

# PRACTICA 3.3. HQL.CONSULTAS

**OBJETIVO:** Crear aplicaciones que usen consultal HQL.

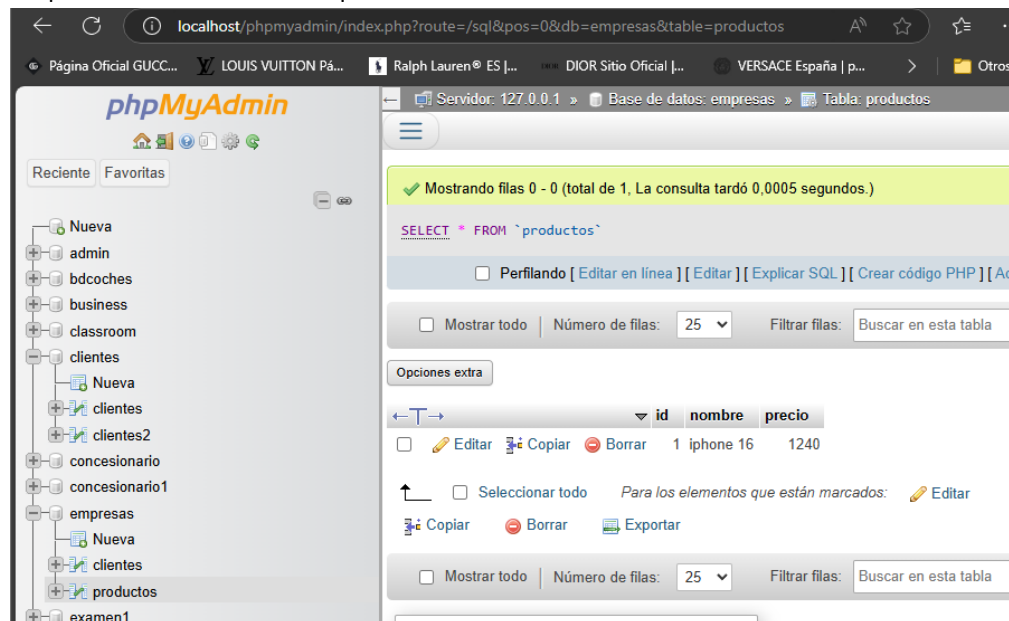
PASO 2: CAMBIANDO LA CONSULTA HQL, HAZ MÉTODOS PARA :

- void mostrarPorNombre(String texto): muestra todos los productos que el texto en el nombre
- void productosOrdenadosPorPrecio(): muestra todos los productos ordenados por precios (ascendente)
- void precioDe(String nombre): Muestra el precio de todos los productos que tienen ese nombre
- void buscaProducto(int id): Usando el método uniqueResult, buscar el producto con la id y mostrar toda su información.

Todos los métodos deben tener try/catch en caso de fallo, si no hacen lo que se pide porque no existe, deben avisar con un mensaje.

IMPORTANTE: REALIZA UN WORD EN QUE SE VEA:

1. Un pantallazo con los valores que tienes en la tabla PRODUCTOS



## 2. El código de cada método:

```
package martinmd.martinmd;

import java.util.Iterator;
import java.util.List;
import org.hibernate.HibernateException;
import org.hibernate.ObjectNotFoundException;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;
import org.hibernate.cfg.Configuration;

public class HibernateEnterprise {

    private static SessionFactory sf;

    // Constructor que inicializa la SessionFactory
    HibernateEnterprise() {
        sf = new Configuration().configure().buildSessionFactory(); // Creación de la SessionFactory
    }

    public void close() {
        sf.close(); // Cierre de la SessionFactory
    }

    // Método para mostrar todos los productos
    public void showProducts() {
        Session session = sf.openSession();
        Transaction tx = null;

        try {
            tx = session.beginTransaction();
            List<Productos> allproducts = session.createQuery("FROM Productos",
Productos.class).list();

            System.out.println("=====");
            System.out.println("Buscando Productos...");
            System.out.println("=====");

            for (Productos p : allproducts) {
                System.out.println("=====");
                System.out.println("Id: " + p.getId());
                System.out.println("Nombre: " + p.getNombre());
                System.out.println("Precio: " + p.getPrecio());
                System.out.println("=====");
            }
            tx.commit();
            System.out.println("=====");
        }
    }
}
```

```

        System.out.println("Finalizada la Búsqueda...");
        System.out.println("=====");
    } catch (HibernateException e) {
        if (tx != null) {
            tx.rollback();
        }
        e.printStackTrace();
    } finally {
        session.close();
    }
}

// Método para mostrar productos por nombre
public void mostrarPorNombre(String texto) {
    Session session = sf.openSession();
    Transaction tx = null;

    try {
        tx = session.beginTransaction();
        List<Productos> productosPorNombre = session.createQuery("FROM Productos WHERE nombre LIKE :nombre", Productos.class)
            .setParameter("nombre", "%" + texto + "%").list();

        if (!productosPorNombre.isEmpty()) {
            System.out.println("=====");
            System.out.println("Buscando productos con el nombre: " + texto);
            System.out.println("=====");

            for (Productos p : productosPorNombre) {
                System.out.println("Id: " + p.getId());
                System.out.println("Nombre: " + p.getNombre());
                System.out.println("Precio: " + p.getPrecio());
                System.out.println("=====");
            }
        } else {
            System.out.println("No se encontraron productos con el nombre: " + texto);
        }
        tx.commit();
    } catch (HibernateException e) {
        if (tx != null) {
            tx.rollback();
        }
        e.printStackTrace();
    } finally {
        session.close();
    }
}

```

// Método para mostrar productos ordenados por precio (ascendente)

```

public void productosOrdenadosPorPrecio() {
    Session session = sf.openSession();
    Transaction tx = null;

    try {
        tx = session.beginTransaction();
        List<Productos> productosOrdenados = session.createQuery("FROM Productos ORDER BY
precio ASC", Productos.class).list();

        if (!productosOrdenados.isEmpty()) {
            System.out.println("=====");
            System.out.println("Productos ordenados por precio:");
            System.out.println("=====");

            for (Productos p : productosOrdenados) {
                System.out.println("Id: " + p.getId());
                System.out.println("Nombre: " + p.getNombre());
                System.out.println("Precio: " + p.getPrecio());
                System.out.println("=====");
            }
        } else {
            System.out.println("No se encontraron productos.");
        }
        tx.commit();
    } catch (HibernateException e) {
        if (tx != null) {
            tx.rollback();
        }
        e.printStackTrace();
    } finally {
        session.close();
    }
}

```

```

// Método para mostrar el precio de productos con un nombre específico
public void precioDe(String nombre) {
    Session session = sf.openSession();
    Transaction tx = null;

    try {
        tx = session.beginTransaction();
        List<Productos> productosConNombre = session.createQuery("FROM Productos WHERE
nombre = :nombre", Productos.class)
            .setParameter("nombre", nombre).list();

        if (!productosConNombre.isEmpty()) {
            System.out.println("=====");
            System.out.println("Buscando el precio de los productos con nombre: " + nombre);
            System.out.println("=====");
        }
    }
}

```

```

        for (Productos p : productosConNombre) {
            System.out.println("Nombre: " + p.getNombre());
            System.out.println("Precio: " + p.getPrecio());
            System.out.println("=====");
        }
    } else {
        System.out.println("No se encontraron productos con el nombre: " + nombre);
    }
    tx.commit();
} catch (HibernateException e) {
    if (tx != null) {
        tx.rollback();
    }
    e.printStackTrace();
} finally {
    session.close();
}
}

// Método para buscar un producto por ID usando uniqueResult
public void buscaProducto(int id) {
    Session session = sf.openSession();
    Transaction tx = null;

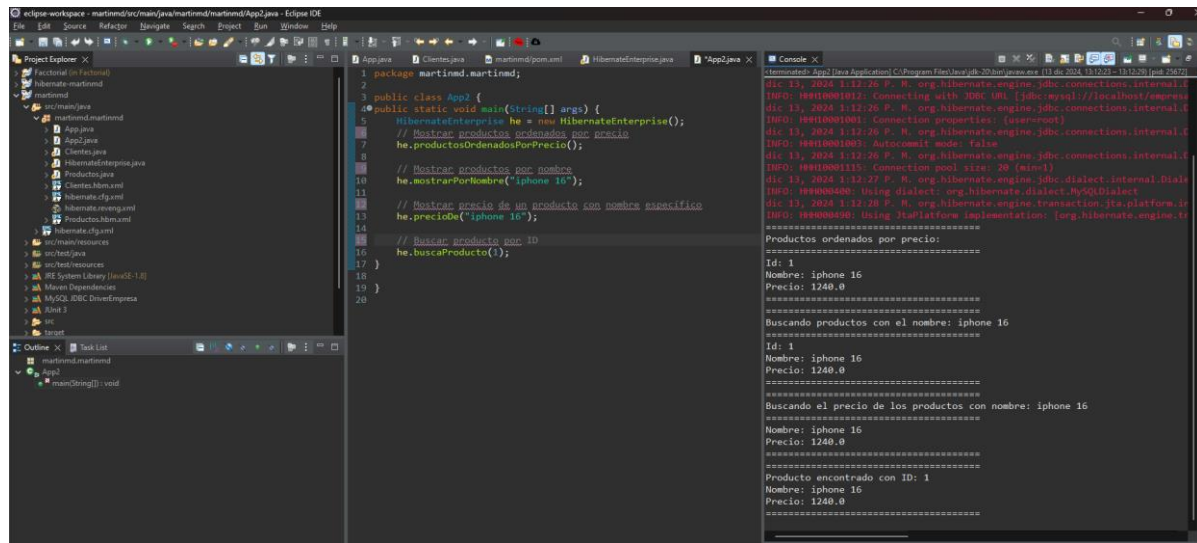
    try {
        tx = session.beginTransaction();
        Productos p = session.createQuery("FROM Productos WHERE id = :id", Productos.class)
            .setParameter("id", id).uniqueResult();

        if (p != null) {
            System.out.println("=====");
            System.out.println("Producto encontrado con ID: " + id);
            System.out.println("Nombre: " + p.getNombre());
            System.out.println("Precio: " + p.getPrecio());
            System.out.println("=====");
        } else {
            System.out.println("No se encontró el producto con ID: " + id);
        }
        tx.commit();
    } catch (HibernateException e) {
        if (tx != null) {
            tx.rollback();
        }
        e.printStackTrace();
    } finally {
        session.close();
    }
}
}

```

}

3. Un pantallazo de la consola al ejecutar cada método.



```
1 package martindm.martindm;
2
3 public class App2 {
4     public static void main(String[] args) {
5         HibernateEnterprise he = new HibernateEnterprise();
6         // Mostrar productos ordenados por precio
7         he.productosOrdenadosPorPrecio();
8
9         // Mostrar productos por nombre
10        he.mostrarPorNombre("iphone 16");
11
12        // Mostrar precio de un producto con nombre específico
13        he.precioDe("iphone 16");
14
15        // Buscar producto por ID
16        he.buscaProducto(1);
17    }
18 }
19
20
```

```

Terminated: App2 [Java Application] C:\Program Files\Java\jdk-20\bin\java.exe (11 de 2024 13:12:25 - 13:12:29 [pid 75672])
dic 13, 2024 1:12:26 P. M. org.hibernate.engine.jdbc.connections.internal.
INFO: HHH10001012: Connecting with JDBC URL jdbc:mysql://localhost/emprer
dic 13, 2024 1:12:26 P. M. org.hibernate.engine.jdbc.connections.internal.
INFO: HHH10001001: Connection properties: {user=root}
dic 13, 2024 1:12:26 P. M. org.hibernate.engine.jdbc.connections.internal.
INFO: HHH10001003: autocommit mode: false
dic 13, 2024 1:12:26 P. M. org.hibernate.engine.jdbc.connections.internal.
INFO: HHH10001115: Connection pool size: 20 (min=1)
dic 13, 2024 1:12:27 P. M. org.hibernate.engine.jdbc.dialect.internal.Dia
INFO: HHH000040: Using dialect: org.hibernate.dialect.MySQLDialect
dic 13, 2024 1:12:28 P. M. org.hibernate.engine.transaction.jta.platform.in
INFO: HHH000040: Using JtaPlatform implementation: [org.hibernate.engine.j
=====
Productos ordenados por precio:
=====
Id: 1
Nombre: iphone 16
Precio: 1240.0
=====
Buscando productos con el nombre: iphone 16
=====
Id: 1
Nombre: iphone 16
Precio: 1240.0
=====
Buscando el precio de los productos con nombre: iphone 16
=====
Nombre: iphone 16
Precio: 1240.0
=====
Producto encontrado con ID: 1
Nombre: iphone 16
Precio: 1240.0
=====

```