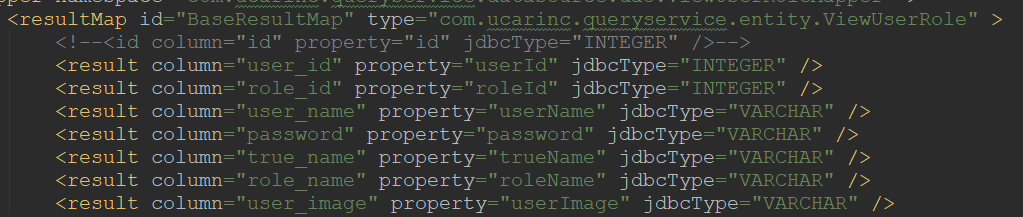
# SSM问题笔记日常记录

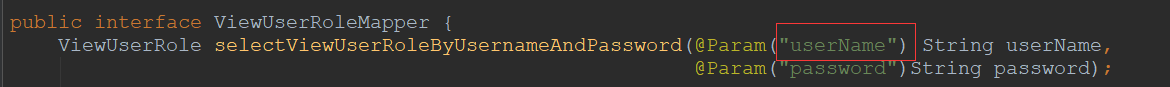
## 一、MyBatis知识点

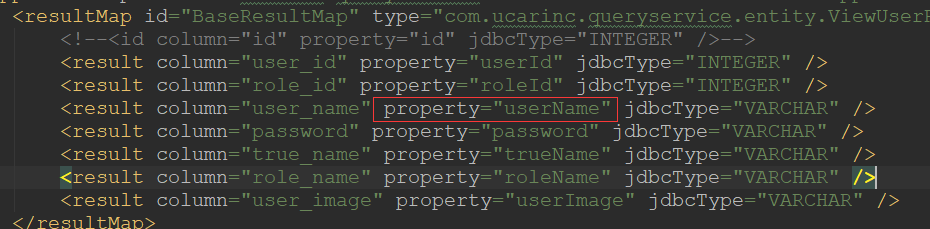
1.视图操作与一般的表一样处理，在xml中少了<id/>映射。

在设计视图时一定要全面考虑将来需要用到的数据，以免重复劳动。（确定所需列数）



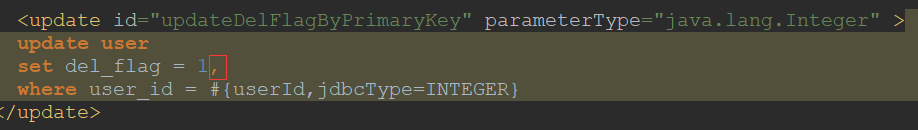
2.在多参数下，注意要将mapper接口使用“@Param(参数)”注解，并且参数要与映射xml文件中的参数名相同。（mapper多参数问题）



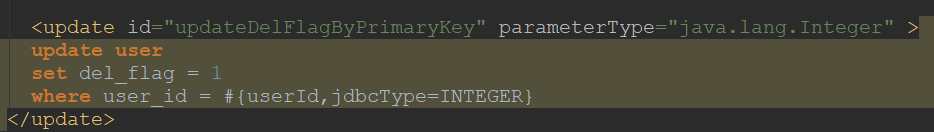


3.xml配置时，注意细节格式，此处如图所示无逗号。

错误：



正确：



4.模糊查询时，其xml配置为：

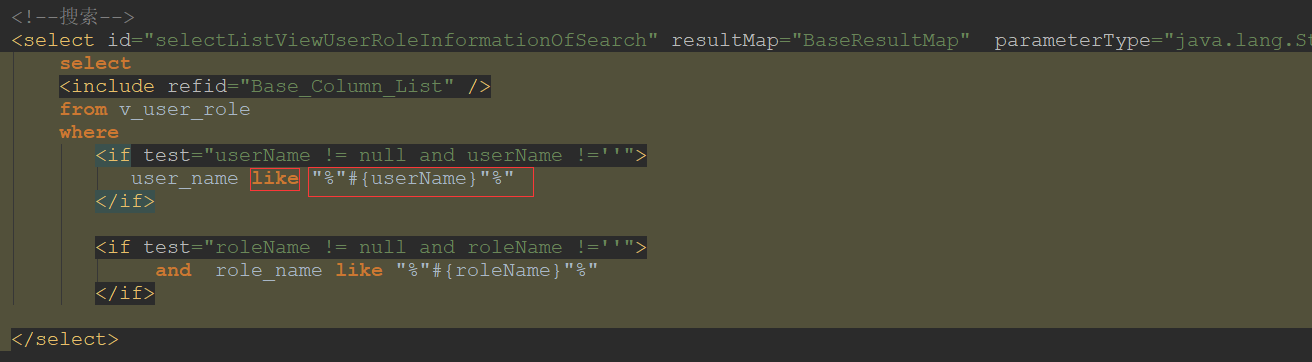
<if test="userName != null and userName !=''">

user\_name like ‘%’#{userName}‘%’

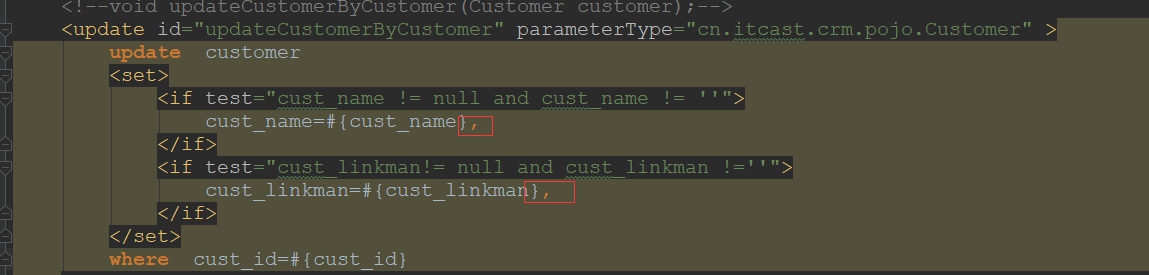
或者user\_name like ‘%#{userName}%’

</if>

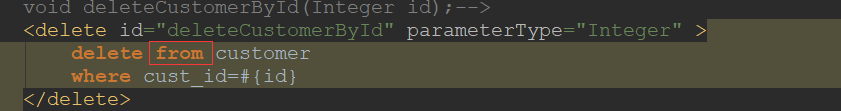
如图所示：



5.书写更新操作时，<set>标签内没有写逗号



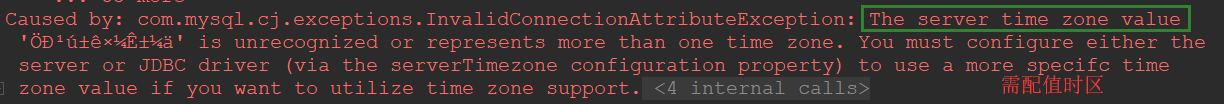
6.书写delete操作是，必须有from



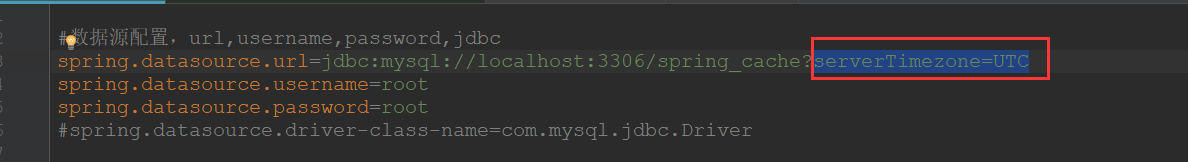
7.数据源配置问题

1）时区问题，在数据源url后添加serverTimezone=UTC

问题：



解决：



8.delete后忘了加from

delete from employee where id=#{id}

## 二、iBaits

1.当查询List<Model>时，返回结果集合“resultClass=Model”,Dao层调用用queryForList()。

2.操作数据库时注意时间问题，默认为00.00.00 00:00映射到结果时会报错。

原因：空时间在映射到结果时会出现转换错误

解决方法：测试时，在数据库里修改00.00.00 00:00为其他时间，不要空就行。

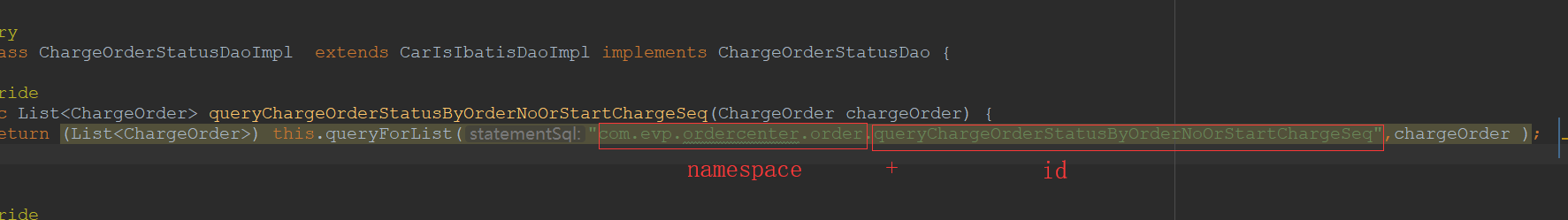
3.Dao接口调用对应的sql中的id问题。

StatementSql语句=namespace+id

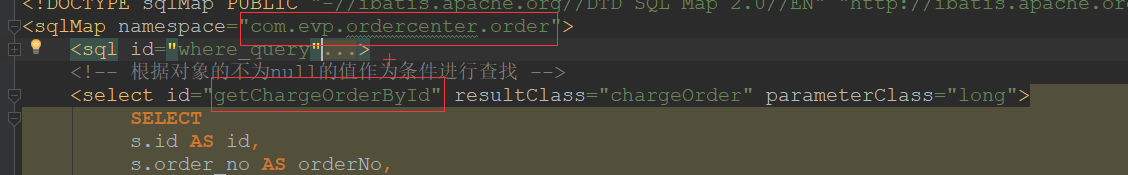
例如：

update("com.evp.ordercenter.order.updateChargeOrderStatusByOrderNoOrStartChargeSeq", record);

参考理解：图一图二



图一：DaoImpl层



图二：sql.xml

4. dynamic动态sql查询

方式一：定义引用**dynamic**的方式

<!—1.定义动态sql-->

<sql id="where\_query">  
 <dynamic prepend="where">  
 <isNotEmpty property="id" prepend="and">  
 s.id = #id#  
 </isNotEmpty>  
 <isNotEmpty property="orderNo" prepend="and">  
 s.order\_no = #orderNo#  
 </isNotEmpty>  
 <isNotEmpty property="startChargeSeq" prepend="and">  
 s.start\_charge\_seq = #startChargeSeq#  
 </isNotEmpty>

</dynamic>

</sql>

<!—2.动态sql查询-->  
<select id="queryChargeOrder" resultClass="chargeOrder" parameterClass="chargeOrder">  
 SELECT  
 s.id AS id,  
 s.order\_no AS orderNo,  
 s.start\_charge\_seq AS startChargeSeq,  
 s.order\_status AS orderStatus,  
 s.plate\_id AS plateId,  
 s.user\_id AS userId,  
 s.operator\_id AS operatorId,  
 s.city\_id AS cityId,  
 s.equipment\_id AS equipmentId,  
 s.city\_name AS cityName,  
 s.station\_id AS stationId,  
 s.connector\_id AS connectorId,  
 s.place\_time AS placeTime,  
 s.start\_fail\_reason AS startFailReason,  
 s.stop\_reason AS stopReason,  
 s.current\_quantity AS currentQuantity,  
 s.before\_quantity AS beforeQuantity,  
 s.after\_quantity AS afterQuantity,  
 s.change\_quantity AS changeQuantity,  
 s.charge\_fee AS chargeFee,  
 s.service\_fee AS serviceFee,  
 s.reduce\_fee AS reduceFee,  
 s.total\_fee AS totalFee,  
 s.pay\_status AS payStatus,  
 s.charge\_start\_time AS chargeStartTime,  
 s.charge\_end\_time AS chargeEndTime,  
 s.scan\_no AS scanNo,  
 s.create\_time AS createTime,  
 s.modify\_time AS modifyTime  
 FROM t\_charge\_order s  
 <include refid="where\_query"/><!—引用-->  
</select>

方式二：将dynamic直接写入语句的方式

<select id="queryChargeOrder" resultClass="chargeOrder" parameterClass="chargeOrder">  
 SELECT  
 s.id AS id,  
 s.order\_no AS orderNo,  
 s.start\_charge\_seq AS startChargeSeq,  
 s.order\_status AS orderStatus,  
 s.plate\_id AS plateId,  
 s.user\_id AS userId,  
 s.operator\_id AS operatorId,  
 s.city\_id AS cityId,  
 s.equipment\_id AS equipmentId,  
 s.city\_name AS cityName,  
 s.station\_id AS stationId,  
 s.connector\_id AS connectorId,  
 s.place\_time AS placeTime,  
 s.start\_fail\_reason AS startFailReason,  
 s.stop\_reason AS stopReason,  
 s.current\_quantity AS currentQuantity,  
 s.before\_quantity AS beforeQuantity,  
 s.after\_quantity AS afterQuantity,  
 s.change\_quantity AS changeQuantity,  
 s.charge\_fee AS chargeFee,  
 s.service\_fee AS serviceFee,  
 s.reduce\_fee AS reduceFee,  
 s.total\_fee AS totalFee,  
 s.pay\_status AS payStatus,  
 s.charge\_start\_time AS chargeStartTime,  
 s.charge\_end\_time AS chargeEndTime,  
 s.scan\_no AS scanNo,  
 s.create\_time AS createTime,  
 s.modify\_time AS modifyTime  
 FROM t\_charge\_order s  
 <dynamic prepend="where">  
 <isNotEmpty property="orderNo" prepend="and">  
 s.order\_no = #orderNo#  
 </isNotEmpty>  
 <isNotEmpty property="startChargeSeq" prepend="and">  
 s.start\_charge\_seq = #startChargeSeq#  
 </isNotEmpty>  
 </dynamic>

</select>

5. dynamic动态sql更新

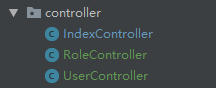
<!--通过本地或第三方订单编号更新订单-->  
 <update id="updateChargeOrderStatusByOrderNoOrStartChargeSeq" parameterClass="chargeOrder">  
 update t\_charge\_order

🡨-set语句—>  
<dynamic prepend="set">  
 <isNotEmpty property="id" prepend=",">  
 id = #id#  
 </isNotEmpty>  
 <isNotEmpty property="orderNo" prepend=",">  
 order\_no = #orderNo#  
 </isNotEmpty>  
 <isNotEmpty property="startChargeSeq" prepend=",">  
 start\_charge\_seq = #startChargeSeq#  
 </isNotEmpty>  
 <isNotEmpty property="orderStatus" prepend=",">  
 order\_status = #orderStatus#  
 </isNotEmpty>  
 <isNotEmpty property="plateId" prepend=",">  
 plate\_id = #plateId#  
 </isNotEmpty>  
 <isNotEmpty property="userId" prepend=",">  
 user\_id = #userId#  
 </isNotEmpty>  
 <isNotEmpty property="operatorId" prepend=",">  
 operator\_id = #operatorId#  
 </isNotEmpty>  
 <isNotEmpty property="cityId" prepend=",">  
 city\_id = #cityId#  
 </isNotEmpty>  
 <isNotEmpty property="equipmentId" prepend=",">  
 equipment\_id = #equipmentId#  
 </isNotEmpty>  
 <isNotEmpty property="cityName" prepend=",">  
 city\_name = #cityName#  
 </isNotEmpty>  
 <isNotEmpty property="stationId" prepend=",">  
 station\_id = #stationId#  
 </isNotEmpty>  
 <isNotEmpty property="connectorId" prepend=",">  
 connector\_id = #connectorId#  
 </isNotEmpty>  
 <isNotEmpty property="placeTime" prepend=",">  
 place\_time = #placeTime#  
 </isNotEmpty>  
 <isNotEmpty property="startFailReason" prepend=",">  
 start\_fail\_reason = #startFailReason#  
 </isNotEmpty>  
 <isNotEmpty property="stopReason" prepend=",">  
 stop\_reason = #stopReason#  
 </isNotEmpty>  
 <isNotEmpty property="currentQuantity" prepend=",">  
 current\_quantity = #currentQuantity#  
 </isNotEmpty>  
 <isNotEmpty property="beforeQuantity" prepend=",">  
 before\_quantity = #beforeQuantity#  
 </isNotEmpty>  
 <isNotEmpty property="afterQuantity" prepend=",">  
 after\_quantity = #afterQuantity#  
 </isNotEmpty>  
 <isNotEmpty property="changeQuantity" prepend=",">  
 change\_quantity = #changeQuantity#  
 </isNotEmpty>  
 <isNotEmpty property="chargeFee" prepend=",">  
 charge\_fee = #chargeFee#  
 </isNotEmpty>  
 <isNotEmpty property="serviceFee" prepend=",">  
 service\_fee = #serviceFee#  
 </isNotEmpty>  
 <isNotEmpty property="reduceFee" prepend=",">  
 reduce\_fee = #reduceFee#  
 </isNotEmpty>  
 <isNotEmpty property="totalFee" prepend=",">  
 total\_fee = #totalFee#  
 </isNotEmpty>  
 <isNotEmpty property="payStatus" prepend=",">  
 pay\_status = #payStatus#  
 </isNotEmpty>  
 <isNotEmpty property="chargeStartTime" prepend=",">  
 charge\_start\_time = #chargeStartTime#  
 </isNotEmpty>  
 <isNotEmpty property="chargeEndTime" prepend=",">  
 charge\_end\_time = #chargeEndTime#  
 </isNotEmpty>  
 <isNotEmpty property="scanNo" prepend=",">  
 scan\_no = #scanNo#  
 </isNotEmpty>  
 <isNotEmpty property="createTime" prepend=",">  
 create\_time = #createTime#  
 </isNotEmpty>  
 <isNotEmpty property="modifyTime" prepend=",">  
 modify\_time = #modifyTime#  
 </isNotEmpty>  
</dynamic>

🡨-where语句—>  
<dynamic prepend="where">  
 <isNotEmpty property="orderNo" prepend="and">  
 order\_no = #orderNo#  
 </isNotEmpty>  
 <isNotEmpty property="startChargeSeq" prepend="and">  
 start\_charge\_seq = #startChargeSeq#  
 </isNotEmpty>  
</dynamic>  
 </update>

## 三、springMVC

1.同样的功能（此处使用的是登录方法）写在IndexController下可以，但是写在UserController中不行？

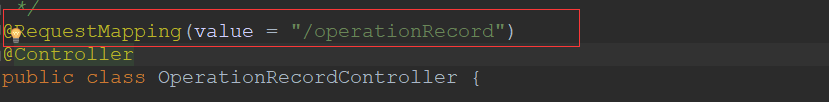


2.访问不到contolller，在配置上可能有以下2中原因

1）在web.xml中没有配置前端控制器DispatcherServlet类（在servlet标签中）

2）在spingMCV.xml中没有配置扫描组件<context:component-scan>或注解驱动<mvc:annotation-driven>

3.编写controller时，为了防止和他人的方法重名，在注解上，除了对方法注解外，需对类注解



4.关于获取不到前端参数问题，在注解上可能有以下几种方式解决。

1）注解@RequestBody，在参数处



1. 注解@Parameter()

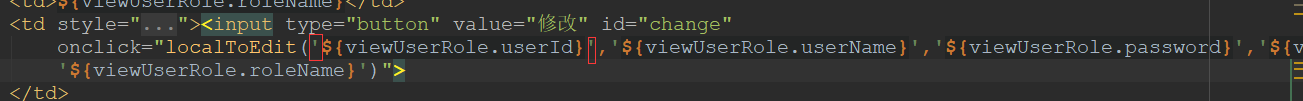
5.serviceImpl忘了implements service接口

6.serviceImpl类忘了注解@Service

## 四、spring

## 五.JSP&EL&JS&JQuery

1.JS函数传参又多个参数时，需要使用单引号处理。

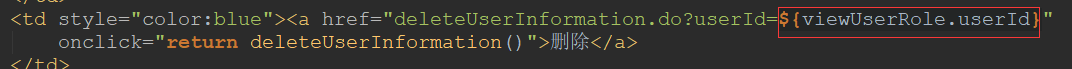


2从前端通过“.do”来加后缀传参为什么不能传对象，而只能穿具体的值

错误：传对象



正确：传值



问题：如果后端直接想获取对象，前端该如何处理？？springMVC直接可以？还是必须在form表单中写上对应的id与name？

3.关于<c:if> 标签tset=“”书写规则，已经错了好几次(谨记)。

正确：



错误：



4.关于EL表达式取值时一定注意与实体entity一致(易错)。

5．使用Jquery时需注意以下几点。

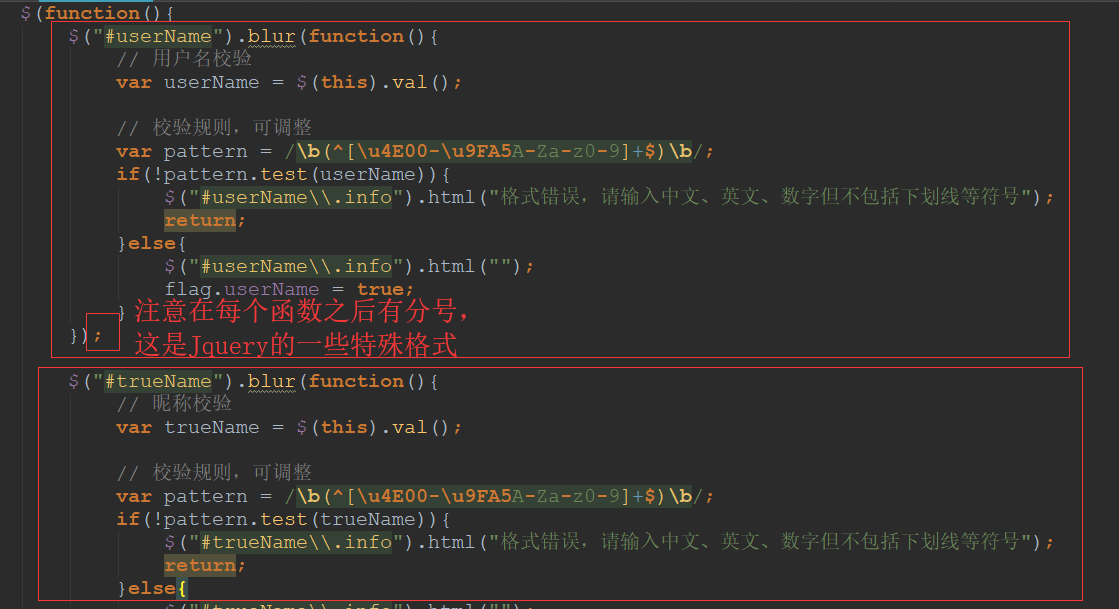
（1）包的引入 注意地址



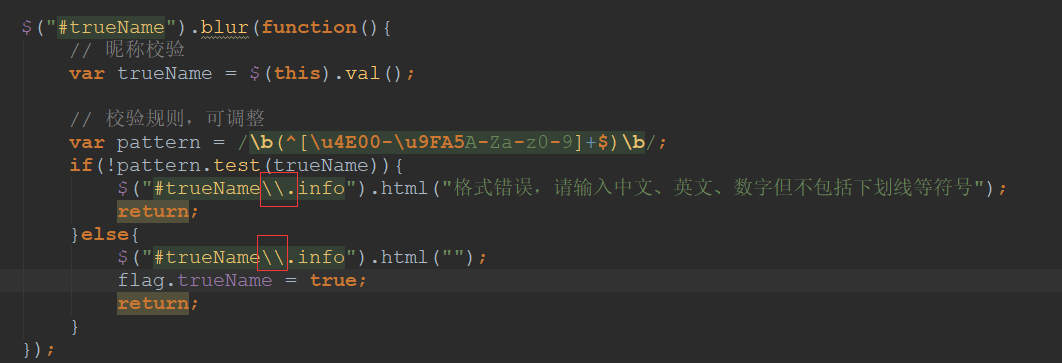
（2）$(function(){});

末尾有分号

（3）每个小的函数之后都包含有分号

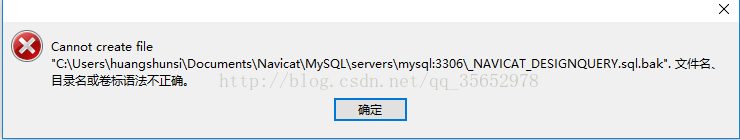


（4）如下所示，不知为何需要在id间加入“\\”才能成功,猜测可能是因为id中包含逗号，需要“\\”来作转义符。



## 五.数据库

1. Navicat 连接MySQL数据不能新建查询，提示文件名、目录名或卷标语法不正确。



原因：

出现该问题的原因的是数据库连接名出现了：分号，而Windows不允许文件名中包含分号，导致Navicat不能创建该目录；

解决方法：

删除原来的连接，新建一个，并且连接名只能包括字母数字下划线等等Windows允许的文件名即可

2.操作数据库时注意时间问题，默认为00.00.00 00:00映射到结果时会报错。

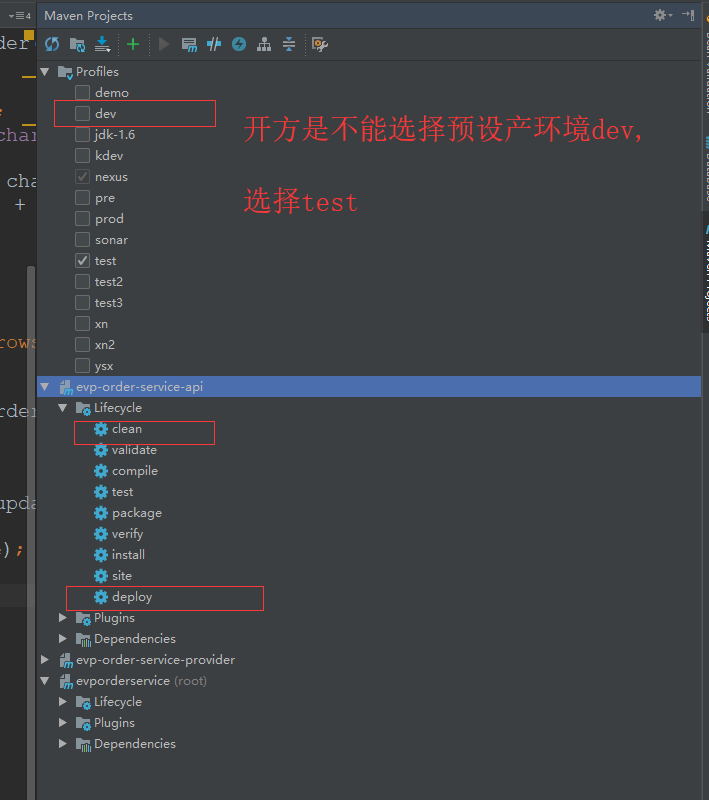
原因：空时间在映射到结果时会出现转换错误

解决方法：测试时，在数据库里修改00.00.00 00:00为其他时间，不要空就行。

## 六、Maven

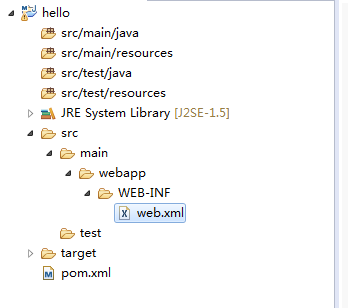
1. 当重新部署环境后，需要clean与deloy

在开发时，选择test环境，而不选dev预生产环境



2.处理红色叉号

**手动**在webapp文件夹下创建一个WEB-INF文件夹，在里面放一个web.xml文件



3、处理编译版本

在pom.xml中添加如下代码

<build>

<!-- 配置了很多插件 -->

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.5.1</version>

<configuration>

<source>1.7</source>

<target>1.7</target>

<encoding>UTF-8</encoding>

</configuration>

</plugin>

</plugins>

</build>

## 七、git版本控制

1.提交代码时出现手动merge情况，导致自己和别人的代码缺失？

原因：git提交时与他人发生冲突

解决：git前先更新(pull)，再执行commit、push

2. 工作区处于冲突状态，无法在进行提交操作。

解决办法有两个:

一是使用git reset放弃合并操作;

二是打开冲突的文件，进行手工编辑解决冲突的部分，之后再进行提交。

## 八、远程service层

1．入参出参校验问题，如何全面校验？

*\* <p>1. 检测参数是否异常，如果异常，则返回错误并打印日志</p>  
\* <p>2. 组装查询参数</p>  
\* <p>3. 检测订单是否存在，如果不存在，则返回错误并打印日志</p>  
\* <p>4. 判断订单状态是否合法，如果合法则返回合法的订单状态，如果不合法则返回错误并打印日志</p>  
\*/*

具体代码示例：

@Override

public Result<ChargeOrderOrderStatusReDTO> queryChargeOrderStatus(ChargeOrderOrderStatusDTO dto) {  
 //1.检查参数是否正常，如果异常，则返回错误并答打印日志  
 logger.info("ChargeOrderOrderStatusDTO"+dto);  
 if(dto==null){  
 logger.error(*CHAGRE\_COMMON\_PARAM\_ERROR*.getMsg());  
 return Result.*getServiceError*(*CHAGRE\_COMMON\_PARAM\_ERROR*.getMsg(), *CHAGRE\_COMMON\_PARAM\_ERROR*.getCode());  
 }  
  
 //2.组装查询参数，如果订单编号不存在(注意：此处只要存在本地或第三方订单任意一个即可)，则返回错误信息  
 List<ChargeOrder> chargeOrderList=null;  
 if((dto.getOrderNo() != null&& !"".equals(dto.getOrderNo()))||(dto.getStartChargeSeq() != null&& !"".equals(dto.getStartChargeSeq()))){  
 ChargeOrder chargeOrder=new ChargeOrder();  
 if(dto.getOrderNo() != null&& !"".equals(dto.getOrderNo())){  
 chargeOrder.setOrderNo(dto.getOrderNo());  
 }  
 if(dto.getStartChargeSeq() != null&& !"".equals(dto.getStartChargeSeq())){  
 chargeOrder.setStartChargeSeq(dto.getStartChargeSeq());  
 }  
 chargeOrderList = chargeOrderStatusService.queryChargeOrderStatus(chargeOrder);  
 }  
  
 //3.检查订单是否存在，如果不存在，则返回错误日志并打印  
 if(chargeOrderList==null || chargeOrderList.isEmpty()){  
 logger.error(*CHARGE\_ORDER\_NOT\_EXIST*.getMsg(), dto);  
 return Result.*getBusinessException*(*CHARGE\_ORDER\_NOT\_EXIST*.getMsg(), *CHARGE\_ORDER\_NOT\_EXIST*.getCode());  
 }  
  
 //4.判断订单状态是否合法，如果合法则返回合法的订单状态，如果不合法则返回错误并打印日志  
 ChargeOrderOrderStatusReDTO reDTO=new ChargeOrderOrderStatusReDTO();  
 for(OrderStatusEnum status : OrderStatusEnum.*values*()){  
 if(chargeOrderList.get(0).getOrderStatus().equals(status.getCode())){  
 reDTO.setOrderStatus(status.getCode());  
 reDTO.setDescribe(status.getDescribe());  
 break;  
 }  
 }  
 if(reDTO.getOrderStatus()==null){  
 logger.error(*CHARGE\_STATUS\_NOT\_EXIST*.getMsg(), reDTO);  
 return Result.*getBusinessException*(*CHARGE\_STATUS\_NOT\_EXIST*.getMsg(), *CHARGE\_ORDER\_NOT\_EXIST*.getCode());  
 }  
  
 return Result.*getSuccessResult*(reDTO);  
}

## 九、综合

1.编写功能时，大致总结流程如下：

Jsp——controller(确定功能名称)——xml——mapper接口——service接口、实现——controller

2.每次写一个接口都要写一套(本地service+dao+mapper.xml),特别是有些类似功能代码重复性更高，从而降低开发效率？

原因：本地servie复用性不高

解决：在service层提高入参出参类型(对象)；另外，在ibatis中利用dynamic动态语句。

3.在文件上传时出现的问题：

（一）不能简单有效地获取普通表单项。

在文件上传并不太难，难的是表单中除了提交文件外还要提交其他信息，可能导致获取前端数据的困难，以及如何保持一次请求(调用commonsMultipartResolver.resolveMultipart(httpRequest); 方法时，原request会消亡！)；

当表单<form>标签加入enctype="multipart/form-data"后，不能使用request.getParametre()传值

（二）编写代码的混乱

发现编码十分困难。

后端：既要单独书写文件上传代码，也要书写controller获取其他前端数据，

前端：考虑前端如何写才能让后端

（三）解决方法

直接使用ajax处理解决这些复杂的问题

$.ajax({

id: '',

url : '${\_ctx}/index/addUserInformation.do',//1.do的调用

type: 'post',

data : $('#addFrom').serialize(),//2.将数据直接系列化封装（)

dataType : 'json',

success : function () {

//3.处理返回数据

}

})

## 十、项目启动

1.jndi配置(数据源配置)

Apache-tomcat下的context.xml

Name值；url值



