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

# JPQL语言

## 概念

JPQL语言，即 Java Persistence Query Language 的简称。JPQL 是一种和 SQL 非常类似的中间性和对象化查询语言，它最终会被编译成针对不同底层数据库的 SQL 查询，从而屏蔽不同数据库的差异。

JPQL语言的语句可以是 select 语句、update 语句或delete语句，它们都通过 Query 接口封装执行

## 执行数据库查询方法

Query接口封装了执行数据库查询的相关方法。调用 EntityManager 的 createQuery、create NamedQuery 及 createNativeQuery 方法可以获得查询对象，进而可调用 Query 接口的相关方法来执行查询操作。

## 主要方法

int executeUpdate()

用于执行update或delete语句。

List getResultList()

用于执行select语句并返回结果集实体列表。

Object getSingleResult()

用于执行只返回单个结果实体的select语句。

Query setFirstResult(int startPosition)

用于设置从哪个实体记录开始返回查询结果。

Query setMaxResults(int maxResult)

用于设置返回结果实体的最大数。与setFirstResult结合使用可实现分页查询。

Query setFlushMode(FlushModeType flushMode)

设置查询对象的Flush模式。参数可以取2个枚举值：FlushModeType.AUTO 为自动更新数据库记录，FlushMode Type.COMMIT 为直到提交事务时才更新数据库记录。

setHint(String hintName, Object value)

设置与查询对象相关的特定供应商参数或提示信息。参数名及其取值需要参考特定 JPA 实现库提供商的文档。如果第二个参数无效将抛出IllegalArgumentException异常。

setParameter(int position, Object value)

为查询语句的指定位置参数赋值。Position 指定参数序号，value 为赋给参数的值。

setParameter(int position, Date d, TemporalType type)

为查询语句的指定位置参数赋 Date 值。Position 指定参数序号，value 为赋给参数的值，temporalType 取 TemporalType 的枚举常量，包括 DATE、TIME 及 TIMESTAMP 三个，，用于将 Java 的 Date 型值临时转换为数据库支持的日期时间类型（java.sql.Date、java.sql.Time及java.sql.Timestamp）。

setParameter(int position, Calendar c, TemporalType type)

为查询语句的指定位置参数赋 Calenda r值。position 指定参数序号，value 为赋给参数的值，temporalType 的含义及取舍同前。

setParameter(String name, Object value)

为查询语句的指定名称参数赋值。

setParameter(String name, Date d, TemporalType type)

为查询语句的指定名称参数赋 Date 值。用法同前。

setParameter(String name, Calendar c, TemporalType type)

为查询语句的指定名称参数设置Calendar值。name为参数名，其它同前。该方法调用时如果参数位置或参数名不正确，或者所赋的参数值类型不匹配，将抛出 IllegalArgumentException 异常。

# 创建查询三个方法

## createQuery（jpql）

### 查询所有

@Test

**public** **void** testHelloJPQL(){

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

Query query = entityManager.createQuery(jpql);

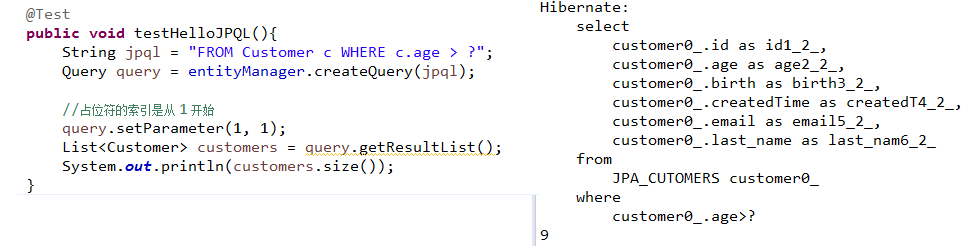
//占位符的索引是从 1 开始

query.setParameter(1, 1);

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

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

}



### 查询部分

//默认情况下, 若只查询部分属性, 则将返回 Object[] 类型的结果. 或者 Object[] 类型的 List.

//也可以在实体类中创建对应的构造器, 然后再 JPQL 语句中利用对应的构造器返回实体类的对象.

@Test

**public** **void** testPartlyProperties(){

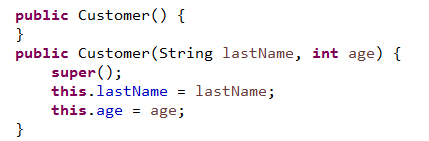
String jpql = "SELECT new Customer(c.lastName, c.age) FROM Customer c WHERE c.id > ?";

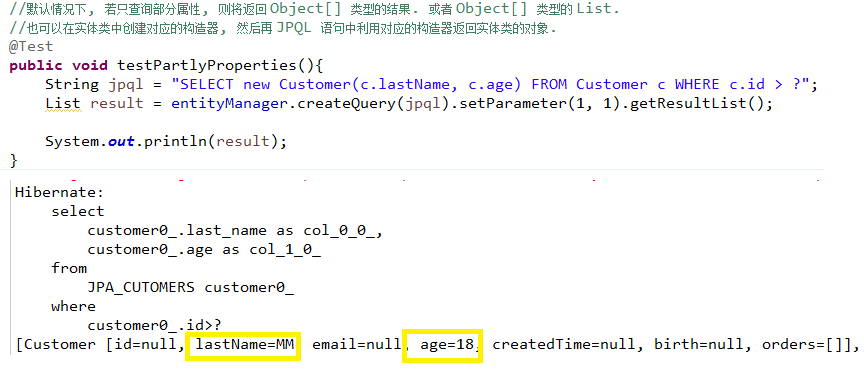
List result = entityManager.createQuery(jpql).setParameter(1, 1).getResultList();

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

}

实体类添加构造方法





## createNamedQuery

## createNativeQuery（sql）