="javax.persistence.jdbc.url" value="jdbc:mysql://localhost:330
      <property name="javax.persistence.jdbc.password" value=""/>
      <property name="javax.persistence.jdbc.driver" value="com.mysql.jdbc.Driver"/>
      <property name="javax.persistence.jdbc.user" value="root"/>
      <property name="eclipselink.ddl-generation" value="create-tables"/>
    </properties>
  </persistence-unit>
</persistence>
```

Persistir la entidad modulo en una tabla, e insertar un modulo llamado bases de datos.

```

import javax.persistence.Persistence;
public class PersistenciaModulo {

    public static void main(String[] args) {
        final Modulo modulo = new Modulo();
        modulo.setNombre("BASES DE DATOS");
        EntityManagerFactory emf = null;
        EntityManager em = null;
        try {
            emf = Persistence.createEntityManagerFactory("PersistenciaModuloPU");
            //nombre de la unidad de persistencia
            em = emf.createEntityManager();
            em.getTransaction().begin();
            em.persist(modulo);
            em.getTransaction().commit();
        } catch (final Exception e) {
            if (em != null) {
                em.getTransaction().rollback();
            }
        } finally {
            if (em != null) {
                em.close();
            }
            if (emf != null) {
                emf.close();
            }
        }
    }
}

```

Probar que se ha creado la tabla, consultando en phpmyadmin

The screenshot shows the phpMyAdmin interface. On the left, the database 'ciclosformativos (2)' is selected, and the 'modulo' table is highlighted. The main area displays the table structure and data. The table has two columns: 'ID' and 'NOMBRE'. There is one record with ID 1 and NOMBRE 'BASES DE DATOS'. A red arrow points to the 'BASES DE DATOS' value in the table. Below the table, there are options to 'Mostrar' (show) 30 rows, and buttons for 'Vista de impresión', 'Previsualización para imprimir', 'Exportar', and 'CREATE VIEW'.