재고관리 및
판매관리 프로그램

백종원

목차

개발환경

구성

Eclipse IDE

프로그램 개발

Oracle SQL Developer

데이터베이스 설정

개발 목적

프로그램 구상

재고 관리

창고업무에서 물건들이 들어오고 나가는 것을 기록하는 과정을 구현

구매 관리

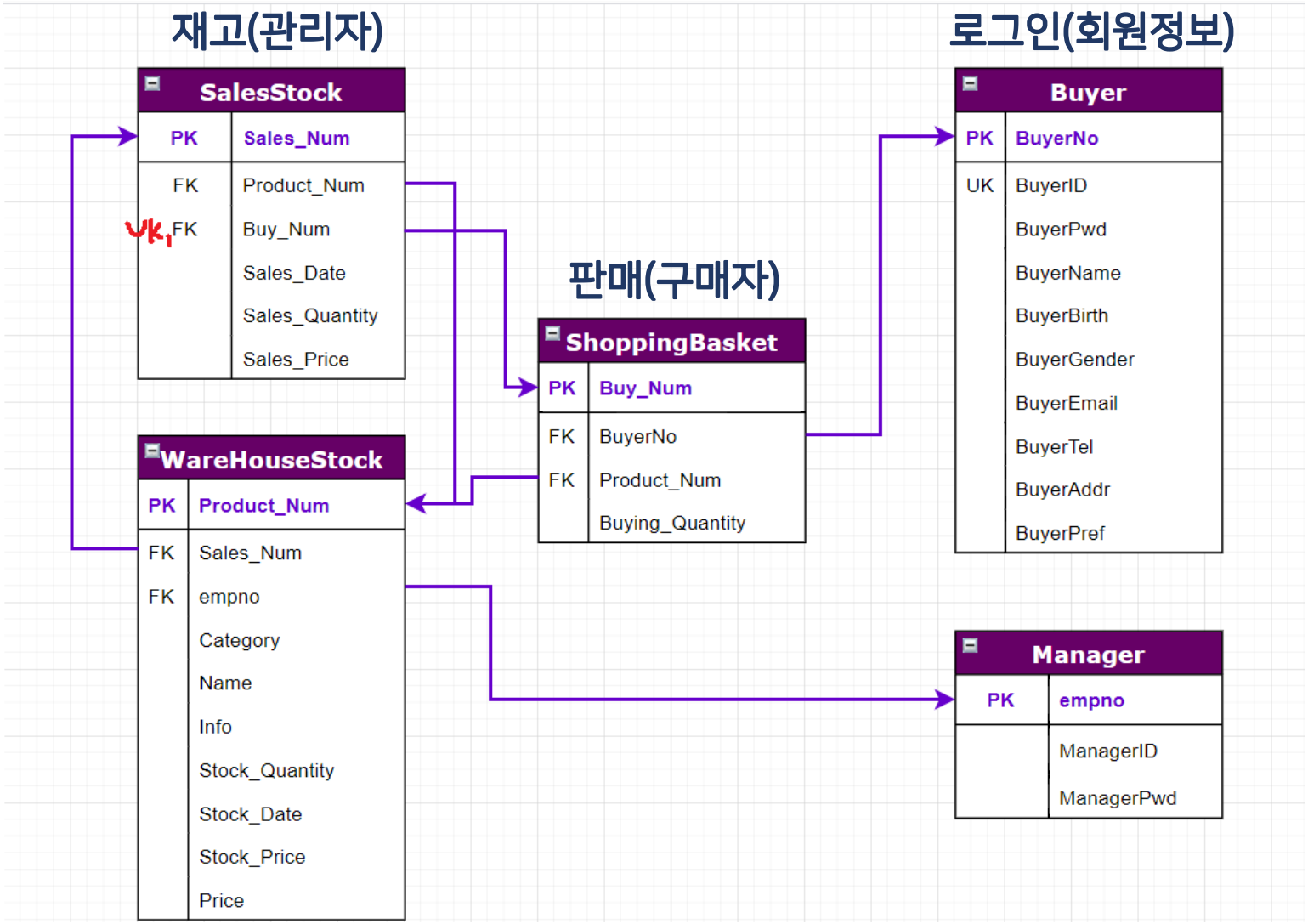
물건이 입고 되는 것과 출고 되는 것은 모두 재고관리 프로그램만으로 표현할 수 있었으나,
물건이 출고되는 이유인 판매과정을 구현하고자 함

로그인 관리

재고관리를 어떤 사람이 하고, 물건을 어떤 사람이 샀는지 구분하기 위해 구현

관계도

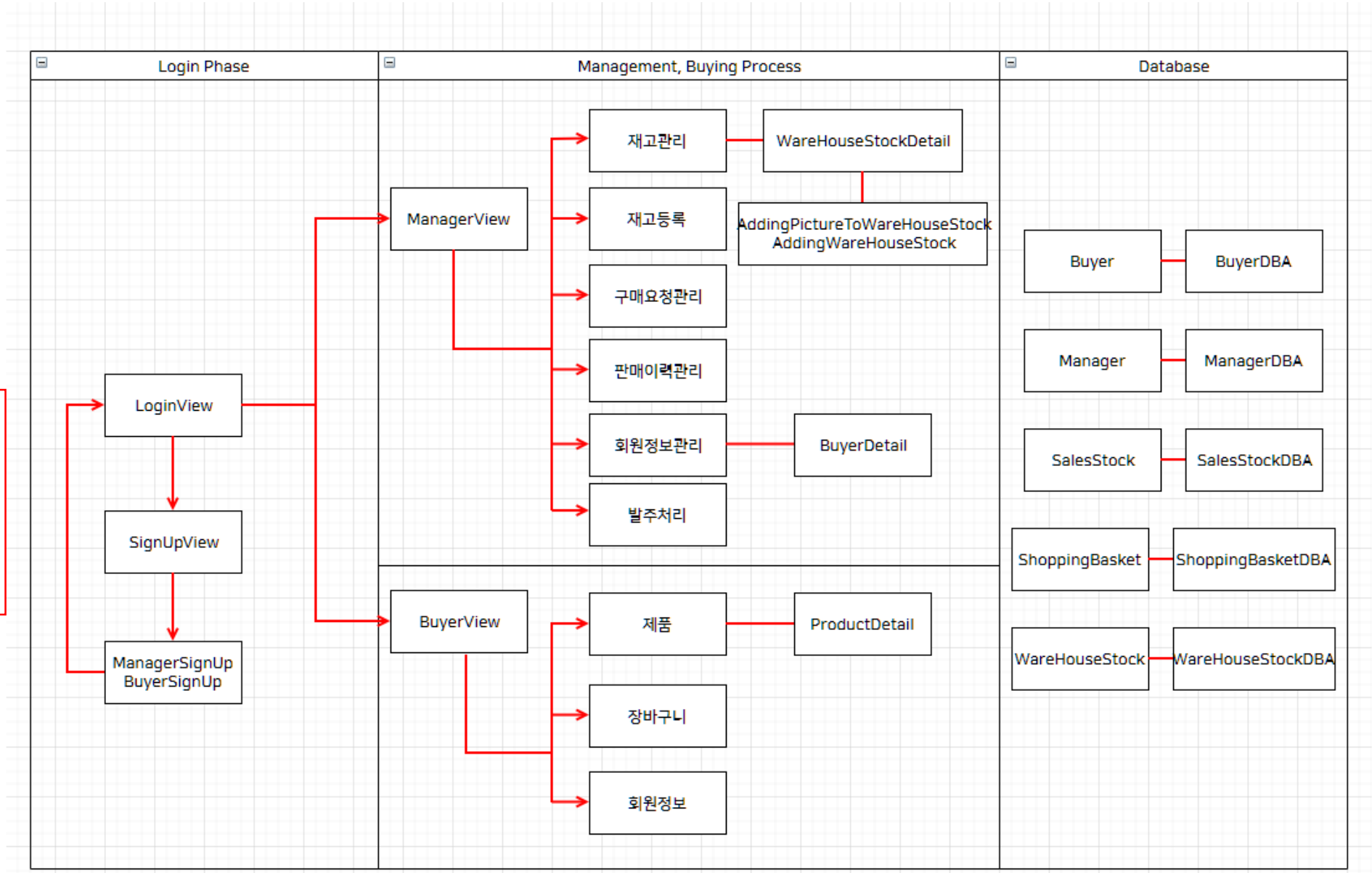
데이터베이스 관계도



관계도

JAVA 클래스 관계도

- NewProJect
  - JRE System Library [JavaSE-1.8]
  - src
    - presentation.model
      - Buyer.java
      - BuyerDBA.java
      - Manager.java
      - ManagerDBA.java
      - SalesStock.java
      - SalesStockDBA.java
      - ShoppingBasket.java
      - ShoppingBasketDBA.java
      - WareHouseStock.java
      - WareHouseStockDBA.java
    - presentation.view
      - AddingPictureToWareHouseStock.java
      - AddingWareHouseStock.java
      - BuyerDetail.java
      - BuyerSignUp.java
      - BuyerView.java
      - GuestView.java
      - LoginView.java
      - ManagerSignUp.java
      - ManagerView.java
      - SignUpView.java
      - WareHouseStockDetail.java
    - presentation.data
      - AddingStockRecord
      - BuyerList.txt
      - LastOrder.txt
      - picturePath.txt
      - SalesList.txt
      - StockList.txt
    - presentation.images
      - background.jpg
      - background2.jpg
  - Referenced Libraries



시연

제품 등록 및 판매 처리

관리자 회원가입

관리자 로그인

재고등록

재고목록확인

재고추가, 사진추가

회원정보확인

구매자 로그인

장바구니에 제품 등록

구매요청

구매요청 관리

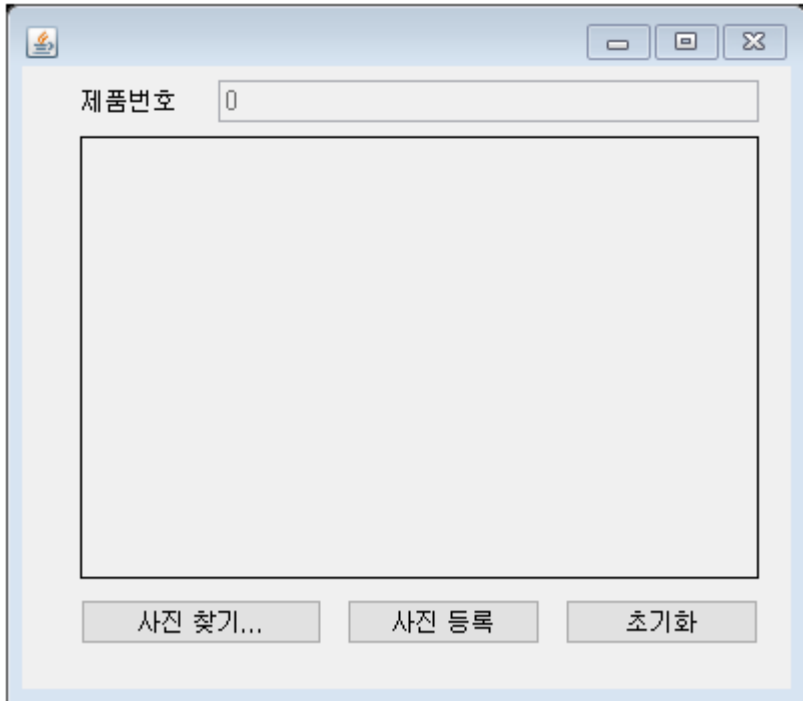
판매이력 관리

분석

코드1 – 상품에 사진추가

```
public class AddingPictureToWareHouseStock extends JFrame {

public static HashMap<Integer,String> pictureHashMap=new HashMap<Integer,String>();
private File dir=new File("src₩₩presentation.data");
private File fPicture=new File(dir,"picturePath.txt");
private String filePath;
```



분석

코드1 – 상품에 사진추가

```
private JButton getBtnNewButton() {
    if (btnNewButton == null) {
        btnNewButton = new JButton("사진 찾기...");
        btnNewButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent arg0) {
/               JOptionPane.show
                JFileChooser fc = new JFileChooser();
                if (fc.showSaveDialog(AddingPictureToWareHouseStock.this) == JFileChooser.CANCEL_OPTION) {
                    return;// 취소버튼 처리
                }
                File f = fc.getSelectedFile();
                filePath=f.getPath();
                iic=new ImageIcon(filePath);
                pictureLabel.setIcon(iic);
            }
        });
        btnNewButton.setBounds(29, 266, 121, 23);
    }
    return btnNewButton;
}
```

분석

코드1 – 상품에 사진추가

```
private JButton getBtnNewButton_1() {
   if (btnNewButton_1 == null) {
      btnNewButton_1 = new JButton("사진 등록");
      btnNewButton_1.addActionListener(new ActionListener() {
         public void actionPerformed(ActionEvent arg0) {
            pictureHashMap.put(pdtNum,filePath);
            JOptionPane.showMessageDialog(null, "사진저장완료");
            PrintStream ps=null;
            try {
               ps=new PrintStream(fPicture);
               Set<Integer> keys=pictureHashMap.keySet();
               Iterator<Integer> it=keys.iterator();
               while(it.hasNext()) {
                  int key=it.next();
                  ps.print(key+"!");
                  ps.print(pictureHashMap.get(key)+"₩n");
               }
               ps.close();
            } catch (FileNotFoundException e) {
               e.printStackTrace();
            }
            dispose();
         }
      });
      btnNewButton_1.setBounds(162, 266, 97, 23);
   }
   return btnNewButton_1;
}
```

분석

코드1 – 상품에 사진추가

```
public BuyerView() {
    Scanner sc = null;
    try {
        sc = new Scanner(fPicture);
        while (sc.hasNextLine()) {
            String[] tmp = sc.nextLine().split("!");
            AddingPictureToWareHouseStock.pictureHashMap.put(Integer.valueOf(tmp[0]), tmp[1]);
        }
        sc.close();
    } catch (FileNotFoundException e) {
        e.printStackTrace();
    }
```

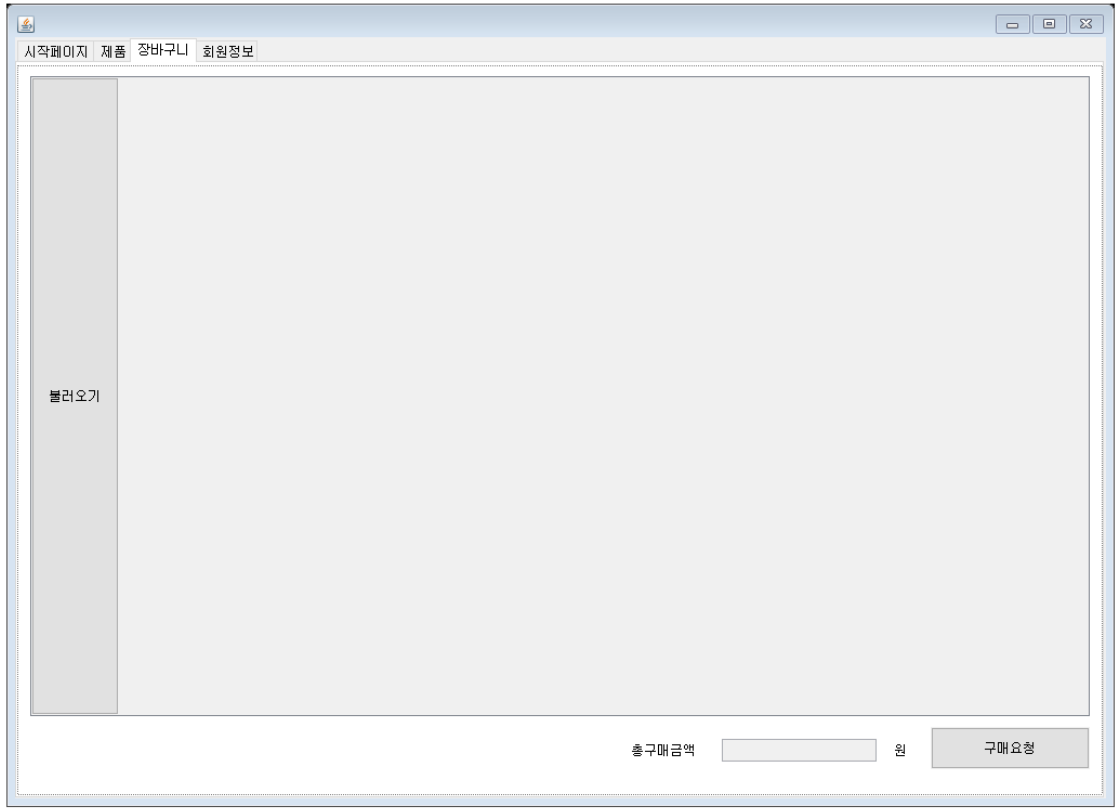
분석

코드1 – 상품에 사진추가

```
class ProductDetail extends JFrame {
    public ProductDetail(WareHouseStock whs) {
        productNum = whs.getProductNum();
        salesNum = whs.getSalesNum();
        empno = whs.getEmpno();
        category = whs.getCategory();
        name = whs.getName();
        info = whs.getInfo();
        stockDate = whs.getStockDate();
        price = whs.getPrice();
        quantity = whs.getStockQuantity();
    private JLabel getLabel_1_1() {
      if (productPictureLabel == null) {
        productPictureLabel = new JLabel("");
        productPictureLabel.setBorder(new LineBorder(new Color(0, 0, 0)));
        productPictureLabel.setBounds(130, 10, 197, 183);
        productPictureLabel
                .setIcon(new ImageIcon(AddingPictureToWareHouseStock.pictureHashMap.get(productNum)));
      }
      return productPictureLabel;
    }
```

분석

코드2 – 구매요청을 처리하여 판매이력으로 저장




```
private JButton getBtnNewButton_1() {
  if (btnNewButton_1 == null) {
    btnNewButton_1 = new JButton("구매요청");
    btnNewButton_1.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent arg0) {
        ArrayList<WareHouseStock> walSB = new ArrayList<WareHouseStock>();

        Set <Integer> keys = sbMap1.keySet();
        Iterator <Integer> it = keys.iterator();

        while(it.hasNext()) {
          walSB.add(whsdba.getOrderedWareHouseStock(it.next()));
        }

        int buyingProductNum=0;
        int[] tmp=new int[walSB.size()];
        for (int i = 0; i < walSB.size(); i++) {
          buyingProductNum=walSB.get(i).getProductNum();
          tmp[i]=buyingProductNum;
        }
        sbdba.addBuyingData(tmp);
      }
    });
    btnNewButton_1.setBounds(844, 611, 147, 39);
  }
  return btnNewButton_1;
}
}

public static HashMap<Integer, Integer> sbMap1 = new HashMap<Integer, Integer>();
```

분석

코드2 – 구매요청을 처리하여 판매이력으로 저장

```
//재고 객체 WareHouseStock을 리턴
public WareHouseStock getOrderedWareHouseStock(int orderPdtNum) {
   Connection con = null;
   Statement st = null;
   ResultSet res = null;
   WareHouseStock whs = null;
   try {
      String sql="SELECT * FROM WareHouseStock WHERE PRODUCT_NUM="+orderPdtNum;
      con = DriverManager.getConnection(url, user, password);
      st = con.createStatement();
      res = st.executeQuery(sql);
      while(res.next()) {
         whs = new WareHouseStock();
         whs.setProductNum(res.getInt("Product_Num"));
         whs.setCategory(res.getString("Category"));
         whs.setName(res.getString("Name"));
         whs.setInfo(res.getString("Info"));
         whs.setStockDate(res.getDate("Stock_Date"));
         whs.setStockQuantity(res.getInt("Stock_Quantity"));
         whs.setPrice(res.getInt("Price"));
         whs.setSalesNum(res.getInt("Sales_Num"));
         whs.setEmpno(res.getInt("empno"));
      }
   } catch (SQLException e) {
      e.printStackTrace();
   } finally {
      closeConnection(con, st, res);
   }
   return whs;
}
```

분석

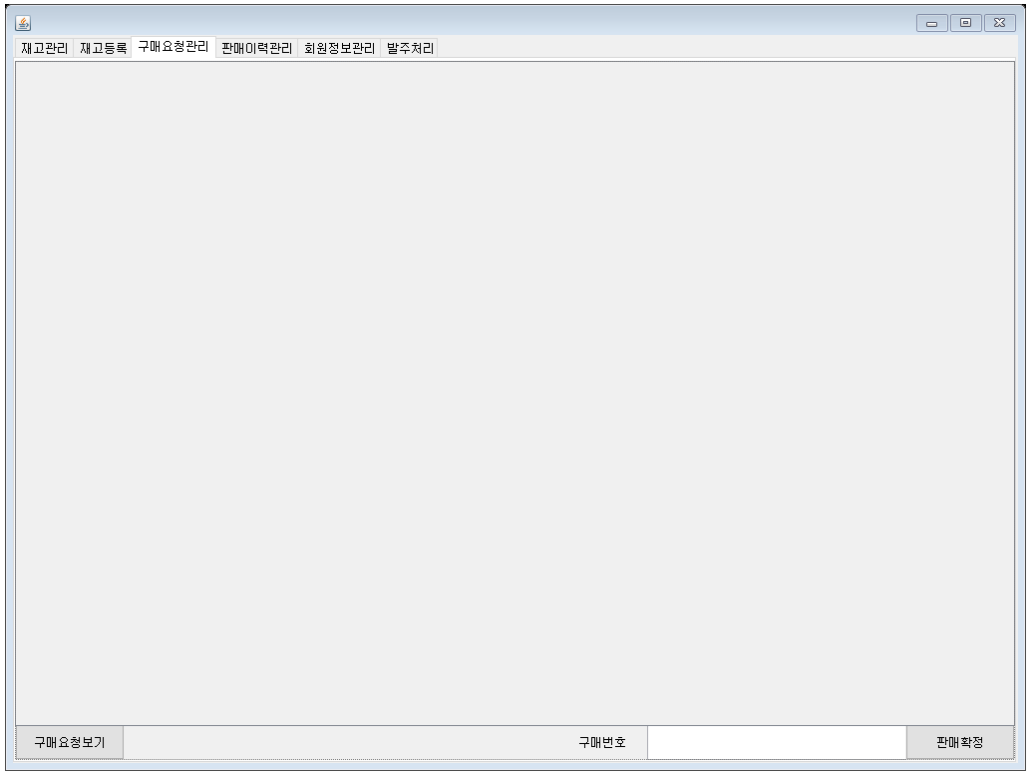
코드2 – 구매요청을 처리하여 판매이력으로 저장

```
  public void addBuyingData(int[] productNum) {
//    INSERT INTO ShoppingBasket
//    VALUES (shoppingbasket_seq.NEXTVAL,1,20190003,10);
      Connection con = null;
      PreparedStatement ps = null;
      try {
         for(int i=0; i<productNum.length; i++) {
            con = DriverManager.getConnection(url, user, password);
            String sql="INSERT INTO ShoppingBasket VALUES (shoppingbasket_seq.NEXTVAL,"+getBuyerNo()+",?,?)";
            ps = con.prepareStatement(sql);
            ps.setInt(1, productNum[i]);
            ps.setInt(2, BuyerView.sbMap1.get(productNum[i]));
            ps.executeUpdate();
         }
      } catch (SQLException e) {
         e.printStackTrace();
      } finally {
         closeConnection(con, ps);
      }

   }
```

분석

코드2 – 구매요청을 처리하여 판매이력으로 저장




```
private JButton getButton_8() {
 if (button_8 == null) {
   button_8 = new JButton("판매확정");
   button_8.addActionListener(new ActionListener() {
     public void actionPerformed(ActionEvent e) {
       SalesStock ss=new SalesStock();
       ss.setBuy_Num(Integer.parseInt(tfBuyingNum.getText()));
       ShoppingBasket sb=sbdba.getShoppingBasket(ss.getBuy_Num());
       ss.setProduct_Num(sb.getProduct_Num());
       ss.setSales_Price(sb.getSellingPrice());
       ss.setSales_Quantity(sb.getBuying_Quantity());
       ssdba.confirmBuyingStuffs(ss);
       btnBuyViewAll.doClick();
       btnSalesViewAll.doClick();
       btnStockViewAll.doClick();
     }
   });
   button_8.setBounds(893, 664, 110, 36);
 }
 return button_8;
}
```

분석

코드2 – 구매요청을 처리하여 판매이력으로 저장

```
public void confirmBuyingStuffs(SalesStock ss) {
//    INSERT INTO SalesBasket
//    VALUES (sales_num, product_num, buy_num, sales_date, sales_quantity, sales_price);
      Connection con1 = null;
      PreparedStatement ps1 = null;
      Connection con2 = null;
      Statement st1 = null;
      try {
         String sql1="INSERT INTO SalesStock VALUES (salesstock_seq.NEXTVAL, ?, SYSDATE, ?, ? ,?)";
         con1 = DriverManager.getConnection(url, user, password);
         ps1=con1.prepareStatement(sql1);
         ps1.setInt(1, ss.getProduct_Num());
         ps1.setInt(2, ss.getSales_Quantity());
         ps1.setInt(3, ss.getSales_Price());
         ps1.setInt(4, ss.getBuy_Num());
         ps1.executeUpdate();
         String sql2="UPDATE WareHouseStock "
               + "SET STOCK_Quantity = (SELECT STOCK_Quantity FROM WareHouseStock WHERE Product_Num="+ss.getProduct_Num()+")-"+ss.getSales_Quantity()
               + " WHERE Product_Num="+ss.getProduct_Num();
         con2 = DriverManager.getConnection(url, user, password);
         st1=con2.createStatement();
         st1.executeUpdate(sql2);
      } catch (SQLException e) {
         JOptionPane.showMessageDialog(null, "이미 처리된 구매요청입니다.");
         return;
      }  finally {
         closeConnection(con1, ps1);
         closeConnection(con2,st1);
         JOptionPane.showMessageDialog(null, "처리완료");
      }
```

한계

로컬 데이터베이스로만 사용가능

실제 사용을 위해 추가해야 할 기능 많음