hibernate的批量处理

hibernate完全以面向对象的方式来操作数据库，当程序里以面向对象的方式操作持久化对象，将被自动转换为对数据库的操作。但存在这样一个问题，如果我们想同时更新100000条记录，是不是要逐一加载100000条记录，然后依次调用setter方法。采用这种方法实在是太繁琐了，而且严重影响了数据的访问性能。Hibernate提供的批量处理的方案来解决这个问题。

# 批量插入

|  |
| --- |
| #如果我们需要插入100000条记录，通过hibernate可能会采用如下的方式处理：  Session session = HibernateUtil.getSession();  Transaction tx = session.beginTransaction();  for (int i = 0; i < 100000; i++) {  User user = new User();  user.setName("userName"+i);  user.setAge(i);  session.save(session);  }  tx.commit();  session.close();  但是这个程序存在一个问题：当程序运行到某个地方，总是会抛出OutOfMemoryException内存溢出异常。这是因为Hibernate的Session持有一个必选的一级缓存，所有的User实例都会Session级别的缓存区进行缓存的缘故。  如果你要执行批量处理并且想要达到一个理想的性能，那么使用JDBC的批量（batching）功能是至关重要。将JDBC的批量抓取数量（batch size）参数设置到一个合适值。  <property name="jdbc.batch\_size">20</property>  可能也会想到关闭二级缓存：  hibernate.cache.use\_second\_level\_cache false |
| 但这个并不是必须的。解决这个问题的方案就是：定时的将session的缓存数据刷入数据库，同时通过调用 clear() 来控制一级缓存的大小。如下  static void addUser() throws Exception{  Session session = HibernateUtil.getSession();  Transaction tx = session.beginTransaction();  //循环1000次，插入1000条记录  for(int i = 0;i < 100000;i++){  //创建User对象  User user = new User();  user.setName("userName"+i);  user.setAge(i);    //在session级别缓存User实例  session.save(user);    //每当累加器是20的倍数的时候，将session中数据刷入数据库，并且情况session缓存  if(i%20==0){  session.flush();  session.clear();  }  } |

# DML风格的批量更新/删除

Hibernate提供通过Hibernate查询语言（HQL）来执行大批量SQL风格的DML语句的方法。 批量update、delete语句的语法格式如下：

( UPDATE | DELETE ) FROM? EntityName (WHERE where\_conditions）。

这里需要注意如下几点：

* 在FROM子句中，FROM关键字是可选的。
* 在FROM子句（from-clause）中只能有一个实体名，它可以是别名。如果实体名是别名，那么任何被引用的属性都必须加上此别名的前缀；如果不是别名，那么任何有前缀的属性引用都是非法的。
* 不能在批量HQL语句中使用连接，显式或者隐式的都不行。不过在WHERE子句中可以使用子查询。可以在where子句中使用子查询，子查询本身可以包含join。
* 整个WHERE子句是可选的。

|  |
| --- |
| static void updateUser() throws Exception{  Session session = HibernateUtil.getSession();  Transaction tx = session.beginTransaction();    //定义批量更新的HQL语句  String hql = "update User as user set user.name = :name";  //执行更新  session.createQuery(hql).setString("name", "name").executeUpdate();    tx.commit();  session.close();  }  使用这种批量更新语法时，通常只需要执行一次SQL的update语句，就可以完成所有满足条件记录的更新。但也有可能需要执行多条update语句，这是因为有继承映射等情况。 |
| 执行一个HQL DELETE，同样使用 Query.executeUpdate() 方法:  static void deleteUser() throws Exception{  Session session = HibernateUtil.getSession();  Transaction tx = session.beginTransaction();  //定义批量更新的HQL语句  String hql = "delete User";    //执行删除 ,返回一个影响行数的整数值  session.createQuery(hql).executeUpdate();  tx.commit();  session.close();  } |