# 项目目的:

管理客户关系

# 项目技术:

Jsp,JavaScript,servlet,JDBC等技术

# 项目页面分析及流程解析:

Index.jsp

登录页面由四个相关的jsp页面构成:

index.jsp frame.jsp welcome.jsp top.jsp(对应下面顺序)

|  |
| --- |
| <jsp:forward page=*"/frame.jsp"* /> |
| <frameset rows=*"20%,\*"*>  <frame src=*"*<c:url value=*'/top.jsp'*/>*"* name=*"top"*/>  <frame src=*"*<c:url value=*'/welcome.jsp'*/>*"* name=*"main"*/>  </frameset> |
| <body>  <h1 align=*"center"*>欢迎登录本系统</h1>  </body> |
| <body style="text-align: *center*;">  <h1>客户关系管理系统</h1>  <a href=*"*<c:url value=*'/add.jsp'*/>*"*>添加客户</a> 　|  <a href=*"*<c:url value=*'/CustomerServlet?method=findAll'*/>*"*>查询客户</a>　|  <a href=*"*<c:url value=*'/query.jsp'*/>*"*>高级搜索</a>  </body> |

添加客户跳转到add.jsp,add.jsp页面收集数据交由servlet处理,响应到list.jsp页面,在该页面完成分页查询(全部数据)

查询客户跳转到list.jsp页面

高级查询跳转到query.jsp页面,在该页面完成查询数据的收集,响应到list.jsp页面

# 功能分析与实现

## 添加客户:

Add.jsp

|  |
| --- |
| <title>添加客户</title>  ............................  <body>  <h3 align=*"center"*>添加客户</h3>  <form action=*"*<c:url value=*'/CustomerServlet'*/>*"* method=*"post"*>  <input type=*"hidden"* name=*"method"* value=*"add"*>  <table border=*"0"* align=*"center"* width=*"40%"* style="margin-left: *100px*;">  <tr>  <td width=*"100px"*>客户名称</td>  <td width=*"40%"*>  <input type=*"text"* name=*"cname"*/>  </td>  <td align=*"left"*>  <label id=*"cnameError"* class=*"error"*>&nbsp;</label>  </td>  </tr>  <tr>  <td>客户性别</td>  <td>  <input type=*"radio"* name=*"gender"* value=*"男"* id=*"male"*/>  <label for=*"male"*>男</label>  <input type=*"radio"* name=*"gender"* value=*"女"* id=*"female"*/>  <label for=*"female"*>女</label>  </td>  <td>  <label id=*"genderError"* class=*"error"*>&nbsp;</label>  </td>  </tr>  <tr>  <td>客户生日</td>  <td>  <input type=*"text"* name=*"birthday"* id=*"birthday"* readonly=*"readonly"*/>  </td>  <td>  <label id=*"birthdayError"* class=*"error"*>&nbsp;</label>  </td>  </tr>  <tr>  <td>手机</td>  <td>  <input type=*"text"* name=*"cellphone"*/>  </td>  <td>  <label id=*"cellphoneError"* class=*"error"*>&nbsp;</label>  </td>  </tr>  <tr>  <td>邮箱</td>  <td>  <input type=*"text"* name=*"email"*/>  </td>  <td>  <label id=*"emailError"* class=*"error"*>&nbsp;</label>  </td>  </tr>  <tr>  <td>描述</td>  <td>  <textarea rows=*"5"* cols=*"30"* name=*"description"*></textarea>  </td>  <td>  <label id=*"descriptionError"* class=*"error"*>&nbsp;</label>  </td>  </tr>  <tr>  <td>&nbsp;</td>  <td>  <input type=*"button"* value=*"添加客户"* onclick="add()"/>  <input type=*"reset"* value=*"重置"*/>  </td> |

表单提交到*CustomerServlet?method=add*

|  |
| --- |
| //创建QueryRunner对象  **private** QueryRunner qr = **new** TxQueryRunner();  //插入用户  **public** **void** add(Customer cstm){  **try** {  //给出SQL模板  String insert = "insert into customer values(?,?,?,?,?,?,?)";  //给出SQL模板中对应的问号的值  Object[] objects = {cstm.getCid(),cstm.getCname(),cstm.getGender(),cstm.getBirthday()  ,cstm.getCellphone(),cstm.getEmail(),cstm.getDescription()};  //更新数据库  qr.update(insert,objects);  } **catch** (SQLException e) {  **throw** **new** RuntimeException(e);  }  } |
| **private** CustomerDao customerDao = **new** CustomerDao();  /\*\*  \* 添加客户  \*/  **public** **void** add(Customer cstm){  customerDao.add(cstm);  } |
| **private** CustomerService service = **new** CustomerService();  /\*\*  \* 添加客户  \* **@param** request  \* **@param** response  \* **@return**  \*/  **public** String add(HttpServletRequest request,HttpServletResponse response){  /\*  \* 1. 封装表单数据到Customer对象中  \* 1.5 给Customer中数据进行补全！补cid  \* 2. 把Customer对象传递给customerService#add()方法，完成添加业务  \* 3. 调用本类的findAll()方法完成显示所有客户  \*/  Customer cstm = CommonUtils.*toBean*(request.getParameterMap(), Customer.**class**);  cstm.setCid(CommonUtils.*uuid*());  service.add(cstm);  **return** findAll(request, response);  } |
| **package** com.customer.domain;  /\*\*  \* 实体类  \* **@author** Yorick  \*  \*/  **public** **class** Customer {  **private** String cid;  **private** String cname;  **private** String gender;  **private** String birthday;  **private** String cellphone;  **private** String email;  **private** String description;  **public** String getCid() {  **return** cid;  }  **public** **void** setCid(String cid) {  **this**.cid = cid;  }  **public** String getCname() {  **return** cname;  }  **public** **void** setCname(String cname) {  **this**.cname = cname;  }  **public** String getGender() {  **return** gender;  }  **public** **void** setGender(String gender) {  **this**.gender = gender;  }  **public** String getBirthday() {  **return** birthday;  }  **public** **void** setBirthday(String birthday) {  **this**.birthday = birthday;  }  **public** String getCellphone() {  **return** cellphone;  }  **public** **void** setCellphone(String cellphone) {  **this**.cellphone = cellphone;  }  **public** String getEmail() {  **return** email;  }  **public** **void** setEmail(String email) {  **this**.email = email;  }  **public** String getDescription() {  **return** description;  }  **public** **void** setDescription(String description) {  **this**.description = description;  }  @Override  **public** String toString() {  **return** "Customer [cid=" + cid + ", cname=" + cname + ", gender="  + gender + ", birthday=" + birthday + ", cellphone="  + cellphone + ", email=" + email + ", description="  + description + "]";  }  } |

## 查询客户

查询客户我们有两种方式:

全部查询不分页

全部查询分页

我们在真正做一个项目的时候,你永远的都不知道你的数据库中有几万条,还是几亿条数据,那么此时如果查询完不分页,你能想象到是什么结果吗?惨不忍睹,下面我们只给出不分页的代码,不在具体说明,重点来说明分页查询

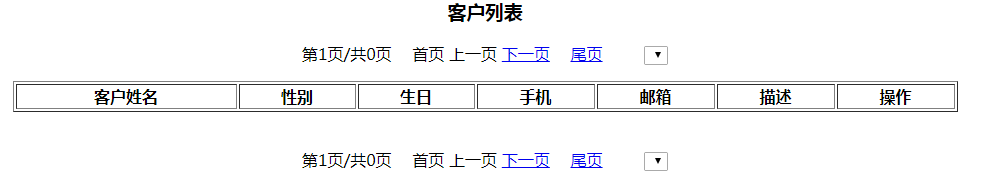
Top.jsp list.jsp

|  |
| --- |
| <body style="text-align: *center*;">  <h1>客户关系管理系统</h1>  <a href=*"*<c:url value=*'/add.jsp'*/>*"*>添加客户</a> 　|  <a href=*"*<c:url value=*'/CustomerServlet?method=findAll'*/>*"*>查询客户</a>　|  <a href=*"*<c:url value=*'/query.jsp'*/>*"*>高级搜索</a>  </body> |
| // public List<Customer> findAll(){  // try {  // //SQL模板  // String select = "select \* from customer";  // return qr.query(select, new BeanListHandler<Customer>(Customer.class));  // } catch (SQLException e) {  // throw new RuntimeException(e);  // }  // } |
| // public List<Customer> findAll(){  // return customerDao.findAll();  // } |
| // public String findAll(HttpServletRequest request,HttpServletResponse response){  // List<Customer> cstmList = service.findAll();  // request.setAttribute("cstmList", cstmList);  // return "f:/list.jsp";  // } |
| <table border=*"1"* width=*"70%"* align=*"center"*>  <tr>  <th>客户姓名</th>  <th>性别</th>  <th>生日</th>  <th>手机</th>  <th>邮箱</th>  <th>描述</th>  <th>操作</th>  </tr>  <c:forEach items="${requestScope.cstmList}" var="cstm">  <tr>  <td>${cstm.cname }</td>  <td>${cstm.gender }</td>  <td>${cstm.birthday }</td>  <td>${cstm.cellphone }</td>  <td>${cstm.email }</td>  <td>${cstm.description }</td>  <td>  <a href=*"*<c:url value='*/CustomerServlet?method=load&cid=*${cstm.cid }'/>*"*>编辑</a>  <a href=*"*<c:url value='*/CustomerServlet?method=delete&cid=*${cstm.cid }'/>*"*>删除</a>  </td>  </tr>  </c:forEach>  </table> |

### 分页查询

#### 分页页面数据分析:

我们常见的分页页面会有哪些数据呢,下面我们来个图片加以理解:



如上图我们看到分页页面需要这么几个数据:

页面需要什么数据：

* 当前页页码（currPageCode）：Servlet提供；

首页时为1,但我们要考虑我们后续在此页面操作时,此数据要和当前页数据联动的情况, 所以作为参数传递给DAO,查询当前页数据.

* 共几页（totalPage）：Servlet提供；
* 当前页数据（datas）：Servlet提供；

Servlet需要什么数据：

* 当前页页码（currPageCode）：页面提供，如果页面没有提供，那么默认为1；
* 总记录数（totalRecord）：通过数据库来查询；
* 每页记录数（pagesize）：系统数据；
* 共几页（totalPage）：通过totalRecord和pagesize来计算；
* 当前页第一行记录位置（currPageBeginIndex）：通过currPageCode和pagesize计算；
* 当前页数据（datas）：通过currPageBginIndex和pagesize查询数据库；

除此之外还有几个这个图上没有显示的,那就是位于上一页和下一页之间的页码

##### 页码列表



其中红框中的就是页码列表！

###### 页面需要的数据：

* 列表的开始页码（beginIndex）；
* 列表的结束页码（endIndex）。

例如开始页码为11，结束页码为18，那么就显示：



对于页面，它只需要beginIndex和endIndex，然后使用<c:forEach>就可以循环显示了！然后再判断一下遍历的数字如果与pageBean.currPageCode相等，那么就不要显示为链接即可。

###### PageBean

页面需要的数据由PageBean来提供，即为pageBean添加两个方法：

* int getBeginIndex()
* int getEndIndex()

但是，PageBean想计算这两个值，也要需要两个系统数据：

* pageCodeListSize：页码列表长度，下图中的列表长度为8，即最多显示8个页码；
* currPageCodeListIndex：当前页码在列表中的位置，下图中当前页码的位置为4，位置是从1开始计算。



###### 计算beginIndex

计算beginIndex分为4步：

1. 如果总页数（totalPage）小于列表长度（pageCodeListSize），那么beginIndex为1;

例如，当前数据一共5页，那么列表的开始页码一定是1。



1. 当上面条件不成立时，使用当前页码在列表中的位置（currPageCodeListIndex），以及当前页码（currPageCode）来推算出beginIndex。



上图中当前页码（currPageCode）为14，当前页码位置为4（currPageCodeListIndex），推算出beginIndex为11，即beginIndex = currPageCode- currPageCodeListIndex+1;

1. 第2步计算出的beginIndex如果小于1，即让beginIndex=1。

第2步的计算可能会出现问题，例如在当前页码（currPageCode）为1时，那么上面的计算就会出现beginIndex小于1的情况！这种beginIndex小于1的情况，只有在currPageCode为1、2、3时会出现。你可以去套用一下第2步的公式，会发现这个问题的！







也就是说，beginIndex的最小值就是1

1. 第2步计算出的beginIndex可能会导致列表长度不正确

当cuarrPageCode为30时，那么通过第2步计算出的beginIndex为27，但是如果totalPage为30呢？因为一共就30页，总不能显示31出来吧。那么显示：27、28、29、30，这就只有4个页码，但列表长度应该为8，所以出错。

我们为了验证第2步是否出现这个错误，需要通过得到的beginIndex来推算endIndex。例如currPageCode为30，计算出beginIndex为27，再通过beginIndex计算出endIndex为34。然后查看endIndex是否大于totalPage，如果大于了totalPage，那么应该使用totalPage减去pageCodeListSize再加1，得到正确的beginIndex。

int endIndex = beginIndex+pageCodeListSize-1;

if(endIndex > totalPage) beginIndex = totalPage-pageCodeListSize + 1;

|  |
| --- |
| **public** **int** getBeginIndex() {  **if**(totalPage <= pageCodeListSize) **return** 1;  **int** begin = currPageCode – currPageCodeListIndex+1;  **if**(begin < 1) begin = 1;  **int** end = begin + pageCodeListSize-1;  **if**(end > totalPage) begin = totalPage – pageCodeListSize+1;  **return** begin;  } |

###### 计算endIndex

计算endIndex也是分为4步

1. 如果totalPage小于pageCodeListSize，那么endIndex为totalPage
2. 通过pageCodeListSize、currPageCode、currPageCodeListIndex来推出endIndex。



currPageCode=14，pageCodeListSize=8，currPageCodeListIndex=4，所以

endIndex = currPageCode+(pageCodeListSize=currPagecodeListIndex)=18。

1. 第2步的计算结果可能会大于totalPage，那么就重置endIndex为totalPage

如果currPageCode为30，那么通过第2步计算出的endIndex为34，但是如果totalPage为30呢？那么就让endIndex=totalPage。









当currPageCode为27、28、29、30时，如果totalPage=30，那么第2步都会计算出错。这时就把endIndex=totalPage

1. 第2步的计算结果可能会导致错误的列表长度

当currPageCode为1时，那么通过第2步计算的endIndex为5，即列表为1、2、3、4、5，但列表长度应该为8，所以当第2步计算出的endIndex< pageCodeListSize，那么让endIndex等于列表长度。

|  |
| --- |
| **public** **int** getEndIndex() {  **if**(totalPage <= pageCodeListSize) **return** totalPage;  **int** end = currPageCode + (pageCodeListSize-currPageCodeListIndex);  **if**(end > totalPage) end = totalPage;  **if**(end < pageCodeListSize) end = pageCodeListSize;  **return** end;  } |

|  |
| --- |
| **package** com.customer.domain;  **import** java.util.List;  **public** **class** PageBean<T> {  **private** **int** cp; // 当前页码  **private** **int** tr; // 总记录数  **private** **int** ps; // 每页记录数  **private** List<T> beans;// 当前页的数据  **private** **int** totalPage;//总页数  **private** **int** beginIndex;//页码列表的开始页码  **private** **int** endIndex;//页码列表的结束页码  **private** **int** pageCodeListSize = 8;//页码列表长度  **private** **int** currPageCodeListIndex = 4;//当前页码在页码列表中的位置  **private** String url;//保留：contextPath + servletPath + queryString  **public** **int** getTotalPage() {  totalPage = tr / ps;  **return** tr%ps==0 ? totalPage : totalPage + 1;  }  **public** **void** setTotalPage(**int** totalPage) {  **this**.totalPage = totalPage;  }  **public** String getUrl() {  **return** url;  }  **public** **void** setUrl(String url) {  **this**.url = url;  }  **public** **int** getCp() {  **return** cp;  }  **public** **void** setCp(**int** cp) {  **this**.cp = cp;  }  **public** **int** getTr() {  **return** tr;  }  **public** **void** setTr(**int** tr) {  **this**.tr = tr;  }  **public** **int** getPs() {  **return** ps;  }  **public** **void** setPs(**int** ps) {  **this**.ps = ps;  }  **public** List<T> getBeans() {  **return** beans;  }  **public** **void** setBeans(List<T> beans) {  **this**.beans = beans;  }  //计算开始页面  **public** **int** getBeginIndex() {  **if**(totalPage<=pageCodeListSize) **return** 1;  **int** begin = cp - currPageCodeListIndex+1;  **if**(begin<1) begin = 1;  **int** end = begin+pageCodeListSize-1;  **if**(end > totalPage){  begin = totalPage - pageCodeListSize + 1;  }  **return** begin;  }  **public** **int** getEndIndex() {  **if**(totalPage<pageCodeListSize) **return** totalPage;  **int** end = cp + (pageCodeListSize - currPageCodeListIndex);  **if**(end > totalPage){  end = totalPage;  }  **if**(end < pageCodeListSize){  end = pageCodeListSize;  }  **return** end;  }  } |
| **public** PageBean<Customer> findAll(**int** cp,**int** ps){  **try** {  //得到所有记录的数目  String sql = "select count(\*) from customer";  Number num = (Number) qr.query(sql, **new** ScalarHandler());  **int** tr = num.intValue();  //得到当前页面数据  sql = "select \* from customer limit ?,?";  //第一页,显示第1到第10条数据,第二页是11到20,(pagenumber-1)\*10  Object[] objects = {(cp-1)\*ps,ps};  List<Customer> beans = qr.query(sql, **new** BeanListHandler<Customer>(Customer.**class**),objects);  /\*  \* 3. 使用tr、beans、cp、ps创建PageBean，返回！  \*/  PageBean<Customer> pb = **new** PageBean<>();  pb.setBeans(beans);  pb.setCp(cp);  pb.setTr(tr);  pb.setPs(ps);  **return** pb;  } **catch** (SQLException e) {  **throw** **new** RuntimeException(e);  }  } |
| **public** PageBean<Customer> findAll(**int** cp,**int** ps){  **return** customerDao.findAll(cp,ps);  } |
| **public** String findAll(HttpServletRequest request,HttpServletResponse response){  // 1. 得到cp  **int** cp = 1;//默认为1  String param = request.getParameter("cp");  **if**(param != **null** && !param.trim().isEmpty()) {  cp = Integer.*parseInt*(param);  }  //2. 设置ps  **int** ps = 10;  PageBean<Customer> pb = service.findAll(cp,ps);  //给pb的url赋值  //url = contextPath + servletpath + queryString  String url = request.getRequestURI() + "?" + request.getQueryString();  **int** index = url.lastIndexOf("&cp=");  **if**(index != -1) {  url = url.substring(0, index);  }  pb.setUrl(url);  //4. 保存到request域中，转发到list.jsp  request.setAttribute("pb", pb);  **return** "f:/list.jsp";  } |
| <title>客户列表</title>  ..................  <script type=*"text/javascript"*>  window.onload(){  **var** selectNodeList = document.getElementsByName("cp");  selectNodeList[0].onchange = **function**(){  //得到selectEle上的值  **var** cp = selectNodeList[0].value;  /\*  2. 向http://localhost:8080/cstms/CustomerServlet?method=findAll&cp=  location.href 属性返回当前页面的 URL。  \*/  location.href = "${pb.url}&cp=" + cp;  };  selectNodeList[1].onchange = **function**(){  **var** cp = selectNodeList[1].value;  location.href = "http://localhost:8080/customer/CustomerServlet?method=findAll&cp=" + cp;  };  }  </script>  .......  <h3 align=*"center"*>客户列表</h3>  <%@include file=*"/pb.jsp"* %>  <table border=*"1"* width=*"70%"* align=*"center"*>  <tr>  <th>客户姓名</th>  <th>性别</th>  <th>生日</th>  <th>手机</th>  <th>邮箱</th>  <th>描述</th>  <th>操作</th>  </tr>  <c:forEach items=*"*${pb.beans}*"* var=*"cstm"*>  <tr>  <td>${cstm.cname }</td>  <td>${cstm.gender }</td>  <td>${cstm.birthday }</td>  <td>${cstm.cellphone }</td>  <td>${cstm.email }</td>  <td>${cstm.description }</td>  <td>  <a href=*"*<c:url value='*/CustomerServlet?method=load&cid=*${cstm.cid }'/>*"*>编辑</a>  <a href=*"*<c:url value='*/CustomerServlet?method=delete&cid=*${cstm.cid }'/>*"*>删除</a>  </td>  </tr>  </c:forEach>  </table>  </br>  <!--分页操作,单独放在一个jsp上 -->  <%@include file=*"/pb.jsp"* %> |
| 第${pb.cp }页/共${pb.totalPage }页  <c:choose>  <c:when test=*"*${pb.cp **eq** 1 }*"*>首页</c:when>  <c:otherwise><a href=*"*${pb.url}*&cp=1"*>首页</a></c:otherwise>  </c:choose>  <c:choose>  <c:when test=*"*${pb.cp **eq** 1 }*"*>上一页</c:when>  <c:otherwise><a href=*"*${pb.url}*&cp=*${pb.cp-1 }*"*>上一页</a>　</c:otherwise>  </c:choose>  <!--处理页码下拉菜单 -->  <select name=*"cp"*>  <c:forEach begin=*"1"* end=*"*${pb.totalPage }*"* var=*"i"*>  <option value=*"*${i }*"* <c:if test="${i **eq** pb.cp }">selected="selected" </c:if>>  ${i }  </option>  </c:forEach>  </select>  <%-----页面列表 --%>  <%--  > 总页数<=10：begin=1, end=tp  > begin=cp-4, end=cp+5  > 头溢出：if(begin < 1) begin=1, end=10  > 尾溢出：if(end > tp) end=tp, begin=tp-9  --%>  <%-----页面列表 --%>  <c:choose>  <c:when test=*"*${pb.totalPage <= 10 }*"*><%--总页数<=10：begin=1, end=tp --%>  <c:set var=*"begin"* value=*"1"*/>  <c:set var=*"end"* value=*"*${pb.totalPage }*"*/>  </c:when>  <c:otherwise>  <%-- begin=cp-4, end=cp+5 --%>  <c:set var=*"begin"* value=*"*${pb.cp-4 }*"*/>  <c:set var=*"end"* value=*"*${pb.cp+5 }*"*/>  <c:if test=*"*${begin < 1 }*"*><%--头溢出：if(begin < 1) begin=1, end=10 --%>  <c:set var=*"begin"* value=*"1"*/>  <c:set var=*"end"* value=*"10"*/>  </c:if>  <c:if test=*"*${end > pb.totalPage }*"*><%-- 尾溢出：if(end > tp) end=tp, begin=tp-9 --%>  <c:set var=*"begin"* value=*"*${pb.totalPage-9 }*"*/>  <c:set var=*"end"* value=*"*${pb.totalPage }*"*/>  </c:if>  </c:otherwise>  </c:choose>  <c:forEach var=*"i"* begin=*"*${begin }*"* end=*"*${end }*"*>  <c:choose>  <c:when test=*"*${i **eq** pb.cp }*"*>[${i}]</c:when>  <c:otherwise>  <a href=*"*${pb.url}*&cp=*${i}*"*>[${i}]</a>  </c:otherwise>  </c:choose>  </c:forEach>  <c:choose>  <c:when test=*"*${pb.cp **eq** pb.totalPage }*"*>下一页</c:when>  <c:otherwise><a href=*"*${pb.url}*&cp=*${pb.cp+1 }*"*>下一页</a>　</c:otherwise>  </c:choose>  <c:choose>  <c:when test=*"*${pb.cp **eq** pb.totalPage }*"*>尾页</c:when>  <c:otherwise><a href=*"*${pb.url}*&cp=*${pb.totalPage}*"*>尾页</a>　</c:otherwise>  </c:choose>  <!--处理页码下拉菜单 -->  <select name=*"cp"*>  <c:forEach begin=*"1"* end=*"*${pb.totalPage }*"* var=*"i"*>  <option value=*"*${i }*"* <c:if test="${i **eq** pb.cp }">selected="selected" </c:if>>  ${i }  </option>  </c:forEach>  </select>  </p> |

## 编辑客户

在List.jsp页面,我们看到可以对客户操作的操作



点击编辑跳转到edit.jsp页面,但是我们要完成数据的加载擦欧洲哦,加载要修改的用户的信息,修改信息,然后edit页面捕获信息,将数据库信息修改,条件就是cid,

|  |
| --- |
| <body>  <h3 align=*"center"*>编辑客户</h3>  <form action=*"*<c:url value=*'/CustomerServlet?method=edit'*/>*"* method=*"post"*>  <input type=*"hidden"* name=*"cid"* value=*"*${cstm.cid }*"*>  <table border=*"0"* align=*"center"* width=*"40%"* style="margin-left: *100px*;">  <tr>  <td width=*"100px"*>客户名称</td>  <td width=*"40%"*>  <input type=*"text"* name=*"cname"* value=*"*${cstm.cname }*"*/>  </td>  <td align=*"left"*>  <label id=*"cnameError"* class=*"error"*>&nbsp;</label>  </td>  </tr>  <tr>  <td>客户性别</td>  <td>  <input type=*"radio"* name=*"gender"* value=*"男"* id=*"male"* ${cstm.gender **eq**'男'?'checked="checked"':"" }/>  <label for=*"male"*>男</label>  <input type=*"radio"* name=*"gender"* value=*"女"* id=*"female"* ${cstm.gender **eq**'女'?'checked="checked"':"" }/>  <label for=*"female"*>女</label>  </td>  <td>  <label id=*"genderError"* class=*"error"*>&nbsp;</label>  </td>  </tr>  <tr>  <td>客户生日</td>  <td>  <input type=*"text"* name=*"birthday"* id=*"birthday"* readonly=*"readonly"* value=*"*${cstm.birthday }*"*/>  </td>  <td>  <label id=*"birthdayError"* class=*"error"*>&nbsp;</label>  </td>  </tr>  <tr>  <td>手机</td>  <td>  <input type=*"text"* name=*"cellphone"* value=*"*${cstm.cellphone }*"*/>  </td>  <td>  <label id=*"cellphoneError"* class=*"error"*>&nbsp;</label>  </td>  </tr>  <tr>  <td>邮箱</td>  <td>  <input type=*"text"* name=*"email"* value=*"*${cstm.email }*"*/>  </td>  <td>  <label id=*"emailError"* class=*"error"*>&nbsp;</label>  </td>  </tr>  <tr>  <td>描述</td>  <td>  <textarea rows=*"5"* cols=*"30"* name=*"description"*>${cstm.description }</textarea>  </td>  <td>  <label id=*"descriptionError"* class=*"error"*>&nbsp;</label>  </td>  </tr>  <tr>  <td>&nbsp;</td>  <td>  <input type=*"button"* value=*"编辑客户"* onclick="add()"/>  <input type=*"reset"* value=*"重置"*/>  </td> |
| /\*\*  \* 按cid查询客户  \*/  **public** Customer load(String cid){  **try** {  //SQL模板  String select = "select \* from customer where cid=?";  **return** qr.query(select, **new** BeanHandler<Customer>(Customer.**class**),cid);  } **catch** (SQLException e) {  **throw** **new** RuntimeException(e);  }  } |
| **public** Customer load(String cid){  **return** customerDao.load(cid);  } |
| **public** String load(HttpServletRequest request,HttpServletResponse response){  request.setAttribute("cstm",service.load(request.getParameter("cid")));  **return** "f:/edit.jsp";  } |
| /\*\*  \* 修改用户  \*/  **public** **void** edit(Customer cstm){  **try** {  String update = "update customer set cname=?,gender=?,birthday=?,cellphone=?" +  ", email=?, description=? where cid=?";  Object[] params = {cstm.getCname(), cstm.getGender(),  cstm.getBirthday(), cstm.getCellphone(), cstm.getEmail(),  cstm.getDescription(), cstm.getCid()};  qr.update(update, params);  } **catch** (Exception e) {  **throw** **new** RuntimeException(e);  }  } |
| **public** **void** edit(Customer cstm){  customerDao.edit(cstm);  } |
| **public** String edit(HttpServletRequest request,HttpServletResponse response){  //封装表单数据到JavaBean  Customer cstm = CommonUtils.*toBean*(request.getParameterMap(), Customer.**class**);  service.edit(cstm);  **return** findAll(request, response);  } |

## 删除客户

|  |
| --- |
| **public** **void** delete(String cid){  **try** {  String delete = "delete from customer where cid=?";  qr.update(delete, cid);  } **catch** (Exception e) {  **throw** **new** RuntimeException(e);  }  } |
| **public** **void** delete(String cid){  customerDao.delete(cid);  } |
| **public** String delete(HttpServletRequest request,HttpServletResponse response){  service.delete(request.getParameter("cid"));  **return** findAll(request, response);  } |

## 条件查询

条件查询对比SQL语句是要有like等字样的,但是条件查询存在着条件存在和不存在的情况,所以我们的SQL语句也是不确定的,那么此时应该怎么办呢?这里就引入了**组合SQL语句,**意思就是根据传进的参数来判断语句的结果,语句不停的组装,知道组装成一个合格的SQL语句,但是此时我们要注意,SQL语句的参数问题,我们传进来的参数可能不是同一类型,那么用来装参数的容器就只能是List了,我们把参数动态的组装SQL语句,同时把参数装进List中,最后查找出我们要的结果;

Query.jsp

|  |
| --- |
| <h3 align=*"center"*>高级搜索</h3>  <form action=*"*<c:url value=*'/CustomerServlet'*/>*"* method=*"get"*>  <input type=*"hidden"* name=*"method"* value=*"query"*/> |
| /\*\*  \* 多条件组合查询  \*/  **public** PageBean<Customer> query(**int** cp,**int** ps,Customer cstm){  **try** {  /\*  \* 1. 拼凑sql模板  \* 2. 给出?对应的值  \* 3. 调用qr的query方法，使用BeanListHandler  \*/  //拼凑SQL模板  //创建List存储?  List<Object> params = **new** ArrayList<Object>();  StringBuilder sb= **new** StringBuilder("from customer where 1=1");  // 判断条件中是否包含cname，如果包含添加条件子句！  String cname = cstm.getCname();  **if**(cname!=**null** && !cname.trim().isEmpty()){  sb.append(" and cname like ?");  params.add("%"+cname+"%");//当这个问号存在时，添加对应的值到params中  }  // 判断条件中是否包含gender，如果包含添加条件子句！  String gender = cstm.getGender();  **if**(gender!=**null** && !gender.trim().isEmpty()){  sb.append("and gender=?");  params.add(gender);  }  // 判断条件中是否包含cellphone，如果包含添加条件子句！  String cellphone = cstm.getCellphone();  **if**(cellphone != **null** && !cellphone.trim().isEmpty()){  sb.append("and cellphone like ?");  params.add("%"+cellphone+"%");  }  // 判断条件中是否包含email，如果包含添加条件子句！  String email = cstm.getEmail();  **if**(email!=**null** && !email.trim().isEmpty()){  sb.append("and email like ?");  params.add("%"+email+"%");  }  // 拼凑count语句，用来得到总记录数  StringBuilder cnt = **new** StringBuilder("select count(\*)").append(sb);  Number num = (Number)qr.query(cnt.toString(), **new** ScalarHandler(), params.toArray());  **int** tr = num.intValue();  // 拼凑查询当前页记录的SQL语句  StringBuilder sql = **new** StringBuilder("select \*").append(sb);  sql.append(" limit ?,?");  // 向参数中添加两个值：  params.add((cp-1)\*ps);  params.add(ps);  // 查询  List<Customer> list = qr.query(sql.toString(), **new** BeanListHandler<Customer>(Customer.**class**), params.toArray());  /\*  \* 创建PageBean  \*/  PageBean<Customer> pb = **new** PageBean<Customer>();  pb.setCp(cp);  pb.setPs(ps);  pb.setTr(tr);  pb.setBeans(list);  **return** pb;  } **catch** (SQLException e) {  **throw** **new** RuntimeException(e);  }  } |
| //条件查询  **public** PageBean<Customer> query(**int** cp,**int** ps,Customer cstm){  **return** customerDao.query(cp, ps, cstm);  } |
| /\*\*  \* 条件查询  \* **@throws** UnsupportedEncodingException  \*/  **public** String query(HttpServletRequest request,HttpServletResponse response) **throws** IOException{  /\*  \* 封装表单参数到Customer对象中  \* 表单中的数据就是查询条件  \*/  /\*  \* 处理GET请求编码问题  \* 1. 表单为GET  \* 2. 超链接也为GET  \*/  Customer cstm = CommonUtils.*toBean*(request.getParameterMap(), Customer.**class**);  **if**(cstm.getCname()!=**null**&&!cstm.getCname().trim().isEmpty())  cstm.setCname(**new** String(cstm.getCname().getBytes("ISO-8859-1"), "UTF-8"));  **if**(cstm.getGender()!=**null**&&!cstm.getGender().trim().isEmpty())  cstm.setGender(**new** String(cstm.getGender().getBytes("ISO-8859-1"), "UTF-8"));  **if**(cstm.getCellphone()!=**null**&&!cstm.getCellphone().trim().isEmpty())  cstm.setCellphone(**new** String(cstm.getCellphone().getBytes("ISO-8859-1"), "UTF-8"));  **if**(cstm.getEmail()!=**null**&&!cstm.getEmail().trim().isEmpty())  cstm.setEmail(**new** String(cstm.getEmail().getBytes("ISO-8859-1"), "UTF-8"));  //得到cp  **int** cp = 1;  String param = request.getParameter("cp");  **if**(param!=**null** && !param.trim().isEmpty()){  cp = Integer.*parseInt*(param);  }  //设置ps  **int** ps = 10;  //使用cp和ps调用service方法，得到PageBean  PageBean<Customer> pb = service.query(cp, ps, cstm);  /\*  \* 给pb的url赋值  \* url = contextPath + servletpath + queryString  \*/  String url = request.getRequestURI() + "?" + request.getQueryString();  **int** index = url.lastIndexOf("&cp=");  **if**(index != -1) {  url = url.substring(0, index);  }    pb.setUrl(url);  request.setAttribute("pb",pb);  **return** "f:/list.jsp";  } |

在这里我们要注意一个问题,那就是通常提交方式我们用到的有两种get和post,post用来提交表单,涉及到的是数据库中的改变,get则是用于查找,但是get涉及到编码的问题,get的编码是URL---iso-8859-1,而我们通常使用的是utf-8,所以此时就要进行一次编码的设置,但要注意了,如果我们对参数不加判断就设置编码是会出现错误的,所以我们加以判断.