链滴

JPA 中的复杂查询

作者：HuanYuanHe

原文链接：https://ld246.com/article/1575277955734

来源网站：链滴

JPQL全称Java Persistence Query Language

基于首次在EJB2.0中引入的EJB查询语言(EJB QL),Java持久化查询语言(JPQL)是一种可移植的查询语，旨在以面向对象表达式语言的表达式，将SQL语法和简单查询语义绑定在一起·使用这种语言编写的询是可移植的，可以被编译成所有主流数据库服务器上的SQL。

其特征与原生SQL语句类似，并且完全面向对象，通过类名和属性访问，而不是表名和表的属性。

1.

```
//查询所有客户
 @Test
 **public**  **void** findAll() {
EntityManager em = **null**;
EntityTransaction tx = **null**;
 **try** {
     //获取实体管理对象
     em = JPAUtil.getEntityManager();
     //获取事务对象
     tx = em.getTransaction();
     tx.begin();
     // 创建query对象
    String jpql = "from Customer";
    Query query = em.createQuery(jpql);
     // 查询并得到返回结果
    List list = query.getResultList(); // 得到集合返回类型
     **for** (Object object : list) {
     System.out.println(object);
 }
 tx.commit();
} **catch** (Exception e) {
 // 回滚事务
```

```
 tx.rollback();
 e.printStackTrace();
} **finally** {
 // 释放资源
 em.close();
 }
 }
```

2.分页查询客户

```
 @Test
 **public**  **void** findPaged () {
EntityManager em = **null**;
EntityTransaction tx = **null**;
 **try** {
 //获取实体管理对象
    em = JPAUtil.getEntityManager();
 //获取事务对象
      tx = em.getTransaction();
      tx.begin();


     //创建query对象
    String jpql = "from Customer";
    Query query = em.createQuery(jpql);
     //起始索引
     query.setFirstResult(0);
     //每页显示条数
     query.setMaxResults(2);
     //查询并得到返回结果
```

```
    List list = query.getResultList(); //得到集合返回类型
     for (Object object : list) {
      System.out.println(object);
 }
     tx.commit();
} **catch** (Exception e) {
 // 回滚事务
     tx.rollback();
     e.printStackTrace();
} **finally** {
 // 释放资源
     em.close();
 }
 }
```

3.条件查询

```
//条件查询
 @Test
 **public**  **void** findCondition () {
    EntityManager em = **null**;
    EntityTransaction tx = **null**;
     **try** {
     //获取实体管理对象
     em = JPAUtil.getEntityManager();
     //获取事务对象
     tx = em.getTransaction();
     tx.begin();
    //创建query对象
```

```
    String jpql = "from Customer where custName like ? ";
    Query query = em.createQuery(jpql);
     //对占位符赋值，从1开始
     query.setParameter(1, "huanyuan%");
     //查询并得到返回结果
    Object object = query.getSingleResult(); //得到唯一的结果集对象
     System.out.println(object);
     tx.commit();
    } **catch** (Exception e) {
     // 回滚事务
     tx.rollback();
     e.printStackTrace();
    } **finally** {
     // 释放资源
     em.close();
     }
}
```

4.

```
//根据客户id倒序查询所有客户
 //查询所有客户
 @Test
 **public**  **void** testOrder() {
    EntityManager em = **null**;
    EntityTransaction tx = **null**;
     **try** {
     //获取实体管理对象
     em = JPAUtil.getEntityManager();
```

```
    //获取事务对象
    tx = em.getTransaction();
    tx.begin();
    // 创建query对象
   String jpql = "from Customer order by custId desc";
   Query query = em.createQuery(jpql);
    // 查询并得到返回结果
   List list = query.getResultList(); // 得到集合返回类型
    **for** (Object object : list) {
    System.***out***.println(object);
    }
    tx.commit();
} **catch** (Exception e) {
    // 回滚事务
    tx.rollback();
    e.printStackTrace();
} **finally** {
 // 释放资源
    em.close();
 }
 }
```

5. 统计查询

```
//统计查询
 @Test
 **public**  **void** findCount() {
   EntityManager em = **null**;
   EntityTransaction tx = **null**;
```

```
 **try** {
     //获取实体管理对象
     em = JPAUtil.getEntityManager();
     //获取事务对象
     tx = em.getTransaction();
     tx.begin();
     // 查询全部客户
     // 1.创建query对象
    String jpql = "select count(custId) from Customer";
    Query query = em.createQuery(jpql);
     // 2.查询并得到返回结果
    Object count = query.getSingleResult(); // 得到集合返回类型
     System.***out***.println(count);
     tx.commit();
} **catch** (Exception e) {
 // 回滚事务
     tx.rollback();
     e.printStackTrace();
} **finally** {
 // 释放资源
     em.close();
 }
 }
```