## 第一天 国际物流 杰信商贸 + 搭建环境

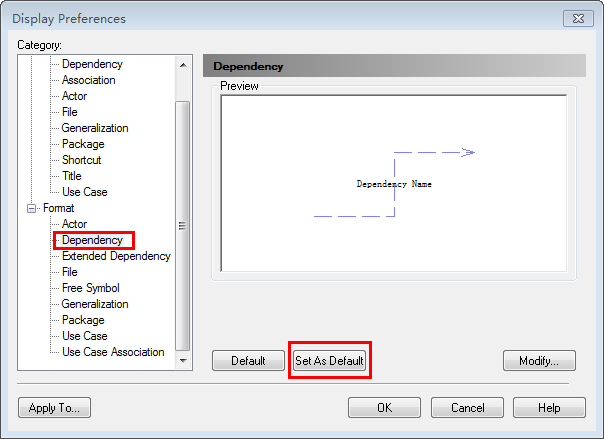
## 项目背景

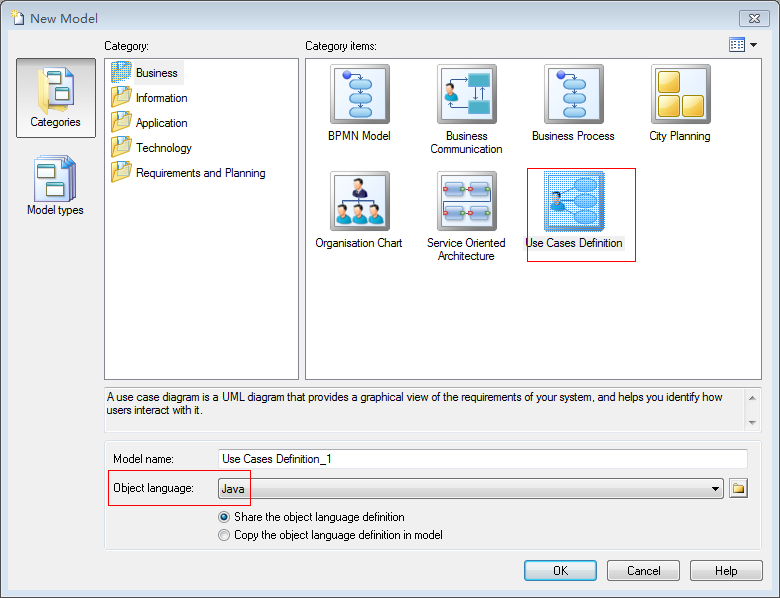
杰信商贸是国际物流行业一家专门从事进出口玻璃器皿贸易的公司。公司总部位于十一个朝代的帝王之都西安，业务遍及欧美。随着公司不断发展壮大，旧的信息系统已无法满足公司的快速发展需求，妨碍公司成长，在此背景下，公司领导决定研发《杰信商贸综合管理平台》。

《杰信商贸综合管理平台》分三期完成。一期完成仓储管理（包括：采购单、仓库、货物、条形码、入库、出库、退货、盘点、库存、库存上限报警、统计查询）和展会管理（包括：展会管理、出单管理），形成货物统一数字化管理。二期完成货运全流程管理，包括购销合同、出货表统计、出口报运单、HOME装箱单、装箱单、委托书、发票、财务统计等。三期完成决策分析（包括：成本分析图、销售情况统计、重点客户、经营情况同期比对统计、工作绩效），为公司经营决策提供数据支持。

## 了解系统概貌

## PowerDesinger 15.1 初始配置，设置连接线拐弯线改成直线





选择java，可以产生java伪代码，实际开发中不实用

## 用例图

画图3招：

1） 角色，一类用户

2） 用例，业务功能

3） 角色和用例的连线，角色都干什么事情



## 了解系统是所有功能模块

利用visio画系统功能结构图，目的让开发人员了解都有哪些业务模块，方便分工合作，根据每个页面模块的难度，复杂，投入的人力情况来衡量每个模块多少钱。验收标准。

数据库建模，业界都使用PowerDesinger 建模

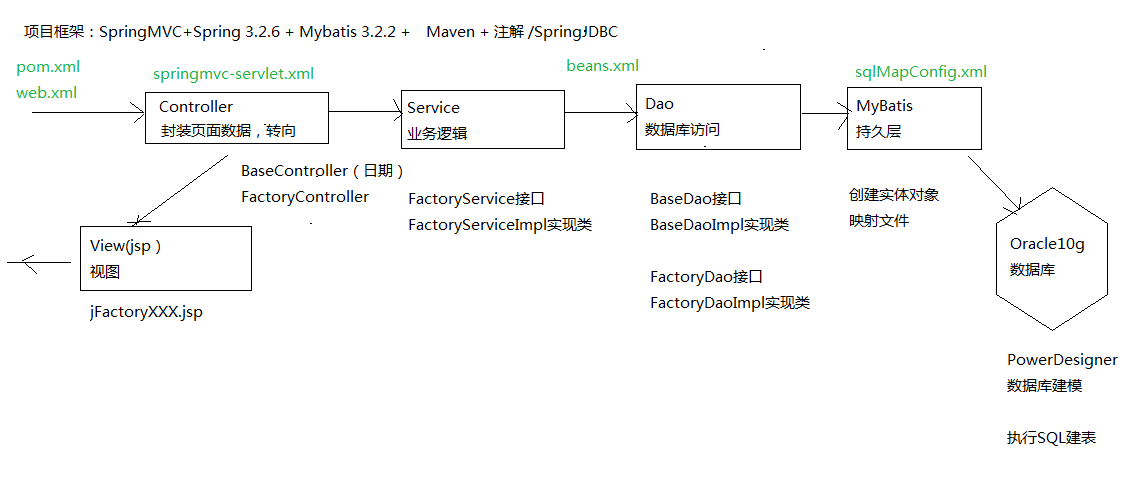
## 【面试】UML、ER、数据库模型的区别

UML 用例图，类图（对应类），序列图，状态图 rose （只有大公司做，ISO要求）

ER 实体和实体的关系（对应domain实体类，配置实体之间的关系）（被模型替代）

模型 （产生数据库表，表之间的关系）（必须）

## 系统架构



## 业务：生产厂家

## 功能

在货物模块和附近模块，它们都有个生产厂家，讲生产厂家信息提出，成为一个单独的表。

## 需求调研《需求说明书》

生产厂家表 FACTORY\_C(\_C业务后缀)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 名称 | 代码 | 是否  主键 | 类型（长度） | 说明 |
|  | 编号 | FACTORY\_ID | 是 | VARCHAR2（40） | UUID |
|  | 全称 | FULL\_NAME | 否 | VARCHAR2（200） | 对应客户比较模糊，翻倍2到4倍 |
|  | 简称 | FACTORY\_NAME | 否 | VARCHAR2(50) |  |
|  | 联系人 | CONTACTS | 否 | VARCHAR2(30) | 人名20/30 |
|  | 电话 | PHONE | 否 | VARCHAR2(20) |  |
|  | 手机 | MOBILE | 否 | VARCHAR2(20) |  |
|  | 传真 | FAX | 否 | VARCHAR220) |  |
|  | 验货员 | INSPECTOR | 否 | VARCHAR2(30) |  |
|  | 排序号 | ORDER\_NO | 否 | INT |  |
|  | 备注 | CNOTE | 否 | VARCHAR2(500) | 如果感觉它可能和关键字冲突，给它加个C前缀 |
|  | 创建人 | CREATE\_BY | 否 | VARCHAR2（40） | 权限字段，登陆人ID |
|  | 创建部门 | CREATE\_DEPT | 否 | VARCHAR2（40） | 登陆人所在部门的ID |
|  | 创建时间 | CREATE\_TIME | 否 | TIMESTAMP |  |

## 【面试】主键INT/LONG/STRING

INT/LONG 都是一般表示自增，根据业务记录数来决定

STRING UUID，业务自己定义的规则xxyyzz，每两位为一层，省市县，部门

## 概要设计《概要设计说明书》

进一步细化功能说明

初步业务逻辑，部分内容由其他内容加工而得；

初步数据库建模

## 详细设计《详细设计说明书》

功能进一步细化

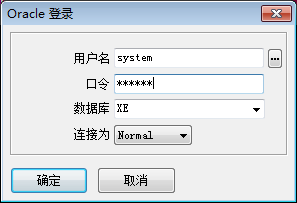
业务逻辑接近最终代码

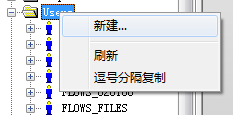
类图、序列图

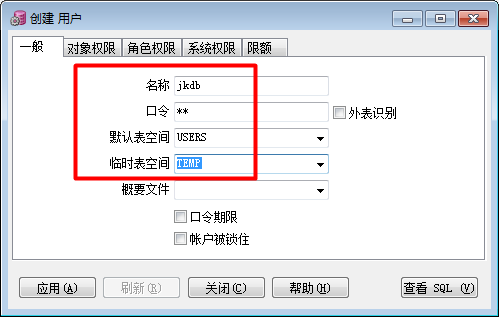
## Oracle创建数据库账号

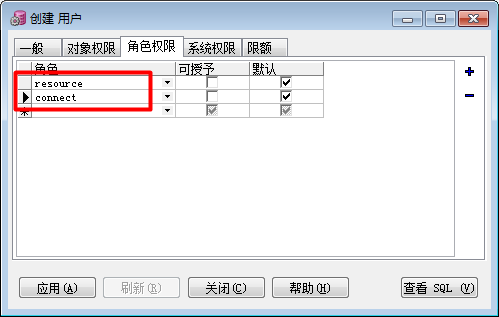
安装Oracle10g EX版本，它会自动启动HTTP服务，会占用8080

使用system账号登陆，创建新用户



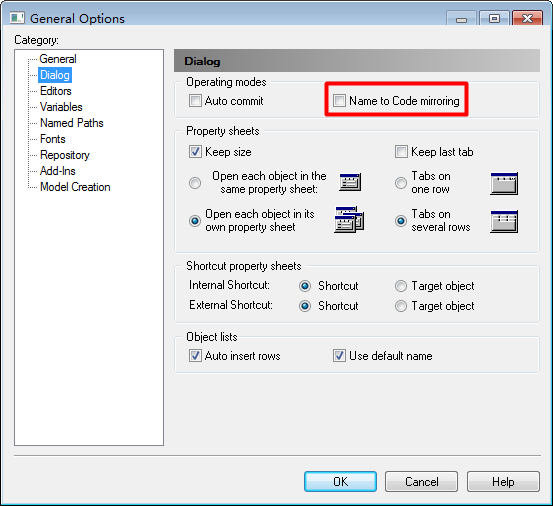




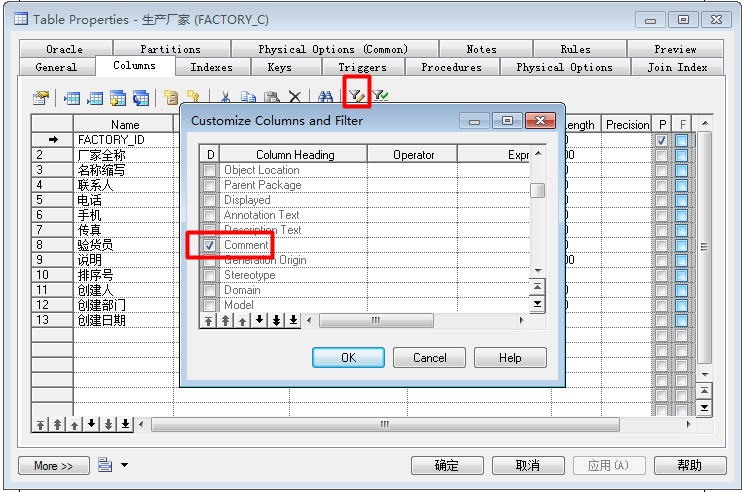


## PD初始化设置

1. 去掉name和code的镜像

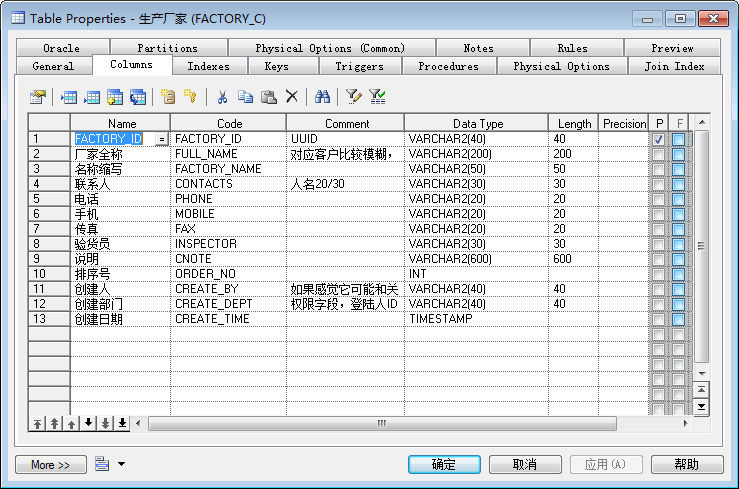


1. 打开备注commend的列



## PD数据库建模





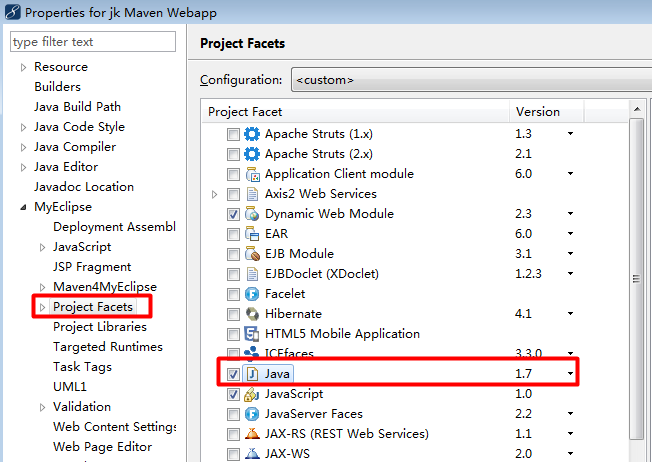
## 创建Maven工程，依赖jar

## Pom文件如何加依赖？

1. spring、springmvc、mybatis 核心的jar包
2. database,log4j 次核心Jar包
3. poi,jfreecharts 第三方jar包

<http://search.maven.org>

maven的案例，demo，现有项目中去找别人写好的依赖，坐标



创建dao层

创建service层

创建controller层

配置文件sqlMapConfig.xml、beans.xml、springmvc-servlet.xml、web.xml

## 帧框架frameset

<html>

<head>

<title>陕西杰信商务综合管理平台</title>

</head>

<frameset rows="125,\*" name="topFrameset" border="0">

<frame name="top\_frame" scrolling="no" target="middleFrameSet" src="title.action">

<frameset cols="202,\*" height="100%" name="middle" frameborder="no" border="0" framespacing="0">

<frame name="leftFrame" class="leftFrame" target="main" scrolling="no" src="left.action" />

<frame name="main" class="rightFrame" src="main.action" />

</frameset>

</frameset>

<noframes>

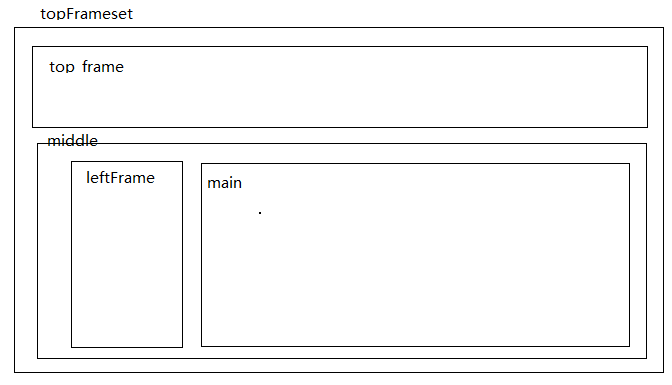
<body>

<p>此网页使用了框架，但您的浏览器不支持框架。</p>

</body>

</noframes>

</html>



## 框架的执行顺序

1）<http://localhost/jk/index.jsp>

2）在Jsp中利用js跳转

<script type="text/javascript">

　　 window.location.href = "home.action"; //javascript页面跳转

</script>

1. 在homeController中跳转，转入登陆页面

@Controller

public class HomeController {

//系统首页模块

@RequestMapping(value={"/home.action"}) //配合web下<url-pattern>/</url-pattern>

public String login(){

return "/index.jsp"; //首页，删除根目录下index.jsp，否则上面url将被拦截进不来

}

1. 在pages/index.jsp中利用按钮提交，fmain.action

@RequestMapping(value="/fmain.action")

public String fmain(){

return "/home/fmain.jsp";

}

## 重点知识回顾:

到软件公司后首先要了解我们公司所在行业，了解行业的特点，了解我们客户它们常用的软件有什么特点，称作业务。只有我们充分了解用户想要的东西，我们才能做的满足客户要求。然后才了解公司的软件架构，以及相关技术。

业务：在某个行业背景下，它的一些习惯性的称谓。

面试时，重点向面试官阐述业务，项目背景，要从项目的业务中提出技术亮点。

搭建系统框架

工具

1）PowerDesigner 业界标准的数据库建模工具

直观的看到表，看到表之间的关系，生成创建表的SQL

1. Oracle 10g EX/11g Oracle数据库，自动创建一个本地服务EX
2. Maven 重要，未来公司发展趋势
3. 帧框架、ajax
4. SpringMVC 3.2.6+Spring + Mybats 3.2.2 Mybats-spring 1.2

## 第二天 生产厂家CRUD + 购销合同

## 业务：启用，停用

功能：和杰信合作生产厂家突然有一天它以次品冒充优质品，杰信不跟这个厂家进行合作。在货物中生产厂家选择框，不能选择这个不再合作的厂家。在系统中不能直接删除这个厂家，因为我们的历史数据中含有对这个生产厂家的引用。软件是采用“伪删除”方式来解决。实际是在数据库表中加入列，作为标识，标识厂家是否可以被引用。0停用1启用。新增厂家默认1，可以修改某个厂家为停用。在货物中选择厂家的下拉列表，只显示为启用。

## 【面试】数据库表是没有boolean，

Mysql tinyint

Oracle number(1)

实际项目中也可以设置为Char(1)

## 加“启用”“停用”功能，修改状态。

可以实现批量操作

<!-- 修改状态 -->

<update id="updateState" parameterType="map">

update FACTORY\_C

set STATE=#{state}

where FACTORY\_ID in

<foreach collection="ids" item="id" open="(" close=")" separator=",">

#{id}

</foreach>

</update>

public void start(String[] ids) {

Map<String, Object> map = new HashMap<String,Object>();

map.put("state", 1); //1启用

map.put("ids", ids);

factoryDao.updateState(map);

}

注：这个方式，就是流程控制的基础，权限控制基础

## 货运管理核心流程：购销合同

## 需求调研过程

软件企业就会到杰信去需求调研，

项目经理，他面对客户的高层中层领导，目的跟客户安排好调研日程。

设计人员，带领初级程序员到具体的业务部门调研。聊天，围绕项目的目标，设计人员会自己记录谈话内容，（主干），初级程序员（所有内容）。回来整理笔记，整理《需求调研报告》

报告中怎么体现纸质报表？初级程序员将纸质报表转化成电子的。

## 购销合同业务分析

## 需求过程

杰信和生产厂家签到的合同，称为购销合同。一个购销合同中包含多个货物信息，一个货物信息包括多个附件信息。

报表打印时，同一个生产厂家的货物信息才能放在同一个页上，不同生产厂家的货物信息不能放在同一个页上。

分析阶段，抠每个字段

## 【面试】主键生成策略 代理主键，业务主键

代理主键，跟业务无关，自增，UUID

业务主键，跟业务有密切关系，XXXYYYZZZ，省市县，国家规范，合同号 14JK009

在实际开发中最好使用代理主键。

## 【面试】CHAR(10)

存储SETS

VARCHAR 4位长度

CHAR10位长度，其他位空格

## 数据库设计原则

数据库三范式，原则，数据库当中列，表不应该由其他的列加工而得。

数据库三范式有什么好处？表存储的内容比较少，节约表存储空间，数据的唯一出处。

总金额=数量\*单价

现今流行的数据库设计原则，反三范式。

随着电脑硬件的发展，原有硬件非常贵，环境对其影响非常大。温度，尘土。价格很便宜。

早期数据设计时强调空间，现在数据库设计强调性能，也就是用户的体验

## 【面试】生产厂家和货物是什么关系？

一对多 对，样子相同，但货号不同，对杰信来说，它们是不同货物

多对多 错

## 出货表

设计方案一（传统设计）

合同表，货物表，附件表多次左连接，SQL极其复杂，由于附件值特殊，SQL不能直接完成，还需要代码去实现。

设计方案二（冗余设计）

通过冗余，都在货物表中，只需要合同表和货物表左连接即可。

## 分析过程

将用户的需求（大白话）转化为开发人员看得懂的数据库表，表之间的关系，业务逻辑。给我们的开发人员看，开发人员依据设计来写代码。复杂，大型项目画类图，序列图辅助开发人员开发。

## 【面试】外键什么时候创建

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 开发阶段 | 测试阶段 | 试运行阶段 | 正式上线阶段 | 维护阶段 |
| 否 | 否 | 否 | 是 | 是 |



在实际开发中，架构师搭建项目框架，框架的基类都有，基础工具类，统一样式表，统一的公用的js，图片。公司提供一个CVS/SVN账号，配置环境，从中检出你负责的项目的代码。代码一般以项目结构。导入工程，在其上面开发新的业务模块。

## 业务需求

销售人员录入合同时，它为草稿状态，这时只能销售人员自己看到。只有当销售人员检查合同无误，提交，状态变成“已上报”。这时候，其他人员才可以看到。销售的领导和船务专责才能看到。利用这个状态字段来控制对当前用户是否可见。（细粒度的权限控制-数据权限）

## 重点知识回顾

货运管理核心业务流程：购销合同

1.定义

购销合同是杰信和生产厂家签订的合同。

2.业务复杂度

购销合同业务比较复杂，它是货运管理的一个流程，它包括三部分内容，合同的主信息和多个货物信息和多个附件信息。合同主信息和货物是一对多，货物和附件信息是一对多。

3.从业务中迁出技术亮点

前期我参与了业务调研，重点负责购销合同。给客户交流后，我编写了《需求说明书》，然后参数购销合同的表设计。从需求说明书结合客户给的电子表格，我抽取出数据库表，利用PD进行建模。

设计：

1. 跟客户聊天，要引导性和他们聊天。业务基础术语，了解业务流程
2. 深入内容，拿到客户手中现有某个模块表单，电子表格，填写的内容，每个内容要注意的事项。来源：手填，从其他模块来获取。抠每个字段，电子表格中每个内容。
3. 《详细说明书》功能的细化，业务逻辑说清楚，画图辅助开发，UML（类图、序列图、状态图）数据库建模细化。

目的：开发人员可以依照详细设计开始开发。

详细设计本意能达到开发人员无需和客户沟通，就可以在设计人员的指导下就可以直接开发出客户满意的代码。但实际中做不到。设计不到位。实际中开发往复比较多。

## 第三天 购销合同+货物信息+附件信息

## Springmvc日期类型转换

如果页面有日期类型，必须继承BaseController，否则转换出错，页面报400错误

## 业务需求：上报、取消

## 功能：

上报实现销售人员确定购销合同录入完毕，检查无误后，上报给领导，领导审核无误后，提交给船运部。

取消实现当船务专责未处理此相关的合同时，可以取消这个合同，重新修改合同内容，然后再次上报。如果船运专责已经开始处理报运的业务，不能取消，必须走货物增补流程。

## 设计实现：

在表中增加状态字段，新增时默认状态为0草稿，当销售专责确认无误后，点击“上报”状态变为1已上报。

当船运专责未处理报运时，销售专责点击“取消”状态变为0，草稿。

状态为0时，只能销售专责自己看到，当状态为1时，他的领导和报运专责都可以看到。

## Jsp相对路径和绝对路径

相对路径，当前文件所在路径，

在当前文件调用一个jsp页面，浏览器自动加上当前路径+jsp文件名

绝对路径，文件前，都加了一个路径 ctx+/cargo/contract/jContractList.jsp；springmvc它要经过视图解析器的封装 WEB-INF/pages + 逻辑名 + “”

## Action相对路径和绝对路径

相对路径，当前action所在路径+action名称

绝对路径，ctx+/cargo/contract 由@RequestMapping决定

## MyBatis自动生成代码的插件

## 功能：

1）生成实体domain

2）生成mapper映射文件

3）生成客户端调用代码 sqlSession.select

## 操作步骤：

1. 覆盖myeclipse插件目录
2. 两个配置文件

generatorConfig.properties 配置信息（常变化）

sqlMapGeneractor.xml 生成基础配置

## sqlMapGeneractor.xml配置

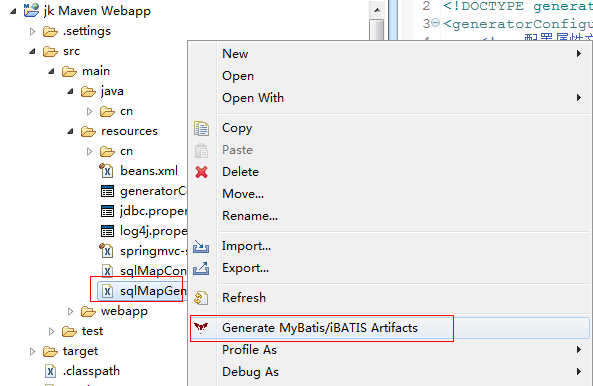
1. 必须配置绝对路径的数据库驱动jar
2. 哪些表要生成代码，注意：生成会覆盖原有文件，手工修改的内容会丢失

缺点：

1. 非数据库定义的伪列（虚拟列），自动生成不会创建
2. 表之间的关系，它无法生成

一般用于第一次生成或者局部拷贝

插件安装成功，点属性文件右键菜单会出现下面的一项



## 项目的mapper文件和mybatis生成工具的差异

1. 命名空间
2. Where所有字段，一般根据业务需要，需要几个写几个
3. 产生多余的Example调用的sql，这些对于项目来说，无用
4. 创建resultMap，主键<id>，需要手工修改
5. 普通属性基本完全拷贝，当需要类型映射比较精确时，只能手工修改；例如：数据库表字段定义的是integer，但是在自动生成时映射称为DECIMAL
6. 查询只有按id来查询，相当于get。需要补充查询多条的find，修改为get
7. 自动生成提供动态insert和一个全部字段的insert；项目中一般只用全部字段的insert
8. 自动生成提供动态update和一个全部自动的update；项目中一般只用动态update
9. 自动生成提供按id来删除；项目中一个按id删除，一个按多个id批量删除

## 业务：装率

箱数=数量 除以 装率的分母

## 导入数据常见方法

1. 在旧系统中导出文件txt,excel,xml；在新系统中导入
2. 在旧系统中创建webservice服务，新系统通过服务查询数据，利用程序写入的新系统的数据库
3. 数据库工具pl/sql导出sql，复制黏贴
4. 万能，程序

## 重点知识回顾

## 业务，主从从

购销合同、合同下货物、货物下的附件，构成复杂的结构，两级主从。一个后台处理的对象结构上，在前台页面组织结构。

## 对象结构

1. 传统对象方式，对象关联关系，在对象的实体中，合同和货物关系，会在合同中多一个Set/List集合对象 Set<ContractProduct> contractProducts
2. 它们三个都是按照单对象，在对象中并没有刻意体现它们三个的关系。对象之间的关系，没有直接体现，而用外键的字段来体现。把复杂的对象关联关系，变成一个普通字段。使代码简单化，使业务简单化，效率更高。体现出mybatis优势。

## 页面组织结构

1. 在列表后面加链接，实际开发中常采用的一种主从关联页面操作方式
2. easyUI、extJS提供了更丰富，更方便操作的强大的页面控件。GRID表格控件。

## 相对路径，绝对路径，jsp,action

大多链接都采用相对路径。直接写文件链接<img src=”../../skin/images/a.jpg”/>

绝对路径 base.jsp ctx /jk

## Mybatis自动生成代码插件

覆盖myeclipse插件目录

配置相关配置，它需要访问绝对路径的数据库驱动jar。配置要产生文件的目录。（推荐你不要设置项目相关的目录，它每次产生文件会覆盖现有文件）

##  第四天 购销合同查看，SQL语句的Power

## 附件业务实现

随时携带主表的ID

## 购销合同查看

合同下货物，货物下面附件，在购销合同中全部浏览，体现货物和附件的关系。

构建对象关系

## 【面试】PO、VO、BO有什么区别？

PO 持久化对象，跟数据库表对应

VO 视图对象，跟页面对应

BO 业务对象，跟业务逻辑加工对应对象（比较少）

PO、VO、BO人为分类，它们本质没有什么区别，它们都是POJO。

构建复杂SQL时原则

1. 从最小的结果集开始查询
2. 将它们进行连接，用左连接

## 合同对象关联SQL

select

cp.contract\_id,

cp.offeror,cp.contract\_no,cp.signing\_date,cp.input\_by,cp.check\_by,cp.inspector,cp.total\_amount,cp.import\_num,cp.crequest,cp.custom\_name,

cp.delivery\_period,cp.ship\_time,cp.trade\_terms,cp.remark,cp.print\_style,cp.old\_state,cp.state,cp.out\_state,cp.create\_by,cp.create\_dept,cp.create\_time,

cp.contract\_product\_id,

cp.product\_no,cp.product\_image,cp.product\_desc,cp.cnumber,cp.out\_number,cp.loading\_rate,cp.box\_num,cp.packing\_unit,cp.price,cp.amount,cp.finished,cp.exts,cp.order\_no,

cp.factory\_id,cp.full\_name,cp.factory\_name,cp.contacts,cp.phone,

ep.ext\_cproduct\_id,ep.contract\_product\_id,

ep.ctype,

ep.product\_no as ext\_product\_no,ep.product\_image as ext\_product\_image,ep.product\_desc as ext\_product\_desc,

ep.cnumber as ext\_cnumber,ep.packing\_unit as ext\_packing\_unit,ep.price as ext\_price,

ep.amount as ext\_amount,ep.product\_request as ext\_product\_request,ep.order\_no as ext\_order\_no,

ep.factory\_id as ext\_factory\_id,ep.full\_name as ext\_full\_name,ep.factory\_name as ext\_factory\_name,ep.contacts as ext\_contacts,ep.phone as ext\_phone

from

(

select

c.contract\_id,

c.offeror,c.contract\_no,c.signing\_date,c.input\_by,c.check\_by,c.inspector,c.total\_amount,c.import\_num,c.crequest,c.custom\_name,

c.delivery\_period,c.ship\_time,c.trade\_terms,c.remark,c.print\_style,c.old\_state,c.state,c.out\_state,c.create\_by,c.create\_dept,c.create\_time,

cp.contract\_product\_id,

cp.product\_no,cp.product\_image,cp.product\_desc,cp.cnumber,cp.out\_number,cp.loading\_rate,cp.box\_num,cp.packing\_unit,cp.price,cp.amount,cp.finished,cp.exts,cp.order\_no,

cp.factory\_id,cp.full\_name,cp.factory\_name,cp.contacts,cp.phone

from

(

select

contract\_id,

offeror,contract\_no,signing\_date,input\_by,check\_by,inspector,total\_amount,import\_num,crequest,custom\_name,

delivery\_period,ship\_time,trade\_terms,remark,print\_style,old\_state,state,out\_state,create\_by,create\_dept,create\_time

from contract\_c

) c

left join

(

select

cp.contract\_product\_id,cp.contract\_id,

cp.product\_no,cp.product\_image,cp.product\_desc,cp.cnumber,cp.out\_number,cp.loading\_rate,cp.box\_num,cp.packing\_unit,cp.price,cp.amount,cp.finished,cp.exts,cp.order\_no,

f.factory\_id,f.full\_name,f.factory\_name,f.contacts,f.phone

from

(

select

contract\_product\_id,contract\_id,factory\_id,

product\_no,product\_image,product\_desc,cnumber,out\_number,loading\_rate,box\_num,packing\_unit,price,amount,finished,exts,order\_no

from contract\_product\_c

) cp

left join

(select factory\_id,full\_name,factory\_name,contacts,phone from factory\_c) f

on cp.factory\_id=f.factory\_id

) cp

on c.contract\_id=cp.contract\_id

) cp

left join

(

select

ep.ext\_cproduct\_id,ep.contract\_product\_id,

ep.ctype,ep.product\_no,ep.product\_image,ep.product\_desc,ep.cnumber,ep.packing\_unit,ep.price,ep.amount,ep.product\_request,ep.order\_no,

f.factory\_id,f.full\_name,f.factory\_name,f.contacts,f.phone

from

(

select

ext\_cproduct\_id,contract\_product\_id,factory\_id,

ctype,product\_no,product\_image,product\_desc,cnumber,packing\_unit,price,amount,product\_request,order\_no

from ext\_cproduct\_c

) ep

left join

( select factory\_id,full\_name,factory\_name,contacts,phone from factory\_c ) f

on ep.factory\_id=f.factory\_id

) ep

on cp.contract\_product\_id=ep.contract\_product\_id

注意MyBatis查询出的结果集中不能有同名字段，必须另起别名。

## 关联映射，一对多，多对一

<resultMap type="cn.itcast.jk.domain.Contract" id="contractRM">

<id property="id" column="CONTRACT\_ID"/>

<result property="cpnum" column="CPNUM"/>

<result property="extnum" column="EXTNUM"/>

<result property="offeror" column="OFFEROR"/>

<result property="contractNo" column="CONTRACT\_NO"/>

<result property="signingDate" column="SIGNING\_DATE"/>

<result property="inputBy" column="INPUT\_BY"/>

<result property="checkBy" column="CHECK\_BY"/>

<result property="inspector" column="INSPECTOR"/>

<result property="totalAmount" column="TOTAL\_AMOUNT"/>

<result property="importNum" column="IMPORT\_NUM"/>

<result property="crequest" column="CREQUEST"/>

<result property="customName" column="CUSTOM\_NAME"/>

<result property="deliveryPeriod" column="DELIVERY\_PERIOD"/>

<result property="shipTime" column="SHIP\_TIME"/>

<result property="tradeTerms" column="TRADE\_TERMS"/>

<result property="remark" column="REMARK"/>

<result property="printStyle" column="PRINT\_STYLE"/>

<result property="oldState" column="OLD\_STATE"/>

<result property="state" column="STATE"/>

<result property="outState" column="OUT\_STATE"/>

<result property="createBy" column="CREATE\_BY"/>

<result property="createDept" column="CREATE\_DEPT"/>

<result property="createTime" column="CREATE\_TIME"/>

</resultMap>

<!-- 对象的关联关系，合同、货物、附件、生产厂家 -->

<resultMap type="cn.itcast.jk.vo.ContractVO" id="contractVORM" extends="contractRM">

<!-- 货物信息（一对多） -->

<collection property="contractProducts" ofType="cn.itcast.jk.vo.ContractProductVO">

<id column="CONTRACT\_PRODUCT\_ID" property="id" jdbcType="VARCHAR" />

<result column="FACTORY\_ID" property="factoryId" jdbcType="VARCHAR" />

<result column="FACTORY\_NAME" property="factoryName" jdbcType="VARCHAR" />

<result column="PRODUCT\_NO" property="productNo" jdbcType="VARCHAR" />

<result column="PRODUCT\_IMAGE" property="productImage" jdbcType="VARCHAR" />

<result column="PRODUCT\_DESC" property="productDesc" jdbcType="VARCHAR" />

<result column="CNUMBER" property="cnumber" jdbcType="INTEGER" />

<result column="OUT\_NUMBER" property="outNumber" jdbcType="INTEGER" />

<result column="LOADING\_RATE" property="loadingRate" jdbcType="VARCHAR" />

<result column="BOX\_NUM" property="boxNum" jdbcType="INTEGER" />

<result column="PACKING\_UNIT" property="packingUnit" jdbcType="VARCHAR" />

<result column="PRICE" property="price" jdbcType="DOUBLE" />

<result column="AMOUNT" property="amount" jdbcType="DOUBLE" />

<result column="FINISHED" property="finished" jdbcType="INTEGER" />

<result column="EXTS" property="exts" jdbcType="VARCHAR" />

<result column="ORDER\_NO" property="orderNo" jdbcType="INTEGER" />

<!-- 生产厂家（多对一）mybatis优化，按需要所取 -->

<association property="factory" javaType="cn.itcast.jk.domain.Factory">

<id property="id" column="FACTORY\_ID"/>

<result property="fullName" column="FULL\_NAME"/>

<result property="factoryName" column="FACTORY\_NAME"/>

<result property="contacts" column="CONTACTS"/>

<result property="phone" column="PHONE"/>

</association>

<!-- 附件信息（一对多） -->

<collection property="extCproducts" ofType="cn.itcast.jk.vo.ExtCproductVO">

<id column="EXT\_CPRODUCT\_ID" property="id" jdbcType="VARCHAR" />

<result column="EXT\_FACTORY\_ID" property="factoryId" jdbcType="VARCHAR" />

<result column="EXT\_FACTORY\_NAME" property="factoryName" jdbcType="VARCHAR" />

<result column="EXT\_CTYPE" property="ctype" jdbcType="INTEGER" />

<result column="EXT\_PRODUCT\_NO" property="productNo" jdbcType="VARCHAR" />

<result column="EXT\_PRODUCT\_IMAGE" property="productImage" jdbcType="VARCHAR" />

<result column="EXT\_PRODUCT\_DESC" property="productDesc" jdbcType="VARCHAR" />

<result column="EXT\_CNUMBER" property="cnumber" jdbcType="INTEGER" />

<result column="EXT\_PACKING\_UNIT" property="packingUnit" jdbcType="VARCHAR" />

<result column="EXT\_PRICE" property="price" jdbcType="DOUBLE" />

<result column="EXT\_AMOUNT" property="amount" jdbcType="DOUBLE" />

<result column="EXT\_PRODUCT\_REQUEST" property="productRequest" jdbcType="VARCHAR" />

<result column="EXT\_ORDER\_NO" property="orderNo" jdbcType="INTEGER" />

<!-- 生产厂家（多对一）mybatis优化，按需要所取 -->

<association property="factory" javaType="cn.itcast.jk.domain.Factory">

<id property="id" column="EXT\_FACTORY\_ID"/>

<result property="fullName" column="EXT\_FULL\_NAME"/>

<result property="factoryName" column="EXT\_FACTORY\_NAME"/>

<result property="contacts" column="EXT\_CONTACTS"/>

<result property="phone" column="EXT\_PHONE"/>

</association>

</collection>

</collection>

</resultMap>

## 业务：货物数和附件数

select

(select count(\*) from contract\_product\_c where contract\_id=t.contract\_id) as cpnum,

(select count(\*) from ext\_cproduct\_c where contract\_product\_id in (select contract\_product\_id from contract\_product\_c where contract\_id=t.contract\_id)) as extnum,

t.\* from CONTRACT\_C t

where 1=1

## 在使用hibernate和mybatis一些区别？

应用hibernate，对象映射，对象的关联关系。

应用mybatis，SQL，利用SQL直接查询所需要的内容，提高了获取数据的效率。SQL可以按照用户的业务来挑取所要的字段。

Hibernate当中SQL能否优化，一般情况无法优化SQL，因为hibernat使用HQL，它自动生成SQL语句。Mybatis可以实现手工SQL优化。

Hibernate开发重点，面向对象，来思考，来开发；全自动ORM

Mybatis开发重点，面向SQL，面向过程思考，面向对象的开发；半自动ORM

## 级联删除

在hibernate中只要配置映射文件中的对象关系，级联删除。当删除主对象时，根本不管是否有子对象，直接删除主对象就实现。Hibernate底层自动删除相关的子对象。

合同、货物、附件两级主从。Hibernate中只关系上下级对象。

在mybatis中没有级联删除的概念，只能手工去删除。引发骨牌效应。

## 业务，购销合同查询

利用状态，可以控制登陆人看到不同的信息。数据权限控制

## \*细粒度权限控制

主菜单，左侧菜单，按钮，URL，方法，业界都称为粗粒度的权限控制。日常的小系统，基于用户，角色，权限架构足够用。

在实际大型项目中，光有上面的用户角色，URL这样的控制粒度不够细致，还有数据访问权限。

1. 对数据的访问列是可以自己定制的。

例如：访问一个人员工资信息表，这个表中含有人员的工资。

1. 对数据的访问行可以受权限控制。

例如：工资表中有销售部人员的工资信息，还有船运部人员的工资信息。当销售部领导登录时，他只能看到销售人员的工资信息，他不能看到船运部人员的工资信息。

1. 直接浏览数据库表，工资列不能直接看到其值。对这个数据库列进行加密。

上面统称为数据访问权限，这个权限加上角色用户这样这个系统才比较安全。也称为细粒度的权限控制。

## 列访问控制：

单独建立表，权限分配某个角色能访问哪些列；

## 行访问控制：

1. 只能看自己创建的记录，不能看到别人创建的记录

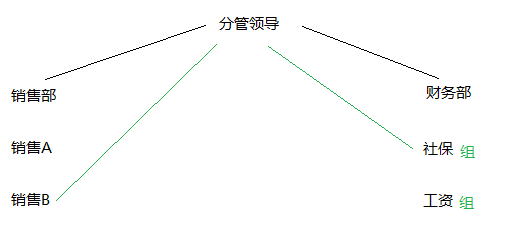
Where create\_by = 当前登陆人ID

1. 部门领导看到，他能看到本部门人员创建的记录，不能看到其他部门人员创建的记录

Where create\_dept = 当前登陆人部门ID and user\_level <4

1. 跨部门访问权限，分管领导跨部门，跨人员

再加一个配置表，分管领导可以管理的人和部门；



## 练习：

1.合同、货物、附件、生产厂家对象关联SQL

2.购销合同列表实现货物数和附件数，合同总金额 SQL语句

合同总金额=所有当前合同下的货物的数量\*单价总和+所有附件的数量\*单价总和

3.对象关联的映射

## 重点知识回顾

## 购销合同查看，采用类似hibernate方式，都已对象关联方式。

1.PO为了利用mybatis性能，在创建时，没有采用对象关联，而是将对象关联字段，也就是外键，利用这个普通属性，来记录值，表数据间关联关系存在，但对象关联关系不存在。代码也就变得简单。在货物新增时，只要从主对象中携带过来，主表ID即可。

2.VO为了方便对象关联时取数据。在列表循环货物信息时，要去查询当前货物下的附件时，如果采用上面的方式，只能再次查询。但是我们已对象关联方式，可以直接获取到当前货物下的附件集合。一次SQL所有的附件都查询出来。

PO/VO都是POJO对象。根据业务的当前环境。

## 货物数、附件数

方法1：代码可以通过合同号，分别去查询货物数、附件数；它需要发出多条的SQL再次查询。

方法2：利用SQL语句直接支持，它可以把一个结果集作为一个字段值，这样的SQL返回结果集必须一个列。同时在这个子查询中它的条件可以拼接动态的条件。contract\_id=t.contract\_id t是总表的别称。这样引入一个动态的条件。以此类推实现了附件数。

方法1和方法2效能够不高。方法1，代码加工，要重新发出SQL子查询。效率低。方法2，SQL嵌套很多子查询，虽然它是数据库级别，也就是允许在强大的服务器上。

方法3：冗余+分次计算【亮点】

在表中建立货物数和附件数字段，在货物新增时，当前合同，货物数+1。删除货物，货物数-1；在附件新增时，当前合同，附件数+1，删除附件时，附件数-1，删除货物时，要将其下面的附件总数从合同附件数中减去。代码考虑的比较复杂些，但它因为分散各处，计算效率很高，查询时，无需动态计算，效率奇高。

## 权限：【亮点】

角色用户权限，大多数系统都使用此种方式。

细粒度权限，它控制到数据访问权限。权限架构才能解决实际项目的应用。（财务）

##  第五天 POI的神奇世界

## 业务，出货表

功能：

对购销合同进行月统计，按船期进行统计。

合同、货物、生产厂家、附件传统设计，关联4张表；

通过冗余设计，只需要从合同、货物表中获取数据；取数据更加方便，代码更加简单。

分析表格，循环货物信息能方便实现打印数据。将这些数据打印到excel中。

## JAVA语言操作excel的api主流有两种方式：

1. POI Apache它是用来操作Office所有的软件excel/word/ppt/..。而且支持所有版本。
2. JXL 它是用来操作excel 2003以下版本，2007以上版本不支持。

早期微软Office系列，当时使用OLE2文档结构；微软在开发Office2207版本时，做了一个重大的改革。重写了Office，使用OOXML文档结构。现在excel文件实际上是一个xml格式文件。

POI支持OLE2格式文件，还支持OOXML，而且在OOXML格式文件时做了很大的优化。

JXL只支持OLE2格式文件。

POI低版本有个致命弱点，当数据量超大（海量数据），性能极具降低。JXL它解决数量大时性能的问题。POI在高版本时迎头追上，加了一个补丁。OOXML支持，解决大数量时的性能问题。

<http://poi.apache.org/index.html> POI 全方位的操作

<http://www.andykhan.com/jexcelapi/index.html> JXL 导入导出

java.lang.IllegalArgumentException: Invalid column index (256). Allowable column range for BIFF8 is (0..255) or ('A'..'IV')

java.lang.IllegalArgumentException: Invalid row number (65536) outside allowable range (0..65535)

HSSF对象支持excel 2003，excel 2003它支持行65536，支持列256

XSSF对象支持excel2007版本以上，行1048576，支持列65536。Sheet

SXSSF对象，支持excel 2007版本以上，构造函数。不支持模板方式。只能作为数据导出。

SXSSF如何实现？

HSSF/XSSF 它们在构建workbook中所有对象，全部放在内存中，数据全放在内存中，wb.write时才从内存向磁盘文件中写。写完才释放内存。数据量大，堆溢出。

SXSSF构造方法Workbook wb = new SXSSFWorkbook(100);

每处理完100个对象，将它写临时文件，这部分数据所占的内存，就释放。

File tmplFile = File.createTempFile("poi-sxssf-template", ".xlsx");

try

{

FileOutputStream os = new FileOutputStream(tmplFile);

try

{

\_wb.write(os);

}

finally

{

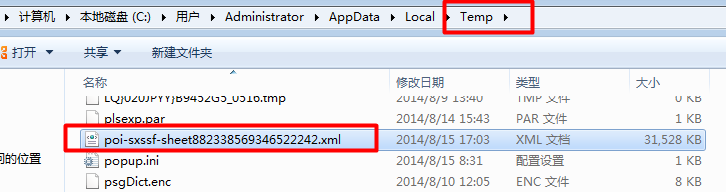
os.close();

}

//Substitute the template entries with the generated sheet data files

injectData(tmplFile, stream);

}



在生成的xml临时文件中，只存放数据

CSV 实际文本文件，存放数据，数据之间，可以用逗号隔开，也可以用TAB隔开。通用导入导出文件。

当数据量大，跟当前电脑环境，堆溢出。

优化Myeclispse

优化jvm启动参数

## POI入门

Excel文件

1. 创建excel文件，称作工作簿workbook
2. Excel文件在创建时，会默认创建3个工作表sheet
3. 定位行
4. 定位列，获得单元格
5. 填写内容
6. 保存，关闭

POI开发步骤：

public void HSSF() throws IOException{

//1.创建一个工作簿excel文件

Workbook wb = new HSSFWorkbook(); //HSSF操作excel 2003以下版本

//2.创建一个工作表sheet

Sheet sheet = wb.createSheet();

//3.创建一个行对象Row

Row nRow = sheet.createRow(4); //第五行,坐标从0开始

//4.创建一个单元格对象，指定列

Cell nCell = nRow.createCell(3); //第四列

//5.给单元格设置内容

nCell.setCellValue("传智播客万年长！");

//6.保存

OutputStream os = new FileOutputStream(new File("c:\\testPOI.xls"));

wb.write(os);

//7.关闭

os.close();

}

上面代码存在的问题：

1. POI创建的这些对象统统在内存中
2. 行对象，列对象，样式对象，字体对象重复创建

## 出货表开发步骤

1）获取数据

select

c.custom\_name,c.contract\_no,cp.product\_no,cp.cnumber,cp.factory\_name,cp.exts,c.delivery\_period,c.ship\_time,c.trade\_terms

from

(select contract\_id,product\_no,cnumber||packing\_unit as cnumber,factory\_name,exts from contract\_product\_c) cp

left join

(select contract\_id,custom\_name,contract\_no,delivery\_period,ship\_time,trade\_terms from contract\_c) c

on cp.contract\_id=c.contract\_id

where to\_char(ship\_time,'yyyy-MM')='2011-12'

2）创建excel文件

3）修饰样式

4）将数据写入excel

5）下载文件

## 常规poi打印存在问题：

1. 列的宽度不精确，272非常相近
2. 设置纸张方向
3. 设置页眉页脚
4. 标题栏
5. 内容样式稍有不同，就需要创建不同的样式，写很多样式应用的语句
6. 表格头，写死表头，静态的文字方便维护

## POI操作EXCEL，复杂报表终极解决方案~模板开发

1. 代码量急剧降低
2. POI日常API，将利用模板直接来设置，可视化修改。
3. 业务功能非常方便维护。



//模板打印

@RequestMapping("/cargo/outproduct/printTemplate.action")

public void printTemplate(String inputDate, HttpServletRequest request, HttpServletResponse response) throws IOException{

List<OutProductVO> dataList = outProductService.find(inputDate);

//打开模板文件

String path = request.getSession().getServletContext().getRealPath("/"); //jdk1.8 bug 在linux下不带后面你写的路径

String tempFile = path + "/make/xlsprint/tOUTPRODUCT.xls";

Workbook wb = new HSSFWorkbook(new FileInputStream(new File(tempFile)));

//写入业务内容

Sheet sheet = wb.getSheetAt(0); //获得工作表sheet

Row nRow = null;

Cell nCell = null;

int rowNo = 2;

int colNo = 1;

//获取样式

nRow = sheet.getRow(2); //获得行对象

nCell = nRow.getCell(1); //获取单元格对象

CellStyle customNameStyle = nCell.getCellStyle(); //获取到样式

nCell = nRow.getCell(2);

CellStyle contractNoStyle = nCell.getCellStyle();

nCell = nRow.getCell(3);

CellStyle productNoStyle = nCell.getCellStyle();

nCell = nRow.getCell(4);

CellStyle cnumberStyle = nCell.getCellStyle();

nCell = nRow.getCell(5);

CellStyle factoryStyle = nCell.getCellStyle();

nCell = nRow.getCell(6);

CellStyle extStyle = nCell.getCellStyle();

nCell = nRow.getCell(7);

CellStyle dateStyle = nCell.getCellStyle();

nCell = nRow.getCell(9);

CellStyle tradeStyle = nCell.getCellStyle();

//大标题

nRow = sheet.getRow(0);

nCell = nRow.getCell(1);

nCell.setCellValue(inputDate.replaceFirst("-0", "-").replaceFirst("-", "年") + "月份出货表"); //yyyy-MM

//处理数据

for(int j=0;j<dataList.size();j++){

colNo = 1; //列号初始化

OutProductVO op = dataList.get(j); //获取出货表对象

nRow = sheet.createRow(rowNo++);

nRow.setHeightInPoints(24);

nCell = nRow.createCell(colNo++);

nCell.setCellValue(op.getCustomName());

nCell.setCellStyle(customNameStyle); //设置样式

nCell = nRow.createCell(colNo++);

nCell.setCellValue(op.getContractNo());

nCell.setCellStyle(contractNoStyle);

nCell = nRow.createCell(colNo++);

nCell.setCellValue(op.getProductNo());

nCell.setCellStyle(productNoStyle);

nCell = nRow.createCell(colNo++);

nCell.setCellValue(op.getCnumber());

nCell.setCellStyle(cnumberStyle);

nCell = nRow.createCell(colNo++);

nCell.setCellValue(op.getFactoryName());

nCell.setCellStyle(factoryStyle);

nCell = nRow.createCell(colNo++);

nCell.setCellValue(op.getExts());

nCell.setCellStyle(extStyle);

nCell = nRow.createCell(colNo++);

nCell.setCellValue(op.getDeliveryPeriod());

nCell.setCellStyle(dateStyle);

nCell = nRow.createCell(colNo++);

nCell.setCellValue(op.getShipTime());

nCell.setCellStyle(dateStyle);

nCell = nRow.createCell(colNo++);

nCell.setCellValue(op.getTradeTerms());

nCell.setCellStyle(tradeStyle);

}

//下载

DownloadUtil du = new DownloadUtil();

ByteArrayOutputStream os = new ByteArrayOutputStream();

wb.write(os);

du.download(os, response, "出货表.xls");

}

## 重点知识回顾

## 全面了解POI

JXL 它导入导出速度很快，业界一般用来数据的导入导出。

## 导入导出：

1. 数据备份 txt,csv,excel（客户比较喜欢）,xml
2. 两个业务系统之间共享数据

JXL局限：它只能操作excel2003版本以下

POI 操作Office 支持excel,word,ppt等。业界习惯使用它来操作excel。

操作word，使用.net，它直接可以访问word对象；使用word宏，宏脚本语言，vba。网上政府办公，申请某个营业执照，下载一个word文件，填写。

业界习惯使用POI来操作excel。

OLE2 2003以下版本 HSSF 行：65536；列：255

OOXML 2007以上版本 XSSF/SXSSF 行：1048576 列：65536

HSSF它操作excel 2003版本，它数据量有限，它支持到65536条。

XSSF它操作excel 2007版本，它不能读取excel2003版本，它支持1048576，单个sheet。

SXSSF 它操作excel 2007版本，只用于大数据量的导出。它实现指定创建一定对象数量后，将这些不再使用的对象，写入临时文件。这样释放出这些对象所占用的内存，内存占用不高，才可以顺利完成海量数据的导出。

## 出货表使用模板

实际工作中最简单的一个表格

1. 无需大量的设置样式，可视化
2. 静态文字，图片，线，可以提前在模板中实现
3. Excel打印的一些设置，纸张方向，列宽，基础行高，标题，页眉页脚
4. 代码功能的可维护性非常方便
5. 通用性不强，根据具体的业务来实现。

## 第六天 复杂制式表单打印~购销合同

## 购销合同报表分析，读程

1. 购销合同是否可以利用模板来开发。POI区域拷贝，网上有类似行拷贝。不支持特殊元素的拷贝。货物信息在变化，有可能两个，有可能一个，一个时货物下面的信息位置都被提上来。

如果非要用模板实现，改造，把每页作为一个sheet的内容。POIsheet拷贝。一个货物的做一个模板，两个货物的做一个模板。或者利用程序判断，一个或者两个货物。

1. 图片如何插入到excel

HSSFPatriarch patriarch = sheet.createDrawingPatriarch(); //add picture

poiUtil.setPicture(wb, patriarch, path+"make/xlsprint/logo.jpg", 开始行, 开始列, 结束行, 结束列);

1. 如何画线

poiUtil.setLine(wb, patriarch, 开始行, 开始列, 结束行, 结束列); //draw line

1. 如何设置单元格换行

curStyle.setWrapText(true); // 换行

1. 如何设置单元格的前缀

public short rmb4Format(HSSFWorkbook wb) {

HSSFDataFormat format = wb.createDataFormat();

return format.getFormat("\"￥\"#,###,##0.00"); // 设置格式

}

1. 如何建立公式

nCell.setCellType(HSSFCell.CELL\_TYPE\_FORMULA);

nCell.setCellFormula("F"+String.valueOf(curRow)+"\*H"+String.valueOf(curRow));

1. 验货员不能由于输入审单人，它的位置不能发生变化

utilFuns.fixSpaceStr(contract.getCheckBy(),26)

1. 行自适应内容的高度

float height = poiUtil.getCellAutoHeight(printMap.get("Request"), 12f); //自动高度

1. 分页符

if(p>0){

sheet.setRowBreak(curRow++); //在第startRow行设置分页符

}

1. 数据和业务分离。单独组织数据源，打印时只关系它从哪一页获取数据？

Map<String,String> pageMap = null;

List<Map> pageList = new ArrayList(); //打印页

Map pageMap 存储每页数据，打印时就无需关心数据从哪个对象中来，直接从它的pageMap拿，只需要按业务的key来获取。获取数据变的简单。打印代码和获取数据就不用交叉，代码结构更清晰，维护人更方便读程。

pageList存放所有页，每条记录就是一页，一页数据都放在pageMap中。

1. 日期格式转换，转为中文格式

UtilFuns.formatDateTimeCN(UtilFuns.dateTimeFormat(contract.getSigningDate())));

1. 合并单元格，如果有边线

操作的样式只设置了第一个单元格。手工设置其它的单元格，补上线。

## 重点知识回顾

复杂报表的打印，

开发步骤：

1. 读程
   1. 通读人家的程序代码
   2. 浏览人家写的注释
   3. 自己一行一行的写注释
   4. 重点放在特殊功能，插入图片，插入线，if,for
   5. 运行起来，跟踪断点
   6. 改造
2. 打印

复杂报表打印开发步骤：

* 1. POI先实现业务，可以先不修饰
  2. 边写代码边写样式，立刻看到打印结果（熟练）

## 第七天 异构数据库支持+出口报运核心业务

## 同时支持多种主流数据，oracle/mysql/sqlserver

## 异构系统

Java实际开发中无法完全支持异构系统，同时支持windows,linux。

西安市政府网站，在windows平台上开发，在linux环境运行。实现页面的静态化。按一定规则new/1009.shtml。（html静态页面缺点，它无法实现文件包含，业界推出新技术shtml，它扩展了html，它也是静态页面，它实现了文件包含。要支持这个tomcat的修改ssi配置，默认这个支持配置关闭）第一次调用时，这个静态文件不存在，系统就报404，系统用filter拦截404错误，去查询数据库，动态生成new/1009.shtml文件。

发现一个问题，在window平台没问题，到linux，它无法获取域名，它可以获取IP地址。

## 异构数据库，都有哪些要注意的呢？

|  |  |  |  |
| --- | --- | --- | --- |
|  | oracle | mysql | SqlServer |
| 创建表语法不同，使用PD针对不同的数据库，创建不同的PDM | | | |
| 字符类型 | VARCHAR2 | VARCHAR | VARCHAR |
| 日期 | TIMESTAMP | DATATIME | DATATIME |
| 浮点数 | NUMBER(x,y) | FLOAT/DOUBLE | FLOAT/DOUBLE |
| 整形 | INT | TINGINT | BIT/TINGINT |
| 系统的数据库连接属性文件jdbc.properties中配置多个驱动连接，部署时指定用户的数据库。开发中一般以一个数据库为主，其他的交给测试组，测试组部署多个数据库。 | | | |
| 在代码中为了适应不同的数据库，要写不同的sql语句（在编写sql时，尽量不要用数据库sql的特性） | | | |
| 在mapper映射文件中 | 写多个SQL，使用谁，给其它的加注释，将其他的数据库，在项目中创建db目录，将不同的sql放在不同的文件中。 | | |
| 字符串连接符 | || | concat | + |
| 日期 | to\_date(),to\_char() | 可以不转类型，可以讲日期类型作为字符串来处理cast() | covert()  cast() |
| 分页 | rownum | limit | 2000/2005/2008  Top |

## 核心业务：货运管理第二个流程，出口报运

## 定义：

做国际物流，货物要运往国外之前，向海关申报。向海关申报符合国家标准的货物，和货物的体积和重量信息。除了在合同中货物的基本信息外，在报运时新增7个字段，长，宽，高，毛重，净重，出口单价，含税。

分次报运

一个可以可以分多次来报运，体现在运输的货物数上。

例如：10000货物，第一次报运运输6000，第二次报运运输4000.

可以多次报运，这种情况很少。日常大多是一次报运。

## 分析设计

一个出口报运单来自多个购销合同（一对多）；给货物信息新增的7个字段的内容。

创建一个出口报运单表

## 冗余设计

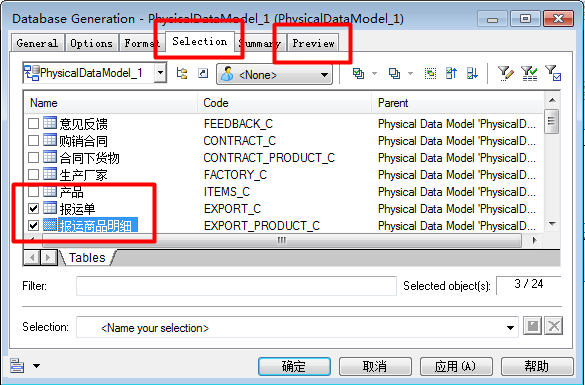
将货物信息冗余到报运单下的货物信息中。一个实现读取数据快速，减小业务的复杂度，业务逻辑简单。

## 打断设计实现跳跃查询

避免关联层级过多，数据量大了后，访问奇慢。打断设计。（表设计不成为文规定，表之间的关联关系不超过4层）在设计时使用一个虚线箭头来表示表之间是存在关联关系，在设计时，不是往常的外键关系。利用一个字段来存储主表和子表之间的关系。X,Y

Where contract\_id in (‘x’,’y’)。经过打断设计，表之间的关系不存在，跳跃查询。无需查询合同表，直接查询合同下的货物表。（新增）

经过上面的设计，后续业务变的更加简单。





创建映射文件

创建PO对象

创建dao层

创建Service层

创建Controller

出口报运新增，新增时要选择多个关联的货物。报运单自身的信息在修改中补录。

## 批量修改，批量提交，批量保存

## 动态表格技术dhtml

可以用js动态创建表格，行，单元格，设置单元格内容 DHTML

创建单元格写入内容innerText/innerHTML，利用innerHTML插入文本框，在这个文本框中显示数据

## 如果页面多个名称，springmvc如何封装，

如果是字符类型，拼接成一个字符串，之间用逗号隔开；

如何是日期类型，整形，浮点型，只保留第一个值，其他值丢掉。

页面有两个boxNum=10 boxNum=20，Integer boxNum，第二值丢失 boxNum=10；用数组声明方式；request.getParameterValues()；--没有类型概念，它都是字符串，手工转类型。

## 批量MRecord控件，自定义控件，esayUI/extJS GRID

拼接js串

## Body onload事件和jQuery ready()事件

Onload有bug，有时执行动态js插入数据到页面时，有时可以有时报错。很不稳定。

所以使用ready事件替代。

Mrecord控件使用方法：

1）引入两个js，jQuery、tabledo.js

<script type="text/javascript" src="${ctx}/components/jquery-ui/jquery-1.2.6.js"></script>

<script type="text/javascript" src="${ctx}/js/tabledo.js"></script>

2）反复调用addRecord方法

3）后台准备数据，拼接addRecord串

4）利用jQuery 的ready事件，调用js串

## 业务，劳保报表

领用周期不同

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 鞋 | 帽子 | 手套 | 毛巾 | 军大衣 | 洗衣粉 | 饭盒 | … |
| 班组1 | 10 | 10 |  |  | 5 |  |  |  |
| 班组2 | 20 | 20 |  |  | 3 |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |

6000个单元格

哪个单元格的内容进行修改，代码只修改修改的框。给每个文本框增加一个隐藏域。12000个信息要提交。本来目的是优化，结果速度更慢。

最终每行添加一个隐藏域，当这行的数据进行了修改，在文本框的失去焦点事件onBlur，动态设置这个隐藏域为1。在修改代码时根据这个值判断，只有1的时候，数据才进行修改。不是1跳过。

批量新增，增加一个新增按钮，点一次调研一次addRecord方法，它的属性值，都为空串。

批量删除，表格dhtml提供删除行操作，remove，它把行从table中删除。数据库并没有删除，删除多个框时，记录下它们的id值，拼成一个串，用逗号隔开。后台代码判断这个隐藏域提交值不为空，删除这几个数据。

## 知识回顾

## 打断设计：

1.数据库设计先按照三范式，之后对一些需要优化的地方，使用反三范式设计（冗余）（以空间换设计）

2.实际业务非常复杂，关联层级过多，随着系统上线运行，用户的数据随着使用的时间，线性增长。不成文规定：关联层级在设计不超过4层。打断设计，在主表中加一个字段，字段来存储两个表之间的关系。多个的时候，之间用逗号隔开。一般情况下都是一次报运，就是一个合同一次报运，个别情况下多个合同一次报运，极端情况下3个合同一次报运。

由于打断设计，实现“跳跃查询”。报运需要合同下的货物信息时，无需通过多个合同对象，就可以直接查询货物的信息，通过这个打断设计的字段，利用SQL的in子查询。

## 核心业务：货运管理，出口报运

业务，杰信获得订单后，安排多个生产厂家生产货物，与此同时杰信准备报关的材料，除了海关要求的一些制式文档和杰信系统打印的《出口商品报运单》，交给海关进行审核。

## 页面控制

## 报运单新增，暗度陈仓。

用户选择报运关联的多个合同，代码直接进行数据的保存。保存相关合同的id集合，合同号集合，保存冗余的货物信息，保存冗余的附件信息。

## 批量修改

Mrecord自定义控件，来信息补录。批量，在页面上利用DHTML动态表格技术，实现表格元素的动态增加，动态创建行，动态创建单元格，利用innerHTML动态向单元格插入文本框，带数据的。在后台准备数据，拼接成js串，将它写入一个变量中，在页面的jQuery的ready事件中，调用这个变量，形成多个js串，依次执行。动态添加到表格中。由于js运行很快，用户感觉不到添加的过程，以为直接展示。这样为用户提供一个非常方便的操作。可以批量来修改货物信息。每行利用一个隐藏域，来存放是否修改标识，这样在记录行数过多时，修改的效应也非常高。

Mrecord控件它可以实现动态新增，还可以实现批量删除，自动排序。

## 第八天 装箱、委托、发票、财务业务+合同归档

货运管理包括：购销合同、出口报运、装箱、委托、发票、财务统计。

## 装箱

海关批准杰信的申请，像船东申请集装箱，杰信找货代公司请他帮着订箱。杰信找拖车公司，订拖车，拖车公司到码头拉空箱。拉着空箱到生产厂家的仓库。生产厂家在杰信的验货员的监督下，先将货物装到纸箱子中，然后在将纸箱子装入集装箱。拖车拉着满箱到海关指定的码头指定的地方卸货。这几个箱子风吹雨淋。海关进行抽检。直到船来了。将集装箱放到船上。开船从装运港出发，到目的港。

装箱时，产生发票号。同时产生发票日期。

一个装箱单来自多个报运单（一对多）

## 发票，委托

在货物装船的同时，杰信就开具发票，这个发票只是一个通知单，它的内容就是告诉客户货物已经发出，请支付剩余款项。同时快递客户一个提单。客户收到发票通知后，支付剩余款项，同时当货物到达目的港后，客户拿着提单去提货。客户联系货代公司，联系拖车公司，拖车拉着满箱到客户指定的仓库，卸货。客户验货，整个货运流程完毕，杰信和客户的合同完成。

## 财务

财务模块不属于货运的业务流程。它是杰信自己内部进行财务核算的模块。统计这单合同有无赚取利润。形成统计报表给领导展示。

## 一对一特殊设计

一个委托来自一个发票，一个发票来自一个委托，一个财务来自一个委托。（一对一）

一对一的所有表内容可以合成一张表，结构不清晰，业务不清晰。

发票，委托，财务表的主键就是装箱单的ID。

上面三张表，它们的主键既外键。主外键一个值。

举例，假定拿到ID，不论是装箱、委托、发票、财务，都可以查询它们其中任何一个数据。

实现跳跃查询。

财务需要货物信息和附件信息，通过打断设计，可以直接跳过发票、委托，直接找到装箱，通过装箱的打断设计字段，直接去搜索报运下的货物和附件信息。因为之前进行了冗余设计，财务所需的货物信息和附件信息，在报运中都有。查询效率提高数倍。

## 出口报运的状态

0-草稿 1-已上报 2-装箱 3-委托 4-发票 5-财务

体现流程，客户随时想了解订单进展情况，流程的跟踪。在报运的列表页面随时了解某个合同进行到哪里。

## 数据库优化

## 数据库表设计

* + 由原来重视节省空间，变为重视性能。利用冗余，实现性能数倍提高。
  + 表的优化，监测数据执行时间。分表1）分字段（纵向分表）精简完核心查询它只需要近30个字段。原来需要1分多的查询，现在只1秒多展现页面。2）数据（横向分表）一个表数据量过大。业务中有些数据不常用，将这些不常用的数据放到另外一张表中，称这个表为历史表。专门做一个历史查询的模块，提供用户还可以查询。
  + 表之间的关联关系不要太复杂。打断设计

## SQL优化，在做复杂关联关系，SQL优化极其重要。

Select \* from contract\_c,contract\_product\_c,ext\_cproduct\_c,factory\_c

Where contract\_id,contract\_product\_id,factory\_id…

先找最小的结果集，只查询某个合同下的货物，只查询某个货物下的附件，只要查看这个合同

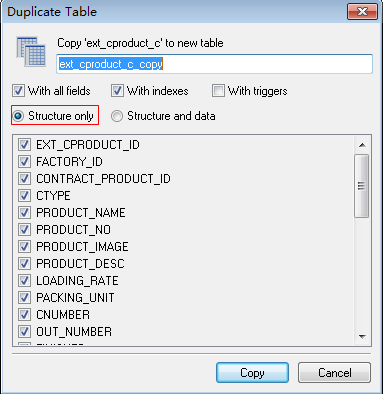
## 数据库的选型

* 单个数据库优先选择Oracle,DB2
* 集群mysql，读写分离，单独读数据数据库，单独写的数据库
* Web应用服务增加，可以增大访问量，同时分担数据库服务器压力。

分表，购销合同历史

当前的活动表，它和历史表的结构一样。

主模块中的所有表，历史中都有有。例如：购销合同，合同、货物、附件都需要对应的历史表。



创建domain

批量新增SQL，JDBC jdbcTemple spring

## 在一个系统架构中

一般访问数据两种方式

* hiberntate/mybatis 完成数据的持久化：新增，修改，删除，简单查询
* jdbc/jdbcTemlpate/dbutil批量操作，批量新增，批量修改，级联删除，统计查询

主要框架来完成日常业务，一般批量查询使用jdbc，对于这些模块，缓存就要注意使用。有时不要配置缓存。

## 购销合同归档，取消归档

--合同表数据进行归档

INSERT INTO contract\_his\_c

SELECT \* FROM contract\_c WHERE contract\_id IN ('x','y')

--货物的数据进行归档

INSERT INTO contract\_product\_his\_c

SELECT \* FROM contract\_product\_c WHERE contract\_product\_id IN (SELECT contract\_product\_id FROM contract\_product\_c WHERE contract\_id IN('x','y'))

--附件的数据进行归档

INSERT INTO ext\_cproduct\_his\_c

SELECT \* FROM ext\_cproduct\_c WHERE ext\_cproduct\_id IN (SELECT ext\_cproduct\_id FROM ext\_cproduct\_c WHERE contract\_product\_id IN (SELECT contract\_product\_id FROM contract\_product\_c WHERE contract\_id IN('x','y'))

--删除合同

DELETE FROM contract\_c

WHERE contract\_id IN ('x','y')

--删除货物

DELETE FROM contract\_product\_c

WHERE contract\_product\_id IN (SELECT contract\_product\_id FROM contract\_product\_c WHERE contract\_id IN('x','y'))

--删除附件

DELETE FROM ext\_cproduct\_c

WHERE ext\_cproduct\_id IN (SELECT ext\_cproduct\_id FROM ext\_cproduct\_c WHERE contract\_product\_id IN (SELECT contract\_product\_id FROM contract\_product\_c WHERE contract\_id IN('x','y'))

## 知识回顾

## 业务：货运管理，

购销合同，出口报运，home装箱单（预装箱），装箱，委托，发票（通知），财务统计。

## 【面试】工作流实现有什么好处？

杰信系统实现状态控制流程。

浏览每步流程开始时间，操作人，结束时间，审核人

突然在流程步骤中增加一步

工作流扩充流程时，非常方便，修改一下流程图，工作流自动记录所有操作的人和操作时间，操作意见。状态的代码配合工作流来实现。状态采用常量来定义，只有新增加的状态，才需要编码。

## 表设计

委托，发票，财务它们的表的主键既外键，它们的ID值都是装箱ID。因为它们4个的关系都是一对一。关联时最近配置，需要前面业务数据时，可以实现跳跃查询。这样查询的效率高。

## 页面设计

装箱新增，它需要在新增页面表现装箱和报运的关系。Div URL链接。

携带的信息ID，checkbox中；no也放到checkbox，用竖杠隔开，一个checkbox传递了两个值，无需再次查询数据库，提升性能，手工解串。（大量（超过10个字段）或者信息比较敏感，不推荐使用这种方法，推荐还使用通过id查找其他内容）

##  第九天 图形报表~让微软都羡慕的报表

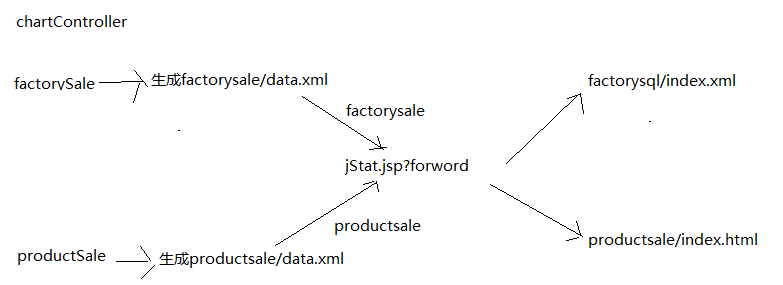
## Java要实现图形报表都有什么方式？

* jFreeChart 纯java API，利用生成一张图片。
* excel 图形报表，poi不能直接操作图形报表控件。变相解决，先做一个excel模板，插入好图表控件，然后利用poi动态设置它的数据。用户打开excel，图表直接展现。
* 第三方的报表软件，可视化开发。开发效率极高。可视化布局界面，配置数据源即可。（SQL的结果集）。收费。数巨报表（10w，每年服务费1w）。做一个应用程序，（C/S程序），IE上安装插件。兼容性。
* amCharts FLASH，只需要构建数据xml。

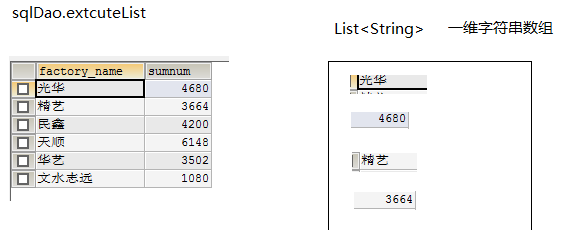
需要swf控件，例如饼形通用

setting.xml 不通用的

data.xml、data.txt，xml格式更加直观，不通用的



* sqlDao查询结果集的封装，通用

,

* 工作原理

amCharts只需要一次配置，公用flash，改造index.html，访问公用的swfobject.js和公用的flash swf原来文件。样式文件访问本地的，这样可以方便修改配置颜色，坐标，显示样式等。数据data.xml文件。将从数据库中读取的数据拼接成xml节点，写入到这个文本的文件中。转向jStat.jsp统一转向对应的forward参数目录下的index.html。（注意：html浏览器默认都有缓存，在做index.html要去掉缓存）

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />

<meta http-equiv="pragma" content="no-cache">

<meta http-equiv="Cache-Control" content="no-cache, must-revalidate">

<meta http-equiv="expires" content="Wed, 26 Feb 1997 08:21:57 GMT">

## jFreeChart、excel、amCharts PK比较

* jFreeChart 纯java api，缺点：图形非常粗糙，它实际是利用jFreeChart api生成的图片，表现力差，专门的api需要记忆。

## 生产厂家销售情况-饼形图

需求：

生产厂家销售情况，厂家+销售

SELECT

f.factory\_name,cp.sumnum

FROM

(SELECT factory\_id,factory\_name FROM factory\_c WHERE state=1) f

LEFT JOIN

(

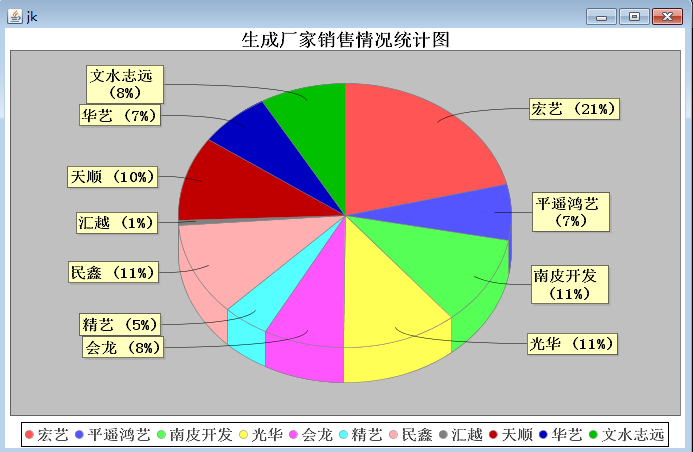
SELECT factory\_id,SUM(cnumber) AS sumnum FROM contract\_product\_c

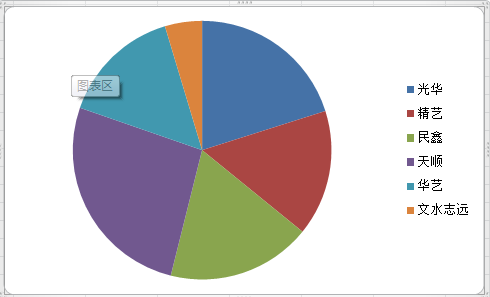
GROUP BY factory\_id

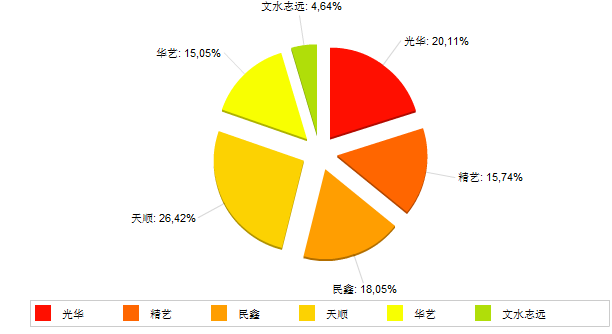
) cp

ON f.factory\_id=cp.factory\_id

WHERE cp.sumnum IS NOT null







## 产品销售排行-柱状图

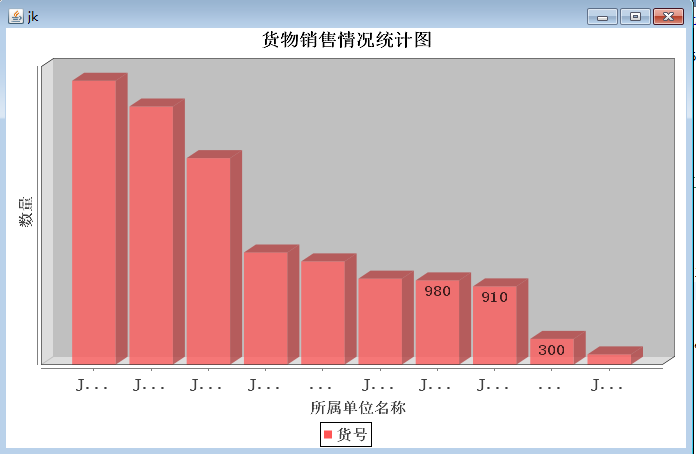
需求：

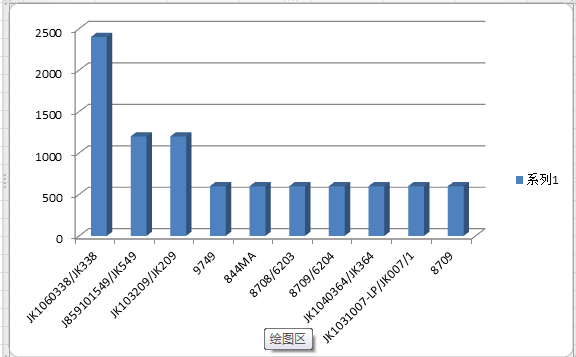
产品销售情况，畅销的产品的前10名，产品+销售

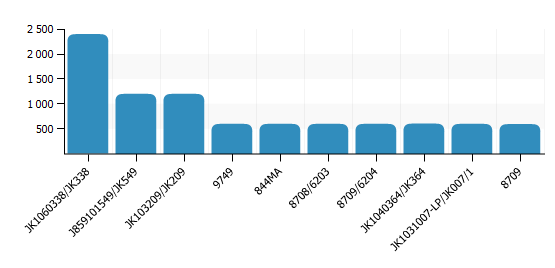
SELECT product\_no,SUM(cnumber) AS sumnum FROM contract\_product\_c

GROUP BY product\_no ORDER BY sumnum DESC

LIMIT 10



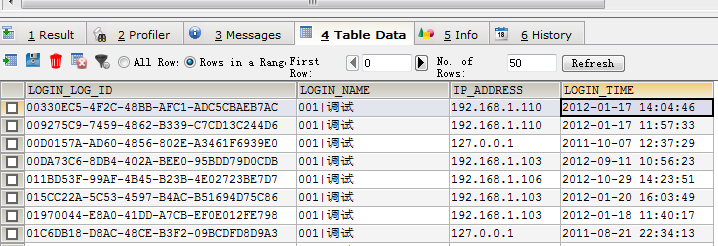




## 系统访问压力图-曲线图

需求：系统访问压力图

记录用户登陆系统，就记录一条



统计每天登陆系统的次数

SELECT SUBSTRING(login\_time,1,10),COUNT(login\_time) FROM login\_log\_p

GROUP BY SUBSTRING(login\_time,1,10)

统计24小时系统访问的次数

当数据不够时，利用临时表凑数据

SELECT

t.a1,IFNULL(p.countnum,0) AS countnum

FROM

(SELECT a1 FROM online\_t) t

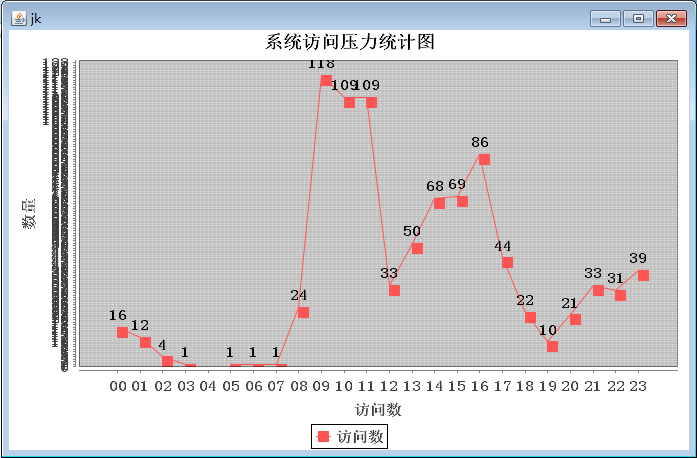
LEFT JOIN

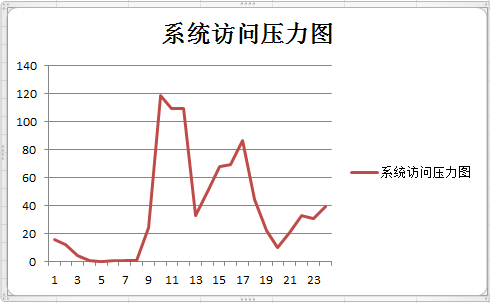
(

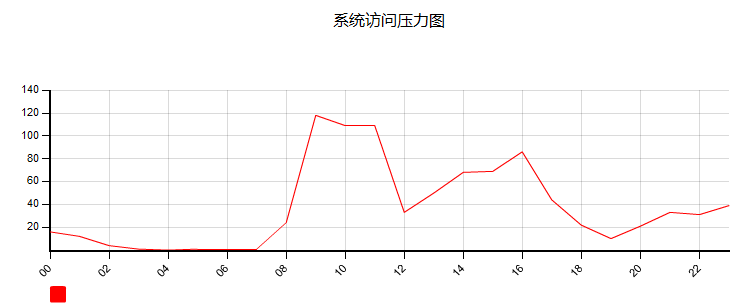
SELECT SUBSTRING(login\_time,12,2) AS a1,COUNT(login\_time) AS countnum FROM login\_log\_p

GROUP BY SUBSTRING(login\_time,12,2)

) p

ON t.a1=p.a1





总结：

图形报表业界已经相当成熟，提出公用，称作报表引擎。

## 系统的监控，系统想知道系统的访问瓶颈在哪里？

需求

系统瓶颈是指，访问最长时间，写数据操作，写磁盘文件操作。都比较耗时，用户体验不好。

* + 1. 找出系统这些操作慢的地方，
    2. 分析原因
    3. 优化，用户使用频繁模块，对其优化

实现一个性能监控的程序

## 早期使用继承方式

记录开始时间，记录结束时间，结束时间-开始时间=耗时

写到日志文件中，写到数据库中

## 拦截器

配置springmvc拦截器，记录开始时间，结束时间，打日志。

## 开发步骤

1. 写一个拦截器，实现一个接口HandlerInterceptor
2. 实现共用的变量，安全性
3. 配置springmvc拦截器配置，springmvc-servlet.xml

package cn.itcast.jk.interceptor;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.springframework.core.NamedThreadLocal;

import org.springframework.web.servlet.HandlerInterceptor;

import org.springframework.web.servlet.ModelAndView;

import common.Logger;

//功能：实现对所有类方法执行时间的监控

public class TimeInterceptor implements HandlerInterceptor {

//引入log4j日志

private static Logger log = Logger.getLogger(TimeInterceptor.class);

//利用ThreadLocal绑定一个变量，完成线程安全

NamedThreadLocal<Long> startTimeThreadLocal = new NamedThreadLocal<Long>("startTimeThreadLocal");

//处理类之前，执行preHandle方法

@Override

public boolean preHandle(HttpServletRequest request,

HttpServletResponse response, Object handler) throws Exception {

long startTime = System.currentTimeMillis(); //记录当前时间

startTimeThreadLocal.set(startTime); //绑定变量

return true;

}

//处理类之后，执行postHandler方法

@Override

public void postHandle(HttpServletRequest request,

HttpServletResponse response, Object handler,

ModelAndView modelAndView) throws Exception {

long stopTime = System.currentTimeMillis(); //记录结束时间

log.info(String.format("%s execute %d ms."

,request.getRequestURI()

, stopTime - startTimeThreadLocal.get()));

}

//所有的动作完成，执行一些方法

@Override

public void afterCompletion(HttpServletRequest request,

HttpServletResponse response, Object handler, Exception ex)

throws Exception {

// TODO Auto-generated method stub

}

}

sprimgmvc-servlet.xml中配置

<!-- 3.拦截器 -->

<mvc:interceptors>

<mvc:interceptor>

<mvc:mapping path="/\*\*"/><!-- 拦截所有请求 -->

<bean class="cn.itcast.jk.interceptor.TimeInterceptor"/>

</mvc:interceptor>

</mvc:interceptors>

## 项目中使用Log4j

## 导入jar包

## 配置log4j.properties

log4j.rootLogger=DEBUG, stdout

配置日志级别 OFF,FANIL,ERROR,WARN,INFO,DEBUG,TRACE,ALL 提供8个级别，log4j推荐开发者使用4个级别：

ERROR：错误信息，try，catch抛出异常，log.error(“”)

,WARN：警告信息

INFO：提示

DEBUG：BUG很多时候，监控过程变量，调试

顺序级别，级别由低到高，高级别的信息会包含打印低级别的信息。

Stout输出地点，包括：控制台，文件，数据库

log4j.rootLogger=DEBUG, stdout

#Console

log4j.appender.stdout=org.apache.log4j.ConsoleAppender

log4j.appender.stdout.layout=org.apache.log4j.PatternLayout

log4j.appender.stdout.layout.ConversionPattern=%-5p - %m%n

#LogFile

log4j.appender.file=org.apache.log4j.FileAppender

log4j.appender.file.File=../logs/jklog.log

log4j.appender.file.layout=org.apache.log4j.PatternLayout

log4j.appender.file.layout.ConversionPattern=%d{ABSOLUTE} %5p %c{1}:%L - %m%n

log4j.logger.org.apache=INFO

log4j.logger.cn.itcast.jk=DEBUG

## 调用方法：

public class TimeInterceptor implements HandlerInterceptor {

//引入log4j日志

private static Logger log = Logger.getLogger(TimeInterceptor.class);

类中类似System.out.println()

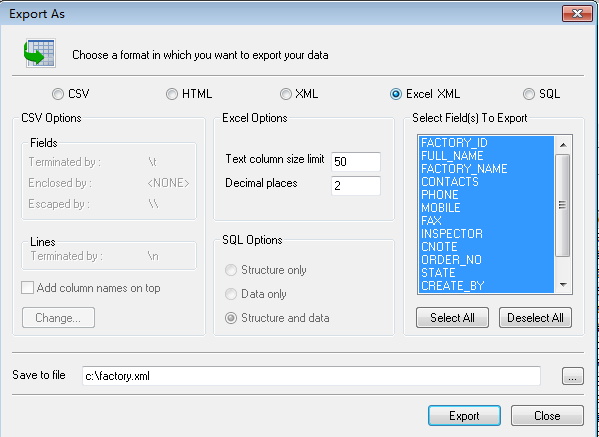
log.info(String.format("%s execute %d ms."

,request.getRequestURI()

, stopTime - startTimeThreadLocal.get()));

## POI导入

直接读取文件内容，形成SQL语句，批量插入数据库。



导入时，excel文档它单元格有类型。

1. 类型，根据不同的类型拼接不同的sql
2. 模板，自己定义一个模板，让用户将数据导入。模板要设置将所有的单元格设置为文本类型，导入的数据就都是文本。
3. 没有通用性，要根据具体业务做一个这样类。
4. 插入SQL语句。

## 知识回顾

## 项目统计分析核心模块

软件的核心价值：

举例：了解杰信有哪些现有的产品畅销，VIP客户，什么产品滞销。指导企业经营。

为企业经营决策提供数据支持。

在基础数据中挖掘有价值的信息。再次提炼。~ 数据挖掘。

## 图表报表

1. 第三方报表，用户展现效果最好，功能齐全，开发效果。收费
2. jFreeChart 古老，api繁杂，开发效率不高，生成图片
3. jReport 复杂
4. excel （企业中常采用）
5. javascript amCharts (收费)
6. amCharts Flash

## amCharts Flash 表现力酷炫，使用最简单，

1. SQL的数据源
2. 动态生成data.xml

## 图形报表常见

1. 饼形图
2. 柱状图
3. 曲线图

## 第十天 整合ApacheCXF实现WebService服务

## CXF apache webservice

简单快捷的开发一个web service程序

## 需求：

杰信系统是一个内部使用的业务系统，

VPN 企业向电信申请在inter网上的内部网。

客户实时了解订单情况，了解订单走到货运流程的哪一步。

原先：客户给销售打电话，销售人员询问公司同事，问船务。船务回馈信息。

给客户系统提供一个webservice服务，让他能实时查询订单的状态，

实现上面的要求，在出口报运中提供这个web service服务。

将出口报运查看发布成WebService服务，在这里客户可以随时查看出口报运单，查看订单的流程。

## 开发步骤：

1. 引入apache cxf jar

<dependency>

<groupId>org.apache.cxf</groupId>

<artifactId>cxf-rt-frontend-jaxws</artifactId>

<version>3.0.0-milestone2</version>

</dependency>

<dependency>

<groupId>org.apache.cxf</groupId>

<artifactId>cxf-rt-transports-http</artifactId>

<version>3.0.0-milestone2</version>

</dependency>

<!-- Jetty is needed if you're using the CXFServlet -->

<dependency>

<groupId>org.apache.cxf</groupId>

<artifactId>cxf-rt-transports-http-jetty</artifactId>

<version>3.0.0-milestone2</version>

</dependency>

1. 改造出口报运单Service为WebService

WebService公开方法中的参数不能使接口，必须实现类。

类改造为：

@WebService

public class ContractServiceImpl implements ContractService {

将不需要公开的方法设置为

@WebMethod(exclude=true)

实体po对象，必须序列化

public class Export implements Serializable

1. 配置cxf-servlet.xml 配置文件

<?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xmlns:jaxws="http://cxf.apache.org/jaxws"

xsi:schemaLocation=" http://www.springframework.org/schema/beans

http://www.springframework.org/schema/beans/spring-beans.xsd

http://cxf.apache.org/jaxws http://cxf.apache.org/schemas/jaxws.xsd">

<import resource="classpath:META-INF/cxf/cxf.xml"/>

<import resource="classpath:META-INF/cxf/cxf-servlet.xml"/>

<!-- 将cxf和spring进行整合 -->

<!-- 声明服务 -->

<bean id="exportService" class="cn.itcast.jk.service.impl.ExportServiceImpl">

<!-- 注入spring容器中dao -->

<property name="exportDao" ref="exportDaoImpl"/>

</bean>

<!-- 发布服务 implementor实现类，address发布访问地址-->

<jaxws:endpoint implementor="#exportService" address="/ExportServiceImpl"/>

</beans>

1. Web.xml配置

<!-- Apache CXF servlet -->

<servlet>

<servlet-name>cxf</servlet-name>

<servlet-class>org.apache.cxf.transport.servlet.CXFServlet</servlet-class>

<init-param>

<param-name>config-location</param-name>

<param-value>classpath:cxf-servlet.xml</param-value>

</init-param>

</servlet>

<servlet-mapping>

<servlet-name>cxf</servlet-name>

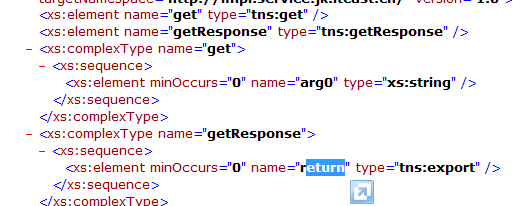
<url-pattern>/cxf/\*</url-pattern>

</servlet-mapping>

1. 发布服务 wsdl WebService说明书

<http://localhost/jk/cxf/ExportServiceImpl?wsdl>

1. 模拟调用这个webService服务 ajax SOAP xml



<script type="text/javascript">

var xmlHttpRequest = new ActiveXObject("Microsoft.XMLHTTP");

function sendMsg(){

/\*

1.创建ajax对象 XMLHTTP

2.打开一个链接open("POST",url,true)

3.soap xml request

4.设置requestHeader请求头

5.回调函数，处理返回的xml，将信息展现到页面

6.send(xml)

\*/

var findId = document.getElementById("findId").value;

findId = "%"+findId+"%";

if(findId==null){

alert("请输入编号");

return false;

}

var requestBody = "<soapenv:Envelope xmlns:soapenv=\"http://schemas.xmlsoap.org/soap/envelope/\" xmlns:q0=\"http://impl.service.jk.itcast.cn/\" xmlns:xsd=\"http://www.w3.org/2001/XMLSchema\" xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\">"

+ "<soapenv:Body><q0:wsview><arg0>"+findId+"</arg0></q0:wsview></soapenv:Body></soapenv:Envelope>"

var url = "${ctx}/cxf/ExportServiceImpl";

xmlHttpRequest.open("POST", url , true);

xmlHttpRequest.setRequestHeader("Content-Type","text/xml;charset=utf-8;");

xmlHttpRequest.onreadystatechange = \_back;

xmlHttpRequest.send(requestBody);

}

function \_back(){

if(xmlHttpRequest.readyState==4){ //处理完成

if(xmlHttpRequest.status==200){

var retxml = xmlHttpRequest.responseXML; //返回的XML内容

var ret = retxml.getElementsByTagName("return")[0];

if(ret==null){

alert("查询不到记录!");

return false;

}

//将数据插入到对应的td

document.getElementById("customerContract").innerHTML = ret.getElementsByTagName("customerContract")[0].text;

if(ret.getElementsByTagName("inputDate")[0]!=null){

document.getElementById("inputDate").innerHTML = ret.getElementsByTagName("inputDate")[0].text.substring(0,10);

}

if(ret.getElementsByTagName("consignee")[0]!=null){

document.getElementById("consignee").innerHTML = ret.getElementsByTagName("consignee")[0].text;

}

if(ret.getElementsByTagName("lcno")[0]!=null){

document.getElementById("lcno").innerHTML = ret.getElementsByTagName("lcno")[0].text;

}

if(ret.getElementsByTagName("shipmentPort")[0]!=null){

document.getElementById("shipmentPort").innerHTML = ret.getElementsByTagName("shipmentPort")[0].text;

}

if(ret.getElementsByTagName("transportMode")[0]!=null){

document.getElementById("transportMode").innerHTML = ret.getElementsByTagName("transportMode")[0].text;

}

if(ret.getElementsByTagName("priceCondition")[0]!=null){

document.getElementById("priceCondition").innerHTML = ret.getElementsByTagName("priceCondition")[0].text;

}

if(ret.getElementsByTagName("marks")[0]!=null){

document.getElementById("marks").innerHTML = "<pre>"+ret.getElementsByTagName("marks")[0].text+"</pre>";

}

if(ret.getElementsByTagName("remark")[0]!=null){

document.getElementById("remark").innerHTML = "<pre>"+ret.getElementsByTagName("remark")[0].text+"</pre>";

}

}else{

//出错信息

alert("你出错了!")

}

}

}

## 国际物流杰信项目面试总结

## 面试时如何讲解项目？

讲出三个层次，

1. 讲项目的背景，讲特色的业务
2. 讲业务的复杂度
3. 从业务角度牵扯出技术亮点

每一层都要挖陷阱，让面试官问问题。

从面试题中找出十个重点，把它们自己总结一段话，300~500字。

综合所学的所有知识

## 业务

## 购销合同

杰信和生产厂家要签订一个购销合同，购销合同包括三部分内容，包括合同主信息和多个货物信息和多个附件信息。合同和货物是一对多，货物和附件是一对多。货物和附件都有指定的生产厂家，附件还有一个分类需要取自基础表。大概字段有近90个。在购销合同中还有一个复杂的报表打印，这个表单内容非常复杂，它打印出合同的主信息，还有多个货物信息，一页纸如果是一个生产厂家，必须另起一页打印。一个页用户可以选择打印一款货物，也可以选择打印两款货物。购销合同合同数量比较多，积累10年多的数据，客户要求，将这些数据导出到excel中备份这些数据。我就研究实现了一个数百万海量数据的导出。

## 出口报运

杰信和客户签订合同后，找到生产厂家开始生产货物，于此同时向海关申报准备运输的货物，货物的信息和重量信息和体积信息，为装箱做准备。出口报运时货物信息来自合同下的货物信息，本身环境新增了部分货物的信息包括重量信息和体积信息。一个报运来自多个合同。在出口报运时，它其他业务的集中体现，在这里建立一个流程状态，它能反馈后续流程流转到哪里。在后续流程流转时，都要回执这个状态。我们系统给客户系统也提供了这个货运流程的跟踪，让可以可以在他的系统中实时查询出他下的订单的当前订单流转到哪里，是装箱，是委托，还是走到发票等等。我们的系统采用的是J2EEE技术实现，客户的系统采用.net技术实现。我们系统给它们的系统提供了一个WebService服务，使用Apache CXF实现，它可以异构系统来直接访问我们的系统，进行订单的流程查询。

## 货运管理流程

货运管理是杰信项目的二期的核心内容，它包括：购销合同、出口报运单、装箱单、委托书、发票通知、财务统计这几步。购销合同是客户和杰信签订合同后，向生产厂家签订的合同称为购销合同。然后杰信向海关进行出口报运的申请，海关批准后，杰信找货代公司订箱子，杰信找拖车公司订拖车，拖车拉着空箱到生产厂家指定的仓库在验货员的监督下装货，拖车拉着满箱到海关指定的码头指定的区域卸货。等待装船。在装船前，海关进行抽检。船来后进行转船。然后船离开装运港，我们杰信的这单订单就顺利完成。船离开前，杰信向客户快递一个发票通知，同时附上提单。客户收到发票，支付剩余尾款，收到提单，等船到目的港，卸货后，客户拿着提单提货。杰信最终进行内部财务核算，核算这单生意获取的利润，提交给领导审核。

## 设计

## 冗余设计

在设计时我们首先遵循三范式来设计表，然后才用反三范式的设计。三范式追求的是节省空间，创建表信息时，不能有重复的内容，例如：表必须有主键，列不能重复，列信息不能通过现有字段数据加工而成。这样实现存储的数据信息所占空间比较小，数据出处都唯一。反三范式，采用冗余技术实现表信息在查询时非常快捷，用户体验非常好，用户满意度非常高。

## 打断设计，跳跃查询

在杰信的业务中，它的货运管理流程分为6步，关联表达到10级以上。系统在建设之初我们就考虑到关联层级过多后，随着用户系统使用，数据线性增长，数据量不断增加，当数据量越来越大时，系统的访问会越来越慢。于是我们在表设计时，遵循一个原则，关联关系不四层。具体我们实现时通过主表中加一个字段，来存储主表和子表的一对多关系。它们都存储代理主键，它们之间用特殊符号隔开。这样在实际开发业务时，需要查询的时候，就可以跳过中间的层级，直接查询业务所需要的信息。在出口报运中就使用这种设计方式。出口报运单来自多个购销合同，出口报运单中需要获取合同下的货物的部分信息。新增时利用报运关联的合同的编号值，也就是打断设计的这个字段值，直接拼接为in子查询串，直接查询合同下的货物，这样跳过合同表。查询非常快捷。效率倍增。

## 一对一的特殊设计

在杰信系统中有个很有特点的地方，装箱、委托、发票、财务它们都是一对一的关系。一个委托来自一个装箱，一个发票来自一个委托，一个财务来自一个发票。它们每个业务都需要取报运下的货物信息。我们设计时，将委托、发票、财务的主键都存放装箱的ID。委托、发票、财务表主外键合一。这样设计后，实现业务只要获取到其中一个对象的ID，不论是装箱、还是委托的，还去其他的。都可以直接查询其他的对象。操作更加灵活。可以实现跳跃查询，本来财务需要报运下的货物和附件信息，本来要通过发票，委托，装箱，在找到关联的多个报运，在找到多个货物，在找到多个附件。我们新的设计后，直接找到装箱，找到装箱的打断字段，直接查询货物信息和附件信息。中间跳过5、6层。查询效率极高。

## 技术亮点

## 工作流

我们项目中采用工作流，Activiti 5，在货运管理中使用它，先实现货运管理流程，从购销合同销售人员的申请开始，销售人员处理完，提交给销售部领导进行审核；销售部领导审核通过后，提交到船运部门，船运专责进行报运处理，处理完提交给船运部报运领导进行审核，船运部报运领导审核通过后，提交给装箱专责，装箱处理完，提交给船运部装箱领导，船运部装箱领导审核通过后，提交给委托专责，委托处理完，提交给船运部发票专责，发票专责处理完， 提交给船运部发票领导，船运部发票领导审核通过后，提交给财务专责，财务专责进行统计分析后，交给财务部领导审核，财务部审核通过后，流程结束。如果其中的环节领导审核不通过，打回上一个流程。上一个流程补充缺失内容，重新提交审核。

## 百万海量数据的导出

采用poi来实现，开始时我们采用HSSF对象来实现，当我们导出6万5千多条数据时，系统报错，记录数超出excel 2003版本的限制，它支持65536行数据。远远不够存放我们的记录。我查阅了相关资料，将jar包升级poi 3.9版本，升级workbook对象为XSSF对象，它可以操作2007以上版本，这样将数据扩充到1048576行，列扩充到65536列。这样我们就实现一个sheet支持百万数据。然后打印又出现了一个问题，当打印将近30万时，系统越来越慢，CPU占用率很高，内存占用率很高，打印到40万，系统崩溃，报堆溢出错误。然后我项目组研究查阅资料，发现高版本的poi提供SXSSF对象它支持大数据量的导出，我们换成这个对象后，打印非常顺畅，CPU占用率不高，内存占用率不高。顺利完成数据的导出。

## 全文检索

购销合同查询非常频繁，用户要查，销售要查，船运要查，各个业务都需要查看合同数据，合同又有很多项，可以按合同号进行查询，按客户名称模糊查询，等等。我们系统采用lusece实现全文检索，我们将核心的查询的条件都作为分词索引，构建分词索引库，这样实现购销合同的一个全文检索，为各个业务提供查询。

## Aapche CXF

现今最强大的一个WebService实现。我们项目使用maven，依赖jar。修改报运的Service将其改造为一个WebService，对其不需要对客户开发的公用方法进行屏蔽，只留它查询某个报运信息的接口方法，然后配置cxf-servlet.xml。发布我们的报运状态跟踪的服务。这样服务端完成。客户从它的系统中使用ajax访问我们的系统，从系统中获取数据，提交一个SOAP requestXML，获得数据后，来解析xml，从中摘出数据，显示到他们的系统中。

## Maven

我们系统都采用Maven构建，我们公司搭建了一个私服，我们项目采用maven 工程，写好jar包的坐标，保存pom.xml文件时，myeclipse进行jar搜索。它先到本地仓库进行搜索，找到就引入项目中，如果找不到，它会去我们的公司的私服去搜索，搜索到，下载到本地仓库；搜索不到，它自动去远程maven中央仓库下载，下载到公司的私服中，再下载到本地仓库。