# 1.重新到了one\_to\_many中

## 1.department 中如果lazy =false;

在查询学生的时候会将有关学生的信息全部查询，比如department

1.2如果lazy = true 懒得加载

就不会去查询department ，但是当我们通过student去查询一个部门的的时候会报错

System.*out*.println(student1.getName()+" 所在的系"

+student1.getDept().getName());

2. 但是对于student来说不管有没有懒加载，都会自动去调用 department中的数据

但是假如我们的session关闭的时候就不会执行了，还会报错，‘

Department department = (Department) session.get(Department.**class**, 2);

session.close();

Set<StudentMany> set = department.getStudentManys();

**for**(StudentMany s:set ){

System.*out*.println(s.getName());

}

# 2.我们希望尽量减少hql 语句，这个就是需要保证session不要快速的自动关闭，如果使用下面的我们就不要关闭session了

**public** **class** MyFilter1 **extends** HttpServlet **implements** Filter {

**public** **void** doFilter(ServletRequest arg0, ServletResponse arg1,

FilterChain arg2) **throws** IOException, ServletException {

// **TODO** Auto-generated method stub

Session s=**null**;

Transaction tx=**null**;

**try** {

s =HibernateUtil.*getCurrentSession*();

tx=s.beginTransaction();

// 假设有了这个doFilter 有了事物 会先在前面跑，

// 当我们看到ok的的时候整个请求都会往回返，

// 在一传工作全部完成之后，这个session关闭的情况不能像从前。看下面的finally

arg2.doFilter(arg0, arg1);

//System.out.prrintln("ok")

//System.out.println("ok");x.

tx.commit();

} **catch** (Exception e) {

**if**(tx!=**null**){

tx.rollback();

}

**throw** **new** RuntimeException(e.getMessage());

// **TODO**: handle exception

}**finally**{

HibernateUtil.*closeCurrentSession*();

}

}

**public** **void** init(FilterConfig arg0) **throws** ServletException {

// **TODO** Auto-generated method stub

}

}

# 3。然后我们修改下工具中的更新和删除的方法到时候session中的提交在 过滤器中，就是说整个过程中我们的session是打开的，

## 缺点，就是回话关闭延迟来了

//如果要配置opensissionInView

//同意的更新 和删除， **public** **static** **void** executeUpdateOpenInView(String hql,String [] parameters){

Session s=*getCurrentSession*();

Query query=s.createQuery(hql);

//���ж��Ƿ��в���Ҫ��

# if(parameters!=null&& parameters.length>0){

**for**(**int** i=0;i<parameters.length;i++){

query.setString(i, parameters[i]);

}

}

query.executeUpdate();

}