

# 登录系统

## 1 程序说明

该程序与数据库建立连接,为用户提供一个完整的登录程序。主要内容包括:用户登录及用户管理两大内容。其中用户管理中,用户可以根据需要进行用户添加、删除、编辑等操作。通过此教程,大家可以掌握如何和数据库进行连接,以及通过 LabView 对数据库中的内容进行操作。为以后的数据采集的信息记录奠定基础。

## 2 各部分简介

(1) 软件: LabView 基本模块(2014 版本)、 LabView Database Connectivity Toolkit(数据库连接工具包)。

(2) 前面板及程序框图介绍

① 前面板主要是用于用户的登录及操选择。(这里简介的是登录的前面板,其他相关子 VI 的相关内容会在后面介绍)。

登录前如图? 所示



图?

登陆后如图? 所示

登录界面

用户名 CZU

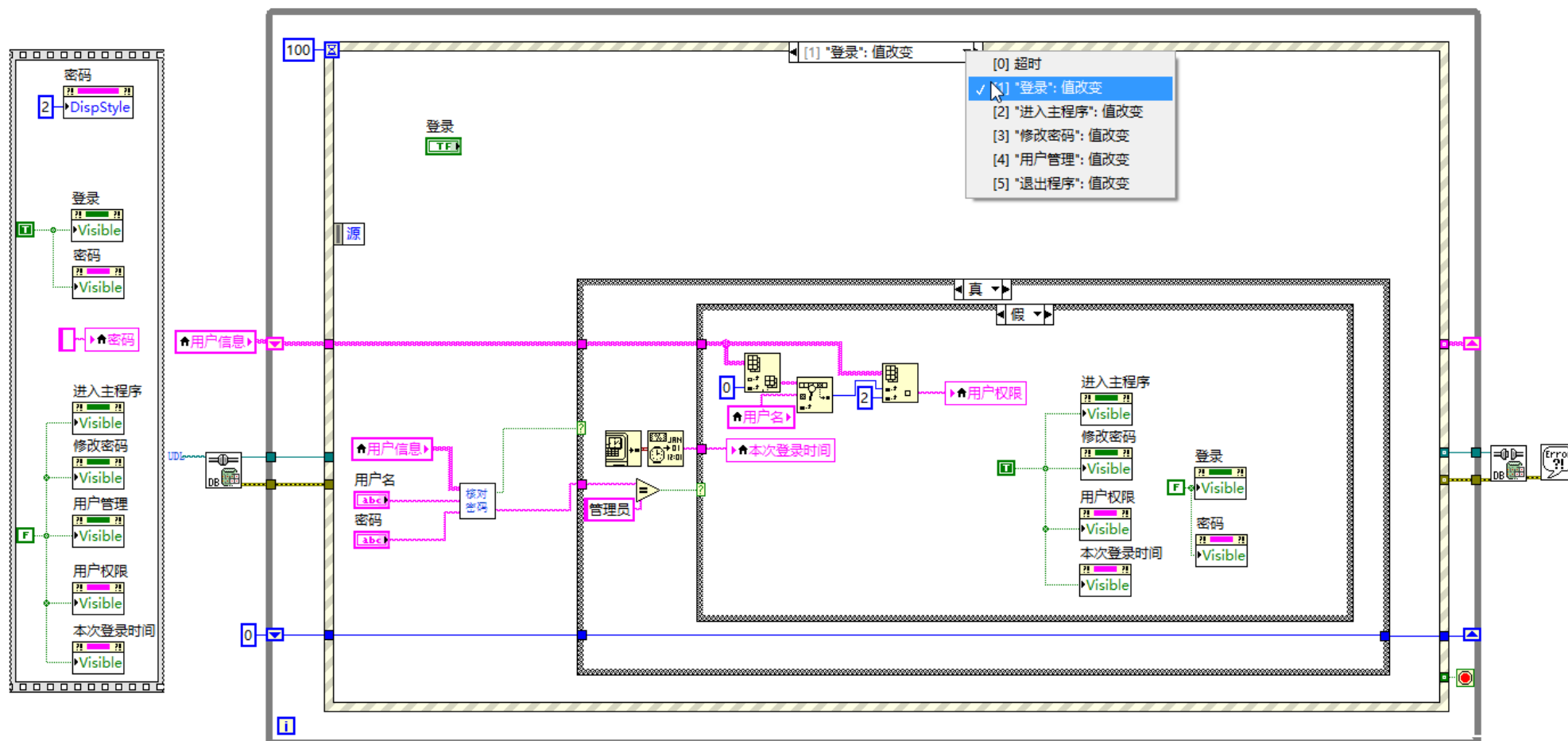
用户权限 管理员

本次登录时间 2015/8/27 9:48:10

进入主程序 修改密码 用户管理 退出程序

图？

② 程序框图使用事件结构对前面板上对应控件的操作进行相应。如图? 所示

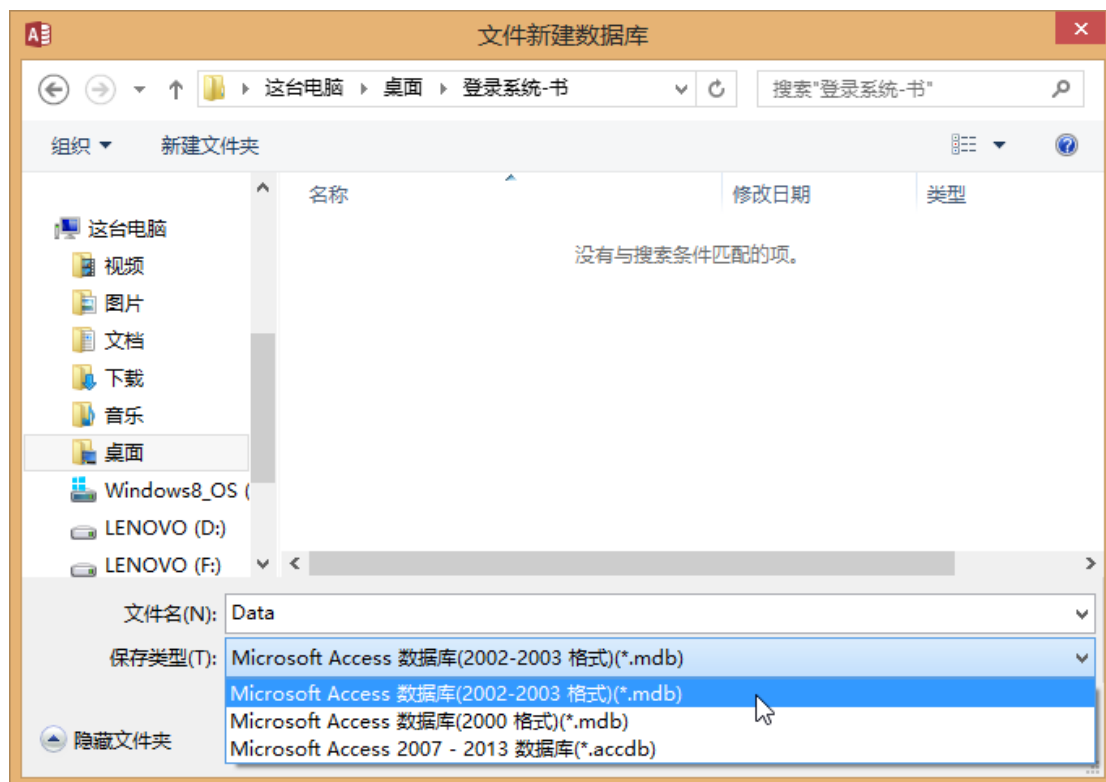


图?

### 3 操作步骤

#### 3.1 与数据库建立连接

使用 NI 所提供的数据库工具包 LabView Database Connectivity Toolkit 只能操作数据库,但是无法创建数据库,因此需要借助第三方数据库管理系统,如 Access 等来创建数据库。首先我们建立一个 Data.mdb(或 Data.accdb) 的数据库文件。如图?所示。

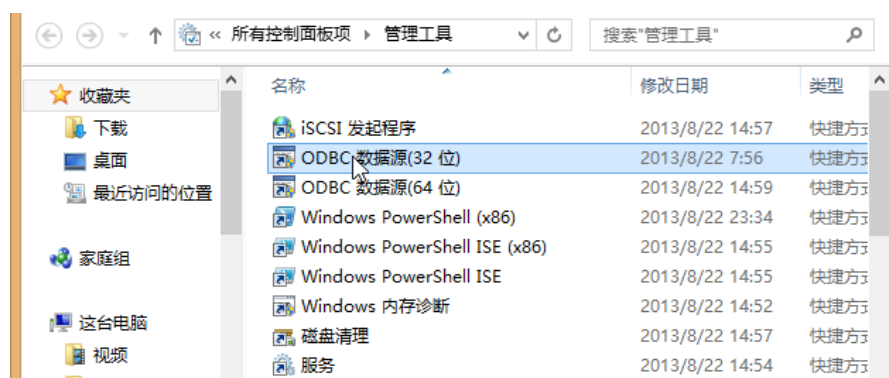


在利用数据库工具包操作数据库之前,需要先连接数据库。数据库文件有两种格式,老版本: .mdb; 新版本: .accdb; 连接数据库的方法有以下三种。

##### (1) .mdb 数据库的连接

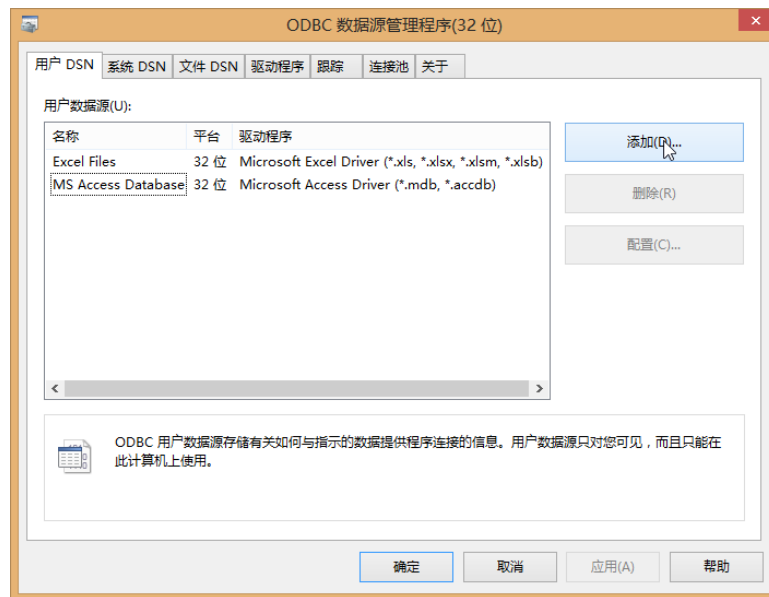
###### ① 利用 DSN 连接数据库

a 打开 Windows 控制面板/管理工具。如图?所示。



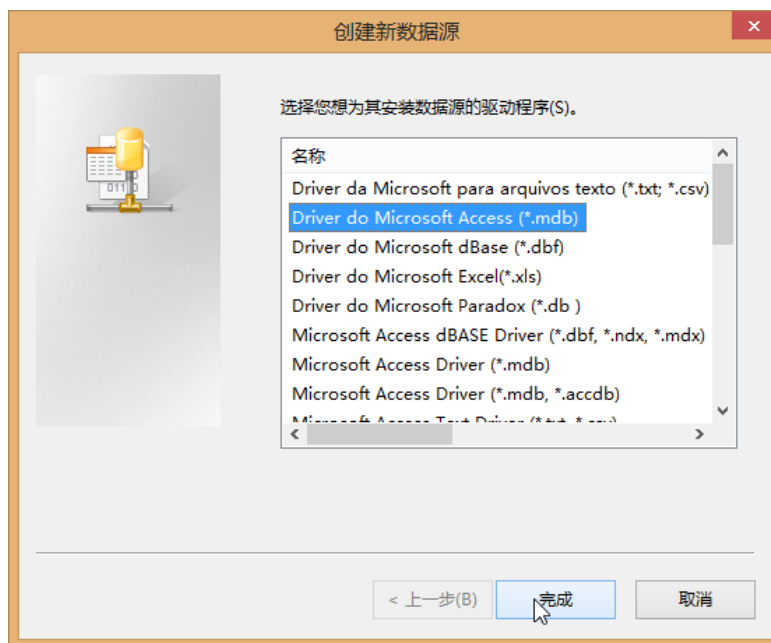
图？

b 在此选择“用户 DSN”，单击“添加”按钮。如图？所示。



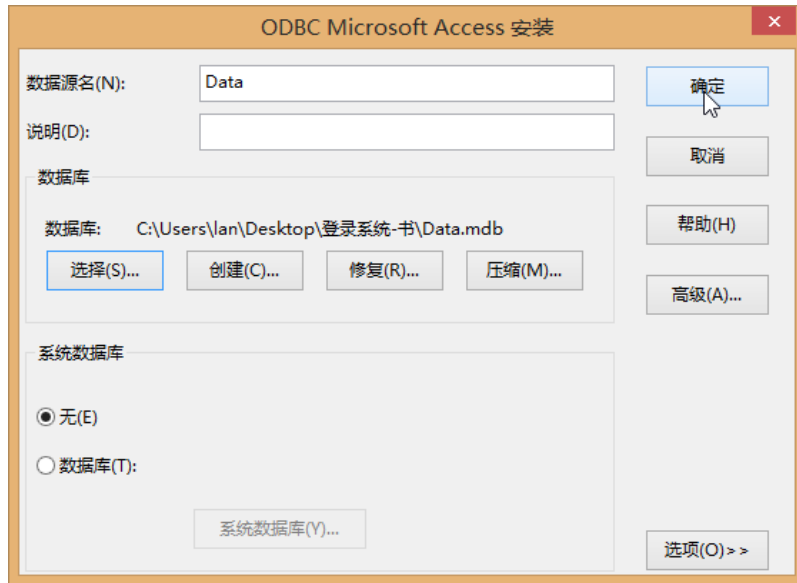
图？

c 在弹出的“创建新数据源”对话框中选择“Driver do Microsoft Access(\*.mdb)”，然后单击“完成”按钮。如图？所示。



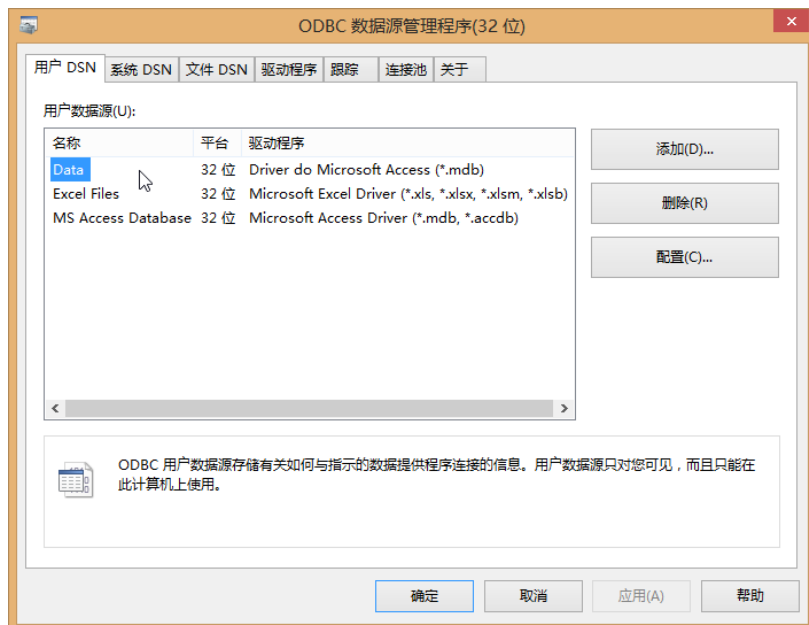
图？

d 随后弹出“ODBC Microsoft Access 安装”对话框，输入“数据源名”，如“Data”，然后单击“选择”按钮，选择建立好的 Data.mdb 数据库文件，单击“确定”按钮。如图？所示。



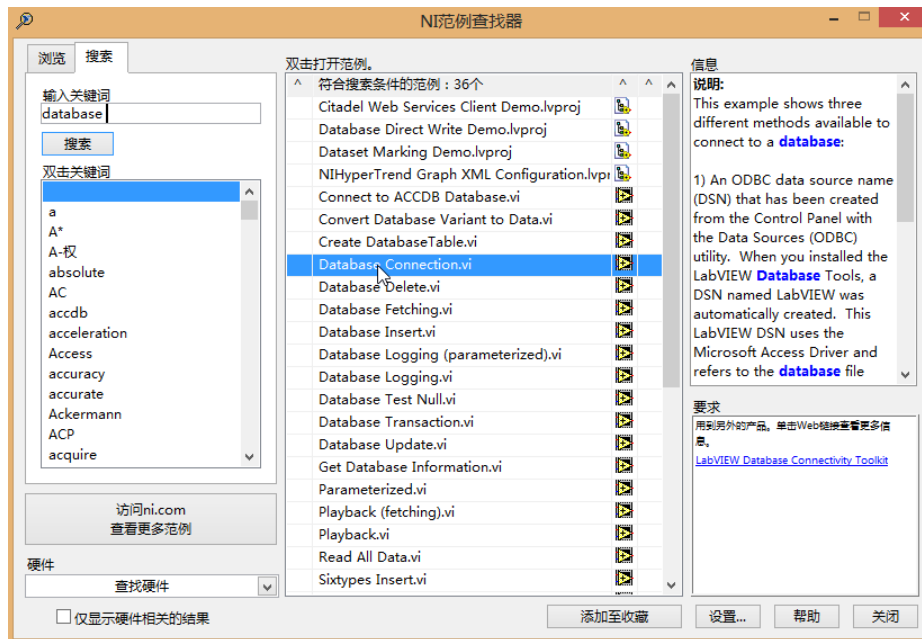
图?

e 完成以上配置后，就可以在“用户 DSN”页面下看到新创建的名为 Data 的 DSN 了。单击“确定”按钮可完成 DSN 的建立。如图? 所示。



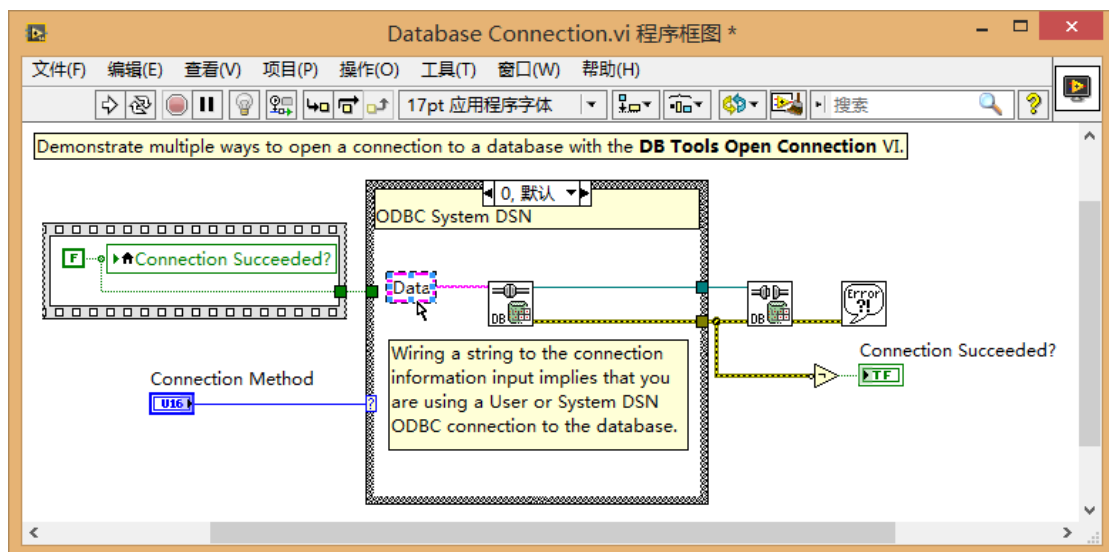
图?

f 在 LabView 范例查找器中，输入关键字“database”，然后选择“database connection.vi”。如图? 所示。



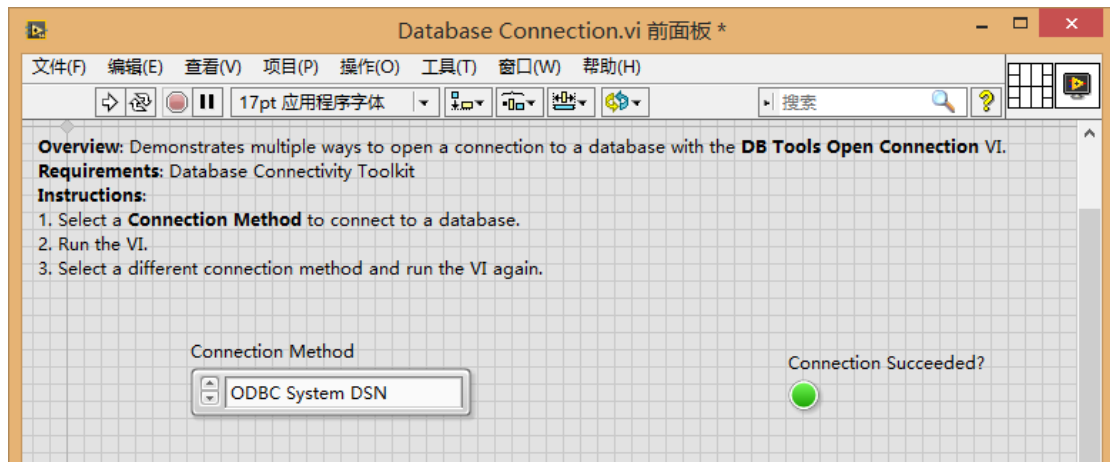
图?

g 打开 VI 后，将程序框图中【条件结构】中的“0”分支的【字符串常量】的字符串改成“Data”。如图? 所示。



图?

h 转至程序前面板，在【枚举】下拉菜单中选择“ODBC System DSN”，运行程序，“Connection Succeeded?”绿灯亮起，则连接成功。如图? 所示。



图？

② 利用 UDL 连接数据库(本教程以此为例)

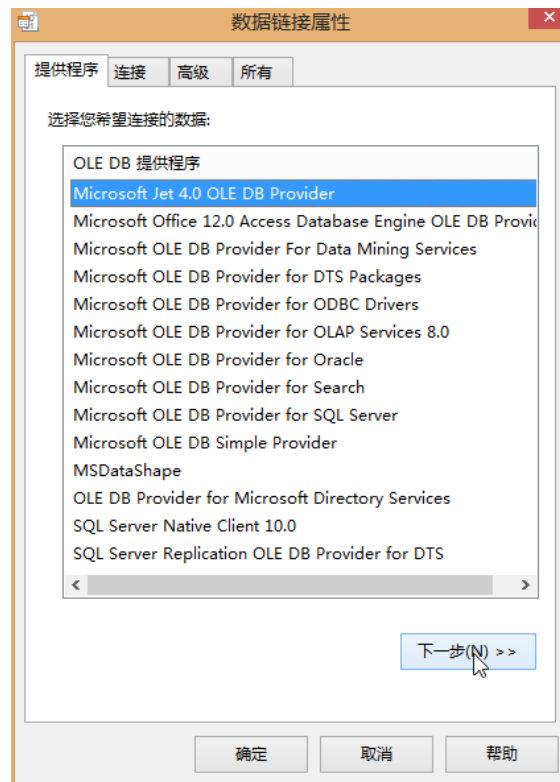
a 在 Labview 的工具栏中选择<工具>/<Create Data Link...>。如图？所示。



图？

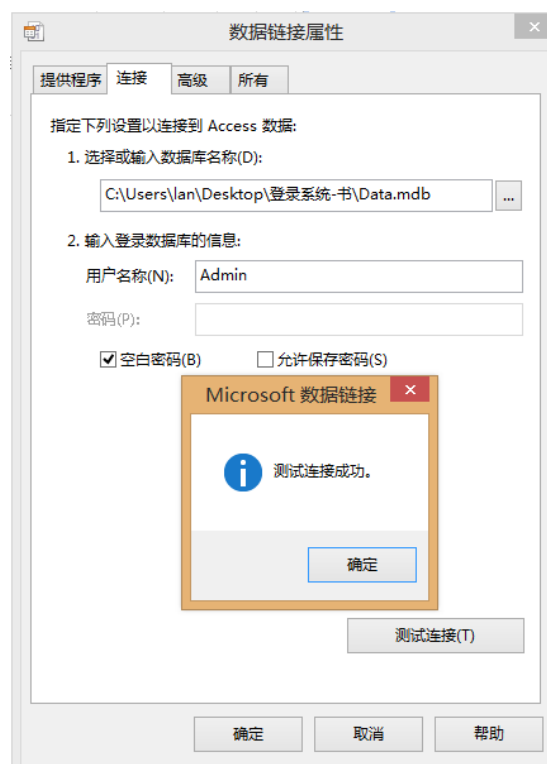
b 弹出“数据链接属性”对话框。在连接的数据:选择“Microsoft Jet 4.0 OLE DB Provider”选项，然后单击“下一步”。如图？所示。





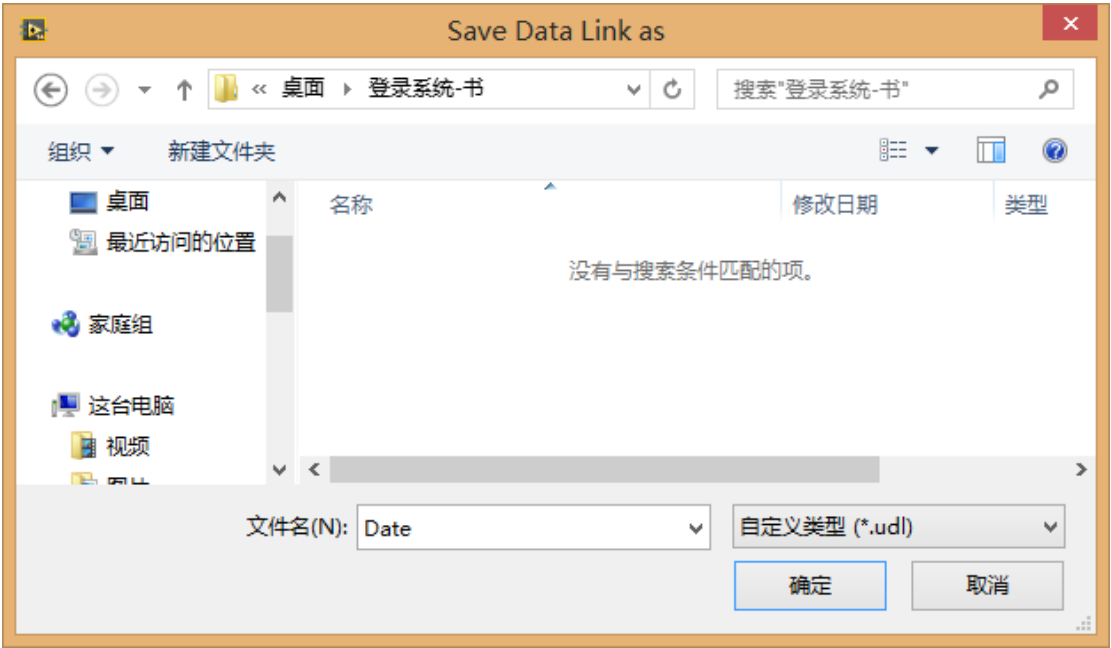
图？

c 选择建立好的 Data.mdb 数据库文件，单击“测试连接”按钮。弹出“测试成功对话框”。如图？所示。



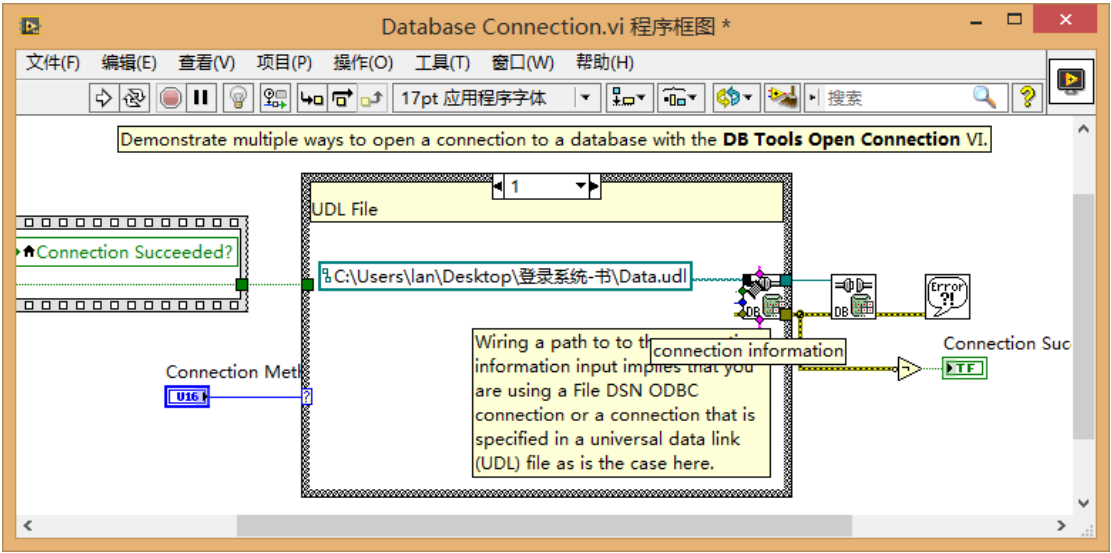
图？

d 点击“确认”按钮后，弹出“Save Data Link as”对话框。选择先前创建 Data.mdb 文件的位置，将“Data”设为文件名。点击“确认”按钮即可。如图? 所示。



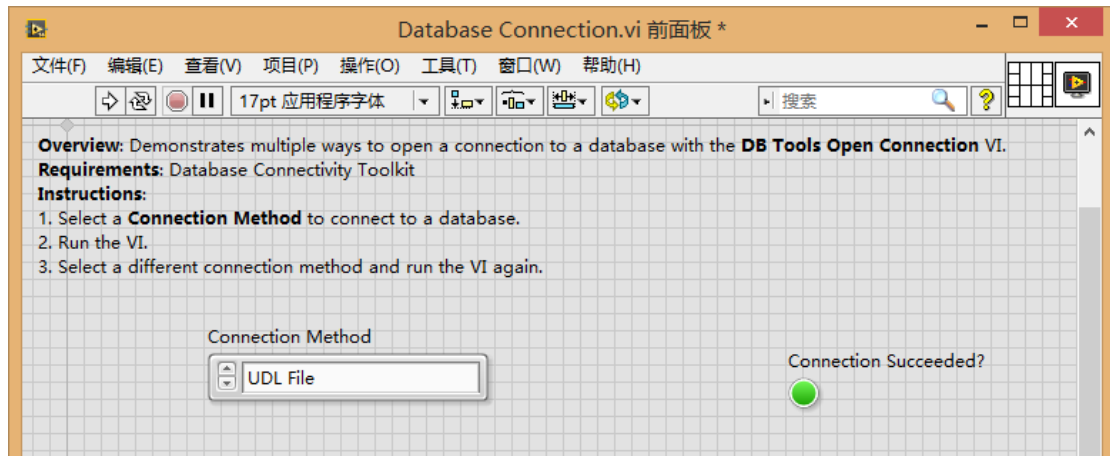
图?

e 在“database connection.vi”的程序框图中，将【条件结构】“1”分支中“DB Tools Open Connection”的“connection information”输入端改为 Data.udl 的文件路径。如图? 所示。



图?

f 转至前面板，在【枚举】下拉菜单中选择“UDL File”，运行程序，“Connection Succeeded?”绿灯亮起，则连接成功。如图? 所示。



图？

(2) .accdb 数据库连接(创建数据库文件时选择 Data.accdb)

① 需要安装 AccessDatabaseEngine.exe (文件夹中附有)也可自行下载, 下载地址:

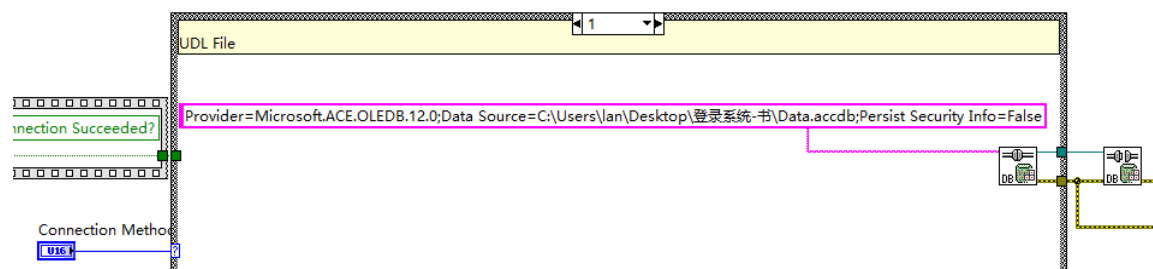
<http://www.microsoft.com/en-us/download/details.aspx?id=13255MicrosoftsAccess>

选择 DatabaseEngine2010 Redistributable 然后就可以访问.accdb 格式的数据库了。

② 访问的字符串格式为: Provider=Microsoft.ACE.OLEDB.12.0; DataSource= (.accdb 格式数据库文件的地址例如: C:\Users\lan\Desktop\登录系统-书\Data.accdb) ;Persist Security Info=False

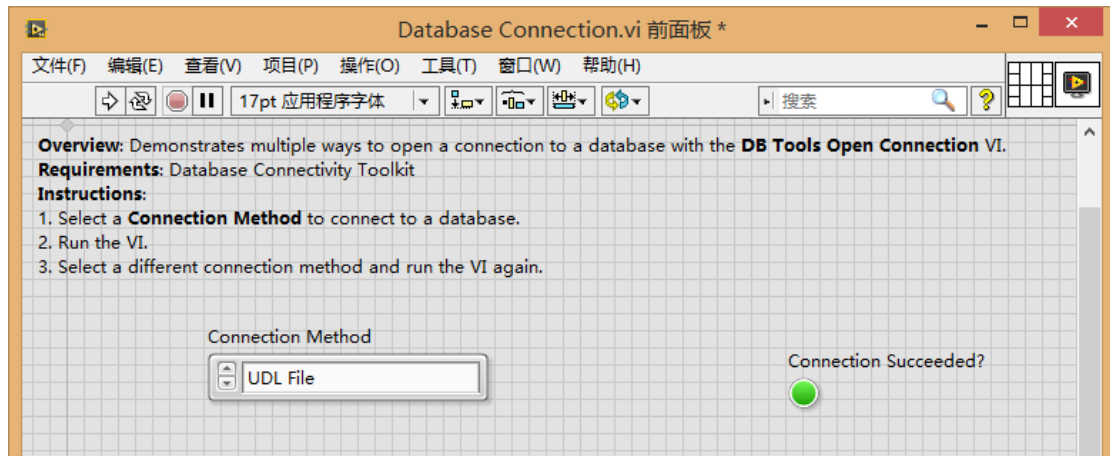
例如: Provider=Microsoft.ACE.OLEDB.12.0;Data Source=C:\Users\lan\Desktop\登录系统-书\Data.accdb;Persist Security Info=False

③ 在“database connection.vi”的程序框图中, 将【条件结构】“1”分支中“DB Tools Open Connection”的“connection information”输入端改为如②所示的字符串路径。如图？所示。



图？

④ 转至程序前面板, 在【枚举】下拉菜单中选择“UDL File”, 运行程序, “Connection Succeeded?”绿灯亮起, 则连接成功。如图？所示。




图?

### 3.2 创建表格



以下是在 LabView 中创建表格的方法。如对 Access 数据库熟悉，可在 Access 中创建以下信息，效果一致。

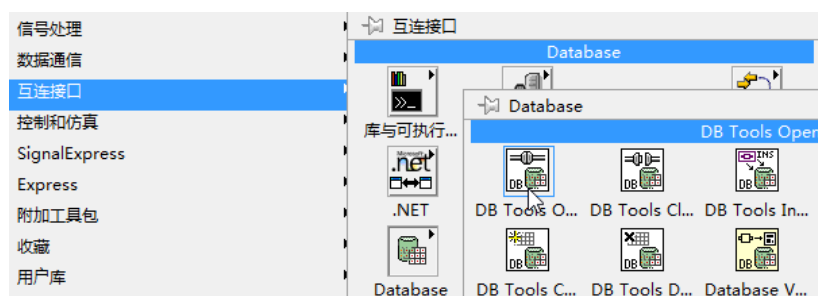
(1) 新建一个 VI 保存至 Data.udl 同一目录下并名为“创建表格”，在程序框图中依

次创建【DB Tools Open Connection】 (右击选择【互连接口】/【Database】

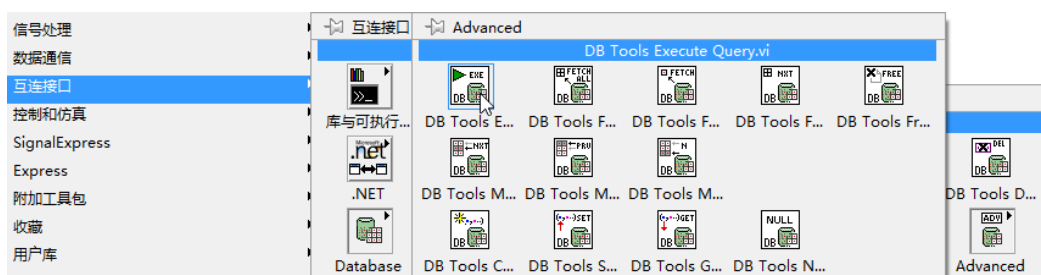
/【DB Tools Open Connection】如图? 所示)、【DB Tools Execute Query】 (【互连接口】/【Database】/【Advanced】/【DB Tools Execute Query】如图? 所示)、

【DB Tools Fetch Recordset Data】、【DB Tools Free Object】、【DB Tools

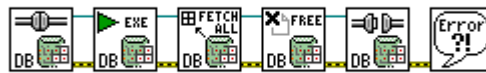
Close Connection】 以及一个【简易错误处理器】 (右击选择【编程】/【对话框与用户界面】/【简单错误处理器】)依次连接，结果如图? 所示。



图?

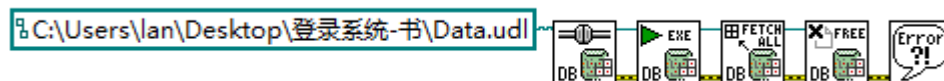


图?



图?

(2) 在【DB Tools Open Connection】“connection information”处右击选择<创建>/<常量>。将 Data.udl 文件路径输入其中。结果如图? 所示。

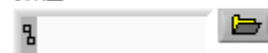


图?

考虑到后面的程序都需要使用 Data.udl 文件的路径。故创建一个子 VI。

a 创建一个新 VI 保存至 Data.udl 同一目录下并名为“UDL 文件路径”。在前面板创建【文件路径输入控件】(【新式】/【字符串与路径】/【文件路径输入控件】)

路径

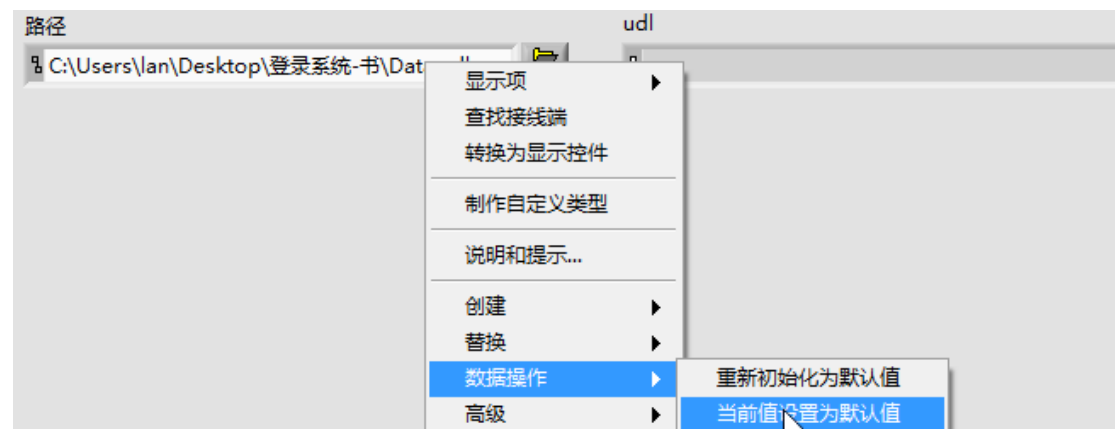


以及【文件路径显示控件】，更改标签为“udl”。【文件路径输入控件】路径选择为 Data.udl 文件路径。结果如图? 所示。



图?

b 在【文件路径输入控件】上右击选择<数据操作>/<当前值设置为默认值>。结果如图? 所示。



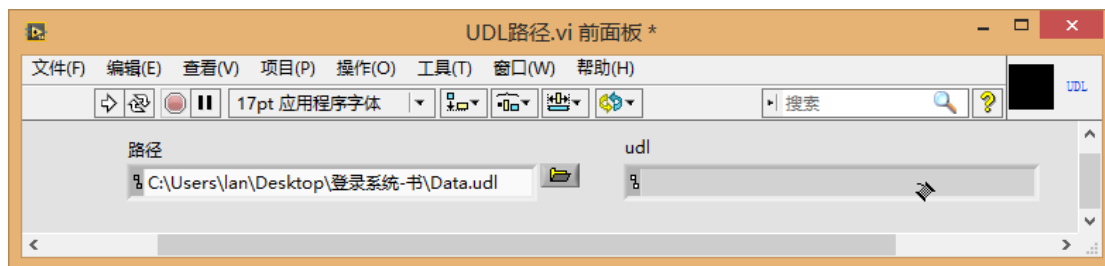
图?

c 在程序框图中将两个控件相连接。结果如图? 所示。



图?

d 在前面板编辑连线模式，选择单线的连接模式。并于【文件路径显示控件】建立连接。结果如图? 所示。



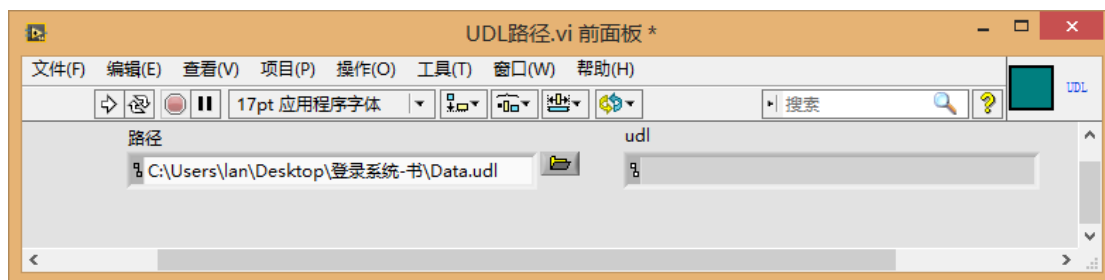
图?

e 修改图标，在图表上右击选择<编辑图标...>，使用快捷键<Ctrl+u>清除原有图层。在“图标文本”项目下的“第一行文本”中输入“UDL”。结果如图? 所示。



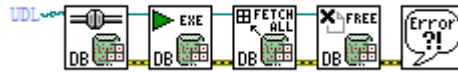
图?

修改完成后，如图? 所示。保存 VI。



图?

f 在“创建表格.vi”的程序框图中进行修改，右击选择<选择 VI...>选择“UDL 路径”。则程序可简化成如图? 所示。



图?

g 在【DB Tools Execute Query】控件“SQL query”处右击创建一个【字符串常量】。并在内输入以下内容。

```
create table 用户信息 (用户名 varchar(50),密码 varchar (50),用户权限  
varchar(50),上次登录时间 varchar(50),ID int identity);
```

其中“用户信息”即为表名。

“用户名”、“密码”、“用户权限”、“上次登录时间”、“ID”都为列名。

“Varchar(50)”表示字符的长度限制为 50。

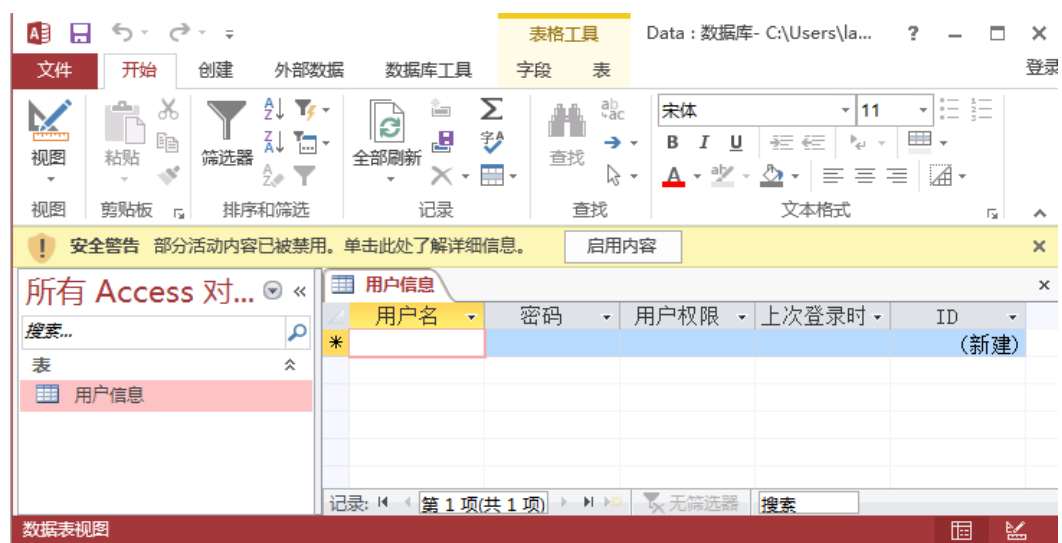
“Int”表示整型。“identity”为递增，从 1 开始，递增量为 1。结果如图? 所示。

```
create table 用户信息 (用户名 varchar(50),密码 varchar (50),用户权限 varchar(50),上次登录时间 varchar(50),ID int identity);
```



图?

保存并运行 VI，打开 Data.mdb 文件如图? 所示，即可看见，创建的信息。



图?

### 3.3 创建子 VI

先前介绍过在登录界面里有许多布尔控件对应有不同的触发动作。也就意味着对应着一系列不同的子 VI。(考虑到主程序的不同，因此不对主程序进行说明)

#### 3.3.1 修改密码

用于对登录用户密码的修改。

思路：需要进行 3 次判断过程。

第一次判断：输入原始密码与数据库中的密码进行比较，判断是否存在。存在进行第二次判断，否则不进行。

第二次判断：判断新密码是否不为零。不为零进行第三次判断，否则不进行。

第三次判断：判断新密码和确定新密码的值是否相等。相等输出最终密码，否则不进行。



前面板如图?所示。

修改密码

输入原始密码

新密码

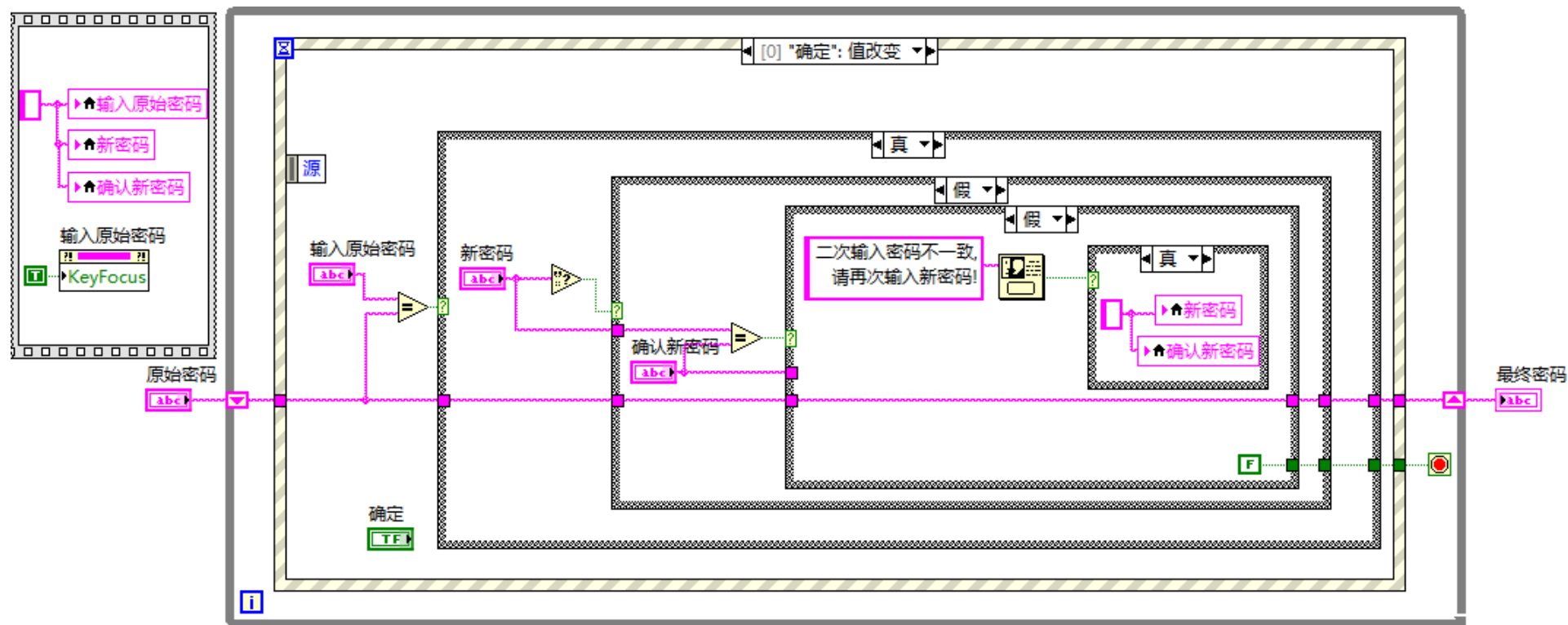
确认新密码

 确定

 退出

图?

程序框图如图? 所示。



图?

创建过程

- ① 在前面板创建三个【字符串输入控件(银色)】(右击选择【银色】/【字符串与路径】/【字符串输入控件(银色)】)依次将标签改名

为“原始密码”、“输入原始密码”、“新密码”、“确认新密码”并调整位置。结果如图? 所示。(因为【原始密码】只作为判断条件，并不作为实际控件操作，故不列出)

图?

② 创建【确定按钮(银色)】和【取消按钮(银色)】，将标签修改为“确定”、“取消”并隐藏标签(在控件上右击<显示项>/标签)。结果如图? 所示。

图?

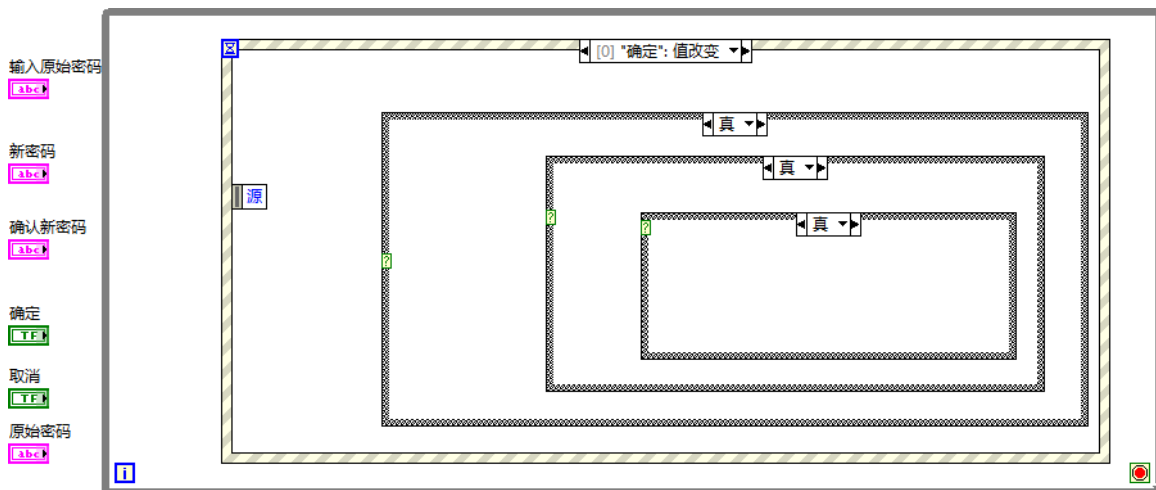
③ 在程序框图中创建一个【While 循环】和【时间结构】。结果如图? 所示。





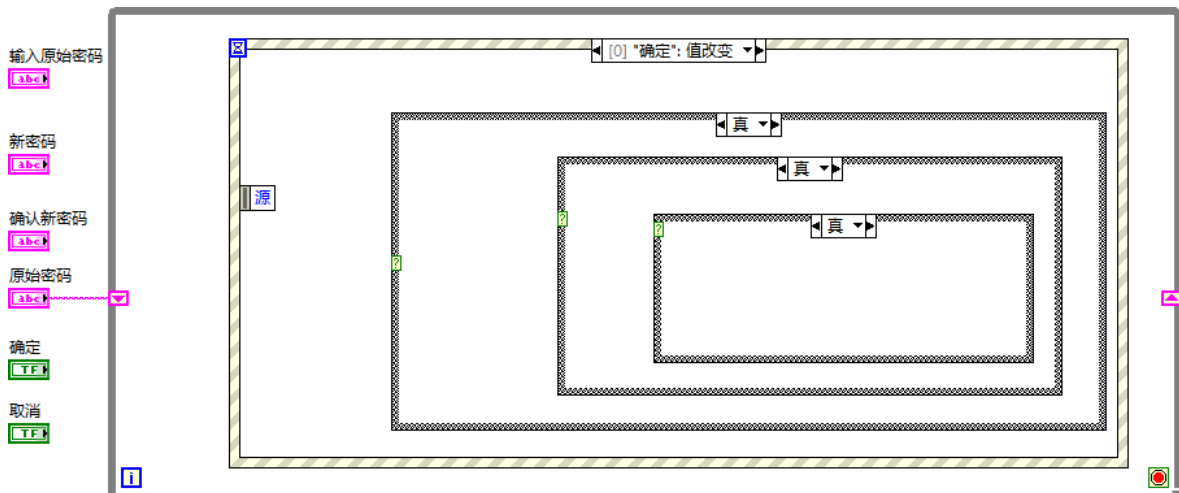
图?

⑤ 在【事件结构】的“0”分支下，创建三个【条件结构】。结果如图? 所示。



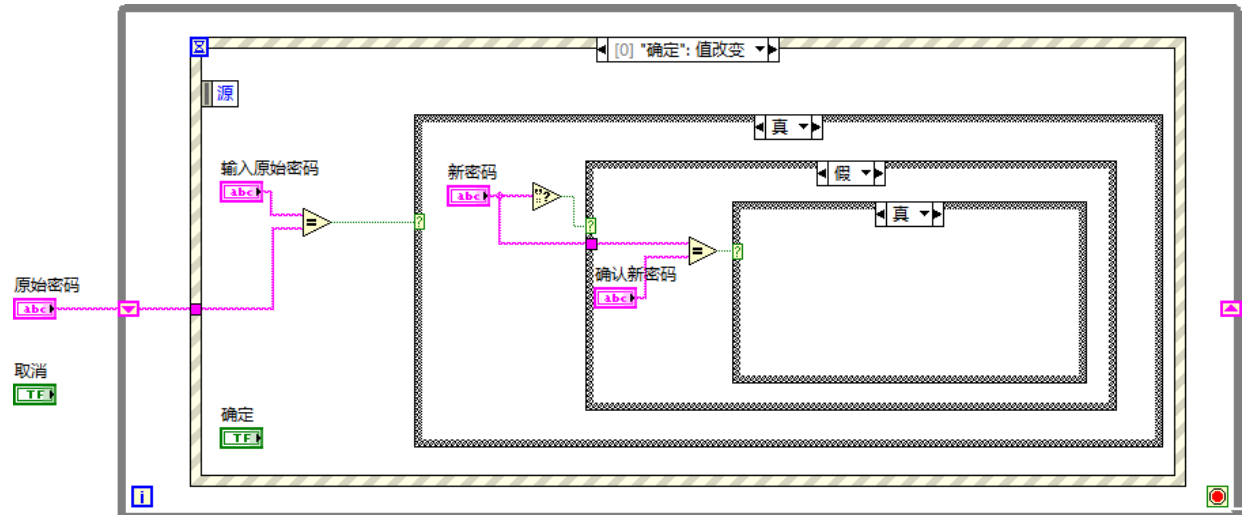
图?

⑥ 在【While 循环】上创建<添加移位寄存器>(在【While 循环】上右击选择<添加移位寄存器>), 并于【原始密码】相连接。结果如图? 所示。

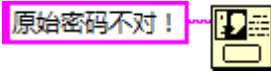



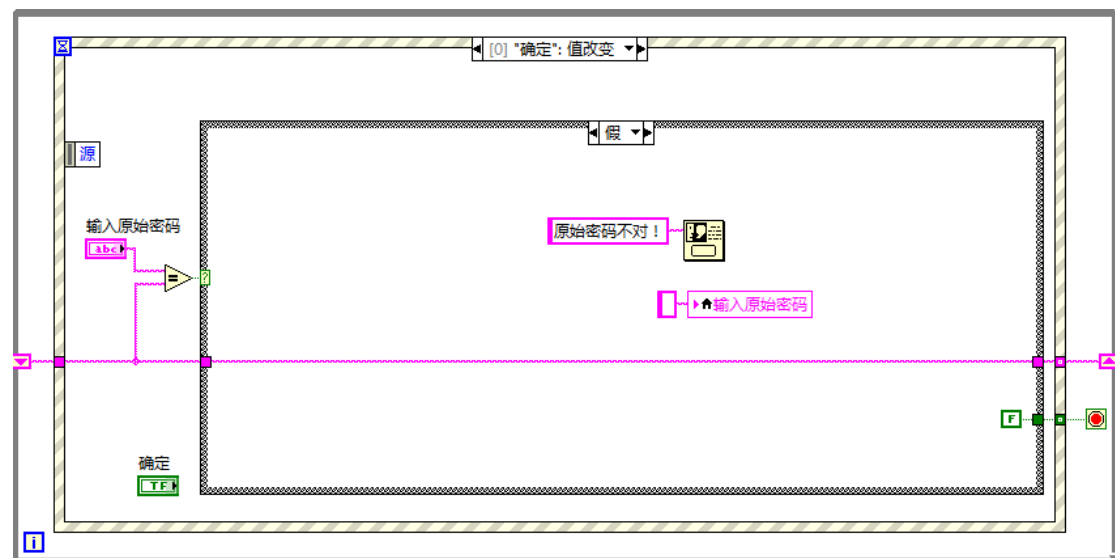
图？

- ⑦ 创建两个【等于？】和一个【空字符串/路径？】 (【编程】/【比较】/【空字符串/路径？】)建立如图？所示的连接。(注意【条件结构】的分支)





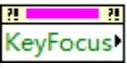
图？

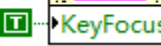
- ⑧ 在第一个【条件结构】的“假”分支中创建一个【单按钮对话框】并添加一个【字符串常量】，内容为“原始密码不对！”。如  所示。为【输入原始密码】创建一个【局部变量】(在【输入原始密码】控件上右击选择<创建>/<局部变量>)并赋予一个【字符串常量】，如  所示。再创建一个【假常量】与【While 循环】的“循环条件”相连。将“移位寄存器”两端相连。结果如图？所示。

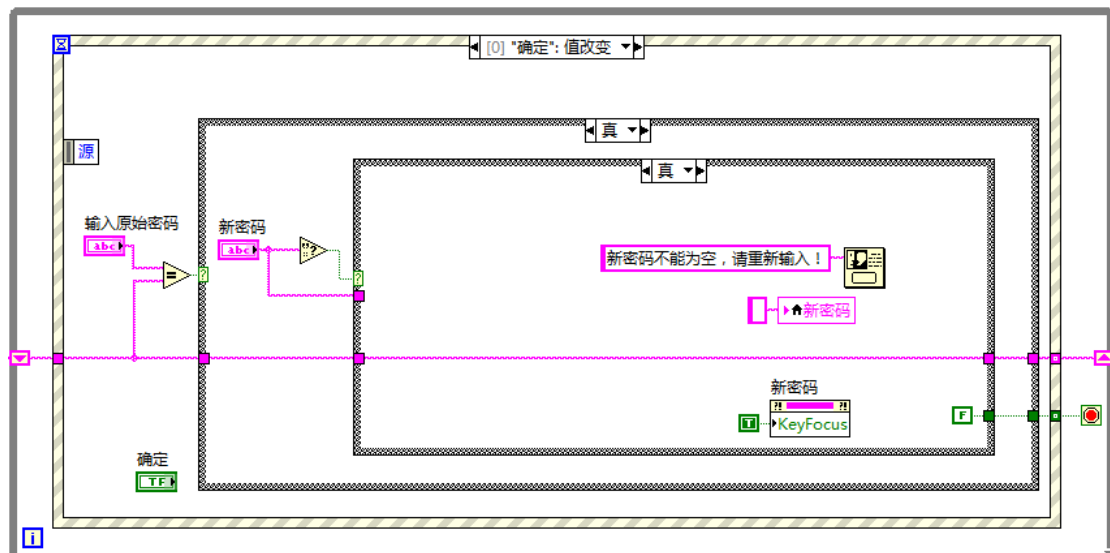


图?

⑨ 在第二个【条件结构】的“真”分支中创建一个【单按钮对话框】并添加一个【字符串常量】，内容为“新密码不能为空，请重新输入！”，如  所示。为【新密码】创建一个【局部变量】，并赋予一个【字符串常量】，如  所示。再为【新密码】创建一个【键选中属性节点】

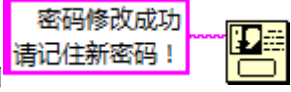
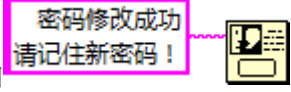
（在【新密码】上右击选择<创建>/<属性节点>/<键选中>），在【键选中属性节点】上右击选择<全部转化为写入>，并赋值真常量。如  所示。再创建一个【假常量】与【While 循环】的“循环条件”相连。

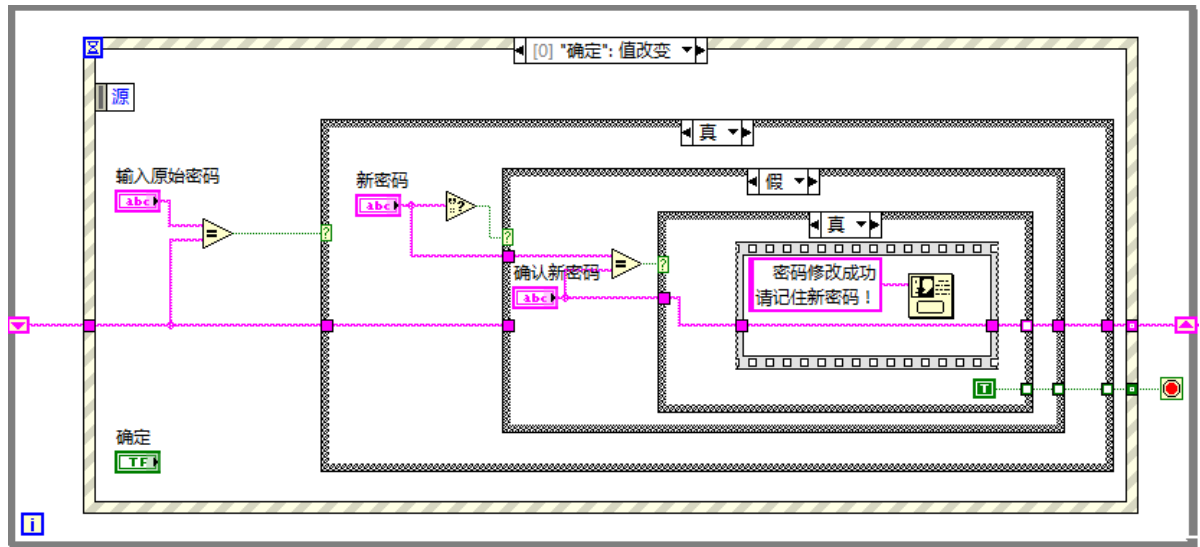
将“移位寄存器”两端相连。结果如  所示。



图?

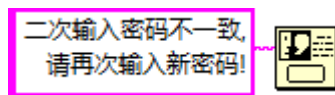
⑩ 在第三个【条件结构】的“真”分支中创建一个【单按钮对话框】并添加一个

【字符串常量】，内容为“密码修改成功，请记住新密码！”，如  所示。创建一个【顺序结构】将【点按钮对话框】和【字符串常量】放入。将【确认新密码】的值与“移位寄存器”末端相连，创建一个【真常量】与【While 循环】的“循环条件”相连。结果如  所示。



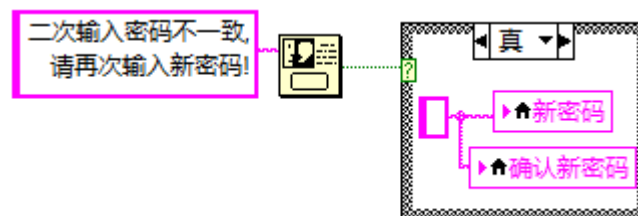
图?

- ⑪ 在第三个【条件结构】的“假”分支中创建一个【单按钮对话框】并添加一个【字符串常量】，内容为“二次输入密码不一致,请再次输入新密码!”如



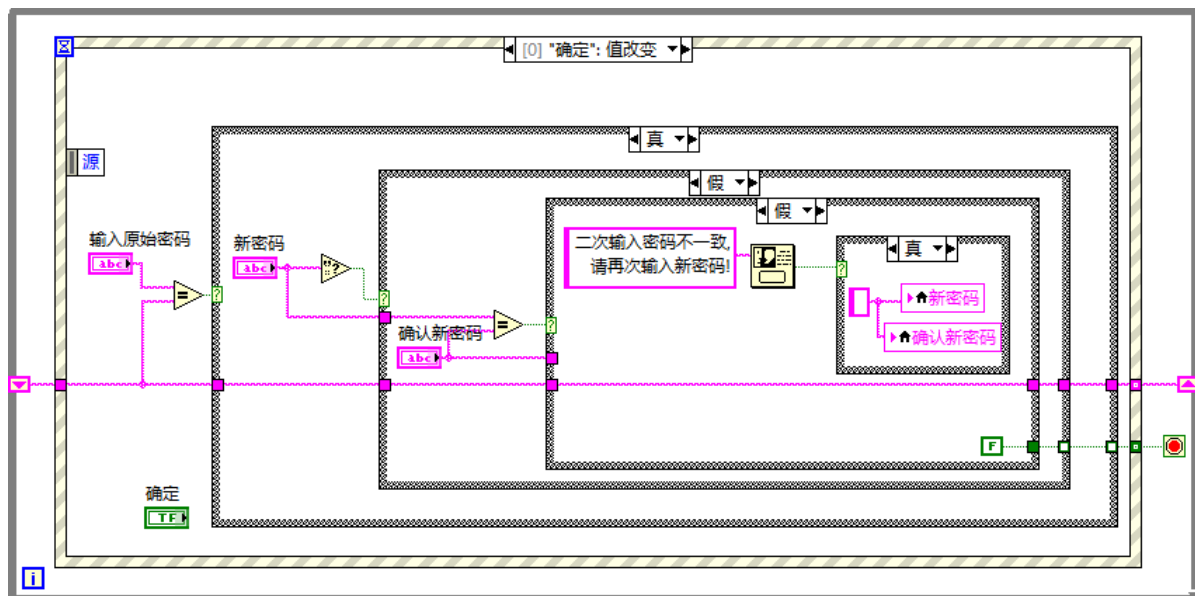
所示。再创建一个【条件结构】其“真”分支为【新密码】

和【确认新密码】的【局部变量】都赋值【字符串常量】。如图? 所示。其“假”分支为空。再在第三个【条件结构】中创建一个【假常量】，将“移位寄存器”两端相连。结果如图? 所示。



图?





图?

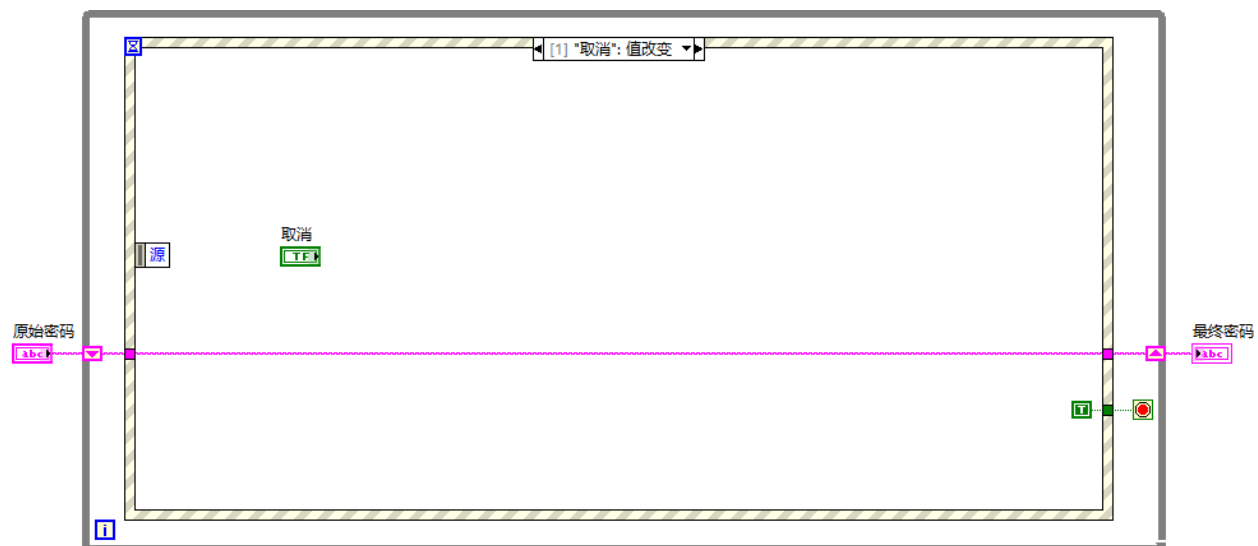
⑫ 在前面板创建【字符串显示控件(银色)】并将标签改为“最终密码”。如

最终密码

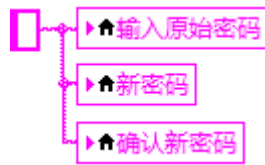


所示。在程序框图中将其与【While 循环】“移位寄存器”的末端相连。


⑬ 在【事件结构】的“1”分支中，将“移位寄存器”两端相连。并创建一个【真常量】与【While 循环】的“循环条件”相连。结果如图? 所示。

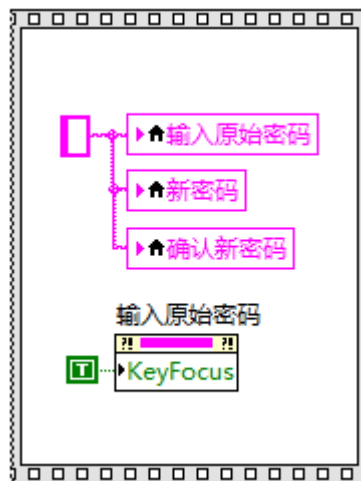


⑭ 创建【输入原始密码】、【新密码】、【确认新密码】的【局部变量】，都赋值【字符串常量】。结果如图? 所示。



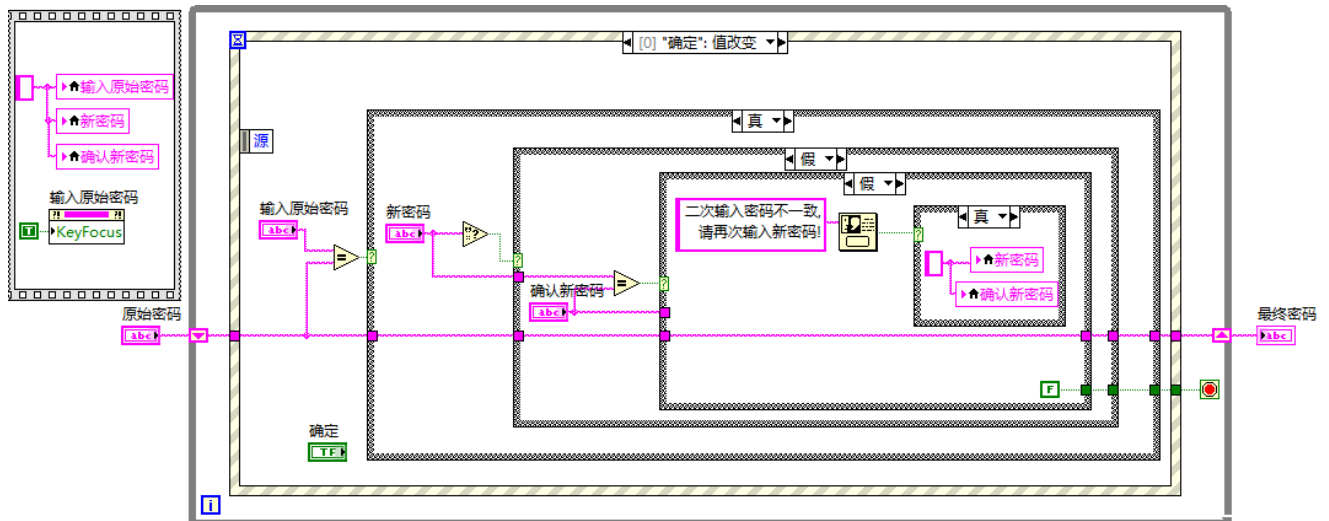
图?


再为【输入原始密码】创建【键选中属性节点】，并赋值【真常量】。如  所示。创建【顺序结构】将两部分放入。结果如图?所示。

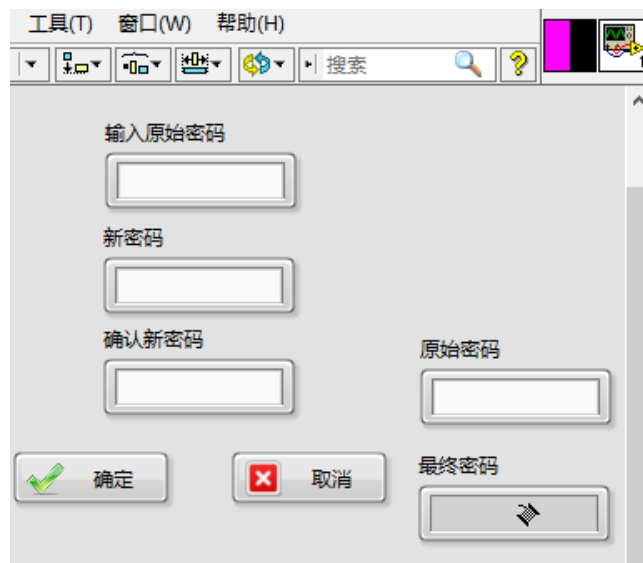


图?

最终结构如图?所示。

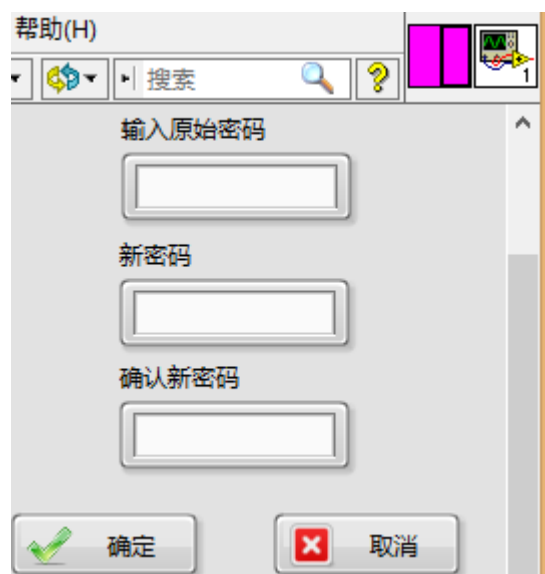


⑮ 编辑程序的连线模式。选择  模式。连线框左半边与【原始密码】相连，连线框右半边与【最终密码】相连。结果如图?所示。



图？

连线完成后，在【原始密码】上右击选择<高级>/<隐藏输入控件>，对【最终密码】也进行此操作。结果如图？所示。



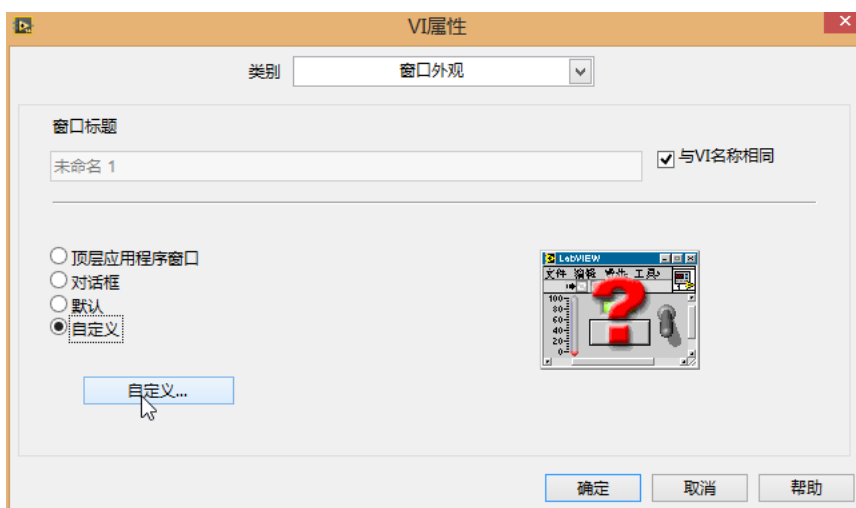
图？

⑩ 编辑 VI 图标。与前面操作一致。不同的是输出第一、第二行文本后。双击如图？所示边框即可出现黑框。



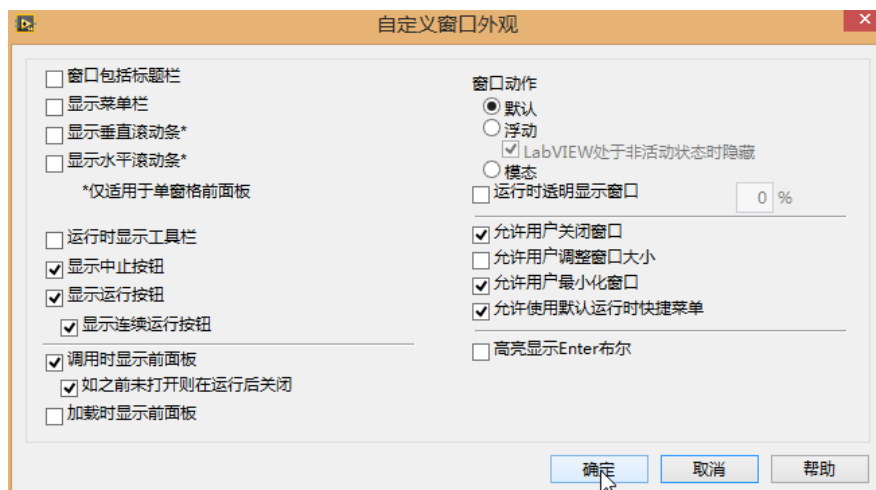
图？

⑰ 编辑 VI 属性。在 VI 图标上右击，选择<VI 属性>，在“类别”处选择“窗口外观”，选择“自定义”。如图？所示。



图？

将不必要的选项勾选掉，如图？所示。



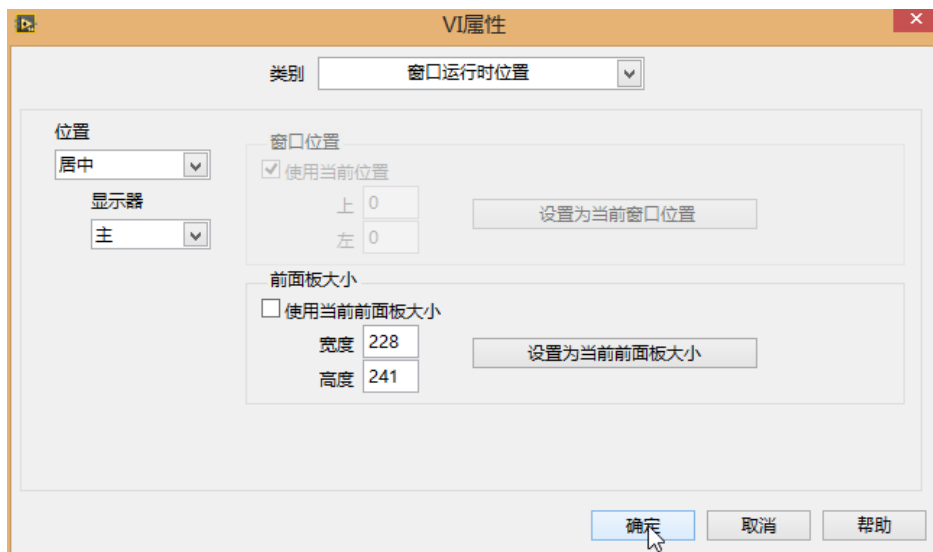
图？

⑮ 编辑好前面板后，将窗口调成如图？所示。



图？

然后在 VI 图标上右击，选择<VI 属性>，在“类别”处选择“窗口运行时位置”，“位置”选择“居中”，“显示器”选择“主”，“前面板大小”将“使用当前前面板大小”勾去即可。结果如图？所示。(若觉得位置没有设置好，可重新设置后。再返回点击“设置为当前前面板大小”即可)

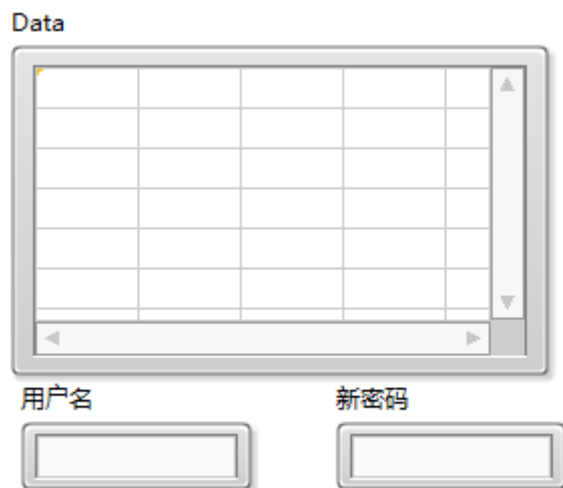


图？

⑯ 保存 VI 至 Data.mdb 在同一级目录下，并改名为“修改密码”。

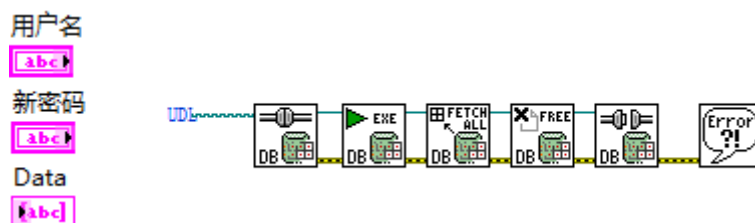
### 3.3.2 更新密码

① 在前面板创建两个【字符串输入控件(银色)】，分别将标签改为“用户名”和“新密码”。创建一个【表格(银色)】(右击选择【银色】/【列表、表格和树】/【表格(银色)】)并将标签改为“Data”。在【表格(银色)】上右击选择<转化为显示控件>。结果如图? 所示。



图?

② 在程序框图能创建如图? 所示结构。(先前详细讲解过，故不在赘述)




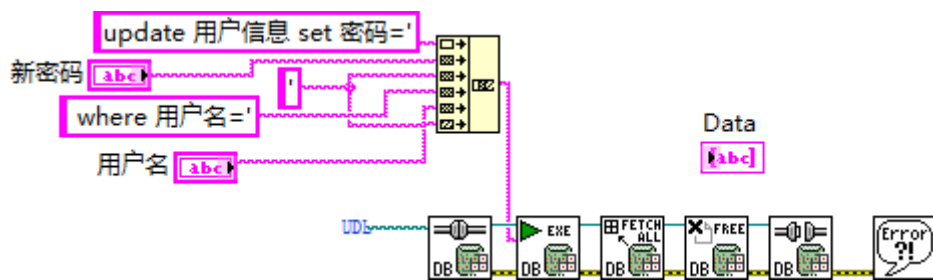
图?

③ SQL 语句中的更新语句为:





```
update 用户信息 set 密码='**' where 用户名='**'
```

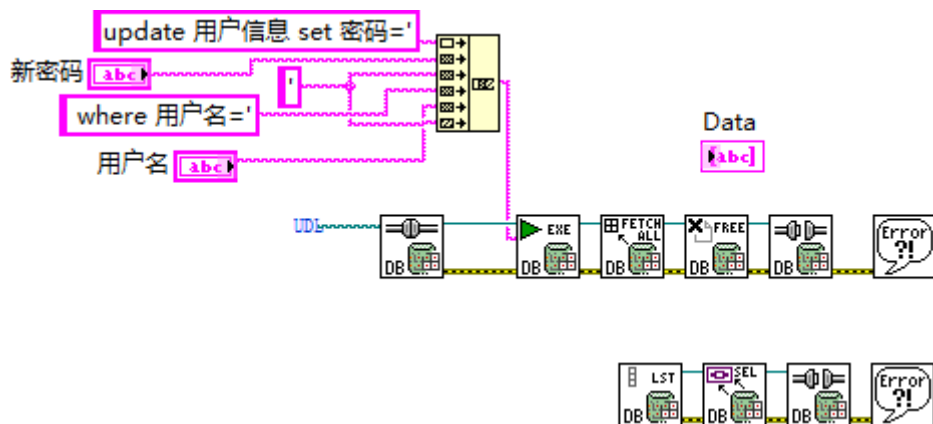
即为更新“用户信息”表中与“用户名”列相同行上“密码”列中的数据。

根据上述 SQL 语句，创建【连接字符串】, 并创建相应的【字符串常量】，将最终的输出端与【DB Tools Execute Query】控件的“SQL query”端相连。结果如图? 所示。



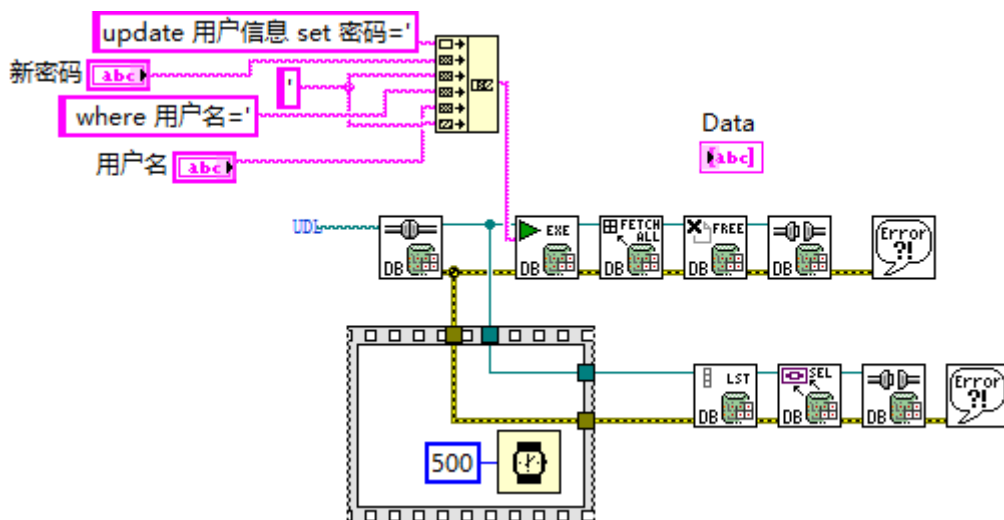
图？

- ④ 创建【DB Tools List Columns】 (右击选择【互联接口】/【Database】/【Utility】/【DB Tools List Columns】)、【DB Tools Select Data】、【DB Tools Close Connection】 以及一个【简易错误处理器】, 依次连接。结果如图？所示。




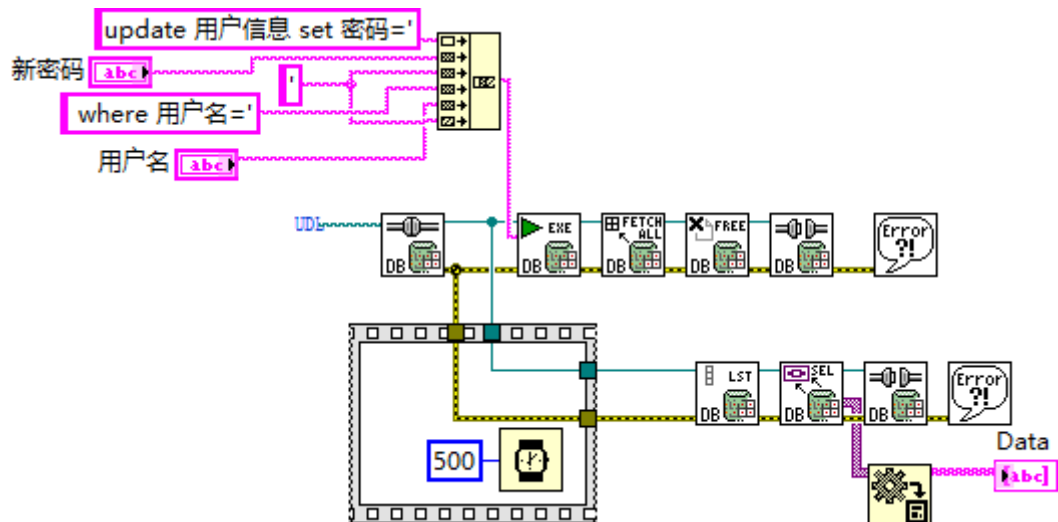
图？

- ⑤ 创建一个【顺序结构】和【等待 ms】设置等待时间为 500ms, 连接相关控件。结果如图？所示。



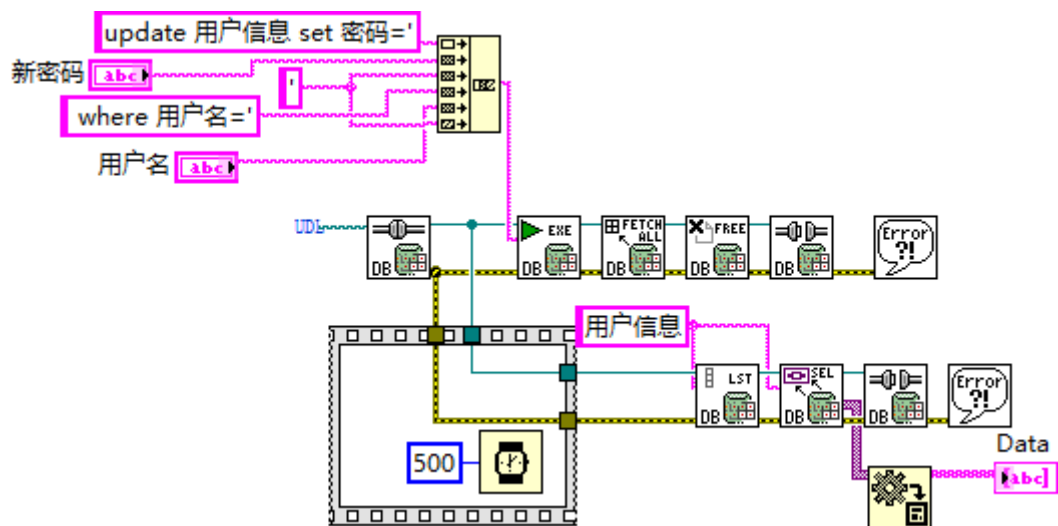
图？

⑥ 创建【变体至数据转换】 (【簇、类与变体】/【变体】/【变体至数据转换】), 输入端与【DB Tools Select Data】连接, 输出端与【Data】连接。结果如图? 所示。




图？

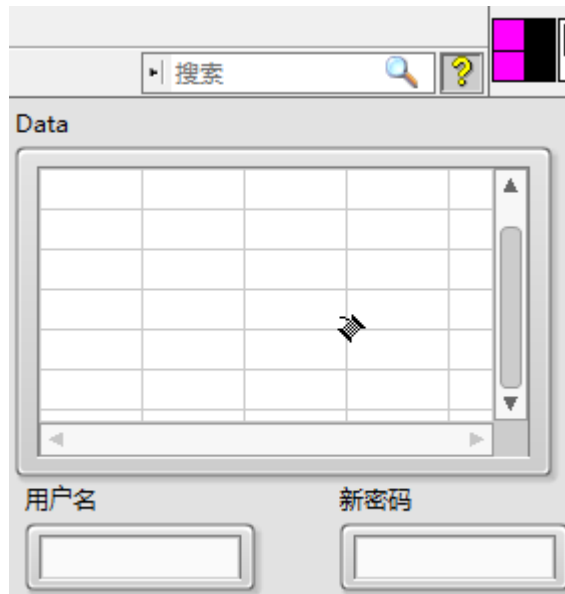
⑦ 为【DB Tools List Columns】和【DB Tools Select Data】的“table”端创建一个共用【字符串常量】并赋值为“用户信息”。结果如图?所示。



图？

⑧ 编辑连线模式，选择三线的连线模式。连线框左半边的上半部分连接【新密码】，连线框左半边的下半部分连接【用户名】。右半边连接【Data】。结果如图?所示。





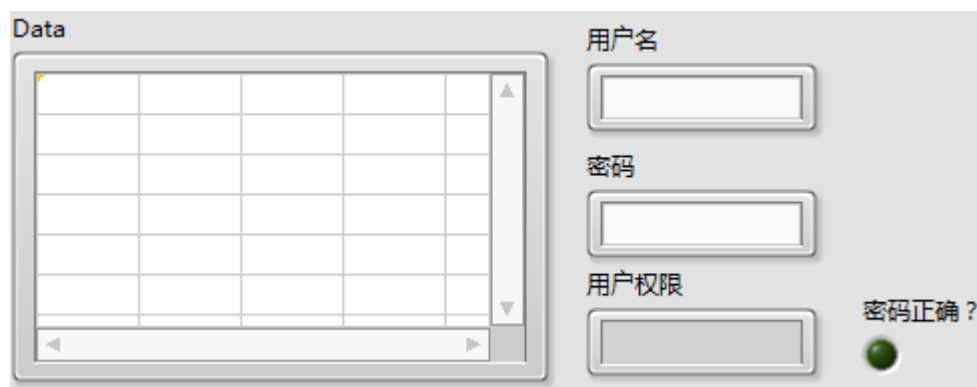
图?

⑨ 编辑图标，改名为“密码更新”。

⑩ 保存 VI 至 Data.mdb 在同一级目录下，并改名为“密码更新”。

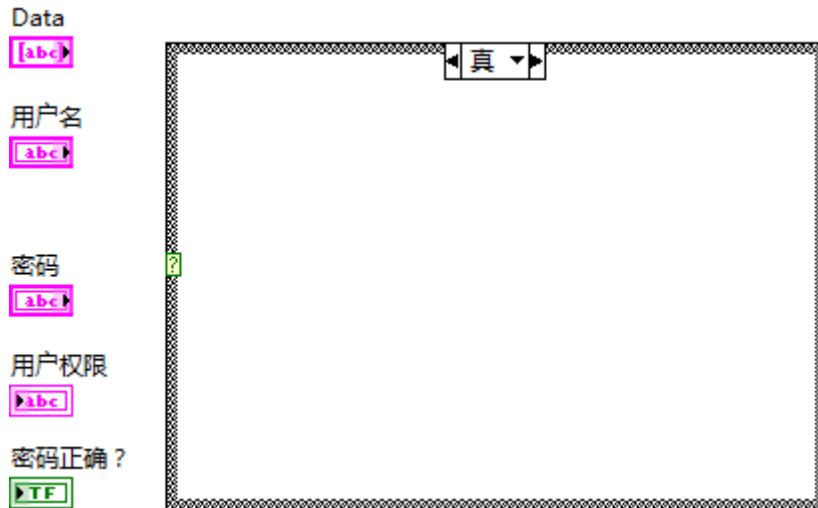
### 3.3.3 核对密码

① 在前面板创建一个【表格(银色)】将标签改为“Data”。创建两个【字符串输入控件(银色)】(将标签改名为“用户名”和“密码”)、一个【字符串显示控件(银色)】(将标签改名为“用户权限”)以及一个【圆形指示灯】(将标签改名为“密码正确? ”)。结果如图?所示。






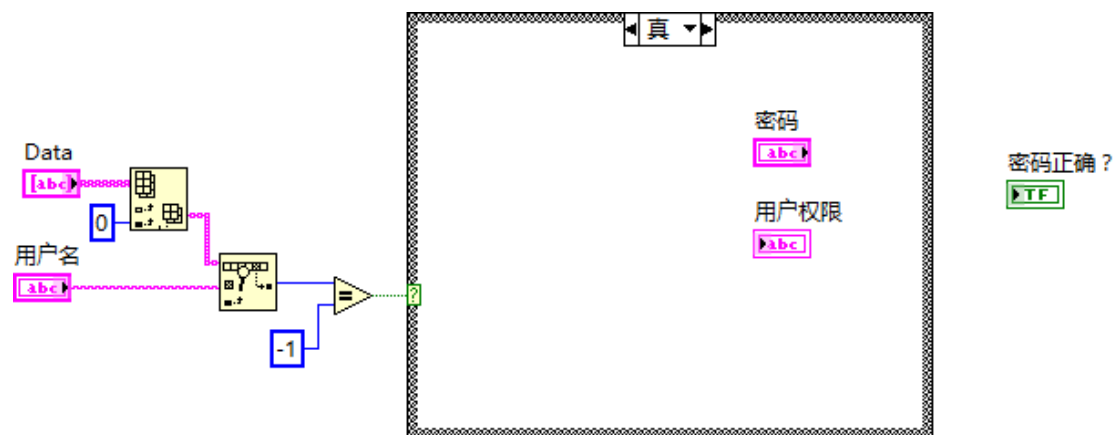
图?

② 在程序框图中创建一个【条件结构】。结果如图?所示。



图?

- ③ 创建【引索数组】 (【编程】/【数组】/【引索数组】)、【搜索一维数组】、【等于?】 以及两个【数值常量】。创建如图? 所示的连接。(该部分是为了判断输入的用户名是否存在数据中)



图?

- ④ 在【条件结构】的“真”分支中创建四个【引索数组】以及在“引索列”对应的【数值常量】和一个【等于?】。创建如图? 所示的连接。(该部分是为了在用户名存在的基础上核对密码以及输出用户权限)

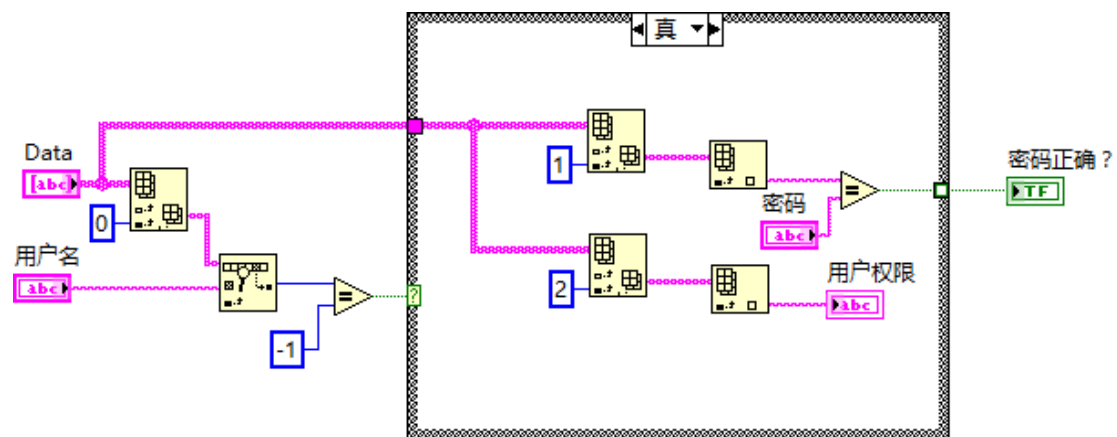


图7

⑤ 在【条件结构】的“假”分支中，创建一个【假常量】，与【密码正确？】连接。结果如图8所示。

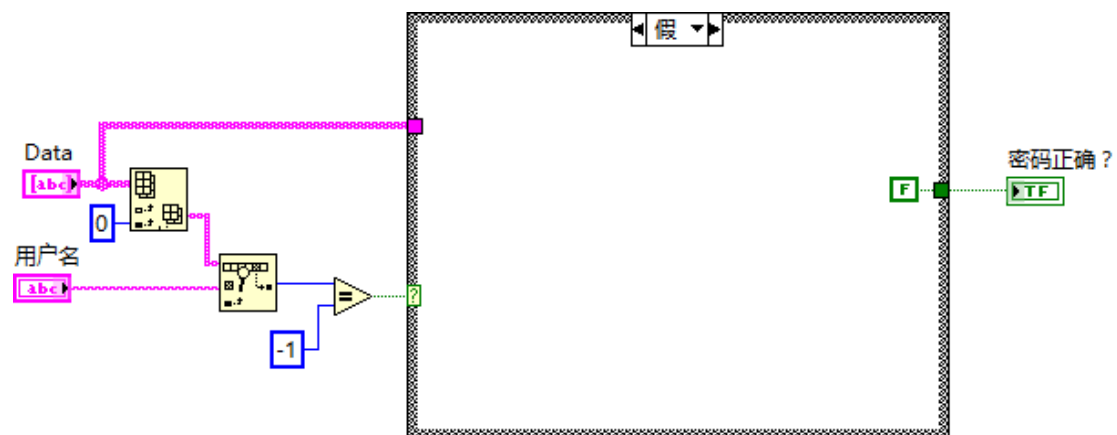

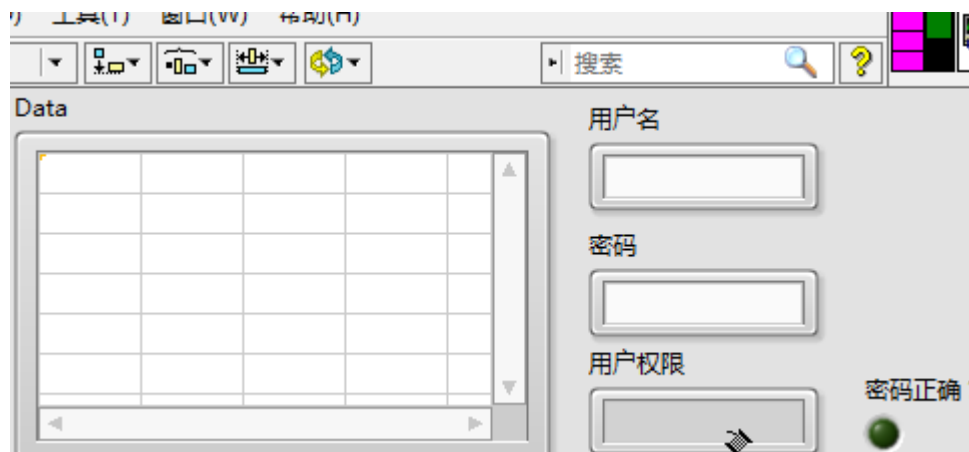


图8

⑥ 编辑连线模式。选择  连线模式。连线框左半边依次连接【Data】、【用户名】、【密码】。连线框右半边依次连接【密码正确？】、【用户权限】。结果如图9所示。



图？

⑦ 编辑 VI 图标，改名为“核对密码”。



⑧ 保存 VI 至 Data.mdb 在同一级目录下，并改名为“核对密码”。

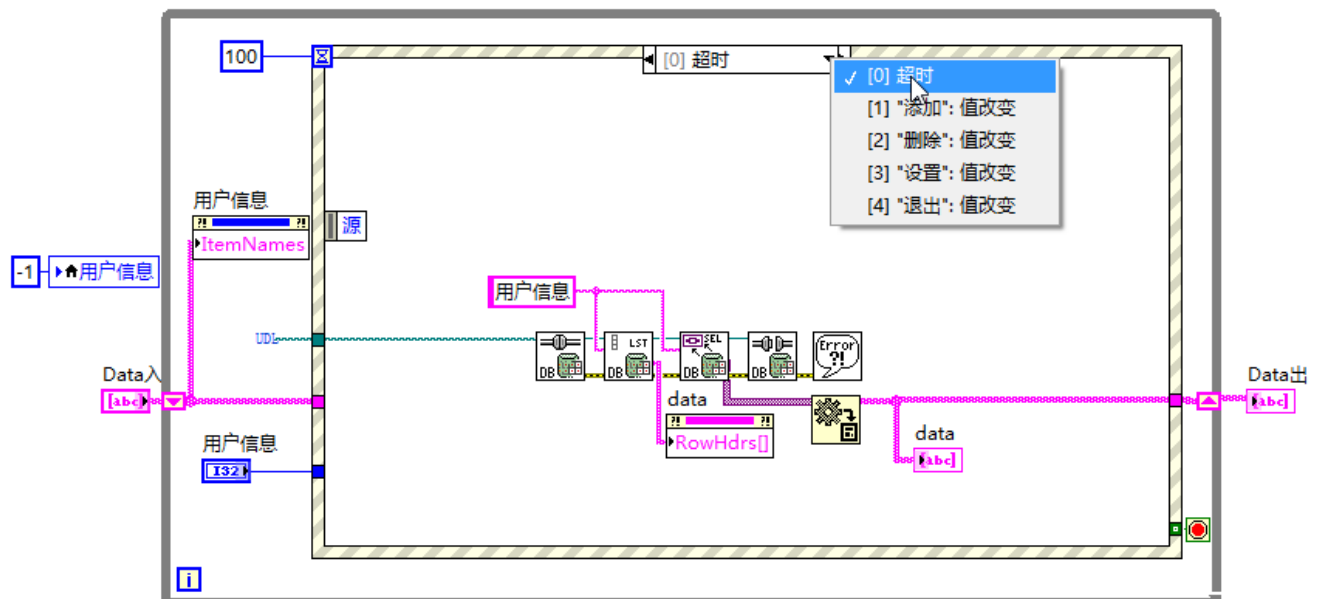
### 3.3.4 用户管理

前面板如图？所示。



图？

程序框图如图？所示。



图？

本 VI 有四个布尔控件对应不同的动作，因此也许要创建子 VI。

“用户管理”子 VI 创建过程：

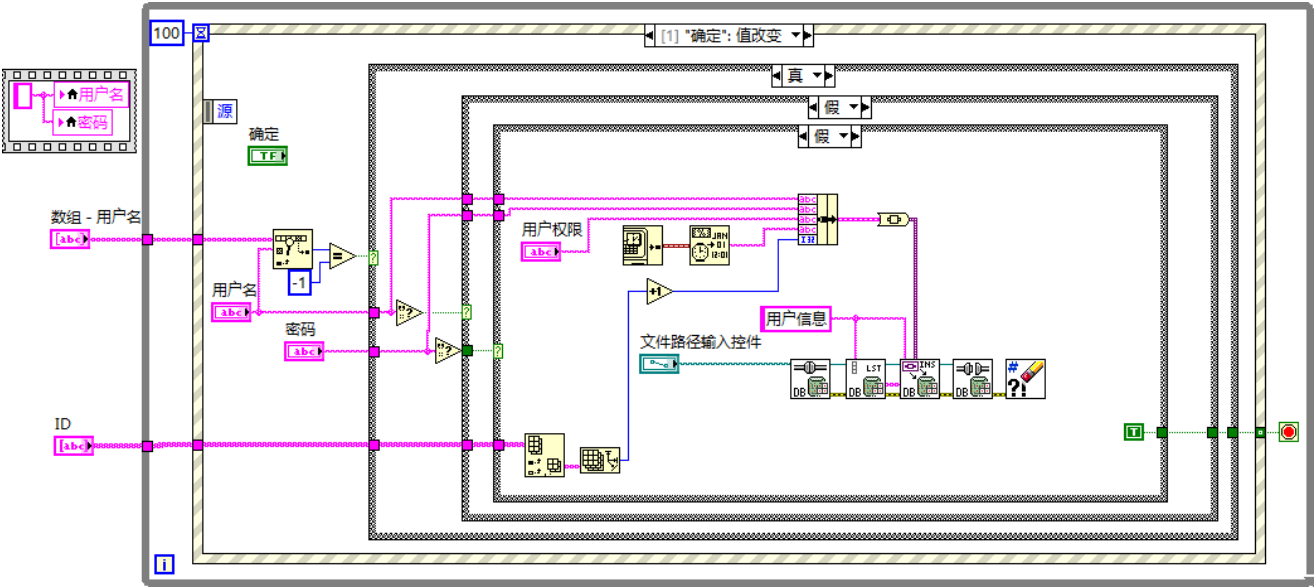
① 添加用户

前面板如图? 所示。



图?

程序框图如图? 所示。



思路：需要进行三次判断。

第一次判断:判断所创建的用户名是否存在，不存在则进行下一判断。

第二次判断:判断所创建的用户名是否为空，不为空则进行下一判断。

第三次判断:判断所创建的密码是否为空，不为空则写入数据库。

创建过程:

a 在前面板创建【组合框(银色)】(空白处右击选择【银色】/【字符串与路径】/【组合框(银色)】)(将标签改名为“用户权限”)和两个【字符串输入控件(银色)】(分别将标签改为“用户名”和“密码”)。再创建【确定按钮】和【取消按钮】(分别将标签改为“确定”和“取消”)。结果如图? 所示。在【用户权限】上右击选择<编辑项>, 插入如图? 所示内容。

用户权限

用户名

密码

确定 取消

图?

组合框属性: 用户权限

外观 编辑项 说明信息 数据绑定 快捷键 安全

☒ 值与项值匹配

项	值
普通用户	普通用户
管理员	管理员

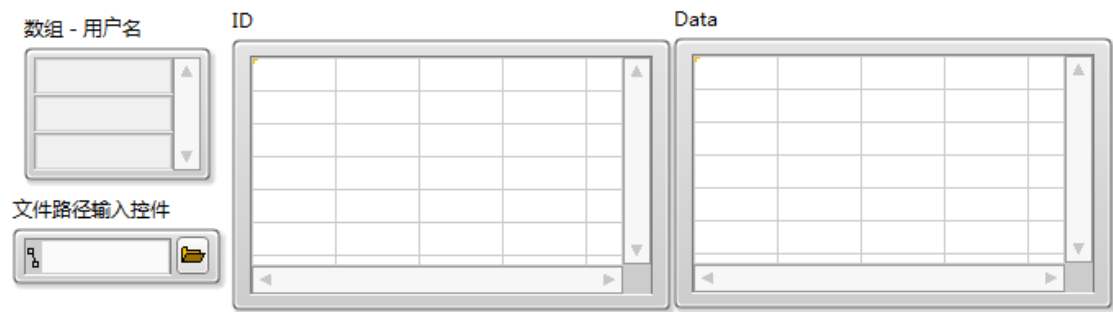
插入 删除 上移 下移 禁用项

☒ 允许在运行时有未定义值

确定 取消 帮助

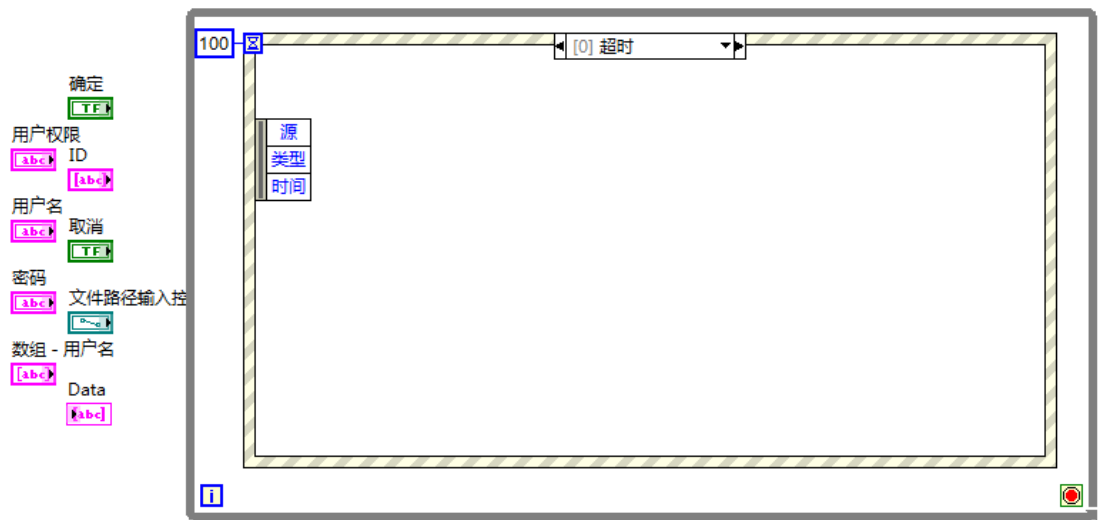
图?

b 在前面板创建【数组-字符串】(空白处右击选择【银色】/【数组、矩阵和簇】)(将标签改名为“数组-用户名”)、两个【表格(银色)】(依次将标签改名为“ID”和“Data”并将【Data】转化为显示控件)、【文件路径输入控件】。结果如图? 所示。



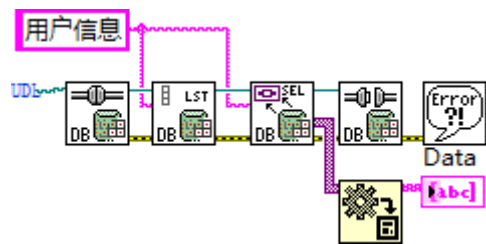
图?

c 在程序框图中创建一个【While 循环】、一个【事件结构】编辑【事件结构】的分支，共有三个分支，“0”分支为超时、“1”分支的“事件源”为【确定】，“事件”为“值改变”、“2”分支的“事件源”为【取消】，“事件”为“值改变”。为【事件结构】加上超时时间100。结果如图? 所示。




图?

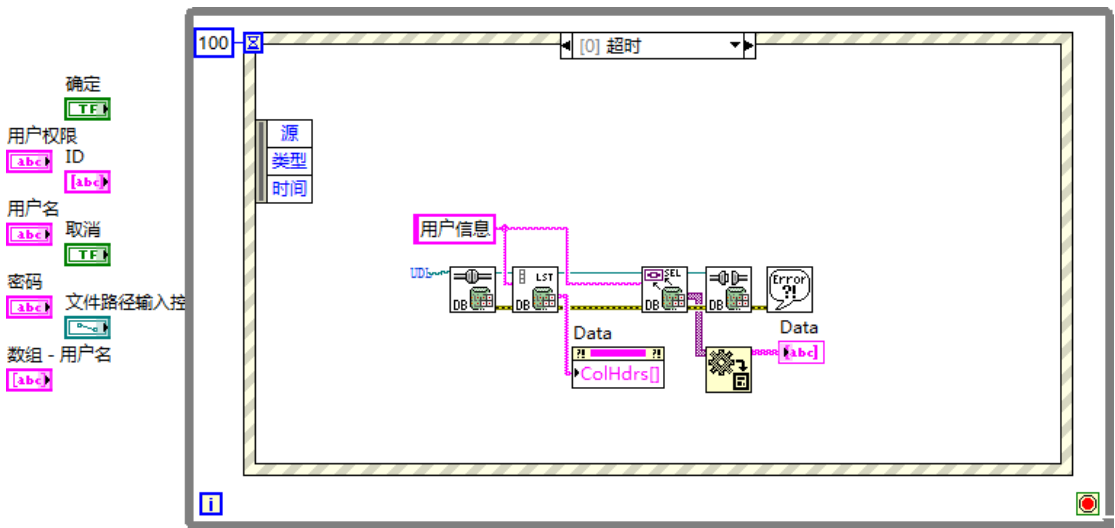
d 在【事件结构】的“0”分支内创建如图? 所示结构。



图?

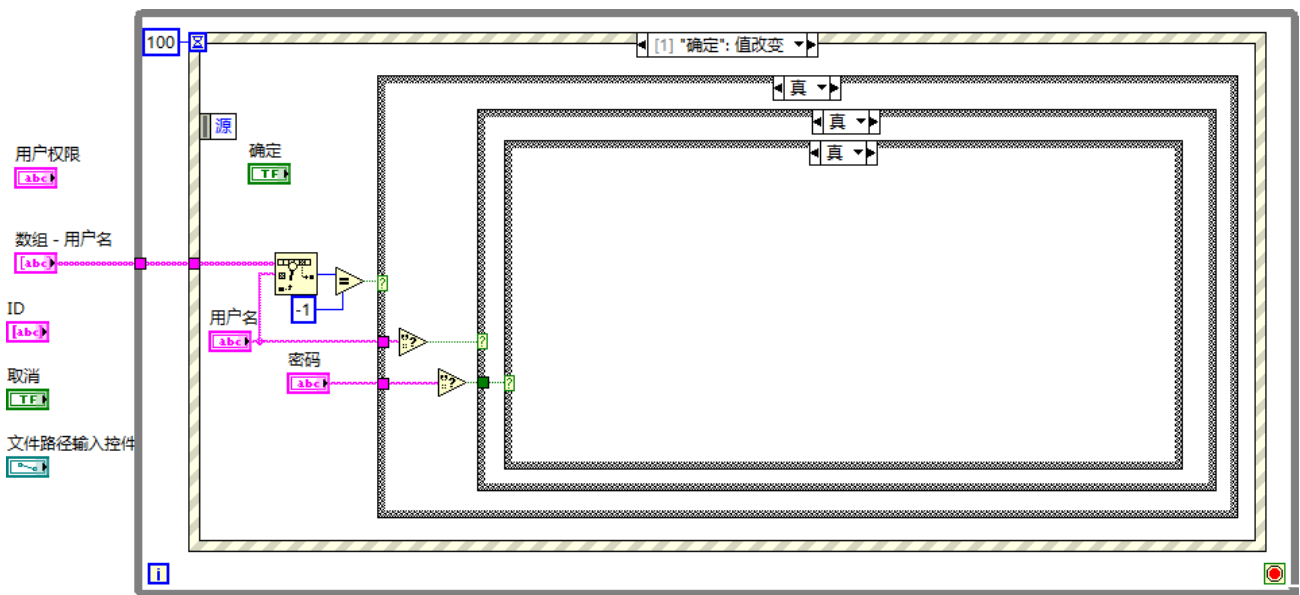
在【Data】上右击选择<创建>/<属性节点>/<列首字符串[]>转化为输入控件

。将【DB Tools List Columns】的“columns”处与【Data 列首字符串[]】相连。结果如图? 所示。



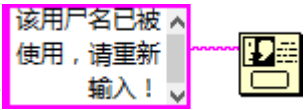
图?

e 在【事件结构】的“1”分支内创建三个【条件结构】、【搜索一维数组】和两个【空字符串/路径?】，建立如图? 所示连接图。



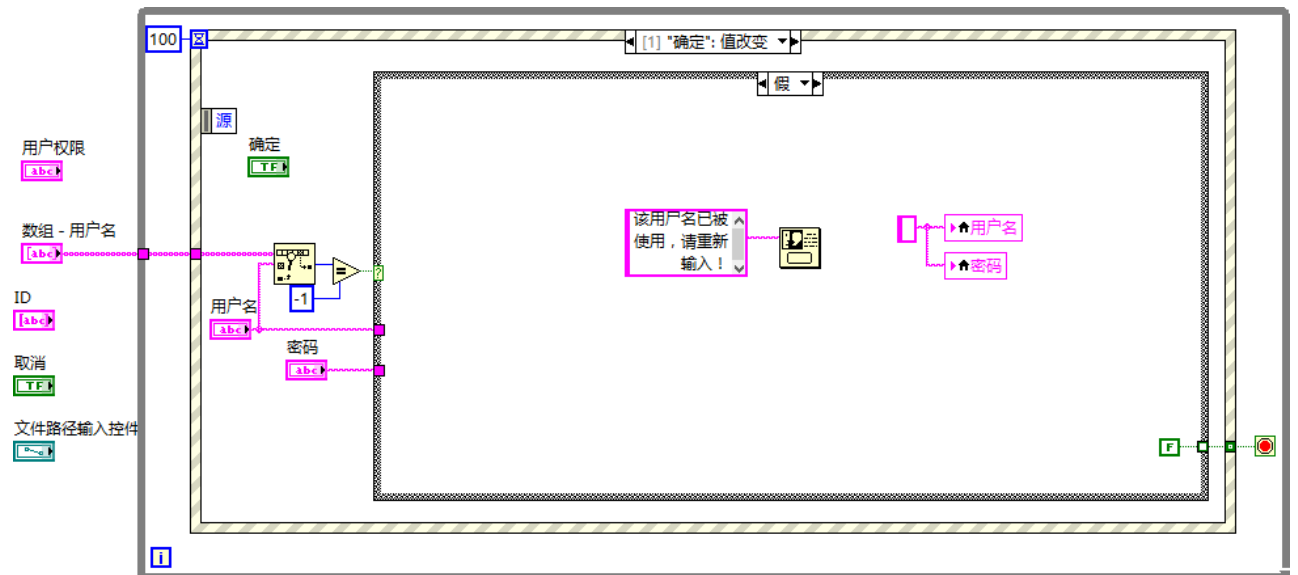
f 在第一个【条件结构】的“假”分支内创建【单按钮对话框】并用【字符串常量】

赋值，内容为“该用户名已被使用，请重新输入!”如图? 所示。并为【用户名】和【密码】创建【局部变量】，用【字符串常量】为它们赋值。

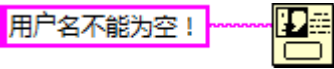


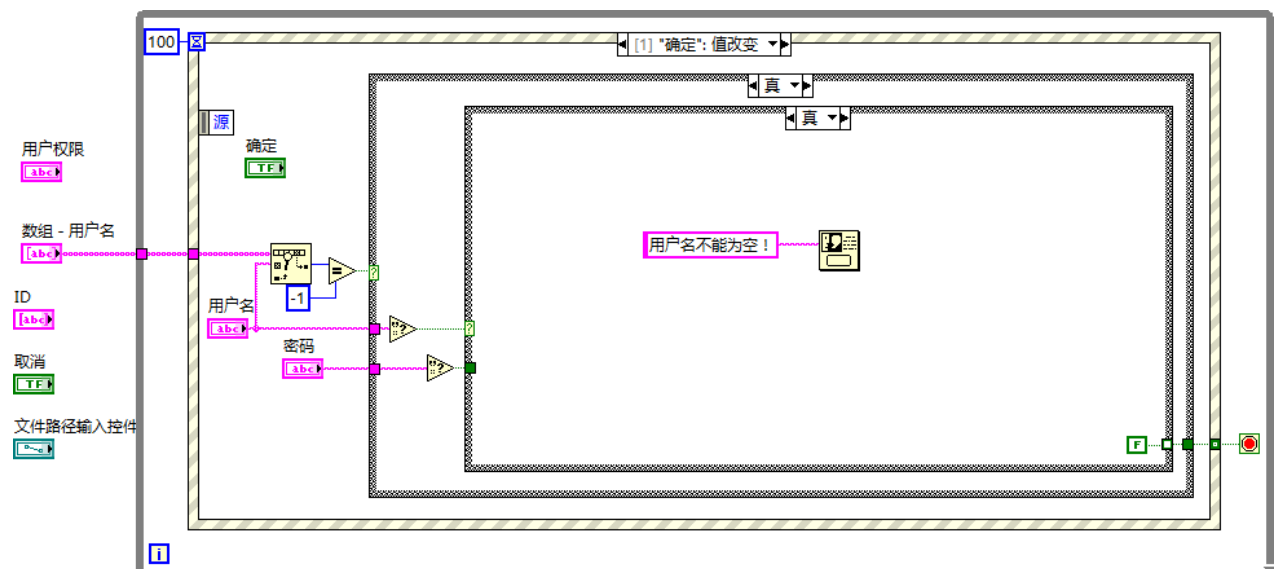


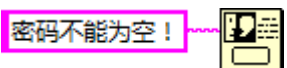
如  所示。创建【假常量】与【While 循环】的“循环条件”相连。结果如图? 所示。

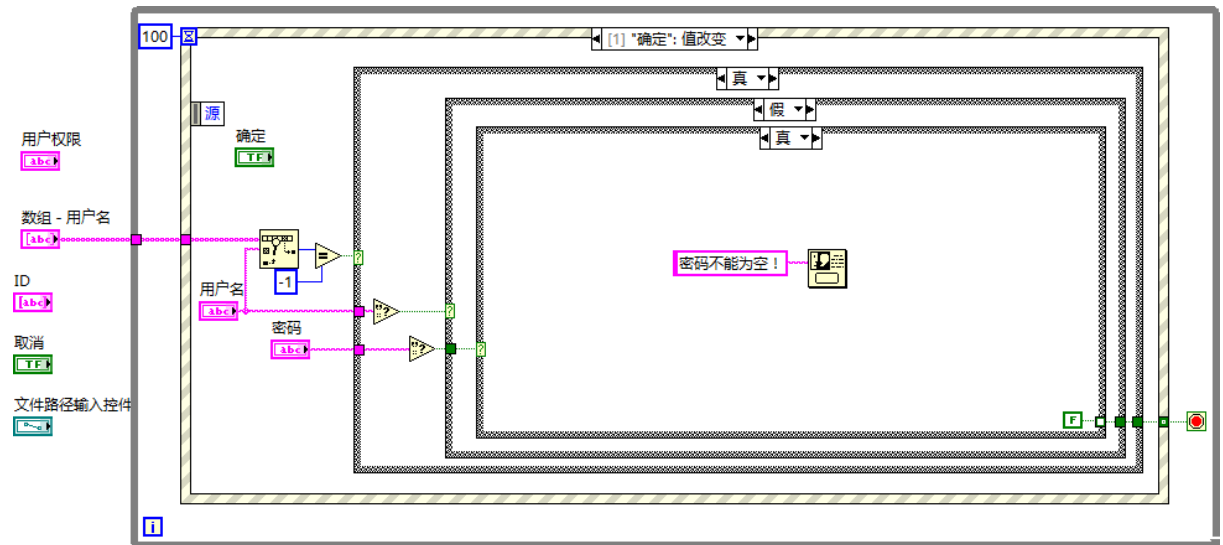


图?

g 在第二个【条件结构】的“真”分支内创建【单按钮对话框】并用【字符串常量】赋值，内容为“用户名不能为空！”。如  所示。创建【假常量】与【While 循环】的“循环条件”相连。结果如图? 所示。

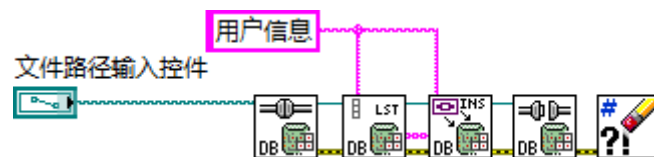


h 在第三个【条件结构】的“真”分支内创建【单按钮对话框】并用【字符串常量】赋值，内容为“密码不能为空！”。如  所示。创建【假常量】与【While 循环】的“循环条件”相连。结果如图? 所示。






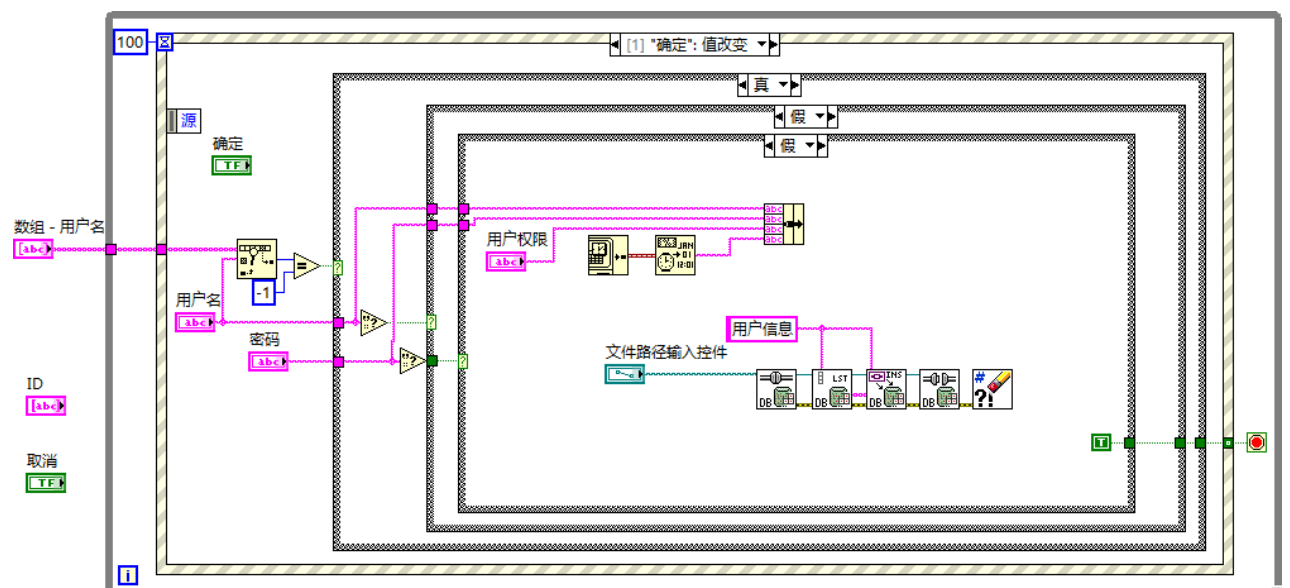
图?

j 在第三个【条件结构】的“假”分支内创建如图? 所示结构。



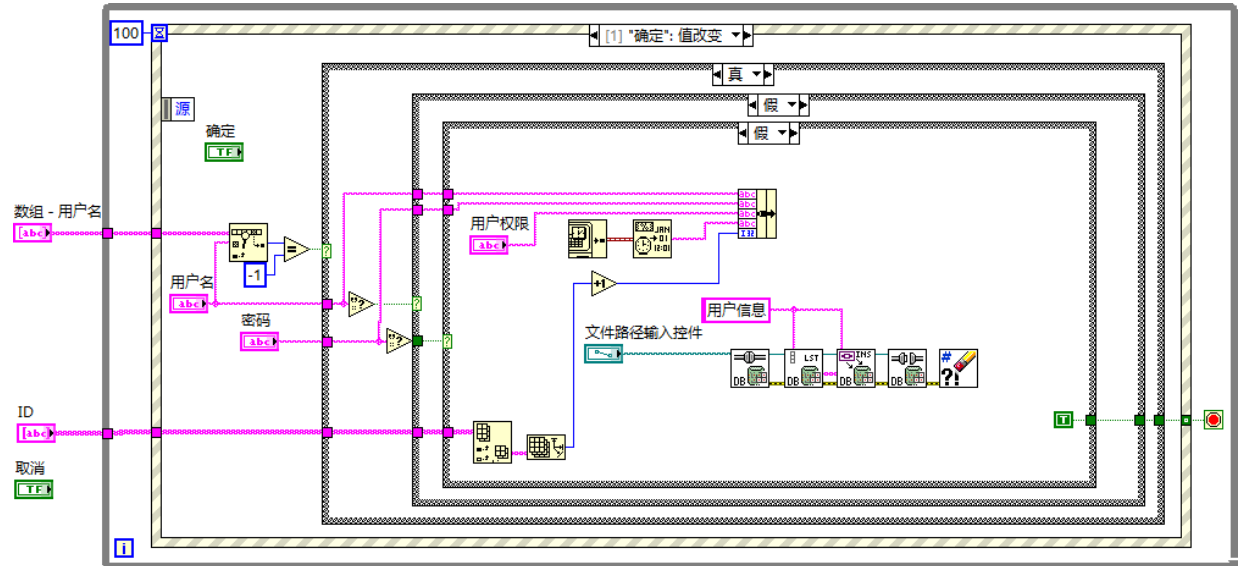
图?

k 在第三个【条件结构】的“假”分支内创建【捆绑】(【编程】/【簇、类与变体】/【捆绑】) , 创建【获取日期/时间(秒)】(【编程】/【定时】/【获取日期/时间(秒)】)  和【格式化日期/时间字符串】。再创建【假常量】与【While 循环】的“循环条件”相连，连接如图? 所示。




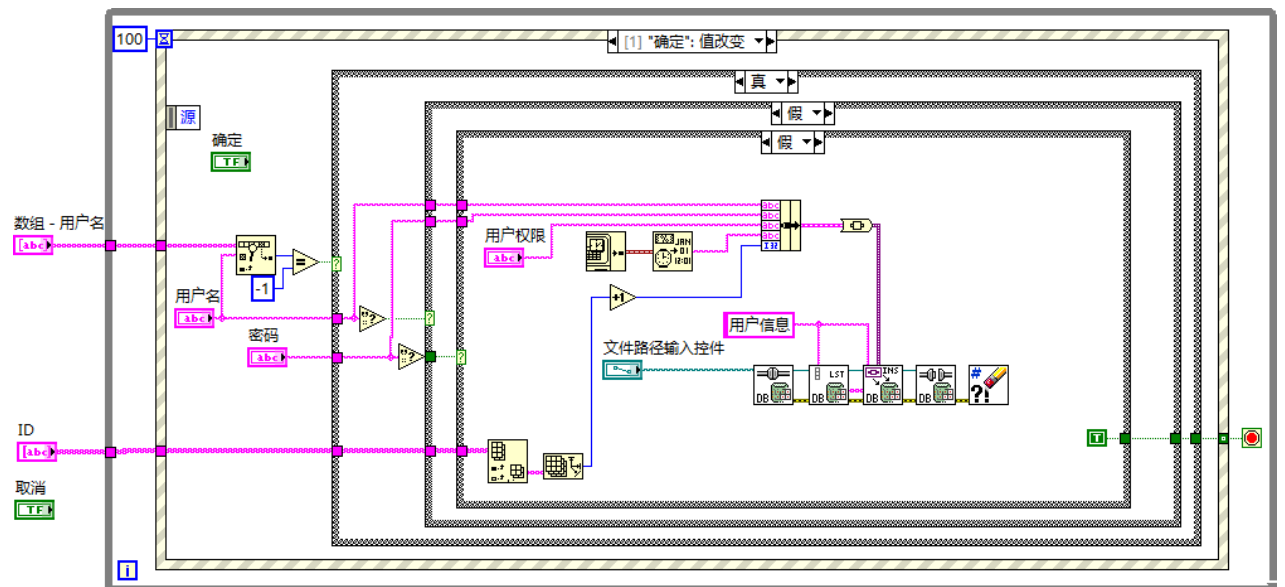
图？

l 在第三个【条件结构】的“假”分支内创建【引索数组】，在“引索列”处创建【数值常量 4】，创建【数组大小】和【加 1】连接如图？所示。



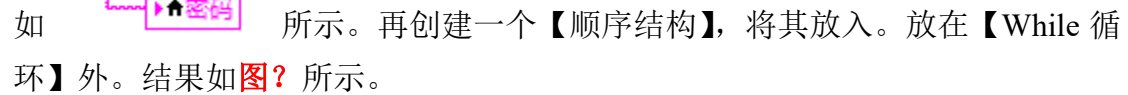
图？

m 在第三个【条件结构】的“假”分支内创建【转换为变体】(【编程】/【簇、类与变体】/【变体】/【转换为变体】) 。将其输入端与【捆绑】的“输出簇”相连，输出端与【DB Tools Insert Data】的“data”端相连。结果如图？所示。



图？

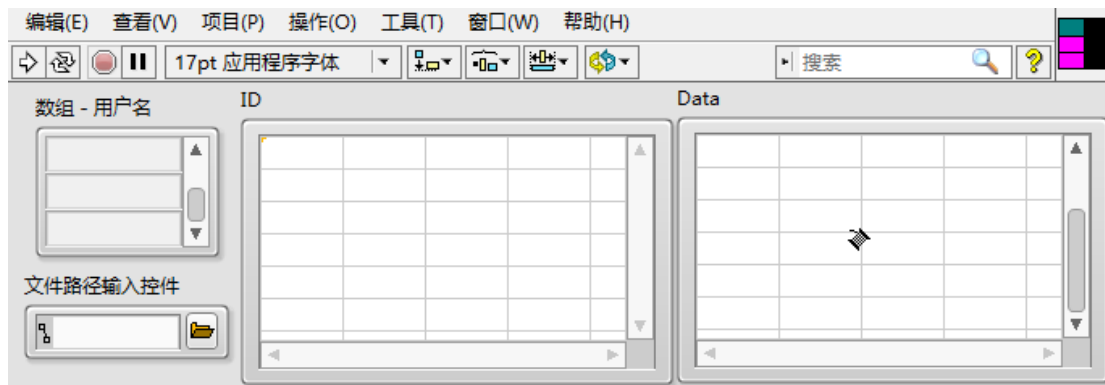
n 为【用户名】和【密码】创建【局部变量】，用【字符串常量】为它们赋值。



- 

图？

- p 在前面板编辑连线模式，选择连线模式。连线框的左半边依次连接【文件路径输入控件】、【数组-用户名】和【ID】，连线框的右半边连接【Data】。如图?所示。连接完成后，隐藏控件。



q 编辑 VI 图标，改名为“添加用户”。结果如图? 所示。



图?

r 编辑好前面板后，设置 VI 属性。设置方式与“修改密码.vi”的方式一致。

s 保存 VI 至 Data.mdb 在同一级目录下，并改名为“添加用户”。

## ② 删除用户

a 在前面板创建【文件路径输入控件(银色)】、【字符串输入控件(银色)】(将标签改名为“用户名”)、【表格(银色)】(将标签改名为“Data”，并转换为显示控件)。结果如图? 所示。

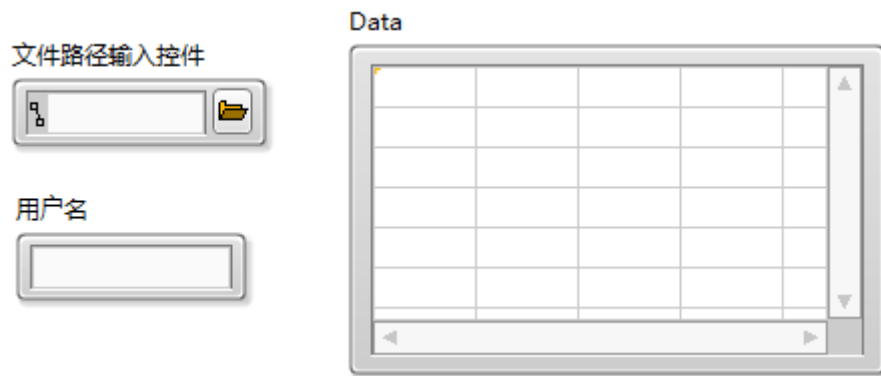




图1

b 在程序框图创建与先前一致的框架。唯一不同的是将【简单错误处理器】改成【清除错误】。结果如图2所示。

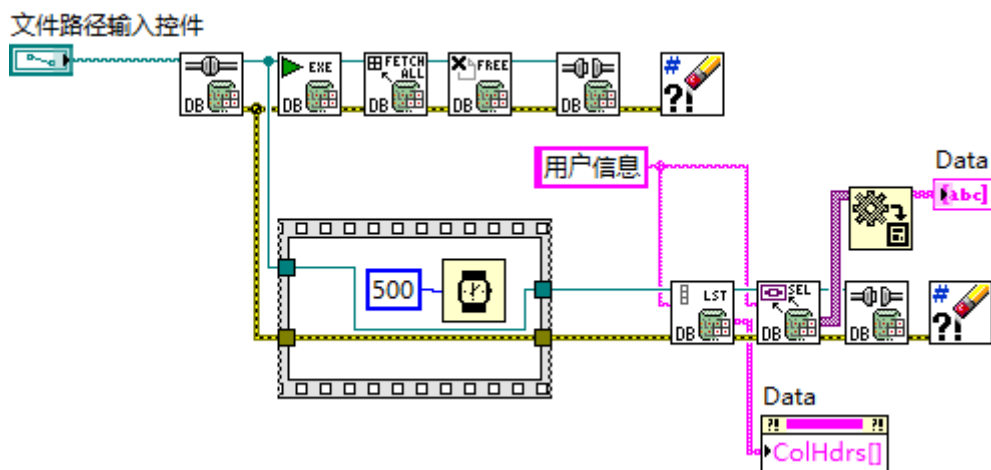


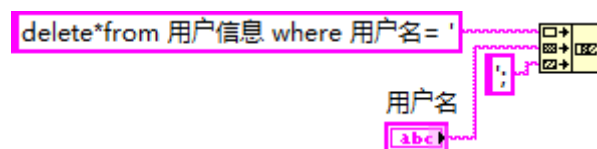
图2

c SQL 语句中的删除语句。

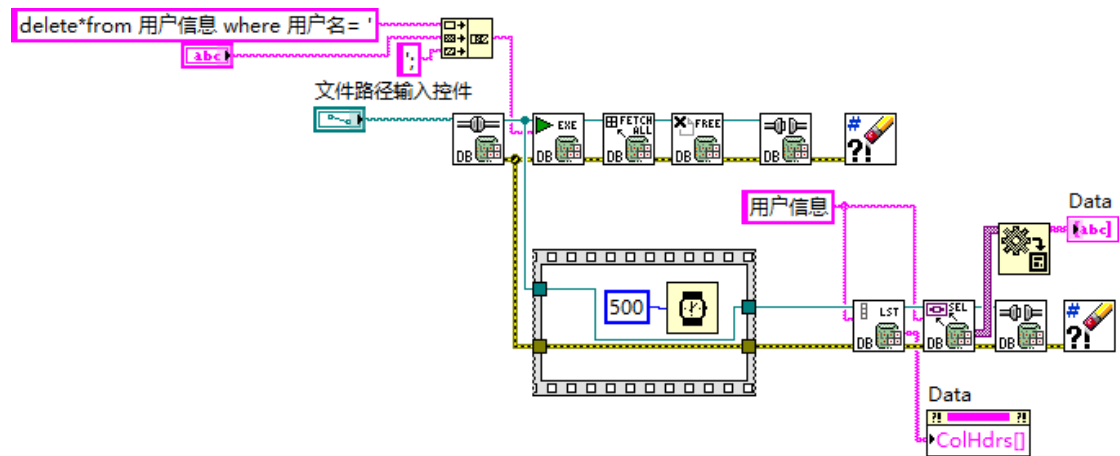
如: `delete*from 用户信息 where 用户名= '*';`

表示删除表“用户信息”中“用户名”列中为“\*”的整行信息。


因此创建【连接字符串】以及【字符串常量】, 建立如图3所示连接。

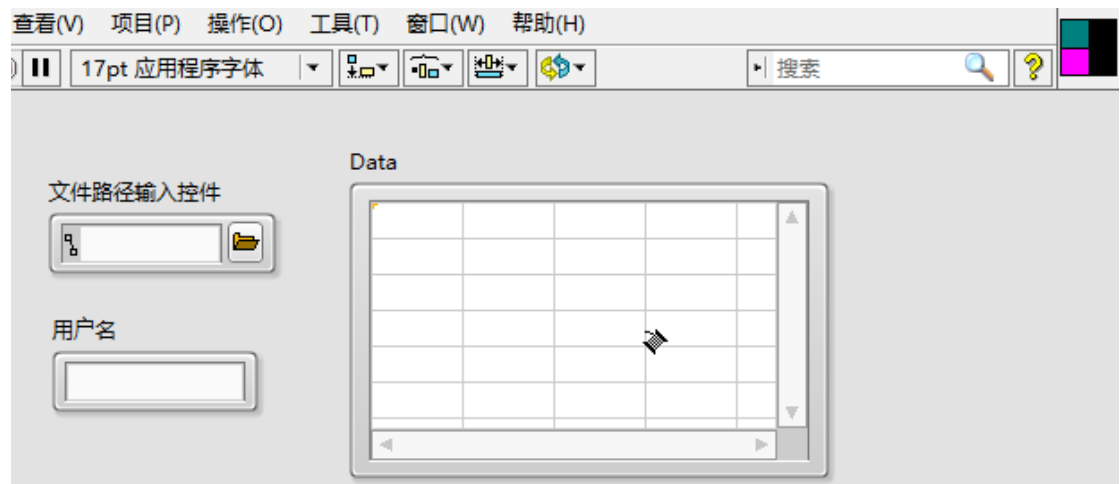


d 将【连接字符串】的输出端与【DB Tools Execute Query】的“SQL query”端相连。结果如图4所示。




图？

e 编辑连线模式，选择  连线模式。连线框的左半边依次连接【文件路径输入控件】和【用户名】，连线框的右半边连接【Data】。结果如图？所示。



图？

f 编辑 VI 图标，编辑完成后如  所示。

g 保存 VI 至 Data.mdb 在同一级目录下，并改名为“删除用户”。

### ③ 编辑用户

前面板如图？所示。

编辑用户

用户名

密码

用户权限

管理员

上次登录时间

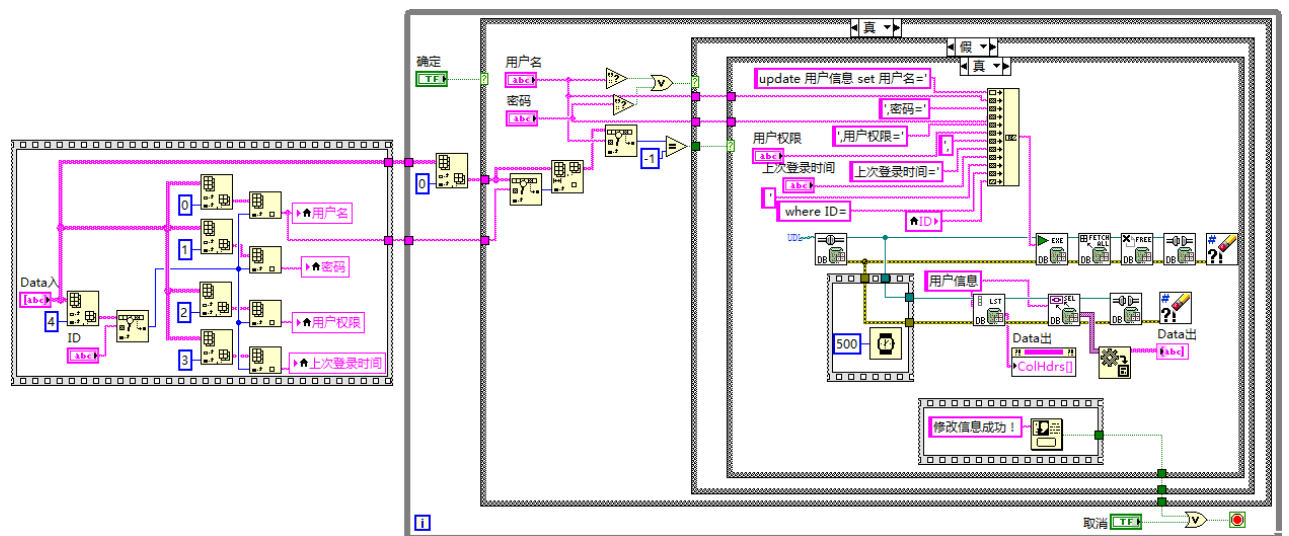
 确定



退出

图？

程序框图如图? 所示。



图?

思路：需要进行 3 次判断过程。

第一次判断：判断【确定】是否执行，执行则进行第二次判断。否则不进行。

第二次判断：判断【用户名】和【密码】是否不为零。不为零进行第三次判断，否则不进行。

第三次判断：判断【用户名】是否存在。不存在输入结果至数据库中，否则不进行。

操作过程:

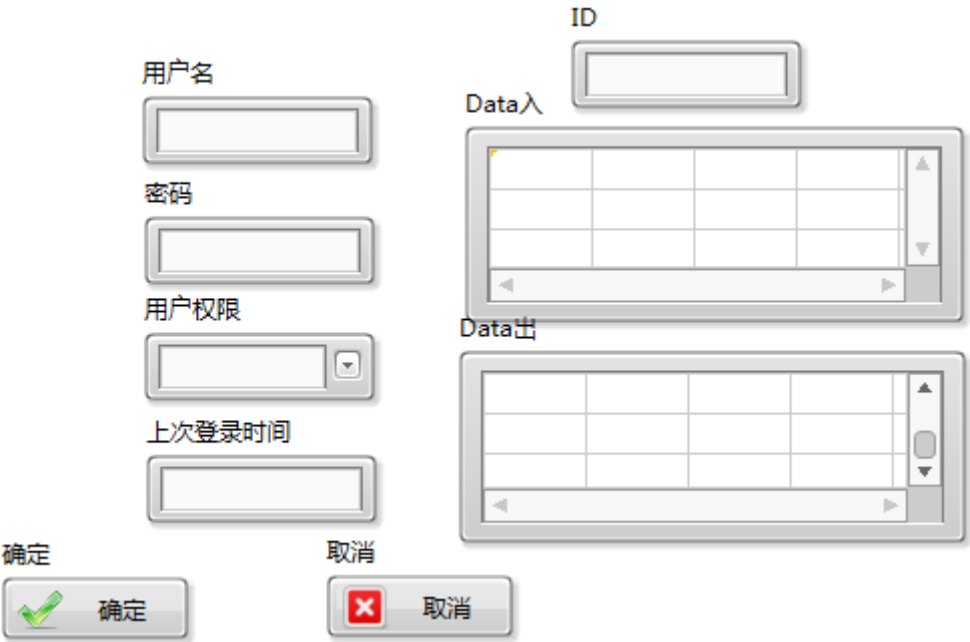


a 在前面板创建一个【组合框(银色)】(将标签改名为“用户权限”，<编辑项>将项设为“普通用户”和“管理员”)、三个【字符串输入控件(银色)】(依次将标签改名为“用户名”、“密码”和“上次登录时间”)以及【确定按钮】和【取消按钮】(将标签名改为“确定”和“取消”)。结果如图? 所示。



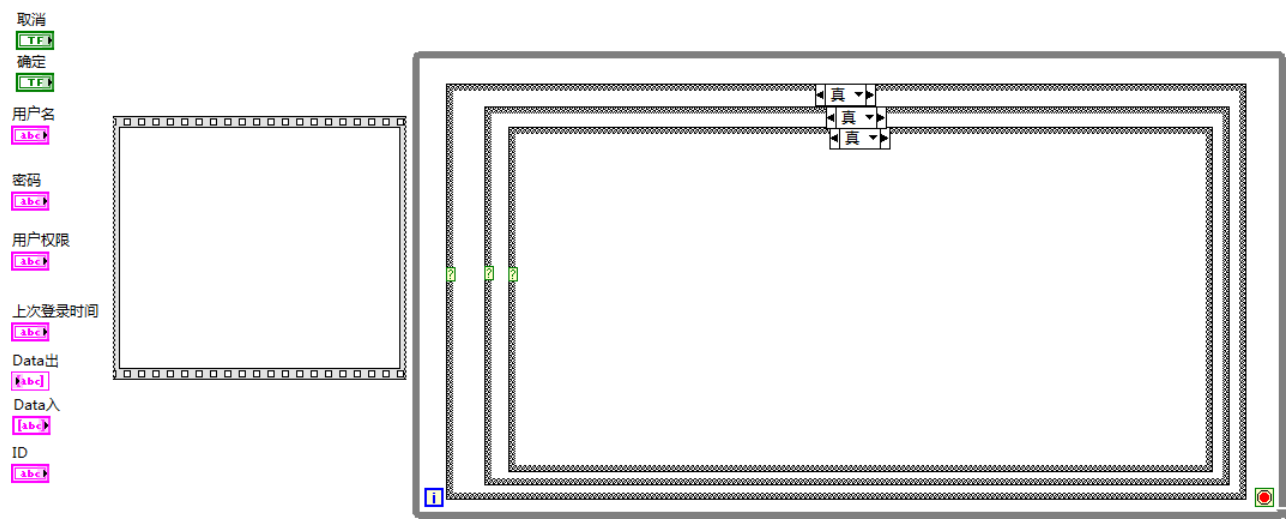
图?

b 在前面板创建【字符串输入控件(银色)】(将标签改名为“ID”)和两个【表格(银色)】(分别将标签改名为“Data 入”和“Data-出”并将【Data 出】转换为显示控件)。结果如图?所示。



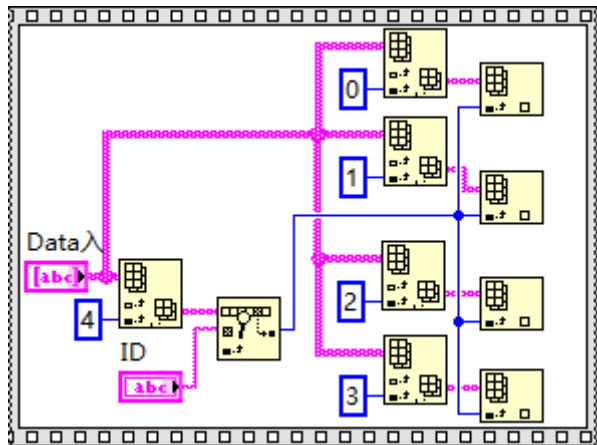
图?

c 在程序框图创建一个【While 循环】、一个【顺序结构】和三个【条件事件】。  
结果如图? 所示。



图?

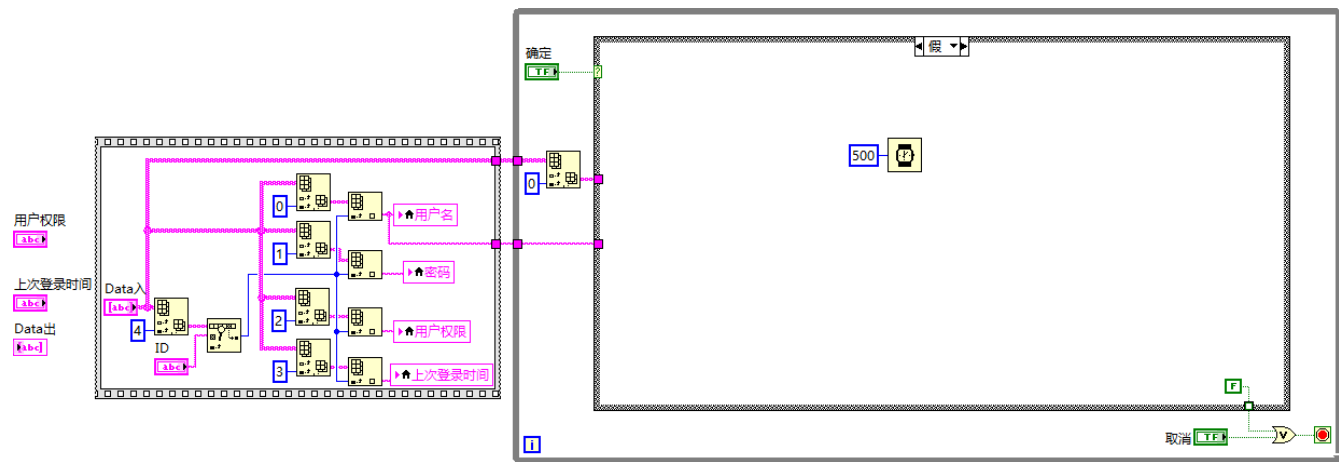
d 在【顺序结构】中创建一个【搜索一维数组】和多个【引索数组】，目的在数据库导出，选中信息的各项参数。连接方式如图? 所示。



图?


e 分别为【用户名】、【密码】、【用户权限】、【上次登录时间】创建【局部变量】依次和四个【引索数组】连接。结果如图? 所示。

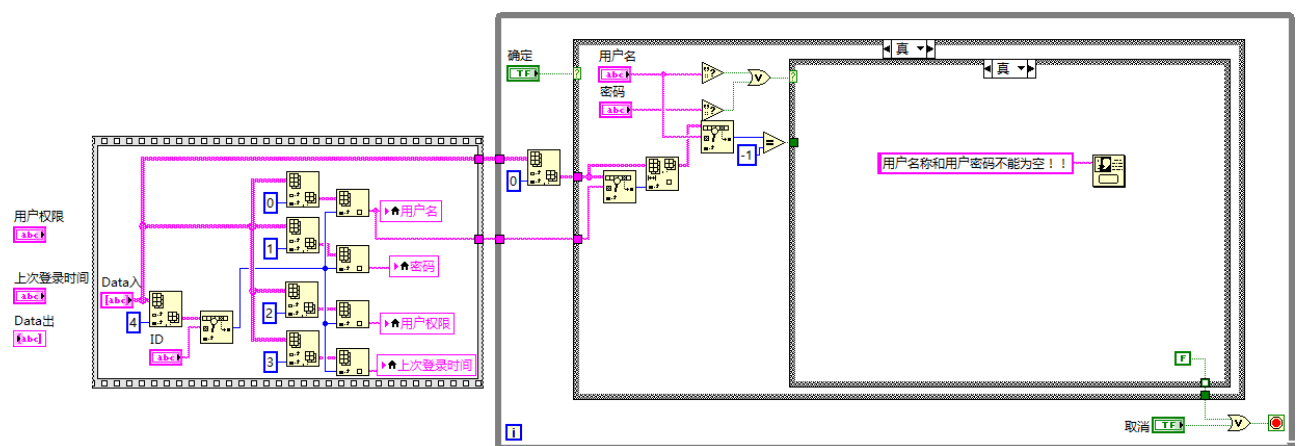




图?

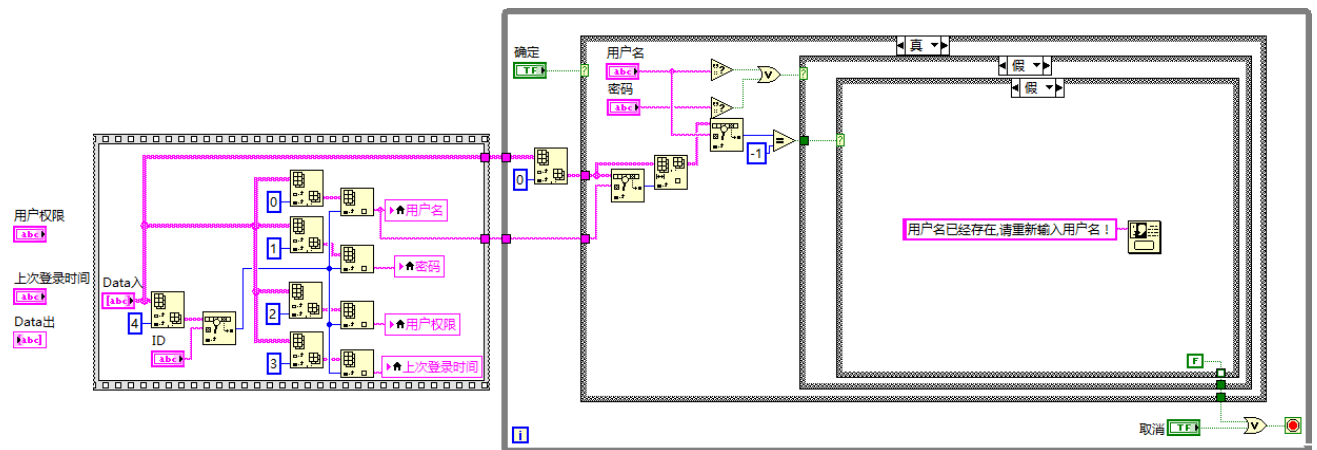
h 在第二个【条件结构】的“真”分支中创建【单按钮对话框】并用【字符串常量】为其赋值，内容为“用户名称和用户密码不能为空!! ”

如图  所示。创建【假常量】与【While 循环】的“循环条件”相连。如图? 所示。



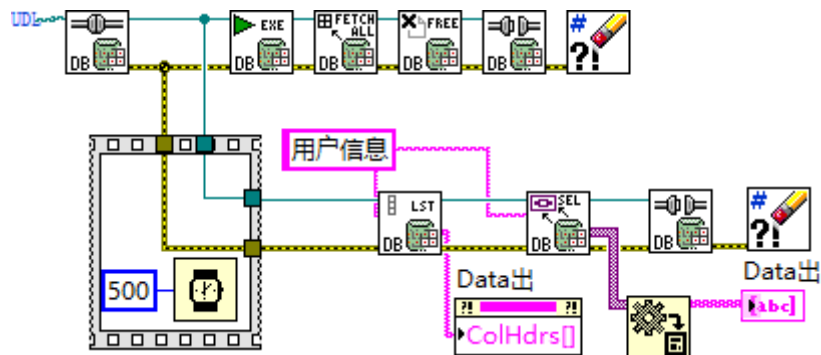
图?

i 在第三个【条件结构】的“假”分支内创建【单按钮对话框】并用【字符串常量】为其赋值，内容为“用户名已经存在,请重新输入用户名! ”。创建【假常量】与【While 循环】的“循环条件”相连。如图? 所示。



图?

j 在第三个【条件结构】的“真”分支内创建如图?所示结构。



图?

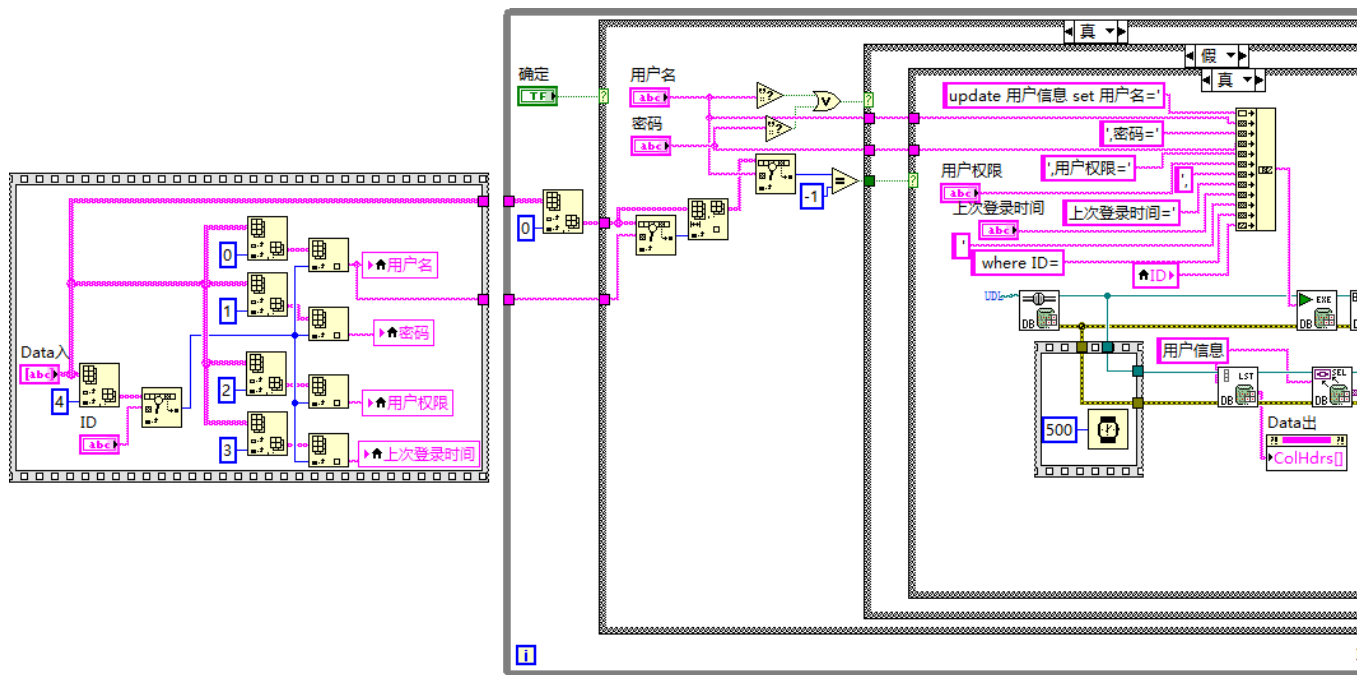
k SQL 语句中的更新语句。

如: updata 用户信息 set 用户名='\*' where 条件语句

表示当条件语句成立时, 更新“用户信息”表中的“用户名”列的信息为“\*”。

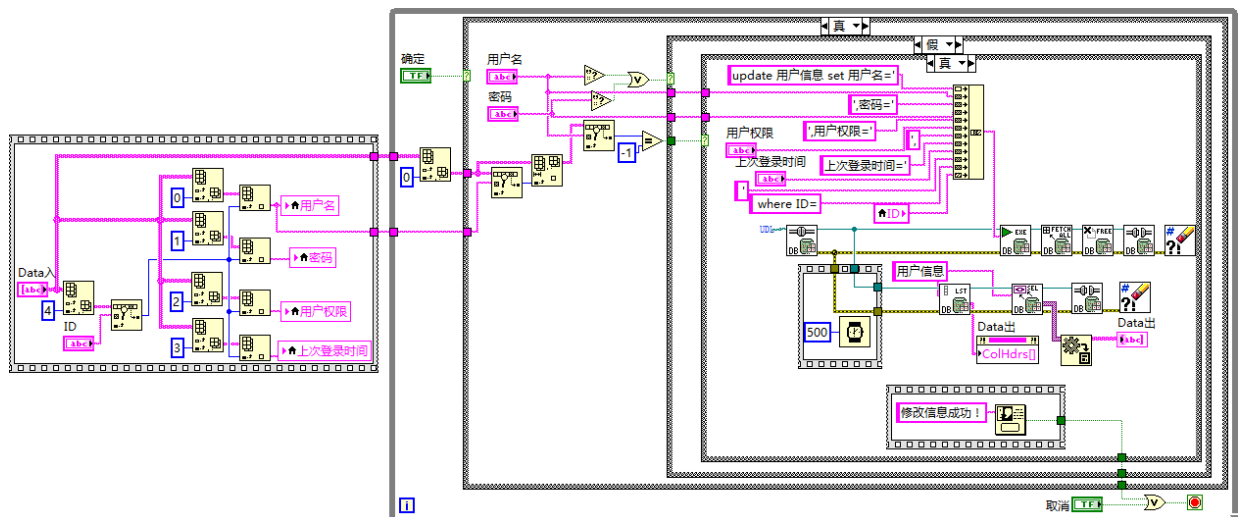
根据此语句创建【连接字符串】和【ID】的【局部变量】(转换为读取), 创建如

图? 所示的结构。




图?

1 在第三个【条件结构】的“真”分支内创建【单按钮对话框】并用【字符串常量】为其赋值，内容为“修改信息成功！”，外面放置【顺序结构】。将【单按钮对话框】的输出端与【While 循环】的“循环条件”相连。如图? 所示。



图?

m 编辑连线模式，选择  模式。连线框的左半边依次连接【Data 入】和【ID】，右半边连接，如图? 所示。设置好连线模式后，将【Data 入】、【ID】、【Data 出】隐藏。

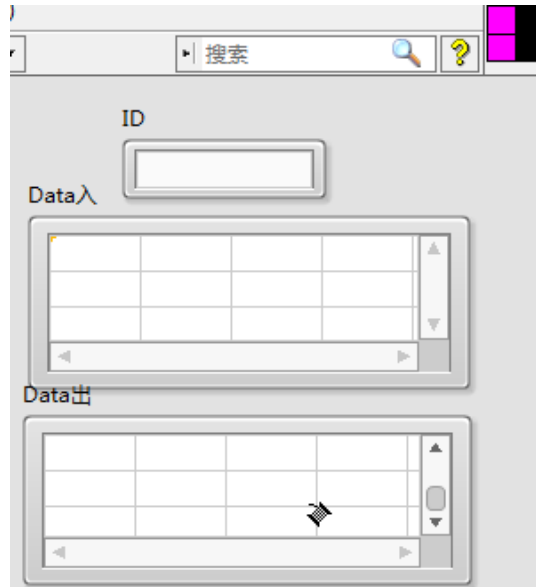


图7

- n 编辑 VI 图标，改名为“编辑用户”，如图8所示。
- o 编辑好前面板后，设置 VI 属性。设置方式与“修改密码.vi”的方式一致。
- p 保存 VI 至 Data.mdb 在同一级目录下，并改名为“编辑用户”。

#### ④ 用户管理创建过程

a 在前面板创建【多列列表框(银色)】(将标签名“用户信息”并在表上右击选择<选择模式>/<高亮显示整行>同时选择<属性>/<外观>/<显示列首>和<显示水平滚动条>如图7所示)和两个【表格(银色)】(分别将标签改名为“Data 入”和“Data-出”并将【Data 出】转换为显示控件)。再创建【添加按钮】、【删除按钮】、【设置按钮】、【取消按钮】(分别将标签名改为“添加”、“删除”、“设置”、“退出”并隐藏“标签”)。结果如图7所示。

外观

说明信息

数据绑定

快捷键

安全

标签

☒ 可见

用户信息

启用状态

☒ 启用

☐ 禁用

☐ 禁用并变灰

大小

高度

159

宽度

476

5

行

4

列

☒ 显示垂直滚动条

☐ 显示水平滚动条

☒ 显示列首

☐ 显示行首

☒ 显示垂直线

☒ 显示水平线

☐ 显示符号

☐ 显示索引框

确定

取消

帮助

图？

用户信息


+

添加

-

删除

⚙

设置

✖

取消

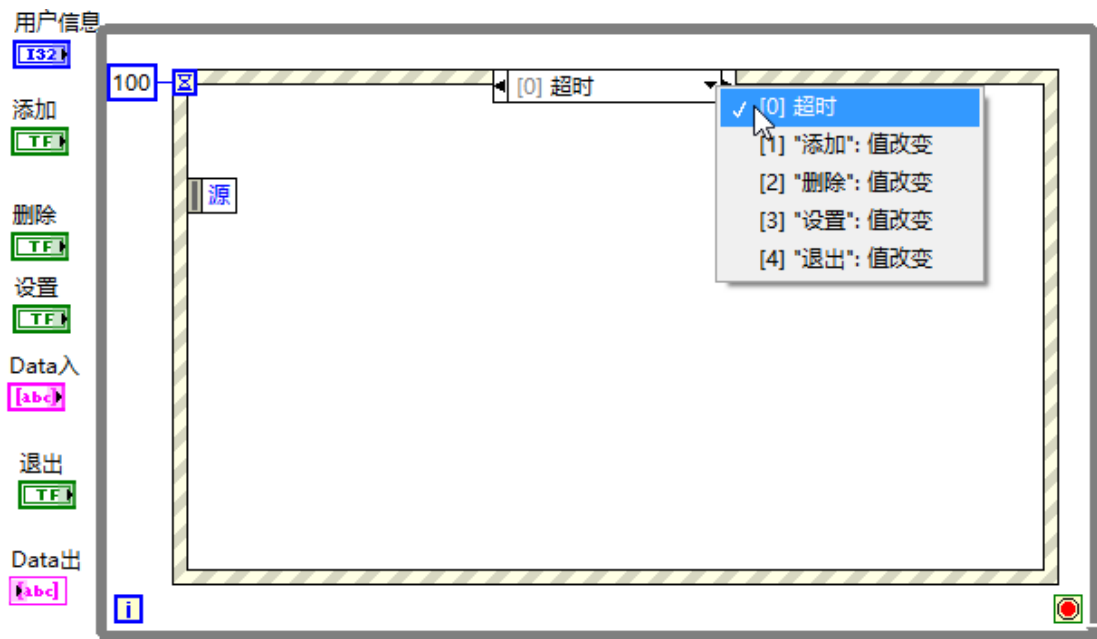
Data入


Data出


图？

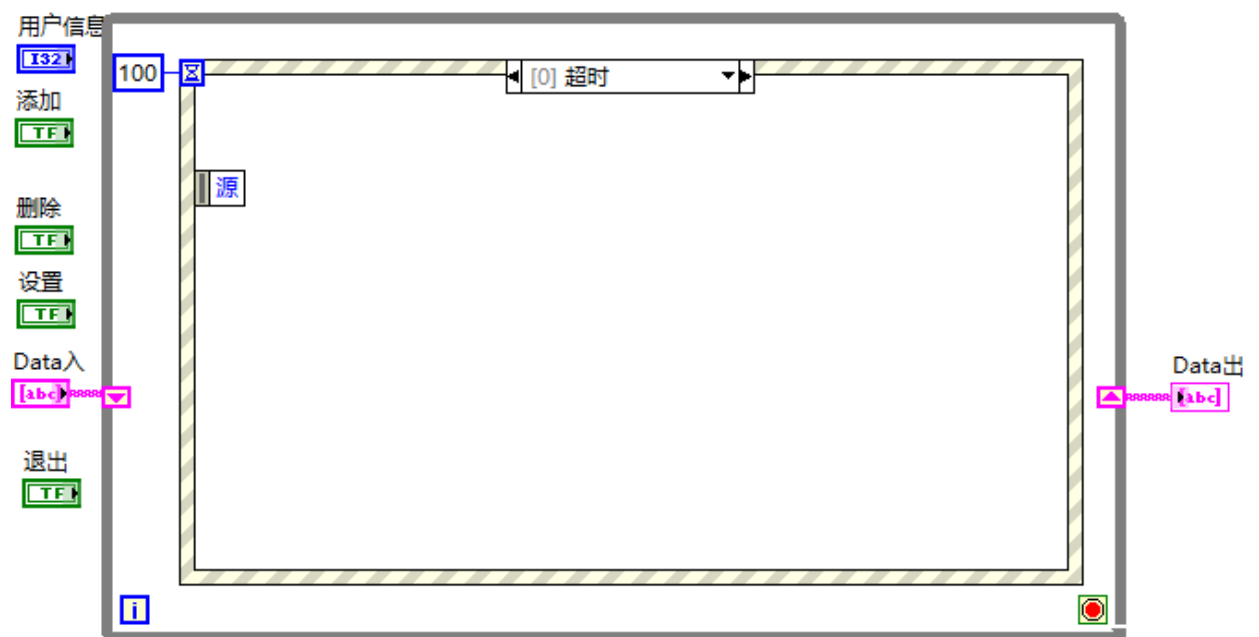
b 在程序框图中创建【While 循环】和【事件结构】，编辑和添加【事件结构】的分支并创建延时时间。结果如图?所示。





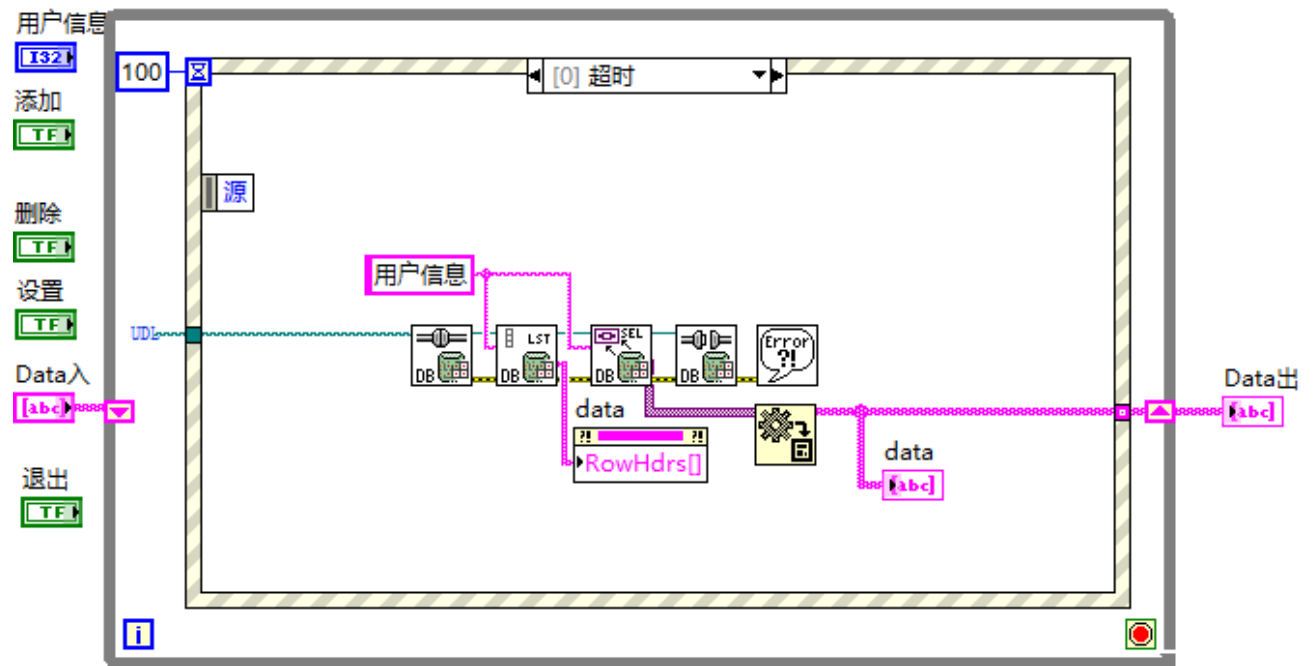
图?

c 在【While 循环】上创建“移位寄存器”将输入端与【Data 入】相连，将输出端【Data 出】相连。如图?所示。



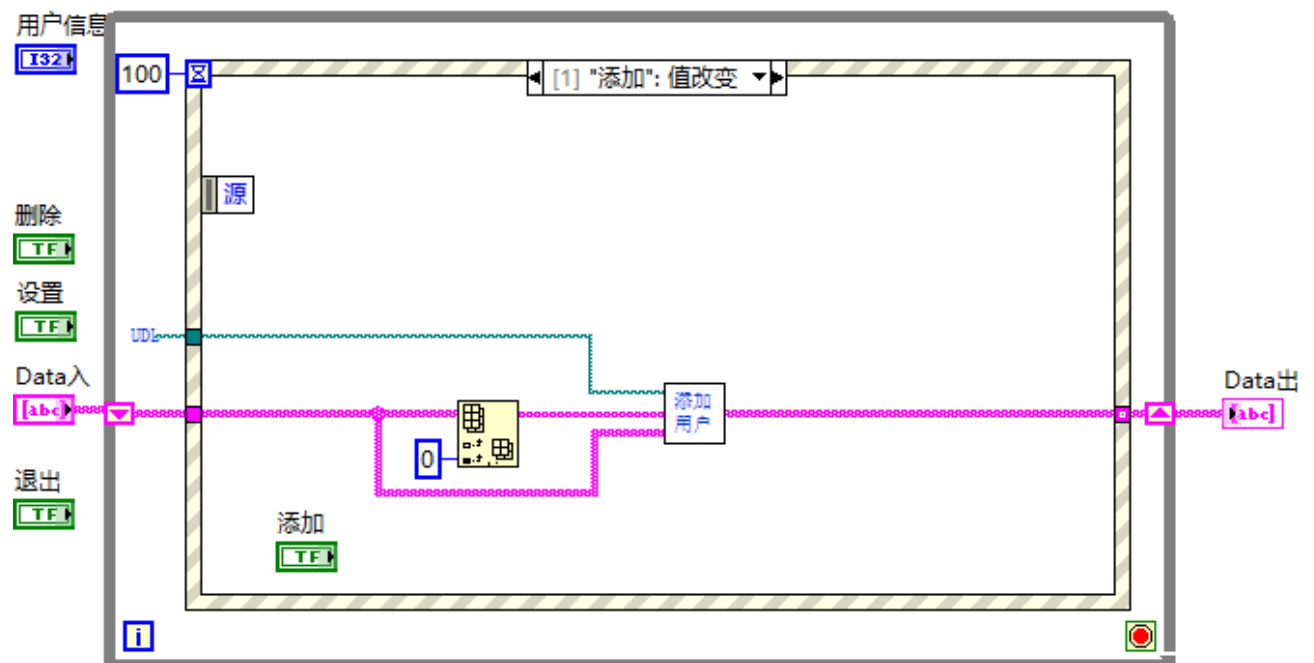
图?

d 在前面板创建【data】转换为显示控件，创建后将其隐藏。在【事件结构】的“0”分支中创建如图?所示结构。



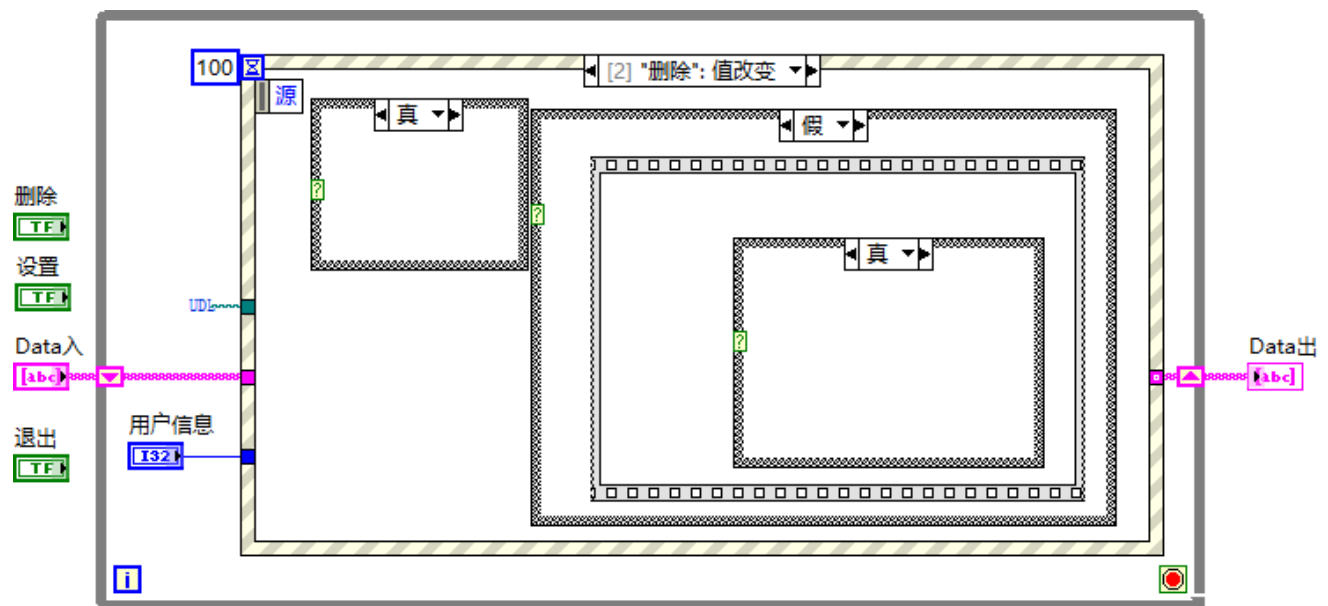
图?

e 在【事件结构】的“1”分支中添加【引索数组】，“引索列”为 0，添加【添加用户.vi】(空白处右击选择<选择 VI...>/<添加用户.vi>)创建如图?所示结构。



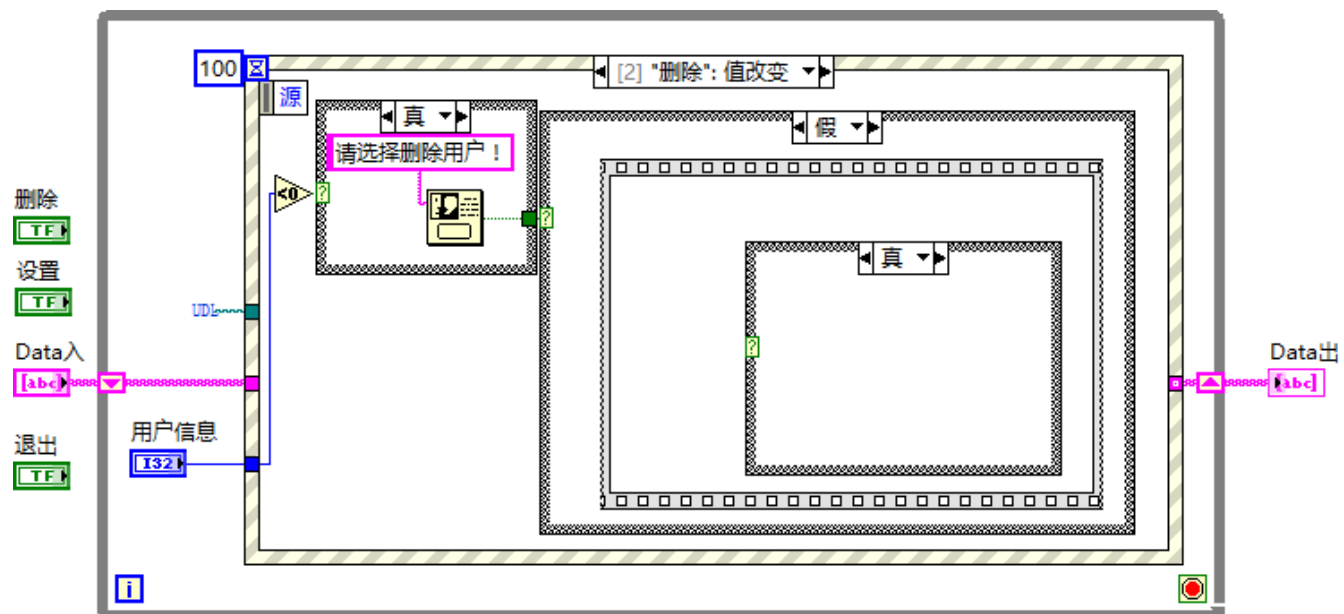
图?

f 在【事件结构】的“2”分支中创建三个【条件结构】和一个【顺序结构】，嵌套关系如图?所示。



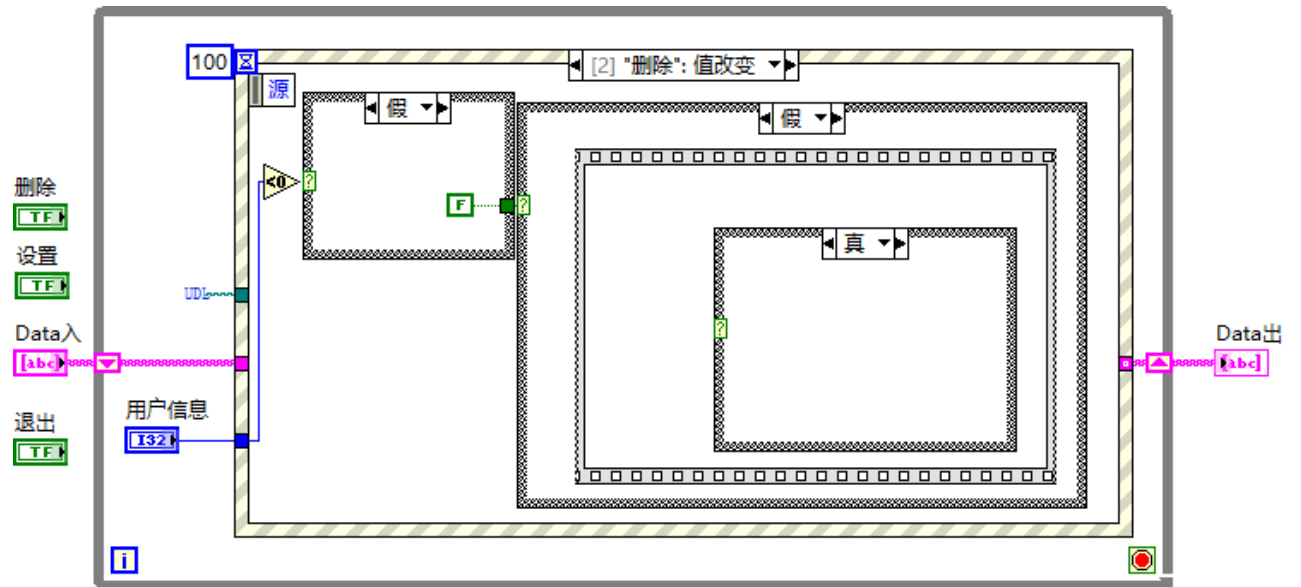
图?

创建【小于 0?】<0>，在第一个【条件结构】的“真”分支中创建【单按钮对话框】并用【字符串常量】为其赋值，内容为“请选择删除用户！”。如  
 图?所示。并将【单按钮对话框】的输出端与第二个【条件结构】的“分支选择器”相连。结果如图?所示。



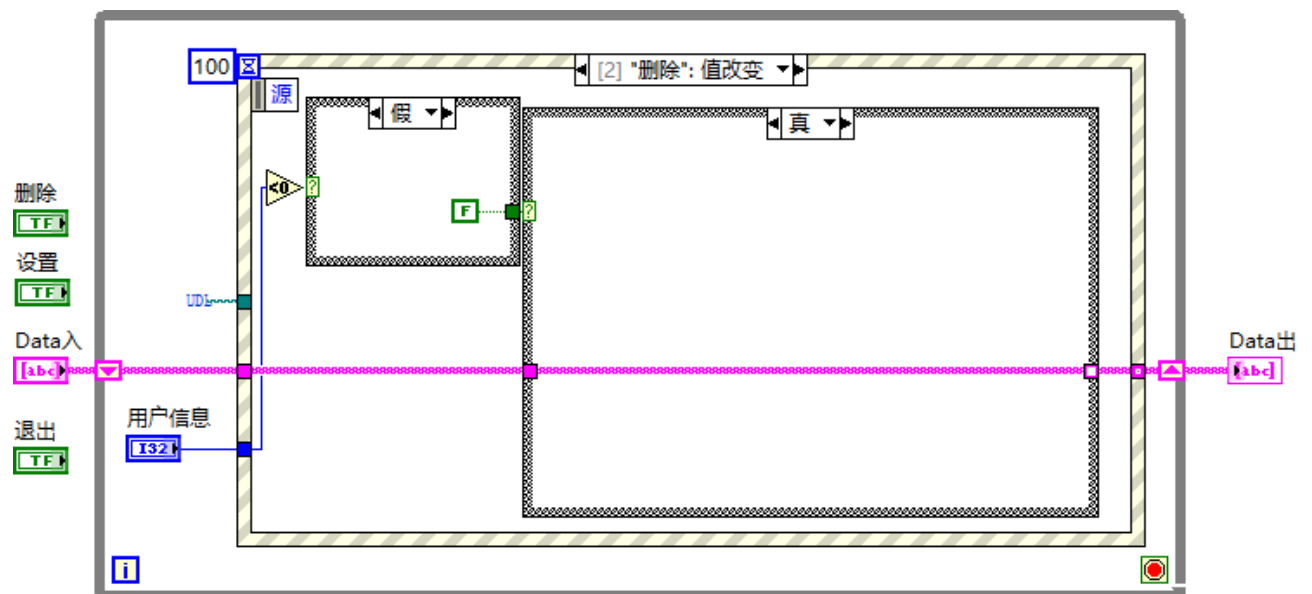
图?

在“假”分支中创建【假常量】，并第二个【条件结构】的“分支选择器”相连。结果如图?所示。




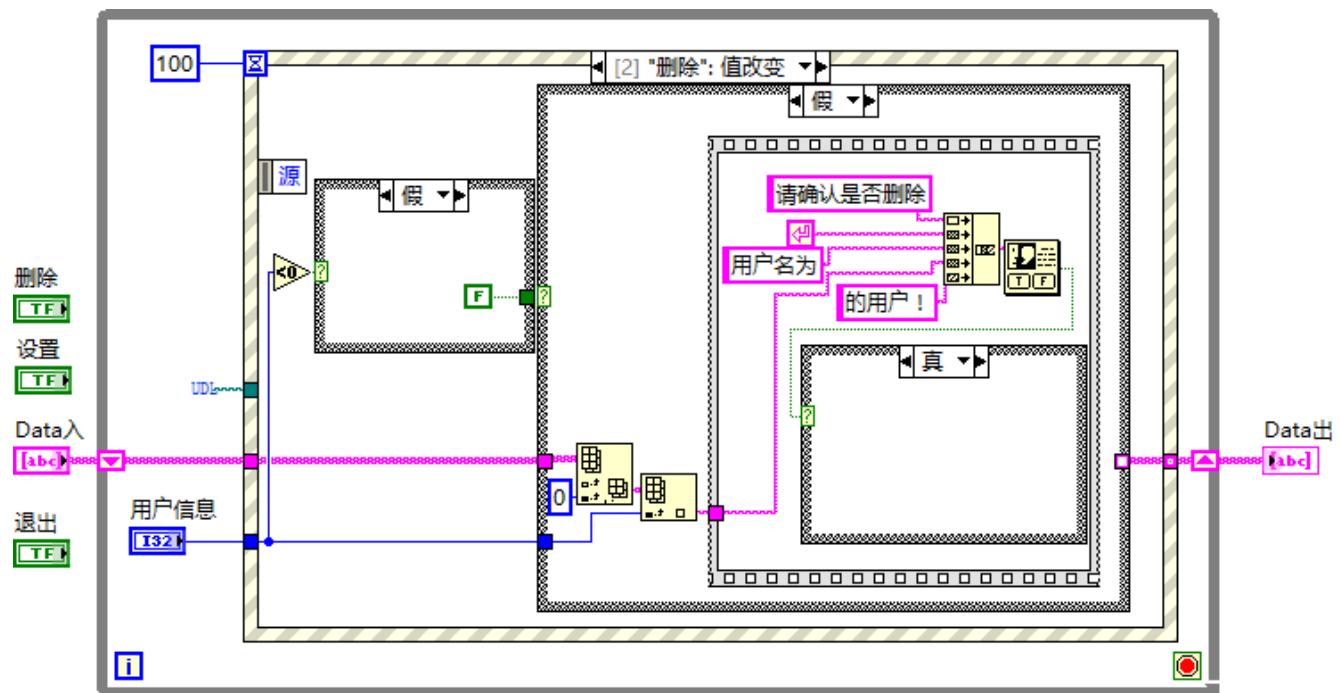
图?

g 在第二个【条件结构】的“真”分支中将“移位寄存器”两端相连。如图?所示。



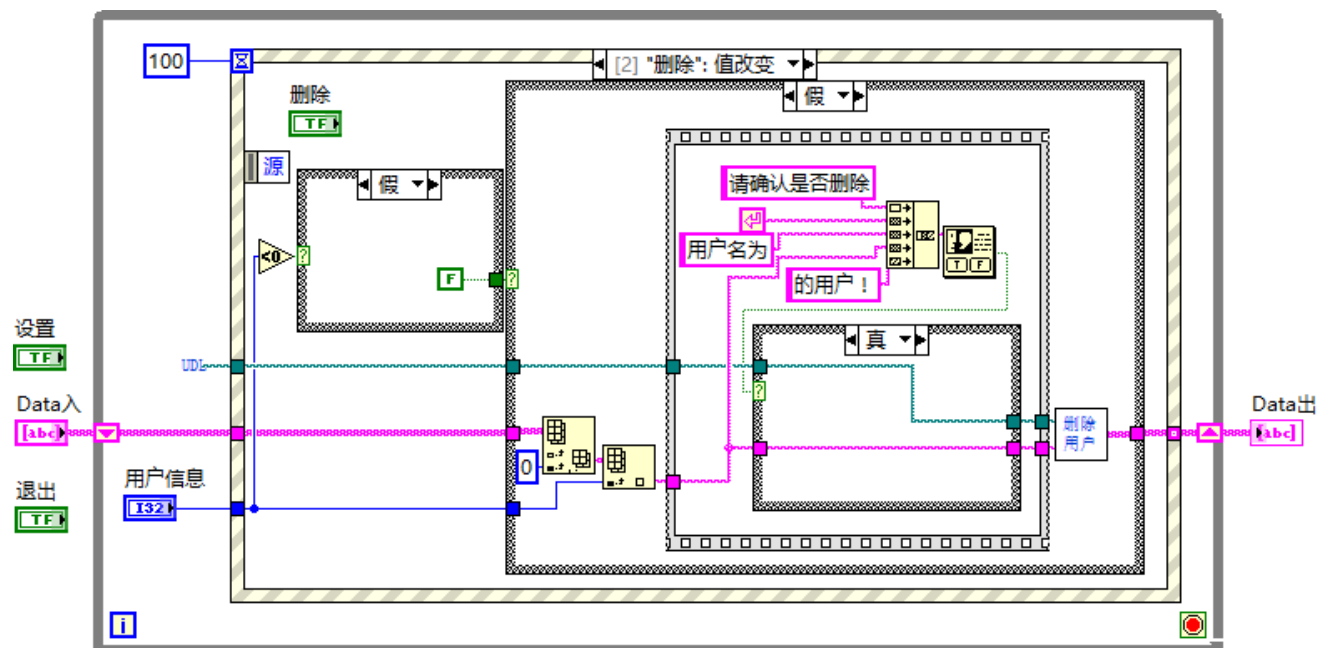
图?

在第二个【条件结构】的“假”分支中创建两个【引索数组】相连，前一个【引索数组】的“引索列”为 0，后一个【引索数组】的“引索”与【用户信息】相连。【顺序结构】中创建【连接字符串】、【字符串常量】和【回车键常量】, 将【连接字符串】的输出端与【双按钮对话框】相连。【双按钮对话框】的输出端与第三个【条件结构】的“分支选择器”相连。连接方式如图? 所示。



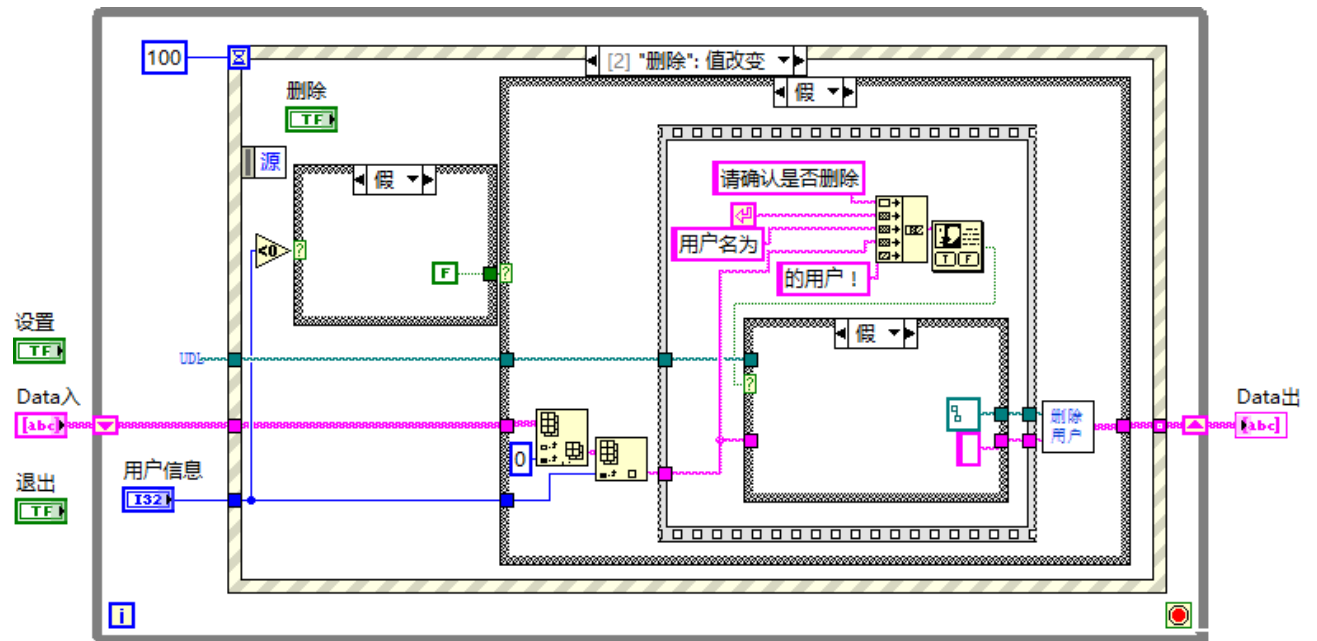
图？

添加【删除用户.vi】在第三个【条件结构】的“真”分支中创建如图？所示连接。



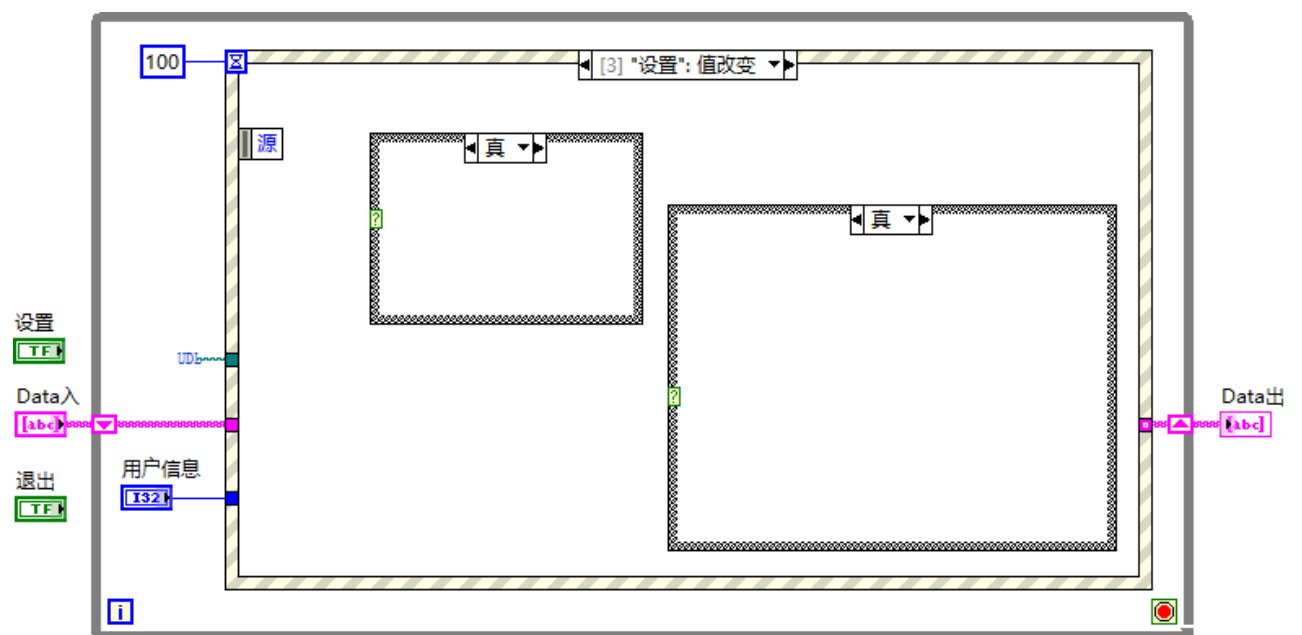
图？

在第三个【条件结构】的“假”分支中创建【路径常量】和【字符串常量】。建立如图？所示连接。





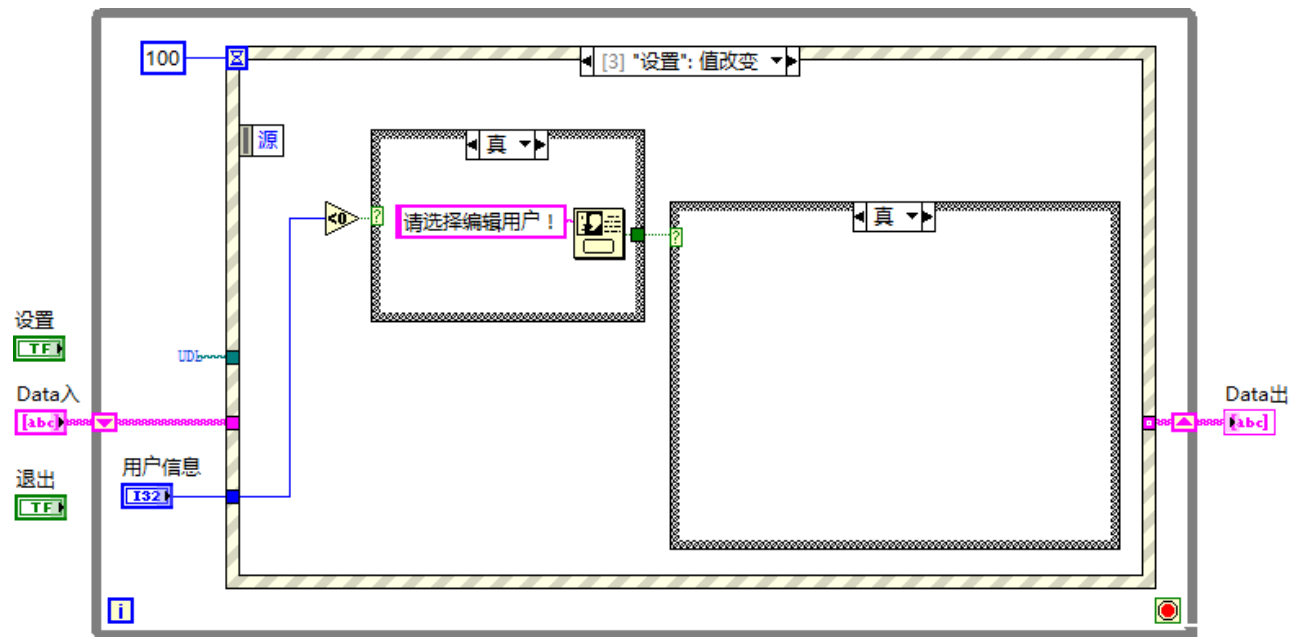
图?

h 在【事件结构】的“3”分支中创建二个【条件结构】，结果如图?所示。



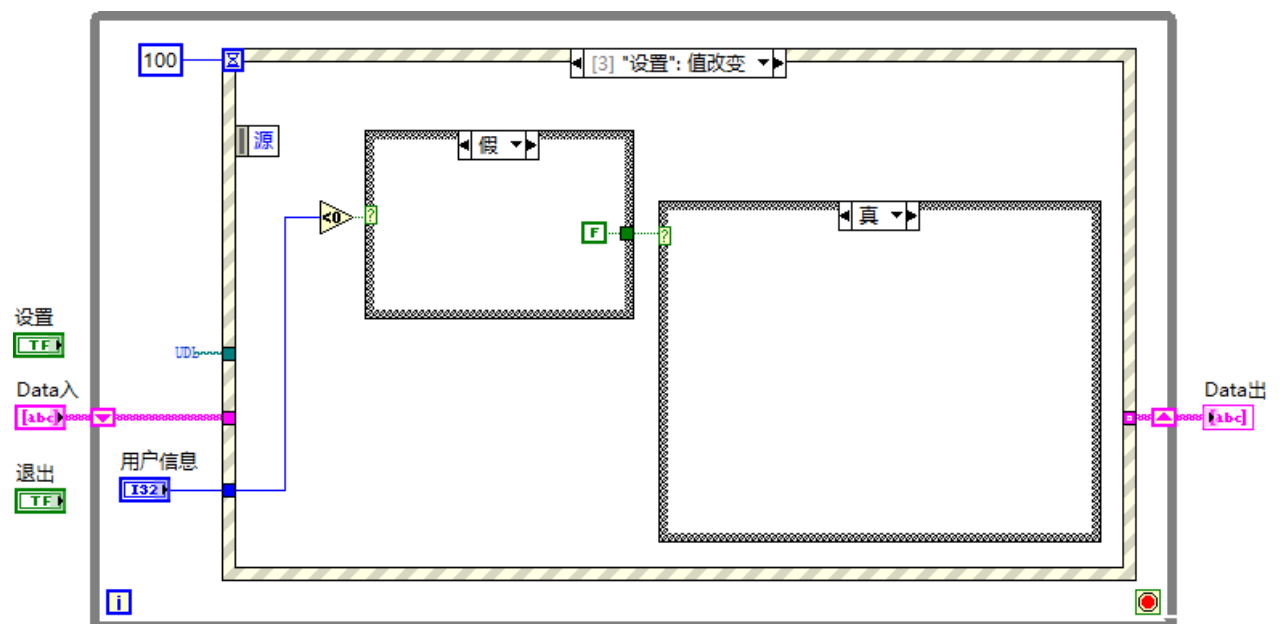
图?

创建【小于 0?】，在第一个【条件结构】的“真”分支中创建【单按钮对话框】并用【字符串常量】为其赋值，内容为“请选择编辑用户！”。如  所示。并将【单按钮对话框】的输出端与第二个【条件结构】的“分支选择器”相连。结果如图? 所示。

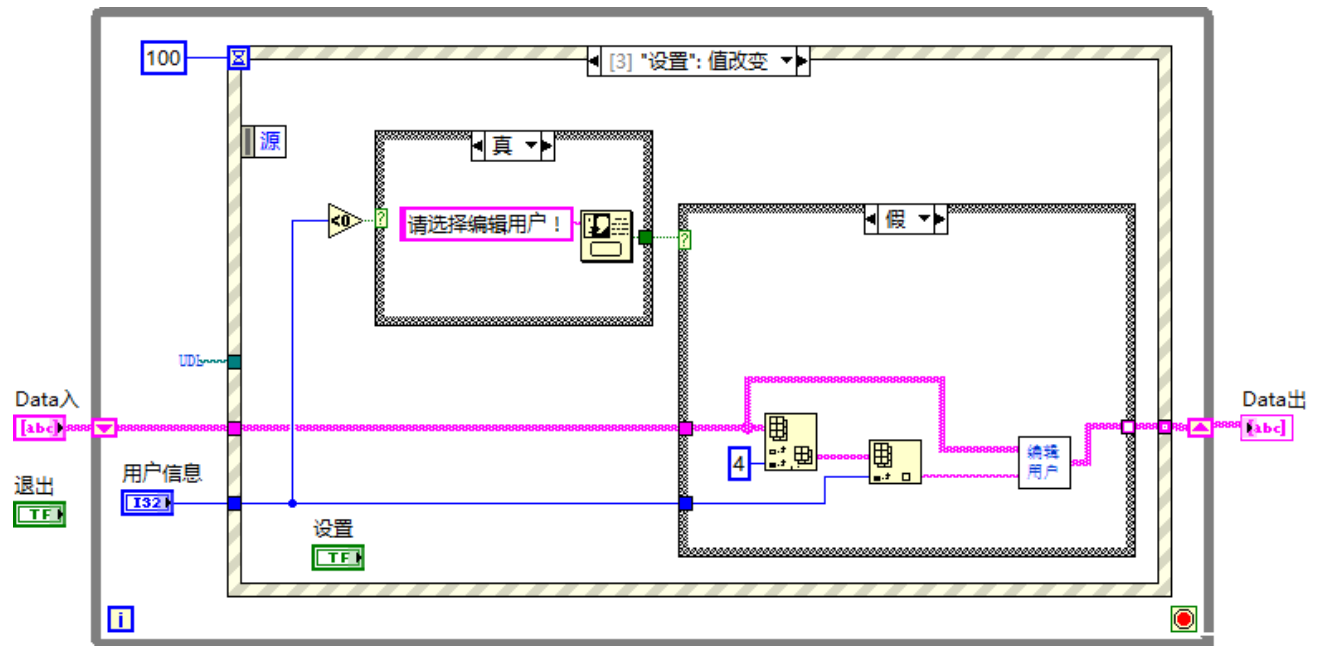


图?

在“假”分支中创建【假常量】，并与第二个【条件结构】的“分支选择器”相连。  
结果如图? 所示。

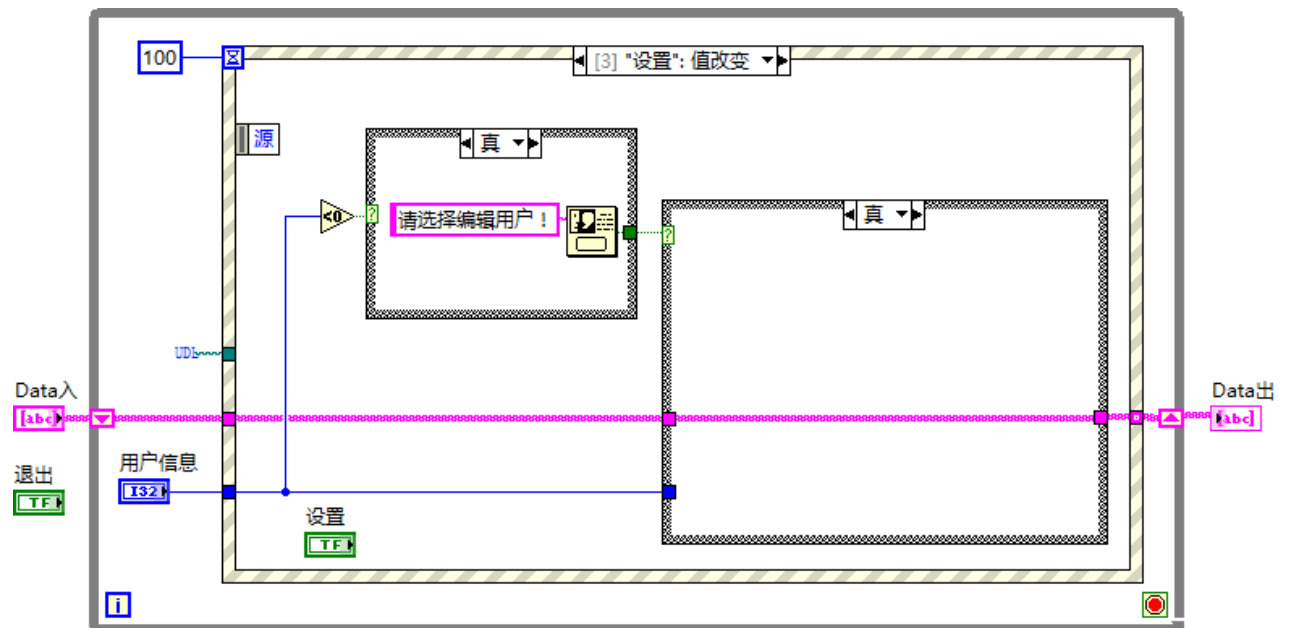


在第二个【条件结构】的“假”分支中创建两个【索引数组】相连，前一个【索引数组】的“索引列”为 4，后一个【索引数组】的“索引”与【用户信息】相连。添加【编辑用户.vi】，连线方式如图? 所示。



图?

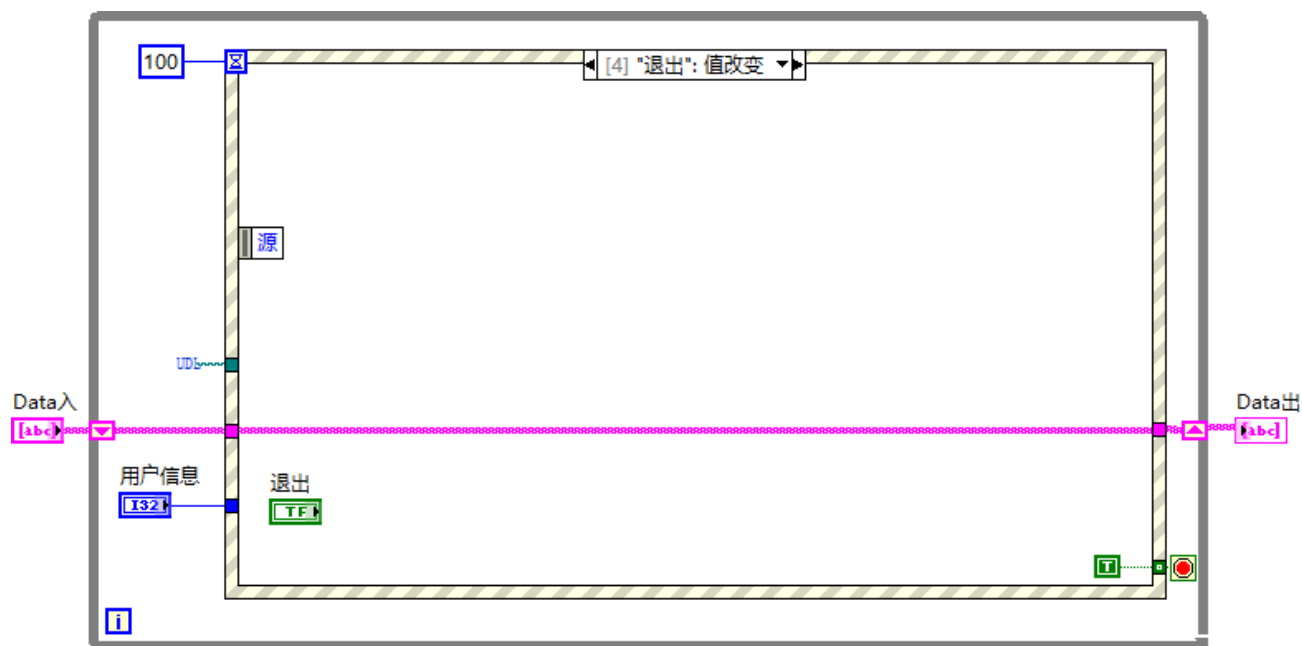
在第二个【条件结构】的“真”分支中将“移位寄存器”的两端相连。结果如图?所示。



图?

j 在【事件结构】的“4”分支中将“移位寄存器”两端相连。创建【真常量】与【While 循环】的“循环条件”相连。结果如图?所示。

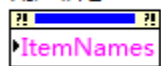




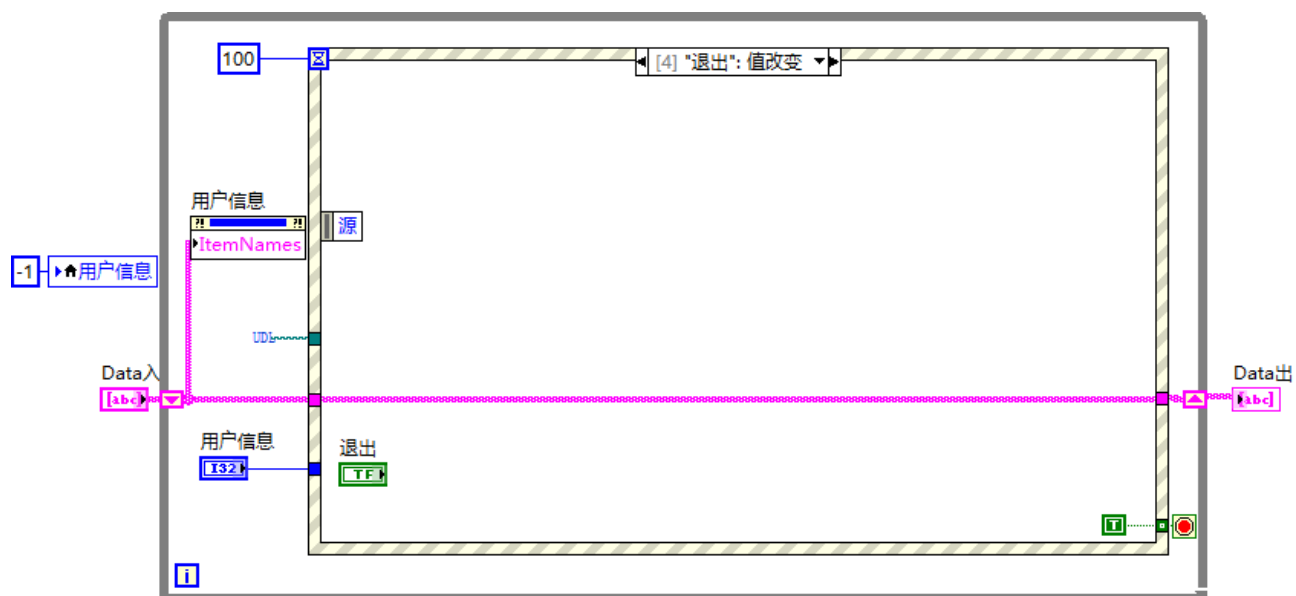
图?

k 在【用户信息】上右击选择<创建>/<属性节点>/<项目>, 转换为输入控件。如


用户信息




所示。并与【Data 入】连接。为【用户信息】创建【局部变量】并赋值【数值常量-1】，结果如图? 所示。



图?

1 编辑连线模式, 选择  连线模式。连线框左半边连接【Data 入】, 右半边【Data 出】。建立连接完成后, 再将他们隐藏。

- m 编辑 VI 图标，改名为“用户管理”。如  所示。
- n 编辑好前面板后，设置 VI 属性。设置方式与“修改密码.vi”的方式一致。
- p 保存 VI 至 Data.mdb 在同一级目录下，并改名为“用户管理”。

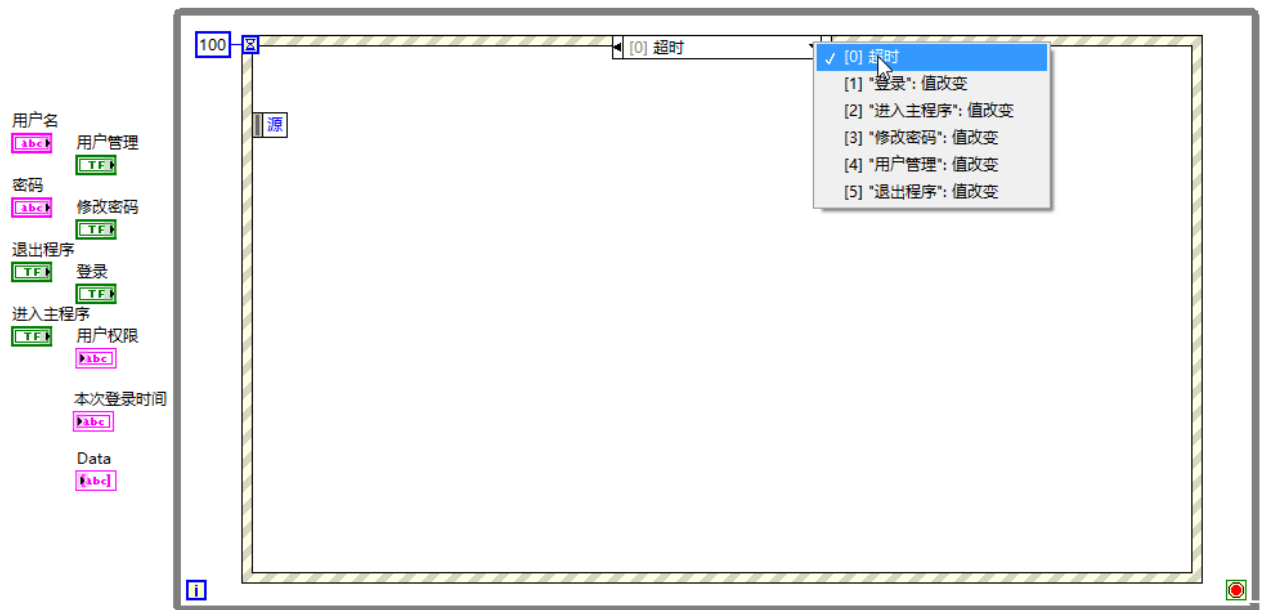
### 3.3.5 创建登录程序

① 在前面板创建两个【字符串输入控件(银色)】(分别将标签改名为“用户名”和“密码”)、两个【字符串显示控件(银色)】(分别将标签改名为“用户权限”和“本次登录时间”)、五个【空白按钮(银色)】(分别将标签名改为“登录”、“进入主程序”、“修改密码”、“用户管理”和“退出程序”，隐藏“标签”。同时将“布尔文本”改成相同内容(在控件上右击选择<显示项>/<布尔文本>))以及【表格(银色)】(将标签名改为“Data”并转换为显示控件)，结果如图? 所示。



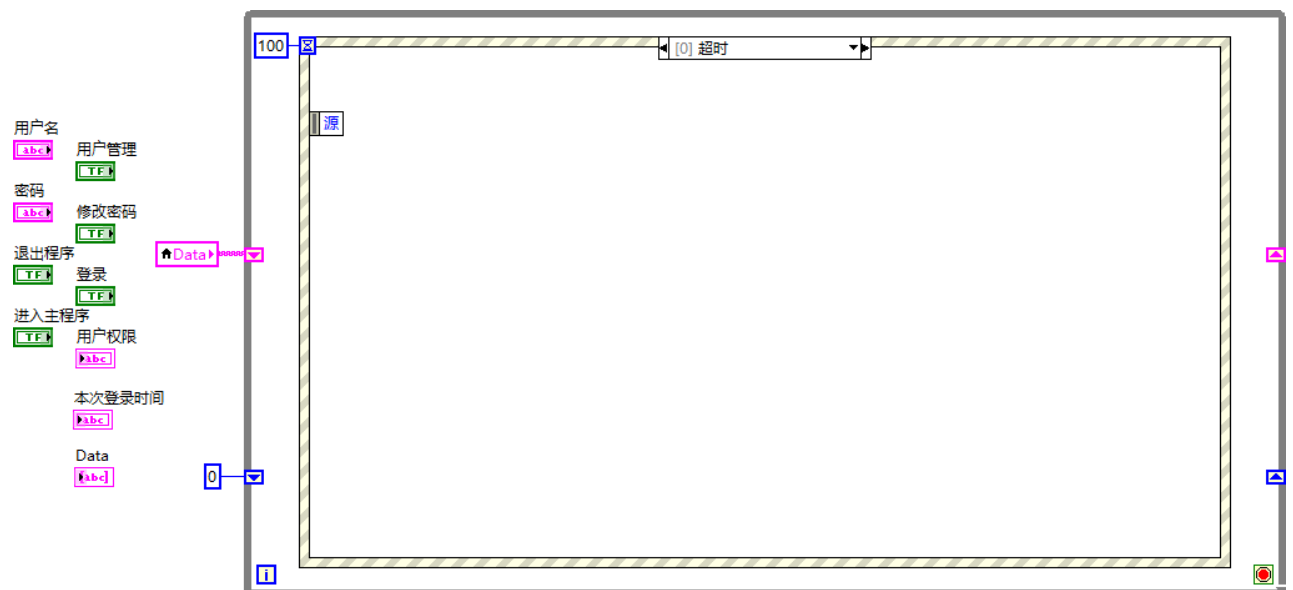
图?

② 在程序框图内创建【While 循环】和【事件结构】，编辑和创建【事件结构】的项并创建超时时间为 100ms。结果如图? 所示。



图?

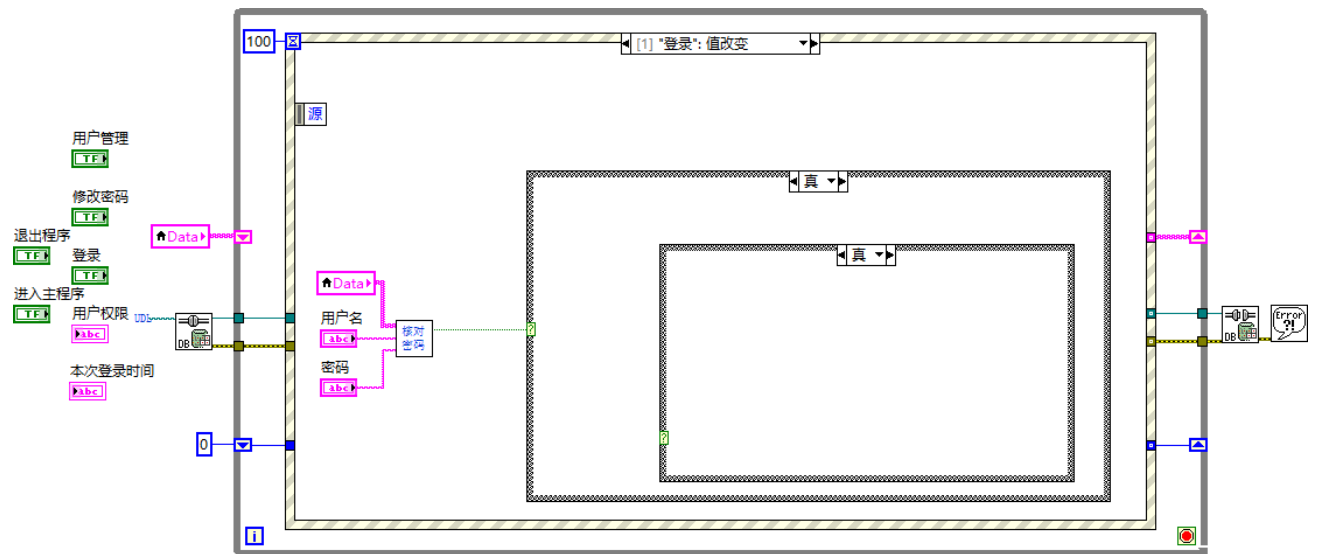
③ 为【Data】创建【局部变量】(转换为读取), 再为【While 循环】创建两个“移位寄存器”。将【Data 的局部变量】的输出端与一个“移位寄存器”的输入端相连。另一个“移位寄存器”赋值【数值常量 0】结果如图? 所示。



图?

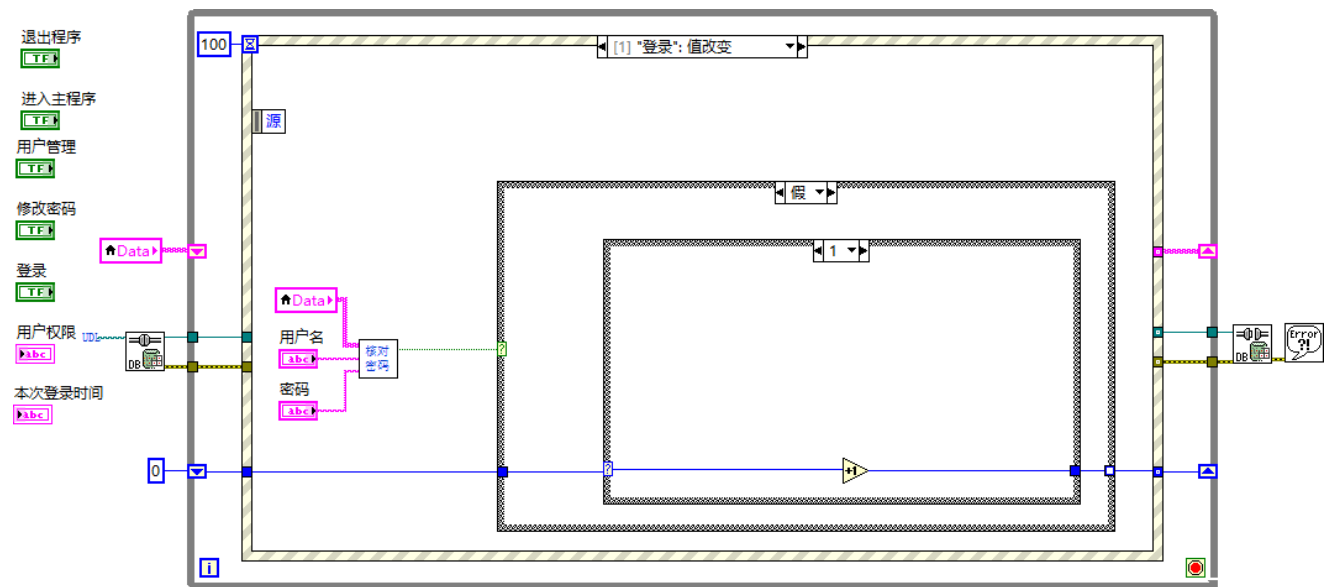
④ 在【事件结构】的“0”分支创建如图?所示结构。





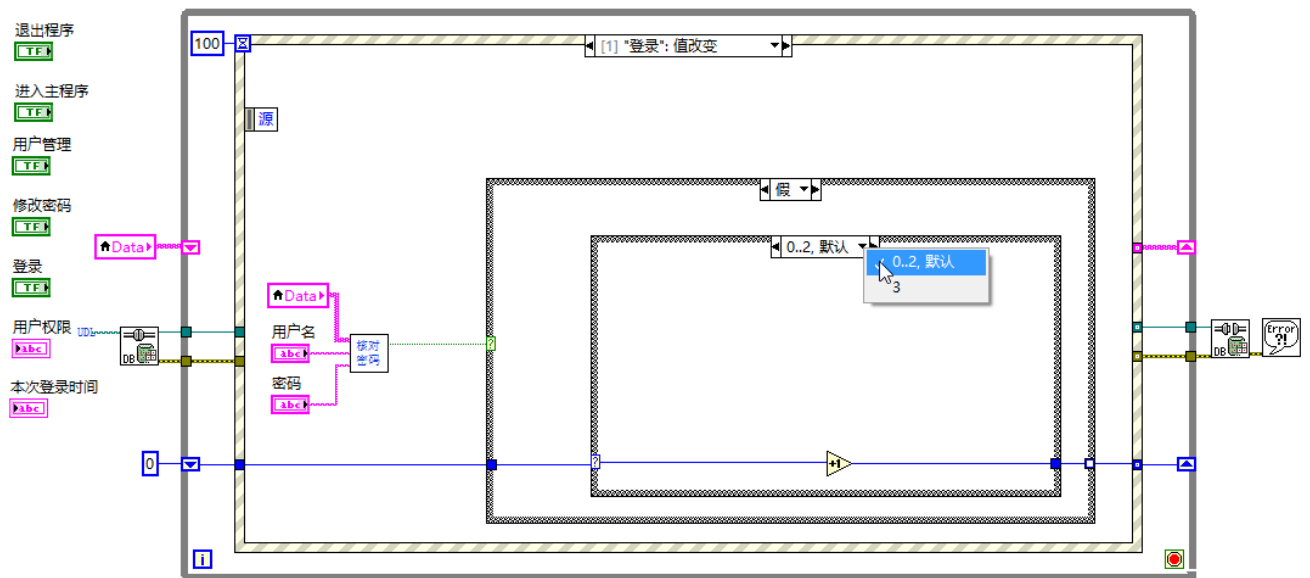
图？

在第一个【条件结构】的“假”分支中再创建一个【条件结构】。将下面的“移位寄存器”输入和输出端相连(中间创建【加 1】)。结果如图？所示。



图？

将第一个【条件结构】的“假”分支中的【条件结构】的“选择器标签”改为 “0..2”（将此标签设为默认，右击选择<本分支设为默认分支>）和“3”。结果如图？所示。

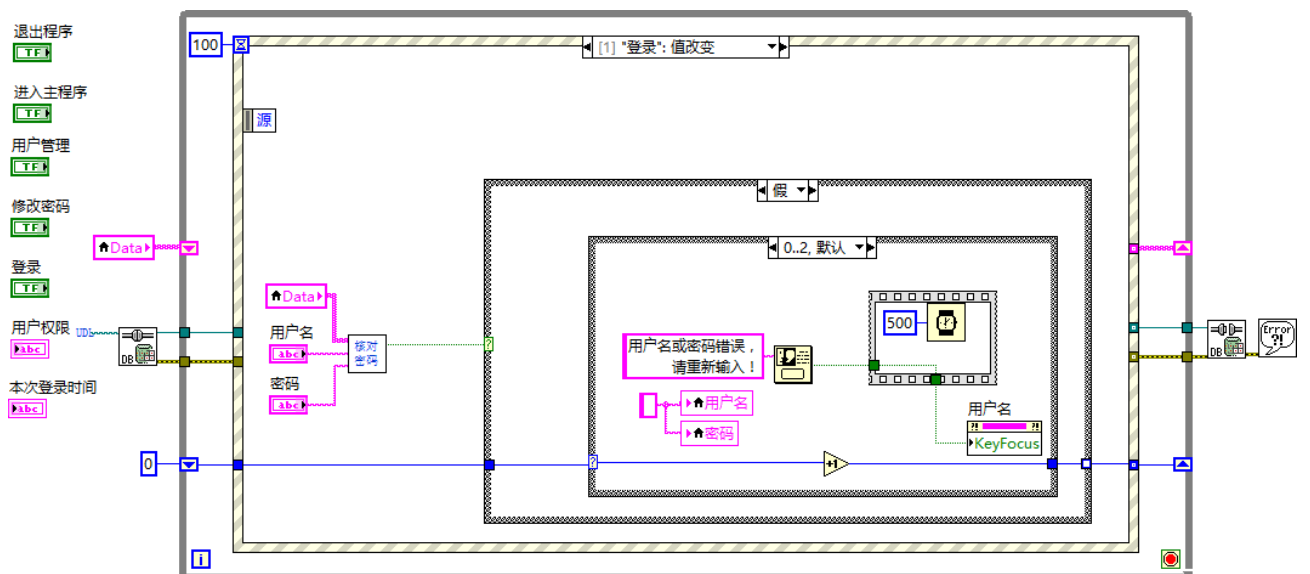


图?

在“0..2,默认”分支中创建【用户名】和【密码】创建【局部变量】，并赋值字符串常量】。创建【单按钮对话框】并用【字符串常量】赋值，内容为“用户名或密码错误，请重新输入！”。如

所示。创建【等待(ms)】并赋值 500ms，外部创建【顺序结构】。为【用户名】创建【键选中属性节点】(转换为写入)

与【单按钮对话框】输出端相连。结果如图?所示。



图?

“3”分支中将下面的“移位寄存器”输入与输出端相连。创建【单按钮对话框】并用【字符串常量】赋值，内容为“输入错误的用户名或密码次数超出限制，


请联系系统管理员确认您的用户名和密码！”。如

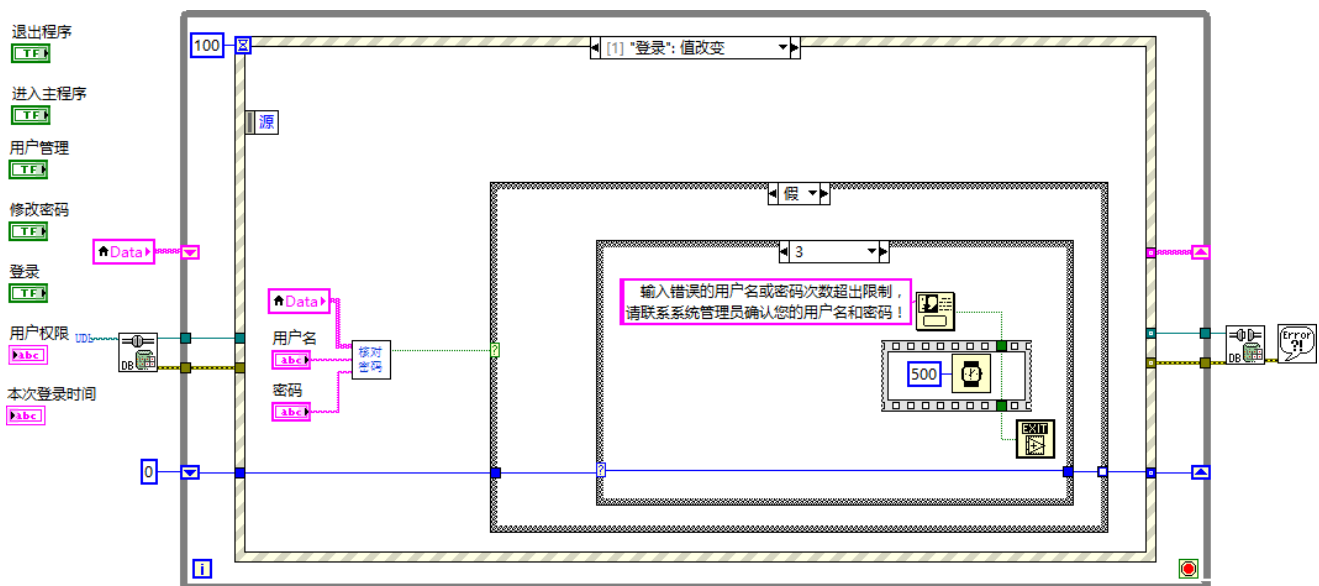
输入错误的用户名或密码次数超出限制，  
请联系系统管理员确认您的用户名和密码！



所示。

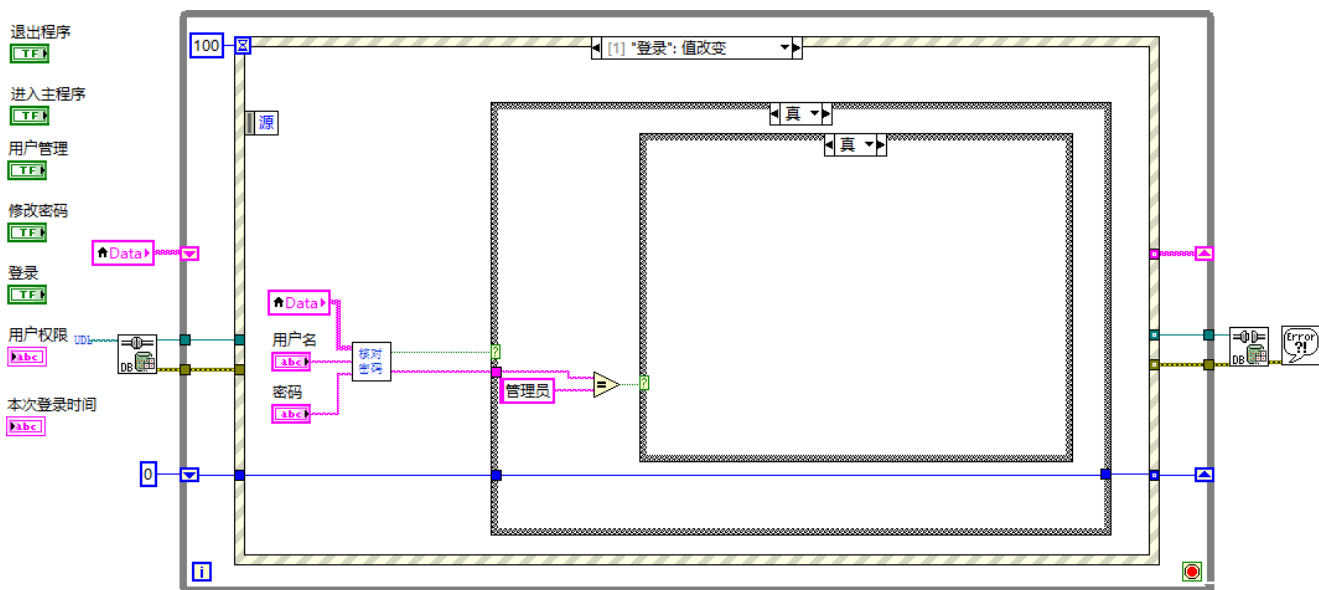
创建【等待(ms)】并赋值 500ms，外部创建【顺序结构】。创建【退出 LabVIEW】

(空白处右击选择【编程】/【应用程序控制】/【退出 LabVIEW】) 。并与【单按钮对话框】输出端相连。结果如图?所示。



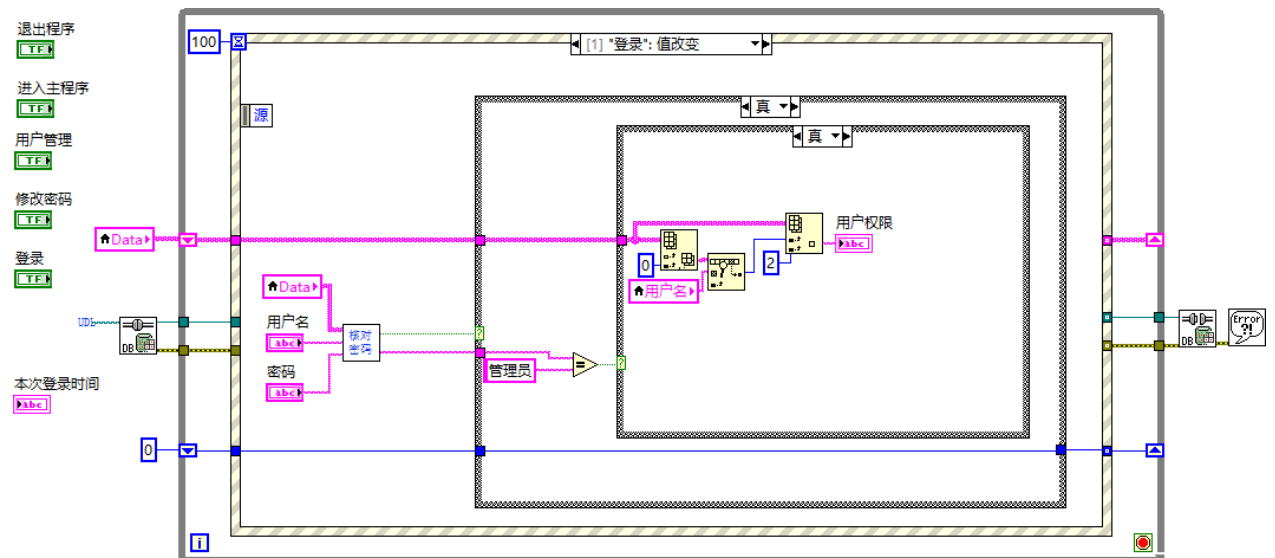
图?

在第一个【条件结构】的“真”分支中创建【等于?】并在“y”创建【字符串常量】并赋值“管理员”。【等于?】的输出端与第二个【条件分支】的“分支选择器”相连。结果如图?所示。



图？

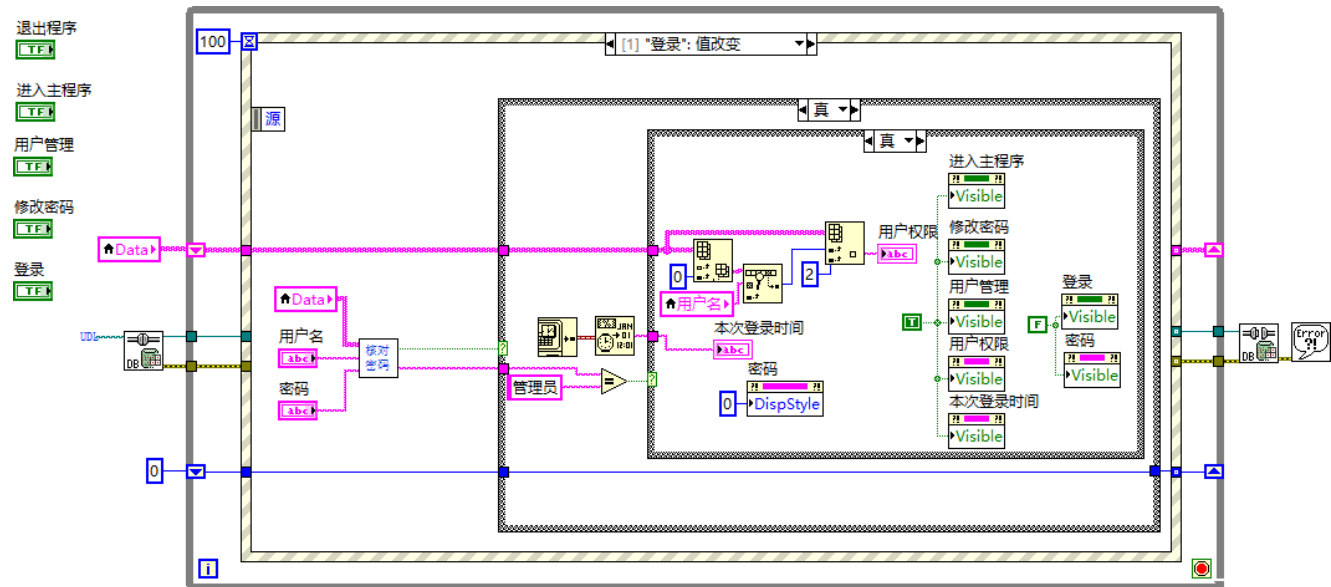
在第一个【条件结构】的“真”分支中的【条件结构】的“真”分支中创建两个【索引数组】(“索引列”分别为 0、2)和【搜索一维数组】。创建【用户名】的【局部变量】(转换为读取)与【搜索一维数组】的“元素”端相连。结果如图？所示。



图？

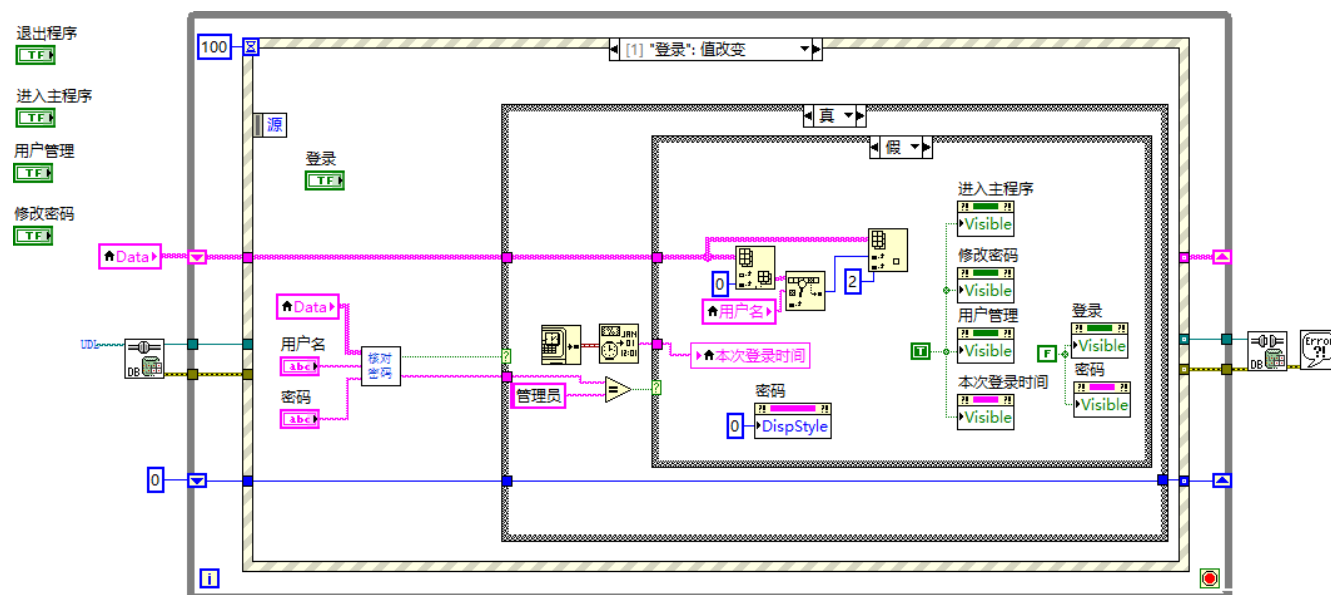
接着创建【进入主程序】、【修改密码】、【用户管理】、【用户权限】、【本次登录时间】、【登录】、【密码】的【可见】属性节点(全部转换为写入)。将【进入主程序】、【修改密码】、【用户管理】、【用户权限】和【本次登录时间】赋值【真常量】。将【登录】和【密码】赋值【假常量】。为【密码】创建【显示样式】属性节点(转换为写入)并赋值 0。在创建【获取日期/时间(秒)】和【格式化日期/时间字符串】并与【用户权限】相连。结果如图？所示。





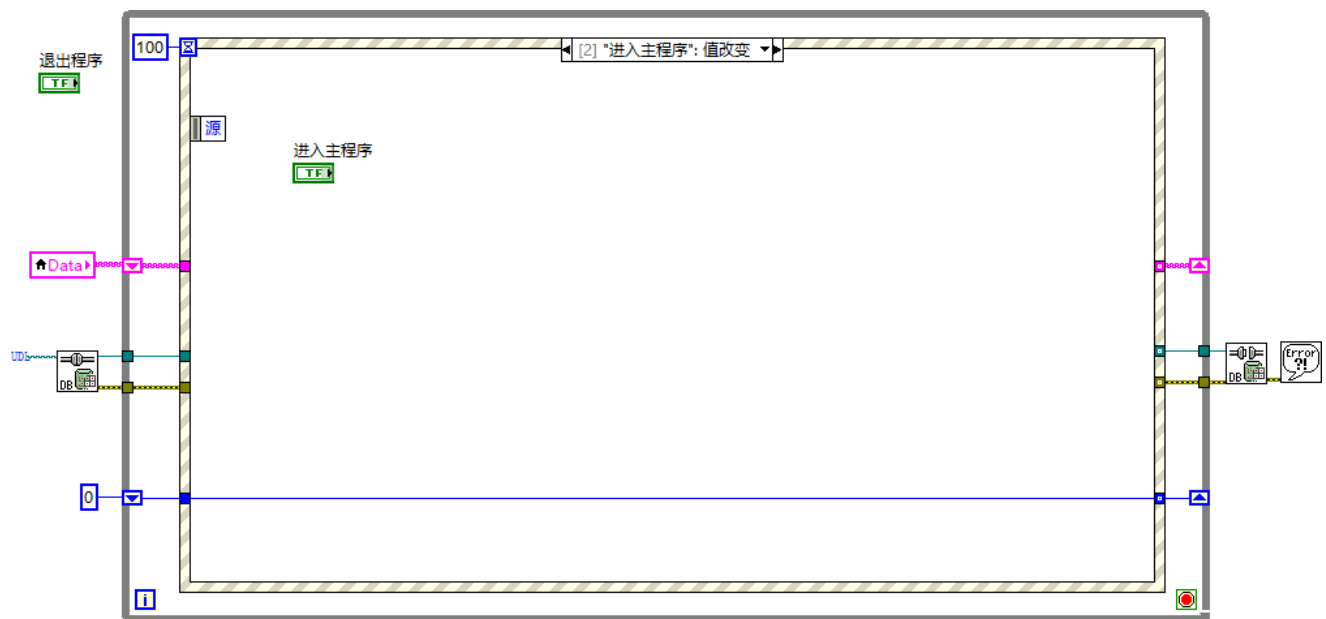
图？

第一个【条件结构】的“真”分支中的【条件结构】的“假”分支中创建函数与“真”分支相同。只是将【用户权限】和【本次登录时间】换成了【局部变量】。以及【可见】中的【用户管理】去除。结果如图？所示。



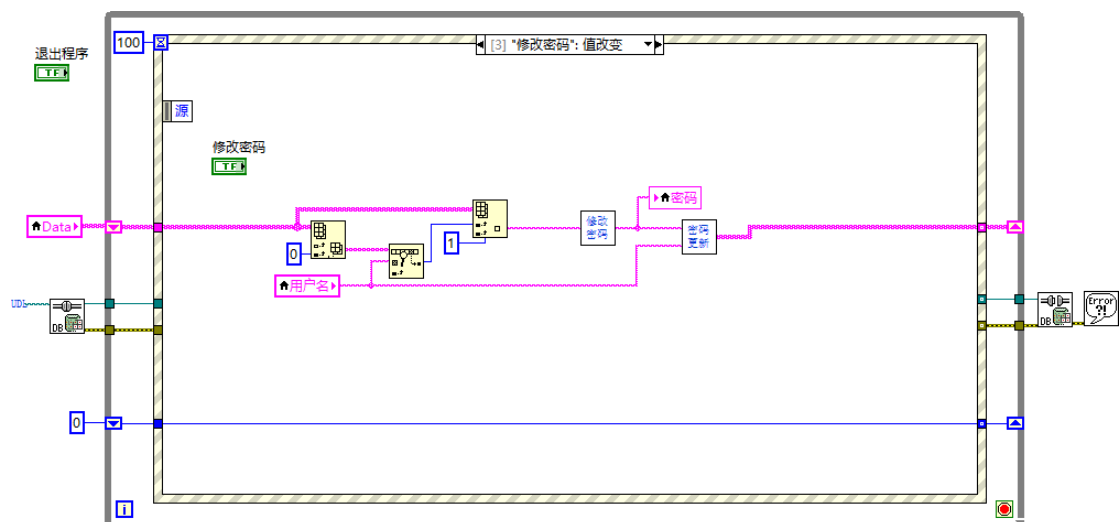
图？

⑥ 在【事件结构】的“2”分支中，可根据具体需要选择。故不陈述。结果如图？所示。



图?

⑦ 在【事件结构】的“3”分支中添加【修改密码.vi】和【密码更新.vi】。创建两个【索引数组】(“索引列”分别设为 0、1)和【搜索一维数组】。创建【用户名】和【密码】的【局部变量】(【用户名】的【局部变量】转换为读取)。下面的“移位寄存器”输入与输出相连。结果如图? 所示。



图?

⑧ 在【事件结构】的“4”分支中添加【用户管理.vi】输出和输入端与上面的“移位寄存器”相连。下面的“移位寄存器”输入与输出相连。结果如图? 所示。



图？

 Visible



设置方式与“修改密码.vi”的方式一致。

录下，并改名为“登录界面”。

说明:

① 文件夹位置改变时，将【UDL 路径.vi】.udl 文件路径改变即可，改变并设为默认值即可。

② 重复“UDL 与数据库连接”的过程。