# 第4章 单表功能开发

## 关键知识点

本章通过最常用的“报销单信息基本操作”案例，讲述基础 CRUD 操作过程。通过对案例的分析，帮助您进一步理解EOS ,NUI,逻辑流。



**关键知识点：**







DataGrid 的使用

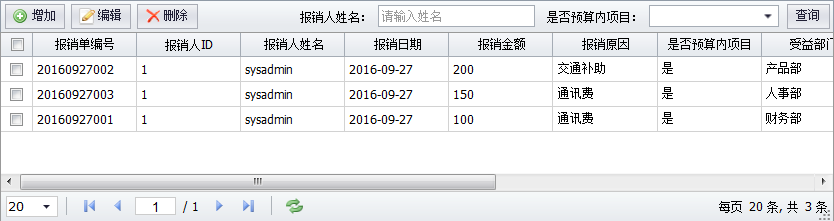
构件库中数据库操作类的使用

数据实体定义及使用

## 报销单查询案例开发

### 案例描述

案例的基本需求：页面输入报销人姓名、是否预算内项目作为查询条件，其中报销人姓名为模糊查询，是否预算内项目通过下拉的业务字典列表选取，点击【查询】后进行查询，查询结果分页显示，形式如图所示。



**图4-1** 报销单信息查询

### 案例开发

#### 数据准备:建表 TRAIN\_EXPENSEINFO

员工信息表 TRAIN\_EXPENSEINFO 表结构如图所示。

**表4-1** TRAIN\_EXPENSEINFO 的表结构

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名称** | **数据类型** | **数据长度** | **是否主键** | **备注** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| EXPID | NUMBER | 22 | YES | 报销单 ID |
| EXPCODE | VARCHAR2 | 64 | NO | 报销单编号 |
| EMPID | NUMBER | 22 | NO | 报销人 ID |
| EXPNAME | VARCHAR2 | 64 | NO | 报销人姓名 |
| EXPDATE | DATE | 7 | NO | 报销日期 |
| EXPMONEY | NUMBER | 22 | NO | 报销金额 |
| REASON | VARCHAR2 | 255 | NO | 报销原因 |
| ISBUDGETITEM | CHAR | 1 | NO | 是否预算内项目 |
| BENEFITORGID | NUMBER | 22 | NO | 受益部门 |
| MANAGER | VARCHAR2 | 64 | NO | 部门经理 |
| MANAGER\_ADVICE | VARCHAR2 | 255 |  | 部门经理意见 |
| AUDITOR | VARCHAR2 | 64 |  | 财务审核 |
| AUDITOR\_ADVICE | VARCHAR2 | 255 |  | 财务审核意见 |
| PROCESSINSTID | NUMBER | 22 |  | 流程实例 ID |

在数据库中执行 SQL 语句，建表 TRAIN\_EXPENSEINFO，并增加 7 条记录：

DROP TABLE TRAIN\_EXPENSEINFO CASCADE CONSTRAINTS;

CREATE TABLE TRAIN\_EXPENSEINFO (

EXPID NUMERIC(10, 0) NOT NULL, EXPCODE VARCHAR2(64) NOT NULL, EMPID NUMERIC(10,0) NOT NULL, EXPNAME VARCHAR2(64) NOT NULL, EXPDATE DATE,

EXPMONEY NUMERIC(12, 2) NOT NULL, REASON VARCHAR2(255), ISBUDGETITEM CHAR(1),

BENEFITORGID NUMERIC(10, 0), MANAGER VARCHAR2(64), MANAGER\_ADVICE VARCHAR2(255), AUDITOR VARCHAR2(64), AUDITOR\_ADVICE VARCHAR2(255), PROCESSINSTID NUMERIC(20, 0), PRIMARY KEY (EXPID)

);

-- Add comments to the columns

comment on column TRAIN\_EXPENSEINFO.EXPID is ' 报 销 单 ID'; comment on column TRAIN\_EXPENSEINFO.EXPCODE is '报销单编号'; comment on column TRAIN\_EXPENSEINFO.EMPID is ' 报 销 人 ID'; comment on column TRAIN\_EXPENSEINFO.EXPNAME is '报销人姓名'; comment on column TRAIN\_EXPENSEINFO.EXPDATE is '报销日期'; comment on column TRAIN\_EXPENSEINFO.EXPMONEY is '报销金额'; comment on column TRAIN\_EXPENSEINFO.REASON is '报销原因';

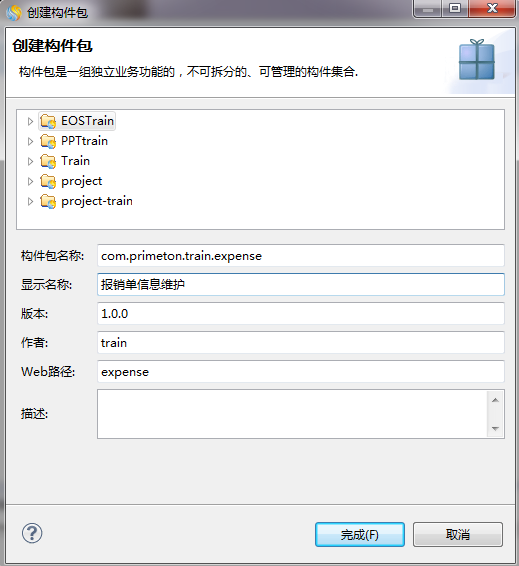
comment on column TRAIN\_EXPENSEINFO.ISBUDGETITEM is '是否预算内项目'; comment on column TRAIN\_EXPENSEINFO.BENEFITORGID is ' 受 益 部 门 '; comment on column TRAIN\_EXPENSEINFO.MANAGER is '部门经理';

comment on column TRAIN\_EXPENSEINFO.MANAGER\_ADVICE is '部门经理意见'; comment on column TRAIN\_EXPENSEINFO.AUDITOR is '财务审核';

comment on column TRAIN\_EXPENSEINFO.AUDITOR\_ADVICE is '财务审核意见'; comment on column TRAIN\_EXPENSEINFO.PROCESSINSTID is '流程实例 ID';

#### 构件包设计: com.primeton.train.expense

* 1. 在项目“EOSTrain”上点击右键，选择“创建构件包”。
  2. 在弹出的“创建构件包”对话框中，输入如图所示信息，点击【完成】。

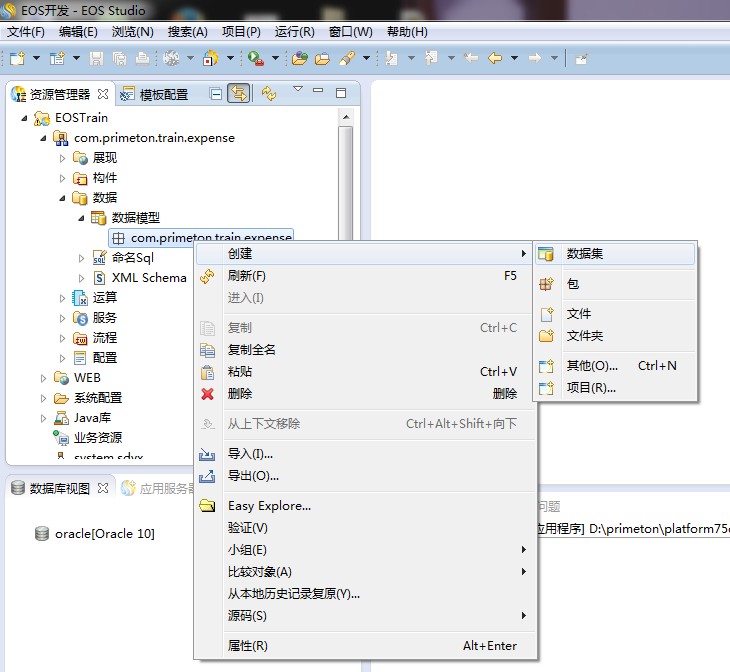


**图4-2** 创建构件包向导

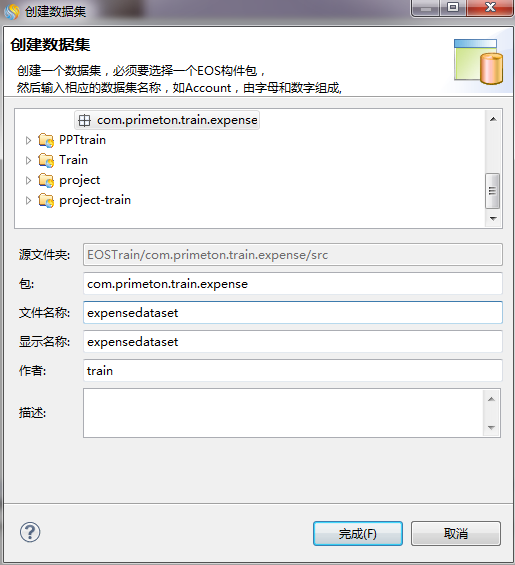
#### 数据实体

1. 创建数据集

构件包–>数据–>数据模型路径，右键：创建数据集expensedataset ，如图所示：



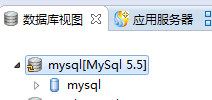
**图4-3** 创建数据集（一）



**图4-4** 创建数据集（二）

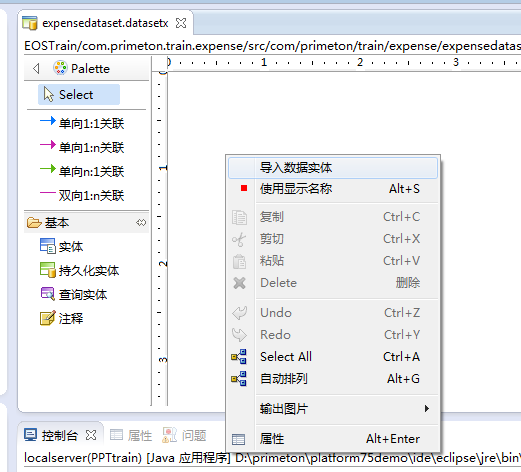
1. 导入数据实体

本案例说明：在将数据实体导入数据集之前要确保数据库是已连接状态的，若没有展示成如图所示，可以鼠标右键选中数据库，点击连接数据库；

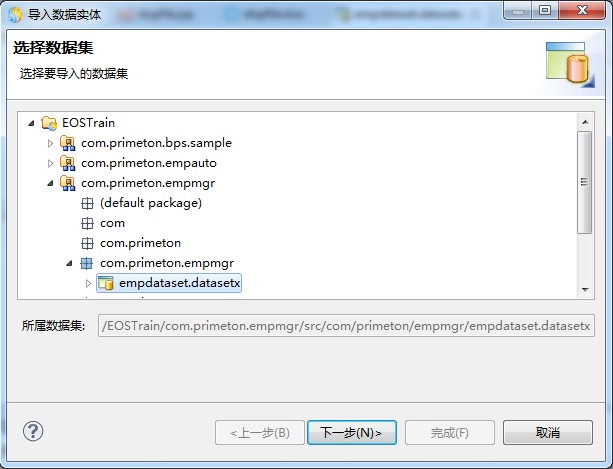


**图4-5** 确定数据库是否已连接

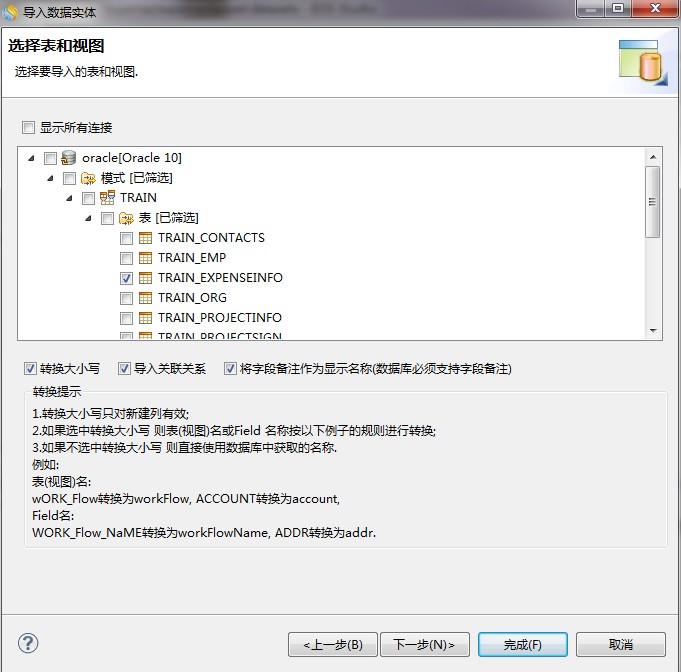
在数据集 expensedataset.datasetx 空白处右键，导入数据实体，如图所示：



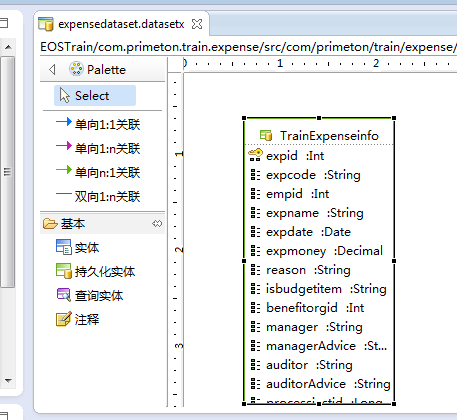
**图4-6** 导入数据实体（一） 弹出导入数据实体窗口，如图所示：

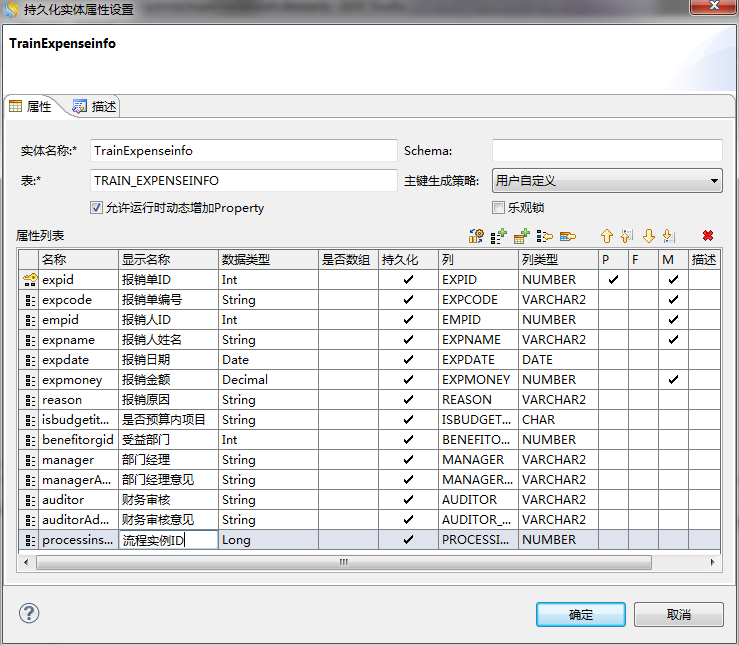


**图4-7** 导入数据实体（二）

下一步，在“连接”窗口内找到刚刚建好的表 TRAIN\_EXPENSEINFO，选中，如图所示：

**图4-8** 导入数据实体（三） 完成，出现数据实体 TrainExpenseinfo，如图所示：

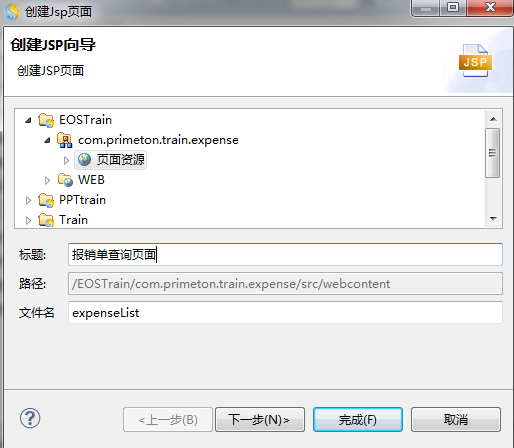


**图4-9** 导入数据实体（四） 双击数据实体，修改显示名称，如图所示：

**图4-10** 修改数据实体显示名称

#### 页面开发: expenseList.jsp

报销单信息查询页面的开发思路：根据 nui-sample 拷贝页面基础代码、设置查询条件、显示查询结果，提交页面数据的方法。

（1）在 com.primeton.train.expense 构件包下的“展现页面资源”上点右键，选择“创建Jsp 页面”，在弹出的创建 Jsp 页面对话框中，输入相关信息，如下图所示。

**图4-11** 创建Jsp 页面

自动生成的 Jsp 文件里展示了使用NUI 框架必需的代码，片段如下：

<%@page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8" session="false" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "[http://www.w3.org/TR/html4/loose.dtd"](http://www.w3.org/TR/html4/loose.dtd)>

<html>

<!--

* Author(s): primeton

- Date: 2014-11-25 16:57:31

* Description:

-->

<head>

<title>报销单查询页面</title>

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

<script src="<%= request.getContextPath() %>/common/nui/nui.js" type="text/javascript"></script>

</head>

<body>

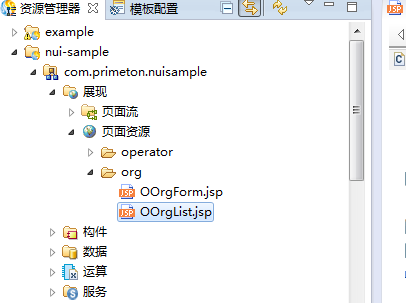
<script type="text/javascript"> nui.parse();

</script>

</body>

</html>

1. 根据 nui-sample 项目中 org/OOrgList.jsp 拷贝页面基础代码

根据第 3 节准备的 nui-sample 项目展现页面资源orgOOrgList.jsp 如图所示。

**图4-12** nui-sample

打开 OOrgList.jsp 页面，显示了查询页面的完整源代码，拷贝<body></body>标签体内的所有 HTML 代码至expenseList.jsp 页面的相同位置。接下来将在此段 HTML 代码基础上，根据实际需求调整代码，NUI 页面采用 DIV 实现页面布局。



1.此处参考的代码除了参考 NUI EOS Sample 示例项目，还可以参考 NUI Demo 中的 Data Grid

中的代码；

1. 设置查询条件

根据功能需求，以报销人姓名、是否预算内项目作为查询条件，其中报销人姓名为模糊查询，是否预算内项目通过下拉的业务字典列表选取。

页面原有的查询条件代码，如下：

<tr>

<td class=*"form\_label"*>

机构名称:

</td>

<td colspan=*"1"*>

<input class=*"nui-textbox"* name=*"criteria/\_expr[1]/orgName"*/>

<input class=*"nui-hidden"* name=*"criteria/\_expr[1]/\_op"* value=*"like"*>

<input class=*"nui-hidden"* name=*"criteria/\_expr[1]/\_likeRule"* value=*"all"*>

</td>

<td class=*"form\_label"*>

机构代码:

</td>

<td colspan=*"1"*>

<input class=*"nui-textbox"* name=*"criteria/\_expr[2]/orgCode"*/>

<input class=*"nui-hidden"* name=*"criteria/\_expr[2]/\_op"* value=*"="*>

</td>

</tr>

调整为：



页面通过发送 Ajax 请求提交 json 数据，并与后台逻辑流发生交互操作。一般页面通过NUI 控件提交形式如{"expname":"张","isbudgetitem":"1"}的 json 字符串，访问逻辑流时传入的参数即 expname 和 isbudgetitem。

该案例依次提交形式如“criteria/\_expr[1]/expname”的参数，目的是为了将所有的查询条件遵循一定的数据结构组装出一个 json 对象。最终提交的 json 数据格式将为：

{"criteria":{"\_expr":[{"expname":" 张 ","\_op":"like","\_likeRule":"all"},{"isbudgetitem ":"1","\_op":"="}]}}，将查询条件遵循查询实体对象 criteriaType 的数据结构组装给 criteria 变量，方便后续的逻辑流操作。

业务字典的设置，可以在 coframe 里完成，具体操作请参考 coframe 操作手册，其中是否预算内项目使用的“TRAIN\_YESNO”字典可以在 coframe 中增加。

<tr>

<td class=*"form\_label"*>

报销人姓名:

</td>

<td colspan=*"1"*>

<input class=*"nui-textbox"* name=*"criteria/\_expr[1]/expname"*/>

<input class=*"nui-hidden"* name=*"criteria/\_expr[1]/\_op"* value=*"like"*>

<input class=*"nui-hidden"* name=*"criteria/\_expr[1]/\_likeRule"* value=*"all"*>

</td>

<td class=*"form\_label"*>

是否预算内项目:

</td>

<td colspan=*"1"*>

<input class="nui-dictcombobox" dictTypeId="TRAIN\_YESNO" name="criteria/\_expr[2]/isbudgetitem" showNullItem="true" nullText=""/>

<input class="nui-hidden" name="criteria/\_expr[2]/\_op" value="=">

</td>

</tr>

1. 提交页面数据

在页面原有<script></script>代码基础上增加一段代码如下：

var grid = nui.get("datagrid1"); grid.load();

function search() {

var form = newnui.Form("#form1");

var data = form.getData(); //获取表单 JS 对象数据

grid.load(data); //datagrid 加载数据

}

function onGenderRenderer(e) {

return nui.getDictText('TRAIN\_YESNO',e.value);//设置业务字典值

}

1. 显示查询结果

查询结果页面使用表格控件 DataGrid，该控件内置 ajax 调用，通过设置 url 即与逻辑流交互数据，并分页显示查询结果。调整后的代码如下：

|  |  |
| --- | --- |
| <div id="datagrid1"dataField="expenses" class="nui-datagrid" style="width: 800px; height: 360px;" allowResize="true" url="com.primeton.train.expense.expense.expenseQuery.biz.ext" sizeList="[20,30,50,100]"pageSize="20" showPageIndex="true"  multiSelect="true">  <div property="columns">  <div type="checkcolumn"></div>  <div field="expid" headerAlign="center"allowSort="true"  visible="false">  报销单 ID  </div>  <div field="expcode" headerAlign="center" allowSort="true">  报销单编号  </div> | |
|  | <div field="empid" headerAlign="center" allowSort="true">  报销人 ID  </div>  <div field="expname" headerAlign="center" allowSort="true">  报销人姓名  </div>  <div field="expdate" headerAlign="center" allowSort="true">  报销日期  </div>  <div field="expmoney" headerAlign="center" allowSort="true">  报销金额  </div>  <div field="reason" headerAlign="center" allowSort="true">  报销原因  </div>  <div field="isbudgetitem" headerAlign="center" allowSort="true" renderer="OnIsRenderer">  是否预算内项目  </div>  <div field="benefitorgid" headerAlign="center" allowSort="true">  受益部门  </div>  <div field="manager" headerAlign="center" allowSort="true">  部门经理  </div>  <div field="managerAdvice" headerAlign="center" allowSort="true">  部门经理意见  </div>  <div field="auditor" headerAlign="center" allowSort="true">  财务审核  </div>  <div field="auditorAdvice" headerAlign="center" allowSort="true">  财务审核意见 |

</div>

<div field="processinstid" headerAlign="center" allowSort="true">

流程实例 ID

</div>

</div>

</div>



查询结果通过 DataGrid 控件的属性“url”指定发送 Ajax 请求调用的逻辑流，通过属性“dataFiled”指定查询结果取值的数据字段名。查询结果表格列集合 Columns 是一个数组， 通过属性“field”指定单元格值字段，将对应持久化实体 TrainExpenseinfo 的属性字段名。查询结果表格列的值字段“isbudgetitem”（是否预算内项目）保存在数据库里的字段值为“1”、 “0”，需要经过业务字典翻译成“是”、“否”作为页面的查询结果列显示，这里通过 isbudgetitem 列的表格绘制处理函数 renderer 设置业务字典值。

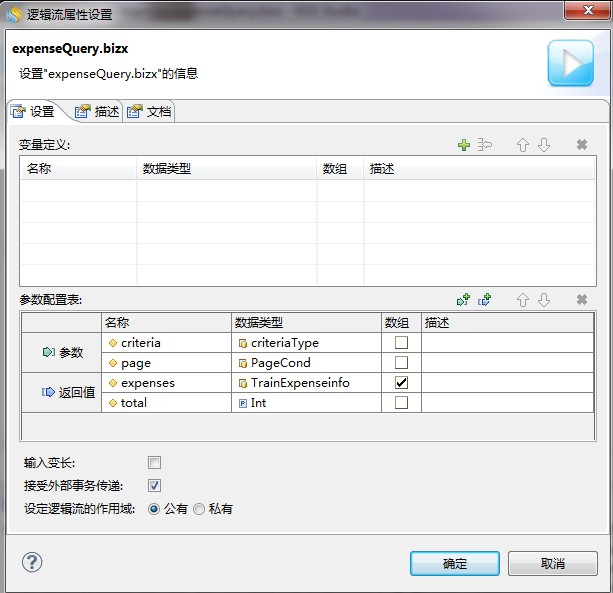
#### 逻辑流开发:expenseQuery.bizx

在“资源管理器”中，展开“EOSTrain com.primeton.train.expense 构件

com.primeton.train.expense”路径，在“com.primeton.train.expense”上点击右键，在弹出的菜单中选择“创建逻辑构件”名为“expense”。在逻辑构件 expense 上右键，创建逻辑流“expenseQuery”。

双击逻辑流“expenseQuery”操作界面的空白处，弹出逻辑流属性设置窗口，如图所

示：

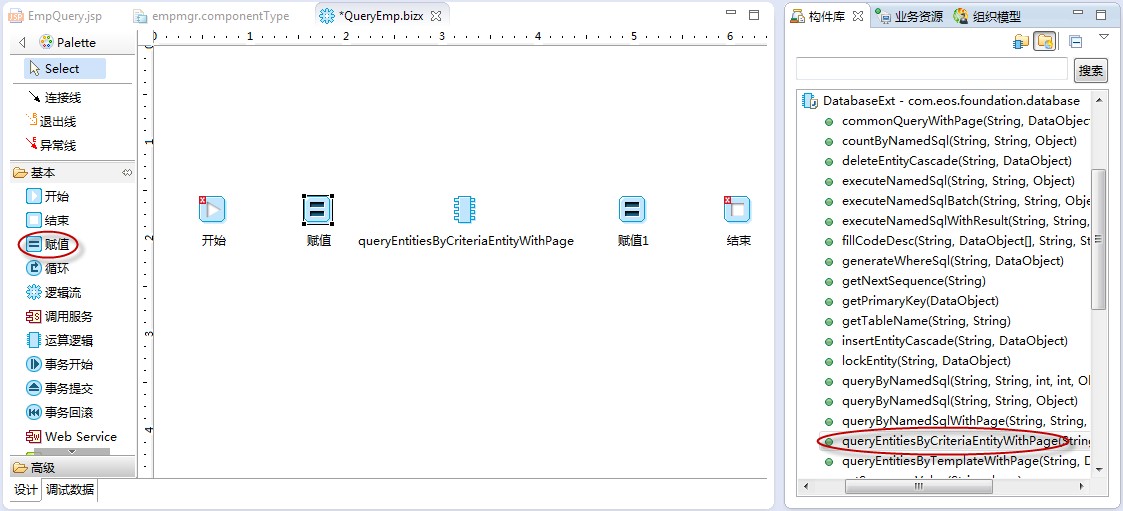


**图4-13** 定义参数和返回值

返回值中，必须返回一个名为 total 的值，类型通过下拉菜单选择“Int”，作为统计出的记录条数使用，实现页面分页效果。

拖出 3 个图元：2 个“赋值”，第 3 个图元来自构件库 com.eos.fundation ->运算逻辑 ->

DatabaseExt : queryEntitiesByCriteriaEntityWithPage，如图所示：



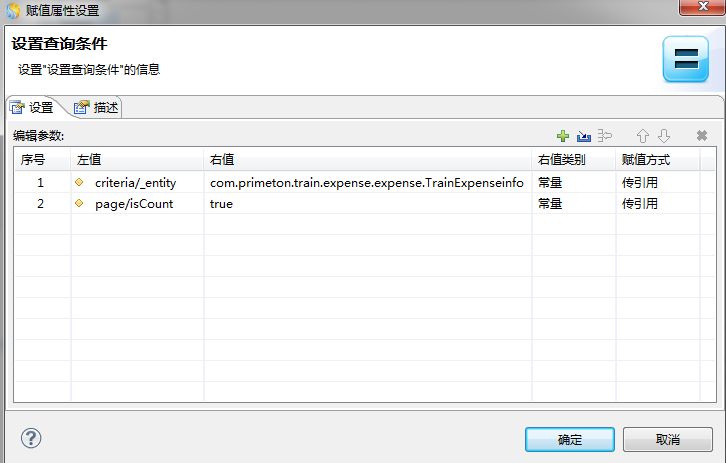
**图4-14** 逻辑流图元排列

用连接线将所有图元连接在一起，双击图元下方的名称，根据功能修改每个名称，如图所示：



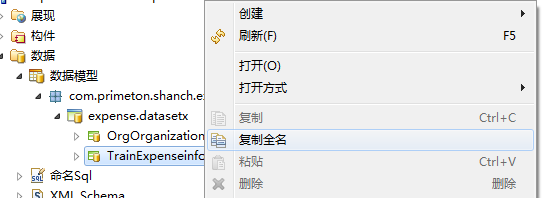
**图4-15** 逻辑流

双击“设置查询条件”图元，进行属性设置，如图所示：

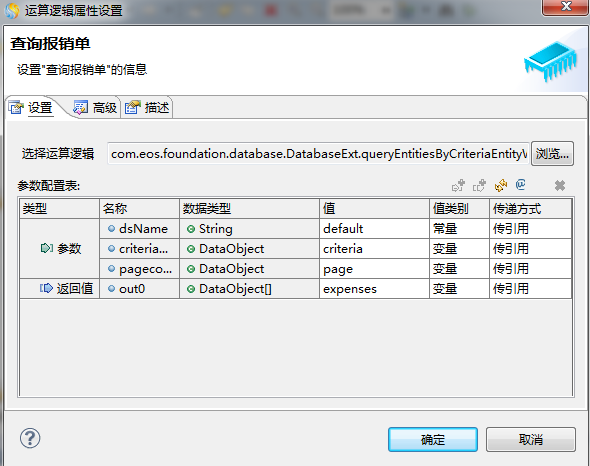


**图4-16** “设置查询条件”属性设置

本案例说明：其中第一个参数的右值对应的是 TrainExpenseinfo 数据实体的全名，可以在左侧资源管理器处找到 TrainExpenseinfo 数据实体，右键复制全名，如下图：

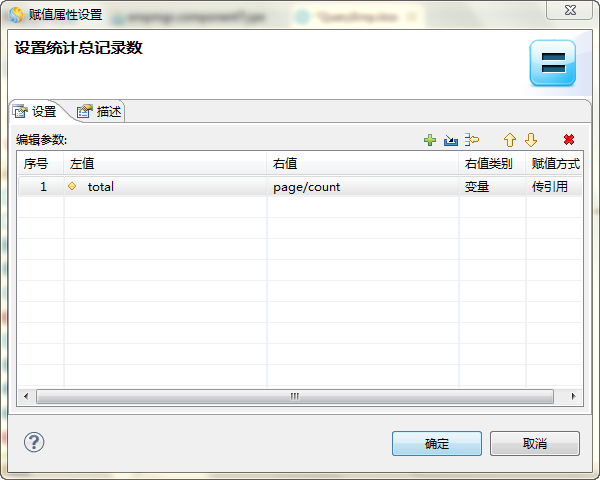


**图4-17** “设置查询条件”属性设置双击“查询报销单”图元，进行属性设置，如图所示：



**图4-18** “查询报销单”属性设置

双击“设置统计总记录数”，设置赋值属性，如图所示。左值即逻辑流的返回值 total，用来存放统计出的记录条数，作为整个逻辑流的返回值返回到页面中，右值代表查询结果的总记录条数。



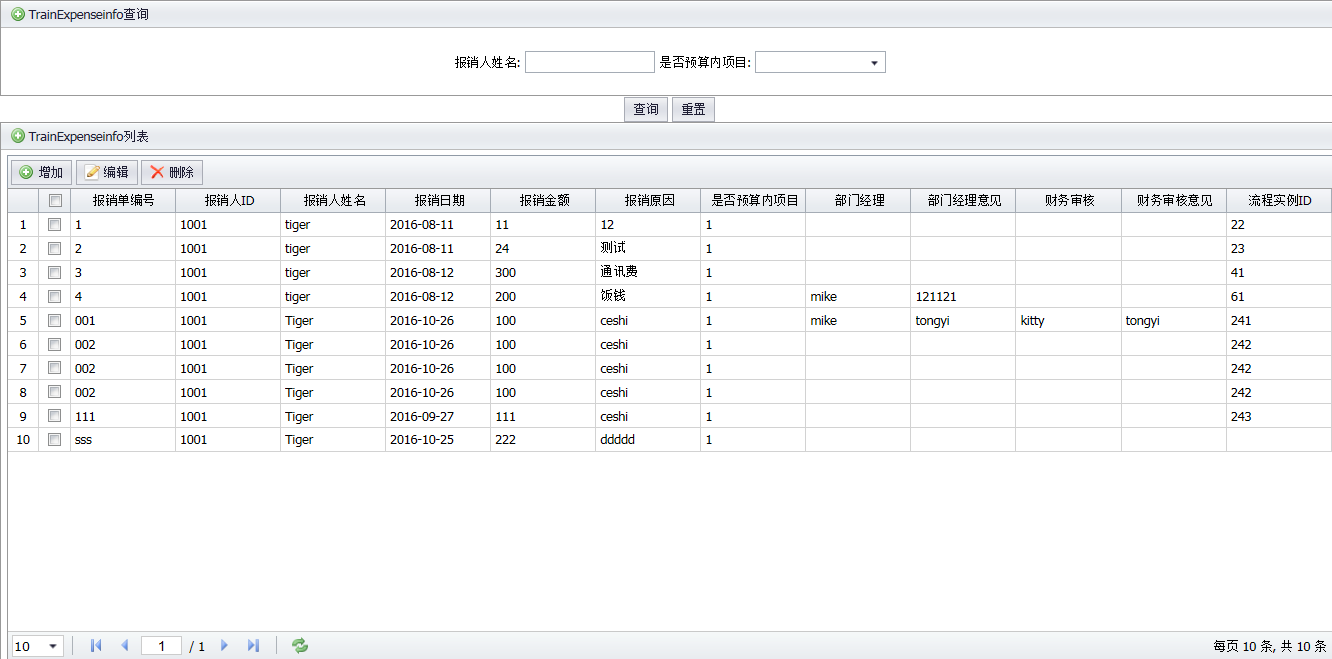
#### 完善页面

**图4-19** “设置统计总记录数”属性设置

双击打开“expenseList.jsp”，指定 DataGrid 的 url 为上一步开发的逻辑流全名加固定后缀。通过资源管理器，选中“expenseQuery.bizx”，右键下拉菜单选择“复制全名”，获得的逻辑流全称加上“.biz.ext”的后缀名得到 url 属性值，如下所示：

|  |  |  |  |
| --- | --- | --- | --- |
| <div id=*"datagrid1"* class=*"nui-datagrid"* style="width:*800px*;height:*280px*;" allowResize=*"true"* | | | |
|  | url="com.primeton.train.expense.expense.expenseQuery.biz.ext" idField="expid" | |  |
|  | dataField="expenses" |  | |
| multiSelect=*"true"*> | | | |

#### 测试运行

1. 启动EOS Server，启动成功后，在资源管理器视图，展开“EOSTrain com.primeton.train.expense 展现页面资源”，选中下面的启动 EOS Server。启动成功后， 在资源管理器视图，展开“EOSTrain com.primeton.train.expense 展现页面资源”，选中下面的expenseList.jsp，右键，在弹出的菜单中选择“复制 URL”。
2. 打开浏览器，在地址栏中右键粘贴，显示“http://127.0.0.1:8080/default/expense/expenseList.jsp”，回车，显示程序入口，如图所示。

**图4-20** 初始页面

1. 输入查询条件，点击查询后，显示结果如图所示，则程序运行正确。



**图4-21** 查询页面

**知识点解读：**

* **criteriaType 中\_entity 条件必须有**、**标\*为数组、标?为可出现 0-1 次、多个查询条件组合默认为 and**
* **分页条件规则：grid.load 时系统自动封装、逻辑流分页参数名必须为 page、逻辑流返回值名为 total**

本案例说明：

* 1. dataField=”expenses” 与 逻 辑 流 中 定 义 的 返 回 值 expenses ， 以 及queryEntitiesByCriteriaEntityWithPage 的返回值expenses 一致才会有正确的结果展示在页面中；
  2. queryEntitiesByCriteriaEntityWithPage 构件的功能及参数说明可以参考studio 帮助中的介绍，studio帮助帮助内容，将构件的名字放至搜索栏里，点击转至即可；

## 报销单新增案例开发

### 案例描述

案例的基本需求：基于报销单查询页面，点击【新增】，弹出窗口显示报销单信息录入页面，录入信息保存完毕回到父查询页面。

### 案例开发

1. 页面开发

打开“expenseList.jsp”页面，在<script></script>标签内增加一段 JS，对应【新增】按钮的 onclick="add()"事件函数。增加的部分可从 OOrgList.jsp 中找到。

打开 OOrgList.jsp 页面，拷贝下段代码并修改成如下黄色标出部分：

function add() {

nui.open({

url: "<%=request.getContextPath() %>/expense/expenseForm.jsp", title: "新增记录", width: 600, height: 360,

onload: function () {//弹出页面加载完成variframe = this.getIFrameEl();

var data = { pageType: "add"};//传入页面的 json 格式

iframe.contentWindow.setFormData(data);

},

ondestroy: function (action) { grid.reload();

}

});

}

url 改为 url: "<%=request.getContextPath() %>/expense/expenseForm.jsp"

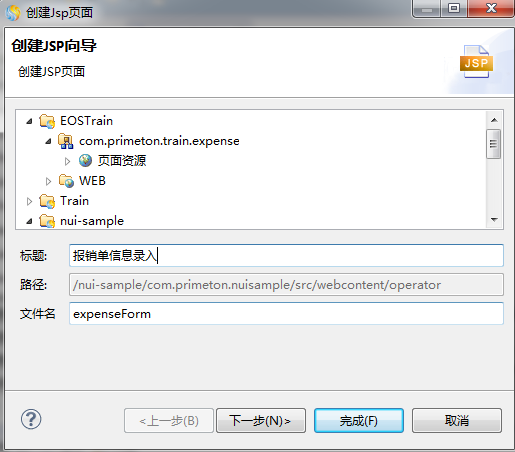


通过 nui.open 方法弹出子窗口，可以根据参数调整子窗口大小等。

1. 子页面开发

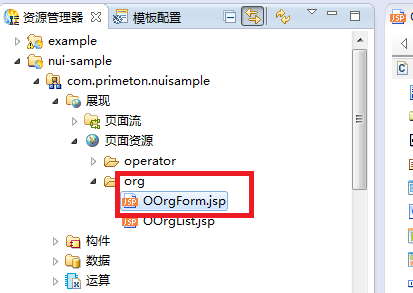
在 com.primeton.train.expense 构件包下的“展现页面资源”上点右键，选择“创建

Jsp 页面”，在弹出的创建 Jsp 页面对话框中，输入相关信息，点击【完成】，如图所示。



**图4-22** 创建Jsp 页面

利用 nui-sample 里弹出页面代码构造页面，打开 OOrgForm.jsp 页面，



**图4-23** 打开OOrgForm.jsp 页面

将 OOrgForm.jsp 的<form></form>标签内的所有代码复制黏贴至expenseForm.jsp 页面代码的<body></body>标签内，根据员工信息录入字段的需求调整代码如下：

<!-- 标识页面是查看(query)、修改(edit)、新增(add) -->

<input name=*"pageType"* class=*"nui-hidden"*/>

<div id=*"dataform1"* style="padding-top:*5px*;">

<!-- hidden 域 -->

<input class=*"nui-hidden"* name=*"trainexpenseinfo.expid"*/>

<input class=*"nui-hidden"* name=*"trainexpenseinfo.empid"*/>

<input class=*"nui-hidden"* name=*"trainexpenseinfo.expname"*/>

<input class=*"nui-hidden"* name=*"trainexpenseinfo.processinstid"*/>

<table style="width:*100%*;height:*100%*;table-layout:*fixed*;" class=*"nui-form-table"*>

<tr>

<td class=*"form\_label"*>

报销单编号:

</td>

<td colspan=*"1"*>

<input class=*"nui-textbox"* name=*"trainexpenseinfo.expcode"*

required=*"true"*/>

</td>

<td class=*"form\_label"*>

报销日期:

</td>

<td colspan=*"1"*>

<input class=*"nui-datepicker"* name=*"trainexpenseinfo.expdate"*/>

</td>

</tr>

<tr>

<td class=*"form\_label"*>

报销金额:

</td>

<td colspan=*"1"*>

<input class=*"nui-textbox"* name=*"trainexpenseinfo.expmoney"*/>

</td>

<td class=*"form\_label"*>

报销原因:

</td>

<td colspan=*"1"*>

<input class=*"nui-textbox"* name=*"trainexpenseinfo.reason"*/>

</td>

</tr>

<tr>

<td class=*"form\_label"*>

是否预算内项目:

</td>

<td colspan=*"1"*>

<input class=*"nui-dictcombobox"* dictTypeId=*"train\_yesno"* name=*"trainexpenseinfo.isbudgetitem"*/>

</td>

<td class=*"form\_label"*>

受益部门:

</td>

<td colspan=*"1"*>

<input class=*"nui-textbox"* name=*"trainexpenseinfo.benefitorgid"*/>

</td>

</tr>

</table>

<div class=*"nui-toolbar"* style="padding:*0px*;" borderStyle=*"border:0;"*>

<table width=*"100%"*>

<tr>

<td style="text-align:*center*;" colspan=*"4"*>

<a class=*"nui-button"* iconCls=*"icon-save"* onclick="onOk()">

保存

</a>

<span style="display:*inline-block*;width:*25px*;">

</span>

<a class=*"nui-button"* iconCls=*"icon-cancel"* onclick="onCancel()">

取消

</a>

</td>

</tr>

</table>

</div>

</div>

员工信息录入页面通过 NUI 控件的 name 属性指定提交数据的变量名，通过“trainexpenseinfo.expname”



的数据格式，将字段值组装给 trainexpenseinfo 变量，方便后续的逻辑流操作。

拷贝 JS 部分即<script></script>中间部分，用来提交表单数据，发送 Ajax 请求，与后台发生交互，代码调整如下：

<script type="text/javascript">

nui.parse();

**function** saveData(){

**var** form = **new** nui.Form("#dataform1");

form.setChanged(**false**);

//保存

**var** urlStr = "com.primeton.train.expense.expense.expenseAdd.biz.ext";

**var** pageType = nui.getbyName("pageType").getValue();//获取当前页面是编辑还是新增状态

//编辑

**if**(pageType=="edit"){

urlStr = "com.primeton.nuisample.oorgbiz.updateOOrg.biz.ext.biz.ext";

}

form.validate();

**if**(form.isValid()==**false**) **return**;

**var** data = form.getData(**false**,**true**);

**var** json = nui.encode(data);

$.ajax({

url:urlStr, type:'POST', data:json, cache:**false**,

contentType:'text/json', success:**function**(text){

**var** returnJson = nui.decode(text);

**if**(returnJson.exception == **null**){

CloseWindow("saveSuccess");

}**else**{

nui.alert("保存失败", "系统提示",

**function**(action){

**if**(action == "ok" || action == "close"){

//CloseWindow("saveFailed");

}

});

}

}

});

}

//关闭窗口

**function** CloseWindow(action) {

**if** (action == "close"&& form.isChanged()) {

**if** (confirm("数据被修改了，是否先保存？")) { saveData();

}

}

**if** (window.CloseOwnerWindow)

**return** window.CloseOwnerWindow(action);

**else** window.close();

}

//确定保存或更新

**function** onOk() { saveData();

}

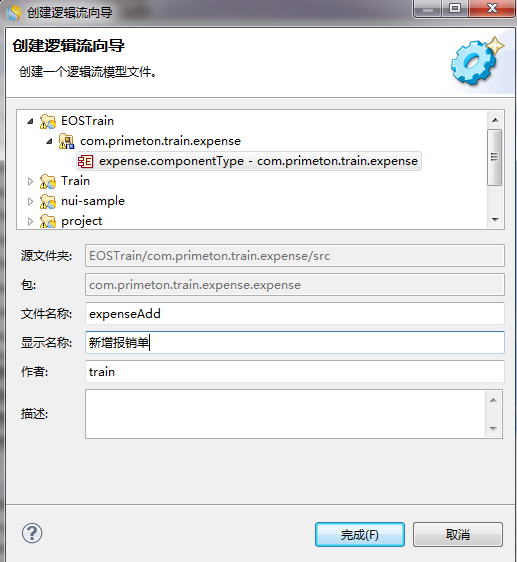
//取消

**function** onCancel() { CloseWindow("cancel");

}

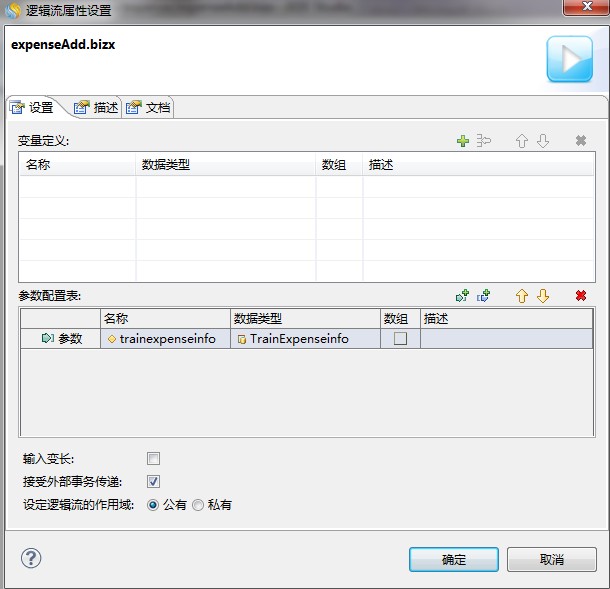
</script>

1. 逻辑流开发

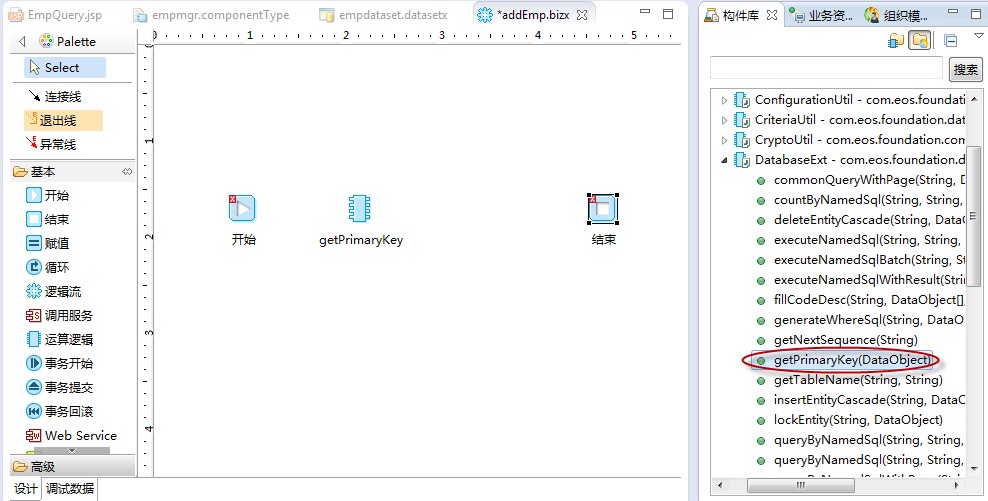
在逻辑构件“expense”上点右键，选择“创建逻辑流”，在弹出的创建逻辑流对话框中，输入相关信息，点击【完成】，如图所示。

**图4-24** 创建逻辑流

右侧展开的逻辑流编辑面板里默认产生了“开始”和“结束”两个图元，双击空白处定义逻辑流的参数和返回值，如图所示。参数来自于 expenseForm.jsp 页面提交组装出的 json 对象 trainexpenseinfo，类型通过下拉菜单选择“数据实体—浏览”，在弹出的的对话框里选择“TrainExpenseinfo”。

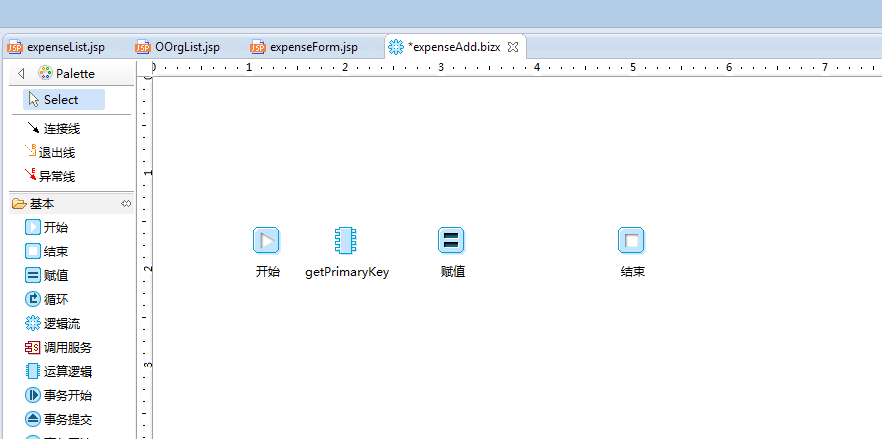


**图4-25** 定义参数

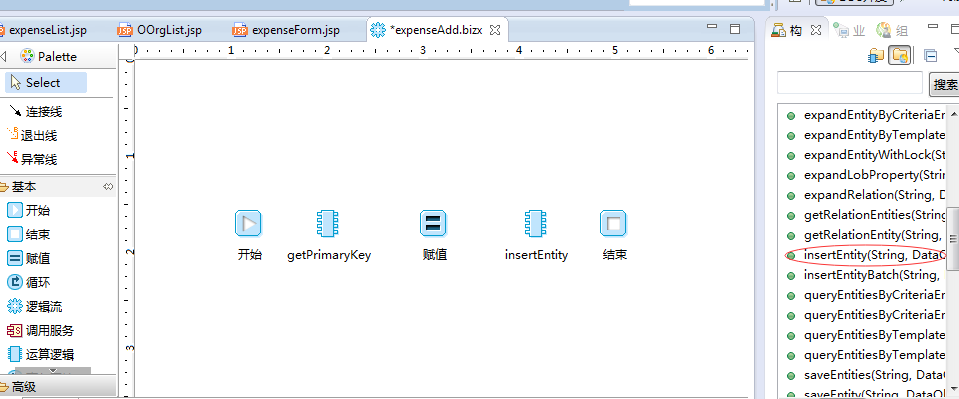
展开右边“构件库”视图中的“com.eos.foundation运算逻辑DatabaseExt”，选中 “getPrimaryKey（DataObject）”运算逻辑，它是获取持久化实体主键序列号值的构件，将该运算逻辑拖拽到逻辑流“expenseAdd”编辑器中，如图所示。

**图4-26** 逻辑流图元排列

从左侧拖一个赋值图元放至 getPrimaryKey 后面，如图：



**图4-27** 逻辑流图元排列

再继续展开右边“构件库”视图中的“com.eos.foundation运算逻辑DatabaseUtil”，选中“insertEntity（String,DataObject）”运算逻辑，将该运算逻辑拖拽到逻辑流“expenseAdd”编辑器中，排列图元如图所示。

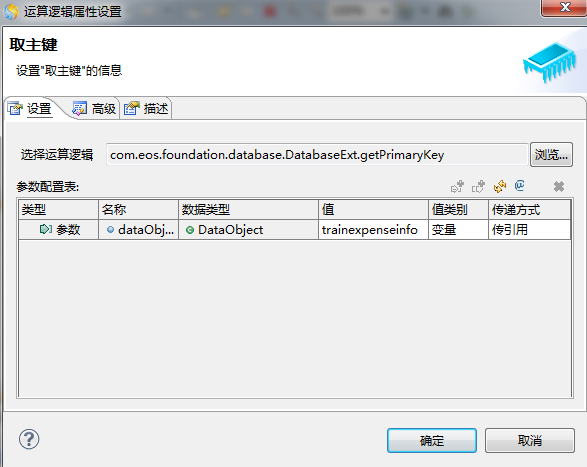
**图4-28** 逻辑流图元排列

选择左侧选用板的“连接线”，依次连接各图元，修改图元的显示名称。最终的逻辑流如图所示。

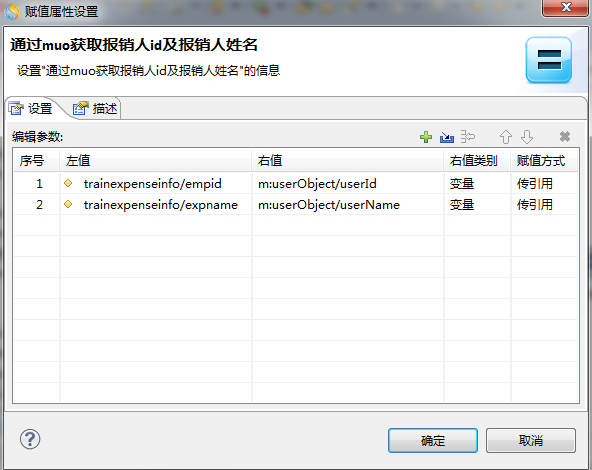


**图4-29** 逻辑流

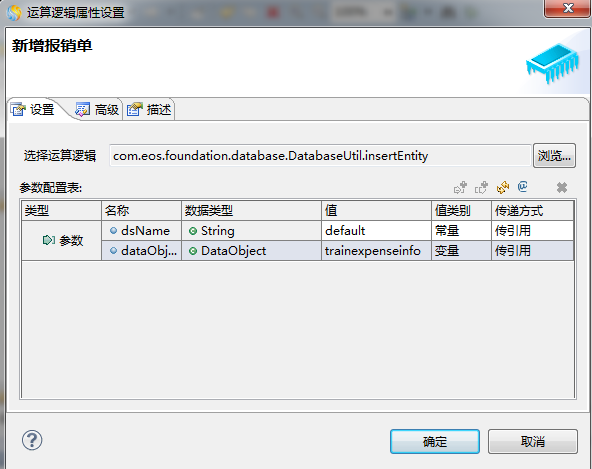
双击“取主键”运算逻辑图元，在弹出的“运算逻辑属性设置”对话框中，设置该方法的参数。这个方法有一个参数： 数据实体数组是图中定义的传入逻辑流中的参数trainexpenseinfo。具体设置如图所示。



**图4-30** “取主键”属性设置

双击“通过 muo 获取报销人 id 及报销人姓名”运算逻辑图元，设置值如下图：

**图4-31** “取主键”属性设置

双击“新增报销单”运算逻辑图元，在弹出的“运算逻辑属性设置”对话框中，设置该方法的参数。这个方法有二个参数：数据源默认为 default，数据实体数组是图中定义的传入逻辑流中的参数trainexpenseinfo。具体设置如图所示，点击【确定】后，保存逻辑流。

**图4-32** “新增员工”属性设置

1. 完善页面

打开“expenseForm.jsp”，修改 JS 部分的 addURL 参数值，通过资源管理器，选中 “expenseAdd.bizx”，右键下拉菜单选择“复制全名”，获得的逻辑流全称加上“.biz.ext”的后缀名得到 url 参数值，如下所示：

var addURL = "com.primeton.train.expense.expense.expenseAdd.biz.ext";

### 测试运行

* + - 1. 浏览器中访问地址“http://127.0.0.1:8080/default/expense/expenseList.jsp”，显示查询页面，点击工具栏的【增加】按钮，弹出窗口，如图所示。



**图4-33** 新增页面

输入报销单信息，点击【确定】，并重新加载查询结果，新增记录成功。

**知识点解读：**

* **通过 nui.open 弹出子页面**
* **通过 form.validate 进行表单数据校验**
* **通过.设置的 xpath，getData(false,true)为默认参数**
* **通过/设置的 xpath，getData(false,false)为默认参数**
* **新增逻辑中的主键规则：根据实体名称.主键属性名称（如： OOperator.operatorId**

**从 EOS\_UNIQUE\_TABLE 表中获取当前最大值，在缓存中递增处理**

## 报销单删除案例开发

### 案例描述

案例的基本需求：基于查询页面，通过查询结果列表的复选框选择多条记录，点击工具栏里的【删除】按钮，批量删除多条记录，并返回查询页面。

### 案例开发

* + - 1. 页面开发

双击打开“expenseQuery.jsp”页面，在<script></script>标签内增加一段 JS，对应【删除】按钮的onclick="remove()"事件函数。增加的部分可从 OOrgList.jsp 中找到。

打开 OOrgList.jsp 页面，拷贝 function remove()中的代码并做如下修改：

function remove() {

var rows = grid.getSelecteds(); //获取所有选中的行对象var data = {expenses:rows};

if (confirm("确定删除选中记录？")) {

grid.loading("操作中，请稍后 ");

nui.ajax({ url:

"org.gocom.components.nui.demo.newdataset.impl.TEmployee.deleteEmployee.biz.ex t",

type: "post", data: json, cache: false,

contentType: 'text/json', success: function (text) { grid.reload();

},

error: function () {

}

});

}

} else {

alert("请选中一条记录");

}

}

//序列化成 json 字符串

var json = nui.encode(data); if (rows.length> 0) {

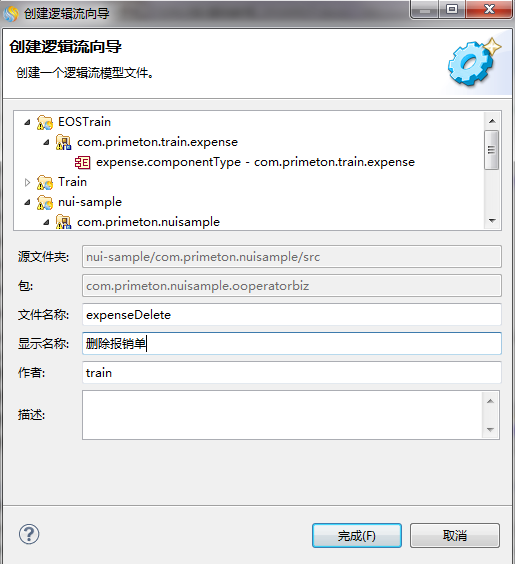


删除操作通过 nui.ajax 方法发送 Ajax 请求，与后台发生交互，指定交互的 url，交互数据等。

其中 DataGrid 的相关 js 方法，如 getSelecteds，可参考 NUI 的 API 文档说明。var data = {expenses:rows}是为了遵循 json 的数据结构，将获取到的多条数据组装给expenses 变量，方便后续的逻辑流操作。

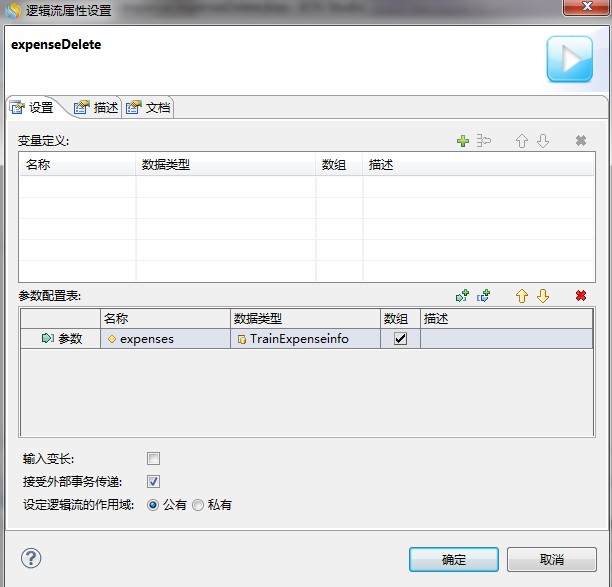
* + - 1. 逻辑流开发

在逻辑构件“expense”上点右键，选择“创建逻辑流”，在弹出的创建逻辑流对话框中，输入相关信息，点击【完成】，如图所示。

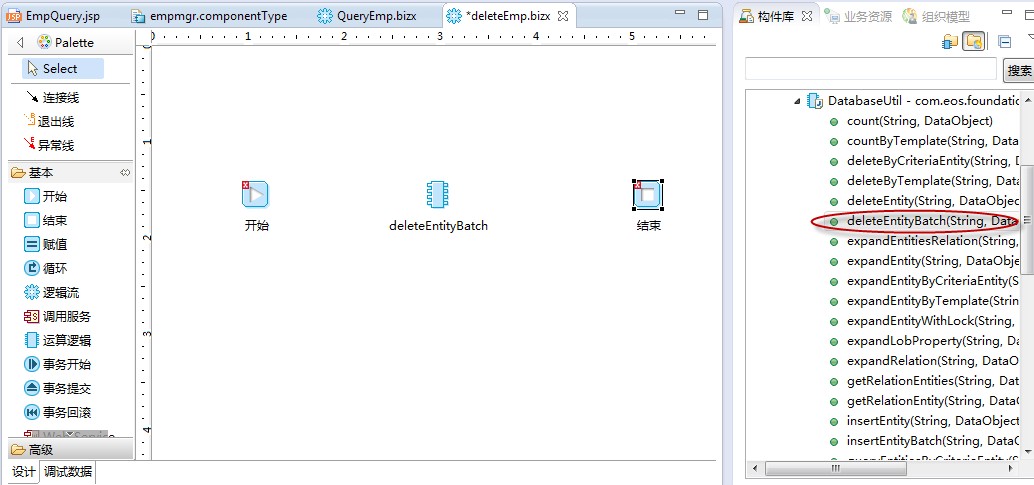


**图4-34** 创建逻辑流

右侧展开的逻辑流编辑面板里默认产生了“开始”和“结束”两个图元，双击空白处定义逻辑流的参数和返回值，如图所示。参数来自于 ExpenseQuery.jsp 页面提交的复选框选中行组装出的 json 对象 expenses，类型通过下拉菜单选择“数据实体—浏览”，在弹出的的对话框里选择“TrainExpenseinfo”，并勾选“数组”选项。



**图4-35** 定义参数

展开右边“构件库”视图中的“com.eos.foundation运算逻辑DatabaseUtil”，选中 “deleteEntityBatch（String,DataObject[]）”运算逻辑，它是用于根据主键批量删除多条记录的构件，将该运算逻辑拖拽到逻辑流“expenseDelete”编辑器中，排列图元如图所示。

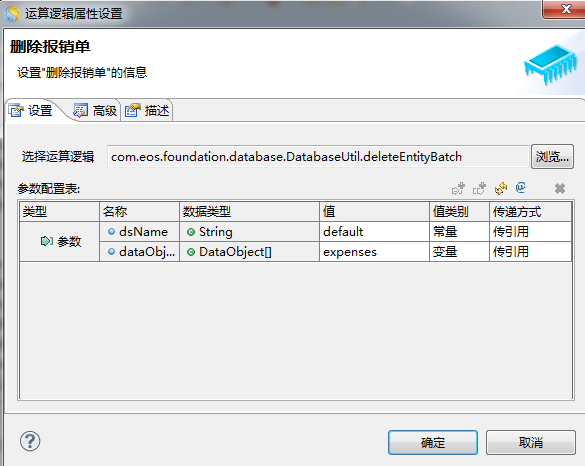
**图4-36** 逻辑流图元排列

选择左侧选用板的“连接线”，依次连接各图元，修改图元的显示名称。最终的逻辑流如下图所示。



**图4-37** 逻辑流

双击“删除报销单”运算逻辑图元，在弹出的“运算逻辑属性设置”对话框中，设置该方法的参数。这个方法有二个参数：数据源默认为 default，数据实体数组是图中定义的传入逻辑流中的参数expenses。具体设置如下图所示，点击【确定】后，保存逻辑流。



**图4-38** “删除报销单”属性设置

* + - 1. 完善页面

打开“expenseQuery.jsp”，修改 remove 函数里 nui.ajax 方法的 url 参数值，通过资源管理器，选中“expenseDelete.bizx”，右键下拉菜单选择“复制全名”，获得的逻辑流全称加上 “.biz.ext”的后缀名得到 url 参数值，如下所示：

nui.ajax({

url: "com.primeton.train.expense.expense.expenseDelete.biz.ext",

type: "post", data: json, cache: **false**,

contentType: 'text/json', success: **function** (text) {

grid.reload();

},

error: **function** () {

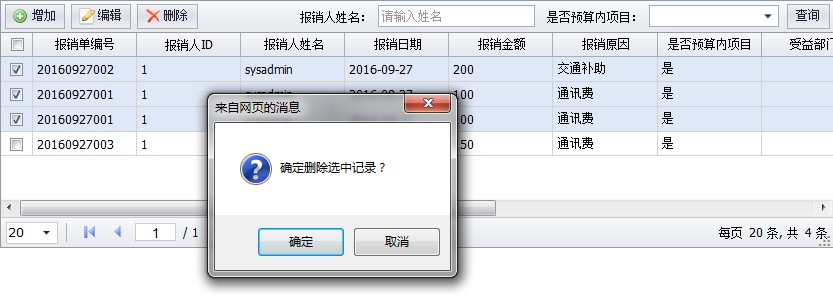
}

});

### 测试运行

* + - 1. 浏览器中访问地址“http://127.0.0.1:8080/default/expense/expenseList.jsp”，显示多条查询结果记录，通过复选框勾选多条记录，如图所示。

**图4-39** 初始页面

点击【删除】，弹出提示框“确定删除选中记录？”，点击“确定”，删除多条记录后返回查询页面，并重新加载查询结果，如图所示。

**图4-40** 删除页面

**知识点解读：**

* **getSelecteds()获取所有选中行**
* **将选中行组装给 json 参数**
* **逐条循环删除，可能部分删除成功，部分删除失败**
* **批量删除，所有操作隶属于一个原子事务**
* **逐条循环删除，所有操作隶属于一个原子事务**

本案例说明：对于删除案例，除了使用批量删除构件外还可以使用 deleteEntity 构件配合循环的形式来实现。

## 报销单编辑案例开发

### 案例描述

案例的基本需求：基于查询页面，通过查询结果列表的复选框选择一条记录，点击工具栏里的【编辑】按钮，修改一条记录，并返回查询页面。

### 案例开发

* + - 1. 页面开发

双击打开“expenseQuery.jsp”页面，在<script></script>标签内增加一段 JS，对应【编辑】按钮的onclick="edit()"事件函数。增加的部分可从 OOrgList.jsp 中找到。

打开 OOrgList.jsp 页面，拷贝 function edit()中的代码并做如下修改： 修改为：



编辑操作通过 nui.ajax 方法发送 Ajax 请求，与后台发生交互，指定交互的 url，交互数据等

**function** edit() {

**var** row = grid.getSelected();//获取当前选中的行；

**if** (row) {

nui.open({

url: "<%=request.getContextPath() %>/expense/expenseForm.jsp",

title: "编辑员工", width: 600, height: 360, onload: **function** () {

**var** iframe = **this**.getIFrameEl();

**var** data = {pageType:”edit”,record:{trainexpenseinfo:row}}; iframe.contentWindow.setFormData(data);

},

ondestroy: **function** (action) {

grid.reload();

}

});

} **else** {

alert("请选中一条记录");

}

}

在 expenseForm.jsp 页面里面 SaveData()的 function 后面加如下方法：

//页面间传输 json 数据

**function** setFormData(data){

//跨页面传递的数据对象，克隆后才可以安全使用

**var** infos = nui.clone(data);

//保存 list 页面传递过来的页面类型：add 表示新增、edit 表示编辑

nui.getbyName("pageType").setValue(infos.pageType);

//如果是点击编辑类型页面

**if** (infos.pageType == "edit") {

**var** json = infos.record;

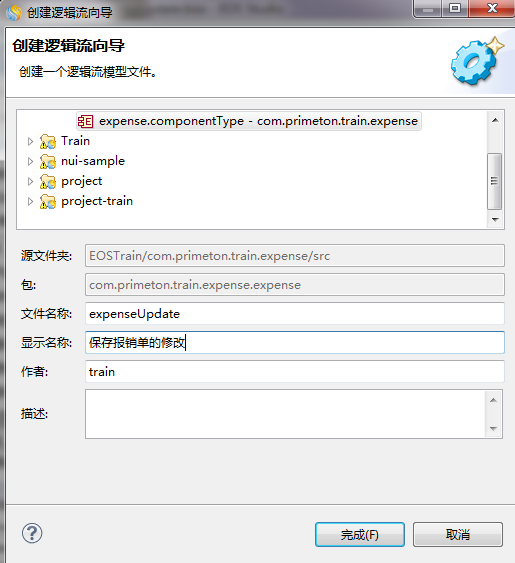
**var** form = **new** nui.Form("#form1");//将普通 form 转为 nui 的 form form.setData(json);

form.setChanged(**false**);

}

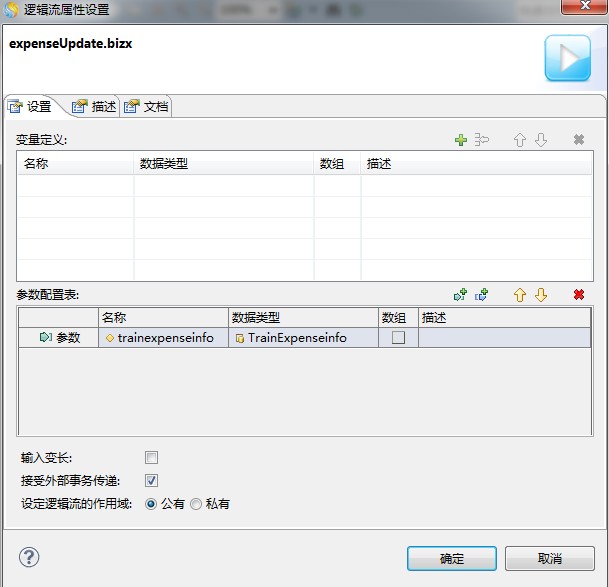
}

* + - 1. 逻辑流开发

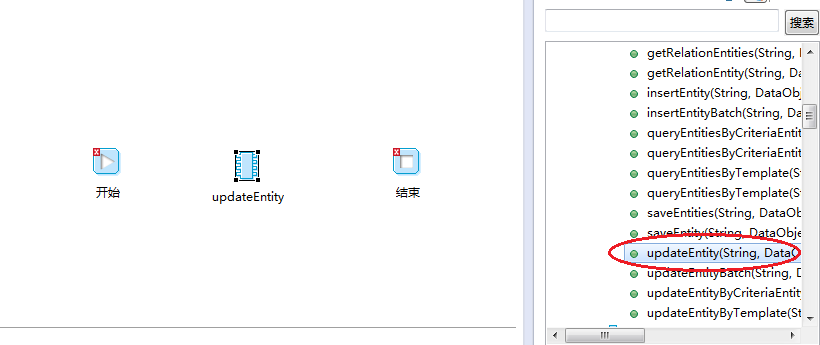
在逻辑构件“expense”上点右键，选择“创建逻辑流”，在弹出的创建逻辑流对话框中，输入相关信息，点击【完成】，如图所示。

**图4-41** 报销单修改

右侧展开的逻辑流编辑面板里默认产生了“开始”和“结束”两个图元，双击空白处定义逻辑流的参数，如图所示。参数来自于 expenseForm.jsp 页面提交的复选框选中行组装出的 json 对象 trainexpenseinfo，类型通过下拉菜单选择“数据实体—浏览”，在弹出的的对话框里选择“TrainExpenseinfo”。



**图4-42** 逻辑流设置

展开右边“构件库”视图中的“com.eos.foundation运算逻辑DatabaseUtil”，选中“updateEntity（String,DataObject）”运算逻辑，它是用于根据主键扩展查询一条记录的构件，将该运算逻辑拖拽到逻辑流“expenseUpdate”编编辑器中，排列图元如图所示。

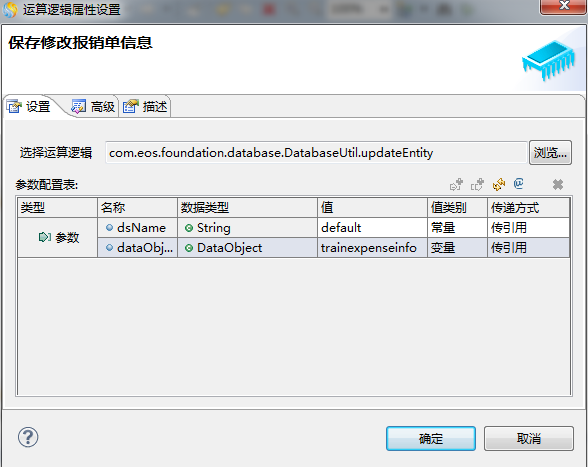
**图4-43** 逻辑流设置

选择左侧选用板的“连接线”，依次连接各图元，修改图元的显示名称。最终的逻辑流如下图所示。



**图4-44** 逻辑流截图

双击“保存修改报销单信息”运算逻辑图元，在弹出的“运算逻辑属性设置”对话框中， 设置该方法的参数。这个方法有二个参数：数据源默认为 default，数据实体数组是图中定义的传入逻辑流中的参数 trainexpenseinfo。具体设置如下图所示，点击【确定】后，保存逻辑流。



**图4-45** 修改报销单图元

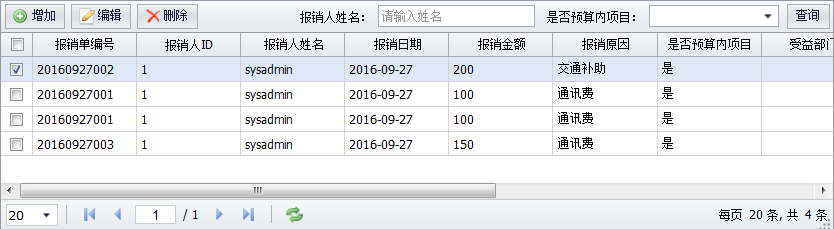
* + - 1. 完善页面

打开“expenseForm.jsp”，修改 JS 部分的 updateURL 参数值，通过资源管理器，选中 “expenseUpdate.bizx”，右键下拉菜单选择“复制全名”，获得的逻辑流全称加上“.biz.ext”的后缀名得到url 参数值，如下所示：

**var** updateURL = "com.primeton.train.expense.expense.expenseUpdate.biz.ext";

### 测试运行

浏览器中访问地址“http://127.0.0.1:8080/default/expense/expenseList.jsp”，显示多条查询结果记录，通过复选框勾选多条记录，如图所示。



**图4-46** 编辑页面

点击【编辑】，弹出修改页面，可进行修改，修改后点击【确定】，保存修改内容并重新加载查询结果。

**知识点解读：**

* **getSelecteds()获取选中行**
* **将选中行组装给 json 参数**
* **克隆跨页面传递的数据对象**
* **从后台加载记录**
* **通过 Form 的 setFormData 方法设置表单数据**