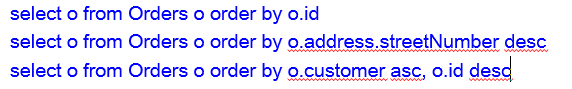
参看易百教程：https://www.yiibai.com/jpa/jpa\_jpql.html

# order by子句

## 语法

order by子句用于对查询结果集进行排序。和SQL的用法类似，可以用 “asc“ 和 "desc“ 指定升降序。如果不显式注明，默认为升序。



## 测试代码

@Test

**public** **void** testOrderBy(){

String jpql = "FROM Customer c WHERE c.age > ? ORDER BY c.age DESC";

Query query = entityManager.createQuery(jpql).setHint(QueryHints.***HINT\_CACHEABLE***, **true**);

//占位符的索引是从 1 开始

query.setParameter(1, 4);

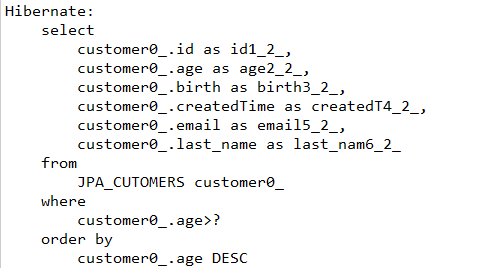
List<Customer> customers = query.getResultList();

**for** (Customer customer : customers) {

System.***out***.println(customer);

}

}

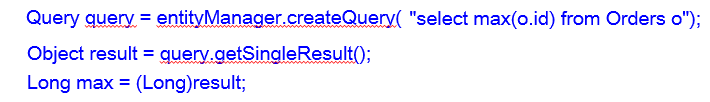


# group by子句

## group by子句

group by 子句用于对查询结果分组统计，通常需要使用聚合函数。常用的聚合函数主要有 AVG、SUM、COUNT、MAX、MIN 等，它们的含义与SQL相同。例如：select max(o.id) from Orders o

没有 group by 子句的查询是基于整个实体类的，使用聚合函数将返回单个结果值，可以使用Query.getSingleResult()得到查询结果。例如：



## having子句

Having 子句用于对 group by 分组设置约束条件，用法与where 子句基本相同，不同是 where 子句作用于基表或视图，以便从中选择满足条件的记录；having 子句则作用于分组，用于选择满足条件的组，其条件表达式中通常会使用聚合函数。

例如，以下语句用于查询订购总数大于100的商家所售商品及数量：

select o.seller, o.goodId, sum(o.amount) from V\_Orders o group by o.seller, o.goodId having sum(o.amount) > 100

having子句与where子句一样都可以使用参数。

## 测试代码

//查询 order 数量大于 2 的那些 Customer

@Test

**public** **void** testGroupBy(){

String jpql = "SELECT o.customer FROM Order o "

+ "GROUP BY o.customer "

+ "HAVING count(o.id) >= 2";

List<Customer> customers = entityManager.createQuery(jpql).getResultList();

System.***out***.println(customers);

}

