实验 15　数据库应用系统设计与开发

实验目的

掌握数据库设计的基本方法；了解 C/S 与 B/S 结构应用系统的特点与适用场合；了解 C/S 与 B/S 结构应用系统的不同开发设计环境与开发设计方法；综合运用前面实验掌握的数据库知识与技术设计开发出小型数据库应用系统。

背景知识

《数据库原理及应用》课程的学习，其主要的目标是能利用课程中学习到的数据库知识与技术较好地设计开发出数据库应用系统，去解决各行各业信息化处理的要求。本实验主要在于巩固学生对数据库基本原理和基础理论的理解，掌握数据库应用系统设计开发的基本方法，进一步提高学生综合运用所学知识的能力。

数据库应用设计是指对于一个给定的应用环境，构造最优的数据库模式，建立数据库及其应用系统，有效存储数据，满足用户信息要求和处理要求。

为了使数据库应用系统开发设计合理、规范、有序、正确、高效进行，现在广泛采用的工程化 6 阶段开发设计过程与方法，它们是需求分析阶段、概念结构设计阶段、逻辑结构设计阶段、物理结构设计阶段、数据库的实施、数据库系统运行与维护阶段。以下实验示例的介绍，就是力求按照 6 阶段开发设计过程展开的，以求给读者一个开发设计数据库应用系统的样例。

本实验除了要求你较好地掌握数据库知识与技术外，还要求你熟练掌握某种客户端开发工具或语言。这里，我们分别采用 JAVA、.NET 平台的 C#与 ASP.NET 来实现了两个简单应用系统。

如果你对本实验给出系统所采用的开发工具不熟悉，也无妨的。因为实验示例重点是罗列出开发设计过程及如何利用嵌入的 SQL 命令操作数据库数据的技能，你利用其他工具或语言开发设计系统的过程及操作数据库的技术是相同的，你完全可以利用你掌握的工具或语言来实现相应类似系统。

实验示例

15.1 企业员工管理系统(Java 技术)

随着企业对人才需求的加大，对人力资源管理意识的提高，传统的人事档案管理已经不能满足各个企业对人员管理的需求，企业迫切需要使用新的管理方法与技术来管理员工的相关信息。本系统在极大简化的情况下，想要体现企业员工管理系统的基本雏形，想要体现 Java 技术在传统 C/S 模式、多窗体方式下数据库应用系统的开发方法。本系统的设计与实现能充分体现出 Java 的编程技术，特别是 Java 操作数据库数据的技术。

15.1.1 开发环境与开发工具

系统开发环境为局域网或广域网网络环境，网络中有一台服务器上安装 SQL Server 2014/2012/2008/2005/2000、ORACLE、MySQL 或 PostgreSQL 这样的数据库管理系统，本子系统采用 Java 语言设计实现，使用 jdk1.5.0_15 及 Eclipse SDK Version: 3.3.2（http://www.eclipse.org/platform）为开发工具，服务器操作系统为 Windows Server 2003 family Build 3790 Service Pack 2 及以上版本。

15.1.2 系统需求分析

企业可以通过员工管理系统实现对企业人员信息及其相关信息的管理，简化的企业员工管理系统具有如下功能：

系统的用户管理：包括用户的添加、删除，密码修改等；
员工的信息管理：包括员工基本信息的查询、添加、删除、修改等；
员工的薪资管理：包括员工薪资的查询、添加、删除、修改等；
员工的培训管理：包括员工培训计划的查询、添加、删除、修改等；
员工的奖惩管理：包括对员工的奖惩信息的查询、添加、删除、修改等；
部门的信息管理：包括部门查询、添加、修改、删除等；
其它充分实现对员工信息高效率管理的内容。

15.1.3 功能需求分析

1、系统功能的描述

企业员工管理系统按如上所假设，管理功能是比较简单的，主要实现了对员工、部门、员工的薪资、员工奖惩、员工培训等的管理，具体管理功能有添加、修改、删除、查询、统计等。系统功能布局见系统功能模块图 15-1 所示。

2、系统功能模块图

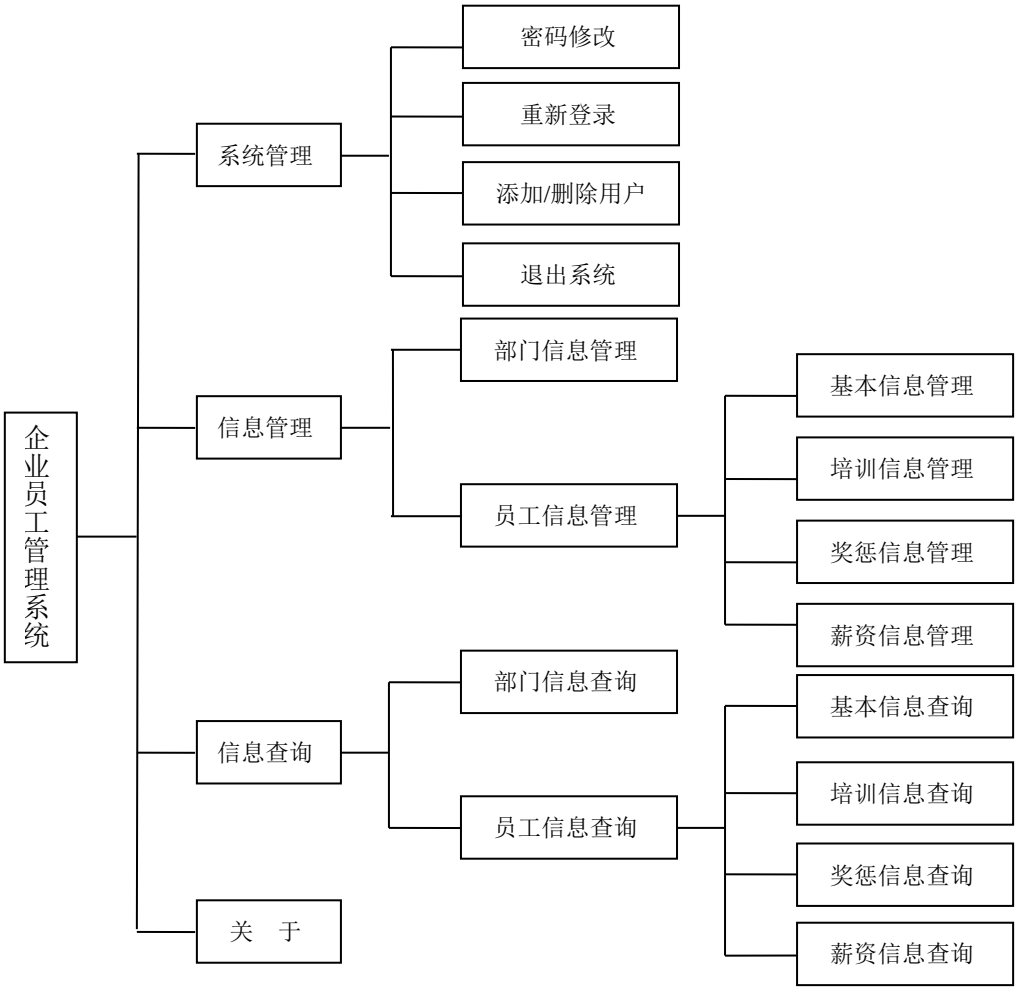


图 15-1 系统功能模块图

其中“信息管理”板块中的每一个功能管理项都包括查看、添加、删除、修改等功能。

15.1.4 系统设计

1、数据概念结构设计

（1）数据流程图

系统数据流程图如图 15-2 所示。

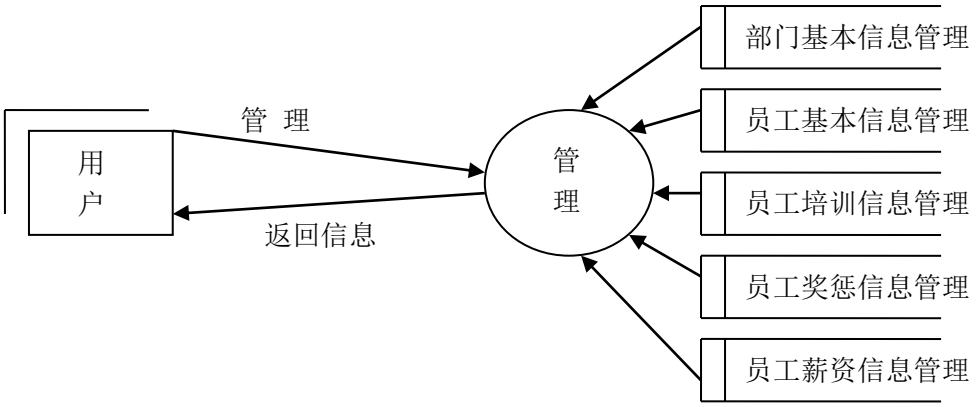


图 15-2 简易系统数据流程图

（2）系统 E-R 图

经调研分析后得简化企业员工管理系统整体基本 E-R 图如图 15-3 所示。

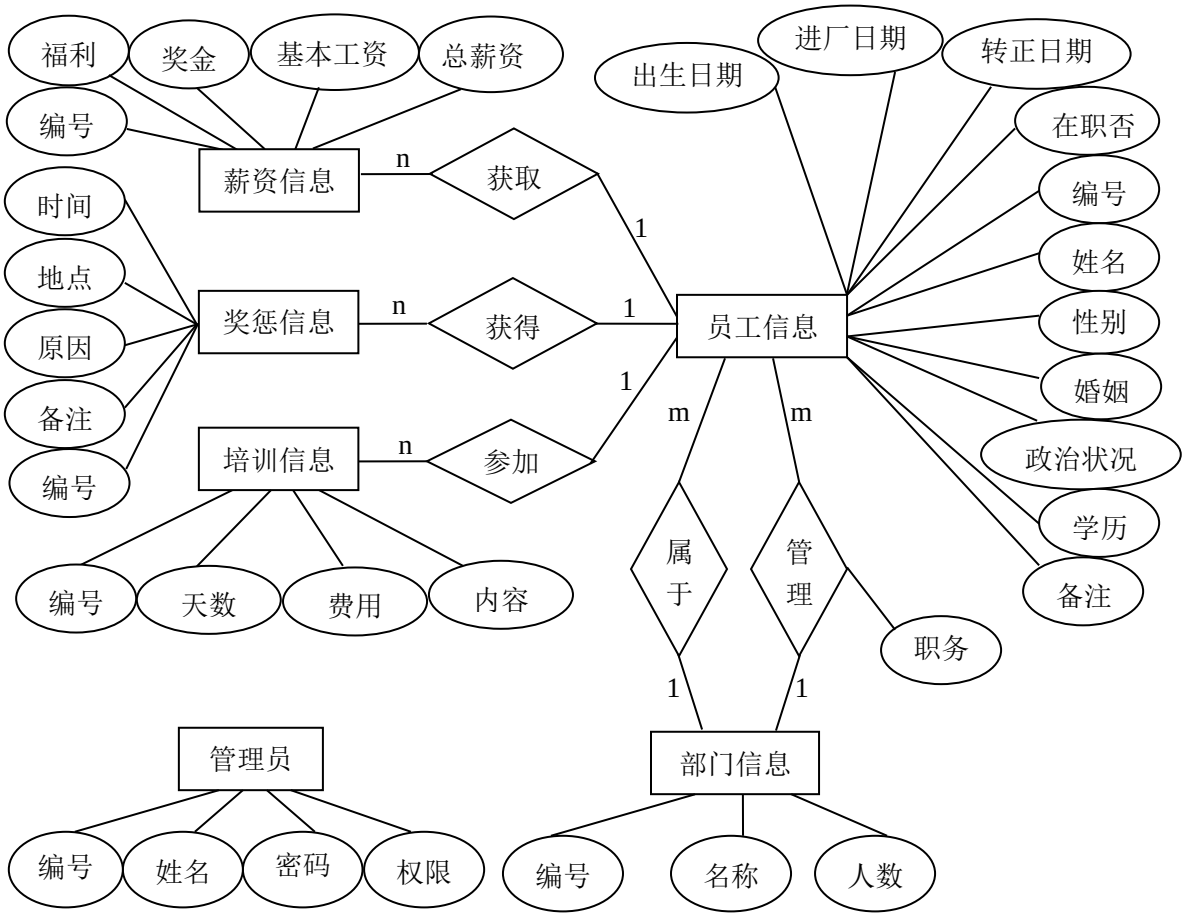


图 15-3 系统基本 E-R 图

2、数据库逻辑结构（关系模式）设计

按照 E-R 图到逻辑关系模式的转换规则，可得到系统如下 6 个关系。

（1）员工信息（员工编号，姓名，性别，学历，政治状况，婚姻，出生日期，在职否，进厂日期，转正日期，部门编号，职务，备注）

（2）奖惩信息（顺序号，奖惩编号，员工编号，奖惩时间，奖惩地点，奖惩原因，备注）

（3）培训信息（顺序号，培训编号，员工编号，培训天数，培训费用，培训内容）

（4）薪资信息（顺序号，薪资编号，员工编号，基本工资，奖金，福利，总薪资）

（5）部门信息（部门编号，部门名称，部门人数）

（6）管理员信息（编号，姓名，密码，权限）

其中带下划线的为关系关键字（即主码）。

3、数据库物理结构设计

本系统数据库表的物理设计通过创建表的 SQL 命令及数据库关系图来呈现，下面只列出 Transact SQL 创建命令（即 T-SQL 命令），针对其它数据库系统的创建命令略。

（1）创建数据库表的 T-SQL 命令

```
CREATE DATABASE EmployeeIMS    -- 创建数据库
GO
-- 以下为创建各表的 SQL 命令
CREATE TABLE [dbo].[DepartmentInformation](
   [D_Number] [int] IDENTITY(1,1) NOT NULL,
   [D_Name] [varchar](20) NOT NULL,
   [D_Count] [int] NOT NULL, CONSTRAINT [PK_DepartmentInformation] PRIMARY KEY CLUSTERED
([D_Number] ASC ));
CREATE TABLE [dbo].[EmployeeInformation](
   [E_Number] [int] IDENTITY(1,1) NOT NULL,
   [E_Name] [varchar](30) NOT NULL,
   [E_Sex] [varchar](2) NOT NULL,
   [E_BornDate] [smalldatetime] NOT NULL,
   [E_Marriage] [varchar](4) NOT NULL,
   [E_PoliticsVisage] [varchar](20) NOT NULL,
   [E_SchoolAge] [varchar](20) NULL,
   [E_EnterDate] [smalldatetime] NULL,
   [E_InDueFormDate] [smalldatetime] NOT NULL,
   [D_Number] [int] NOT NULL,
   [E_Headship] [varchar](20) NOT NULL,
   [E_Estate] [varchar](10) NOT NULL,
   [E_Remark] [varchar](500) NULL, CONSTRAINT [PK_EmployeeInformation] PRIMARY KEY
CLUSTERED ([E_Number] ASC ));
```


```
    CREATE TABLE [dbo].[TrainInformation](
       [ID] [int] IDENTITY(1,1) NOT NULL,
       [T_Number] [varchar](20) NOT NULL,
       [T_Content] [varchar](100) NOT NULL,
       [E_Number] [int] NOT NULL,
       [T_Date] [int] NULL,
       [T_Money] [int] NULL, CONSTRAINT [PK_TrainInformation] PRIMARY KEY CLUSTERED([ID]
ASC ));
    CREATE TABLE [dbo].[WageInformation](
       [ID] [int] IDENTITY(1,1) NOT NULL,
       [W_Number] [int] NOT NULL,
       [E_Number] [int] NOT NULL,
       [W_BasicWage] [decimal](18, 2) NOT NULL,
       [W_Boon] [decimal](18, 2) NOT NULL,
       [W_Bonus] [decimal](18, 2) NOT NULL,
       [W_FactWage] [decimal](18, 2) NOT NULL, CONSTRAINT [PK_WageInformation] PRIMARY KEY
CLUSTERED ([ID] ASC ));
    CREATE TABLE [dbo].[RewardspunishmentInformation](
       [ID] [int] IDENTITY(1,1) NOT NULL,
       [R_Number] [int] NOT NULL,
       [E_Number] [int] NOT NULL,
       [R_Date] [datetime] NOT NULL,
       [R_Address] [varchar](50) NOT NULL,
       [R_Causation] [varchar](200) NOT NULL,
       [R_Remark] [varchar](500) NULL, CONSTRAINT [PK_EncouragementPunishInformation] PRIMARY
KEY CLUSTERED ([ID] ASC ));
    CREATE TABLE [dbo].[UserInformation](
       [User_ID] [int] IDENTITY(1,1) NOT NULL,
       [User_Name] [varchar](20) NOT NULL,
       [Password] [varchar](20) NOT NULL,
       [Authority] [varchar](20) NULL DEFAULT ('B'),CONSTRAINT [PK_UserInformation] PRIMARY
KEY CLUSTERED ([User_ID] ASC ));
    --
    ALTER TABLE [dbo].[EmployeeInformation]  WITH CHECK ADD  CONSTRAINT
[FK_EmployeeInformation_DepartmentInformation] FOREIGN KEY([D_Number]) REFERENCES
[dbo].[DepartmentInformation] ([D_Number])
    ALTER TABLE [dbo].[TrainInformation]  WITH CHECK ADD  CONSTRAINT
[FK_TrainInformation_EmployeeInformation] FOREIGN KEY([E_Number]) REFERENCES
[dbo].[EmployeeInformation] ([E_Number])
    ALTER TABLE [dbo].[WageInformation]  WITH CHECK ADD  CONSTRAINT
[FK_WageInformation_EmployeeInformation] FOREIGN KEY([E_Number]) REFERENCES
[dbo].[EmployeeInformation] ([E_Number])
    ALTER TABLE [dbo].[RewardspunishmentInformation]  WITH CHECK ADD  CONSTRAINT
[FK_EncouragementPunishInformation_EmployeeInformation] FOREIGN KEY([E_Number]) REFERENCES
[dbo].[EmployeeInformation] ([E_Number])
```

（2）数据库关系图

数据库关系图如图 15-4 所示。

图 15-4 数据库关系图

按需还可创建索引及视图的，此处略。

15.1.5 系统功能的实现

1、数据库连接通用模块

数据库连接、公用操作函数等代码见如下数据库类 Database。

```
package qxz;
import java.sql.*;
```

```
import javax.swing.JComboBox;
import javax.swing.JList;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;
public class Database {
    public static Connection cn;
    public static Statement st;
    public static Statement st2;
    public static ResultSet rs;
    public static String dbms;
   // below for SQL Server
   static String user = ConfigIni.getIniKey ("UserID");
   static String pwd  = ConfigIni.getIniKey ("Password");
   static String ip   = ConfigIni.getIniKey ("IP");
   static String acc  = ConfigIni.getIniKey ("Access");
   static String dbf  = ConfigIni.getIniKey ("DataBase");
   // below for Oracle
   static String UID=ConfigIni.getIniKey ("UID");
   static String Passd=ConfigIni.getIniKey ("Passd");
   static String Server=ConfigIni.getIniKey ("Server");
   static String DB=ConfigIni.getIniKey ("DB");
   static String Port=ConfigIni.getIniKey ("Port");
   // below for MySQL
   static String UID2=ConfigIni.getIniKey ("UID2");
   static String Passd2=ConfigIni.getIniKey ("Passd2");
   static String Server2=ConfigIni.getIniKey ("Server2");
   static String DB2=ConfigIni.getIniKey ("DB2");
   static String Port2=ConfigIni.getIniKey ("Port2");
   // below for PostgreSQL
   static String UID3=ConfigIni.getIniKey ("UID3");
   static String Passd3=ConfigIni.getIniKey ("Passd3");
   static String Server3=ConfigIni.getIniKey ("Server3");
   static String DB3=ConfigIni.getIniKey ("DB3");
   static String Port3=ConfigIni.getIniKey ("Port3");
   static {
      try {
        if(ConfigIni.getIniKey ("Default_Link").equals ("1")) {//JDBC--SQL Server 2005
          DriverManager.registerDriver (new
com.microsoft.sqlserver.jdbc.SQLServerDriver());  //注册驱动
          String url="jdbc:sqlserver://"+ip+":"+acc+";"+"databasename="+dbf;//获得一
个连接
          cn = DriverManager.getConnection (url, user, pwd);
          dbms="SQL Server";
        }
        else if(ConfigIni.getIniKey ("Default_Link").equals ("2")) {//JDBC-SQL Server
2000
            DriverManager.registerDriver (new
com.microsoft.jdbc.sqlserver.SQLServerDriver()); //注册驱动
            String url  = "jdbc:microsoft:sqlserver://" + ip + ":" + acc + ";" +
"databasename=" + dbf;   //获得一个连接
            cn = DriverManager.getConnection (url, user, pwd);
```

```
            dbms="SQL Server";
        }
        else if(ConfigIni.getIniKey ("Default_Link").equals ("4")) {//JDBC-ODBC to
Oracle
            DriverManager.registerDriver(new sun.jdbc.odbc.JdbcOdbcDriver());
            cn = DriverManager.getConnection
("jdbc:odbc:"+ConfigIni.getIniKey("LinkNameORA").trim(),UID,Passd); //获得一个连接
            dbms="Oracle";
        }
        else if(ConfigIni.getIniKey ("Default_Link").equals ("5")) {//JDBC to Oracle
           DriverManager.registerDriver(new oracle.jdbc.driver.OracleDriver());
           String url = "jdbc:oracle:thin:@"+Server+":"+Port+":"+DB;
           cn =DriverManager.getConnection (url, UID, Passd);
           dbms="Oracle";
        }
        else if(ConfigIni.getIniKey("Default_Link").equals ("6")) {//JDBC to MySQL
           try { Class.forName("com.mysql.jdbc.Driver").newInstance();
           } catch (Exception ex) {  }
            String url
="jdbc:mysql://localhost/"+DB2+"?"+"user="+UID2+"&"+"password="+Passd2;
           try {  cn=DriverManager.getConnection(url);
           } catch (SQLException ex) {
              System.out.println("SQLException:"+ex.getMessage());
              System.out.println("SQLState: "+ex.getSQLState());
              System.out.println("VendorError:"+ex.getErrorCode());
           }
           dbms="MySQL";
        }
        else if(ConfigIni.getIniKey ("Default_Link").equals ("7")) {//JDBC-ODBC to MySQL
           DriverManager.registerDriver(new sun.jdbc.odbc.JdbcOdbcDriver());
           cn = DriverManager.getConnection
("jdbc:odbc:"+ConfigIni.getIniKey("LinkNameMySQL").trim(),UID2,Passd2);
           dbms="MySQL";
           Linknum="7";
        }
        else if(ConfigIni.getIniKey ("Default_Link").equals ("8")) {//JDBC-ODBC to
postgresql
           Class.forName("org.postgresql.Driver");
           String url = "jdbc:postgresql://"+Server3+":"+Port3+"/"+DB3;
           cn = DriverManager.getConnection(url, UID3, Passd3);
           dbms="PostgreSQL";
        }
        else if(ConfigIni.getIniKey ("Default_Link").equals ("9")) {//JDBC-ODBC to
PostgreSQL
           DriverManager.registerDriver(new sun.jdbc.odbc.JdbcOdbcDriver());
           cn = DriverManager.getConnection
("jdbc:odbc:"+ConfigIni.getIniKey("LinkNamePostgreSQL").trim(),UID3,Passd3);
           dbms="PostgreSQL";
        }
        else {//ConfigIni.getIniKey("Default_Link").equals("3") //JDBC-ODBC to SQL
Server
```


```
            DriverManager.registerDriver(new sun.jdbc.odbc.JdbcOdbcDriver());
            cn = DriverManager.getConnection
  ("jdbc:odbc:"+ConfigIni.getIniKey("LinkName").trim(),user,pwd);
            dbms="SQL Server";
          }
          st =
  cn.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,ResultSet.CONCUR_READ_ONLY);
          st2 =
  cn.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,ResultSet.CONCUR_READ_ONLY);
        }
        catch (Exception ex) {
          System.out.println(ex.getMessage().toString()+"--");
          JOptionPane.showMessageDialog (null, "数据库连接失败...", "错误",
  JOptionPane.ERROR_MESSAGE);
          System.exit(0);
        } //End try
    }
    // 执行查询SQL命令，返回记录集对象函数
    public static ResultSet executeQuery(String sql) {
        ResultSet rs = null ;
        try {
           st2 =
cn.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,ResultSet.CONCUR_READ_ONLY);// 需要
此句否则相互干扰
            rs = st2.executeQuery(sql);
        }catch(Exception e){
            e.printStackTrace();
        }//End try
        return rs;
    }
    // 执行更新类SQL命令的函数
    public static int executeUpdate(String sql) {
        int i = 0 ;
        try {
           st2 =
cn.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,ResultSet.CONCUR_READ_ONLY);
            i = st2.executeUpdate(sql) ;
            cn.commit();
        }
        catch(Exception e) {
           e.printStackTrace();
        }
        return i;
    }
    // 执行查询SQL命令，返回是否成功的函数
    public static boolean query(String sqlString){
        try {
            rs = null;
            rs = st.executeQuery(sqlString);
        }catch (Exception Ex) {
            System.out.println("sql exception:" + Ex);
```

```
            return false;
        }
        return true;
    }
    // 执行更新类SQL命令，返回是否成功的函数
    public static boolean executeSQL(String sqlString){
        boolean executeFlag;
        try{st.execute(sqlString);
            executeFlag = true;
        } catch (Exception e) {
            executeFlag = false;
            System.out.println("sql exception:" + e.getMessage());
        }
        return executeFlag;
    }
    // 执行SQL查询命令，初始化到组合框的函数
     public static void initJComboBox (JComboBox jComboBox, String sqlCode){
          jComboBox.removeAllItems();
          try{ ResultSet rs = executeQuery (sqlCode);
               int row = recCount (rs);
               //从结果集中取出Item加入JComboBox中
               if(row != 0) rs.beforeFirst();
               for (int i = 0; i < row; i++) {
                    rs.next();
                    jComboBox.addItem (rs.getString(1));
               }
               jComboBox.addItem("");
          }
          catch (Exception ex) {
               System.out.println ("sunsql.initJComboBox (): false");
          }
     }
    ..... // 其它公用函数略。
}
```

本程序通过ConfigIni.java文件中的ConfigIni类来获取连接数据库的相关信息。这些连接数据库的相关信息组织存放于Config.ini系统配置文件中，这样便于修改与配置连接数据库的相关参数值。该文件的参考内容如下：

```
[SOFTINFO]=
  UserName=qxz
  CompName=jndx
[CONFIG]=
  Soft_First=0
  Default_Link=6
  Default_Page=1
[JDBC 1--SQL Server 2005, 2--SQL Server 2000]=
  IP=127.0.0.1
  Access=1433
  DataBase=EmployeeIMS
  UserID=sa
  Password=sasasasa
[ODBC 3--odbc to SQL Server]=
```


```
  LinkName=EmployeeIMS
[ODBC 4--odbc to Oracle]=
  LinkNameORA=EmployeeIMSORA
[JDBC 5--Oracle ]=
  UID=scott
  Passd=tiger
  Server=qxz1
  DB=qxz1
  Port=1521
[JDBC 6--MySQL ]=
  UID2=root
  Passd2=qxz
  Server2=qxz1
  DB2=EmployeeIMS
  Port2=3306
[ODBC 7--odbc to MySQL]=
  LinkNameMySQL=EmployeeMySQL
[JDBC 8--PostgreSQL ]=
  UID3=qxz
  Passd3=qxz
  Server3=localhost
  DB3=EmployeeIMS2
  Port3=5432
[ODBC 9--odbc to PostgreSQL]=
  LinkNamePostgreSQL=EmployeePostgreSQL
```

其中“Default_Link=”指定1-9中某数字，代表着连接数据库的某种方式方法。可以看到：“Default_Link=1”表示通过JDBC连接到SQL Server 2005；“Default_Link=2”表示通过JDBC连接到SQL Server 2000；“Default_Link=3”表示通过JDBC-ODBC桥连接到SQL Server 2000或SQL Server 2005；“Default_Link=4”表示通过JDBC-ODBC桥连接到Oracle；“Default_Link=5”表示通过JDBC连接到Oracle；“Default_Link=6”表示通过JDBC连接到MySQL；“Default_Link=7”表示通过JDBC-ODBC桥连接到MySQL；“Default_Link=8”表示通过JDBC连接到PostgreSQL；“Default_Link=9”表示通过JDBC-ODBC桥连接到PostgreSQL。

要说明的是：要使1到9种连接数据库方法能正常工作，需要先在服务器端安装相应数据库管理系统并正确配置,再通过执行SQL脚本等方法在某数据库系统下创建系统库表等对象，在Config.ini系统配置文件中正确配置相应某数据库的连接选项值，只有这样才能成功运行的。特别注意，**从Java SE8起，JDK中将不再包含JDBC-ODBC桥了**。这样要么采用Java SE8以前的版本，要么就不用JDBC-ODBC桥连接方式了。

2、部分功能界面的实现

（1）系统登录及主界面类模块

```
//用户登陆类
package qxz;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
```

```
public class Login extends JFrame{
    JFrame jf ;
    JTextField textName=new JTextField("admin");
    JPasswordField textPassword=new JPasswordField("123456");
    JLabel label = new JLabel("企业员工管理系统");
    ……  // 其它界面元素及变量等定义略
    public Login(){
        jf=this;
        setTitle("登录");
        Font f = new Font("新宋体",Font.PLAIN,12);
        Container con = getContentPane();
        con.setLayout(null);
        label.setBounds(80,10,140,20);
        label.setFont(new Font("新宋体",Font.BOLD,16));
        con.add(label);
        labelName.setBounds(55,45,55,20);
        labelName.setFont(f);
        con.add(labelName);
        textName.setBounds(105,45,120,20);
        con.add(textName);
        ……  // 其它界面元素定位、赋属性等略
        //登录的鼠标监听
        buttonEnter.addMouseListener(new MouseAdapter(){
            public void mouseClicked(MouseEvent me){
                if(textName.getText().equals("")){
                   new JOptionPane().showMessageDialog(null,"用户名不能为空!");
                }
                else if(textPassword.getText().equals("")){
                   new JOptionPane().showMessageDialog(null,"密码不能为空!");
                }
                else{
                   String sql="select * from UserInformation where User_Name = '" +
textName.getText() + "' and Password = '" + textPassword.getText()+ "'"; // 查找是否有
该用户的SQL查询命令
                     Judge(sql);   // 调用判断函数
                }
            }
        });
       //登录的键盘监听
        buttonEnter.addKeyListener(new KeyAdapter(){
          public void keyPressed(KeyEvent e){
           // 略
          }
       });
        buttoncancel.setBounds(155,115,60,20);
        buttoncancel.setFont(f);
        con.add(buttoncancel);
        //清空按钮的鼠标监听方法
        buttoncancel.addMouseListener(new MouseAdapter(){
            public void mouseClicked(MouseEvent me){
                textName.setText("");
```


```
                    textPassword.setText("");
                }
            });
            //窗口大小不可调
            setResizable(false);
            //窗口图标
            Image img=Toolkit.getDefaultToolkit().getImage("image\\main.gif");
            setIconImage(img);
            Toolkit t = Toolkit.getDefaultToolkit();
            int w = t.getScreenSize().width;
            int h = t.getScreenSize().height;
            setBounds(w/2-150,h/2-90,300,180);
            setVisible(true);
            // 获取焦点
            buttonEnter.grabFocus();
            buttonEnter.requestFocusInWindow();
        }
        private void Judge(String sqlString) {
            if (Database.joinDB()) {
              if (Database.query(sqlString))
                 try{  if(Database.rs.isBeforeFirst()) {
                       System.out.println("密码正确");
                       jf.setVisible(false);
                       //关闭数据库连接
                       //Database.cn.close();
                       new Main();
                    }
                    else{System.out.println("错误");
                       new JOptionPane().showMessageDialog(null,"用户名或密码错
误!","",JOptionPane.ERROR_MESSAGE);//!!!!!!!!!!!!!!
                    }
                 }catch(Exception ex) {
                     System.out.println(ex.getMessage());
                 }
            }
            else{ System.out.println("连接数据库不成功!!!"); }
      }
      public static void main(String args[]){
         new Login();
      }
```

运行界面如图 15-5 所示。

图 15-5 系统登录界面

```
//主程序类，可以独立运行
package qxz;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class Main extends JFrame implements Runnable{
    Thread t=new Thread(this);//在窗体里创建线程并实例化
    JDesktopPane deskpane = new JDesktopPane();//在窗体里建立虚拟桌面并实例化
    JPanel p = new JPanel();//创建一个面板并实例化
    Label lp1=new Label("欢 迎 使 用 企 业 员 工 管 理 系 统 ！ 本 系 统 纯 属 练 习 ！");
   //菜单上的图标创建并实例化----------------------------------
    ImageIcon icon1=new ImageIcon("image//tjsc.gif");
   …… //其它略--------------------------------
    public Main(){//构造函数
        setTitle("企业员工管理系统");//设置窗体标题
        Container con = getContentPane();
        con.setLayout(new BorderLayout());    //创建一个布局
        con.add(deskpane,BorderLayout.CENTER);//实例虚拟桌面的布局
        Font f =new Font("新宋体",Font.PLAIN,12);//设置一个字体，以后都使用这种字体
        JMenuBar mb = new JMenuBar();          //实例化菜单栏
        //实例化菜单开始
        JMenu systemM = new JMenu("系统管理");
        systemM.setFont(f);
        JMenu manageM = new JMenu("信息管理");
        manageM.setFont(f);
       …… //其它略
       // 退出窗体事件
        this.addWindowListener(new WindowAdapter(){
            public void windowClosing(WindowEvent e){
               System.exit(0);
           }});
       //为系统管理菜单加事件-----------------------------------------------------
       password.addActionListener(new ActionListener(){//密码修改监听
        public void actionPerformed(ActionEvent e){
            System.out.println("AmendPassword");
            deskpane.add(new AmendPassword());
        }});
       …… //其它事件略
       p.setLayout(new BorderLayout());
       p.add(lp1,BorderLayout.EAST);
       t.start();
       con.add(p,BorderLayout.SOUTH);
       Toolkit t = Toolkit.getDefaultToolkit();
       int width = t.getScreenSize().width - 120 ;
       int height = t.getScreenSize().height - 100;
       setSize(width,height);
       setLocation(50,25);
       setVisible(true);
       setResizable(false);
```

```
        }
        //线程的方法
        public void run(){
           System.out.println("线程启动了!");
           Toolkit t = Toolkit.getDefaultToolkit();
           int x=t.getScreenSize().width;
           lp1.setForeground(Color.red);
           while(true)
           {   if(x<-600){ x=t.getScreenSize().width;}
               lp1.setBounds(x,0,700,20);
               x-=10;
               try{Thread.sleep(100);}catch(Exception e){}
           }
        }
        // 退出窗体事件
        public void windowClosing(WindowEvent e) {
            System.exit(0);
        }
        public static void main(String[] args){ //主函数
            new Main();
        }
}
```

运行界面如下图 15-6 所示。

图 15-6 系统主界面

（2）员工基本信息维护类模块

```
//员工信息管理类
package qxz;
import java.awt.*;
```

```
import javax.swing.*;
import javax.swing.text.DateFormatter;
import java.awt.event.*;
import java.sql.*;
import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.util.Date;
public class Employeemanage extends JInternalFrame{
     JInternalFrame jif;
    public Employeemanage() {
         jif=this;
        initComponents(); }
    private JTextField tdepartment;
    private JComboBox jComboBox,jComboBoxCode;
    private ResultSet rs;
    private void initComponents() { //初始化界面组件
         //定义与初始化组合框
        jComboBox = new JComboBox();
        jComboBox.addItem("");
         jComboBox.setBackground(new Color(204, 204, 204));
        jComboBox.setPreferredSize(new Dimension(100, 20));
        ……  // 其它初始化略
        // 部门编码信息添加到组合框jComboBox,jComboBoxCode
          jComboBox.removeAllItems();
          try {
               ResultSet rs2 = Database.executeQuery("select D_Name,D_Number from
DepartmentInformation order by D_Name");
               int row = Database.recCount(rs2);
               //从结果集中取出Item加入JComboBox中
               if(row != 0) rs2.beforeFirst ();
               for (int i = 0; i < row; i++) {
                    rs2.next();
                    jComboBox.addItem (rs2.getString (1));
                    jComboBoxCode.addItem (rs2.getString (2));
               }
               jComboBox.addItem("");
               jComboBoxCode.addItem("");
               rs2.close();
          }
          catch (Exception ex) {
              System.out.println ("initJComboBox (): false");
          }
        //初始化窗体数据
        String csql="select
E_Number,E_Name,E_Sex,E_BornDate,D_Number,E_Marriage,E_Headship,E_InDueFormDate,E_Poli
ticsVisage,E_SchoolAge,E_EnterDate,E_Estate,E_Remark from EmployeeInformation";
        try{
           rs= Database.executeQuery(csql);
           if(Database.recCount(rs)>0){
              rs.next();
              txt_number.setText("" + rs.getInt("E_Number"));
```


```
            txt_name.setText(rs.getString("E_Name"));
            if(rs.getString("E_Sex").equals("男")){
                 sex_cb.setSelectedIndex(0);
            }
            else{
                 sex_cb.setSelectedIndex(1);
            }
            txt_borndate.setValue(rs.getDate("E_BornDate"));
            tdepartment.setText(rs.getString("D_Number"));
                if(rs.getString("E_Marriage").equals("未婚")){
                 marriage_cb.setSelectedIndex(0);
            }
            else if(rs.getString("E_Marriage").equals("已婚")){
                 marriage_cb.setSelectedIndex(1);
            }
            else{
                 marriage_cb.setSelectedIndex(2);
            }
            headship_cb.setSelectedItem(rs.getString("E_Headship"));
            txt_InDueFormDate.setValue(rs.getDate("E_InDueFormDate"));
            if(rs.getString("E_PoliticsVisage").equals("党员")){
                 politicsVisage_cb.setSelectedIndex(0);
            }
            else{
                 politicsVisage_cb.setSelectedIndex(1);
            }
            schoolage_cb.setSelectedItem(rs.getString("E_SchoolAge"));
            txt_enterdate.setValue(rs.getDate("E_EnterDate"));
            if(rs.getString("E_Estate").equals("在职")){
                 estate_cb.setSelectedIndex(0);
            }
            else if(rs.getString("E_Estate").equals("停薪留职")){
                 estate_cb.setSelectedIndex(1);
            }
            else{
                 estate_cb.setSelectedIndex(2);
            }
            remark_ta.setText(rs.getString("E_Remark"));
          }
          jComboBoxCode.setSelectedItem(tdepartment.getText());
           jComboBox.setSelectedIndex(jComboBoxCode.getSelectedIndex());
      }
      catch(Exception e){System.out.println(e);};
       //上一条按钮事件
       rm_bt.addActionListener(new ActionListener(){
          public void actionPerformed(ActionEvent e){
            try{
                 if(rs.next()){
                      txt_number.setText("" + rs.getInt("E_Number"));
                      txt_name.setText(rs.getString("E_Name"));
                      if(rs.getString("E_Sex").equals("男")){
```

```
                        sex_cb.setSelectedIndex(0);
                    }
                      else{
                      sex_cb.setSelectedIndex(1);
                      }
                    txt_borndate.setValue(rs.getDate("E_BornDate"));
                       tdepartment.setText(rs.getString("D_Number"));
    Database.setJComboBox(jComboBoxCode,rs.getString("D_Number"));
jComboBox.setSelectedIndex(jComboBoxCode.getSelectedIndex());
                       if(rs.getString("E_Marriage").equals("未婚")){
                          marriage_cb.setSelectedIndex(0);
                       }
                       else if(rs.getString("E_Marriage").equals("已婚")){
                          marriage_cb.setSelectedIndex(1);
                       }
                       else{
                          marriage_cb.setSelectedIndex(2);
                       }
headship_cb.setSelectedItem(rs.getString("E_Headship"));
    txt_InDueFormDate.setValue(rs.getDate("E_InDueFormDate"));
                       if(rs.getString("E_PoliticsVisage").equals("党员")){
                          politicsVisage_cb.setSelectedIndex(0);
                       }
                       else{
                          politicsVisage_cb.setSelectedIndex(1);
                       }
                       schoolage_cb.setSelectedItem(rs.getString("E_SchoolAge"));
                       txt_enterdate.setValue(rs.getDate("E_EnterDate"));
                       if(rs.getString("E_Estate").equals("在职")){
                          estate_cb.setSelectedIndex(0);
                       }
                       else if(rs.getString("E_Estate").equals("停薪留职")){
                          estate_cb.setSelectedIndex(1);
                       }
                       else{
                          estate_cb.setSelectedIndex(2);
                       }
                       remark_ta.setText(rs.getString("E_Remark"));
                   }
               }
               catch(Exception erm){ System.out.println(erm);}
               }
          });
          //下一条按钮事件
          lm_bt.addActionListener(new ActionListener(){
           ·····  //具体略
          });
          //最前一条按钮事件
          left_bt.addActionListener(new ActionListener(){
           ·····  //具体略
          });
```


```
            right_bt.addActionListener(new ActionListener(){
             ·····  //具体略
            });
        //为添加保存按钮加事件
        append_bt.addActionListener(new ActionListener(){
           public void actionPerformed(ActionEvent e){
                save_bt.setEnabled(true);        txt_number.setText("");
                txt_number.setEditable(false);   txt_name.setText("");
                sex_cb.setSelectedIndex(0);      txt_borndate.setValue(new Date());
                marriage_cb.setSelectedIndex(0); txt_InDueFormDate.setValue(new Date());
                politicsVisage_cb.setSelectedIndex(0);txt_enterdate.setValue(new Date());
                estate_cb.setSelectedIndex(0);   remark_ta.setText("");
            }
         });
        // 组合框，选项事件
        jComboBox.addItemListener(new ItemListener() {
            public void itemStateChanged(ItemEvent e) {
             jComboBoxCode.setSelectedIndex(jComboBox.getSelectedIndex());
        }});
         //为添加保存按钮添加事件
         save_bt.addActionListener(new ActionListener(){
                public void actionPerformed(ActionEvent e){
                   if(txt_name.getText().equals("")||txt_borndate.getText().equals("")
                    ||txt_InDueFormDate.getText().equals("")
                     ||txt_enterdate.getText().equals("")){
            new JOptionPane().showMessageDialog(null,"除备注外，其余数据均不能为空！");}
                     else{
                       String name=txt_name.getText();
                       String borndate=txt_borndate.getText();
                       String department=tdepartment.getText();
                       String headship=("" + headship_cb.getSelectedItem());
                       String indueformdate=txt_InDueFormDate.getText();
                       String schoolage=("" + schoolage_cb.getSelectedItem());
                       String enterdate=txt_enterdate.getText();
                       String remark=remark_ta.getText();
                       String sex=("" + sex_cb.getSelectedItem());
                       String marriage=(""+marriage_cb.getSelectedItem());
                       String estate=("" + estate_cb.getSelectedItem());
                       String politicsVisage=("" + politicsVisage_cb.getSelectedItem());
                       jComboBoxCode.setSelectedIndex(jComboBox.getSelectedIndex());
                       String sInsert = "";
                       if(Database.dbms.equals("SQL Server")){
                          sInsert="insert EmployeeInformation values('"+ name +"','"+ sex
+"','"+  borndate+"',"+ "'"+ marriage +"','"+ politicsVisage +"','"+ schoolage +"','"+
enterdate +"','"+ indueformdate +"',"+jComboBoxCode.getSelectedItem() +",'"+ headship
+"','"+  estate +"','"+ remark +"')";
                       }
                       if(Database.dbms.equals("Oracle")){
                          sInsert="insert into EmployeeInformation values(SEI.nextval,'"+
name +"','"+ sex +"',to_date('"+  borndate+"','YYYY-MM-DD'),"+"'"+ marriage +"','"+
politicsVisage +"','"+ schoolage +"',to_date('"+ enterdate +"','YYYY-MM-DD'),to_date('"+
```

```
indueformdate +"','YYYY-MM-DD'),"+ jComboBoxCode.getSelectedItem() +","+ headship
+"','"+  estate +"','"+ remark +"')";
                    }
                    if(Database.dbms.equals("MySQL")){
                        sInsert="insert into EmployeeInformation values(null,'"+ name
+"','"+ sex +"','"+borndate+"',"+"'"+ marriage +"','"+ politicsVisage +"','"+ schoolage
+"','"+ enterdate +"','"+ indueformdate +"',"+jComboBoxCode.getSelectedItem() +",'"+
headship +"','"+  estate +"','"+ remark +"')";
                    }
                    if(Database.dbms.equals("PostgreSQL")){
                        sInsert="insert into EmployeeInformation
values(nextval('EmployeeInformation_E_Number_seq'),'"+ name +"','"+ sex
+"','"+borndate+"',"+"'"+ marriage +"','"+ politicsVisage +"','"+ schoolage +"','"+
enterdate +"','"+ indueformdate +"',"+jComboBoxCode.getSelectedItem() +",'"+ headship
+"','"+  estate +"','"+ remark +"')";
                    }
                    try{  if(Database.executeUpdate(sInsert)!=0){
                              txt_number.setEditable(true);
                              save_bt.setEnabled(false);
                              new JOptionPane().showMessageDialog(null,"添加数据成功！");
//                            Database.joinDB();
                              String sql="select
E_Number,E_Name,E_Sex,E_BornDate,D_Number,E_Marriage,E_Headship,E_InDueFormDate,E_Poli
ticsVisage,E_SchoolAge,E_EnterDate,E_Estate,E_Remark from EmployeeInformation";
                              rs= Database.executeQuery(sql);
                              rs.last();
                              txt_number.setText("" + rs.getInt("E_Number"));
                              }
                    }
                    catch(Exception einsert){ System.out.println(einsert);}
                    save_bt.setEnabled(false);
                }
            }
        });
    //为修改按钮添加事件
    amend_bt.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent e){
            String name=txt_name.getText();
            String borndate=txt_borndate.getText();
            String department=tdepartment.getText();
            String headship=("" + headship_cb.getSelectedItem());
            String indueformdate=txt_InDueFormDate.getText();
            String schoolage=("" + schoolage_cb.getSelectedItem());
            String enterdate=txt_enterdate.getText();
            String remark=remark_ta.getText();
            String sex=("" + sex_cb.getSelectedItem());
            String marriage=(""+marriage_cb.getSelectedItem());
            String estate=("" + estate_cb.getSelectedItem());
            String politicsVisage=("" + politicsVisage_cb.getSelectedItem());
            String supdate = "";
    if(Database.dbms.equals("SQL Server")){
```


```
        supdate="update EmployeeInformation set E_Name ='"+ name +"',E_Sex='"+ sex +"',"
+"E_BornDate='"+ borndate +"',E_Marriage='"+ marriage +"',E_PoliticsVisage='"+
politicsVisage +"'," +"E_SchoolAge='"+ schoolage+"',E_EnterDate='"+ enterdate
+"',E_InDueFormDate='"+ indueformdate +"',"+ "D_Number="+ jComboBoxCode.getSelectedItem()
+",E_Headship='"+ headship +"',E_Estate='"+ estate +"'," + "E_Remark='"+ remark +"' where
E_Number='"+ txt_number.getText() +"'";}
    if(Database.dbms.equals("Oracle")){
        supdate="update EmployeeInformation set E_Name ='"+ name +"',E_Sex='"+ sex +"',"
+"E_BornDate=to_date('"+ borndate +"','YYYY-MM-DD'),E_Marriage='"+ marriage
+"',E_PoliticsVisage='"+ politicsVisage +"'," +"E_SchoolAge='"+ schoolage+
"',E_EnterDate=to_date('"+ enterdate+"','YYYY-MM-DD'),E_InDueFormDate=to_date('"+
indueformdate +"','YYYY-MM-DD'),"+"D_Number="+ jComboBoxCode.getSelectedItem()
+",E_Headship='"+ headship +"',E_Estate='"+ estate +"'," +"E_Remark='"+ remark +"' where
E_Number='"+ txt_number.getText() +"'";}
    if(Database.dbms.equals("MySQL")){
        supdate="update EmployeeInformation set E_Name ='"+ name +"',E_Sex='"+ sex +"',"
+"E_BornDate='"+ borndate +"',E_Marriage='"+ marriage +"',E_PoliticsVisage='"+
politicsVisage +"'," + "E_SchoolAge='"+ schoolage+"',E_EnterDate='"+ enterdate
+"',E_InDueFormDate='"+ indueformdate +"',"+ "D_Number="+ jComboBoxCode.getSelectedItem()
+",E_Headship='"+ headship +"',E_Estate='"+ estate +"'," + "E_Remark='"+ remark +"' where
E_Number='"+ txt_number.getText() +"'";}
    if(Database.dbms.equals("PostgreSQL")){
        supdate="update EmployeeInformation set E_Name ='"+ name +"',E_Sex='"+ sex +"',"
+"E_BornDate='"+ borndate +"',E_Marriage='"+ marriage +"',E_PoliticsVisage='"+
politicsVisage +"'," +"E_SchoolAge='"+ schoolage+"',E_EnterDate='"+ enterdate
+"',E_InDueFormDate='"+ indueformdate +"',"+ "D_Number="+ jComboBoxCode.getSelectedItem()
+",E_Headship='"+ headship +"',E_Estate='"+ estate +"'," + "E_Remark='"+ remark +"' where
E_Number='"+ txt_number.getText() +"'"; }
    try{
        if(Database.executeUpdate(supdate)!=0){
            new JOptionPane().showMessageDialog(null,"数据修改成功！");
            String sqll="select
E_Number,E_Name,E_Sex,E_BornDate,D_Number,E_Marriage,E_Headship,E_InDueFormDate,E_Poli
ticsVisage,E_SchoolAge,E_EnterDate,E_Estate,E_Remark from EmployeeInformation";
            rs= Database.executeQuery(sqll);
        }
    }
    catch(Exception eupdate){}}
    });
    //为删除按钮添加事件
    delet_bt.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent e){
            String sdelete = "delete from EmployeeInformation where E_Number ='"+
txt_number.getText()+"'";
            try{
             if(Database.executeUpdate(sdelete)!=0){
               new JOptionPane().showMessageDialog(null,"数据删除成功！");
               String sql="select
E_Number,E_Name,E_Sex,E_BornDate,D_Number,E_Marriage,E_Headship,E_InDueFormDate,E_Poli
ticsVisage,E_SchoolAge,E_EnterDate,E_Estate,E_Remark from EmployeeInformation";
                rs= Database.executeQuery(sql);
```

```
                rs.next();
                txt_number.setText("" + rs.getInt("E_Number"));
                txt_name.setText(rs.getString("E_Name"));
                if(rs.getString("E_Sex").equals("男")){
                   sex_cb.setSelectedIndex(0);
                }
                else{
                  sex_cb.setSelectedIndex(1);
                }
                txt_borndate.setValue(rs.getDate("E_BornDate"));
                tdepartment.setText(rs.getString("D_Number"));
                ComboBoxCode.setSelectedItem(rs.getString("D_Number"));
               jComboBox.setSelectedIndex(jComboBoxCode.getSelectedIndex());
              if(rs.getString("E_Marriage").equals("未婚")){
                   marriage_cb.setSelectedIndex(0);
                }
                else if(rs.getString("E_Marriage").equals("已婚")){
                   marriage_cb.setSelectedIndex(1);
                }
                else{
                   marriage_cb.setSelectedIndex(2);
                }
                headship_cb.setSelectedItem(rs.getString("E_Headship"));
                txt_InDueFormDate.setValue(rs.getDate("E_InDueFormDate"));
                if(rs.getString("E_PoliticsVisage").equals("党员")){
                   politicsVisage_cb.setSelectedIndex(0);
                }
                else{
                   politicsVisage_cb.setSelectedIndex(1);
                }
                schoolage_cb.setSelectedItem(rs.getString("E_SchoolAge"));
                txt_enterdate.setValue(rs.getDate("E_EnterDate"));
                if(rs.getString("E_Estate").equals("在职")){
                   estate_cb.setSelectedIndex(0);
                }
                else if(rs.getString("E_Estate").equals("停薪留职")){
                   estate_cb.setSelectedIndex(1);
                }
                else{
                   estate_cb.setSelectedIndex(2);
                }
                remark_ta.setText(rs.getString("E_Remark"));
             }
          }
          catch(Exception er){ System.out.println(er); }
       }
});
…… // 其它事件略
Dimension screenSize = Toolkit.getDefaultToolkit().getScreenSize();
setBounds((screenSize.width-658)/2, (screenSize.height-607)/2, 558, 455);
this.setClosable(true);
```


```
        this.setMaximizable(true);
        setVisible(true);
    }
    private JButton save_bt;
    ……// 其它界面元素定义略
}
```

运行界面如下图 15-7 所示。

图 15-7 员工基本信息管理操作界面

（3）查询与统计类模块界面

```
//部门查询类
package qxz;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.table.*;
import java.util.*;
import java.sql.*;
public class DIQ extends JInternalFrame{
    JLabel lbl=new JLabel("部 门 信 息 查 询");
    JLabel lb2=new JLabel("部门编号：");
    JLabel lb3=new JLabel("部门名称：");
    JTextField setxtid=new JTextField(10);
    JTextField setxtname=new JTextField(10);
    JButton btn1=new JButton("查询");
    JButton btn2=new JButton("统计人数");
    private static ResultSet rs;
    JTable table;
    DefaultTableModel dtm;
    String columns[] = {"部门编号","部门名称","部门人数"};
    public DIQ(){
        setTitle("部 门 信 息 查 询");
        dtm = new DefaultTableModel();
```

```
        table = new JTable(dtm);
        JScrollPane sl = new JScrollPane(table);
        dtm.setColumnCount(3);
        dtm.setColumnIdentifiers(columns);
        getContentPane().setLayout(null);
        lb1.setBounds(200,10,300,30);  lb1.setFont(new Font("宋体",Font.BOLD,24));
        getContentPane().add(lb1);
        Font f=new Font("宋体",Font.PLAIN,12);
        lb2.setBounds(10,60,80,25); lb2.setFont(f);getContentPane().add(lb2);
        setxtid.setBounds(80,60,80,23); setxtid.setFont(f);
         getContentPane().add(setxtid);
        lb3.setBounds(10,90,80,25); lb3.setFont(f);getContentPane().add(lb3);
       setxtname.setBounds(80,90,80,23);setxtname.setFont(f);
       getContentPane().add(setxtname);
       btn2.setBounds(20,130,60,25);btn2.setFont(f); getContentPane().add(btn2);
       btn1.setBounds(90,130,60,25);btn1.setFont(f); getContentPane().add(btn1);
        sl.setBounds(180,60,425,290);getContentPane().add(sl);
      //设置边框
    setxtid.setBorder(BorderFactory.createLineBorder(Color.black));
    setxtname.setBorder(BorderFactory.createLineBorder(Color.black));
      btn1.setBorder(BorderFactory.createRaisedBevelBorder());
      btn2.setBorder(BorderFactory.createRaisedBevelBorder());
       sl.setBorder(BorderFactory.createLineBorder(Color.black));
      //为表格初使化数据
       String csf="select * from DepartmentInformation";
       rs=Database.executeQuery(csf);
       if(Database.recCount(rs)>0){
       try{
            while(rs.next()){
                 String num = ("" + rs.getInt("D_Number"));
                 String name = rs.getString("D_Name");
                 String count = rs.getString("D_Count");
                 Vector v=new Vector();
                 v.add(num);v.add(name);v.add(count);
                 dtm.addRow(v);
            }
       }catch(Exception ecsf){System.out.println("初使化表格数据出错!");}
       }
     //为查询按钮加事件
     btn1.addActionListener(new ActionListener(){
       public void actionPerformed(ActionEvent e){
            String sql;
            int rc=dtm.getRowCount();
            for(int i=0;i<rc;i++){
               dtm.removeRow(0);
            }
if(setxtid.getText().equals("")&&setxtname.getText().equals("")){
                 sql="select * from DepartmentInformation";
            }
            else if(setxtname.getText().equals("")){
                 sql = "select * from DepartmentInformation where D_Number = '" +
```


```
setxtid.getText() +"'";
                }
                else{
                     sql = "select * from DepartmentInformation where D_Number = '" +
setxtid.getText() +"' or D_Name like '%"+ setxtname.getText() +"%'";
                }
                rs=Database.executeQuery(sql);
                if(Database.recCount(rs)>0){
                     try{
                          while(rs.next()){
                               String num = ("" + rs.getInt("D_Number"));
                               String name = rs.getString("D_Name");
                               String count = rs.getString("D_Count");
                               Vector v=new Vector();
                               v.add(num);v.add(name);v.add(count);
                               dtm.addRow(v);
                          }
                     }
                     catch(Exception ee){}
                }
           }
           });
          //为“统计人数”按钮添加事件
          btn2.addActionListener(new ActionListener(){
           public void actionPerformed(ActionEvent e){
                String sql;
                Database.executeUpdate("update DepartmentInformation set D_Count=(select
count(*) from EmployeeInformation e where e.D_Number=DepartmentInformation.D_Number)");
                int rc=dtm.getRowCount();
                for(int i=0;i<rc;i++){ dtm.removeRow(0); }
                if(setxtid.getText().equals("")&&setxtname.getText().equals("")){
                   sql="select * from DepartmentInformation";
                }
                else if(setxtname.getText().equals("")){
                   sql = "select * from DepartmentInformation where D_Number = '" +
setxtid.getText() +"'";
                }
                else{
                   sql = "select * from DepartmentInformation where D_Number = '" +
setxtid.getText() +"' or D_Name like '%"+ setxtname.getText() +"%'";
               }
                rs=Database.executeQuery(sql);
                if(Database.recCount(rs)>0){
                     try{
                          while(rs.next()){
                               String num = ("" + rs.getInt("D_Number"));
                               String name = rs.getString("D_Name");
                               String count = rs.getString("D_Count");
                               Vector v=new Vector();
                               v.add(num);v.add(name);v.add(count);
                               dtm.addRow(v);
```

```
                    }
                }
                catch(Exception ee){}
            }
        });
        setSize(630,400);
        this.setClosable(true);
        setVisible(true);
    }
}
```

运行界面如下图 15-8 所示：

图 15-8 员工基本信息管理操作界面

3、Java 常用方法

1. 获取字符串的长度：s.length()

2. 比较两个字符串:s1.equals(String s)、int s1.compareTo(String anotherString)

3. 把字符串转化为相应的数值：int 型 Integer.parseInt(字符串)、Integer.valueOf(my_str).intValue()、long 型 Long.parseLong(字符串)、float 型 Folat.valueOf(字符串).floatValue()、double 型 Double.valueOf(字符串).doubleValue()。

4. 将数值转化为字符串：String.valueOf(数值)、Integer.toString(i)

5. 将字符串转化为日期：java.sql.Date.valueOf(dateStr)

6. 将日期转化为字符串：java.sql.Date.toString()

7. 字符串检索：s1.indexOf(Srting s) 从头开始检索；s1.indexOf(String s ,int startpoint) 从 startpoint 处开始检索，如果没有检索到，将返回-1

8. 得到字符串的子字符串：s1.substring(int startpoint) 从 startpoint 处开始获取；s1.substring(int start,int end) 从 start 到 end 中间的字符。

9. 替换字符串中的字符,去掉字符串前后空格：replace(char old,char new) 用 new 替换 old；s1.trim()；s1.replaceAll(String sold,String snew)。

10. 分析字符串：StringTokenizer(String s) 构造一个分析器，使用默认分隔字符（空格，换行，回车，Tab，进纸符)；StringTokenizer(String s,String delim) delim 是自己定义的分隔符

11. 文本框：TextField(String s) 构造文本框，显示 s；setText(String s) 设置文本为 s；getText() 获取文本 ；setEchoChar(char c) 设置显示字符为 c；setEditable(boolean) 设置文本框是否可以被修改 ；addActionListener() 添加监视器 ；removeActionListener() 移去监视器。

12. 按钮 ：Button() 构造按钮 ；Button(String s) 构造按钮，标签是 s ；setLabel(String s) 设置按钮标签是 s ；getLabel() 获取按钮标签 ；addActionListener() 添加监视器 ；removeActionListener() 移去监视器。

13.标签:Label() 构造标签;Label(String s) 构造标签,显示 s;Label(String s,int x) x 是对齐方式，取值：Label.LEFT、Label.RIGHT、Label.CENTER；setText(String s) 设置文本 s ；getText() 获取文本；setBackground(Color c) 设置标签背景颜色；setForeground(Color c) 设置字体颜色。

14.类型及其转换：Java 基本类型有以下四种：1）int 长度数据类型有：byte(8bits)、short(16bits)、int(32bits)、long(64bits)；2）float 长度数据类型有：单精度（32bits float)、双精度（64bits double)；3）boolean 类型变量的取值有：ture、false；4）char 数据类型有：unicode 字符,16 位。

对应的类类型：Integer、Float、Boolean、Character、Double、Short、Byte、Long；从低精度向高精度转换 byte、short、int、long、float、double、char，类型转换举例：

（1）int i=Integer.valueOf(“123”).intValue()

说明：上例是将一个字符串转化成一个 Integer 对象，然后再调用这个对象的 intValue()方法返回其对应的 int 数值。

（2）float f=Float.valueOf(“123”).floatValue() 本例是将一个字符串转化成一个 Float 对象，然后再调用这个对象的 floatValue()方法返回其对应的 float 数值。

（3）double d=Double.valueOf(“123”).doubleValue() 本例是将一个字符串转化成一个 Double 对象，然后再调用这个对象的 doubleValue()方法返回其对应的 double 数值。

（4）int i=Integer.parseInt(“123”) 此方法只能适用于字符串转化成整型变量

（5）float f=Float.valueOf(“123”).floatValue() 本例是将一个字符串转化成一个 Float 对象，然后再调用这个对象的 floatValue()方法返回其对应的 float 数值。

（6）long l=Long.valueOf(“123”).longValue() 本例是将一个字符串转化成一个 Long 对象，然后再调用这个对象的 longValue()方法返回其对应的 long 数值。

15. 获取记录集记录字段的值：i=rs.getInt(编号字段);s1=rs.getString(名称字段);

15.1.6 测试运行和维护

1、系统运行与维护

经测试，系统功能运行良好。虽然在不同操作系统中系统运行方式有所不同，但系统在多种操作系统下都能正常运行，可见本系统的兼容性是不错的。这里说明两个操作系统平台下的运行方式：

（1）Windows XP/Windows 7/Windows 8：直接双击“qxz.jar”文件包（下文将说明其如何制作）即可运行，前提是先附加数据库，而且建立数据源（若直接使用 JDBC 驱动可不必建数据源的）。

（2）windows 2000：2000 下不能直接运行 jar 文件，在附加数据库，而且建立数据源之后，打开 MS-DOS 命令窗体，改变当前目录到“qxz.jar”文件所在的目录，运行如下命令即可：java -jar qxz.jar

维护阶段最主要需要保存好最新的数据库文件，可以定期周期性做好系统的备份。

系统编码完成后，要经过反复调试、测试与试用运行后，才能正式交付企业使用。

2、系统的相关文件及如何制作 jar 文件包

下面来补充说明本系统的相关文件及如何制作 jar 文件包。

（1）本系统的文件组成

本系统在“Eclipse SDK Ver 3.3.2”集成环境下编辑、调试与运行的。通过新建项目来组织系统文件，如图 15-9 所示。左边子窗体呈现了项目“yuangong2”及其所包含的系统组成部分：1）“image”目录存放系统使用的图形图象文件；2）“qxz”目录存放系统所有 Java 源程序及其编译产生的“class”目标文件；3）“JRE System Library[jdk1.5.0_15]”引用的系统库文件；4）“Referenced Libraries”引用的其它库文件；5）“sqlserver20002005jdbc”存放连接 SQL Server 2000/2005 的 JDBC 库文件目录；6）“oraclejdbc”存放连接 Oracle 数据库的 JDBC 库文件目录；7）“mysql-connector”存放连接 MySQL 的 JDBC 库文件目录；8）“postgresql-jdbc”存放连接 PostgreSQL 的 JDBC 库文件目录；9）Config.ini 系统配置文件；10）其它相关系统文件。

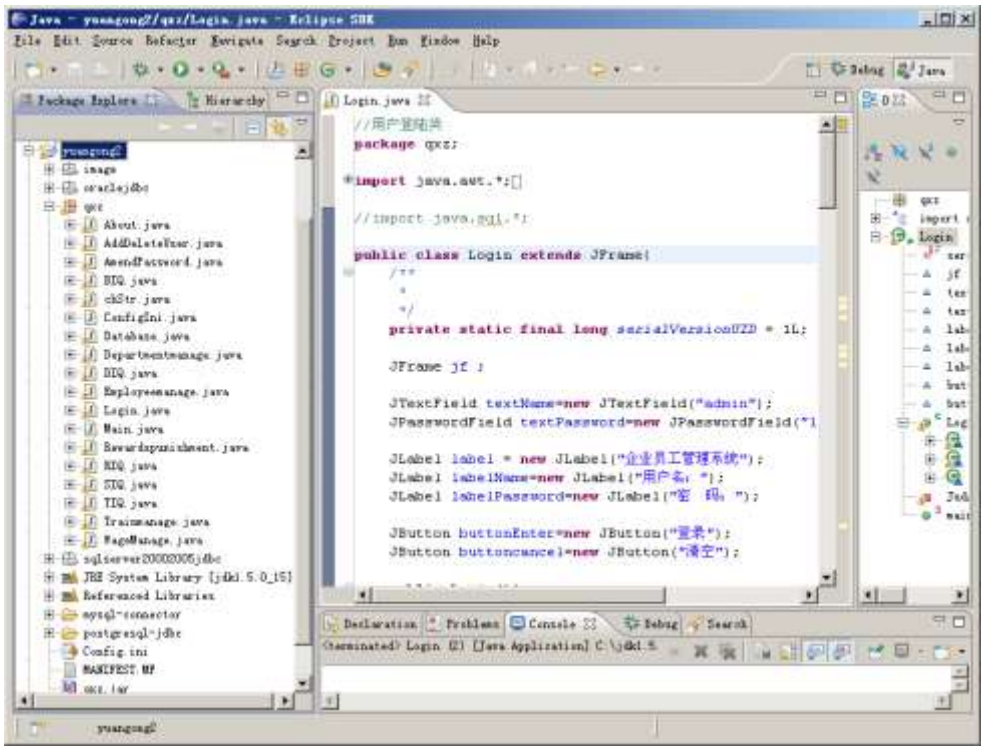


图 15-9 本系统的 Eclipse 集成开发环境

（2）如何制作本系统的 jar 文件包

制作可执行的 jar 文件包要利用 jar 命令。jar 命令文件一般位于 Java jdk 安装目录的“bin”子目录中，如在“C:\jdk1.5.0_15\bin”中。在 DOS 窗口中运行不带参数的 jar 命令能得到命令参数的说明（注意：运行前应通过 set path 命令设置路径，如：set path=C:\jdk1.5.0_15\bin），这里不再展开，只对制作本系统的 jar 文件包举例说明，命令有：

1）jar -cvfm qxz.jar MANIFEST.MF qxz*.* image*.* Config.ini sqlserver20002005jdbc\msbase.jar sqlserver20002005jdbc\mssqlserver.jar sqlserver20002005jdbc\msutil.jar sqlserver20002005jdbc\sqljdbc.jar mysql-connector\mysql-connector-java-5.0.8-bin.jar oraclejdbc\classes12.jar postgresql-jdbc\postgresql-8.2-506.jdbc3.jar

以上命令把系统所有相关文件压缩制作到 qxz.jar 文件包中。

2）jar -cvfm qxz.jar MANIFEST.MF qxz*.class

以上命令只把所有系统程序的 class 字节码文件压缩制作到 qxz.jar 文件包中。

以上两命令中使用到清单文件 MANIFEST.MF（为文本文件），其内容如下：

```
Manifest-Version: 1.0
Created-By: 1.5.0_15 (Sun Microsystems Inc.)
Class-Path: sqlserver2000&2005jdbc\msbase.jar sqlserver20002005jdbc\mssqlserver.jar
sqlserver20002005jdbc\msutil.jar sqlserver20002005jdbc\sqljdbc.jar
mysql-connector\mysql-connector-java-5.0.8-bin.jar oraclejdbc\classes12.jar
postgresql-jdbc\postgresql-8.2-506.jdbc3.jar
Main-Class: qxz.Login
```

该文件中的以上内容主要指定了引用到的 JDBC 类库及系统的主类为 qxz.Login（即：

qxz 目录中 Login.class 中的 Login 类）。

说明：以上两种情况，运行时 qxz.jar 所在的目录中都需要有“image”目录及其文件、“sqlserver2000&2005jdbc”目录及其文件，“sqlserver2000&2005jdbc”目录及其文件，“sqlserver2000&2005jdbc”目录及其文件，及 Config.ini 系统配置文件。

3）jar -cvfm qxz.jar MANIFEST2.MF qxz*.class com*.*

以上命令把所有系统程序的 class 字节码文件及 com 子目录下的所有类文件（是把 SQL Server 数据库的 JDBC 类库释放后得到的，就是 JDBC 类文件）压缩制作到 qxz.jar 文件包中。其中命令中的清单文件 MANIFEST2.MF（为文本文件），其内容可简单为：

```
Manifest-Version: 1.0
Created-By: 1.5.0_15 (Sun Microsystems Inc.)
Main-Class: qxz.Login
```

这样运行 qxz.jar 时，其所在的目录中不再需要“sqlserver20002005jdbc”目录及其类库文件了。

说明：其它 JDBC 类库，也可以释放成文件夹及文件后，直接压缩到系统文件包中，这样运行时就不再需要相应目录及其类库文件了。

15.2 企业库存管理及 Web 网上订购系统(C#/ASP.NET 技术)

企业库存管理子系统，往往是企业众多管理子系统中企业物资供应管理子系统或企业产品销售管理子系统的核心模块。有的企业在管理系统规划设计时，根据企业管理的现状或重点对仓库管理的需要，专门设置仓库管理子系统，实际上核心管理内容主要是对出入仓库的各类物品的管理或者说是对仓库中物品库存的有效管理。

天辰冷拉型钢有限公司是无锡的小型钢铁加工企业，本案例介绍的企业库存管理系统的原型就来自该企业，该企业以钢铁产品的物理加工如拉伸、压制、锻造等为主，为此，企业的加工原料与生产产品的描述属性相似，企业在原料（即坯料）采购与产品销售中原根据手工制作的 Excel 表格来管理库存数据。希望开发库存管理子系统能对原料与产品的库存计算机自动管理，原料采购与产品销售中能实时获取库存信息，以利于更有效开展企业活动。

Web 网上订购系统是企业为适应不断发展着的 Internet 上电子商务活动的需要。通过 Web 网页方式，企业能更好地宣传自己，扩大影响力，能方便快捷地开展网上产品销售活动。

企业库存管理与 Web 网上订购系统对整个企业管理信息系统来说是较小范围的局部系统，然而，它较具典型性与实用性，把它应用于本书，用来介绍数据库应用系统的设计与开发是较适合的，因为简单小系统能让初学者更容易了解与把握系统全貌，学习与借鉴系统的分析、设计与实现，更能说明问题，章节的篇幅也有限可控。

15.2.1 开发环境与开发工具

天辰冷拉型钢有限公司内部已有局域网，网络中有若干配置较高的台式机可以用作服务器，服务器上安装 SQL Server 2014/2012/2008/2005/2000，其中有一台服务器能以 ADSL 方式宽带上网并安装有 IIS Web 服务器。服务器或各部门的客户机都安装了各种类型的 Windows 操作系统，一般还安装了如 Word、Excel 等 OFFICE 软件。

为此，开发设计的库存管理子系统，首先是基于局域网的客户机/服务器系统（C/S 模式），支持企业信息集中存放在 SQL Server 数据库中，承担数据服务器功能，使用系统的客户机上安装有将开发设计出的库存管理子系统，多客户机同时共享使用服务器中的库存系统数据。随着 Internet 上企业商务活动的广泛开展，本 C/S 模式的库存管理子系统可以容易地扩展成支持 B/S 模式的 Internet 上的商务系统，实际上我们也确实这样做了。因为未来的企业管理系统往往是 C/S、B/S、基于 Web 服务等模式共存的系统，如图 15-10 所示。

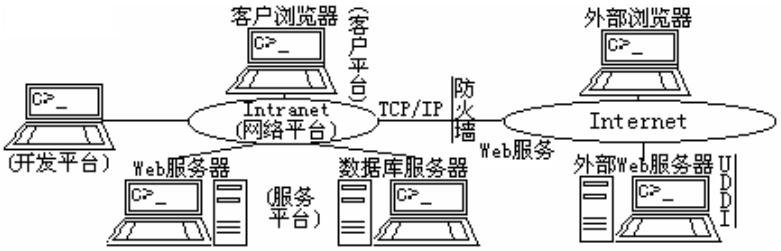


图 15-10 C/S、B/S、基于 Web 服务等模式共存的系统示意图

本子系统可以使用 Visual C# 2013 与 ASP.NET 2013 或 Visual Basic 6.0 与 ASP 等开发工具开发与运行，系统能在企业内部局域网上共享使用，库存查询与网上订购功能的网页发布到 Web 服务器上能支持在 Internet 上使用 Web 网上订购系统。

本书对本系统的实现介绍是基于 2013 版本的 C#与 ASP.NET 的，本书相关资料中还提供了本系统采用 Visual Basic 6.0 与 ASP 功能实现的源程序，读者可一并学习参考。

15.2.2 系统需求分析

经过调查，对企业库存管理和 Web 网上订购的业务流程进行分析。能知库存的变化通常是通过入库、出库操作来进行。系统对每个入库操作均要求用户填写入库单，对每个出库操作均要求用户填写出库单，网上订购则更直接，通过订购系统在网上直接下单。在完成出入库操作的同时，可以进行增加、删除和修改数据记录等操作。用户可以随时进行各种查询、统计、报表打印、账目核对等工作。另外，需要时也可以用图表等形式来反映查询结果。

在使用本系统之前，企业通过手工维护 Excel 表格来管理原料与产品库存的数据。但是在使用中遇到很多问题，如：①文件级共享，共享性差，安全性低；②实时性差，Excel 表中的内容只有及时保存后，其它电脑才能读到，另外，不能允许两个以上的人同时更新库存文件；③查询、统计等操作不方便；④根本不能实现 Web 网上订购功能；……。

在充分了解原 Excel 工作模式，多次深入询问调研后，基本了解了企业就库存管理及网上订购系统对数据与处理的需求。

本系统的主要要处理的数据有：产品与坯料的入、出库信息；产品与坯料的实时库存信息；产品与坯料月明细库存信息（如包括每产品每天的入出库信息）；产品与坯料月区段统计表（包括累计月初值、月入库、月出库、月末库存值等情况）；产品与坯料月末累计统计表（包括累计入库、累计出库、月末库存值等情况）；模具库存信息。网上订购需要有：用户一次订购信息包括订购明细信息；月份的设定信息（如某月从某日到某日的信息等）；其它还包括从安全性与权限控制考虑的各级别用户信息等。总体上而言，输入入出库信息后，能得到库存、各种统计、汇总、分类信息等，Web 用户能查阅库存信息，决定网上订购量等。

基于以上系统涉及的处理数据，C/S 模式实现的库存管理系统具体涉及到：①能方便及时多用户地录入产品、坯料、模具等入出库单数据；②能方便查阅、核对入出库单数据，并能方便维护产品、坯料、模具等入出库单原始数据；③能以组合方式快速查阅产品、坯

料、模具等入出库单原始数据；④能按一键完成对库存、按月或分日对产品、坯料的统计；⑤能自动产生产品或坯料的实时库存；⑥能以树型结构或表格方式方便查阅各类各种产品或坯料的实时库存；⑦能由分类统计值，反查其明细清单；⑧能把主要表或查询信息按需导出到 Excel 中，支持原有手工处理要求，导出到 Excel 的数据能用于保存或排版打印等需要；⑨分级别用户管理；⑩月份设定与统计管理等等。

B/S 模式实现的网上订购系统的具体处理与数据主要有：①能实现网上用户的注册与登录，登录用户的管理；②能方便查阅（如分页查询）产品及库存信息，方便产品选购；③能实现基本的购物车功能；④能完成订购、实现网上支付过程，并自动产生订购明细数据，产生产品 Web 销售对应的出库记录；自动更改产品库存；⑤事后能查阅自己的历史订单及明细数据；⑥具有商务网站的基本功能，如：网站公告、系统简介、自己的用户信息维护、找回密码、联系我们、友情链接等等。

C/S 与 B/S 两类系统共用同一个数据库，数据间紧密依赖、密切关联与联动，数据库则集中存放在企业服务器上的 SQL Server 2014/2012/2008/2005/2000 数据库管理系统中。

1、系统数据流图

在仔细分析调查有关信息需要的基础上，能得到库存管理之产品库存管理系统的基本模型图如图 15-11。产品库存管理系统的功能级(1 级)数据流图如图 15-12（坯料或原料库存系统的功能级数据流图略，请读者参照完成）。对图 15-11 中的“处理事务”分解后的 2 级数据流图如图 15-13。

Web 网上订购系统的基本模型图如图 15-14，系统的功能级(1 级)数据流图如图 15-15。对图 15-15 中的“网上订购”分解后的 2 级数据流图如图 15-16。

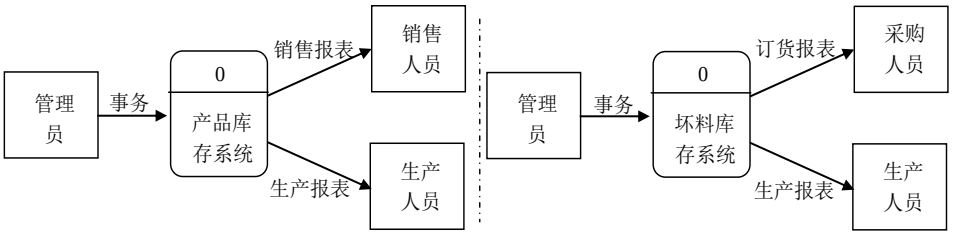


图 15-11 坯料与产品库存系统的基本系统模型

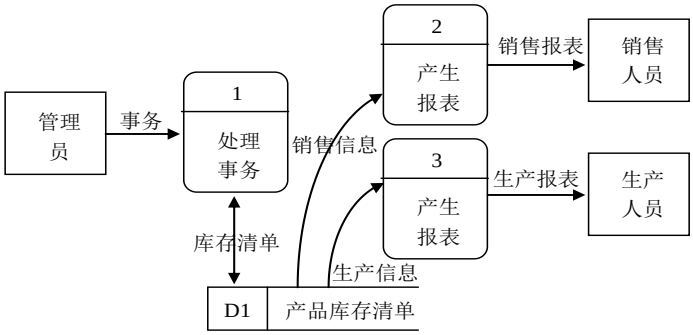


图 15-12 产品库存系统的功能级数据流图

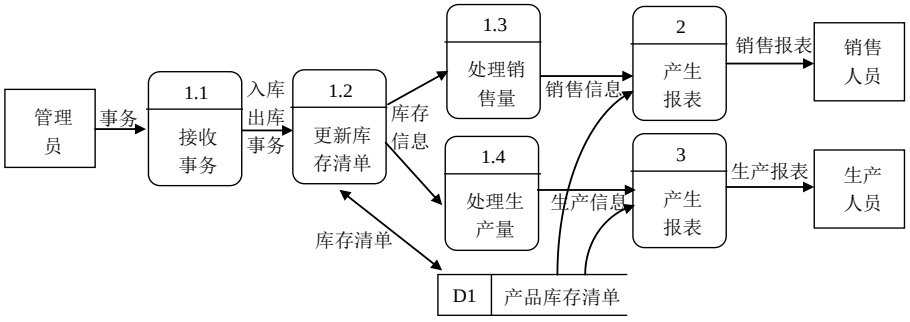


图 15-13 产品库存系统中“处理事务”分解后的数据流图

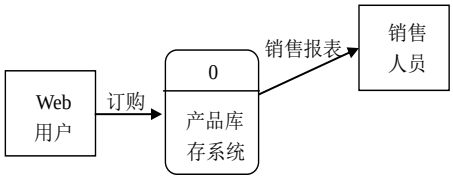


图 15-14 Web 网上订购系统的基本系统模型

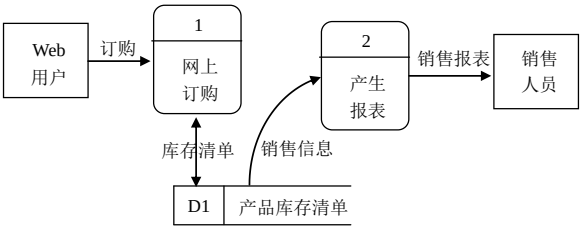


图 15-15 Web 网上订购系统的功能级数据流图

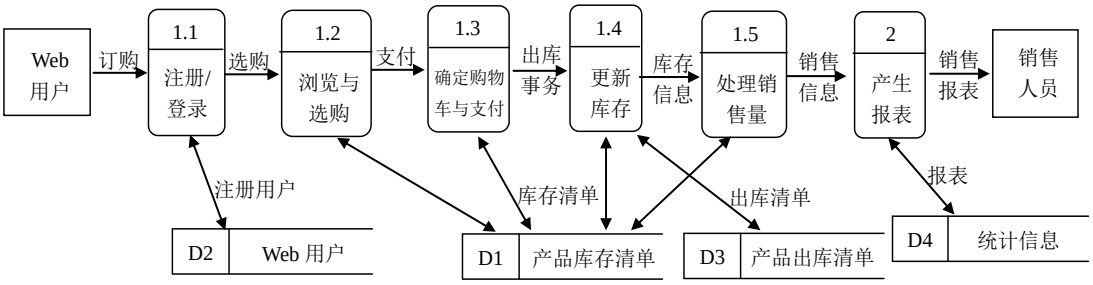


图 15-16 Web 网上订购系统中“网上订购”分解后的数据流图

2、系统数据字典

数据流图表达了数据和处理的关系，数据字典则是系统中各类数据描述的集合，是进行详细的数据收集和数据分析所获得的主要成果。数据字典通常包括数据项、数据结构、数据流、数据存储和处理过程五个部分。以下数据字典卡片的形式来举例说明。

（1）“产品入库单”数据结构：

名字：产品入库单

别名：产品生产量

描述：每天生产或加工车间，以入库单形式来记录其产量，并登记入库

定义：产品入库单=入库单号+大类+规格+材质+单位+生产车间+成本+
日期+入库值+经办人

位置：保存到入出库表或打印保存

（2）“产品入库单”数据结构之数据项：

表 15-1 “入库单号”数据项

名字：入库单号
别名：顺序号
描述：唯一标识某产品入库的数字编号
定义：整型数
位置：产品入库表、产品入出库表

表 15-2 “大类”数据项

名字：大类
别名：产品大类名
描述：产品的第一大分类名
定义：字符型汉字名称，汉字数<=3
位置：产品入库表、产品入出库表、产品库存表、各统计表

…… 其它数据项的定义略。

（3）数据流

数据流是数据结构在系统内传输的路径。前面已画出的数据流图能较好地反映出数据的前后流动关系，除此外还能描述为（以“入库单数据流”来说明）：

数据流名：入库单数据流　　说明：“产品入库单”数据结构在系统内的流向

数据流来源：管理员接收事务　　数据流去向：库存处理事务

平均流量：每天几十次　　高峰期流量：每天上百次

（4）数据存储

数据存储是数据结构停留或保存的地方，也是数据流的来源和去向之一。它可以是手工文档或手工凭单，也可以是计算机文档。对数据存储的描述通常包括（以入库表数据存储来说明）：

数据存储名：入库表

说明：入库单数据，作为原始数据需要保存与备查

编号：入库单为唯一标识，顺序整数，从 1 开始每次增加 1

输入的数据流：入库单数据流，来自生产车间

输出的数据流：出库单数据流，用于销售部门销售

数据结构：“产品入库单”、“产品出库单”、“产品库存”

数据量：一天，100*100=10000 字节

存取频度：每小时存取更新 10-20 次，查询>=100 次

存取方式：联机处理、检索与更新、顺序检索与随机检索

（5）处理过程

处理过程的具体处理逻辑一般用判定表或判定树来描述。数据字典中只需要描述处理过程的说明性信息。如“实时产品库存计算”的处理过程说明如下：

处理过程名：实时产品库存计算

说明：随着入库单、出库单的不断输入，要能实时计算出当前各产品的库存

输入：入库单数据流，来自生产车间；出库单数据流，来自销售部门销售

输出：计算出各产品当前库存

处理：产品库存计算的功能就是实时计算产品库存，处理频度：每小时 20-40 次，每当有入库单数据流或出库单数据流发生都要引发库存计算事务，计算库存涉及的数据：每小时 4kb-10kb，希望在发生入库或出库信息时实时计算库存。

以上通过几个例子说明了数据字典的基本表示方法，只是起到引导的作用。完整、详

尽的系统数据字典是在需求分析阶段，充分调研、分析、讨论的基础上建立，并将在数据库设计过程中不断修改、充实、完善的，它是数据库应用系统良好设计与实现的基础与保障。

3、本系统需要管理的实体信息

⑴Web 订单：顺序号、订单号、订单日期、订购总额、支付方式、确认标志、地址、email 地址、备注等；⑵Web 用户表：用户编号、用户名、口令、email 地址、地区、地址、邮编、ＱＱ号、电话、用户级别、其它；⑶产品年月设置：年月、起始日期、终止日期、创建标志、生成次数、已结转、已删除等；⑷产品入库单：顺序号、大类、规格、材质、单位、生产车间、成本、日期、入库值、经办人、处理标记等；⑸产品出库单：顺序号、大类、规格、材质、单位、发货去向、单价、日期、出库值、经办人、处理标记等；⑹产品实时库存：大类、规格、材质、产品入库、产品出库、产品库存、图片、图片文件、单价、折扣率、产品说明、顺序号等；⑺坯料年月设置：年月、起始日期、终止日期、创建标志、生成次数、已结转、已删除等；⑻坯料入库单：顺序号、材质、钢号、规格、单位、钢产地、单价、日期、入库值、经办人、处理标记等；⑼坯料出库单：顺序号、材质、钢号、规格、单位、领用车间、单价、日期、出库值、经办人、处理标记等；⑽坯料实时库存：材质、钢号、规格、入库量、出库量、库存量、图片等；⑾模具库存：顺序号、分类、厚度、乘、宽度、库存数量、备注等；⑿系统用户：用户编号、用户姓名、口令、等级等。

4 本系统要管理的实体联系信息

①Web 订单与产品库存间的“Web 订单明细”联系要反映：订单号、产品编号、订购量等；②“月累计库存”联系要反映：年月、大类、规格、产量、销量、产品库存等；③“产品月区段库存”联系要反映：年月、大类、规格、期初值、产量、销量、期末值等；④“月产品明细库存”（不同月份属性个数也不同）联系要反映：年月、大类、规格、材质、单位、发货去向、期初值、期末值、１号、２号、.....３１号等；⑤“坯料累计库存”联系要反映：年月、规格、钢产地、入库量、出库量、库存量等；⑥“坯料月区段库存”联系要反映：年月、规格、钢产地、期初值、入库、出库、期末值等；⑦“坯料月区段库存 2”联系要反映：年月、规格、钢产地、期初值、入库、出库、期末值等；⑧“月坯料明细库存”（不同年月属性个数也不同）联系要反映：年月、材质、钢号、规格、单位、钢产地、期初值、期末值、１号、２号、.....３１号等。

15.2.3 功能需求分析

在数据库服务器如 SQL Server 2014/2012/2008/2005/2000 中，要创建 KCGL 数据库，在数据库上建立各关系模式对应的库表信息，并确定主键、索引、参照完整性、用户自定义完整性等约束要求。

（1）C/S 模式实现的库存管理系统功能需求

⑴能对各原始数据表实现输入、修改、删除、添加、查询、打印等基本操作；⑵能方

便及时多用户地录入产品、坯料、模具等入出库单数据；⑶能方便查阅、核对入出库单数据，并能方便维护产品、坯料、模具等入出库单原始数据；⑷能以组合方式快速查阅产品、坯料、模具等入出库单原始数据；⑸能按一键完成对库存、按月或分日对产品、坯料的统计；⑹能自动产生产品或坯料的实时库存；⑺能以树型结构或表格方式方便查阅各类各种产品或坯料的实时库存；⑻能由分类统计值，反查其明细清单；⑼能把主要表或查询信息按需导出到 Excel 中，支持原有手工处理要求，导出到 Excel 的数据能用于保存或排版打印等需要；⑽分级别用户管理；⑾月份设定与统计管理；⑿高级管理员的管理操作如：系统数据的备份与恢复、系统用户的维护、动态 SQL 命令操作、系统日志查阅等；⒀系统设计成传统的 Windows 多文档多窗口操作界面，要求系统具有操作方便、简捷等特点；⒁用户管理功能，包括用户登录、注册新用户、更改用户密码等功能；⒂其它你认为子系统应有的查询、统计功能；⒃要求所设计系统界面友好，功能安排合理，操作使用方便，并能进一步考虑子系统在安全性、完整性、并发控制、备份恢复等方面的功能要求。

（2）B/S 模式实现的网上订购系统功能需求

①能实现网上用户的注册与登录，登录用户的管理；②能方便查阅（如分页查询）产品及库存信息，方便产品选购；③能实现基本的购物车功能；④能完成订购、实现网上支付过程，并自动产生订购明细数据，产生产品 Web 销售对应的出库记录；自动更改产品库存；⑤事后能查阅自己的历史订单及明细数据；⑥具有商务网站的基本功能，如：网站公告、系统简介、自己的用户信息维护、找回密码、联系我们、友情链接等等；⑦要求 Web 网页系统要运行稳定、可靠，操作简单、方便。

15.2.4 系统设计

1、数据库概念结构设计

数据库在一个信息管理系统中占有非常重要的地位，数据库结构设计的好坏将直接对应用系统的效率以及实现的效果产生影响。合理的数据库结构设计可以提高数据存储的效率，保证数据的完整和一致。同时，合理的数据库结构也将有利于程序的实现。

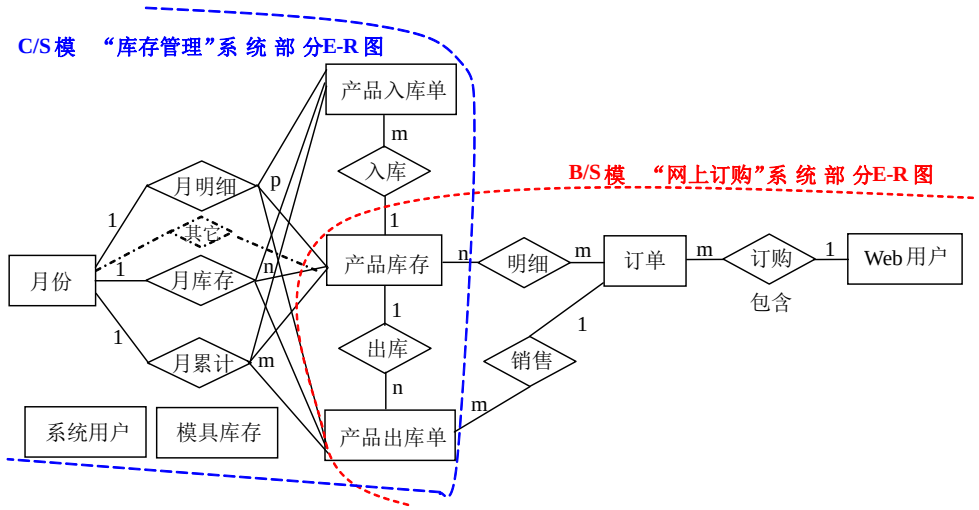
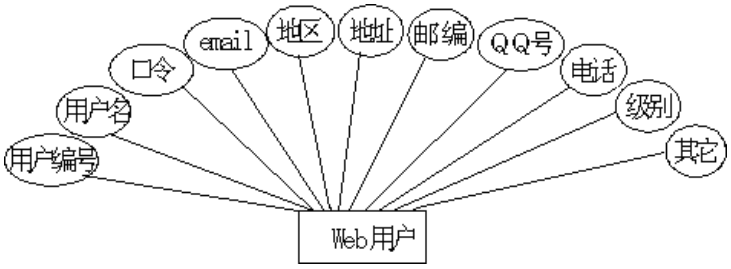
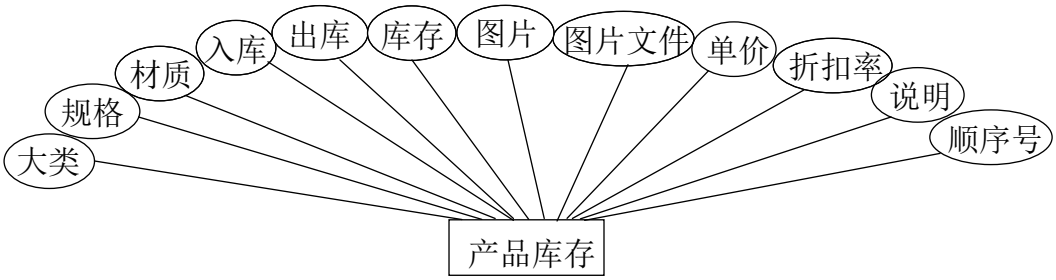


图 15-17 产品库存管理与网上订购系统的实体及其联系图

在充分需求分析的基础上，经过逐步抽象、概括、分析、充分研讨，可画出如下反映产品库存管理与产品网上订购的数据的整体 E-R 图(如图 15-17、图 15-18、图 15-19 所示)。至于原料（坯料）库存管理及其网上订购的数据的 E-R 图部分类似，可以请读者自己参考设计出来，此处略。





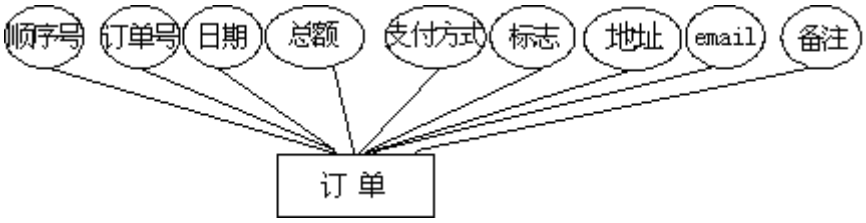


图 15-18 系统部分实体及其属性图

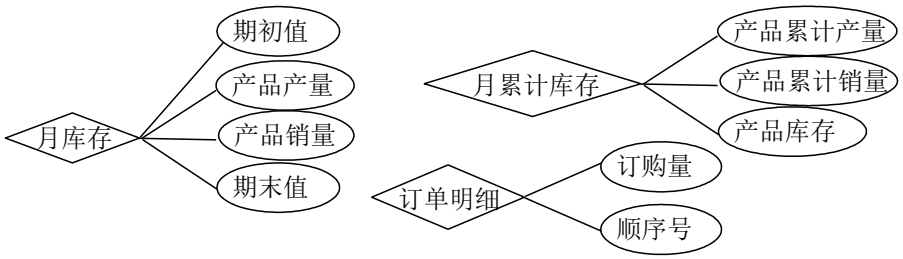


图 15-19 系统部分联系及其属性图

2、系统功能模块设计

对库存管理系统各项功能进行集中、分类，按照结构化程序设计的要求，可得出系统的功能模块图如图 15-20，而 Web 网上订购系统的功能模块如图 15-21 所示。

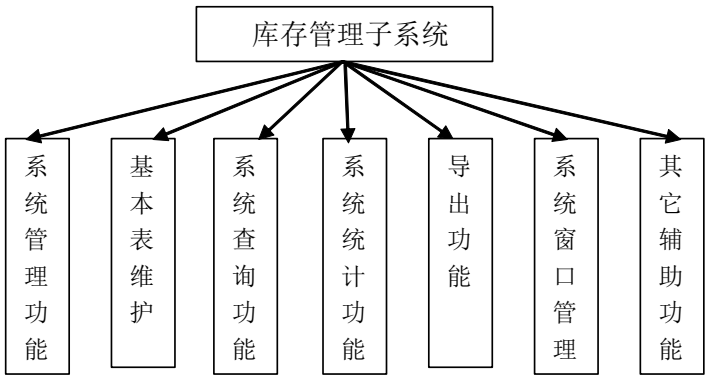


图 15-20 库存管理子系统的一级功能模块图

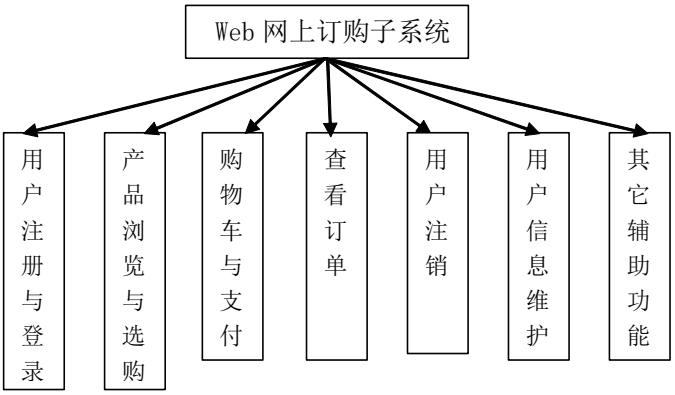


图 15-21 Web 网上订购子系统的一级功能模块图

3、数据库逻辑及物理结构设计

1．数据库关系模式

按照实体-联系图转化为关系模式的方法，本系统共使用到至少 23 个关系模式(含 4 个辅助关系)，这里只给出表名，中文属性名见后“2. 数据库及表结构的创建”。

（1）Web 订单表(weborders)；（2）Web 订单明细表(weborderdetails)；（3）Web 用户表(webuser)；(4)Web 购买折扣表(webdiscount)；(5)Web 支付方式表(Webpaydefault)；（6）Web 即时信息表(webmessage)；（7）产品年月设置表(tccpny)；（8）产品入出库表(tccprck)；（9）产品实时库存表(tccpsskc)；（10）月累计库存表(tccptj)；（11）产品月区段库存表(tccpkctj)；（12）月产品明细库存表(tccpkc200412)；（13）坯料年月设置表(tcplny)；（14）坯料入出库表(tcplrck)；（15）坯料实时库存表(tcplsskc)；（16）坯料累计库存表(tcpltj)；（17）坯料月区段库存表(tcplkctj)；（18）坯料月区段库存表 2(tcplkctj2)；（19）月坯料明细库存表(tcplkc200412)；（20）模具库存表(tcmjkc)；（21）系统用户表(users)；（22）日志表(logs)；（23）系统参数表(tcsyspara)。

转化与设计关系模式的说明：

（1）实体“Web 用户”与实体“订单”间的一对多“订购”联系，通过把“用户编号”加到“订单”实体中而合并到“订单”多方实体。

（2）通用的把一对多联系合并到多方实体的联系还有：“订单”与“产品出库”间的“销售”联系、“产品库存”与“产品入库”间的“入库”、“产品库存”与“产品出库”间的“出库”、“月份”与“产品库存、产品入库或产品出库等”间的“月明细、月库存、月累计等”联系。

（3）“产品入库单”与“产品出库单”两实体属性稍有区别，主要是入库单含生产车间与产品成本，而出库单上是发货去向与销售单价，从简单化与使用单位处理习惯出发，把它们设计成同类属性。这样，我们考虑把产品入库与产品出库合并起来形成一个关系模式，称为“产品入出库表”，其中“入”与“出”主要通过“出入库值”的正负来体现，值为正是入库值，值为负是出库值。同样的情况还有“坯料入出库表”

表名与中文属性名对应的英文名及各表主码属性，参阅下面各表的 T-SQL 创建命令。

2．数据库及表结构的创建

设本系统使用的数据库名为 KCGL，根据已设计出的关系模式及各模式的完整性的要求，现在就可以在 SQL Server 数据库系统中实现这些逻辑结构。下面是创建数据库及以产品相关为主的表结构的 T-SQL 命令(SQL Server 中的 SQL 命令)：

CREATE DATABASE KCGL; USE KCGL;

（1）Web 订单表(weborders)，其属性对应的含义：顺序号,用户编号,订单号,订单日期,订购总额,支付方式,确认标志,地址,email 地址,备注。

```
CREATE TABLE [weborders](
  [ID] [int] IDENTITY(1,1) NOT NULL,
  [userid] [int] NOT NULL,
  [orderid] [varchar](20) NOT NULL,
  [ordertime] [varchar](20) NOT NULL,
  [summoney] [varchar](20) NULL,
  [paymenttype] [varchar](50) NULL,
  [validate] [bit] NULL CONSTRAINT [DF_weborders_comp] DEFAULT ((0)),
  [address] [varchar](50) NULL,
  [email] [varchar](20) NULL,
  [bz] [varchar](500) NULL,
  CONSTRAINT [PK_weborders] PRIMARY KEY CLUSTERED([ID] ASC))
  -- 触发器定义，参见相关章节或系统数据库，此处略，下同。
  CREATE TRIGGER [tr_weborders_d] ON [weborders] AFTER delete AS ……
  CREATE TRIGGER [tr_weborders_i] ON [weborders] AFTER INSERT AS ……
```

（2）Web 订单明细表(weborderdetails)，其属性对应的含义：顺序号,订单号,产品编号,订购量（说明：因篇幅原因，可能要牺牲逐列换行的格式要求，下同）。

```
CREATE TABLE [weborderdetails]([id] [int] IDENTITY(1,1) NOT NULL,
  [orderid] [varchar](20) NOT NULL, [cpid] [int] NOT NULL, [dgl] [decimal](18,3) NULL,
  CONSTRAINT [PK_weborderdetails] PRIMARY KEY CLUSTERED([id] ASC))
  CREATE TRIGGER [tr_weborderdetails_d] ON [weborderdetails] AFTER delete AS ……
  CREATE TRIGGER [tr_weborderdetails_i] ON [weborderdetails] AFTER INSERT AS ……
```

（3）Web 用户表(webuser)，其属性对应的含义：用户编号,用户名,口令,email 地址,地区,地址,邮编,ＱＱ号,电话,用户级别,其它。

```
CREATE TABLE [webuser]([id] [int] IDENTITY(1000,1) NOT NULL,
  [UserName] [varchar](20) NOT NULL,  [password] [varchar](16) NULL,
  [Email] [varchar](20) NULL,[Area] [varchar](10) NULL,[Address] [varchar](50) NULL,
  [PostID] [char](6) NULL,   [QQ] [varchar](15) NULL,  [telphone] [varchar](21) NULL,
  [Ulevel] [tinyint] NULL CONSTRAINT [DF_webuser_Ulevel] DEFAULT((1)),
  [other] [varchar](50) NULL,CONSTRAINT [PK_webuser] PRIMARY KEY CLUSTERED([id] ASC))
  CREATE UNIQUE NONCLUSTERED INDEX [UK_webuser] ON [webuser] ([UserName] ASC)
```

（4）Web 购买折扣表(webdiscount)，其属性对应的含义：顺序号,折扣率,等级,累计金额。

```
CREATE TABLE [webdiscount](
  [ID] [int] IDENTITY(1,1) NOT NULL,   [discount] [decimal](18, 4) NOT NULL,
  [ulevel] [tinyint] NOT NULL CONSTRAINT [DF_webdiscount_ulevel] DEFAULT ((1)),
  [mmoney] [money] NOT NULL CONSTRAINT [DF_webdiscount_mustmoney] DEFAULT ((100000)),
  CONSTRAINT [PK_webdiscount] PRIMARY KEY CLUSTERED([ID] ASC))
```

（5）Web 支付方式表(Webpaydefault)，其属性对应的含义：顺序号,支付类型,支付信息,起用日期,联系人。

```
CREATE TABLE [Webpaydefault]([ID] [int] NOT NULL, [paymenttype] [nvarchar](50) NULL,
  [paymentmessage] [nvarchar](max) NULL,[idate] [nvarchar](50) NULL,
```

```
  [senduser] [nvarchar](50) NULL,
  CONSTRAINT [PK_Webpaydefault] PRIMARY KEY CLUSTERED ([ID] ASC))
```

（6）Web 即时信息表(webmessage)，其属性对应的含义：顺序号,主题,内容,发表日期,发布人。

```
CREATE TABLE [webmessage]([ID] [int] NOT NULL, [subject] [nvarchar](50) NULL,
  [message] nvarchar(100) NULL,[idate] nvarchar(50) NULL, [senduser] nvarchar(50) NULL,
  CONSTRAINT [PK_webmessage] PRIMARY KEY CLUSTERED ([ID] ASC))
```

（7）产品年月设置表(tccpny)，其属性对应的含义：年月,起始日期,终止日期,创建标志,生成次数,已结转,已删除。

```
CREATE TABLE [tccpny]([ny] char(6) NOT NULL, [qsrq] datetime NOT NULL DEFAULT (getdate()),
  [zzrq] [datetime] NOT NULL DEFAULT (getdate()),[cjbz] [char](2) NULL DEFAULT('否'),
  [sccs] [int] NULL DEFAULT (0),[wc] [char](2) NULL DEFAULT ('否'),
  [qc] [char](2) NULL DEFAULT ('否'),
  PRIMARY KEY NONCLUSTERED ([ny] ASC))
```

（8）产品入出库表(tccprck)，其属性对应的含义：顺序号,大类,规格,材质,单位,发货去向,单价,日期,出入库值,经办人,处理标记。

```
CREATE TABLE [tccprck]([id] [int] IDENTITY(1,1) NOT NULL,
  [dl] [char](6) NOT NULL CONSTRAINT [DF__tccprck__dl__0519C6AF] DEFAULT ('圆钢'),
  [gg] [char](30) NOT NULL,       [cz1] [char](10) NOT NULL, [dw] [char](4) NULL,
  [fhqx] [varchar](50) NULL,      [dj] [numeric](18, 3) NULL,
  [rq] [datetime] NULL CONSTRAINT [DF__tccprck__rq__060DEAE8] DEFAULT (getdate()),
  [crkz] [numeric](18, 3) NULL,   [jbr] [char](8) NULL,
  [clbj] [char](1) NULL CONSTRAINT [DF__tccprck__clbj__07020F21] DEFAULT ('0'),
  CONSTRAINT [PK__tccprck__0425A276] PRIMARY KEY CLUSTERED([id] ASC))
  CREATE TRIGGER [tr_tccprck_d] ON [tccprck] AFTER delete AS ……
  CREATE TRIGGER [tr_tccprck_i_instead_of] ON [tccprck] instead of INSERT AS ……
  CREATE TRIGGER [tr_tccprck_i] ON [tccprck] AFTER INSERT AS ……
  CREATE TRIGGER [tr_tccprck_u_instead_of] ON  [dbo].[tccprck] instead of update AS
  CREATE TRIGGER [tr_tccprck_u] ON [tccprck] AFTER update AS ……
```

（9）产品实时库存表(tccpsskc)，其属性对应的含义：大类,规格,材质,产品入库,产品出库,产品库存,图片,图片文件,单价,折扣率,产品说明,顺序号。

```
CREATE TABLE [tccpsskc]([dl] [char](6) NOT NULL,[gg] [char](30) NOT NULL,
  [cz1] [char](10) NOT NULL, [cprk] [numeric](38,3)NULL,[cpck] [numeric](38,3) NULL,
  [cpkc] [numeric](38,3) NULL,[tp] [varbinary](max) NULL,
  [tpwj] [nvarchar](200) NULL,[dj] [money] NULL,[zkl] [numeric](18,2) NULL,
  [sm] [nvarchar](200) NULL,  [id] [int] IDENTITY(1,1) NOT NULL,
  CONSTRAINT [PK_tccpsskc] PRIMARY KEY CLUSTERED([dl] ASC,[gg] ASC,[cz1] ASC) )
CREATE UNIQUE NONCLUSTERED INDEX [IX_tccpsskc_id] ON [tccpsskc]([id] ASC)
```

（10）月累计库存表(tccptj)，其属性对应的含义：年月,大类,规格,产量,销量,产品库存。

```
CREATE TABLE [tccptj]([ny] [char](6) NOT NULL, [dl] [char](6) NOT NULL,
  [gg] [char](30) NOT NULL,    [cl] [numeric](18, 3) NULL,
  [xl] [numeric](18, 3) NULL,  [cpkc] [numeric](18, 3) NULL,
   CONSTRAINT [PK__tccptj__08EA5793] PRIMARY KEY CLUSTERED([ny] ASC,[dl] ASC,[gg] ASC))
```

（11）产品月区段库存表(tccpkctj)，其属性对应的含义：年月,大类,规格,期初值,产量,销量,期末值。

```
CREATE TABLE [tccpkctj]([ny] [char](6) NOT NULL,     [dl] [char](6) NOT NULL,
  [gg] [char](30) NOT NULL,  [qcz] [numeric](38,3) NULL,[cl] [numeric](18,3) NULL,
  [xl] [numeric](18,3) NULL, [qmz] [numeric](38,3) NULL,
```


```
    CONSTRAINT [PK_tccpkctj] PRIMARY KEY CLUSTERED([ny] ASC,[dl] ASC,[gg] ASC))
```

（12）月产品明细库存表(tccpkc200412)（不同年月表名不同，表属性个数也不同），其属性对应含义：年月,大类,规格,材质,单位,发货去向,期初值,期末值,1 号,2 号,...31 号。

```
CREATE TABLE [tccpkc200412]([ny] [char](6) NOT NULL, [dl] [char](6) NOT NULL,
  [gg] [char](30) NOT NULL, [cz1] [char](10) NOT NULL,     [dw] [char](4) NULL,
  [fhqx] [varchar](50) NULL, [qcz] [numeric](11,3) NOT NULL, [qmz] [numeric](11, 3) NOT NULL,
  [F041201] [numeric](9, 3) NULL,[F041202] [numeric](9,3) NULL,……(类似略)
  [F041231] [numeric](9, 3) NULL,
  CONSTRAINT [PK_tccpkc200412] PRIMARY KEY CLUSTERED([ny],[dl],[gg],[cz1]))
```

（13）模具库存表(tcmjkc)

```
CREATE TABLE [tcmjkc]([顺序号] [int] IDENTITY(1,1) NOT NULL, [分类] [char](6) NOT NULL,
  [厚度] varchar(10) NOT NULL,[乘] [char](1) NULL DEFAULT('*'),[宽度] [varchar](10) NULL,
  [库存数量] int NULL DEFAULT(1),[备注] varchar(50),PRIMARY KEY NONCLUSTERED([顺序号] ASC))
```

（14）系统用户表(users)(C/S 模式系统用户表)，其属性对应的含义：用户编号,用户姓名,口令,等级。

```
CREATE TABLE [users]([uno] [char](6) NOT NULL,  [uname] [char](10) NOT NULL,
  [upassword] [varchar](10) NULL, [uclass] [char](1) NULL DEFAULT('A'),
  PRIMARY KEY CLUSTERED([uno] ASC))
```

（15）日志表(logs)，其属性对应的含义：顺序号,用户编号,操作类型,操作内容,操作日期时间。

```
CREATE TABLE [logs]([id] [int] IDENTITY(1,1) NOT NULL,   [uno] [char](6) NOT NULL,
  [opclass] [char](10) NULL DEFAULT('INSERT'),    [opcommand] [varchar](400) NULL,
  [opdatetime] [datetime] NULL,PRIMARY KEY CLUSTERED([id] ASC))
```

（16）系统参数表(tcsyspara)，其属性对应的含义：显示所有,显示近若干天,库存表保存天数,库存最少生产次数,自动记录日志标记,在线人数,备注,备用 1,备用 2,备用 3,备用 4,备用 5。

```
CREATE TABLE [tcsyspara]([sysall] [int] NULL,[sysdays] [int] NULL,[syskcdays] [int] NULL,
  [syssccs] [int] NULL,    [syslogg] [int] NULL,   [sysrs] [int] NULL,
  [sysbz] [char](200) NULL,[sysp1] [char](10) NULL,[sysp2] [char](10) NULL,
  [sysp3] [char](10) NULL, [sysp4] [char](10) NULL,[sysp5] [char](10) NULL)
```

（17）联系表所需的参照完整性设定语句如下：

```
IF NOT EXISTS (SELECT * FROM sys.foreign_keys WHERE object_id = OBJECT_ID(N'[FK_tccptj_tccpny]') AND parent_object_id = OBJECT_ID(N'[tccptj]'))
ALTER TABLE [tccptj] WITH CHECK ADD CONSTRAINT [FK_tccptj_tccpny] FOREIGN KEY([ny]) REFERENCES [tccpny] ([ny])  -- 其它 ALTER TABLE 语句的 IF 语句（为执行所需）略
ALTER TABLE [tccpkc200412] WITH CHECK ADD CONSTRAINT [FK_tccpkc200412_tccpny] FOREIGN KEY([ny]) REFERENCES [tccpny]([ny])
ALTER TABLE [tccpkctj] WITH CHECK ADD CONSTRAINT [FK_tccpkctj_tccpny] FOREIGN KEY([ny]) REFERENCES [tccpny]([ny])
ALTER TABLE [tccprck] WITH CHECK ADD CONSTRAINT [FK_tccprck_tccpsskc] FOREIGN KEY([dl], [gg], [cz1]) REFERENCES [tccpsskc]([dl], [gg], [cz1])
ALTER TABLE [weborderdetails] WITH CHECK ADD CONSTRAINT [FK_weborderdetails_tccpsskc] FOREIGN KEY([cpid]) REFERENCES [tccpsskc]([id])
ALTER TABLE [weborderdetails] WITH CHECK ADD CONSTRAINT [FK_weborderdetails_weborders] FOREIGN KEY([orderid]) REFERENCES [weborders]([orderid]) ON DELETE CASCADE
ALTER TABLE [weborders] WITH CHECK ADD CONSTRAINT [FK_weborders_webuser] FOREIGN KEY([userid]) REFERENCES [webuser]([id])
```

3．数据库表关系图

数据库名称：kcgl,创建至少 23 张用户表后，表间能形成如图 15-22 的关系图。

4．基于数据库表的视图与索引

（1）视图：基于该数据库库表关系图 15-22，可定义出各种常用的用户视图。如：

```
create view webuser_orders as                    /* 用户订单表 */
    SELECT  u.id AS 序号, u.UserName AS 用户名, u.Email AS 邮箱, o.orderid AS 订单号,
o.ordertime AS 订单日期, o.summoney AS 订购总额
    FROM  dbo.webuser AS u INNER JOIN dbo.weborders AS o ON u.id = o.userid;
create view weborders_details_products as  /* 订单产品明细表 */
    SELECT o.orderid AS 订单号, o.ordertime AS 订单日期, o.summoney AS 订购总额, p.dl AS 大
类, p.gg AS 规格, p.cz1 AS 材质, d.dgl AS 订购量
    FROM  dbo.weborders AS o INNER JOIN dbo.weborderdetails AS d ON
o.orderid = d.orderid INNER JOIN dbo.tccpsskc AS p ON d.cpid = p.id;
create view webuser_orderdetails as   /* 订单信息视图 */
    Select wo.id,wo.userName,wo.orderid,tc.dl,tc.gg ,tc.cz1,wo.dgl
    From weborderdetails wo,tccpsskc tc where wo.cpid=tc.id and wo.clbj='1'
```

按需可以定义出不同视图，这里不再一一列出。

（2）索引：从系统运行性能考虑，可以对系统数据库中记录数多、查询与统计等操作频繁的表（如成品入出库表 tccprck、坯料入出库表 tcplrck 等）创建适量索引，举例如下：

在成品入出库表 tccprck 的 dl、gg 和 cz1 三个属性上创建非聚集非唯一索引：

```
CREATE NONCLUSTERED INDEX [IX_tccprck_dl_gg_cz] ON [dbo].[tccprck](dl ASC,gg ASC,cz1
ASC)
```

在坯料入出库表 tccprck 的 cz1、cz2 和 gg 三个属性上创建非聚集非唯一索引：

```
CREATE NONCLUSTERED INDEX [IX_tcplrck_cz12gg] ON [dbo].[tcplrck](cz1 ASC,cz2 ASC,gg ASC)
```

其它表的所需索引，此略。

注意：表索引的对性能的影响及是否采用，是需要通过实际系统的运行来比较而判定的。

图 15-22 系统数据库库表关系图

15.2.5 数据库初始数据的加载

数据库创建后，要为下一阶段窗体模块、WEB 网页模块的设计与调试作好数据准备，需要整体加载数据，加载数据可以手工一条一条界面录入，也可设计对各表的数据记录的 Insert 命令集，这样执行插入命令集后表数据就有了(一旦要重建数据非常方便)，在准备数据过程中一般要注意以下几点：①尽可能，使用真实数据，这样在录入数据中，能发现一些结构设计中可能的不足之处，并能及早更正；②由于表内或表之间已设置了系统所要的完整性约束规则，如外码、主码等，为此，加载数据时，可能有时序问题，如在生成“产品月统计表”前，一定要先在“产品年月设置表”中先录入该月的数据记录，因为“产品月统计表”中的年月属性值要参照“产品年月设置表”中的年月属性值；③加载数据，应尽可能全面些，能反映各种表数据与表间数据的关系，这样便于模块设计时，程序的充分调试。一般全部加载后，对数据库要及时做备份，因为测试中会频繁更改或无意损坏数据，而建立起完整的测试数据库数据是很费时的。

15.2.6 库存管理系统的设计与实现

库存管理系统（C/S）使用 Visual C# 2013 语言在 Visual Studio 2013 开发平台中设计实现。系统采用多项目共同组成系统解决方案来实现，其中除一个是输出类型为

Windows 应用程序的主启动项目外，其它都是输出类型为类型（dll 动态连接库型）的辅助项目。

创建系统解决方案及项目过程为：首先，在 Visual Studio 2013 中文件→新建→项目→其他项目类型→Visual Studio 解决方案，解决方案名称取 KCGL，解决方案存放位置可按需浏览确定某文件夹；然后，在解决方案中添加主启动项目 KCGLWinForm，方法是：文件→添加→新建项目，出现“添加新建项目”对话框，其中项目类型选“Visual C#→Windows→Windows 应用程序”，项目名称为 KCGLWinForm，位置为 KCGL 解决方案所在目录下的子目录，如 KCGL；再次，按需逐个添加其它辅助类型项目，方法类似于添加主启动项目 KCGLWinForm，不同处为项目类型为“Visual C#→类型”，项目名与项目位置取不同的。

本系统组织成多辅助类型项目构成，主要有公用类型项目 KCGLCommon、公共变量类 KCGLStatic、功能窗体接口类 KCGLInterFace、功能窗体方法实现类 KCGLMethod 等。这样的组织使得系统具有更好的维护性，更清晰的层次性。系统解决方案及其组成项目如下图。

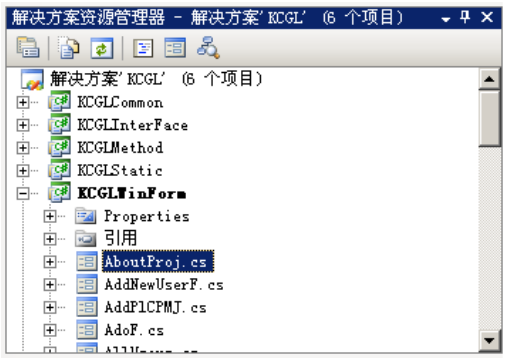


图 15-23　系统解决方案及其组成项目

1、库存管理系统的主窗体设计

本系统主窗体还采用多文档界面窗体，其它功能界面设计成子窗体，为此文档界面主窗体“MainF”上可加入主菜单、工具栏与状态栏等，运行后，登录窗体如图 15-24 所示，顺利登录系统后，系统主窗体如图 15-25 所示。

图 15-24　系统登录窗体

图 15-25 库存管理子系统的主界面

在主窗体上，功能菜单体现了系统的主要功能模块，如图 15-26 所示。

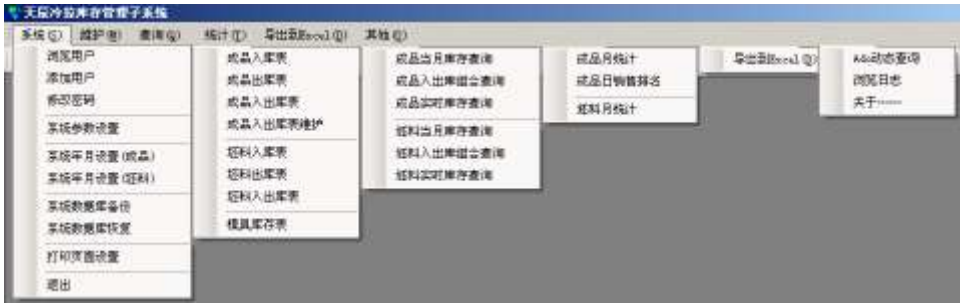


图 15-26 主菜单

2、创建公用模块

（1）在系统中可以用公用类（在类型项目 KCGLStatic 中）来存放整个工程项目公共的全局变量等，这样便于管理与使用这些公共变量。具体如下：

```
using System;
using System.Collections.Generic;
using System.Text;
namespace KCGLStatic
{   /// 定义一组公共静态变量
    public class StaticMember
    {   public static string connectString = null; //记录当前的数据库连接字符串
        public static string userPassword = null;  //记录当前用户的登录密码
        public static string userName = null;      //记录当前的用户名
        public static int  userClass;              //记录当前用户的级别
```

```
        public static int icount;                    //记录系统的操作次数
        public static string YhSR;                   //记录用户输入的用于比较判断的密码
        public static bool showAll=true;             //显示所有的出入库值
        public static int sysdays;                   //记录系统参数的日期
        public static int sysKcdays;                 //记录库存日期
        public static int syssccs;                   //每月库存统计的次数
        public static string sysServerName = null; //数据库服务器名
        public static string sysDatabaseName=null; //系统数据库名
        public static string sysDbUserName = null; //数据库用户名
        public static string sysDbPassword = null; //数据库登录密码
        public static bool sysLogg = true;           //是否自动记录系统日志
        public static bool sysdlggcz = true;         //是否修改大类，规格，材质表
        public static int cpNumber;                  //记录产品数量
        public static double cpTot;                  //记录产品总数量
        public static int plNumber;                  //记录坯料数量
        public static double plTot;                  //记录坯料总数量
        public static int sysrs;                     //记录系统在线人数
        public static string sysbz;                  //记录系统备注
        public static bool Isplrk = false;           //判断是否为坯料入库，默认为false
        public static bool IsCprk = false;           //判断是否为产品入库，默认为false
        public static bool IsMjrk = false;           //判断是否为模具入库，默认为false
        public static string selectRq="";            //系统选定日期
    }
}
```

（2）各功能模块对数据库中数据的操作，主要是通过 ADO.NET 模型类 Command、DataAdapter、DataSet、DataTabel、connection、SqlCommandBuilder 的对象递交执行 SQL 命令来完成的。本系统把这些最基本的数据操作函数放置在 Command.cs（在类型项目 KCGLCommon 中）类中。下面罗列些最重要的类函数：

```
using System;
using System.Collections.Generic;
using System.Text;
using System.Data;
using System.Data.Sql;
using System.Data.SqlClient;
using System.IO;
using KCGLStatic;
using Microsoft.Office.Core;
namespace KCGLCommon
{   /// 定义一组方法，用来操作数据库，以便在后面的程序中直接调用
    public class Command
    {   //定义一组变量，表示各种操作的数据
        // Command
        private SqlCommand SelectCommand = null;
        private SqlCommand UpdateCommand = null;
        private SqlCommand StoreCommand = null;
        private static SqlDataAdapter myDataAdapter = null; //定义DataAdapter
        private static DataSet myDataSet = null;            //定义DataSet
        private DataTable myDataTable = null;               //定义DataTabel
        private static SqlConnection myConnection = null;   //定义connection
```

```
        private string connectString = string.Empty;
        private static SqlCommandBuilder myCommandBuilder = null;//定义sqlCommandBuilder
        public Command()                        // 初始化类
        {
            this.connectString = StaticMember.connectString;
        }
        public Command(string connectString) // 初始化类
        {   this.connectString = connectString; }
        /// 建立与数据库的连接
        public bool ConnectDB()
        {   bool successFlag = false;
            try
            {   myConnection = new SqlConnection();
                myConnection.ConnectionString = connectString;
                myConnection.Open();
                successFlag = true;
            }
            catch (Exception ex)
            {   throw ex; }
            return successFlag;
        }
        public void disConnect()
        {   try
            {   myConnection.Close(); }
            catch (SqlException ex)
            {  throw ex; }
        }
        // 从数据库中查询数据，并将其填充到dataset中
        public DataSet selectMember(string sqlText, string DataSetName)
        {   try
            {   if (ConnectDB())
                {   myDataSet = new DataSet(DataSetName);
                    SelectCommand = new SqlCommand();
                    SelectCommand.CommandText = sqlText;
                    SelectCommand.CommandType = CommandType.Text;
                    SelectCommand.Connection = myConnection;
                    myDataAdapter = new SqlDataAdapter();
                    myDataAdapter.SelectCommand = SelectCommand;
                    myCommandBuilder = new SqlCommandBuilder(myDataAdapter);
                    myDataAdapter.FillSchema(myDataSet, SchemaType.Source,
DataSetName);
                    myDataAdapter.Fill(myDataSet, DataSetName);
                }
            }
            catch (SqlException sqlex)
            {   throw sqlex;  }
            finally
            {   disConnect(); }
            return myDataSet;
        }
```

```
// 从数据库中取得数据，并填充到dataTable
public DataTable selectMemberToTable(string sqlText, string datatablename)
{   try
    {   if (ConnectDB())
        {   myDataTable = new DataTable(datatablename);
            SelectCommand = new SqlCommand();
            SelectCommand.CommandText = sqlText;
            SelectCommand.CommandType = CommandType.Text;
            SelectCommand.Connection = myConnection;
            myDataAdapter = new SqlDataAdapter();
            myDataAdapter.SelectCommand = SelectCommand;
            myCommandBuilder = new SqlCommandBuilder(myDataAdapter);
            myDataAdapter.FillSchema(myDataTable, SchemaType.Source);
            myDataAdapter.Fill(myDataTable);
        }
    }
    catch (SqlException sqlex)
    {   throw sqlex;}
    finally
    { disConnect(); }
    return myDataTable;
}
// 更新数据库中的信息
public int updateMember(string sqlText)
{   int count = 0;
    if (ConnectDB())
    {   try
        {   UpdateCommand = new SqlCommand();
            UpdateCommand.CommandText = sqlText;
            UpdateCommand.CommandType = CommandType.Text;
            UpdateCommand.Connection = myConnection;
            count = UpdateCommand.ExecuteNonQuery();
        }
        catch (SqlException sqlex)
        {   throw sqlex; }
        finally
        {   disConnect();}
    }
    return count;
}
// …… 省略其它函数
/// 执行无参存储过程
public bool execStore(string storeName, ref string errorMessage)
{   bool successFlag = false;
    if (ConnectDB())
    {   try
        {   StoreCommand = new SqlCommand();
            StoreCommand.CommandText = storeName;
            StoreCommand.CommandType = CommandType.StoredProcedure;
            StoreCommand.CommandTimeout = 10;
```


```
                    StoreCommand.Connection = myConnection;
                    StoreCommand.ExecuteNonQuery();
                    successFlag = true;
                }
                catch (Exception ex)
                {   errorMessage = ex.ToString(); }
                finally
                {   disConnect();  }
            }
            return successFlag;
        }
        // …… 省略其它函数
    }
}
```

3、系统运行线路及连接字符串的配置

本系统的组织、组成显得复杂，然而其运行线路是唯一的。

（1）Windows 应用程序从如下 Main()开始运行。

```
/// The main entry point for the application.
[STAThread]
static void Main()
{   Application.EnableVisualStyles();
    Application.SetCompatibleTextRenderingDefault(false);
    Application.Run(new ConnectDBF());
}
```

（2）“Application.Run(new ConnectDBF());”语句，运行转到连接字符串获取与选定功能窗体。ConnectDBF窗体运行时，先从系统的XML配置文件“xml\connectStringX.xml”中读取预设置的连接字符串信息到可选数据源组合框中等待选取。

位于项目KCGL所在目录“KCGL”下“bin\Release\xml”或“bin\Debug\xml”下“connectStringX.xml”文件中的内容如下所示：

```
<?xml version="1.0" encoding="utf-8" ?>
<!-- 插入一些连接数据库字符串-->
<connectString>
  <connectStringIP>
    <value>Data Source= WIN-4313FJ7ON1E\SQLEXPRESS2014;Initial Catalog=KCGL;User ID=sa;
       password=sasasasa;Pooling=True</value >
  </connectStringIP >
<!—其它可选连接数据库字符串略 -->
</connectString>
```

其中“Data Source= WIN-4313FJ7ON1E\SQLEXPRESS2014;Initial Catalog=KCGL;User ID=sa;password=sasasasa;
Pooling=True”指定了连接系统数据库的服务器名为“WIN-4313FJ7ON1E\SQLEXPRESS2014”，数据库名为“KCGL”，用户名为“sa”，用户密码为“sasasasa”。

若缺省取第一种连接字符串的话，可以在ConnectDBF窗体运行时，自动选取获得连接数据库字符串。

（3）ConnectDBF窗体运行并获得连接数据库字符串后，运行转到系统登录窗体。命令如下：

```
LoginF Login = new LoginF();
```

```
            Login.Show();
            this.Hide();
```

（4）LoginF登录窗体运行时，在输入用户名与密码后，通过如下MLogin类来判断某用户是否能进入本系统。

```
using System;
using System.Collections.Generic;
using System.Text;
using System.Data;
using System.Data.Sql;
using System.Data.SqlClient;
using KCGLStatic;
namespace KCGLMethod
{
    public class MLogin
    {   protected string userName = null;
        protected string userPassword = null;
        protected bool successFlag = false;
        public MLogin(string userName, string userPassword)
        {
            this.userName = userName;
            this.userPassword = userPassword;
        }
        public bool LoginTo()
        {   SqlConnection myConnection=new SqlConnection(StaticMember.connectString);
            SqlCommand myCommand = new SqlCommand();
            myCommand.CommandText = "select uname,upassword,uclass from users where
uname='"+this.userName+"' And upassword='"+this.userPassword+"'";
            myCommand.CommandType = CommandType.Text;
            myCommand.Connection = myConnection;
            myConnection.Open();
            try
            {   SqlDataAdapter myDataAdapter = new SqlDataAdapter();
                myDataAdapter.SelectCommand = myCommand;
                DataSet userDataset = new DataSet();
                myDataAdapter.Fill(userDataset, "user");
                if (userDataset.Tables["user"].Rows.Count == 1)
                {   StaticMember.userClass =
                        Convert.ToInt32(userDataset.Tables[0].Rows[0][2]);
                    StaticMember.userPassword =
                        Convert.ToString(userDataset.Tables[0].Rows[0][1]);
                    successFlag = true;
                }
                else
                {   successFlag = false; }
                myConnection.Close();
            }
            catch (Exception e)
            {   throw e; }
            return successFlag;
```

```
            }
        }
    }
```

（5）若验证通过，LoginF登录窗体中，运行如下命令，真正打开系统主界面窗体。

```
Main.Show();
this.Hide();
```

4、成品出库或入库录入模块的实现

成品（即产品）出库或入库录入窗口，其运行界面(只列出子窗口，下同)如图 15-27 所示：

图 15-27 成品入出库维护窗体

图 15-28 成品出库或入库组合查询窗口

成品出入库录入窗口，以网格形式提供了对入库或出库单的录入、修改、删除等维护

原始单据数据的功能，功能设计操作简单又直观。系统中除提供网格形式直观维护成品出入库数据外，还提供单记录输入界面。

成品出入库数据录入后，除了能在录入窗口中查找到出入库原始数据外，还可以通过如图 15-28 成品出库或入库组合查询窗口中更有效的进行查询与数据核对等。

5、成品月明细库存生成与查询模块的实现

图 15-29 成品月明细库存生成与查询模块的运行界面

成品月明细库存生成与查询模块的运行界面如图 15-29 所示，模块实现简述：利用组合条件实现查询，能方便并快速地查找到信息。本功能窗体被设计成上下两部分，上部分数据网格控件显示查到的记录；下部分组合 3 种条件，每个条件能指定独立的比较运算符以形成条件表达式，当按“显示”按钮时，程序能组合你的各选择条件形成最终组合条件以查询并显示记录；而“生成并显示”按钮能完成成品月明细库存的及时生成；选择网格数据的某行（代表某产品）与某列（代表某天等），再按“详细”按钮，能弹出窗体显示相应数据对应的入出库原始记录，以便对原始数据的查阅与核对。

成品月明细库存“生成并显示”与“显示”两按钮实现功能的程序代码（特别注意 ADO.NET 对象的创建与使用、SQL 命令的使用）参阅本书相关资料中的相应程序，此略。

系统年月设置表控制着成品月明细库存的天数范围及对月明细库存表的创建、生成、结转、删除等管理功能，图 15-30 所示的窗口简明地实现了这些功能。

图 15-30 系统年月设置表的控制功能

6、成品实时库存计算与组合查询模块的实现

成品实时库存计算与组合查询模块的运行界面如图 15-31 所示，模块实现简述：本功能窗体被设计成上下两部分，上部分数据网格控件显示查到的库存记录；下部分可组合 6 种条件。当按“显示”按钮时，程序能组合你的各选择条件以查询并显示记录；而“计算库存”按钮能重新统计计算出库存（要说明的是，由于通过对成品出入库表设置添加、修改、删除触发器，来自动更新成品实时库存的，为此“计算库存”按钮是很少需要使用的）；选择网格数据的某行（代表某种产品），再按“详细”按钮，能弹出窗体显示相应产品的入出库原始记录，以便对原始数据的查阅与核对。

图 15-31 成品实时库存组合查询窗体

成品实时库存的组合查询实现方法同上节“5”中“显示”按钮组合查询的实现，“计算库存”按钮的实现，则采取了调用数据库存储过程的方法实现的，这样能充分利用存储过程的优点。“计算库存”按钮的单击事件代码（使用了存储过程代码显得非常简单）为：

```
/// 库存重新计算事件
private void Cmdjskc_Click(object sender, EventArgs e)
```

```
    {   if (MessageBox.Show("正常情况下不需要重新统计库存，真的要重新统计库存吗",
"Question", MessageBoxButtons.YesNo,
            MessageBoxIcon.Question) == DialogResult.Yes)
        {   try
            {   _ds_store = _Cpsskc.getByStore("p_refresh_tccpsskc", ref ErrorMessage);
                this.cpView.DataSource = _ds_store.Tables[0];
                MessageBox.Show("库存已经重新统计完毕", "Information",
MessageBoxButtons.OK, MessageBoxIcon.Asterisk);
            }
            catch (Exception ex)
            {   MessageBox.Show(ex.Message + ErrorMessage); }
        }
        else return;
    }
```

其中存储过程 p_refresh_tccpsskc 的内容为：

```
CREATE PROCEDURE [dbo].[p_refresh_tccpsskc] AS
BEGIN
SET NOCOUNT ON
  update tccpsskc set cprk=0,cpck=0,cpkc=0; -- 置成 0
  -- 添加新的
  insert into tccpsskc(dl,gg,cz1,cprk,cpck,cpkc)
      select tt.dl,tt.gg,tt.cz1,0,0,0 from
         (select dl,gg,cz1 from tccprck group by dl,gg,cz1) as tt
       where (tt.dl+ tt.gg+ tt.cz1) not in (select dl+gg+cz1 from tccpsskc);
  -- 合计入库量
  WITH t1(dl,gg,cz1,kc) AS (select dl,gg,cz1,sum(crkz) as kc from tccprck where crkz>=0
group by dl,gg,cz1)
  update tccpsskc set cprk=t1.kc from t1
  where tccpsskc.dl=t1.dl and tccpsskc.gg=t1.gg and tccpsskc.cz1=t1.cz1;
  -- 合计出库量
  WITH t2(dl,gg,cz1,kc) AS (select dl,gg,cz1,sum(crkz) as kc from tccprck where crkz<0
group by dl,gg,cz1)
  update tccpsskc set cpck=t2.kc from t2
  where tccpsskc.dl=t2.dl and tccpsskc.gg=t2.gg and tccpsskc.cz1=t2.cz1;
update tccpsskc set cpkc=cprk+cpck  -- 计算库存
END
```

7、成品产量与销量月统计模块的实现

图 15-32 成品产量销售月统计窗口

成品产量与销量月统计模块的运行界面如图 15-32 所示,模块主要实现月产品结余统计（主要包含月产量、销量及结余等）与显示，本功能的实现主要通过两个存储过程来实现的。它们是 P_KCGL_CPTJ 与 P_CPGL_Q，具体请查阅数据库 KCGL。

8、系统用户表导出到 Excel 模块的实现

为便于熟悉 Excel 电子表格的用户，编辑、排版与打印系统的表数据，本系统设计实现了便捷的表记录导出到 Excel 的功能，这样极大方便了系统应用的灵活性与实用性。该功能窗体的运行界面如图 15-33 所示,。左表是所有系统用户表,需要时移到右边列表框中，选定要导出的表，按“导出到 Excel”按钮开始自动的导出到某 Excel 文件的过程，过程中可以指定已有 ExceL 文件，否则系统会新建一个缺省的 Excel 文件。其具体实现代码略。

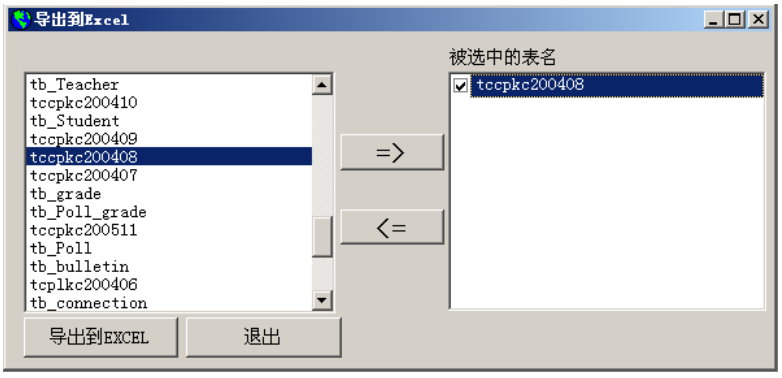


图 15-33 系统用户表导出到 Excel 的实现窗口

限于篇幅，其它功能模块及辅助功能等说明略，请参阅相关资料中的相应程序。

15.2.7 系统的编译与发行

企业库存管理系统的各相关模块设计与调试完成后，接着要对整个系统编译发布。按“项目”→“属性”，打开解决方案“KCGL”属性页，选中“配置属性”中“配置”节点，在对话框首行“配置”组合框选“活动(Release)”，按“确定”退出对话框。在解决方案资源管理器中，鼠标右键点击“解决方案’KCGL’”，在弹出菜单中点击“重新生成解决方案”，系统重新生成解决方案后，即生成了系统可执行文件 kcgl.exe 及相关 DLL 动态连接库。生成的相关文件在“KCGL\bin\Release”子目录中。Release 子目录中的这些系统文件即是可发布应用系统程序。

15.2.8 网上订购系统的设计与实现

1、网站操作流程

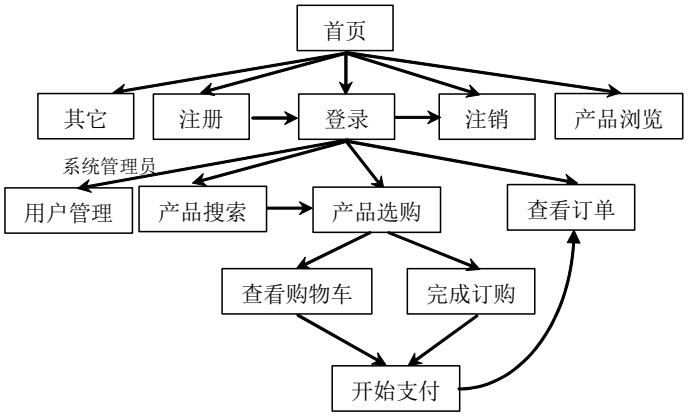


图 15-34 网站操作流程示意图

网上订购系统运行时常常按图 15-34 所示的操作流程进行操作。

2、网上订购的 Web 首页

图 15-35 Web 网上订购子系统的首页界面

利用 ASP.NET 设计的 Web 首页，如图 15-35 所示。Web 首页（index.htm）由上、左、中、下四部份组成。

上部是图标等显示区，主要显示企业图标、动态宣传图片等。

左部是带状功能展示区，主要有资源搜索功能，能实现订购产品的组合查询；操作链接区能显示常用功能链接及分用户等级显示的管理功能链接等。另为还有“登录”、“重置”、“退出”等链接。

中部是主显示区，产品的查阅、订阅、支付、Web 信息页面的显示等都在中区进行，为此该区占据显示屏幕的大部分。

本系统产品种类较多，网页应设计成分页显示形式，如图 15-35 所示。

3、产品选购的实现

操作界面请参见图 15-35，为了快速选购需要的产品可以在左上产品搜索区组合设定产品的品名、规格及材质等，按“搜索”按钮，右边操作区即显示搜索到的产品，接着可以上下移动查阅产品、选定产品、指定订购量（不能超过库存量），全面选购所需产品了。如图 15-36 为订购界面。

4、查看购物车与支付的实现

上一节产品分散选购完成后，按“购物车”图标或“购物车”超链接均可以进入到查阅购物车来确定完成产品订购的步骤，如图 15-37 所示。此时按“确定支付”按钮，则正式完成网上订购任务。

图 15-36　产品订购界面

图 15-37　完成产品订购功能的实现窗口

5、查看订单的实现

图 15-38 查看订单功能的实现窗口

查看用户订单功能由文件“LookOrder.aspx”来实现，如图 15-38 所示，页面右中部显示该用户的所有已完成订单。篇幅所限，本系统的所有程序代码主要要通过随书资料查阅获得，至此网上订购系统的主要功能罗列了。要说明的是本系统主要功能由学生设计完成，学生在设计实现中对部分表结构作了改动，实现的功能要求也有所降低。然而，就是如此带着些不足，本系统仍不失其参考价值。另为，稍早用 VB+SQL Server 2005 设计实现的本系统，读者可一并参考学习的。

15.2.10 小结

篇幅所限，这里虽没有给出系统完整模块与完整的程序代码。但我们已能领略到一个完整的基于 C/S 结构与基于 B/S 结构相结合的数据库应用系统的全貌了。我们把真实企业的小系统介绍给大家，是希望大家能领略到以下几点：①数据库应用系统的开发设计是一个规范化的过程，需要遵循一定方式、方法与开发设计步骤；②数据库关系模式设计非常重要，是整个系统设计的中心，其设计合理与否，将全面影响整个系统的成功实现；③应用系统中数据库操作的实质是设计、组织、递交 SQL 命令，并根据 SQL 命令的执行状态，决定后序的数据处理与操作。不同开发工具操作各具特色，只有利用 SQL 命令实现数据的存取这一点是共同的。系统功能设计、实现与代码介绍中，我们力求呈现这一特色。为此，大家学习中应抛开表面看本质，关注 SQL 命令的操作特色，这样，换其它开发工具，在数据操作方面你将照样得心应手；④我们介绍的系统，其实现方法及功能并非无暇可击，更不是最优或最完美。实现中更没有去特意挖掘 Visual C# 2013 与 ASP.NET 网页设计语言的

开发技巧。为此只是给出个例子，起到抛砖引玉作用而已。

实验内容与要求（选做）

实验总体内容

从应用出发，分析用户需求，设计数据库概念模型、逻辑模型、物理模型，并创建数据库，优化系统参数，了解数据库管理系统提供的性能监控机制，设计数据库的维护计划，了解并实践 C/S 或 B/S 结构应用系统开发。

实验具体要求

1、结合某一具体应用，调查分析用户需求，画出组织机构图、数据流图、判定表或判定树，编制数据字典；

2、设计数据库概念模型及应用系统应具有的功能模块；

3、选择一数据库管理系统，根据其所支持的数据模型，设计数据库的逻辑模型（即数据库模式），并针对系统中的各类用户设计用户视图；

4、在所选数据库管理系统的功能范围内设计数据库的物理模型；

5、根据所设计的数据库的物理模型创建数据库，并加载若干初始数据；

6、了解所选数据库管理系统允许设计人员对哪些系统配置参数进行设置，以及这些参数值对系统的性能有何影响，再针对具体应用，选择合适的参数值；

7、了解数据库管理系统提供的性能监控机制；

8、在所选数据库管理系统的功能范围内设计数据库的维护计划。

9、利用某 C/S 或 B/S 结构开发平台或开发工具开发设计，实现某数据库应用系统。

实验报告主要内容

1、数据库设计各阶段的书面文档，说明设计的理由；

2、各系统配置参数的功能及参数值的确定；

3、描述数据库系统实现的软件、硬件环境，说明采用这样环境的原因。

4、说明在数据库设计过程碰到的主要困难，所使用的数据库系统在哪些方面还有待改进。

5、应用系统试运行情况与系统维护计划。

实验系统（或课程设计）参考题目（时间约两周）：

邮局订报管理子系统

设计本系统模拟客户在邮局订购报纸的管理内容，包括查询报纸、订报纸、开票、付钱结算、订购后的查询、统计等的处理情况，简化的系统需要管理的情况如下：

1、可随时查询出可订购报纸的详细情况，如报纸编号(pno)、报纸名称(pna)、报纸单价(ppr)、版面规格(psi)、出版单位(pdw)等，这样便于客户选订。

2、客户查询报纸情况后即可订购所需报纸，可订购多种报纸，每种报纸可订若干份，交清所需金额后，就算订购处理完成。

3、为便于邮局投递报纸，客户需写明如下信息：客户姓名(gna)、电话(gte)、地址(gad)及邮政编码(gpo)，邮局将即时为每一客户编制唯一代码(gno)。

4、邮局对每种报纸订购人数不限，每个客户可多次订购报纸，所订报纸亦可重复。

根据以上信息完成如下要求：

1、请认真做系统需求分析，设计出反映本系统的 E-R 图(需求分析、概念设计)。

2、写出相应你设计的 E-R 图的关系模式，根据设计需要也可增加关系模式，并找出各关系模式的关键字(逻辑设计)。

3、在你设计的关系模式基础上利用 C#+SQL Server（或其它开发设计平台）开发设计该子系统，要求子系统能完成如下功能要求(物理设计、设施与试运行)：(1)在 SQL Server 中建立各关系模式对应的库表，并确定索引等；(2)能对各库表进行输入、修改、删除、添加、查询、打印等基本操作；(3)能根据订报要求订购各报纸，并完成一次订购任务后汇总总金额，模拟付钱、开票操作；(4)能明细查询某客户的订报情况及某报纸的订出情况；(5)能统计出某报纸的总订数量与总金额及某客户订购报纸种数、报纸份数与总订购金额等；(6)其他你认为子系统应有的查询、统计功能；(7)要求子系统设计界面友好，功能操作方便合理，并适当考虑子系统在安全性、完整性、备份、恢复等方面的功能要求。

4、子系统设计完成后请书写课程设计报告，设计报告要围绕数据库应用系统开发设计的步骤来考虑书写，力求清晰流畅。最后根据所设计子系统与书写报告(报告按数据库开发设计六个步骤的顺序逐个说明表达，并说明课程设计体会等)好坏评定成绩。

图书借阅管理子系统

设计本系统模拟学生在图书馆借阅图书的管理内容，包括查询图书、借书、借阅后的查询、统计、超期罚款等的处理情况，简化的系统需要管理的情况如下：

1、可随时查询出可借阅图书的详细情况，如图书编号(bno)、图书名称(bna)、出版日期(bda)、图书出版社(bpu)、图书存放位置(bpl)、图书总数量(bnu)等，这样便于学生选借。

2、学生查询图书情况后即可借阅所需图书，可借阅多种图书，每种图书一般只借一本，若已有图书超期请交清罚金后，才能开始本次借阅；

3、为了唯一标识每一学生，图书室办借书证需如下信息：学生姓名(sna)、学生系别(sde)、学生所学专业(ssp)、借书上限数(sup)及唯一的借书证号(sno)。

4、每学生一次可借多本书，但不能超出该生允许借阅上限数(上限数自定)，每个学生可多次借阅，允许重复借阅同一本书。规定借书期限为二个月，超期每天罚二分。

根据以上信息完成如下要求：

1、请认真做系统需求分析，设计出反映本系统的 E-R 图(需求分析、概念设计)。

2、写出相应你设计的 E-R 图的关系模式，根据设计需要也可增加关系模式，并找出各关系模式的关键字(逻辑设计)。

3、在你设计的关系模式基础上利用 C#+SQL SERVER（或其它开发设计平台）开发设计该子系统，要求子系统能完成如下功能要求(物理设计、设施与试运行)：(1)在 SQL SERVER 中建立各关系模式对应的库表，并确定索引等；(2)能对各库表进行输入、修改、删除、添

加、查询、打印等基本操作；(3)能根据学生要求借阅图书库中有的书，并完成一次借阅任务后汇总已借书本总数，报告还可借书量，已超期的需付清罚款金额后才可借书；(4)能明细查询某学生的借书情况及图书的借出情况；(5)能统计出某图书的总借出数量与库存量及某学生借书总数，当天为止总罚金等；(6)其他你认为子系统应有的查询、统计功能；(7)要求子系统设计界面友好，功能操作方便合理，并适当考虑子系统在安全性、完整性、备份、恢复等方面的功能要求。

4、子系统设计完成后请书写课程设计报告，设计报告要围绕数据库应用系统开发设计的步骤来考虑书写，力求清晰流畅。最后根据所设计子系统与书写报告(报告按数据库开发设计六个步骤的顺序逐个说明表达，并说明课程设计体会等)好坏评定成绩。

其它可选子系统

1、图书销售管理系统

调查新华书店图书销售业务，设计的图书销售点系统主要包括进货、退货、统计、销售功能，具体：(1)进货：根据某种书籍的库存量及销售情况确定进货数量，根据供应商报价选择供应商。输出一份进货单并自动修改库存量，把本次进货的信息添加到进货库中；(2)退货：顾客把已买的书籍退还给书店。输出一份退货单并自动修改库存量，把本次退货的信息添加到退货库中；(3)统计：根据销售情况输出统计的报表。一般内容为每月的销售总额、销售总量及排行榜；(4)销售：输入顾客要买书籍的信息，自动显示此书的库存量，如果可以销售；打印销售单并修改库存，同时把此次销售的有关信息添加到日销售库中。

2、人事工资管理系统

考察某中小型企业，要求设计一套企业工资管理系统，其中应具有一定的人事档案管理功能。工资管理系统是企业进行管理的不可缺少的一部分，它是建立在人事档案系统之上的，其职能部门是财务处和会计室。通过对职工建立人事档案，根据其考勤情况以及相应的工资级别，算出其相应的工资。为了减少输入账目时的错误，可以根据职工的考勤、职务，部门和各种税费自动求出工资。

为了便于企业领导掌握本企业的工资信息，在系统中应加入各种查询功能，包括个人信息、职工工资、本企业内某一个月或某一部门的工资情况查询，系统应能输山各类统计报表。

3、医药销售管理系统

调查从事医药产品的零售、批发等工作的企业，根据其具体情况设计医药销售管理系统。主要功能包括：(1)基础信息管理：药品信息、员工信息、客户信息、供应商信息等；(2)进货管理：入库登记、入库登记查询、入库报表等；(3)库房管理：库存查询、库存盘点、退货处理、库存报表等；(4)销售管理：销售登记、销售退货、销售报表及相应的查询等；(5)财务统计：当日统计、当月统计及相应报表等；(6)系统维护。

4、宾馆客房管理系统

具体考察本市的宾馆，设计客房管理系统，要求：(1)具有方便的登记、结账功能，以及预订客房的功能，能够支持团体登记和团体结账；(2)能快速、准确地了解宾馆内的客房状态，以便管理者决策；(3)提供多种手段查询客人的信息；(4)具备一定的维护手段，有

一定权利的操作员在密码的支持下才可以更改房价、房间类型、增减客房；(5)完善的结账报表系统。

5、车站售票管理系统

考察本市长途汽车站、火车站售票业务，设计车站售票管理系统。要求：(1)具有方便、快速的售票功能，包括车票的预订和退票功能，能够支持团体的预订票和退票；(2)能准确地了解售票情况，提供多种查询和统计功能，如车次的查洵、时刻表的查询；(3)能按情况所需实现对车次的更改、票价的变动及调度功能；(4)完善的报表系统。

6、汽车销售管理系统

调查本地从事汽车销售的企业，根据该企业的具体情况，设计用于汽车销售的管理系统。主要功能有：(1)基础信息管理：厂商信息、车型信息和客户信息等；(2)进货管理：车辆采购、车辆入库；(3)销售管理：车辆销售、收益统计；(4)仓库管理：库存车辆、仓库明细、进销存统计；(5)系统维护：操作员管理、权限设置等。

7、仓储物资管理系统

经过调查，对仓库管理的业务流程进行分析。库存的变化通常是通过入库、出库操作来进行。系统对每个入库操作均要求用户填写入库单，对每个出库操作均要求用户填写出库单。在出入库操作同时可以进行增加、删除和修改等操作。用户可以随时进行各种查询、统计、报表打印、账目核对等工作。另外，也可以用图表形式来反映查洵结果。

8、企业人事管理系统

调查本地的企业，根据企业的具体情况设计企业人事管理系统。主要功能有：(1)人事档案管理：户口状况、政治面貌、生理状况、合同管理等；(2)考勤加班出差管理；(3)人事变动：新进员工登记、员工离职登记、人事变更记录；(4)考核奖惩；(5)员工培训；(6)系统维护：操作员管理、权限设置等。