Guía Completa de Consultas HQL en Hibernate

Introducción

HQL (Hibernate Query Language) permite interactuar con la base de datos usando clases y atributos Java en lugar de tablas y columnas SQL.
Objetivo: simplificar consultas y aprovechar relaciones definidas con anotaciones JPA/Hibernate.

1. Selección simple (FROM)

Objetivo: Obtener todos los registros de una entidad.
HQL:
FROM Alumno
Java:
```
List<Alumno> alumnos = session.createQuery("FROM Alumno", Alumno.class).list();
```
Explicación: Selecciona todos los objetos Alumno. Equivalente a SELECT * FROM alumnos.

2. Filtrado (WHERE)

Objetivo: Restringir registros según una condición.
HQL:
FROM Alumno a WHERE a.curso.nombre = 'Programación Java'
Java:
```
List<Alumno> alumnos = session.createQuery(
"FROM Alumno a WHERE a.curso.nombre = :cursoNombre", Alumno.class)
.setParameter("cursoNombre", "Programación Java")
.list();
```
Explicación: WHERE filtra datos según igualdad, desigualdad, mayor, menor, etc. Usar parámetros para seguridad.

3. Condiciones compuestas (AND / OR)

Objetivo: Combinar varias condiciones lógicas.
HQL:
FROM Alumno a WHERE a.nombre LIKE 'Juan%' AND a.curso.nombre = 'Programación Java'
Explicación: AND → todas las condiciones deben cumplirse. OR → solo una debe cumplirse.

4. Lista de valores (IN)

Objetivo: Filtrar registros que coincidan con un conjunto de valores.
HQL:
FROM Alumno a WHERE a.curso.nombre IN ('Programación Java', 'Base de Datos')
Explicación: Evita múltiples OR. Ideal para listas conocidas.

5. Ordenamiento (ORDER BY)

Objetivo: Ordenar resultados por atributos.
HQL:
FROM Alumno a ORDER BY a.nombre ASC
Explicación: ASC → ascendente, DESC → descendente. Permite ordenar por relaciones también.

6. Agrupamiento (GROUP BY)

Objetivo: Agrupar registros y aplicar funciones de agregación.

HQL:
SELECT a.curso.nombre, COUNT(a) FROM Alumno a GROUP BY a.curso.nombre
Explicación: Muestra la cantidad de alumnos por curso.

7. Inner Join (JOIN / INNER JOIN)

Objetivo: Unir entidades relacionadas.
HQL:
SELECT a.nombre, c.nombre FROM Alumno a INNER JOIN a.curso c
Explicación: INNER JOIN trae registros que tienen relación en ambas entidades.

8. Existencia (EXISTS)

Objetivo: Filtrar registros que cumplan subconsulta.
HQL:
FROM Alumno a WHERE EXISTS (SELECT m FROM a.materias m WHERE m.nombre = 'Hibernate')
Explicación: EXISTS devuelve true si la subconsulta tiene resultados.

9. Patrones (LIKE)

Objetivo: Filtrar texto usando patrones.
HQL:
FROM Alumno a WHERE a.nombre LIKE '%Pérez%'
Explicación: % → cualquier número de caracteres, _ → un solo carácter.

10. Rango (BETWEEN)

Objetivo: Filtrar valores dentro de un rango.
HQL:
FROM Alumno a WHERE a.id BETWEEN 1 AND 5
Explicación: Devuelve registros con id entre 1 y 5.

11. Distintos (DISTINCT)

Objetivo: Evitar resultados duplicados.
HQL:
SELECT DISTINCT a.curso.nombre FROM Alumno a
Explicación: Muestra cursos únicos.

12. Condición en agregación (HAVING)

Objetivo: Filtrar después de agrupar.
HQL:
SELECT a.curso.nombre, COUNT(a) FROM Alumno a GROUP BY a.curso.nombre HAVING COUNT(a) > 2
Explicación: WHERE filtra antes de agrupar, HAVING filtra después.

Buenas prácticas

- Usar parámetros (:param) para seguridad.
- HQL usa nombres de clases y atributos, no tablas/columnas.
- Usar JOIN FETCH para relaciones lazy.
- Combinar GROUP BY, HAVING y funciones de agregación para reportes.