HQL

//HIBERNATE QUERY LANGUAGE

IT BOARDING

BOOTCAMP

Índice

01 Características

02 Sintaxis

03 Ubicación de consultas en SPRING

04 Cuidado con usar SQL con Hibernate

IT BOARDING
BOOTCAMP

// Características

IT BOARDING

BOOTCAMP

// ¿Que es HQL (Hiberanate Query Language)?

Es un lenguaje de consultas que proporciona Hibernate, es similar a SQL Standard

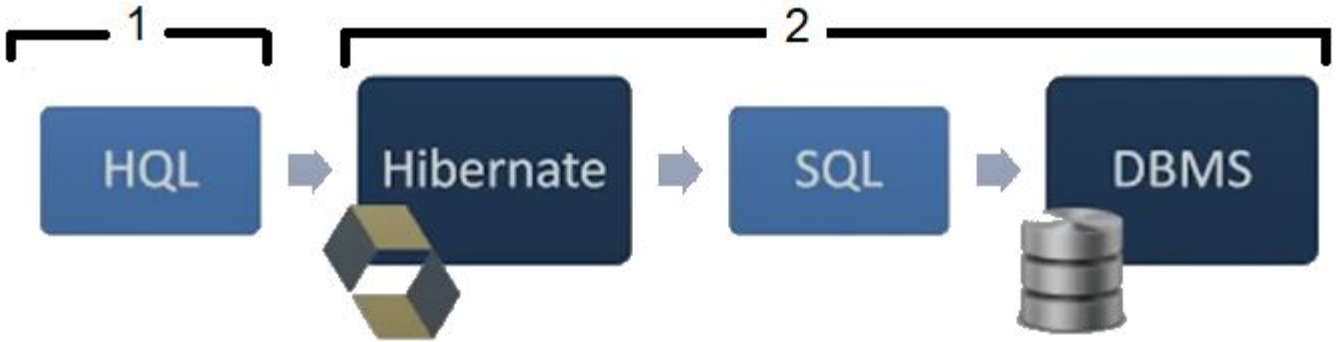
Características de HQL

- Los tipos de datos son los de JAVA.
- Las consultas son independientes del lenguaje SQL
- Las consultas son independientes del modelo de tablas
- **Trabaja con clases y atributos**
- Se puede tratar con colecciones de JAVA
- Se puede navegar entre distintos objetos en la propia consulta

¿Cómo funciona HQL?

1- Escribimos las consultas en HQL

2- Hibernate las convierte (**traduce**) a SQL, según la base de datos que estemos utilizando



Traducción HQL a SQL

- El concepto de “traducción” es importante para entender qué hace Hibernate cuando ejecutamos HQL

HQL

```
FROM Empleado e WHERE e.id = 1
```

traducción

SQL

```
SELECT ID, NOMBRE, APELLIDO FROM empleados WHERE ID = 1
```

HQL

```
FROM Empleado e WHERE e.nombre = 'Damian'
```

traducción

SQL

```
SELECT ID, NOMBRE, APELLIDO FROM Empleados e
WHERE e.NOMBRE = 'Damian'
```

// Sintaxis

IT BOARDING

BOOTCAMP

Elementos HQL

La mayoría de la sintaxis y características son similares a SQL

Una consulta HQL puede constituirse de los siguientes elementos

Elementos									
Cláusulas	from	as	select	where	order by	group by	update	delete	Insert
Funciones Agregadas	avg	sum	min	max	count(*) count(...) count(distinct ...) count(all...)				
Sub Consultas	Consultas dentro de consultas								

Sintaxis de consultas HQL

Clausula
Elemento con palabras claves específica de HQL

Clase
Hace referencia a la clase y no a la tabla de la base de datos.

Atributo de Clase
Hace referencia al atributo de la clase y no a la columna de la tabla

Mayúsculas

- Las palabras clave del lenguaje **NO** son sensibles a Mayúsculas y Minúsculas

- El nombre de las clases Java y sus propiedades **SI** son sensibles a las mayúsculas o minúsculas

- Al realizar comparaciones con los valores de las propiedades, éstas **NO** son sensibles a las mayúsculas o minúsculas.

Cláusulas

- **FROM:** carga un objeto persistente en memoria
- **AS:** asignar alias a la clase
- **SELECT:** obtiene propiedades de objetos, en lugar del objeto completo
- **WHERE:** recupera objetos específicos
- **ORDER BY:** ordena resultados por propiedad de los objetos, ascendente ASC o descendente DESC
- **GROUP BY:** devuelve valores agregados que se agrupan por cualquier propiedad
- **UPDATE:** actualiza una o mas propiedades
- **DELETE:** elimina uno o mas objetos
- **INSERT:** inserta objetos

```
HQL  FROM Usuario
```

```
HQL  FROM Usuario AS u
```

```
HQL  SELECT u.nombre FROM  Usuario u
```

```
HQL  FROM Usuario u WHERE e.id = 999
```

```
HQL  FROM Usuario u WHERE u.id > 10 ORDER BY u.nombre DESC
```

```
HQL  SELECT SUM(P.salary), P.firtName FROM Person P
```

```
HQL  "UPDATE Employee set salary =: salary "+ "WHERE id =:empId"
```

```
HQL  "DELETE FROM Employee " + "WHERE id = :empId"
```

```
HQL  "INSERT INTO Person(firstName, lastName, salary)"  +
     "SELECT firstName, lastName, salary FROM old_person"
```

Sintaxis de consultas en SPRING

Consulta HQL
Consulta con sintaxis HQL

Parámetro de la consulta HQL
:[nombre del Parámetro]
Debe ser usado luego en el método

HQL

```
@Query("select u from UsuarioModel u where u.userName like :name order by u.userName")
List<UsuarioModel> findUsuarioModelByName(@Param("name") String name);
```

Método
Expuesto por la Interface

Si la Query posee parámetros, se debe utilizar

@Param(“[nombre del Parámetro]”) +[Tipo de dato java] +[nombre de variable]

Consultas: Nombre de Métodos

Como buena práctica crear nombres de métodos siguiendo las siguientes pautas.

Keyword	*Sample*
And	**findBy**Lastname**And**Firstname
Or	**findBy**Lastname**Or**Firstname
Is,Equals	**findBy**Firstname, **findBy**Firstname**Is**, **findBy**Firstname**Equals**
Between	**findBy**StartDate**Between**
LessThan	**findBy**Age**LessThan**
LessThanEqual	**findBy**Age**LessThanEqual**
GreaterThan	**findBy**Age**GreaterThan**
GreaterThanEqual	**findBy**Age**GreaterThanEqual**
After	**findBy**StartDate**After**
Before	**findBy**StartDate**Before**
IsNull	**findBy**Age**IsNull**
IsNotNull,NotNull	**findBy**Age**NotNull**, **findBy**Age**IsNotNull**
Like	**findBy**Firstname**Like**
NotLike	**findBy**Firstname**NotLike**
StartingWith	**findBy**Firstname**StartingWith**
EndingWith	**findBy**Firstname**EndingWith**
Containing	**findBy**Firstname**Containing**
OrderBy	**findBy**Age**OrderBy**Lastname**Desc**

// Ubicación de consultas en SPRING

IT BOARDING

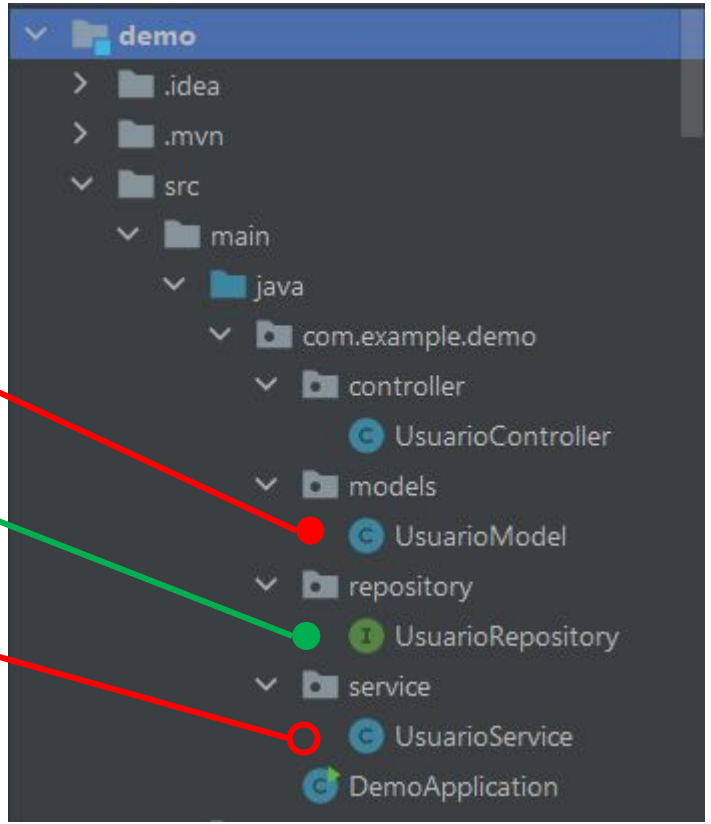
BOOTCAMP

¿Donde ubicar las consultas HQL?

Dentro del model @NameQuery **(POCO RECOMENDADO)**

Dentro del repository @Query **(IDEAL)**

Dentro del service EntityManager **(POCO RECOMENDADO)**



Creando una Consulta HQL en Repository

1- crear la interface UsuarioRepository

2- extender de CrudRepository

3- crear la consulta HQL con la Anotación @Query

4- crear el método para esa consulta

```
@Repository
public interface UsuarioRepository extends CrudRepository<UsuarioModel, Long> {

    @Query("select u from UsuarioModel u where u.userName like :name order by u.userName")
    List<UsuarioModel> findUsuarioModelByName(@Param("name") String name);
}
```

Usando la Consulta HQL creada en Service

1- crear la clase UsuarioService

2- inyectar dependencia de UsuarioRepository

3- crear Método para llamar al método de Repository

```
@Service
public class UsuarioService  {
    @Autowired
    UsuarioRepository usuarioRepository;

    public ArrayList<UsuarioModel> obtenerUsuarios(){
        return (ArrayList<UsuarioModel>) usuarioRepository.findAll();
    }

    public ArrayList<UsuarioModel> obtenerUsuariosPorNombre(String nombre){
        return (ArrayList<UsuarioModel>) usuarioRepository.findUsuarioModelByName(nombre);
    }

    public  UsuarioModel guardarUsuario(UsuarioModel nuevoUsuario) { return usuarioRepository.save(nuevoUsuario); }

}
```

Creando una Consulta HQL en Service

1- crear la clase UsuarioService

2- inyectar EntityManager **@PersistenceContext**

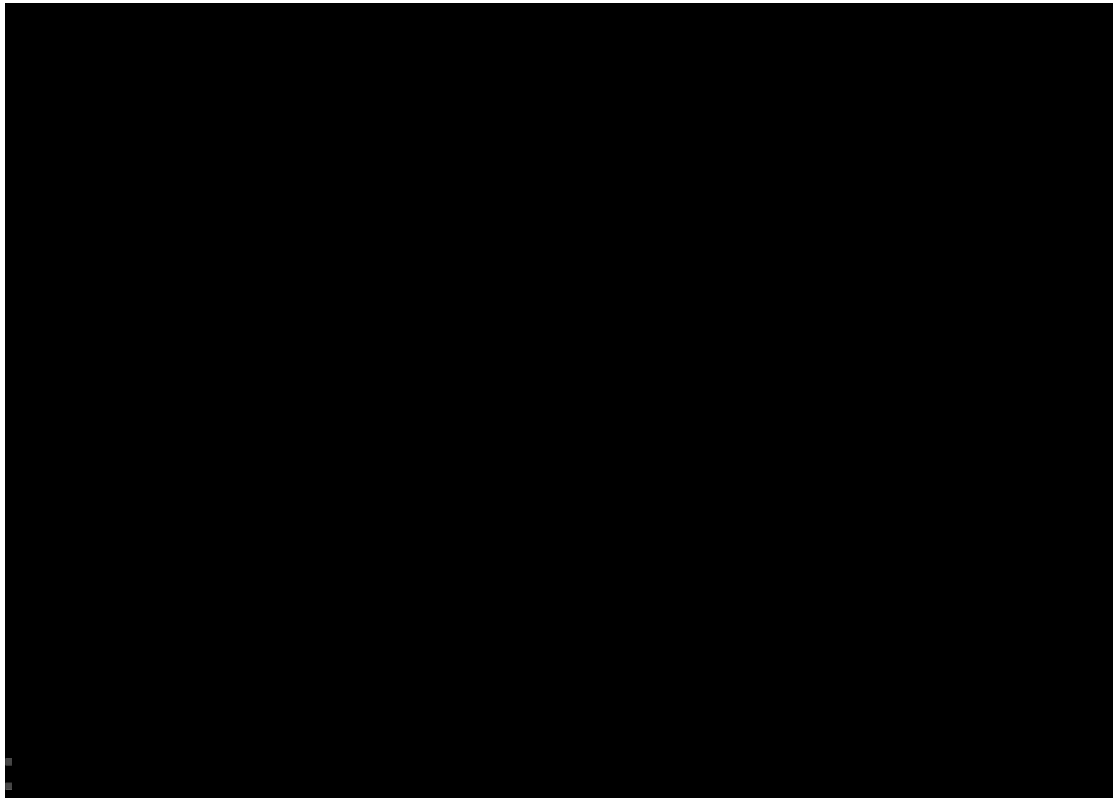
4- crear Método para usar EntityManager

5- a partir del entityManger se crea la query

```
@Service
public class UsuarioService  {
    @PersistenceContext
    private EntityManager entityManager;

    public ArrayList<UsuarioModel> obtenerUsuariosPorNombre2(String nombre){

        ArrayList<UsuarioModel> users = (ArrayList<UsuarioModel>) entityManager.
                createQuery( qlString: "select u from UsuarioModel u where u.userName like ?1 order by u.userName")
                .setParameter( position: 1, nombre)
                .getResultList();
        return users;
    }

}
```

SE PUEDE HACER, pero no lo hagas

Creando las consultas HQL en SPRING



// Cuidado con usar SQL en Hibernate

IT BOARDING

BOOTCAMP

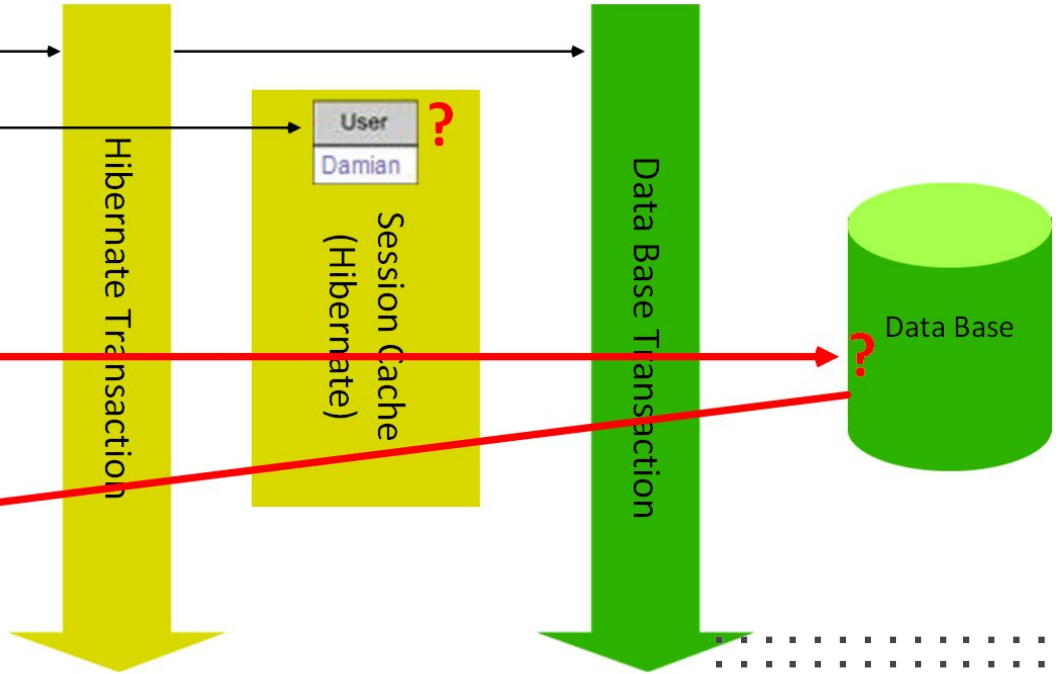
Cómo funcionan las Queries SQL (no HQL/Criteria) con la cache de session en Hibernate

(Cuidado con el Uso de SQL Queries)

```
public void testFindUser throws Exception(){

        Session session = Factorory.openSession();
        Transation tx = session.beginTransaction();


        //Crear un Usuario y guardarlo
        User user = ner User();
        user.setName("Damian");
        session.save(user);



List result = session.createSqlQuery("select name from user where name =:userName")
.setparameter("userName", user.getName())
.list();

//El assert de usuario no funcionara
asssertEquals(1, list.size());
asssertEquals("Damian", (String) result.get(0));


session.rollbackTransaction();


}
```

SQL en createSQLQuery
Al contrario que con SQL/ Criteria, Hibernate No Comprueba la cache para objetos involucrados en la consulta SQL

Como resultado la consulta NO Encontrara al usuario "Damian" en la base de datos



Enlaces de Interés

Documentación HIBERNATE HQL

https://docs.jboss.org/hibernate/orm/3.6/reference/es-ES/html/queryhql.html#queryhql-polymorphism

Spring Data JPA - Reference Documentation

https://docs.spring.io/spring-data/data-jpa/docs/1.0.0.M1/reference/html/#jpa.query-methods.query-creation

Gracias.

IT BOARDING

BOOTCAMP