Hibernate Query Language:HQL

一、HQL 概述

HQL(Hibernate Query Language)是面向对象的查询语言，它和 SQL 查询语言有些相似。在 Hibernate 提供的各种检索方式中，HQL 是使用最广的一种检索方式。它有如下特点：

- ●以面向对象的方式查询数据库表
- ●在查询语句中设定各种查询条件
- ●支持投影查询，即仅检索出对象的部分属性
- ●支持分页查询
- ●支持连接查询
- ●支持分组查询，允许使用 HAVING 和 GROUP BY 关键字
- ●提供内置聚集函数，如 sum()，min()和 max()等
- ●支持子查询
- ●支持动态绑定参数

二、语法细节

1.FROM 子句

在 SQL 中，FROM 子句是用于指定要查询的数据库表，在 HQL 中，替换为与数据库表对应的 Java 实体类即可。

例如：

SQL	SELECT * FROM EMPS
HQL	FROM Employee

此时 HQL 语句将查询数据库表中的所有字段，并自动注入到对应实体类对象的对应属性中。

同时需要说明的是：HQL 语句中 FROM 等关键字不区分大小写，但 Java 类的类名严格区分大小写。

2.执行 HQL 查询语句

在 Hibernate 中，HQL 语句由 Query 对象执行，Query 对象可以通过 Session 对象获取。

```
//1.通过 session 对象创建 Query 对象
String hql = "FROM Employee";
Query query = session.createQuery(hql);

//2.调用 Query 对象的方法获取查询结果
//结果集中有多条记录：list()
//结果集中只有一条记录：uniqueResult()
List<Employee> list = query.list();
for (Employee employee : list) {
```

```
            System.out.println(employee);
        }
```

3.WHERE 子句

和 SQL 的语法一样，HQL 中的 WHERE 子句也用来指定查询条件。只不过这里指定查询条件使用的不是数据库表的字段，而是 Java 类的属性。

例如：

SQL	SELECT * FROM EMPS WHERE EMPS.SALARY>5000
HQL	FROM Employee e WHERE e.salary>5000

这里 Employee e 的语法非常像 Java 中声明一个 Employee 类型的变量：e。其实也确实可以这样理解——使用 e 作为 Employee 对象的引用。

4.使用基于位置的占位符参数

将上例中的具体值使用占位符“?”代替，并调用 Query 对象的 setXxx()方法按照参数的不同类型动态填充即可，需要注意的是和 JDBC 中的 PreparedStatement 接口不同，这里占位符的索引从 0 开始。

HQL	FROM Employee e WHERE e.salary>?

填充占位符	query.setDouble(0, 8000);

5.使用具名参数

在 HQL 中不但能够使用基于位置的占位符参数，还能够使用基于名称的具名参数，使用具名参数的好处是不必关心当前参数的索引值。

具名参数的格式是：“:参数名称”。

例如：

HQL	FROM Employee e WHERE e.salary>:salaryParam

填充占位符	query.setDouble("salaryParam", 9000);

6.以实体类对象作为参数

对于已经通过 Hibernate 关联关系映射建立了关联关系的实体类，HQL 还支持直接使用实体类对象本身作为参数值。

例如：Employee 和 Department 之间建立了单向（或双向）多对一关联关系，Employee 类中使用 department 属性关联 Department 类的对象，那么 HQL 语句可以为：

HQL	from Employee e where e.department=?

这里填充占位符可以使用一个 Department 对象：

```
Department department = new Department();
department.setDeptId(5);
```

填充占位符	query.setEntity(0, department)

Hibernate 会自动按照关联关系中规定的主外键关系进行查询。

7.ORDER BY 子句

使用 ORDER BY 子句可以进行排序

HQL	FROM Employee e WHERE e.salary>:salaryParam ORDER BY e.salary DESC

和 SQL 一样，默认按照升序排列。DESC 表示降序，ASC 表示升序。

8.分页查询

分页查询是 HQL 的一大亮点，不必关心底层数据库的具体实现是什么，使用 HQL 调用固定的方法就能够实现垮数据库平台的分页查询。

在分页时我们需要指定两个最基本的数据，一个是当前页的页码：pageNo，一个是每页显示多少条数据：pageSize。

下表列出了 Query 接口中与分页相关的两个函数

函数名	作用
setFirstResult(int index)	指定查询结果从 index 位置开始取，index 从 0 开始
setMaxResults(int maxResults)	指定查询结果取 maxResults 条数据

index 和 pageNo 的关系是：index=(pageNo-1)*pageSize

具体操作方法是：

```
List<Employee> list = query.setFirstResult((pageNo - 1)*pageSize)
                           .setMaxResults(pageSize)
                           .list();
```

这里我们使用了连缀的方式调用 Query 对象的 API，之所以能够实现连缀是因为每个方法的返回值都仍然是 Query 对象本身。

9.投影查询方式一

所谓投影查询其实就是仅查询实体类对象的部分字段，这里用到了 HQL 语句的 SELECT 关键词。

HQL	SELECT e.empName,e.salary From Employee e WHERE e.salary>9000

那么此时的查询结果以什么形式返回呢？HQL 并没有将使用空的 Employee 对象接收 empName 和 salary 的值，而是把它们放在了一个 Object 数组中。

```
Query query = session.createQuery(queryString);
List<Object[]> list = query.list();
for (Object[] objects : list) {
   System. out.println(objects[0]+" "+objects[1]);
```

```
}
```

10.投影查询方式二

接收投影查询结果可以仍然使用实体类的对象，但要求实体类中提供对应的构造器。

HQL	SELECT **new Employee**(e.empName,e.salary) From Employee e

这样得到的每一条数据都将被封装到 Employee 对象中。

11.多表查询

HQL 支持使用外连接、内连接等方式进行连表查询，甚至支持使用 FETCH 关键字进行“迫切”连接查询。

①迫切左外连接

HQL	From Department d **LEFT JOIN FETCH** d.empSet

这里使用 LEFT JOIN 表示进行“左外连接”查询，生成的 SQL 语句将查询关联的 Employee 类对应的全部数据，所以 Employee 的数据既然查询得到了，那么就应该将它们设置到 Department 对象的 empSet 属性中，否则这个已经执行了的操作就浪费了。而是否对 empSet 属性进行设置就看是否包含了 FETCH 关键字，包含就设置，不包含就不设置，这就是“迫切”的含义。

②迫切内连接

HQL	From Department d **INNER JOIN FETCH** d.empSet

12.报表查询

和 SQL 一样，HQL 也使用 GROUP BY 和 HAVING 子句配合起来进行分组，再结合统计函数进行报表查询。

HQL	SELECT **min**(e.salary),**max**(e.salary) From Employee e **GROUP BY** e.department **HAVING** min(e.salary)>3000

min()和 max()统计函数的结果最终被放在了一个 Object 数组中。

13.子查询

子查询是 SQL 语句中非常重要的功能，它可以在 SQL 语句中利用另外一条 SQL 语句的查询结果。HQL 同样对子查询功能提供了支持。与 SQL 子查询不同的是，HQL 不支持在 FROM 子句中使用子查询。

例如：查询员工数量大于 5 的部门

HQL	From Department d where (**select count(emp) From d.empSet emp**)>4

查询部门名以“A”开头的部门的员工姓名和部门名称

HQL	select e.empName,e.department.deptName from Employee e where e.department in

	(**From Department d where d.deptName like 'A%'**)

14.删除数据

HQL	Delete From Employee e WHERE e.empId=117

session.createQuery(queryString).**executeUpdate**()

15.更新数据

HQL	UPDATE Employee e Set e.empName=’Tom’ WHERE e.empId=115

session.createQuery(queryString).**executeUpdate**()