

Realizar una aplicación con persistencia A BASES DE DATOS

Programación Web Avanzada Bonilla Hidalgo Jairo Smith Página 3 de 11

Unidad: 3

REALIZAR UNA APLICACIÓN CON PERSISTENCIA A BASES DE DATOS

INTRODUCCION

En el presente trabajo se desarrolla una aplicación Java con persistencia de datos utilizando Hibernate como framework ORM (Object Relational Mapping) y MySQL como sistema de gestión de base de datos relacional. El proyecto fue implementado en IntelliJ IDEA y tiene como propósito demostrar el ciclo completo de operaciones CRUD (Crear, Leer, Actualizar y Eliminar) sobre una tabla denominada productos, que contiene los campos id, nombre, fecha y precio. Se parte de la creación del proyecto y la conexión nativa a MySQL para la consulta inicial de datos, avanzando posteriormente hacia la integración con Hibernate para realizar consultas más avanzadas, filtradas y parametrizadas. Además, se incluyen servicios para centralizar la lógica de negocio y se presentan capturas de la funcionalidad y el código fuente para su documentación.

OBJETIVOS

Objetivo general:

Desarrollar una aplicación Java con persistencia en MySQL utilizando Hibernate, que permita gestionar la tabla **productos** mediante operaciones CRUD y consultas personalizadas, aplicando buenas prácticas de programación y organización de código.

Objetivos específicos:

- Configurar un proyecto en IntelliJ IDEA que integre Hibernate con MySQL, asegurando una conexión correcta a la base de datos.
- Implementar operaciones CRUD y consultas personalizadas sobre la tabla productos utilizando Hibernate, incluyendo filtrado por ID y condiciones específicas.
- **Documentar** el desarrollo del proyecto con capturas de pantalla y código fuente, generando un informe final en PDF que evidencie la funcionalidad implementada.



Realizar una aplicación con persistencia A BASES DE DATOS

Página 4 de 11

Unidad:

NRC: 23275

Programación Web Avanzada Bonilla Hidalgo Jairo Smith

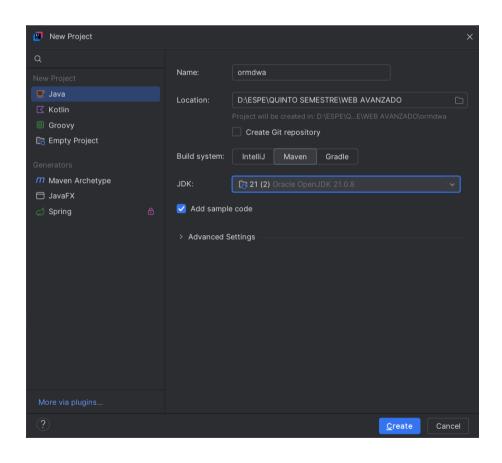
DESARROLLO

Creacion del proyecto

Para dar inicio al desarrollo, se procedió a crear un nuevo proyecto en IntelliJ IDEA utilizando el lenguaje Java. Durante la configuración inicial, se optó por Maven como gestor de dependencias, lo que permitió simplificar la incorporación de Hibernate y el conector de MySQL a través del archivo pom.xml.

Los pasos esenciales fueron los siguientes:

- Abrir IntelliJ IDEA y acceder a la opción File > New > Project.
- Seleccionar Maven como tipo de proyecto e indicar la versión de Java a utilizar.
- Asignar un nombre al proyecto y definir la ubicación de la carpeta de trabajo.



Crear la tabla productos (id, nombre, fecha, precio)

En **mysql** se generó la tabla **productos** con los siguientes campos:



Realizar una aplicación con persistencia A BASES DE		
DATOS	Página 5 de 11	
Programación Web Avanzada	Unidad:	3
Bonilla Hidalgo Jairo Smith	NRC:	23275

- id → entero, clave primaria, autoincremental.
- nombre → texto (VARCHAR 200).
- fecha → tipo DATE.
- precio → tipo DECIMAL(10,0)

```
mysql> desc productos;
         | int
                         I NO
                                 | PRI | NULL
                                                 | auto_increment
 nombre | varchar(255)
                           NO
                                         NULL
 precio | decimal(10,2) |
                           NO
                                         NULL
  fecha
         date
                           NO
                                         NULL
 rows in set (0,00 sec)
```

En esta etapa se realizó el acceso directo a la base de datos empleando JDBC. La aplicación abrió una conexión con MySQL, ejecutó una sentencia SELECT * FROM productos y recuperó la información almacenada. Posteriormente, se imprimió en la consola el valor correspondiente a la columna nombre para cada uno de los registros obtenidos.

En esta fase se empleó Hibernate para recuperar la totalidad de los registros presentes en la tabla productos utilizando una consulta JPQL.

La aplicación estableció la conexión con la base de datos a través de un EntityManager, ejecutó la sentencia SELECT c FROM Product c y posteriormente iteró sobre la colección de objetos Product para mostrar su información en la consola.



Realizar una aplicación con persistencia A BASES DE DATOS Página 6 de 11 Programación Web Avanzada Unidad: 3 Bonilla Hidalgo Jairo Smith NRC: 23275

```
| Greater Product | Servicios Main | Decision | Decisio
```

Listar con hibernate pe: hibernatelistarid

En este paso se utilizó Hibernate para buscar un producto específico según su ID. El programa solicita al usuario que ingrese un identificador, utiliza el método em.find() para obtener el registro correspondiente y lo muestra por consola.



Realizar una aplicación con persistencia A BASES DE DATOS Programación Web Avanzada

Bonilla Hidalgo Jairo Smith

Página 7 de 11

Unidad:

NRC:

3 23275

```
■ HbProductFind ×

ago 15, 2025 12:05:19 A.M. org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
INFO: HHH009490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
Enter product id: 1
Hibernate: select product0_.id as id1_1_0_, product0_.fecha as fecha2_1_0_, product0_.nombre as nombre3_1_0_, product0_.pro
Product{id=1, nombre='Laptop', fecha=2025-07-18 00:00:00.0, precio=800.0}

Process finished with exit code 0
```

En esta etapa se utilizó Hibernate para ejecutar una consulta JPQL con cláusula WHERE, destinada a obtener únicamente los productos cuyo nombre coincida con un criterio específico. La aplicación solicita al usuario el nombre a buscar, incorpora dicho valor como parámetro en la consulta y, finalmente, presenta en la consola los registros encontrados.

```
■ HbProductWhere ×

ago 15, 2025 12:25:59 A.M. org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.inter
Enter the product name: Laptop
Hibernate: select product0_.id as id1_1_, product0_.fecha as fecha2_1_, product0_.nombre as nombre3_1_,
[Product{id=1, nombre='Laptop', fecha=2025-07-18 00:00:00.0, precio=800.0}]

Process finished with exit code 0
```

En esta fase se implementó la funcionalidad de creación de registros en la tabla productos utilizando

Hibernate.

La aplicación solicita al usuario la información del nuevo producto nombre, fecha y precio, realiza la conversión de la fecha y el precio a los tipos de datos adecuados y, posteriormente, guarda el objeto **Product** en la base de datos dentro de una transacción controlada.



Realizar una aplicación con persistencia A BASES DE DATOS Programación Web Avanzada

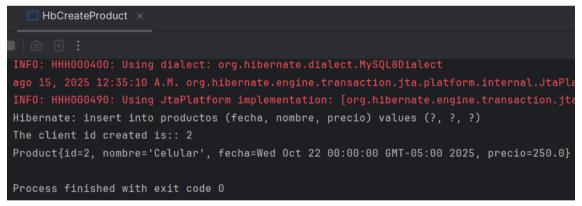
Bonilla Hidalgo Jairo Smith

Página 8 de 11

Unidad: NRC: 23275

```
public static void main(String[] args) {
       EntityManager em = JpaUtil.getEntityManager();
       try {
          String name = JOptionPane.showInputDialog("name: ");
          String dateStr = JOptionPane.showInputDialog("Date (dd/MM/yyyy):");
           SimpleDateFormat sdf = new SimpleDateFormat( pattern: "dd/MM/yyyy");
          Date date = null:
           try {
              date = sdf.parse(dateStr);
           } catch (ParseException e) {
              JOptionPane.showMessageDialog( parentComponent: null, message: "Incorrect date format. Use dd/MM/yyyy");
           String priceStr = JOptionPane.showInputDialog("price: ");
           float price = 0f;
           try {
              price = Float.parseFloat(priceStr);
            catch (NumberFormatException e) {
               JOptionPane.showMessageDialog( parentComponent null, message: "Invalid price. Enter a valid number.");
```





En esta etapa se implementó la actualización de un registro existente en la tabla productos utilizando

La aplicación solicita al usuario el ID del producto que desea modificar, obtiene la información actual del registro y ofrece la posibilidad de cambiar sus valores a través de cuadros de diálogo. En caso de que algún campo no sea modificado, se mantiene el dato original. Finalmente, los cambios se aplican en la base de datos empleando el método merge() dentro de una transacción.



Realizar una aplicación con persistencia A BASES DE DATOS

Página 9 de 11

Programación Web Avanzada Bonilla Hidalgo Jairo Smith Unidad: NRC:

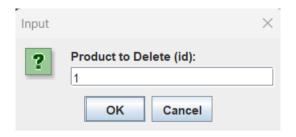
23275

En esta fase se desarrolló la operación de eliminación de registros en la tabla productos utilizando Hibernate.

La aplicación solicita al usuario el ID del producto que desea suprimir, localiza el registro correspondiente en la base de datos y, en caso de existir, procede a eliminarlo empleando el método remove() dentro de una transacción.



Realizar una aplicación con persistencia A BASES DE DATOS Página 10 de 11 Programación Web Avanzada Unidad: 3 Bonilla Hidalgo Jairo Smith NRC: 23275



Verificamos en la base de datos, y se elimino el producto con ID 1

```
mysql> select * from productos;
  id
                 | precio |
                            fecha
       nombre
   2
       Leche
                     1.50 | 2025-07-25
   3
       Pan
                           2025-07-20
                     0.75
                     2.30 | 2025-08-01
   4
       Queso
       Producto
                   12.00 | 2025-12-12
 rows in set (0,00 sec)
```

En esta etapa se implementó la clase **ServiciosMain**, concebida como el punto de entrada principal para gestionar la comunicación con la base de datos mediante **Hibernate**.

```
package org.espe;

public class Main {
    public static void main(String[] args) {
        System.out.println("Hello and welcome!");
    }
}
```

CONCLUSIONES

 Se comprobó que Hibernate facilita la gestión de la persistencia de datos en Java, permitiendo realizar operaciones CRUD de manera eficiente y reduciendo la complejidad de las consultas SQL nativas.



Realizar una aplicación con persistencia A BASES DE DATOS

Página 11 de 11 Programación Web Avanzada

Unidad: Bonilla Hidalgo Jairo Smith NRC:

- 2. El uso de JPQL y consultas parametrizadas demostró la capacidad de Hibernate para realizar búsquedas dinámicas y filtradas, mejorando la interacción con la base de datos frente a consultas nativas.
- 3. La implementación de la clase ServiciosMain permitió centralizar las operaciones sobre la base de datos, promoviendo un diseño más limpio y facilitando futuras ampliaciones del provecto.
- 4. La práctica permitió entender de forma práctica cómo crear, leer, actualizar y eliminar registros en la base de datos usando Hibernate, así como manejar transacciones y parámetros de manera segura.

PARA MI

Con los conocimientos adquiridos en esta práctica, puedo aplicar en el futuro la creación de aplicaciones Java que requieran persistencia de datos de manera eficiente y segura. Seré capaz de diseñar sistemas que gestionen bases de datos relacionales mediante Hibernate, implementando operaciones CRUD, consultas parametrizadas y transacciones controladas.

BIBLIOGRAFIA

- Lesson: JDBC Basics (The JavaTM Tutorials > JDBC Database Access). (n.d.). https://docs.oracle.com/javase/tutorial/jdbc/basics/
- The leading IDE for professional Java and Kotlin development. (2021, June 1). JetBrains. https://www.jetbrains.com/idea/#
- Documentation 7.1 Hibernate ORM. (n.d.). Hibernate. https://hibernate.org/orm/documentation/7.1/

borado por:	Revisador por:
 nbre: Jairo Smith Bonilla Hidalgo ha:15/08/2025	Nombre: