JPA NamedQueries es una de las características más utilizadas de JPA ya que nos permite diseñar las consultas en las propias entidades y tenerlas muy a mano. Ademas JPA es el Standard y nos encontraremos con muchisimas aplicaciones desarrolladas con él. Sin embargo según va creciendo el proyecto podemos tener algunos problemas y limitaciones. Vamos a mostrar un posible bloque de código:

```
package com.arquitecturajava;

import java.io.Serializable;

import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.NamedQueries;
import javax.persistence.NamedQuery;

@Entity
@NamedQueries({
    @NamedQuery(name="seleccionarAlumnosNombre", query="select a from
Alumno a where a.nombre=:nombre"),
    @NamedQuery(name="seleccionarAlumnosApellidos", query="select a
from Alumno a where a.apellidos=:apellidos")
})
public class Alumno implements Serializable {

    private static final long serialVersionUID = 1L;

    @Id
    private String dni;
    private String nombre;
    private String apellidos;
    private int edad;
```

```
    public Alumno() {
        super();
    }

    public Alumno(String dni, String nombre, String apellidos, int
edad) {
        super();
        this.dni = dni;
        this.nombre = nombre;
        this.apellidos = apellidos;
        this.edad = edad;
    }

    public String getDni() {
        return dni;
    }
    public void setDni(String dni) {
        this.dni = dni;
    }
    public String getNombre() {
        return nombre;
    }
    public void setNombre(String nombre) {
        this.nombre = nombre;
    }
    public String getApellidos() {
        return apellidos;
    }
    public void setApellidos(String apellidos) {
        this.apellidos = apellidos;
    }
```

```
    public int getEdad() {
        return edad;
    }
    public void setEdad(int edad) {
        this.edad = edad;
    }
}
```

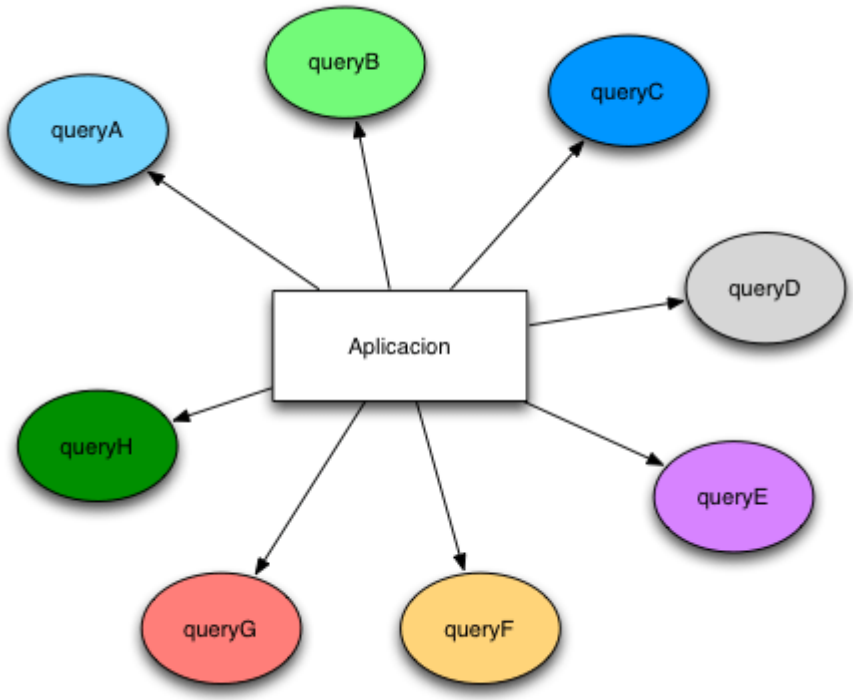
JPA NamedQueries y problemas

Se trata de una entidad muy sencilla que contiene el concepto de Alumno y dos NamedQueries (seleccionarAlumnosNombre, seleccionarAlumnosApellidos). Son consultas muy sencillas de manejar por cualquier programador. Para invocar estas consultas diseñaremos el siguiente programa main:

```
EntityManagerFactory emf =
Persistence.createEntityManagerFactory("UnidadCurso");
EntityManager em = emf.createEntityManager();
TypedQuery<Alumno> consultaAlumnosNombre = em.createNamedQuery(
    "seleccionarAlumnosNombre", Alumno.class);
consultaAlumnosNombre.setParameter("nombre", "miguel");
List<Alumno> lista = consultaAlumnosNombre.getResultList();

System.out.println("*************Alumnos*********");
for (Alumno a : lista) {
    System.out.println(a.getNombre() + "," + a.getApellidos());
}

TypedQuery<Alumno> consultaAlumnosApellidos = em.createNamedQuery(
    "seleccionarAlumnosApellidos", Alumno.class);
consultaAlumnosApellidos.setParameter("apellidos", "alvarez");
List<Alumno> lista2 = consultaAlumnosApellidos.getResultList();
```

```
System.out.println("*************Alumnos*********");
for (Alumno a : lista2) {
    System.out.println(a.getNombre() + "," + a.getApellidos());
}

em.close();
System.out.println("termino");
```

Las consultas funcionarán sin problemas. Ahora bien :¿Qué pasará cuando tengamos más clases y más consultas?.

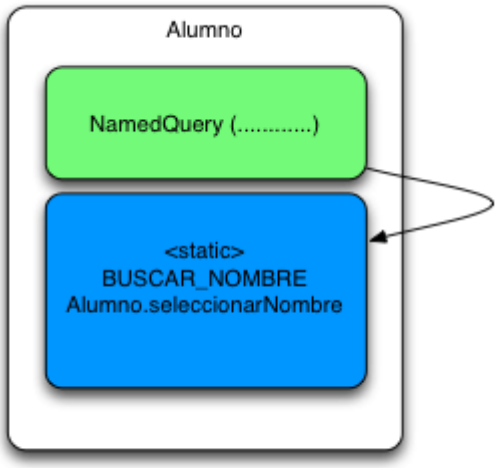


JPA Organización

Será mucho más sencillo equivocarnos y no saber cuales son los nombres de las consultas correctos y acabaremos con mucho lio . Para solventar esta problemática necesitamos organizarnos. El primer paso a realizar será asignar a las consultas un nombre que empiece por la clase a la que pertenecen.

```
@Entity
@NamedQueries({
    @NamedQuery(name="Alumno.seleccionarNombre", query="select a from
Alumno a where a.nombre=:nombre"),
    @NamedQuery(name="Alumno.seleccionarApellidos", query="select a
from Alumno a where a.apellidos=:apellidos")
})
class Alumno {
}
```

De esta forma será mas fácil organizarse entre cientos de consultas y saber donde se encuentra ubicada cada una de ellas. En segundo lugar podemos convertir el propio nombre de la consulta en una variable estática para que cuando la invoquemos desde un programa main no cometamos errores.




```
@Entity
@NamedQueries({
    @NamedQuery(name=Alumno.BUSCAR_NOMBRE, query="select a from Alumno
a where a.nombre=:nombre"),
    @NamedQuery(name=Alumno.BUSCAR_APELLIDOS, query="select a from
Alumno a where a.apellidos=:apellidos")
```

```
})
public class Alumno implements Serializable {

    private static final long serialVersionUID = 1L;
    public static final String BUSCAR_NOMBRE =
"Alumno.seleccionarNombre";
    public static final String BUSCAR_APELLIDOS =
"Alumno.seleccionarApellidos";
}
```

De esta forma cuando tengamos que invocar las JPA NamedQueries todo será mucho más fácil y sobre todo mucho más ordenado y con menos posibilidad de cometer errores en el día a día.

```
TypedQuery<Alumno> consultaAlumnosNombre = em.createNamedQuery(
    Alumno.BUSCAR_NOMBRE, Alumno.class);
```

De esta forma reduciremos los problemas a nivel de nomenclatura dentro de nuestras clases y consultas.

Otros artículos relacionados

- Utilizando JPA NamedQueries
- JPA Query Language Objetos vs Tablas
- Cursos Arquitectura Java para Empresas
- Spring responsabilidades y organización

Links Externos

JPA Specificacion