**进销存整体业务**

**商品采购业务**

0 前提条件（可以先补充数据）

商品信息

供应商信息

采购表

1. 采购入库

确定采购的商品 - 填写采购数量-采购价格-供应商 ——添加到采购表

—— 修改商品库存

1.1查询商品（借用商品管理业务）



ProductDao extends BaseDao

ProductBiz 业务接口

|  |
| --- |
| //7模糊查询  public List<Product> findByCondition(String conditions); |

ProductBizImpl 业务实现类

|  |
| --- |
| public List<Product> findByCondition(String conditions) {  String sql = "select \* from product2 where state=1 ";  if(conditions !=null && conditions.length()>0){  sql +=" and concat(pname,ptype) like '%"+conditions+"%'";  }  return pdao.query(sql, Product.class);  } |

查询按钮方法：

|  |
| --- |
| //1 获得条件  String conditions = this.txtConditions.getText().trim();  //2 通过查询业务获得商品集合  List<Product> list = pbiz.findByCondition(conditions);  //3 显示在表格  showOnProductTable(list); |

showOnProductTable()方法：

|  |
| --- |
| private void showOnProductTable(List<Product> list) {  //1 获得表格模型  DefaultTableModel dtm = (DefaultTableModel) this.tblProduct.getModel();  //2 清空表格  while(dtm.getRowCount()>0){  dtm.removeRow(0);  }  // 3添加到表格  for(Product p: list){  Vector vt = new Vector();  vt.add(p.getPid());  vt.add(p.getPname());  vt.add(p.getPtype());  vt.add(p.getCurrentamount());  vt.add(p.getSuggestpurchaseprice());  dtm.addRow(vt);  }  } |

1.2 选择商品，将选择的商品添加到需要采购的Table上

1.2.1 给表格添加鼠标单击事件

|  |
| --- |
| // 将选择的商品添加到需要采购的table上  int selectrow = this.tblProduct.getSelectedRow();  Integer pid = (Integer) this.tblProduct.getValueAt(selectrow, 0);  String pname = (String) this.tblProduct.getValueAt(selectrow, 1);  String ptype = (String) this.tblProduct.getValueAt(selectrow, 2);  Integer currentamount = (Integer) this.tblProduct.getValueAt(selectrow, 3);  BigDecimal suggestpurchaseprice = (BigDecimal) this.tblProduct.getValueAt(selectrow, 4);  Vector vt = new Vector();  vt.add(pid); // vt.get(0)  vt.add(pname);  vt.add(ptype);  vt.add(suggestpurchaseprice);  //添加到采购表格  addPurchaseTable(vt); |

1.2.2 添加到采购表addPurchaseTable(vt):

|  |
| --- |
| private void addPurchaseTable(Vector vt) {  boolean flag = false;  DefaultTableModel dtm = (DefaultTableModel) this.tblPurchase.getModel();  for(int i=0;i<dtm.getRowCount();i++){  if(vt.get(0) == tblPurchase.getValueAt(i, 0)){  flag = true;  break;  }  }  if(flag == false){  dtm.addRow(vt);  }  } |

1.3 初始化供应商

供应商的信息需要全部查询出来，以组合框的形式存放到采购信息表中

1.3.1 Supplier实体类添加toString方法，目的是在显示对象的时候只显示供应商名称

|  |
| --- |
| public String toString(){  return this.supname;  } |

1.3.2 SupplierDao extends BaseDao

1.3.3 SupplierBiz中添加findAll方法

|  |
| --- |
| //查询所有供应商  public List<Supplier> findAll(); |

1.3.4 SupplierBizImpl实现SupplierBiz接口

|  |
| --- |
| //引入dao  SupplierDao supdao = new SupplierDao();  @Override  public List<Supplier> findAll() {  String sql = "select \* from supplier where state = 1";  return supdao.query(sql, Supplier.class);  } |

1.3.5 PurchaseInFrame添加初始化供应商方法

|  |
| --- |
| public void initPurchaseTable(){  JComboBox cobsup = new JComboBox();  List<Supplier> suplist = supbiz.findAll();  for(Supplier sup: suplist){  cobsup.addItem(sup);  }  this.tblPurchase.getColumnModel().getColumn(6).setCellEditor(new DefaultCellEditor(cobsup));  } |

1.3.6 PurchaseInFrame构造方法调用initPurchaseTable()

1.4 采购信息表添加鼠标单击事件，获得所选择的行号

PurchaseInFrame类 添加属性 int selectedRow = -1; //供删除用

事件方法：selectedRow = this.tblPurchase.getSelectedRow();

1.5 “删除”按钮添加事件，选择要采购的商品进行删除操作

|  |
| --- |
| //删除需要采购的信息  if(selectedRow == -1){  JOptionPane.showMessageDialog(this, "请选择要删除的采购信息");  return ;  }  //tblPurchase :采购信息表变量名  DefaultTableModel dtm = (DefaultTableModel) this.tblPurchase.getModel();  dtm.removeRow(selectedRow);  selectedRow = -1; //还原选择行号 |

1.6 重构BaseDao增加批量更新方法，抛出异常是为了在业务层进行捕获

|  |
| --- |
| /\*\*  \* 批量更新  \* @param conn 连接对象  \* @param sql sql语句  \* @param params 批量更新参数  \* @return  \*/  public boolean batchUpdate(Connection conn,String sql,Object[][] params) throws SQLException{  QueryRunner qRunner = new QueryRunner();  int result = 0;  boolean flag = false;  result = qRunner.batch(conn, sql, params).length;  if(result > 0){  flag = true;  }  return flag;  } |

1.7 PurchaseBiz业务接口方法

|  |
| --- |
| public interface PurchaseBiz {  //采购入库  public boolean purchasein(Object[][] purchases,Object[][] stocks);    //采购退货  public boolean purchasereturn(Object[][] purchasereturn,Object[][] stocks);  } |

1.8 PurchaseBizImpl实现类

|  |
| --- |
| public class PurchaseBizImpl implements PurchaseBiz{  ProductDao prodao = new ProductDao();  PurchaseDao purdao = new PurchaseDao();  @Override  public boolean purchasein(Object[][] purchases,Object[][]stocks) {  boolean result = true;  String sql1 = "insert into purchase(purno,pid,purchaseamount,purchaseprice,purtotals,supplierid,purchasedate) values(?,?,?,?,?,?,?)";  String sql2 = "update product2 set currentamount=currentamount+? where pid=?";//库存增加  Connection conn = new BaseDao().getConnection();//获得连接    try {  conn.setAutoCommit(false);//设置不自动提交  purdao.batchUpdate(conn, sql1, purchases);//批量添加采购表  prodao.batchUpdate(conn, sql2, stocks);//更新商品库存  conn.commit(); //提交事务  } catch (Exception e) {  try {  result = false;  conn.rollback(); //回滚事务  } catch (SQLException ex) {  ex.printStackTrace();  }  e.printStackTrace();  }  return result;  }  @Override  public boolean purchasereturn(Object[][] purchasereturn,Object[][] stocks) {  return false;  } |

1.9 采购入库按钮事件

|  |
| --- |
| //获取信息  DefaultTableModel dtm = (DefaultTableModel) this.tblPurchase.getModel();  Object[][] purchase = new Object[dtm.getRowCount()][7];//根据具体数值数创建采购数组  Object[][] stocks = new Object[dtm.getRowCount()][2];//商品库存数组  //遍历表格  for(int i=0;i<dtm.getRowCount();i++){  purchase[i][0] = NoUtil.createPurno();//采购单号  purchase[i][1] = this.tblPurchase.getValueAt(i, 0);//商品编号  stocks[i][1] = this.tblPurchase.getValueAt(i, 0);  purchase[i][2] = this.tblPurchase.getValueAt(i, 4);//采购数量  stocks[i][0] = this.tblPurchase.getValueAt(i, 4);  purchase[i][3] = this.tblPurchase.getValueAt(i, 5);//采购价格  //总金额，价格 \* 数量  purchase[i][4] = new BigDecimal(purchase[i][3].toString()).multiply(new BigDecimal(purchase[i][2].toString()));  Supplier sup = (Supplier)this.tblPurchase.getValueAt(i, 6);  purchase[i][5] = sup.getSupplierid();  purchase[i][6] = new java.util.Date();  }  //调用批量添加  boolean result = purbiz.purchasein(purchase, stocks);  if(result == true){  JOptionPane.showMessageDialog(this, "采购入库成功");  //更新商品表  btnQueryActionPerformed(evt);//调用查询方法  }else{  JOptionPane.showMessageDialog(this, "采购入库失败");  }  //清空采购表  while(dtm.getRowCount()>0){  dtm.removeRow(0);  } |

2.采购退货

确定采购单 - 填写退货数量 —— 添加到采购退货表

—— 修改商品库存

2.1 查询采购单

2.1.1 创建视图：

|  |
| --- |
| Create view view\_purchase  as  SELECT  purchase.purid,  purchase.purno,  purchase.pid,  purchase.purchaseamount,  purchase.purchaseprice,  purchase.purtotals,  purchase.supplierid,  purchase.purchasedate,  product2.pname,  supplier.supname  FROM  purchase ,  product2 ,  supplier  WHERE  purchase.pid = product2.pid AND  purchase.supplierid = supplier.supplierid |

2.1.2 创建实体类ViewPurchase ,对应视图

|  |
| --- |
| public class ViewPurchase {  private Integer purid;  private String purno;  private Integer pid;  private Integer purchaseamount;  private BigDecimal purchaseprice;  private Integer supplierid;  private Date purchasedate;  private Integer state;  private String pname;  private String supname;  //set get方法  } |

2.1.3 PurchaseBiz增加业务接口

// 查询采购单

public List<ViewPurchase> findByCondition(String start,String end,String conditions);

2.1.4 PurchaseBizImpl实现方法

|  |
| --- |
| public List<ViewPurchase> findByCondition(String start, String end, String conditions) {  String sql = "select \* from view\_purchase where purchasedate between '"+start+"' and '"+end+"'";  if(conditions != null && conditions.length()>0){  sql +=" and concat(purno,pname,supname) like '%"+conditions+"%'";  }  return purdao.query(sql, ViewPurchase.class);  } |

2.1.5 PurchaseOutFrame 中给“查询”按钮增加事件

|  |
| --- |
| //按条件查询采购单  //1获取条件  String start = this.dateStart.getText();  String end = this.dateEnd.getText();  String conditions = this.txtConditions.getText().trim();  //调用业务  List<ViewPurchase> list = pbiz.findByCondition(start, end, conditions);  //显示在表格中  showOnTable(list); |

2.1.6 showOnTable(list)方法：

|  |
| --- |
| private void showOnTable(List<ViewPurchase> list) {  //1 表格模型  DefaultTableModel dtm = (DefaultTableModel) this.tblPurchase.getModel();  //2清空表格  while(dtm.getRowCount()>0){  dtm.removeRow(0);  }  //3 添加数据  for(ViewPurchase pur:list){  Vector vt = new Vector();  vt.add(pur.getPurno());  vt.add(pur.getPid());  vt.add(pur.getPname());  vt.add(pur.getPurchaseamount());  vt.add(pur.getPurchaseprice());  vt.add(pur.getSupplierid());  vt.add(pur.getSupname());  vt.add(pur.getPurchasedate());  dtm.addRow(vt);    }  } |

2.1.7 JTable隐藏列方法

|  |
| --- |
| public void HiddenCell(JTable table, int column) {  TableColumn tc = table.getTableHeader().getColumnModel().getColumn(  column);  tc.setMaxWidth(0);  tc.setPreferredWidth(0);  tc.setWidth(0);  tc.setMinWidth(0);  table.getTableHeader().getColumnModel().getColumn(column)  .setMaxWidth(0);  table.getTableHeader().getColumnModel().getColumn(column)  .setMinWidth(0);  } |

2.1.8 构造方法中调用

HiddenCell(this.tblPurchase,1); //隐藏第1列

HiddenCell(this.tblPurchase,5); //隐藏第5列