# Lenguaje HQL 2 – Manipulación de los datos

## Tabla de contenido



## Manipulación de la base de datos a través del lenguaje HQL

### Interfaz Query

La interfaz Query permite hacer consultas HQL usando la API de Hibernate.

Vamos a ver como se puede obtener una sentencia de Query a partir de una instancia de Session mediante el método:

| | |
|---|---|
| createQuery (String consulta) | Devuelve una instancia de la interfaz org.hibernate.query.Query |

Para comprender el funcionamiento de createQuery, podemos pensar en PreparedStatement de JDBC.

Una Query puede tener parámetros que se pueden identificar por nombre o por posición.

### Funcionalidades de la clase Query: List()

El objeto Query que nos da acceso a todas las funcionalidades para poder leer objetos desde la base de datos.

**Lista de Objetos**

El método list() devuelve una lista con todos los objetos que ha retornado la consulta. En caso de que no se encuentre ningún resultado se retornará una lista sin ningún elemento.

Ejemplo:

```
String hql = "select d from Departamentos d";
Query query = sesion.createQuery(hql);
//************LISTA list()
List<Departamentos> departamentos = query.list();
for (Departamentos dep : departamentos) {
    System.out.println("Departamento = " + dep.toString());
}
```

**Lista de Array de Objetos**

También se permite que las consultas retornen datos escalares en vez de clases completas.

Ejemplo:

```
String hql = "select d.nombre, d.localidad from Departamentos d";
Query query = sesion.createQuery(hql);
List<Object[]> datos = query.list();
for(Object[] dato: datos){
    System.out.println("Datos = " + dato[0] + " - " + dato[1]);
}
```

**Lista de Objeto**

Hay otro caso cuando hay una única columna en el SELECT de datos escalares. Entonces, el array a devolver dentro de la lista solo tendría un elemento. Por lo tanto, no se retorna una lista de arrays List<Object[]> sino únicamente una lista de elementos List<Object>.

Si modificamos la anterior consulta de forma que sólo devuelva el nombre, quedará de la siguiente forma:

```
String hql = "select d.nombre from Departamentos d";
Query query = sesion.createQuery(hql);
List<Object> datos = query.list();
for(Object dato: datos){
    System.out.println("Datos = " + dato);
}
```

**uniqueResult()**

En muchas ocasiones una consulta únicamente devuelve cero o un resultado. Entonces, es poco práctico que devuelva una lista con un único elemento. Para facilitarnos dicha tarea Hibernate dispone del método uniqueResult().

Este método retornará directamente el único objeto que ha obtenido la consulta. En caso de que no encuentre ninguno se retornará null.

```
String hql = "select d from Departamentos d where d.pkdepartamento=30";
Query query = sesion.createQuery(hql);
Departamentos dep = (Departamentos) query.uniqueResult();
if (dep!=null){
    System.out.println("Departamento =  " + dep);
}
else{
    System.out.println("Nada que mostrar");
}
```

**Paginación**

La paginación es parte fundamental de cualquier aplicación ya que una consulta puede tener miles de resultados y no queremos mostrarlos todos a la vez.

Para conseguir paginar el resultado de una consulta la clase Query dispone de los siguientes métodos:

- setMaxResults(int maxResults): Establece el nº máximo de objetos que van a retornarse.
- setFirstResult(int firstResult): Establece el primer de los objetos que se van a retornar.

Al realizar la paginación son necesarios al menos 2 valores:

- El tamaño de la página
- El nº de la página a mostrar

Con esos 2 valores hemos creado el siguiente código Java que muestra una única página en función del tamanyoPagina y primeroPagina.

```
int tamanyoPagina = 4;
int primeroPagina = 2;
String hql = "select d from Departamentos d order by d.localidad asc";
Query query = sesion.createQuery(hql);
query.setMaxResults(tamanyoPagina);
query.setFirstResult(primeroPagina);
List<Departamentos> departamentos = query.list();
for(Departamentos dep : departamentos){
    System.out.println("Datos = " + dep.toString());
}
```

El mayor problema que tiene la paginación es determinar el nº de páginas que hay para acotar lo que se muestra al usuario.

Para saber el nº de páginas es necesario conocer el nº de objetos que devuelve la consulta y la forma más rápida es con una consulta que las cuente.

```
//Consulta de count(*^) para obtener el nº de objetos que retorna la consulta
long numTotalObjetos = (Long) sesion.createQuery("select count(*) from Departamentos d").uniqueResult();
//divide el nº total de objetos entre el tamaño de la página obteniéndose el nº total de páginas
int numPaginas = (int) Math.ceil((double)numTotalObjetos/(double)tamanyoPagina);
```

## Diferencias entre HQL y SQL.

En el siguiente enlace tenemos un sencillo manual para empezar a trabajar con las consultas HQL

http://www.cursohibernate.es/doku.php?id=unidades:05_hibernate_query_language:02_hql