华中科技大学计算机科学与技术学院

《C 语言程序设计》课程设计

实验报告

题目：购物网店铺信息管理系统

专业：计算机科学与技术

班级：0910

学号：U200915185

姓名：黄志耿

成绩：

指导老师：

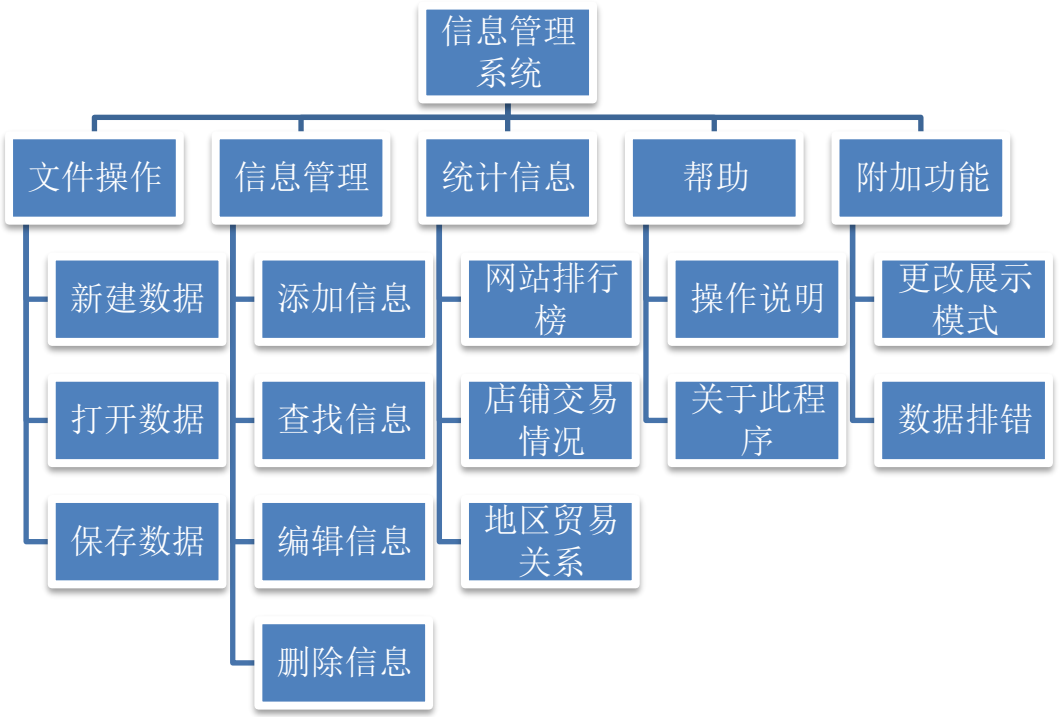
完成日期：2010 年 10 月 20 日

目录

(一) 程序设计环境

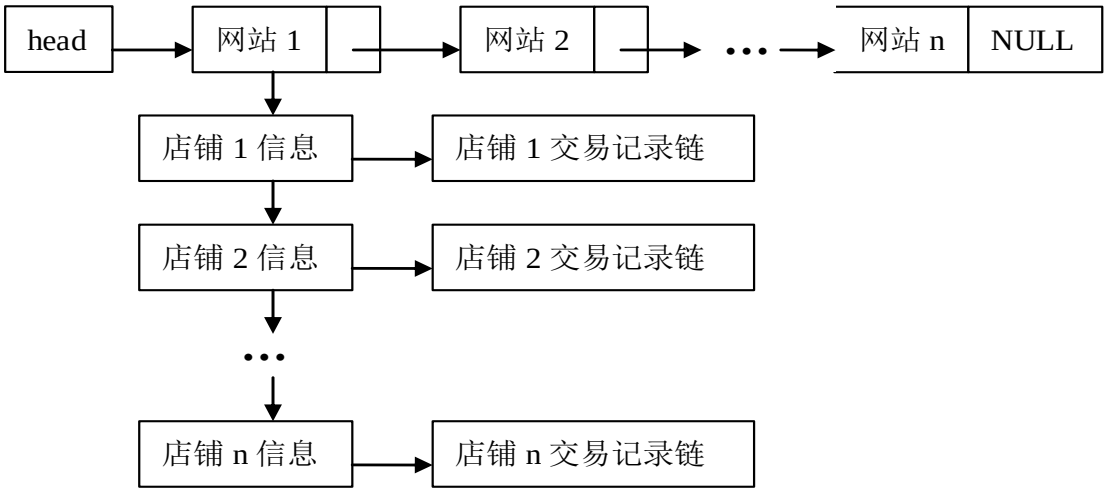
- 操作系统：Ubuntu 10.04
- 代码编辑器：gedit
- 编译器：gcc
- 调试方法：使用 g_message 函数将程序运行时的某些信息(如某些重要变量的值)输出到终端

(二) 系统功能模块结构图



(三) 数据结构设计及用法说明

1. 系统采用三个方向的十字交叉链表(如下图)：



2. 主要用法说明：

i. 程序声明了对应三个结构体。其中网站、店铺结构体中各包含了一个指向下级链表的结构体指针，用于将它们联系起来。

ii. 添加：添加网站比较简单，而添加店铺或交易信息则会要求输入已经存在的网站或店铺编号。

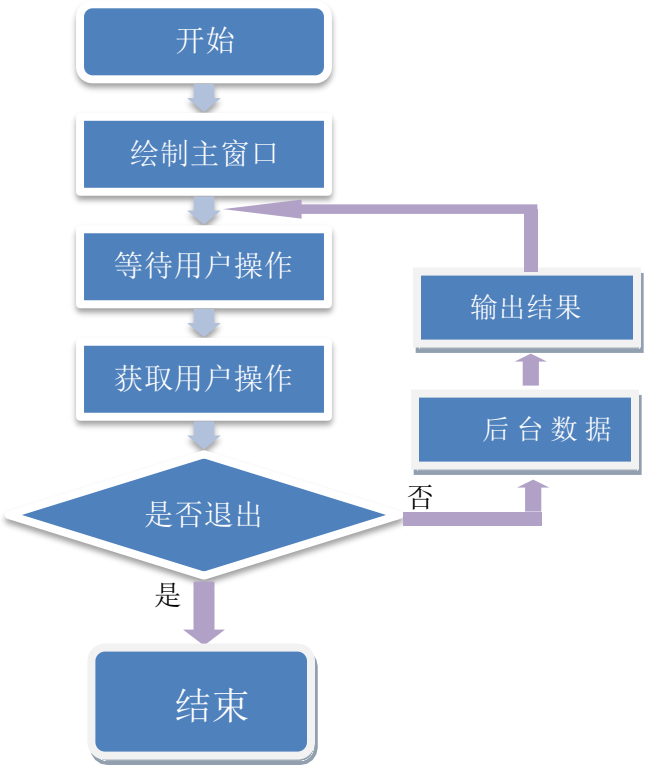
iii. 编辑：通过层层的遍历(网站则只需一层)找到对应节点并将其进行修改。如果所编辑的店铺(或交易)被修改了所属网站(店铺)编号，将在链表中移除其原来位置并找到插入到对应的网站(店铺)链中。

iv. 删除：删除操作会删除该节点信息以及其下级链表(如果有的话)的所有信息。另外，由于系统采用自动编号的形式，故执行删除操作后会对该节点后面所有节点

进行编号自减(即等于原来的编号减 1)，同时其下级链表中的所有节点都会进行重新编号。

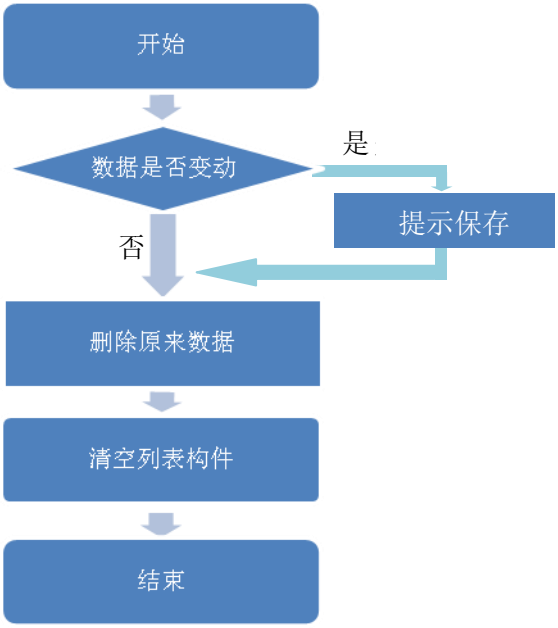
(四) 程序结构

1. 程序整体结构：

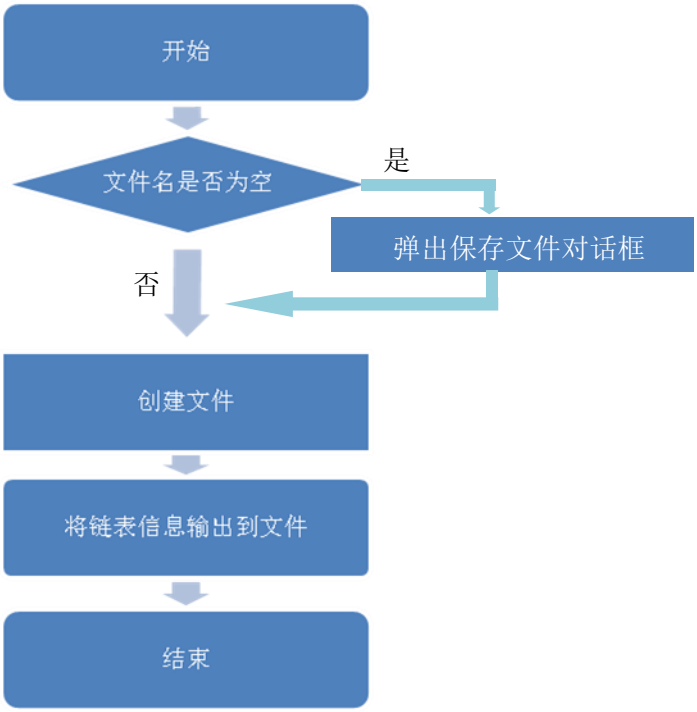


2. 部分功能流程图

i. 文件操作

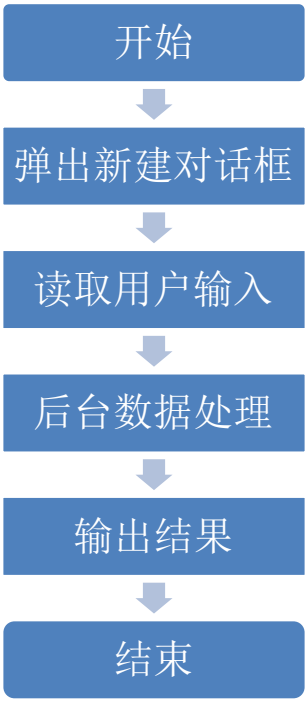


新建文件流程图

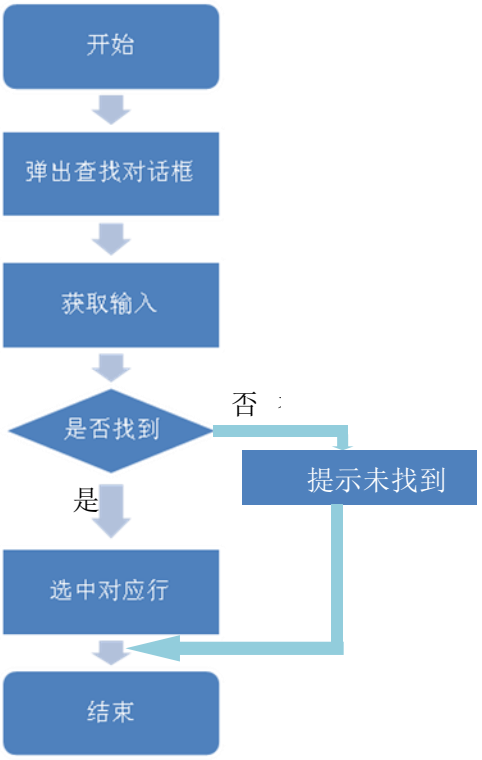


保存文件流程图

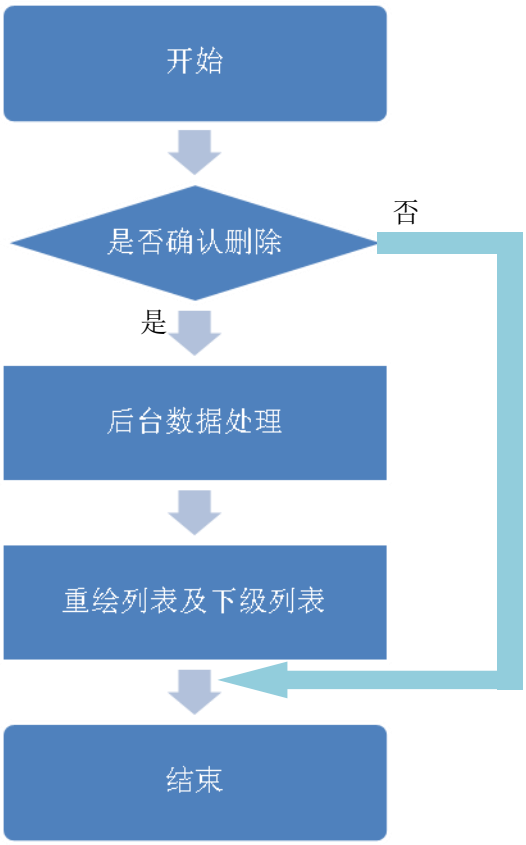
ii. 信息管理



新建网站流程图

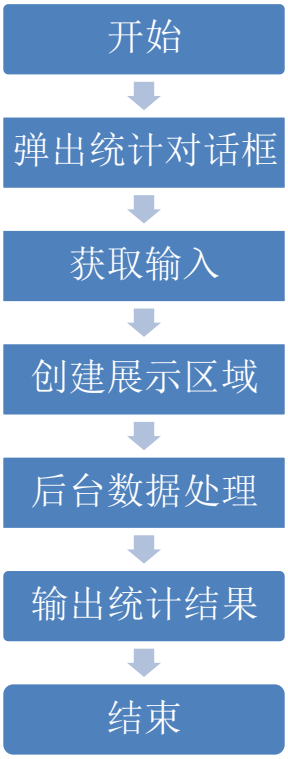


查找网站流程图



删除网站流程图

iii. 统计模块



(五) 各模块功能

1. 文件操作：

i. 新建文件：清空原来内存中所有链表所有信息(如数据未保存会有相关提示)，然后新建一数据文件并重新开始输入各种信息。

ii. 打开文件：打开磁盘原先已存在的数据文件(只有网站信息，其他两个文件将自动载入)，将其所有信息读入并显示在列表构件中。

iii. 保存文件：将内存中所有链表的信息全部输出到文件中(对应有三个文件，但只会要求输入第一个文件的文件名，其他两个文件为该文件名加上“_shop”或

“_deal”)。包括保存与另存为功能。同样，如果原先数据未保存将会有相关提示。

2. 信息管理

i. 添加信息：分为添加网站、添加店铺、添加交易三个部分。不过，此功能只有一个菜单选项跟一个工具条按钮。因为程序会根据当前所处页面判断并调用相对应的功能函数。添加完成后会自动加入到 clist 中。

ii. 查找信息：包括查找网站、查找店铺、查找交易。同时，用户可选择通过哪一个数据项进行查找，例如可根据网站名称或网站地址进行查找。若找到信息则会选中该行并将使其可视，即如果信息太多，会自动将该行移动到当前 clist 的第一行。

iii. 编辑信息：包括编辑网站、编辑店铺、编辑交易。该功能会将编辑对象原来的所有信息展示出来，用户可根据需要修改其中一项或多项信息。不过，用户无法修改信息的编号以及其他系统自动处理的数据。

iv. 删除信息：包括删除网站、删除店铺、删除交易。由于数据删除后具有不可恢复性，故用户选择此功能时系统将会弹出确定对话框，里面写有删除对象的信息及提示用户是否删除的语句。注意：删除网站或店铺不仅会删除其下级链表的信息，同时将对其下级链表中所有节点进行重新编号。

3. 统计信息

i. 网站排行榜：首先提示输入欲统计的年月份。之后经过后台数据处理之后将会新建一个 notebook 的标签页(如果原来没有此标签页)并将统计结果以 clist 的方式展示给用户查看。

ii. 店铺交易情况：提示输入统计的年份后用上面的方式将统计结果输出。

iii. 地区贸易关系：此功能要求用户输入欲统计的两个地区的名称以及年月份。若两个地区在此月份无贸易关系将以对话框形式提示用户，否则将以 clist 的方式展示出来。

4. 帮助：

i. 操作说明：以文字形式用对话框告知用户使用本程序应该注意的地方以及附加功能的相关介绍。

ii. 关于此程序：输出版权信息以及一些参考书籍。

5. 附加功能：

i. 更加展示模式：系统有“展开全部”以及“收起其他”两种展示模式。在“展开全部”模式下，用户可查看到所有信息；而在“收起其他”模式下，可以只查看某项信息对应的下级信息。如用户在“网站信息”标签页中选择网站“淘宝”，那么在切换到“店铺信息”时将只看“淘宝”下面所有店铺的信息，同样，从“网

站信息”切换到“交易信息”时也将只看到“淘宝”网站下所有的交易信息。此模式在信息里过多的情况下将对用户快速检索并找到自己想要的信息有所帮助。

ii. 数据排错：此功能是为了保证系统的健壮性以及数据的正确性。由于系统采取自动编号的方式，同时网站的店铺总数以及店铺的交易次数均由系统自动计算得出，因而用户无法手动修改此类信息。如果在使用过程中发生某些意料不到的事件而导致此类信息出现错误，用户可选择此功能对数据进行检查并改错。不过须要注意的是，此功能将会对所有信息进行重新编号。

(六) 程序特色

1. 用 C 语言实现了普通应用程序的外观及基本特性。如菜单功能、快捷键功能、工具条(可拖动)功能。使得用户可以十分方便查看、编辑、查找、删除各类信息。
2. 程序有多个地方检查了用户的输入是否合法，同时用简单的形式提示用户检查输入。
3. 程序有多个地方做了人性化化处理，如添加信息后会默认选择新添加的数据行，为方便用户快速检索到所需信息特意提供了“收起其他”模式，如点击列表某一列的标签会对该列进行排序等。

4. 注重数据安全。如果原先数据发生改变，在执行新建、打开或关闭功能时都会提示用户是否保存原来的数据。

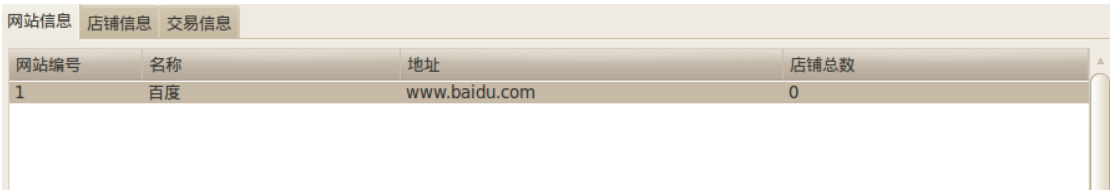
(七) 试验过程及结果

1. 信息管理(只展示网站的相关操作，其他两类操作相似)

i. 添加网站：

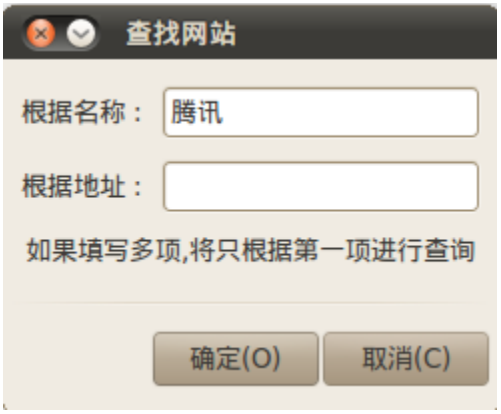
a) 添加网站对话框：

b) 添加完成



ii. 查找网站：

a) 查找网站对话框：



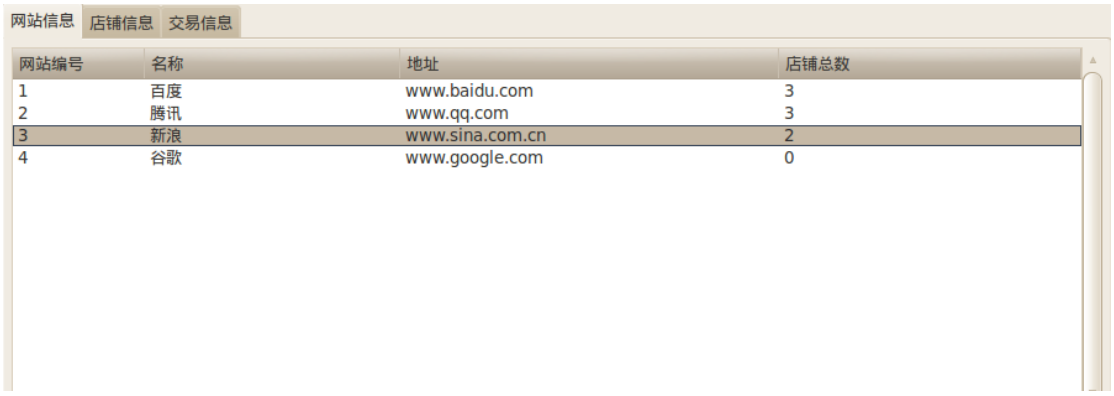
b) 找到结果：

未找到任何结果：

iii. 编辑网站：

a) 编辑网站对话框：

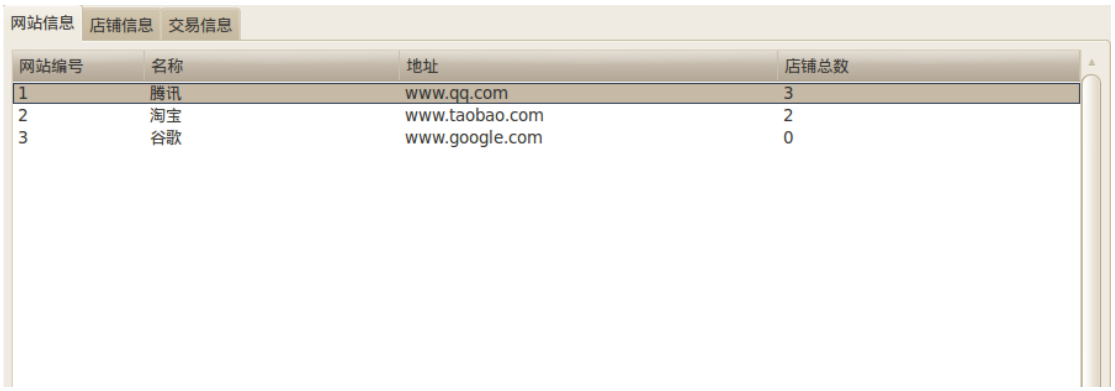
b) 编辑完成：



iv. 删除网站：

a) 提示是否删除的对话框：

b) 删除完成：



2. 统计信息(只展示第一个统计功能，其他类似)：

i. 统计网站对话框：

ii. 统计结果：

3. 更改展示模式：

在选择菜单项“收起其他”后，在“网站信息”页面中选择编号为 2 的网站，切换到“店铺信息”标签页后：

从“网站信息”切换到“交易信息”标签页后：

(八)　体会

1. 用 GTK 库绘制图形界面不仅比 TC 方便很多，其所画界面更比 TC 美观很多。另外，该库天生就具有支持鼠标、支持中文的特性，不需要你加入任何代码就可以自然、完美地支持鼠标操作、中文显示，这一点也是 TC 无法企及的。所以，选择一个优秀的工具有利于提高自己的开发效率以及强化开发程序的人性化程度。
2. 使用指针虽然方便，但是对其操作也有可能发生危险，特别是当你未给指针分配内存单元时，更要格外小心，因为很有可能你对此指针的操作会修改到内存中其他地方的数据。
3. 数组指针不同于普通指针。其使用方法不能完全等价。

4. 写大程序时要特别注意程序的模块化，第一是要尽量让自己的函数只执行一个功能(这点我做得并不好)，第二是要将程序分为几大模块，同时每个模块用单独一个文件去描绘。
5. 编写过程中我经常遇到这种矛盾：为了尽量发挥代码的可重用性，我常常让某些函数执行多个功能，然而代价是使得这些函数变得稍显臃肿而复杂。如果要分成几个函数又会导致相同的一些代码被使用多次。当然这有可能是因为自己考虑得不够周到。
6. 调试是写程序很重要的一个部分。

(九) 参考文献

- 《实用技术：开发 linux 应用》：电子工业出版社 2000 年 1 月出版，作者：Eric Harlow，童寿彬等译
- 《LINUX 应用程序开发指南：使用 GTK+ GNOME 库》：机械工业出版社 2000 年 7 月出版，作者：许宏松等
- 《GTK 编程范例》：清华大学出版社 2002 年 11 月出版，作者：宋国伟
- 《C 语言程序设计》：科学出版社 2008 年 2 月出版，作者：曹计昌等

(十) 附录：程序清单及源代码

1. 程序采用多文件编译而成的，主要有以下几个文件：

   main.c, create_window.c, get_user_data.c, handle_data.c;

   create_window.h, get_user_data.h, handle_data.h

   以下为几点说明：

   i. main 函数位于 main.c 文件中。create_window.c 为前台展示模块，主要用于绘制主窗口以及为主窗口的菜单以及某些信号建立对应的回响函数，同时获取必要信息传递给 get_user_data.c(用户交互模块)里的某些函数；get_user_data.c 是用户交互模块，它获取用户的操作，并根据这些操作，利用前台展示模块里面某些函数传递的信息，调用对应的对话框以获取用户的输入信息或者获取用户的确定，以便交给 handle_data.c(后台处理模块)里面的某些函数进行数据处理；handle_data.c 为后台数据处理模块，主要处理交互模块传递过来的信息执行对应的链表和文件操作。

   ii. 每个头文件提供了其同名 C 文件的一些函数接口，用于被其他 C 文件使用。其中 handle_data.c 声明了必要的三个结构体。

   iii. 为方便编译，我另外写了一个 Makefile 文件，详细内容见后面。

2. 所有文件源代码

i. main.c

```
#include <gtk/gtk.h>
#include "create_window.h"
gint main(gint argc, gchar *argv[])
{
        gtk_init(&argc, &argv);
        CreateWindow();
        gtk_main();
        return 0;
}
```

ii. create_window.c

```
/*
 * 主窗口绘制模块
 *
 * 主要功能:绘制主窗口,同时为主窗口中建立回呼函数并获取必要的信息.例如当前选中标签页,当前选
 * 中行数
 */

  #include <gnome.h>
  #include <gtk/gtk.h>
  #include <stdlib.h>
  #include <libgnomeui/gnome-about.h>
  #include <libgnomeui/gnome-app-helper.h>
  #include <libgnomeui/gnome-messagebox.h>
  #include "get_user_data.h"

  #define SITE_COLUMN 4
  #define SHOP_COLUMN 7
  #define DEAL_COLUMN 6
  #define SITE_PAGE 0
  #define SHOP_PAGE 1
  #define DEAL_PAGE 2
  #define STATI_PAGE 3


  GtkWidget *window = NULL;
  GtkWidget *notebook;
  GtkWidget *dist[4];

  gint site_selected_row = -1;          /*存储网站页面选中的行数*/
```

```
gintshop_selected_row = -1;          /*存储店铺页面选中的行数*/
gint deal_selected_row = -1;         /*存储交易页面选中的行数*/

void OnFileNewActivate(void);
void OnFileOpenActivate(void);           /*选择"文件/打开"*/
void OnFileSaveActivate(void);           /*选择"文件/保存"*/
void OnFileSaveasActivate(void);         /*选择"文件/另存为"*/
void OnExpandAllActivate(void);          /*选择"文件/更多/展开全部"*/
void OnRetractOthersActivate(void);      /*选择"文件/更多/收起其他"*/
void OnCheckDataActivate(void);          /*选择"文件/更多/数据排错"*/
void OnFileExitActivate(void);           /*选择"文件/退出"*/

void OnAddInfoActivate(void);            /*选择"编辑/添加信息"*/
void OnSearchInfoActivate(void);         /*选择"编辑/查找信息"*/
void OnEditInfoActivate(void);           /*选择"编辑/编辑信息"*/
void OnDelInfoActivate(void);            /*选择"编辑/删除信息"*/

void OnStatiSiteActivate(void);          /*选择"统计/网站排行榜"*/
void OnStatiShopActivate(void);          /*选择"统计/店铺交易情况"*/
void OnStatiDealActivate(void);          /*选择"统计/地区贸易关系"*/

void OnHelpActivate(void);               /*选择"帮助/操作说明"*/
void OnAboutActivate(void);              /*选择"帮助/关于本程序"*/

void ShowMessageBox(gchar *message);             /*弹出内容为 message 的信息框*/
void SetColumnWidth(GtkWidget *clist, gint id);  /*设置标签宽度*/
void SelectionMade( GtkWidget *clist, gint row, gint column, GdkEventButton *event, gpointer data );
    /*选中某一行调用的回呼函数*/
void UnselectionMade(GtkWidget *clist,gint row,gint column,  GdkEventButton *event, gpointer data);
    /*取消选中某一行调用的回呼函数*/
gboolean PageSelected(GtkNotebook *notebook, gint arg, gpointer data) ;     /*选中某一页面调用的回
响函数*/
void SortByColumn(GtkCList *clist, gint column);     /*根据某一列对整个 CList 进行排序*/



/*****************************************************************************/
/*********************************创建主窗口的相关函数*****************************/
/*****************************************************************************/
GtkItemFactoryEntry menu[] = {       /*菜单*/

        {"/文件(_D)",                    "<alt>D",          0,                    0,
    "<Branch>"},
        {"/文件(_D)/新建(_N)",           "<ctrl>N",         OnFileNewActivate,    0},
```

```
        {"/文件(_D)/打开(_O)",              "<ctrl>O",          OnFileOpenActivate,      0},
        {"/文件(_D)/保存(_S)",              "<ctrl>S",          OnFileSaveActivate,      0},
        {"/文件(_D)/另存为(_S)",            "<ctrl><shift>S",   OnFileSaveasActivate,    0},
        {"/文件(_D)/更多",                  NULL,               0,                       0,
    "<Branch>"},
        {"/文件(_D)/更多/展开全部(_E)",     "<ctrl><shift>E",   OnExpandAllActivate,     0},
        {"/文件(_D)/更多/收起其他(_R)",     "<ctrl><shift>R",   OnRetractOthersActivate, 0},
        {"/文件(_D)/更多/数据排错(_D)",     "<ctrl><shift>D",   OnCheckDataActivate,     0},
        {"/文件(_D)/退出(_Q)",              "<ctrl>Q",          OnFileExitActivate,      0},

        {"/编辑(_E)",                       "<alt>E",           0,                       0,
    "<Branch>"},
        {"/编辑(_E)/添加信息(_A)",          "<ctrl>A",          OnAddInfoActivate,       0},
        {"/编辑(_E)/查找信息(_F)",          "<ctrl>F",          OnSearchInfoActivate,    0},
        {"/编辑(_E)/修改信息(_E)",          "<ctrl>E",          OnEditInfoActivate,      0},
        {"/编辑(_E)/删除信息(_D)",          "<ctrl>D",          OnDelInfoActivate,       0},

        {"/统计(_S)",                       "<alt>S",           0,                       0,
    "<Branch>"},
        {"/统计(_S)/网站排名榜(_B)",        "<ctrl>B",          OnStatiSiteActivate,     0},
        {"/统计(_S)/店铺交易情况(_C)",      "<ctrl>C",          OnStatiShopActivate,     0},
        {"/统计(_S)/地区贸易关系(_T)",      "<ctrl>T",          OnStatiDealActivate,     0},


        {"/帮助(_H)",                       "<alt>H",           0,                       0,
    "<Branch>"},
        {"/帮助(_H)/操作说明(_H)",          "<ctrl>H",          OnHelpActivate,          0},
        {"/帮助(_H)/关于本程序(_A)",        "<ctrl><shift>A",   OnAboutActivate,         0}

};

/*创建主窗口*/
void CreateWindow(void)
{
    GtkWidget *v_box;
    GtkWidget *scrolled_window;
    GtkWidget *label;
    GtkWidget *toolbar;
    GtkWidget *statusbar;
    GtkWidget *handlebox;
    GtkAccelGroup *accel_group;
    GtkItemFactory *item_factory;

    gint count;
```

```
gint dist_column[3] = {SITE_COLUMN, SHOP_COLUMN, DEAL_COLUMN};                /*每个 dist 的
列数*/
gint nmenu_items = sizeof(menu) / sizeof(menu[0]);                            /*菜单数目*/
gchar *dist_titles[][7] ={
        {"网站编号", "名称", "地址", "店铺总数"},
        {"所属网站编号", "店铺编号", "名称", "负责人", "联系地", "开户银行", "交易次数"},
        {"所属店铺编号", "交易编号", "支付类型", "金额", "日期", "客户所在地"},
};          /*每个 dist 的标签文字*/
gchar *window_titles = "购物网站信息管理系统";
gchar *label_titles[3] = {"网站信息",  "店铺信息", "交易信息"};    /*notebook 构件每个页面的标
签文字*/

/*创建工具条*/
static GnomeUIInfo gnome_toolbar[] = {
        GNOMEUIINFO_ITEM_STOCK (N_("新建"), N_("新建数据文件"), OnFileNewActivate,
                GNOME_STOCK_PIXMAP_NEW),
        GNOMEUIINFO_ITEM_STOCK (N_("打开"), N_("打开数据文件"), OnFileOpenActivate,
                GNOME_STOCK_PIXMAP_OPEN),
        GNOMEUIINFO_ITEM_STOCK (N_("保存"), N_("保存当前数据"), OnFileSaveActivate,
                GNOME_STOCK_PIXMAP_SAVE),
       GNOMEUIINFO_ITEM_STOCK (N_("另存为"), N_("保存数据到其他目录"), OnFileSaveasActivate,
                GNOME_STOCK_PIXMAP_SAVE_AS),
        GNOMEUIINFO_SEPARATOR,

        GNOMEUIINFO_ITEM_STOCK (N_("添加"), N_("添加信息"), OnAddInfoActivate,
                GNOME_STOCK_PIXMAP_ADD),
        GNOMEUIINFO_ITEM_STOCK (N_("查找"), N_("查找信息"), OnSearchInfoActivate,
                GNOME_STOCK_PIXMAP_SEARCH),
        GNOMEUIINFO_ITEM_STOCK (N_("修改"), N_("修改信息"), OnEditInfoActivate,
                GNOME_STOCK_PIXMAP_SRCHRPL),
        GNOMEUIINFO_ITEM_STOCK (N_("删除"), N_("删除信息"), OnDelInfoActivate,
                GNOME_STOCK_PIXMAP_REMOVE),

        GNOMEUIINFO_END
};

/*主窗口相关设置*/
window = gtk_window_new(GTK_WINDOW_TOPLEVEL);
gtk_widget_set_usize(window, 900, 700);
gtk_container_border_width(GTK_CONTAINER(window), 0);
gtk_window_set_title(GTK_WINDOW(window), window_titles);

/*建立回呼函数以处理信号*/
gtk_signal_connect(GTK_OBJECT(window), "destroy", GTK_SIGNAL_FUNC(gtk_widget_destroyed),
```

```
        &window);
gtk_signal_connect(GTK_OBJECT(window), "delete_event",
GTK_SIGNAL_FUNC(OnWindowDeleteEvent),   NULL);

/*建立菜单*/
accel_group = gtk_accel_group_new();
item_factory = gtk_item_factory_new(GTK_TYPE_MENU_BAR, "<blah>", accel_group);
gtk_item_factory_create_items(item_factory, nmenu_items, menu, NULL);
gtk_window_add_accel_group((GtkWindow *)GTK_OBJECT(window), accel_group);

/*将菜单包含在 v_box 里面*/
v_box = gtk_vbox_new(FALSE, 0);
gtk_container_add(GTK_CONTAINER(window), v_box);
gtk_box_pack_start(GTK_BOX(v_box), gtk_item_factory_get_widget(item_factory, "<blah>"), FALSE,
        FALSE, 0);

/*建立 handle_box 跟 toolbar*/
toolbar = gtk_toolbar_new();
gnome_app_fill_toolbar(GTK_TOOLBAR(toolbar), gnome_toolbar, NULL);
handlebox = gtk_handle_box_new();
gtk_container_add(GTK_CONTAINER(handlebox), toolbar);
gtk_box_pack_start(GTK_BOX(v_box), handlebox, FALSE, FALSE, 0);

/*建立笔记本构件*/
notebook = gtk_notebook_new();
gtk_notebook_set_tab_pos(GTK_NOTEBOOK(notebook), GTK_POS_TOP);
gtk_box_pack_start(GTK_BOX(v_box), notebook, TRUE, TRUE, 0);

for (count = 0; count < 3; count++)
{
        /*建立滚动窗口*/
        scrolled_window = gtk_scrolled_window_new(NULL, NULL);
        gtk_scrolled_window_set_policy(GTK_SCROLLED_WINDOW(scrolled_window),
                GTK_POLICY_AUTOMATIC, GTK_POLICY_ALWAYS);
        label = gtk_label_new(label_titles[count]);
        gtk_notebook_append_page(GTK_NOTEBOOK(notebook), scrolled_window, label);
        gtk_container_set_border_width(GTK_CONTAINER(scrolled_window),5);

        /*建立 GtkCList*/
        clist[count] = gtk_dist_new_with_titles(dist_column[count], dist_titles[count]);
        /*设置边框阴影*/
        gtk_dist_set_shadow_type(GTK_CLIST(dist[count]), GTK_SHADOW_OUT);
        /*设置宽度*/
        SetColumnWidth(dist[count], count);
```

```
            gtk_container_add(GTK_CONTAINER(scrolled_window), dist[count]);

            /*点击 notebook 页面的 dist 的标签即会对该列数据进行排序*/
            gtk_dist_column_titles_active(GTK_CLIST(dist[count]));
            gtk_signal_connect(GTK_OBJECT(dist[count]), "dick_column",
GTK_SIGNAL_FUNC(SortByColumn),   NULL);

           gtk_signal_connect(GTK_OBJECT(dist[count]), "select_row",GTK_SIGNAL_FUNC(SelectionMade),
                NULL) ;
            gtk_signal_connect(GTK_OBJECT(dist[count]), "unselect_row",
                GTK_SIGNAL_FUNC(UnselectionMade), NULL) ;
      }
      g_signal_connect_after(G_OBJECT(notebook), "switch-page", G_CALLBACK(PageSelected), NULL);


      /*状态栏*/
      statusbar = gtk_statusbar_new();
      gtk_box_pack_start(GTK_BOX(v_box), statusbar, FALSE, FALSE, 0);

      gtk_widget_show_all(window);
}
void SetColumnWidth(GtkWidget *dist, gint page_id)     /*设置标签宽度*/
{
      switch(page_id)
      {
            case SITE_PAGE:        /*设置网站页面*/
                  gtk_dist_set_column_width(GTK_CLIST(dist), 0, 100);
                  gtk_dist_set_column_width(GTK_CLIST(dist), 1, 200);
                  gtk_dist_set_column_width(GTK_CLIST(dist), 2, 300);
                  break;
            case SHOP_PAGE:              /*设置店铺页面*/
                  gtk_dist_set_column_width(GTK_CLIST(dist), 0, 90);
                  gtk_dist_set_column_width(GTK_CLIST(dist), 1, 70);
                  gtk_dist_set_column_width(GTK_CLIST(dist), 2, 100);
                  gtk_dist_set_column_width(GTK_CLIST(dist), 3, 100);
                  gtk_dist_set_column_width(GTK_CLIST(dist), 4, 200);
                  gtk_dist_set_column_width(GTK_CLIST(dist), 5, 100);
                  break;
            case DEAL_PAGE:       /*设置交易页面*/
                  gtk_dist_set_column_width(GTK_CLIST(dist), 0, 90);
                  gtk_dist_set_column_width(GTK_CLIST(dist), 1, 90);
                  gtk_dist_set_column_width(GTK_CLIST(dist), 2, 90);
                  gtk_dist_set_column_width(GTK_CLIST(dist), 3, 100);
                  gtk_dist_set_column_width(GTK_CLIST(dist), 4, 150);
```

```
                break;
        case STATI_PAGE:        /*设置统计页面*/
                gtk_dist_set_column_width(GTK_CLIST(dist), 0, 150);
                gtk_dist_set_column_width(GTK_CLIST(dist), 1, 150);
                gtk_dist_set_column_width(GTK_CLIST(dist), 2, 150);
                gtk_dist_set_column_width(GTK_CLIST(dist), 3, 150);
                break;
        default:
                g_message("Impossible!!");
                break;
    }
}
void SelectionMade(GtkWidget *dist,gint row,gint column,      GdkEventButton *event, gpointer data)
    /*  用户选中某一行时的回调函数*/
{
    /*获取当前页面*/
    gint current_page = gtk_notebook_get_current_page(GTK_NOTEBOOK(notebook));

    if (current_page == SITE_PAGE)
        site_selected_row = row;
    else if (current_page == SHOP_PAGE)
                shop_selected_row = row;
         else
                deal_selected_row = row;

    g_message("You have selected row %d in page %d.", row, current_page);
}
gboolean PageSelected(GtkNotebook *notebook, gint arg, gpointer data)  /*用户选中某一标签页的回调
函数*/
{
    static gint former_page = SITE_PAGE;            /*上一次选择的页面*/
    gint current_page = gtk_notebook_get_current_page(GTK_NOTEBOOK(notebook));    /*当前页面*/

    HandleSwitchPage(current_page, former_page);

    former_page = current_page;
    return FALSE;
}
/*用户点击 CList 里某一列调用的回调函数*/
void SortByColumn(GtkCList *dist, gint column)
{
    gtk_dist_set_sort_column(GTK_CLIST(dist), column);
    gtk_dist_sort(GTK_CLIST(dist));
}
```

```
void UnselectionMade(GtkWidget *dist,gint row,gint column,  GdkEventButton *event,     gpointer data)
        /*  用户取消选中某一行时的回调函数*/
{
    gint current_page = gtk_notebook_get_current_page(GTK_NOTEBOOK(notebook));

    if (current_page == SITE_PAGE)
        site_selected_row = -1;
    else if (current_page == SHOP_PAGE)
            shop_selected_row = -1;
         else
            deal_selected_row = -1;
}
/*****************************************************************************************/
/*****************************文件菜单栏对应的回调函数*****************************/
/*****************************************************************************************/
void OnFileNewActivate(void)          /*选择"文件/新建"*/
{
    FileNew();
}
void OnFileOpenActivate(void)         /*选择"文件/打开"*/
{
    FileOpen();
}
void OnFileSaveActivate(void)         /*选择"文件/保存"*/
{
    FileSave();
}
void OnFileSaveasActivate(void)       /*选择"文件/另存为"*/
{
    FileSaveas();
}
void OnFileExitActivate(void)         /*选择"文件/退出"*/
{
    FileExit();
}
void OnExpandAllActivate(void)        /*选择"文件/更多/展开全部"*/
{
    ExpandAll();
}
void OnRetractOthersActivate(void)    /*选择"文件/更多/收起其他"*/
{
    gint current_page = gtk_notebook_get_current_page(GTK_NOTEBOOK(notebook));
```

```
    RetractOthers(current_page);
}
void OnCheckDataActivate(void)          /*选择"文件/更多/数据排错"*/
{
    CheckData();
}

/*****************************************************************************************/
/*****************************编辑菜单栏对应的回调函数*********************************/
/*****************************************************************************************/
gint GetSelectedRow(gint current_page)              /*返回相应页面所选中的行数*/
{
    switch (current_page)
    {
        case SITE_PAGE:
            return site_selected_row;
        case SHOP_PAGE:
            return shop_selected_row;
        case DEAL_PAGE:
            return deal_selected_row;
        default:
            return -1;
    }
}
void OnAddInfoActivate(void)                    /*选择"编辑/添加信息"*/
{
    gint current_page = gtk_notebook_get_current_page(GTK_NOTEBOOK(notebook));
    if (current_page != STATI_PAGE)
        AddInfo(current_page);
}

void OnSearchInfoActivate(void)                 /*选择"编辑/查找信息"*/
{
    gint current_page = gtk_notebook_get_current_page(GTK_NOTEBOOK(notebook));
    if (current_page != STATI_PAGE)
        SearchInfo(current_page);
}

void OnEditInfoActivate(void)                   /*选择"编辑/修改信息"*/
{
    gint current_page = gtk_notebook_get_current_page(GTK_NOTEBOOK(notebook));
    gint selected_row = GetSelectedRow(current_page);
    if (current_page != STATI_PAGE)
    {
```

```
        if (selected_row != -1)
            EditInfo(current_page,selected_row);
        else
            ShowMessageBox("未选中任何项!");
    }

}

void OnDelInfoActivate(void)                    /*选择"编辑/删除信息"*/
{
    gint current_page = gtk_notebook_get_current_page(GTK_NOTEBOOK(notebook));
    gint selected_row = GetSelectedRow(current_page);
    if (current_page != STATI_PAGE)
    {
        if (selected_row != -1)
            DelInfo(current_page, selected_row);
        else
            ShowMessageBox("未选中任何项!");
    }

}

/*****************************************************************************************/
/*********************************统计菜单栏需要调用的函数*****************************/
/*****************************************************************************************/
void OnStatiSiteActivate(void)          /*选择"统计/网站排行榜"*/
{
    StatiInfo(0);
}
void OnStatiShopActivate(void)          /*选择"统计/店铺交易信息"*/
{
    StatiInfo(1);
}

void OnStatiDealActivate(void)          /*选择"统计/地区贸易情况"*/
{
    StatiInfo(2);
}
void CreateStatiNotebook(gint id)       /*创建“统计结果”页面*/
{
    g_message("CreateStatiNotebook");
    GtkWidget *scrolled_window;
    GtkWidget *label;
    gchar *dist_titles[3][5] = {
```

```
        {"名次", "网站编号","网站名称", "月份", "交易次数"},
        {"店铺编号", "店铺名称","年份", "交易次数", "交易金额"},
        {"地区名称", "月份","资金流入", "资金流出", "贸易情况"}
};
gint dist_column = 5;
static gboolean first = TRUE;

if (first)     /*如果是第一次建立“统计结果”页面*/
{
        /*建立滚动窗口*/
        scrolled_window = gtk_scrolled_window_new(NULL, NULL);
        gtk_scrolled_window_set_policy(GTK_SCROLLED_WINDOW(scrolled_window),
GTK_POLICY_AUTOMATIC, GTK_POLICY_ALWAYS);
        label = gtk_label_new("统计结果");
        gtk_notebook_append_page(GTK_NOTEBOOK(notebook),scrolled_window, label);
        gtk_container_set_border_width(GTK_CONTAINER(scrolled_window),5);

        /*建立 GtkCList*/
        clist[3] = gtk_dist_new_with_titles(dist_column, dist_titles[id]);
        /*设置边框阴影*/
        gtk_dist_set_shadow_type(GTK_CLIST(dist[3]), GTK_SHADOW_OUT);
        /*设置宽度*/
        SetColumnWidth(dist[3], STATI_PAGE);
        gtk_container_add(GTK_CONTAINER(scrolled_window), dist[3]);

        /*为一些信息建立回呼函数*/
        gtk_dist_column_titles_active(GTK_CLIST(dist[3]));
        gtk_signal_connect(GTK_OBJECT(dist[3]), "dick_column", GTK_SIGNAL_FUNC(SortByColumn),
                NULL);

        gtk_signal_connect(GTK_OBJECT(dist[3]), "select_row", GTK_SIGNAL_FUNC(SelectionMade),
                NULL) ;
       gtk_signal_connect(GTK_OBJECT(dist[3]),"unselect_row",GTK_SIGNAL_FUNC(UnselectionMade),
                NULL) ;

        gtk_widget_show_all(notebook);
        gtk_notebook_set_current_page(GTK_NOTEBOOK(notebook), STATI_PAGE);
        first = FALSE;
}
else  /*前面已经建立了“统计结果”页面*/
{
        gtk_notebook_remove_page(GTK_NOTEBOOK(notebook), STATI_PAGE);
        first = TRUE;
        CreateStatiNotebook(id);
```

```
    }
}

/*****************************************************************************/
/*****************************帮助菜单栏需要调用的函数*****************************/
/*****************************************************************************/
void OnHelpActivate(void)          /*选择"帮助/操作说明"*/
{
    gchar *message = "1、第一次运行程序请先依次添加网站、店铺、交易信息。\n\n\
2、编辑菜单栏及工具栏对不同页面会执行相应信息的相关操作。\n\n\
3、菜单\"/文件/更多/展开全部\"即将所有信息输出；\"/文件/更多/收起其他\"即只查看某网站、店铺对应的下级信息，选择后即为\"收起其他\"模式。\n\n\
4、在\"收起其他模式下\",在网站页面下选择某一网站后再切换到店铺页面即可查看，同理可查看某个店铺的交易信息。\
如欲查看某网站的所有交易信息，请先在网站信息页面中选择该网站，再切换到交易信息页面即可查看\n\n\
5、菜单\"/文件/更多/数据排错\"会检查并修正当前数据中可能存在的一些错误(主要检查编号以及店铺个数、交易次数)。\
注意：此功能将重新对店铺及交易进行编号。(删除操作也会执行类似操作。)\n\n\
6、查找只能找到符合的第一项，如要查看后面几项，可先点击对应标签进行排序。\n\n\
7、程序有一不足：点击标签进行排序后如欲修改或删除信息须重新选择对象，否则可能导致结果与自己预想的不一样。\n\n\
7、工具栏可随意拖动。";
    ShowMessageBox(message);
}
void OnAboutActivate(void)         /*选择"帮助/关于本程序"*/
{
    GtkWidget *about;
    const gchar *title = "购物网站信息管理系统";
    const gchar *version = "1.0";
    const gchar *copyright = "Copyright © 2010 Huang Zhigeng";
    const char *authors[] = {"Huang Zhigeng", "\n", "参考资料:    《Linux 应用程序开发指南》", "
    《GTK 编程范例》",
     "          《开发 Liunx 应用》",NULL};

    about = gnome_about_new(title, version, copyright, NULL, authors, NULL, NULL, NULL);
    gtk_widget_show(about);
}

/*****************************************************************************/
/*************************************辅助函数*************************************/
/*****************************************************************************/
void ShowMessageBox(gchar *message)            /*弹出内容为 message 的消息框*/
{
```

```
    GtkWidget *message_box;

    message_box = gnome_message_box_new(message, GNOME_MESSAGE_BOX_INFO,
GNOME_STOCK_BUTTON_OK, NULL);
    gnome_dialog_set_parent((GnomeDialog *)message_box, (GtkWindow *)window);
    gtk_widget_show(message_box);
}
```

iii. create_window.h

```
#ifndef __CREATE_WINDOW_H__
#define __CREATE_WINDOW_H__

void CreateWindow(void);                    /*创建主窗口*/
void CreateStatiNotebook(gint id);          /*创建"统计结果"页面*/
void ShowMessageBox(gchar *message);        /*弹出内容为message的消息框*/

#endif
```

iv. get_user_data.c

```
/*
* 前后台交互模块
*
* 主要功能：绘制对话框.同时为后台的链表操作获取必要的信息,例如打开、
*          保存文件时的文件完整路径,修改后的新数据,查找所根据的key等
*/

#include <gtk/gtk.h>
#include <libgnomeui/gnome-dialog.h>
#include <libgnomeui/gnome-messagebox.h>
#include <libgnomeui/gnome-file-entry.h>
#include <string.h>
#include <stdlib.h>
#include "create_window.h"
#include "handle_data.h"

#define SITE_PAGE 0
#define SHOP_PAGE 1
#define DEAL_PAGE 2


/*下面6个变量定义于create_window.c*/
```

```
extern GtkWidget *window;
extern GtkWidget *notebook;
extern GtkWidget *clist[4];
extern gint site_selected_row;              /*存储网站页面选中的行数*/
extern gint shop_selected_row;              /*存储店铺页面选中的行数*/
extern gint deal_selected_row;              /*存储交易页面选中的行数*/

extern gboolean expand_all_info;            /*定义于handle_data.c,标识是否展开所有信息*/

gchar filename[50] = "";                /*存储文件完整路径*/
static gboolean file_modifyed = FALSE; /*标识数据是否改动*/

void FileNew(void);                     /*选择"文件/新建"*/
void FileOpen(void);                    /*选择"文件/打开"*/
void SaveFile(void);                    /*创建保存对话框*/

/*****************************************************************************/
/******************关闭或打开新的文件时检查原数据是否改动的相关函数******************/
/*****************************************************************************/
GtkWidget *yesno_dialog;                            /*提示是否保存的对话框*/
GtkWidget *ConfirmDialog(gpointer data);        /*创建提示是否保存的对话框*/

void OnYesClicked(GtkButton *button, gpointer data)       /*按下YES*/
{
  gtk_widget_destroy(yesno_dialog);
  if (!strcmp(filename, ""))       /*如果文件路径为空*/
       SaveFile();
  else
       RealSave(filename);         /*执行真正保存的操作，位于handle_data.c*/

  file_modifyed = FALSE;

  if (!strcmp(data, "new"))
       FileNew();         /*如果之前是执行“文件/新建”操作才调用此函数*/
  else if (!strcmp(data, "open"))
       FileOpen();        /*如果之前是执行“文件/打开”操作才调用此函数*/
}
void OnNoClicked(GtkButton *button, gpointer data)        /*按下NO*/
{
  gtk_widget_destroy(yesno_dialog);


  if (!strcmp(data, "exit"))
       gtk_main_quit();
```

```
    file_modifyed = FALSE;

    if (!strcmp(data, "new"))
        FileNew();
    else if (!strcmp(data, "open"))
        FileOpen();
}
void OnCancelClicked(GtkButton *button)                    /*按下CANCEL*/
{
    gtk_widget_destroy(yesno_dialog);
}

GtkWidget *ConfirmDialog(gpointer data)                    /*创建提示是否保存的对话框
*/
{
    GtkWidget *dialog;
    GtkWidget *vbox;
    GtkWidget *label;
    GtkWidget *image;
    GtkWidget *hbox;
    GtkWidget *bbox;
    GtkWidget *button;
    GtkWidget *sep;
    gchar *title = "数据已被修改。是否现在保存？";

    /*对话框窗口*/
    dialog = gtk_window_new(GTK_WINDOW_TOPLEVEL);
    g_signal_connect(GTK_OBJECT(dialog), "delete_event", G_CALLBACK(gtk_widget_destroy),
dialog);
    gtk_window_set_modal(GTK_WINDOW(dialog), TRUE);
    gtk_window_set_title(GTK_WINDOW(dialog), "警告");

    vbox = gtk_vbox_new(FALSE, 0);
    gtk_container_add(GTK_CONTAINER(dialog), vbox);
    hbox = gtk_hbox_new(FALSE, 0);
    gtk_box_pack_start(GTK_BOX(vbox), hbox, FALSE, FALSE, 5);

    /*创建按钮*/
    label = gtk_label_new(title);
    image = gtk_image_new_from_stock(GTK_STOCK_DIALOG_WARNING, GTK_ICON_SIZE_DIALOG);
    gtk_box_pack_start(GTK_BOX(hbox), image, FALSE, FALSE, 5);
    gtk_box_pack_start(GTK_BOX(hbox), label, FALSE, FALSE, 5);
```

```
  /*创建分隔符*/
  sep = gtk_hseparator_new();
  gtk_box_pack_start(GTK_BOX(vbox), sep, FALSE, FALSE, 5);
  bbox = gtk_hbutton_box_new();
  gtk_box_pack_start(GTK_BOX(vbox), bbox, FALSE, FALSE, 5);

  /*创建按钮*/
  button = gtk_button_new_from_stock(GTK_STOCK_YES);
  g_signal_connect(GTK_OBJECT(button), "clicked", G_CALLBACK(OnYesClicked), data);
  gtk_box_pack_start(GTK_BOX(bbox), button, FALSE, FALSE, 5);
  button = gtk_button_new_from_stock(GTK_STOCK_NO);
  g_signal_connect(GTK_OBJECT(button), "clicked", G_CALLBACK(OnNoClicked), data);
  gtk_box_pack_start(GTK_BOX(bbox), button, FALSE, FALSE, 5);
  button = gtk_button_new_from_stock(GTK_STOCK_CANCEL);
  g_signal_connect(GTK_OBJECT(button), "clicked", G_CALLBACK(OnCancelClicked), NULL);

  gtk_box_pack_start(GTK_BOX(bbox), button, FALSE, FALSE, 5);
  gtk_widget_show_all(dialog);
  return dialog;
}

gboolean OnWindowDeleteEvent(void)     /*退出前调用的函数*/
{
  if (file_modifyed == TRUE)
  {
       yesno_dialog = ConfirmDialog("exit");
       gtk_widget_show(yesno_dialog);
       return TRUE;
  }
  else
  {
       gtk_main_quit();       /*退出整个程序*/
       return FALSE;
  }
}



/*****************************************************************************************/
/*********************************文件菜单栏需要调用的函数*****************************/
/*****************************************************************************************/
void InitCList(void);              /*除去统计页面并将当前页面切换到"网站"页面*/
void OnDataSaved(void);            /*数据已经保存*/

void StoreFilename(GtkFileSelection *selector, gpointer user_data);          /*保存文件路
```

径*/

```
void FileNew(void)                /*选择"文件/新建"*/
{
  gint index;
  if (file_modifyed)
       yesno_dialog = ConfirmDialog("new");
  else
  {
       for (index = 0; index < 3; index++)
            gtk_clist_clear(GTK_CLIST(clist[index]));

       InitCList();
       strcpy(filename, "");
       ClearList();  /*将链表置为空*/
  }
}
void FileOpen(void)               /*选择"文件/打开"*/
{
  GtkWidget *open_dialog = NULL;

  if (file_modifyed)
       yesno_dialog = ConfirmDialog("open");
  else
  {
       open_dialog = gtk_file_selection_new("请选择一个文件:");
       g_signal_connect(GTK_OBJECT(GTK_FILE_SELECTION(open_dialog)->ok_button), "clicked",
            G_CALLBACK(StoreFilename), open_dialog);
       g_signal_connect_swapped(GTK_OBJECT(GTK_FILE_SELECTION(open_dialog)->ok_button),
            "clicked", G_CALLBACK(RealOpen), filename);
       g_signal_connect_swapped(GTK_OBJECT(GTK_FILE_SELECTION(open_dialog)->ok_button),
            "clicked", G_CALLBACK(InitCList), NULL);
       g_signal_connect_swapped(GTK_OBJECT(GTK_FILE_SELECTION(open_dialog)->ok_button),
            "clicked", G_CALLBACK(gtk_widget_destroy), open_dialog);

  g_signal_connect_swapped(GTK_OBJECT(GTK_FILE_SELECTION(open_dialog)->cancel_button),
            "clicked",G_CALLBACK(gtk_widget_destroy),(gpointer)open_dialog);

       gtk_widget_show(open_dialog);
  }
}
void FileSave(void)               /*选择"文件/保存"*/
{
  if (!strcmp(filename, ""))
```

```
        SaveFile();
    else
    {
        RealSave(filename);
        OnDataSaved();
    }
}
void FileSaveas(void)               /*选择"文件/另存为"*/
{
    SaveFile();
}
void ExpandAll(void)                /*选择"文件/更多/展开全部"*/
{
    ChangeShowMode(TRUE);           /*改变展示方式，TRUE"展开全部“模式，位于handle_data.c*/
    RedrawShopCList(0);
    RedrawDealCList(0, 0);
}
void RetractOthers(gint current_page)       /*选择"文件/更多/收起其他"*/
{
    gchar *str_site_id, *str_shop_id;
    gint site_id, shop_id;
    ShopInfo *shop_current;

    switch (current_page)
    {
        case SITE_PAGE:
            ChangeShowMode(FALSE); /*改变展示方式，FALSE为"收起其他模式"*/
            break;
        case SHOP_PAGE:
            if (shop_selected_row >= 0)
            {
                ChangeShowMode(FALSE);
                gtk_clist_get_text(GTK_CLIST(clist[1]), shop_selected_row, 0,
                    &str_site_id);
                site_id = atoi(str_site_id);
                RedrawShopCList(site_id);
            }
            else
                ShowMessageBox("请先选择某个商家！");

            break;
        case DEAL_PAGE:
            if (deal_selected_row >= 0)
            {
```

```
                ChangeShowMode(FALSE);
                gtk_clist_get_text(GTK_CLIST(clist[2]), deal_selected_row, 0,
                    &str_shop_id);
                shop_id = atoi(str_shop_id);
                shop_current = GetShop(shop_id);
                site_id = shop_current -> site_id;

                RedrawDealCList(site_id, shop_id);
                RedrawShopCList(site_id);
            }
            else
                ShowMessageBox("请先选择某次交易！");

            break;
        default:
            ChangeShowMode(FALSE);
            break;
    }
}
void CheckData(void)          /*选择"文件/更多/数据排错"*/
{
    RealCheckData();
}
gboolean FileExit(void)           /*选择"文件/退出"*/
{
    if (file_modifyed == TRUE)
    {
        yesno_dialog = ConfirmDialog("exit");
        gtk_widget_show(yesno_dialog);
        return TRUE;
    }
    else
    {
        gtk_main_quit();        /*退出整个程序*/
        return FALSE;
    }
}
void OnDataChanged(void)      /*数据发生改变*/
{
    file_modifyed = TRUE;
}
void OnDataSaved(void)            /*数据已经保存*/
{
    file_modifyed = FALSE;
```

```
}
void StoreFilename(GtkFileSelection *selector, gpointer user_data) /*获取文件名*/
{
  strcpy(filename, gtk_file_selection_get_filename(GTK_FILE_SELECTION(user_data)));
}
void SaveFile(void)                /*保存文件对话框*/
{
  GtkWidget *save_dialog = NULL;

  save_dialog = gtk_file_selection_new("请输入文件名:");
  g_signal_connect(GTK_OBJECT(GTK_FILE_SELECTION(save_dialog)->ok_button), "clicked",
       G_CALLBACK(StoreFilename), save_dialog);
  g_message("SaveFile:%s", filename);
  g_signal_connect_swapped(GTK_OBJECT(GTK_FILE_SELECTION(save_dialog)->ok_button),
       "clicked", G_CALLBACK(RealSave), filename);
  g_signal_connect_swapped(GTK_OBJECT(GTK_FILE_SELECTION(save_dialog)->ok_button),
       "clicked", G_CALLBACK(OnDataSaved), NULL);
  g_signal_connect_swapped(GTK_OBJECT(GTK_FILE_SELECTION(save_dialog)->ok_button),
       "clicked", G_CALLBACK(gtk_widget_destroy), save_dialog);
  g_signal_connect_swapped(GTK_OBJECT(GTK_FILE_SELECTION(save_dialog)->cancel_button),
       "clicked", G_CALLBACK(gtk_widget_destroy), (gpointer)save_dialog);

  gtk_widget_show(save_dialog);
}
void InitCList(void)/*除去统计页面并将当前页面切换到"网站"页面*/
{
  g_message("InitCList");
  gtk_notebook_remove_page(GTK_NOTEBOOK(notebook), 3);
  gtk_notebook_set_current_page(GTK_NOTEBOOK(notebook), 0);
}

/*****************************************************************************************/
/*********************************编辑菜单栏需要调用的函数*****************************/
/*****************************************************************************************/
#define SITE_COLUMN 4
#define SHOP_COLUMN 7
#define DEAL_COLUMN 6

#define ADD_SITE 0
#define EDIT_SITE 1
#define ADD_SHOP 0
#define EDIT_SHOP 1
#define ADD_DEAL 0
#define EDIT_DEAL 1
```

```
#define BUTTON_ADD 0
#define BUTTON_OK 1
#define BUTTON_CANCEL 2

/*以下两个变量定义于handle_data.c*/
extern gint site_amount;      /*网站总数*/
extern gint shop_amount;      /*店铺总数*/

/*存储选中行的数据*/
gchar site_data[4][50] = {"", "", "", ""};
gchar shop_data[7][20] = {"", "", "", "", "", ""};
gchar deal_data[6][20] = {"", "", "", "", "", ""};

/*上面三个数组对应的指针*/
gchar *site_data_pointer[4];
gchar *shop_data_pointer[7];
gchar *deal_data_pointer[6];

gchar search_keys[3][50] = {"", "", ""};              /*存储搜索功能的关键字*/

void CreateSiteDialog(gint id, gint selected_row);    /*创建添加或编辑网站对话框*/
void CreateShopDialog(gint id, gint selected_row);    /*创建添加或编辑店铺对话框*/
void CreateDealDialog(gint id, gint selected_row);    /*创建添加或编辑交易对话框*/
gboolean SearchDialog(gint current_page);             /*创建查找对话框*/
gboolean DelSiteDialog(void);                         /*创建删除网站对话框*/
gboolean DelShopDialog(void);                         /*创建删除店铺对话框*/
gboolean DelDealDialog(void);                         /*创建删除网站对话框*/
void CopyData(gint current_page);                     /*拷贝所需的一些系统自动生成的数据
(如商家编号，店铺总数)，用于编辑信息功能*/
gboolean SelectSearchedKey(gint current_page);        /*选取要根据哪些关键字进行搜索*/
gboolean SelectSearchedKind(gint current_page, gint search_key_id, gchar *search_key);
  /*选取搜索哪类信息*/

void AddInfo(gint current_page)                       /*选择"编辑/添加信息"*/
{
  g_message("AddInfo: current_page=%d", current_page);

  switch (current_page)  /*根据当前页面调用不同函数*/
  {
       case SITE_PAGE:
            CreateSiteDialog(ADD_SITE, 0);
            break;
       case SHOP_PAGE:
```

```
            CreateShopDialog(ADD_SHOP, 0);
            break;
        case DEAL_PAGE:
            CreateDealDialog(ADD_DEAL, 0);
            break;
        default:
            break;
    }

}
void SearchInfo(gint current_page)                    /*选择"编辑/查找信息"*/
{
    g_message("SearchInfo: current_page=%d", current_page);

    if (SearchDialog(current_page))
        SelectSearchedKey(current_page);


}
void EditInfo(gint current_page, gint selected_row)  /*选择"编辑/编辑信息"*/
{
    g_message("EditInfo: current_page=%d, selected_row=%d", current_page, selected_row);
    gint index;

    switch (current_page)       /*根据当前页面调用不同函数*/
    {
        case SITE_PAGE:
            /*获取选中行的数据*/
            for (index = 0; index < SITE_COLUMN; index++)
                gtk_clist_get_text(GTK_CLIST(clist[0]), selected_row, index,
                    &site_data_pointer[index]);
            CopyData(SITE_PAGE);
            CreateSiteDialog(EDIT_SITE, selected_row);
            break;
        case SHOP_PAGE:
            for (index = 0; index < SHOP_COLUMN; index++)
                gtk_clist_get_text(GTK_CLIST(clist[1]), selected_row, index,
                    &shop_data_pointer[index]);
            CopyData(SHOP_PAGE);
            CreateShopDialog(EDIT_SHOP, selected_row);
            break;
        case DEAL_PAGE:
            for (index = 0; index < DEAL_COLUMN; index++)
                gtk_clist_get_text(GTK_CLIST(clist[2]), selected_row, index,
```

```
                    &deal_data_pointer[index]);
            CopyData(DEAL_PAGE);
            CreateDealDialog(EDIT_DEAL, selected_row);
            break;
    }
}
void DelInfo(gint current_page, gint selected_row)   /*选择"编辑/删除信息"*/
{
    g_message("DelInfo: current_page=%d, selected_row=%d", current_page, selected_row);
    gint index;

    switch (current_page)  /*根据当前页面调用不同函数*/
    {
        case SITE_PAGE:
            for (index = 0; index < SITE_COLUMN; index++)
                gtk_clist_get_text(GTK_CLIST(clist[0]), selected_row, index,
                    &site_data_pointer[index]);

            strcpy(site_data[0], site_data_pointer[0]);

            if (DelSiteDialog())
                DelSite(selected_row, site_data[0]);
            break;
        case SHOP_PAGE:
            for (index = 0; index < SHOP_COLUMN; index++)
                gtk_clist_get_text(GTK_CLIST(clist[1]), selected_row, index,
                    &shop_data_pointer[index]);

            strcpy(shop_data[0], shop_data_pointer[0]);
            strcpy(shop_data[1], shop_data_pointer[1]);

            if (DelShopDialog())
                DelShop(selected_row, shop_data[0], shop_data[1], TRUE);
            break;
        case DEAL_PAGE:
            for (index = 0; index < DEAL_COLUMN; index++)
                gtk_clist_get_text(GTK_CLIST(clist[2]), selected_row, index,
                    &deal_data_pointer[index]);

            strcpy(deal_data[0], deal_data_pointer[0]);
            strcpy(deal_data[1], deal_data_pointer[1]);

            if (DelDealDialog())
                DelDeal(selected_row, deal_data[0], deal_data[1], TRUE);
```

```
            break;
    }
}

void CreateSiteDialog(gint id, gint selected_row)    /*创建添加或编辑网站对话框*/
{
    GtkWidget *site_dialog = NULL;
    GtkWidget *entry[2];
    GtkWidget *label;
    GtkWidget *table;
    GtkWidget *warning_label = NULL;
    gchar *dialog_title[2] = {"添加网站", "编辑网站"};
    gint index;
    gint reply;
    gboolean continue_add = TRUE;

    if (site_dialog != NULL)
    {
        g_message("site_dialog!=NULL");
        gdk_window_show(site_dialog->window);
        gdk_window_raise(site_dialog->window);
    }
    else
    {
        if (id == ADD_SITE)
            site_dialog = gnome_dialog_new(dialog_title[id], "添加", "确定","取消", NULL);
        else
            site_dialog = gnome_dialog_new(dialog_title[id], "确定","取消", NULL);
        table = gtk_table_new(3, 2, FALSE);
        gtk_box_pack_start(GTK_BOX(GNOME_DIALOG(site_dialog)->vbox),table, TRUE, TRUE, 0);

        /*以下均在绘制对话框*/
        label = gtk_label_new("网站名称 ");
        gtk_table_attach(GTK_TABLE(table), label, 0, 1, 0, 1, GTK_EXPAND | GTK_FILL, GTK_EXPAND
            | GTK_FILL, 5, 5);
        entry[0] = gtk_entry_new_with_max_length(20);
        gtk_table_attach(GTK_TABLE(table), entry[0], 1, 2, 0, 1, GTK_EXPAND | GTK_FILL,
            GTK_EXPAND | GTK_FILL, 5, 5);

        label = gtk_label_new("网站地址 ");
        gtk_table_attach(GTK_TABLE(table), label, 0, 1, 1, 2, GTK_EXPAND | GTK_FILL, GTK_EXPAND
            | GTK_FILL, 5, 5);
        entry[1] = gtk_entry_new_with_max_length(50);
        gtk_table_attach(GTK_TABLE(table), entry[1], 1, 2, 1, 2, GTK_EXPAND | GTK_FILL,
```

```
        GTK_EXPAND | GTK_FILL, 5, 5);

    if (id == EDIT_SITE)    /*如果是编辑对话框，则给entry赋值*/
    {
        gtk_entry_set_text(GTK_ENTRY(entry[0]), site_data_pointer[1]);
        gtk_entry_set_text(GTK_ENTRY(entry[1]), site_data_pointer[2]);
    }

    /*修饰对话框，如设置按下ENTER键的默认操作等*/
    gnome_dialog_set_parent(GNOME_DIALOG(site_dialog), GTK_WINDOW(window));

    for (index = 0; index < 2; index++)
        gnome_dialog_editable_enters(GNOME_DIALOG(site_dialog),
            GTK_EDITABLE(entry[index]));
    if (id == ADD_SITE)
        gnome_dialog_set_default(GNOME_DIALOG(site_dialog), BUTTON_OK);
    else
        gnome_dialog_set_default(GNOME_DIALOG(site_dialog), BUTTON_OK - 1);

    gtk_widget_show_all(site_dialog);
}

while (continue_add)
{
    reply = gnome_dialog_run(GNOME_DIALOG(site_dialog));
    if (id == EDIT_SITE && reply >= 0)
        reply++;

    if (reply == BUTTON_CANCEL)      /*按下取消键*/
    {
        gtk_widget_destroy(site_dialog);
        continue_add = FALSE;
    }
    else if (reply == BUTTON_OK)     /*按下确定键*/
         {
            strcpy((gchar *)site_data[1], (gchar *)
                gtk_entry_get_text(GTK_ENTRY(entry[0])));
            strcpy((gchar *)site_data[2], (gchar *)
                gtk_entry_get_text(GTK_ENTRY(entry[1])));

            if (strcmp(site_data[1], "") && strcmp(site_data[2], ""))
            /*如果输入信息是完整的*/
            {
                if (id == ADD_SITE)
```

```
                AddSite(site_data);
            else
                EditSite(selected_row, site_data);
            OnDataChanged();
            continue_add = FALSE;
            gtk_widget_destroy(site_dialog);
        }
        else          /*如果输入信息不完整*/
        {
            if (warning_label == NULL)
            {
                warning_label = gtk_label_new("请输入完整信息！");
                gtk_table_attach(GTK_TABLE(table), warning_label, 0, 2, 2, 3,
                    GTK_EXPAND | GTK_FILL, GTK_EXPAND | GTK_FILL, 5, 5);
                gtk_widget_show(warning_label);
            }
            continue_add = TRUE;
        }

    }
    else if (reply == BUTTON_ADD)   /*按下添加键*/
    {
        strcpy((gchar *)site_data[1], (gchar *)
            gtk_entry_get_text(GTK_ENTRY(entry[0])));
        strcpy((gchar *)site_data[2], (gchar *)
            gtk_entry_get_text(GTK_ENTRY(entry[1])));
        g_message("网站名称: %s", (gchar *)site_data[1]);
        g_message("网站地址: %s", (gchar *)site_data[2]);

        if (strcmp(site_data[1], "") && strcmp(site_data[2], ""))
        {
            AddSite(site_data);
            OnDataChanged();
            if (warning_label != NULL)
            {
                gtk_widget_destroy(warning_label);
                warning_label = NULL;
            }
            gtk_entry_set_text(GTK_ENTRY(entry[0]), "");
            gtk_entry_set_text(GTK_ENTRY(entry[1]), "");
        }
        else
        {
            if (warning_label == NULL)
```

```
                    {
                        warning_label = gtk_label_new("请输入完整信息！");
                        gtk_table_attach(GTK_TABLE(table), warning_label, 0, 2, 2, 3,
                            GTK_EXPAND | GTK_FILL, GTK_EXPAND | GTK_FILL, 5, 5);
                        gtk_widget_show(warning_label);
                    }
                }

                continue_add = TRUE;
            }
            else
                continue_add = FALSE;
        if (id == EDIT_SITE)
            continue_add = FALSE;
    }
  }
  void CreateShopDialog(gint id, gint selected_row)    /*创建添加或编辑店铺对话框,类似于
CreateSiteDialog，*/
  {
    g_message("CreateShopDialog");
    GtkWidget *shop_dialog = NULL;
    GtkWidget *entry[5];
    GtkWidget *label;
    GtkWidget *table;
    GtkWidget *warning_label = NULL;
    gchar *dialog_title[2] = {"添加店铺", "编辑店铺"};
    gint index, site_id, reply;
    gboolean continue_add = TRUE;
    gchar former_site_id[10];

    if (shop_dialog != NULL)
    {
        g_message("shop_dialog!=NULL");
        gdk_window_show(shop_dialog->window);
        gdk_window_raise(shop_dialog->window);
    }
    else
    {
        /*绘制对话框*/
        if (id == ADD_SHOP)
            shop_dialog = gnome_dialog_new(dialog_title[id], "添加", "确定","取消", NULL);
        else
            shop_dialog = gnome_dialog_new(dialog_title[id], "确定","取消", NULL);
        table = gtk_table_new(5, 2, FALSE);
```

```
gtk_box_pack_start(GTK_BOX(GNOME_DIALOG(shop_dialog)->vbox),table, TRUE, TRUE, 0);

label = gtk_label_new("所属网站编号 ");
gtk_table_attach(GTK_TABLE(table), label, 0, 1, 0, 1, GTK_EXPAND | GTK_FILL, GTK_EXPAND
    | GTK_FILL, 5, 5);
entry[0] = gtk_entry_new_with_max_length(10);
gtk_table_attach(GTK_TABLE(table), entry[0], 1, 2, 0, 1, GTK_EXPAND | GTK_FILL,
    GTK_EXPAND | GTK_FILL, 5, 5);

label = gtk_label_new("名称 ");
gtk_table_attach(GTK_TABLE(table), label, 0, 1, 1, 2, GTK_EXPAND | GTK_FILL, GTK_EXPAND
    | GTK_FILL, 5, 5);
entry[1] = gtk_entry_new_with_max_length(20);
gtk_table_attach(GTK_TABLE(table), entry[1], 1, 2, 1, 2, GTK_EXPAND | GTK_FILL,
    GTK_EXPAND | GTK_FILL, 5, 5);

label = gtk_label_new("负责人 ");
gtk_table_attach(GTK_TABLE(table), label, 0, 1, 2, 3, GTK_EXPAND | GTK_FILL, GTK_EXPAND
    | GTK_FILL, 5, 5);
entry[2] = gtk_entry_new_with_max_length(10);
gtk_table_attach(GTK_TABLE(table), entry[2], 1, 2, 2, 3, GTK_EXPAND | GTK_FILL,
    GTK_EXPAND | GTK_FILL, 5, 5);

label = gtk_label_new("联系地 ");
gtk_table_attach(GTK_TABLE(table), label, 0, 1, 3, 4, GTK_EXPAND | GTK_FILL, GTK_EXPAND
    | GTK_FILL, 5, 5);
entry[3] = gtk_entry_new_with_max_length(20);
gtk_table_attach(GTK_TABLE(table), entry[3], 1, 2, 3, 4, GTK_EXPAND | GTK_FILL,
    GTK_EXPAND | GTK_FILL, 5, 5);

label = gtk_label_new("开户银行 ");
gtk_table_attach(GTK_TABLE(table), label, 0, 1, 4, 5, GTK_EXPAND | GTK_FILL, GTK_EXPAND
    | GTK_FILL, 5, 5);
entry[4] = gtk_entry_new_with_max_length(20);
gtk_table_attach(GTK_TABLE(table), entry[4], 1, 2, 4, 5, GTK_EXPAND | GTK_FILL,
    GTK_EXPAND | GTK_FILL, 5, 5);

if (id == EDIT_SHOP)
{
    gtk_entry_set_text(GTK_ENTRY(entry[0]), shop_data_pointer[0]);
    for (index = 1; index < 5; index++)
        gtk_entry_set_text(GTK_ENTRY(entry[index]), shop_data_pointer[index+1]);

}
```

```
    gnome_dialog_set_parent(GNOME_DIALOG(shop_dialog), GTK_WINDOW(window));

    for (index = 0; index < 5; index++)
        gnome_dialog_editable_enters(GNOME_DIALOG(shop_dialog),
            GTK_EDITABLE(entry[index]));
    if (id == ADD_SHOP)
        gnome_dialog_set_default(GNOME_DIALOG(shop_dialog), BUTTON_OK);
    else
        gnome_dialog_set_default(GNOME_DIALOG(shop_dialog), BUTTON_OK - 1);

    gtk_widget_show_all(shop_dialog);
}

while (continue_add)
{
    reply = gnome_dialog_run(GNOME_DIALOG(shop_dialog));
    if (id == EDIT_SHOP && reply >= 0)
        reply++;

    if (reply == BUTTON_CANCEL)          /*按下取消键*/
    {
        gtk_widget_destroy(shop_dialog);
        continue_add = FALSE;
    }
    else if (reply == BUTTON_OK)         /*按下确定键*/
         {
            strcpy(shop_data[0], (gchar *)gtk_entry_get_text(GTK_ENTRY(entry[0])));
            for (index = 2; index < 6; index++)
                strcpy(shop_data[index], (gchar *)
                    gtk_entry_get_text(GTK_ENTRY(entry[index-1])));

            site_id = atoi(shop_data[0]);
            if (site_id > site_amount || site_id <= 0)
            {
                if (warning_label == NULL)
                {
                    warning_label = gtk_label_new("无此网站,请检查所属网站编号!");
                    gtk_box_pack_start(GTK_BOX(GNOME_DIALOG(shop_dialog)->vbox),
                        warning_label, TRUE, TRUE, 0);
                    gtk_widget_show(warning_label);
                }
                continue_add = TRUE;
            }
            else
```

```
        {
            if (id == ADD_SHOP)
                AddShop(shop_data);
            else
            {
                g_message("shop_data_pointer[0] = %s", shop_data_pointer[0]);

                strcpy(former_site_id, shop_data_pointer[0]);
                EditShop(selected_row, former_site_id, shop_data);

            }
            OnDataChanged();
            gtk_widget_destroy(shop_dialog);
            continue_add =  FALSE;
        }
    }
    else if (reply == BUTTON_ADD)   /*按下添加键*/
    {
        strcpy(shop_data[0], (gchar *)gtk_entry_get_text(GTK_ENTRY(entry[0])));
        for (index = 2; index < 6; index++)
            strcpy(shop_data[index], (gchar *)
                gtk_entry_get_text(GTK_ENTRY(entry[index-1])));

        site_id = atoi(shop_data[0]);
        g_message("site_id=%d,site_amount=%d", site_id, site_amount);
        if (site_id > site_amount || site_id <= 0)
        {
            g_message("无此网站");
            if (warning_label == NULL)
            {
                g_message("warning_label == NULL");
                warning_label = gtk_label_new("无此网站,请检查所属网站编号!");
                gtk_box_pack_start(GTK_BOX(GNOME_DIALOG(shop_dialog)->vbox),
                    warning_label, TRUE, TRUE, 0);
                gtk_widget_show(warning_label);
            }
        }
        else
        {
            if (warning_label != NULL)
            {
                gtk_widget_destroy(warning_label);
                warning_label = NULL;
            }
```

```
                    for (index = 1; index < 5; index++)
                        gtk_entry_set_text(GTK_ENTRY(entry[index]), "");

                    AddShop(shop_data);
                    OnDataChanged();
                }

                continue_add = TRUE;
            }
            else
                continue_add = FALSE;
        if (id == EDIT_SHOP)
            continue_add = FALSE;
    }

}

void CreateDealDialog(gint id, gint selected_row)          /*创建添加或编辑交易对话框*/

{
  GtkWidget *deal_dialog = NULL;
  GtkWidget *entry[5];
  GtkWidget *label;
  GtkWidget *warning_label = NULL;
  GtkWidget *table;
  gchar *dialog_title[2] = {"添加交易", "编辑交易"};
  gchar former_shop_id[10];
  gint index;
  gboolean continue_add = TRUE;
  gint reply;
  gint shop_id;

  if (deal_dialog != NULL)
  {
      g_message("deal_dialog!=NULL");
      gdk_window_show(deal_dialog->window);
      gdk_window_raise(deal_dialog->window);
  }
  else
  {
      /*绘制对话框*/
      if (id == ADD_DEAL)
          deal_dialog = gnome_dialog_new(dialog_title[id], "添加", "确定", "取消", NULL);
      else
```

```
    deal_dialog = gnome_dialog_new(dialog_title[id],  "确定", "取消", NULL);
table = gtk_table_new(5, 2, FALSE);
gtk_box_pack_start(GTK_BOX(GNOME_DIALOG(deal_dialog)->vbox),table, TRUE, TRUE, 0);

label = gtk_label_new("所属店铺编号 ");
gtk_table_attach(GTK_TABLE(table), label, 0, 1, 0, 1, GTK_EXPAND | GTK_FILL, GTK_EXPAND
    | GTK_FILL, 5, 5);
entry[0] = gtk_entry_new_with_max_length(10);
gtk_table_attach(GTK_TABLE(table), entry[0], 1, 2, 0, 1, GTK_EXPAND | GTK_FILL,
    GTK_EXPAND | GTK_FILL, 5, 5);

label = gtk_label_new("支付类型 ");
gtk_table_attach(GTK_TABLE(table), label, 0, 1, 1, 2, GTK_EXPAND | GTK_FILL, GTK_EXPAND
    | GTK_FILL, 5, 5);
entry[1] = gtk_entry_new_with_max_length(1);
gtk_table_attach(GTK_TABLE(table), entry[1], 1, 2, 1, 2, GTK_EXPAND | GTK_FILL,
    GTK_EXPAND | GTK_FILL, 5, 5);

label = gtk_label_new("金额 ");
gtk_table_attach(GTK_TABLE(table), label, 0, 1, 2, 3, GTK_EXPAND | GTK_FILL, GTK_EXPAND
    | GTK_FILL, 5, 5);
entry[2] = gtk_entry_new_with_max_length(20);
gtk_table_attach(GTK_TABLE(table), entry[2], 1, 2, 2, 3, GTK_EXPAND | GTK_FILL,
    GTK_EXPAND | GTK_FILL, 5, 5);

label = gtk_label_new("日期 ");
gtk_table_attach(GTK_TABLE(table), label, 0, 1, 3, 4, GTK_EXPAND | GTK_FILL, GTK_EXPAND
    | GTK_FILL, 5, 5);
entry[3] = gtk_entry_new_with_max_length(20);
gtk_table_attach(GTK_TABLE(table), entry[3], 1, 2, 3, 4, GTK_EXPAND | GTK_FILL,
    GTK_EXPAND | GTK_FILL, 5, 5);

label = gtk_label_new("客户所在地 ");
gtk_table_attach(GTK_TABLE(table), label, 0, 1, 4, 5, GTK_EXPAND | GTK_FILL, GTK_EXPAND
    | GTK_FILL, 5, 5);
entry[4] = gtk_entry_new_with_max_length(20);
gtk_table_attach(GTK_TABLE(table), entry[4], 1, 2, 4, 5, GTK_EXPAND | GTK_FILL,
    GTK_EXPAND | GTK_FILL, 5, 5);


if (id == EDIT_DEAL)
{
    gtk_entry_set_text(GTK_ENTRY(entry[0]), deal_data_pointer[0]);
    for (index = 1; index < 5; index++)
```

```
                gtk_entry_set_text(GTK_ENTRY(entry[index]), deal_data_pointer[index+1]);
        }

        gnome_dialog_set_parent((GnomeDialog *)deal_dialog, (GtkWindow *)window);

        for (index = 0; index < 5; index++)
            gnome_dialog_editable_enters(GNOME_DIALOG(deal_dialog),
                GTK_EDITABLE(entry[index]));
        if (id == ADD_DEAL)
            gnome_dialog_set_default((GnomeDialog *)deal_dialog, BUTTON_OK);
        else
            gnome_dialog_set_default((GnomeDialog *)deal_dialog, BUTTON_OK - 1);

        gtk_widget_show_all(deal_dialog);
    }

    while (continue_add)
    {
        reply = gnome_dialog_run(GNOME_DIALOG(deal_dialog));
        if (id == EDIT_DEAL && reply >= 0)
            reply++;

        if (reply == BUTTON_CANCEL)      /*按下取消键*/
        {
            gtk_widget_destroy(deal_dialog);
            continue_add = FALSE;
        }
        else if (reply == BUTTON_OK)     /*按下确定键*/
             {
                strcpy(deal_data[0], (gchar *)gtk_entry_get_text(GTK_ENTRY(entry[0])));
                for (index = 2; index < 6; index++)
                    strcpy(deal_data[index], (gchar *)
                        gtk_entry_get_text(GTK_ENTRY(entry[index-1])));

                shop_id = atoi(deal_data[0]);
                if (shop_id > shop_amount)      /*如果shop_id大于shop_amount，即不存在该
店铺编号*/
                {
                    if (warning_label == NULL)
                    {
                        warning_label = gtk_label_new("无此店铺,请检查所属店铺编号!");
                        gtk_box_pack_start(GTK_BOX(GNOME_DIALOG(deal_dialog)->vbox),
                            warning_label, TRUE, TRUE, 0);
                        gtk_widget_show(warning_label);
```

```
            }
            continue_add = TRUE;
        }
        else
        {
             if (id == ADD_DEAL)
                 AddDeal(deal_data);
             else
             {
                 strcpy(former_shop_id, deal_data_pointer[0]);
                 g_message("former_shop_id = %s", former_shop_id);
                 EditDeal(selected_row, former_shop_id, deal_data);
             }
             OnDataChanged();
             gtk_widget_destroy(deal_dialog);
             continue_add = FALSE;
        }
    }
    else if (reply == BUTTON_ADD)   /*按下添加键*/
    {
        strcpy(deal_data[0], (gchar *)gtk_entry_get_text(GTK_ENTRY(entry[0])));
        for (index = 2; index < 6; index++)
            strcpy(deal_data[index], (gchar *)
                 gtk_entry_get_text(GTK_ENTRY(entry[index-1])));

        shop_id = atoi(deal_data[0]);
        if (shop_id > shop_amount)
        {
            if (warning_label == NULL)
            {
                 warning_label = gtk_label_new("无此店铺,请检查所属店铺编号!");
                 gtk_box_pack_start(GTK_BOX(GNOME_DIALOG(deal_dialog)->vbox),
                      warning_label, TRUE, TRUE, 0);
                 gtk_widget_show(warning_label);
            }
        }
        else
        {
             if (warning_label != NULL)
             {
                 gtk_widget_destroy(warning_label);
                 warning_label = NULL;
             }
             AddDeal(deal_data);
```

```
                        OnDataChanged();
                        for (index = 1; index < 5; index++)
                            gtk_entry_set_text(GTK_ENTRY(entry[index]), "");
                    }

                    continue_add = TRUE;
                }
                else
                    continue_add = FALSE;
            if (id == EDIT_DEAL)
                continue_add = FALSE;
        }
    }

    #undef BUTTON_OK
    #undef BUTTON_CANCEL
    #define BUTTON_OK 0
    #define BUTTON_CANCEL 1

    gboolean SearchDialog(gint current_page)    /*查找对话框*/
    {
      GtkWidget *search_dialog = NULL;
      GtkWidget *table;
      GtkWidget *entry[3];
      GtkWidget *label;
      gchar *dialog_title[3] = {"查找网站", "查找店铺", "查找交易"};
      gchar *warning = "如果填写多项,将只根据第一项进行查询";
      gchar *search_basis[][3] = {{"根据名称 :", "根据地址 :"}, {"根据名称 :", "根据负责人 :",
"根据联系地 :"}, {"根据日期 :", "根据客户地 :"}};

      gint index;

      if (search_dialog != NULL)
      {
          g_message("deal_dialog!=NULL");
          gdk_window_show(search_dialog->window);
          gdk_window_raise(search_dialog->window);
      }
      else
      {
          search_dialog = gnome_dialog_new(dialog_title[current_page], GNOME_STOCK_BUTTON_OK,
              GNOME_STOCK_BUTTON_CANCEL, NULL);
          if (current_page == SHOP_PAGE)
              table = gtk_table_new(4, 2, FALSE);
```

```
else
    table = gtk_table_new(3, 2, FALSE);
gtk_box_pack_start(GTK_BOX(GNOME_DIALOG(search_dialog)->vbox),table, TRUE, TRUE, 0);



label = gtk_label_new(search_basis[current_page][0]);
gtk_table_attach(GTK_TABLE(table), label, 0, 1, 0, 1, GTK_EXPAND | GTK_FILL, GTK_EXPAND
    | GTK_FILL, 5, 5);
entry[0] = gtk_entry_new_with_max_length(20);
gtk_table_attach(GTK_TABLE(table), entry[0], 1, 2, 0, 1, GTK_EXPAND | GTK_FILL,
    GTK_EXPAND | GTK_FILL, 5, 5);

label = gtk_label_new(search_basis[current_page][1]);
gtk_table_attach(GTK_TABLE(table), label, 0, 1, 1, 2, GTK_EXPAND | GTK_FILL, GTK_EXPAND
    | GTK_FILL, 5, 5);
entry[1] = gtk_entry_new_with_max_length(20);
gtk_table_attach(GTK_TABLE(table), entry[1], 1, 2, 1, 2, GTK_EXPAND | GTK_FILL,
    GTK_EXPAND | GTK_FILL, 5, 5);

if (current_page == SHOP_PAGE)
{
    label = gtk_label_new(search_basis[current_page][2]);
    gtk_table_attach(GTK_TABLE(table), label, 0, 1, 2, 3, GTK_EXPAND | GTK_FILL,
        GTK_EXPAND | GTK_FILL, 5, 5);
    entry[2] = gtk_entry_new_with_max_length(20);
    gtk_table_attach(GTK_TABLE(table), entry[2], 1, 2, 2, 3, GTK_EXPAND | GTK_FILL,
        GTK_EXPAND | GTK_FILL, 5, 5);

    label = gtk_label_new(warning);
    gtk_table_attach(GTK_TABLE(table), label, 0, 2, 3, 4, GTK_EXPAND | GTK_FILL,
        GTK_EXPAND | GTK_FILL, 5, 5);
}
else
{
    label = gtk_label_new(warning);
    gtk_table_attach(GTK_TABLE(table), label, 0, 2, 2, 3, GTK_EXPAND | GTK_FILL,
        GTK_EXPAND | GTK_FILL, 5, 5);
}

gnome_dialog_set_parent((GnomeDialog *)search_dialog, (GtkWindow *)window);
for (index = 0; index < 2; index++)
    gnome_dialog_editable_enters(GNOME_DIALOG(search_dialog),
        GTK_EDITABLE(entry[index]));
```

```
        if (current_page == SHOP_PAGE)
            gnome_dialog_editable_enters(GNOME_DIALOG(search_dialog),
                GTK_EDITABLE(entry[2]));
        gnome_dialog_set_default((GnomeDialog *)search_dialog, BUTTON_OK);

        gtk_widget_show_all(search_dialog);
    }

    gint reply = gnome_dialog_run(GNOME_DIALOG(search_dialog));
    if (reply == BUTTON_CANCEL)
    {
        gtk_widget_destroy(search_dialog);
        return FALSE;
    }
    else if (reply == BUTTON_OK)
        {
            for (index = 0; index < 2; index++)
                strcpy(search_keys[index], (gchar *)
                    gtk_entry_get_text(GTK_ENTRY(entry[index])));
            if (current_page == SHOP_PAGE)
                strcpy(search_keys[2], (gchar *)gtk_entry_get_text(GTK_ENTRY(entry[2])));
            g_message("search_keys[0]:%s", search_keys[0]);
            g_message("search_keys[1]:%s", search_keys[1]);
            g_message("search_keys[2]:%s", search_keys[2]);
            gtk_widget_destroy(search_dialog);
            return TRUE;
        }
    return FALSE;
}

gboolean DelSiteDialog(void)  /*删除网站对话框*/
{
    GtkWidget *del_site_dialog = NULL;
    GtkWidget *label;
    GtkWidget *table;
    gchar *dialog_title = "删除网站";

    if (del_site_dialog != NULL)
    {
        g_message("del_site_dialog!=NULL");
        gdk_window_show(del_site_dialog->window);
        gdk_window_raise(del_site_dialog->window);
    }
    else
```

```
{
    del_site_dialog = gnome_dialog_new(dialog_title, GNOME_STOCK_BUTTON_OK,
        GNOME_STOCK_BUTTON_CANCEL, NULL);
    table = gtk_table_new(2, 2, FALSE);
    gtk_box_pack_start(GTK_BOX(GNOME_DIALOG(del_site_dialog)->vbox),table,
        TRUE, TRUE, 0);

    label = gtk_label_new("网站名称 ");
    gtk_table_attach((GtkTable*)table, label, 0, 1, 0, 1, GTK_EXPAND | GTK_FILL, GTK_EXPAND
        | GTK_FILL, 5, 5);
    label = gtk_label_new(site_data_pointer[1]);
    gtk_table_attach((GtkTable*)table, label, 1, 2, 0, 1, GTK_EXPAND | GTK_FILL, GTK_EXPAND
        | GTK_FILL, 5, 5);

    label = gtk_label_new("网站地址 ");
    gtk_table_attach((GtkTable*)table, label, 0, 1, 1, 2, GTK_EXPAND | GTK_FILL, GTK_EXPAND
        | GTK_FILL, 5, 5);
    label = gtk_label_new(site_data_pointer[2]);
    gtk_table_attach((GtkTable*)table,label, 1, 2, 1, 2, GTK_EXPAND | GTK_FILL, GTK_EXPAND
        | GTK_FILL, 5, 5);

    label = gtk_label_new("");
    gtk_box_pack_start(GTK_BOX(GNOME_DIALOG(del_site_dialog)->vbox),label,
        TRUE, TRUE, 0);
    label = gtk_label_new("你确认要删除此网站及其以下所有信息?");
    gtk_box_pack_start(GTK_BOX(GNOME_DIALOG(del_site_dialog)->vbox),label,
        TRUE, TRUE, 0);
    label = gtk_label_new("(将重新对店铺及交易信息进行编号)");
    gtk_box_pack_start(GTK_BOX(GNOME_DIALOG(del_site_dialog)->vbox),label,
        TRUE, TRUE, 0);

    gnome_dialog_set_parent((GnomeDialog *)del_site_dialog, (GtkWindow *)window);


    gtk_widget_show_all(del_site_dialog);
}
gint reply = gnome_dialog_run(GNOME_DIALOG(del_site_dialog));
if (reply == BUTTON_CANCEL)
{
    gtk_widget_destroy(del_site_dialog);
    return FALSE;
}
else if (reply == BUTTON_OK)
     {
```

```
            gtk_widget_destroy(del_site_dialog);
            OnDataChanged();
            return TRUE;
        }
  return FALSE;
}

gboolean DelShopDialog(void)       /*删除店铺对话框*/
{
  GtkWidget *del_shop_dialog = NULL;
  GtkWidget *label;
  GtkWidget *table;
  gchar *dialog_title = "删除店铺";

  if (del_shop_dialog != NULL)
  {
      g_message("del_shop_dialog!=NULL");
      gdk_window_show(del_shop_dialog->window);
      gdk_window_raise(del_shop_dialog->window);
  }
  else
  {
      del_shop_dialog = gnome_dialog_new(dialog_title, GNOME_STOCK_BUTTON_OK,
          GNOME_STOCK_BUTTON_CANCEL, NULL);
      table = gtk_table_new(5, 2, FALSE);
      gtk_box_pack_start(GTK_BOX(GNOME_DIALOG(del_shop_dialog)->vbox),table,
          TRUE, TRUE, 0);

      label = gtk_label_new("所属网站编号 ");
      gtk_table_attach((GtkTable*)table, label, 0, 1, 0, 1, GTK_EXPAND | GTK_FILL, GTK_EXPAND
          | GTK_FILL, 5, 5);
      label = gtk_label_new(shop_data_pointer[0]);
      gtk_table_attach((GtkTable*)table, label, 1, 2, 0, 1, GTK_EXPAND | GTK_FILL, GTK_EXPAND
          | GTK_FILL, 5, 5);

      label = gtk_label_new("名称 ");
      gtk_table_attach((GtkTable*)table, label, 0, 1, 1, 2, GTK_EXPAND | GTK_FILL, GTK_EXPAND
          | GTK_FILL, 5, 5);
      label = gtk_label_new(shop_data_pointer[2]);
      gtk_table_attach((GtkTable*)table, label, 1, 2, 1, 2, GTK_EXPAND | GTK_FILL, GTK_EXPAND
          | GTK_FILL, 5, 5);

      label = gtk_label_new("负责人 ");
      gtk_table_attach((GtkTable*)table, label, 0, 1, 2, 3, GTK_EXPAND | GTK_FILL, GTK_EXPAND
```

```
        | GTK_FILL, 5, 5);
    label = gtk_label_new(shop_data_pointer[3]);
    gtk_table_attach((GtkTable*)table, label, 1, 2, 2, 3, GTK_EXPAND | GTK_FILL, GTK_EXPAND
        | GTK_FILL, 5, 5);

    label = gtk_label_new("联系地 ");
    gtk_table_attach((GtkTable*)table, label, 0, 1, 3, 4, GTK_EXPAND | GTK_FILL, GTK_EXPAND
        | GTK_FILL, 5, 5);
    label = gtk_label_new(shop_data_pointer[4]);
    gtk_table_attach((GtkTable*)table, label, 1, 2, 3, 4, GTK_EXPAND | GTK_FILL, GTK_EXPAND
        | GTK_FILL, 5, 5);

    label = gtk_label_new("开户银行 ");
    gtk_table_attach((GtkTable*)table, label, 0, 1, 4, 5, GTK_EXPAND | GTK_FILL, GTK_EXPAND
        | GTK_FILL, 5, 5);
    label = gtk_label_new(shop_data_pointer[5]);
    gtk_table_attach((GtkTable*)table, label, 1, 2, 4, 5, GTK_EXPAND | GTK_FILL, GTK_EXPAND
        | GTK_FILL, 5, 5);

    label = gtk_label_new("");
    gtk_box_pack_start(GTK_BOX(GNOME_DIALOG(del_shop_dialog)->vbox),label,
        TRUE, TRUE, 0);

    label = gtk_label_new("你确认要删除此店铺及其以下所有信息?");
    gtk_box_pack_start(GTK_BOX(GNOME_DIALOG(del_shop_dialog)->vbox),label,
        TRUE, TRUE, 0);

    label = gtk_label_new("(将重新对交易信息进行编号)");
    gtk_box_pack_start(GTK_BOX(GNOME_DIALOG(del_shop_dialog)->vbox),label,
        TRUE, TRUE, 0);

    gnome_dialog_set_parent((GnomeDialog *)del_shop_dialog, (GtkWindow *)window);

    gtk_widget_show_all(del_shop_dialog);
}

gint reply = gnome_dialog_run(GNOME_DIALOG(del_shop_dialog));
if (reply == BUTTON_CANCEL)
{
    gtk_widget_destroy(del_shop_dialog);
    return FALSE;
}
else if (reply == BUTTON_OK)
     {
```

```
            gtk_widget_destroy(del_shop_dialog);
            OnDataChanged();
            return TRUE;
        }
  return FALSE;
}

gboolean DelDealDialog(void)       /*删除交易对话框*/
{

  GtkWidget *del_deal_dialog = NULL;
  GtkWidget *label;
  GtkWidget *table;
  gchar *dialog_title = "删除交易";

  if (del_deal_dialog != NULL)
  {
      g_message("del_deal_dialog!=NULL");
      gdk_window_show(del_deal_dialog->window);
      gdk_window_raise(del_deal_dialog->window);
  }
  else
  {
      del_deal_dialog = gnome_dialog_new(dialog_title, GNOME_STOCK_BUTTON_OK,
          GNOME_STOCK_BUTTON_CANCEL, NULL);
      table = gtk_table_new(5, 2, FALSE);
      gtk_box_pack_start(GTK_BOX(GNOME_DIALOG(del_deal_dialog)->vbox),table,
          TRUE, TRUE, 0);

      label = gtk_label_new("所属店铺编号 ");
      gtk_table_attach((GtkTable*)table, label, 0, 1, 0, 1, GTK_EXPAND | GTK_FILL, GTK_EXPAND
          | GTK_FILL, 5, 5);
      label = gtk_label_new(deal_data_pointer[0]);
      gtk_table_attach((GtkTable*)table, label, 1, 2, 0, 1, GTK_EXPAND | GTK_FILL, GTK_EXPAND
          | GTK_FILL, 5, 5);

      label = gtk_label_new("支付类型 ");
      gtk_table_attach((GtkTable*)table, label, 0, 1, 1, 2, GTK_EXPAND | GTK_FILL, GTK_EXPAND
          | GTK_FILL, 5, 5);
      label = gtk_label_new(deal_data_pointer[2]);
      gtk_table_attach((GtkTable*)table, label, 1, 2, 1, 2, GTK_EXPAND | GTK_FILL, GTK_EXPAND
          | GTK_FILL, 5, 5);

      label = gtk_label_new("金额 ");
```

```
    gtk_table_attach((GtkTable*)table, label, 0, 1, 2, 3, GTK_EXPAND | GTK_FILL, GTK_EXPAND
        | GTK_FILL, 5, 5);
    label = gtk_label_new(deal_data_pointer[3]);
    gtk_table_attach((GtkTable*)table, label, 1, 2, 2, 3, GTK_EXPAND | GTK_FILL, GTK_EXPAND
        | GTK_FILL, 5, 5);

    label = gtk_label_new("日期 ");
    gtk_table_attach((GtkTable*)table, label, 0, 1, 3, 4, GTK_EXPAND | GTK_FILL, GTK_EXPAND
        | GTK_FILL, 5, 5);
    label = gtk_label_new(deal_data_pointer[4]);
    gtk_table_attach((GtkTable*)table, label, 1, 2, 3, 4, GTK_EXPAND | GTK_FILL, GTK_EXPAND
        | GTK_FILL, 5, 5);

    label = gtk_label_new("客户所在地 ");
    gtk_table_attach((GtkTable*)table, label, 0, 1, 4, 5, GTK_EXPAND | GTK_FILL, GTK_EXPAND
        | GTK_FILL, 5, 5);
    label = gtk_label_new(deal_data_pointer[5]);
    gtk_table_attach((GtkTable*)table, label, 1, 2, 4, 5, GTK_EXPAND | GTK_FILL, GTK_EXPAND
        | GTK_FILL, 5, 5);

    label = gtk_label_new("");
    gtk_box_pack_start(GTK_BOX(GNOME_DIALOG(del_deal_dialog)->vbox),label,
        TRUE, TRUE, 0);
    label = gtk_label_new("你确认要删除此次交易信息?");
    gtk_box_pack_start(GTK_BOX(GNOME_DIALOG(del_deal_dialog)->vbox),label,
        TRUE, TRUE, 0);

    gnome_dialog_set_parent((GnomeDialog *)del_deal_dialog, (GtkWindow *)window);

    gtk_widget_show_all(del_deal_dialog);
}

gint reply = gnome_dialog_run(GNOME_DIALOG(del_deal_dialog));
if (reply == BUTTON_CANCEL)
{
    gtk_widget_destroy(del_deal_dialog);
    return FALSE;
}
else if (reply == BUTTON_OK)
     {
          gtk_widget_destroy(del_deal_dialog);
          OnDataChanged();
          return TRUE;
     }
```

```
  return FALSE;
}
void CopyData(gint current_page)          /*拷贝所需的一些系统自动生成的数据(如商家编号，
店铺总数)，用于编辑信息功能*/
{
  switch (current_page)
  {
       case SITE_PAGE:
            strcpy(site_data[0], site_data_pointer[0]);
            strcpy(site_data[3], site_data_pointer[3]);
            break;
       case SHOP_PAGE:
            strcpy(shop_data[1], shop_data_pointer[1]);
            strcpy(shop_data[6], shop_data_pointer[6]);
            break;
       case DEAL_PAGE:
            strcpy(deal_data[1], deal_data_pointer[1]);
            break;
  }
}
gboolean SelectSearchedKey(gint current_page)        /*选取搜索根据的关键字*/
{
  g_message("SelectSearchedKey");

  if (strcmp(search_keys[0], ""))       /*如果第一项不为空，根据第一项进行搜索，下面类似*/
       return SelectSearchedKind(current_page, 0, search_keys[0]);
  else if (strcmp(search_keys[1], ""))
       return SelectSearchedKind(current_page, 1, search_keys[1]);
  else if (strcmp(search_keys[2], ""))
       return SelectSearchedKind(current_page, 2, search_keys[2]);
  else
  {
       g_message("no input");
       ShowMessageBox("未输入查找项!");
       return FALSE;
  }
}
gboolean SelectSearchedKind(gint current_page, gint search_key_id, gchar *search_key)
  /*选取搜索哪类信息*/
{
  switch (current_page)
  {
       case SITE_PAGE:
            g_message("search site");
```

```
            return SearchSite(search_key_id, search_key);
        case SHOP_PAGE:
            g_message("search shop");
            return SearchShop(search_key_id, search_key);
        case DEAL_PAGE:
            g_message("search deal");
            return SearchDeal(search_key_id, search_key);
    }
    return FALSE;
}
/*****************************************************************************************/
/*****************************统计菜单栏需要调用的函数*****************************/
/*****************************************************************************************/
void StatiSiteDialog(void);        /*选择"统计/网站排行榜"*/
void StatiShopDialog(void);        /*选择"统计/店铺交易情况"*/
void StatiDealDialog(void);        /*选择"统计/地区贸易情况"*/

void StatiInfo(gint id)        /*根据id选择统计哪方面信息*/
{
    void (*stati_func_pointer[3])(void) = {StatiSiteDialog, StatiShopDialog, StatiDealDialog};

    if (site_amount == 0)
    {
        ShowMessageBox("无网站信息！");
        gtk_notebook_set_current_page(GTK_NOTEBOOK(notebook), SITE_PAGE);
    }
    else if (shop_amount == 0)
    {
        ShowMessageBox("无店铺信息！");
        gtk_notebook_set_current_page(GTK_NOTEBOOK(notebook), SHOP_PAGE);
    }
    else
        stati_func_pointer[id]();
}


void StatiSiteDialog(void)    /*选择"统计/网站排行榜"*/
{
    GtkWidget *stati_site_dialog = NULL;
    GtkWidget *table;
    GtkWidget *label;
    GtkWidget *entry;
    GtkWidget *warning_label = NULL;
    gchar *dialog_title = "网站排行榜";
```

```
gchar date[20];
gint reply;

if (stati_site_dialog != NULL)
{
     g_message("stati_site_dialog!=NULL");
     gdk_window_show(stati_site_dialog->window);
     gdk_window_raise(stati_site_dialog->window);
}
else
{
     stati_site_dialog = gnome_dialog_new(dialog_title, GNOME_STOCK_BUTTON_OK,
          GNOME_STOCK_BUTTON_CANCEL, NULL);
     table = gtk_table_new(3, 1, FALSE);
     gtk_box_pack_start(GTK_BOX(GNOME_DIALOG(stati_site_dialog)->vbox),table,
          TRUE, TRUE, 0);

     label = gtk_label_new("请输入年月份(如201010)：");
     gtk_table_attach((GtkTable*)table, label, 0, 1, 0, 1, GTK_EXPAND | GTK_FILL, GTK_EXPAND
          | GTK_FILL, 5, 5);
     entry = gtk_entry_new_with_max_length(20);
     gtk_table_attach((GtkTable*)table, entry, 0, 1, 1, 2, GTK_EXPAND | GTK_FILL, GTK_EXPAND
          | GTK_FILL, 5, 5);

     gnome_dialog_set_parent((GnomeDialog *)stati_site_dialog, (GtkWindow *)window);

     gnome_dialog_editable_enters(GNOME_DIALOG(stati_site_dialog), GTK_EDITABLE(entry));
     gnome_dialog_set_default((GnomeDialog *)stati_site_dialog, BUTTON_OK);

     gtk_widget_show_all(stati_site_dialog);
}

while (1)
{
     reply = gnome_dialog_run(GNOME_DIALOG(stati_site_dialog));
     if (reply == BUTTON_OK)
     {
          strcpy(date, (gchar *)gtk_entry_get_text(GTK_ENTRY(entry)));
          if (!strcmp(date, ""))
          {
               if (warning_label == NULL)
               {
                    warning_label = gtk_label_new("请输入完整信息！");
                    gtk_table_attach((GtkTable*)table, warning_label, 0, 1, 2,3, GTK_EXPAND
```

```
                        | GTK_FILL, GTK_EXPAND | GTK_FILL, 5, 5);
                    gtk_widget_show(warning_label);
                }
                continue;
            }
            gtk_widget_destroy(stati_site_dialog);
            StatiSite(date);
            break;
        }
        else if (reply == BUTTON_CANCEL)
        {
            gtk_widget_destroy(stati_site_dialog);
            break;
        }
        else
            break;
    }
}
void StatiShopDialog(void)    /*选择"统计/店铺交易情况"*/
{
    GtkWidget *stati_shop_dialog = NULL;
    GtkWidget *table;
    GtkWidget *label;
    GtkWidget *entry;
    GtkWidget *warning_label = NULL;
    gchar *dialog_title = "店铺交易信息";
    gchar date[20];
    gint reply;

    if (stati_shop_dialog != NULL)
    {
        g_message("stati_shop_dialog!=NULL");
        gdk_window_show(stati_shop_dialog->window);
        gdk_window_raise(stati_shop_dialog->window);
    }
    else
    {
        stati_shop_dialog = gnome_dialog_new(dialog_title, GNOME_STOCK_BUTTON_OK,
                GNOME_STOCK_BUTTON_CANCEL, NULL);
        table = gtk_table_new(3, 1, FALSE);
        gtk_box_pack_start(GTK_BOX(GNOME_DIALOG(stati_shop_dialog)->vbox),table,
            TRUE, TRUE, 0);

        label = gtk_label_new("请输入年份(如2010)：");
```

```
    gtk_table_attach((GtkTable*)table, label, 0, 1, 0, 1, GTK_EXPAND | GTK_FILL, GTK_EXPAND
        | GTK_FILL, 5, 5);
    entry = gtk_entry_new_with_max_length(20);
    gtk_table_attach((GtkTable*)table, entry, 0, 1, 1, 2, GTK_EXPAND | GTK_FILL, GTK_EXPAND
        | GTK_FILL, 5, 5);

    gnome_dialog_set_parent((GnomeDialog *)stati_shop_dialog, (GtkWindow *)window);

    gnome_dialog_editable_enters(GNOME_DIALOG(stati_shop_dialog), GTK_EDITABLE(entry));
    gnome_dialog_set_default((GnomeDialog *)stati_shop_dialog, BUTTON_OK);

    gtk_widget_show_all(stati_shop_dialog);
}

while (1)
{
    reply = gnome_dialog_run(GNOME_DIALOG(stati_shop_dialog));
    if (reply == BUTTON_OK)
    {
        strcpy(date, (gchar *)gtk_entry_get_text(GTK_ENTRY(entry)));
        if (!strcmp(date, ""))
        {
            if (warning_label == NULL)
            {
                warning_label = gtk_label_new("请输入完整信息！");
                gtk_table_attach((GtkTable*)table, warning_label, 0, 1, 2,3, GTK_EXPAND
                    | GTK_FILL, GTK_EXPAND | GTK_FILL, 5, 5);
                gtk_widget_show(warning_label);
            }
            continue;
        }

        gtk_widget_destroy(stati_shop_dialog);
        StatiShop(date);
        break;
    }
    else if (reply == BUTTON_CANCEL)
    {
        gtk_widget_destroy(stati_shop_dialog);
        break;
    }
    else
        break;
}
```

```
}
void StatiDealDialog(void)    /*选择"统计/地区贸易情况"*/
{
  GtkWidget *stati_deal_dialog = NULL;
  GtkWidget *table;
  GtkWidget *label;
  GtkWidget *entry[3];
  GtkWidget *warning_label = NULL;
  gchar *dialog_title = "地区贸易关系";
  gchar stati_data[3][20];
  gint reply, index;

  if (stati_deal_dialog != NULL)
  {
       g_message("stati_deal_dialog!=NULL");
       gdk_window_show(stati_deal_dialog->window);
       gdk_window_raise(stati_deal_dialog->window);
  }
  else
  {
       stati_deal_dialog = gnome_dialog_new(dialog_title, GNOME_STOCK_BUTTON_OK,
            GNOME_STOCK_BUTTON_CANCEL, NULL);
       table = gtk_table_new(5, 2, FALSE);
       gtk_box_pack_start(GTK_BOX(GNOME_DIALOG(stati_deal_dialog)->vbox),table,
            TRUE, TRUE, 0);

       label = gtk_label_new("请输入两地区名称和年月份：");
       gtk_table_attach((GtkTable*)table, label, 0, 2, 0, 1, GTK_EXPAND | GTK_FILL, GTK_EXPAND
            | GTK_FILL, 5, 5);

       label = gtk_label_new("地区一：");
       gtk_table_attach((GtkTable*)table, label, 0, 1, 1, 2, GTK_EXPAND | GTK_FILL, GTK_EXPAND
            | GTK_FILL, 5, 5);
       entry[0] = gtk_entry_new_with_max_length(20);
       gtk_table_attach((GtkTable*)table, entry[0], 1, 2, 1, 2, GTK_EXPAND | GTK_FILL,
            GTK_EXPAND | GTK_FILL, 5, 5);

       label = gtk_label_new("地区二：");
       gtk_table_attach((GtkTable*)table, label, 0, 1, 2, 3, GTK_EXPAND | GTK_FILL, GTK_EXPAND
            | GTK_FILL, 5, 5);
       entry[1] = gtk_entry_new_with_max_length(20);
       gtk_table_attach((GtkTable*)table, entry[1], 1, 2, 2, 3, GTK_EXPAND | GTK_FILL,
            GTK_EXPAND | GTK_FILL, 5, 5);
```

```
    label = gtk_label_new("年月份：");
    gtk_table_attach((GtkTable*)table, label, 0, 1, 3, 4, GTK_EXPAND | GTK_FILL, GTK_EXPAND
        | GTK_FILL, 5, 5);
    entry[2] = gtk_entry_new_with_max_length(20);
    gtk_table_attach((GtkTable*)table, entry[2], 1, 2, 3,4, GTK_EXPAND | GTK_FILL,
        GTK_EXPAND | GTK_FILL, 5, 5);

    gnome_dialog_set_parent((GnomeDialog *)stati_deal_dialog, (GtkWindow *)window);

    for (index = 0; index < 3; index++)
        gnome_dialog_editable_enters(GNOME_DIALOG(stati_deal_dialog),
                GTK_EDITABLE(entry[index]));
    gnome_dialog_set_default((GnomeDialog *)stati_deal_dialog, BUTTON_OK);

    gtk_widget_show_all(stati_deal_dialog);
}

while (1)
{
    reply = gnome_dialog_run(GNOME_DIALOG(stati_deal_dialog));
    if (reply == BUTTON_OK)
    {
        strcpy(stati_data[0], (gchar *)gtk_entry_get_text(GTK_ENTRY(entry[0])));
        strcpy(stati_data[1], (gchar *)gtk_entry_get_text(GTK_ENTRY(entry[1])));
        strcpy(stati_data[2], (gchar *)gtk_entry_get_text(GTK_ENTRY(entry[2])));

        if (!strcmp(stati_data[0], "") || !strcmp(stati_data[1], "")
            || !strcmp(stati_data[2], ""))
        {
            if (warning_label == NULL)
            {
                warning_label = gtk_label_new("请输入完整信息！");
                gtk_table_attach((GtkTable*)table, warning_label, 0, 2, 4,5, GTK_EXPAND
                    | GTK_FILL, GTK_EXPAND | GTK_FILL, 5, 5);
                gtk_widget_show(warning_label);
            }
            continue;
        }
        g_message("area1=%s,area2=%s", (gchar *)stati_data[0], (gchar *)stati_data[1]);
        gtk_widget_destroy(stati_deal_dialog);
        StatiDeal(stati_data);
        break;
    }
    else  if (reply == BUTTON_CANCEL)
```

```
        {
            gtk_widget_destroy(stati_deal_dialog);
            break;
        }
        else
            break;
    }
}

/*****************************************************************************************/
/*************************************其他函数*********************************************/
/*****************************************************************************************/
void HandleSwitchPage(gint current_page, gint former_page)          /*处理切换页面*/
{
  gchar *str_site_id, *str_shop_id;
  gint site_id = 0, shop_id = 0;
  SiteInfo *site_current;
  ShopInfo *shop_current;

  if (!expand_all_info)  /*如果是“收起其他”模式*/
  {
      g_message("former_page=%d,current_page=%d", former_page, current_page);
      if (current_page == SHOP_PAGE)
      {
          if (former_page == SITE_PAGE)         /*从网站页面转到店铺页面*/
          {
              gtk_clist_get_text(GTK_CLIST(clist[0]), site_selected_row < 0 ? 0 :
                  site_selected_row, 0, &str_site_id);
              site_id = atoi(str_site_id);
              RedrawShopCList(site_id);
          }
          else if (former_page == DEAL_PAGE)    /*从交易页面转到店铺页面*/
          {
              gtk_clist_get_text(GTK_CLIST(clist[2]), deal_selected_row < 0 ? 0 :
                  deal_selected_row, 0, &str_shop_id);
              shop_id = atoi(str_shop_id);

              g_message("shop_id=%d", shop_id);
              shop_current = GetShop(shop_id); /*GetShop:根据shop_id返回对应的shop节点，
                  位于handle_data.c*/
              if (shop_current != NULL)
                  site_id = shop_current -> site_id;
              else
              {
```

```
                    gtk_clist_get_text(GTK_CLIST(clist[0]), site_selected_row < 0 ? 0 :
                        site_selected_row, 0, &str_site_id);
                    site_id = atoi(str_site_id);
                }
                RedrawShopCList(site_id);   /*RedrawShopCList:根据site_id刷新店铺信息下的
                    clist，位于handle_data.c*/
            }
        }
        else if (current_page == DEAL_PAGE)
        {
            if (former_page == SITE_PAGE)             /*从网站页面切换到交易页面*/
            {
                gtk_clist_get_text(GTK_CLIST(clist[0]), site_selected_row < 0 ? 0 :
                    site_selected_row, 0, &str_site_id);
                site_id = atoi(str_site_id);

                site_current = GetSite(site_id); /*GetSite:根据site_id返回对应的site节点，
                    位于handle_data.c*/
                ShowDealOfSite(site_current);         /*ShowDealOfSite:输出site_current所
                    指网站的所有交易信息，位于handle_data.c*/
            }
            else if (former_page == SHOP_PAGE)        /*从店铺页面切换到交易页面*/
            {
                gtk_clist_get_text(GTK_CLIST(clist[1]), shop_selected_row < 0 ? 0 :
                    shop_selected_row, 0, &str_site_id);
                gtk_clist_get_text(GTK_CLIST(clist[1]), shop_selected_row < 0 ? 0 :
                    shop_selected_row, 1, &str_shop_id);
                site_id = atoi(str_site_id);
                shop_id = atoi(str_shop_id);

                RedrawDealCList(site_id, shop_id);   /*RedrawDealCList:刷新交易信息下的
clist，位于handle_data.c*/
            }

        }
    }
}
```

v. get_user_data.h

```
#ifndef __CALLSBACK_H__
#define __CALLSBACK_H__
```

```
void FileNew(void);                                    /*选择"文件/新建"*/
void FileOpen(void);                                   /*选择"文件/打开"*/
void FileSave(void);                                   /*选择"文件/保存"*/
void FileSaveas(void);                                 /*选择"文件/另存为"*/
void ExpandAll(void);                                  /*选择"文件/更多/展开全部"*/
void RetractOthers(gint current_page);                 /*选择"文件/更多/收起其他"*/
void CheckData(void);                                  /*选择"文件/更多/数据排错"*/
void FileExit(void);                                   /*选择"文件/退出"*/

void AddInfo(gint current_page);                       /*选择"编辑/添加信息*/
void SearchInfo(gint current_page);                    /*选择"编辑/查找信息*/
void EditInfo(gint current_page, gint selected_row); /*选择"编辑/编辑信息"*/
void DelInfo(gint current_page, gint selected_row);  /*选择"编辑/删除信息"*/

void StatiInfo(gint id);      /*选择统计菜单栏任一项时调用，根据id选择统计哪方面信息*/

void HandleSwitchPage(gint current_page, gint former_page);    /*处理切换页面*/
void OnWindowDeleteEvent(GtkNotebook *notebook);               /*退出前调用的函数*/
void OnDataChanged(void);     /*数据发生改变,修改变量file_modifyed使其为TRUE*/
void OnDataSaved(void);       /*数据保存后调用的函数，修改变量file_modifyed使其为FALSE*/

#endif
```

vi. handle_data.c

```
/*
 * 数据操作模块
 *
 * 主要功能：读取、写入文件，链表节点的查询、删除等
 */

#include <gtk/gtk.h>
#include <libgnomeui/gnome-messagebox.h>
#include<stdio.h>
#include <string.h>
#include <stdlib.h>
#include "create_window.h"
#include "handle_data.h"


#define SITE_COLUMN 4
#define SHOP_COLUMN 7
#define DEAL_COLUMN 6
```

```
extern GtkWidget *clist[4];        /*定义于create_window.c*/

SiteInfo *site_head = NULL;             /*网站链表头指针*/
gint site_amount = 0;                   /*网站总数*/
gint shop_amount = 0;                   /*店铺总数*/
gint deal_amount = 0;                   /*交易总数*/
gboolean expand_all_info = TRUE;   /*标识是否展开所有信息*/


/*****************************************************************************************/
/*********************************网站信息相关操作*************************************/
/*****************************************************************************************/
void UpdateShopAndDealData(void);       /*更新店铺、交易的编号以及店铺的所属网站编号、交易
的所属店铺编号*/
void AddSite(gchar (*site_data)[50])         /*添加网站*/
{
  gint index, new_row;
  gchar *site_data_pointer[4];          /*指向site_data的指针数组*/

  SiteInfo *site_current = site_head;
  SiteInfo *new_site = (SiteInfo *)malloc(sizeof(SiteInfo));

  strcpy(new_site -> site_name, (gchar *)site_data[1]);
  strcpy(new_site -> site_addr, (gchar *)site_data[2]);
  new_site -> shop_amount = 0;
  new_site -> shop_head = NULL;

  g_message("new_site -> site_name: %s", new_site -> site_name);
  g_message("new_site -> site_addr: %s", new_site -> site_addr);

  if (site_current == NULL)         /*如果是头指针*/
  {
       g_message("site_current==NULL");
       site_amount++;
       new_site -> site_id = 1;
       site_head = new_site;
       new_site -> site_next = NULL;
  }
  else           /*直接插入链表末尾*/
  {
       g_message("site_current!=NULL");
       while (site_current -> site_next != NULL)
            site_current = site_current -> site_next;
```

```
        site_amount++;
        new_site -> site_id = site_amount;
        site_current -> site_next = new_site;
        new_site -> site_next = NULL;
    }

    sprintf(site_data[0], "%d", new_site -> site_id);
    sprintf(site_data[3], "%d", new_site -> shop_amount);

    for (index = 0; index < SITE_COLUMN; index++)
        site_data_pointer[index] = site_data[index];

    /*在clist增加一条信息*/
    new_row = gtk_clist_append(GTK_CLIST(clist[0]), site_data_pointer);
    gtk_clist_moveto(GTK_CLIST(clist[0]), new_row, 0, 0.0, 0.0);
    gtk_clist_select_row(GTK_CLIST(clist[0]), new_row, 0);
}
gboolean SearchSite(gint search_key_id, gchar *search_key)          /*查找网站*/
{
    g_message("SearchSite");
    g_message("search_key=%s", search_key);
    g_message("site_amount=%d", site_amount);
    gint index;
    gchar *temp;  /*临时变量，用来存储clist中每行对应单元格的数据*/

    for (index = 0; index < site_amount; index++)
    {
        gtk_clist_get_text(GTK_CLIST(clist[0]), index, search_key_id + 1, &temp);
        g_message("temp=%s", temp);
        if (!strcmp(search_key, temp))
        {
            g_message("find site");
            gtk_clist_moveto(GTK_CLIST(clist[0]), index, search_key_id + 1, 0.0, 0.0);
            gtk_clist_select_row(GTK_CLIST(clist[0]), index, search_key_id + 1);
            return TRUE;
        }
    }

    ShowMessageBox("未找到任何结果!");
    return FALSE;
}
void EditSite(gint selected_row, gchar (*site_data)[50])  /*编辑网站*/
{
    gint index, site_id;
```

```
  SiteInfo *site_current = site_head;

  for (index = 0; index < SITE_COLUMN; index++)
       gtk_clist_set_text(GTK_CLIST(clist[0]), selected_row, index, site_data[index]);

  site_id = atoi(site_data[0]);         /*将字符串转换为整型数字*/
  g_message("site_id=%d", site_id);

  for (index = 1; index < site_id; index++)
       site_current = site_current -> site_next;

  strcpy(site_current -> site_name, site_data[1]);
  strcpy(site_current -> site_addr, site_data[2]);

  g_message("site_current->site_name=%s",site_current -> site_name);
  g_message("site_current->site_addr=%s",site_current -> site_addr);
}
void DelSite(gint selected_row, gchar *str_site_id)       /*删除网站*/
{
  gint index, site_id = atoi(str_site_id);
  SiteInfo *site_current = site_head, *site_prior = site_current;
  ShopInfo *shop_current;
  DealInfo *deal_current;

  g_message("site_id=%d", site_id);

  gtk_clist_remove(GTK_CLIST(clist[0]), selected_row);

  for (index = 1; index < site_id; index++)      /*根据site_id进行查找*/
  {
       site_prior = site_current;
       site_current = site_current -> site_next;
  }

  if (site_current == site_head)
       site_head = site_current -> site_next;
  else
       site_prior -> site_next = site_current -> site_next;

  /*修改后面网站的编号*/
  if (site_head != NULL && site_prior -> site_next != NULL)
  {
       while ((site_prior = site_prior -> site_next)!= NULL)
            (site_prior -> site_id)--;
```

```
    }

    site_amount--;
    shop_amount -= site_current -> shop_amount;
    g_message("site_amount=%d", site_amount);

    /*删除site节点下的shop、deal链的数据*/
    shop_current = site_current -> shop_head;
    while (shop_current != NULL)
    {
        deal_current = shop_current -> deal_head;
        while (deal_current != NULL)
        {
            free(deal_current);
            deal_current = deal_current -> deal_next;
        }
        free(shop_current);
        shop_current = shop_current -> shop_next;
    }
    free(site_current);
    UpdateShopAndDealData();

    RedrawAllCList();  /*刷新所有clist的信息*/

    if (selected_row > site_amount - 1)
        selected_row = site_amount - 1;
    gtk_clist_moveto(GTK_CLIST(clist[0]), selected_row, 0, 0.0, 0.0);
    gtk_clist_select_row(GTK_CLIST(clist[0]), selected_row, 0);
  }
  void UpdateShopAndDealData(void)       /*更新店铺、交易的编号以及店铺的所属网站编号、交易
的所属店铺编号*/
  {
    gint shop_count = 0, deal_count = 0;
    SiteInfo *site_current = site_head;
    ShopInfo *shop_current;
    DealInfo *deal_current;

    while (site_current != NULL)
    {
        shop_current = site_current -> shop_head;
        while (shop_current != NULL)
        {
            shop_count++;
            shop_current -> shop_id = shop_count;
```

```
            shop_current -> site_id = site_current -> site_id;

            deal_current = shop_current -> deal_head;
            while (deal_current != NULL)
            {
                deal_count++;
                deal_current -> deal_id = deal_count;
                deal_current -> shop_id = shop_count;
                deal_current = deal_current -> deal_next;
            }

            shop_current = shop_current -> shop_next;
        }
        site_current = site_current -> site_next;
    }
  }
  /*****************************************************************************************/
  /*********************************店铺信息相关操作*********************************/
  /*****************************************************************************************/
  /*找到并编辑店铺*/
  void SearchAndEditShop(ShopInfo *shop_edited, gint selected_row, gchar
(*shop_data)[20]);void UpdateDealData(void);       /*更新店铺下deal的shop_id信息*/
  void UpdateSiteCList(SiteInfo *site_current, gint site_id);        /*更新网站clist中
site_id对应网站的店铺个数*/

  SiteInfo *GetSite(gint site_id)        /*根据site_id查找并返回对应的SiteInfo指针*/
  {
    SiteInfo *site_current = site_head;
    gint index;

    for (index = 1; index < site_id && site_current != NULL; index++)
        site_current = site_current -> site_next;
    return site_current;
  }

  void AddShop(gchar (*shop_data)[20])        /*增加店铺*/
  {
    g_message("AddShop");
    gint index = 1, new_row = 0;
    gchar *shop_data_pointer[7];
    gint site_id = atoi(shop_data[0]);
    SiteInfo *site_current = site_head;
    ShopInfo *shop_current  = NULL;
    ShopInfo *new_shop = (ShopInfo *)malloc(sizeof(ShopInfo));
```

```
site_current = GetSite(site_id);
(site_current -> shop_amount)++;
shop_current = site_current -> shop_head;

new_shop -> site_id = site_id;
strcpy(new_shop -> shop_name, shop_data[2]);
strcpy(new_shop -> seller_name, shop_data[3]);
strcpy(new_shop -> seller_addr, shop_data[4]);
strcpy(new_shop -> deposit_bank, shop_data[5]);

/*插入并给店铺进行编号*/
if (shop_current == NULL)
{
     g_message("shop_current == NULL");
     shop_amount++;

     site_current -> shop_head= new_shop;
     new_shop -> shop_next = NULL;
}
else
{
     g_message("shop_current != NULL");
     while (shop_current -> shop_next != NULL)
          shop_current = shop_current -> shop_next;

     shop_amount++;

     shop_current -> shop_next = new_shop;
     new_shop -> shop_next = NULL;
}

new_shop -> deal_head = NULL;
new_shop -> shop_id = shop_amount;
new_shop -> deal_amount = 0;
g_message("AddShop:shop_id=%d", new_shop -> shop_id);
sprintf(shop_data[1], "%d", new_shop -> shop_id);
sprintf(shop_data[6], "%d", new_shop -> deal_amount);

for (index = 0; index < SHOP_COLUMN; index++)
     shop_data_pointer[index] = shop_data[index];

/*在clist中增加一条信息*/
new_row = gtk_clist_append(GTK_CLIST(clist[1]), shop_data_pointer);
```

```
  gtk_clist_moveto(GTK_CLIST(clist[1]), new_row, 0, 0.0, 0.0);
  gtk_clist_select_row(GTK_CLIST(clist[1]), new_row, 0);

  UpdateSiteCList(site_current, site_id);
}
gboolean SearchShop(gint search_key_id, gchar *search_key)          /*查找店铺*/
{
  g_message("SearchShop");
  g_message("search_key=%s", search_key);
  g_message("shop_amount=%d", shop_amount);
  gint index, shop_row;
  gchar *temp;

  shop_row = GTK_CLIST(clist[1]) -> rows;
  for (index = 0; index < shop_row; index++)
  {
       gtk_clist_get_text(GTK_CLIST(clist[1]), index, search_key_id + 2, &temp);
       g_message("temp=%s", temp);
       if (!strcmp(search_key, temp))
       {
            g_message("find shop");
            gtk_clist_moveto(GTK_CLIST(clist[1]), index, search_key_id + 2, 0.0, 0.0);
            gtk_clist_select_row(GTK_CLIST(clist[1]), index, search_key_id + 2);
            return TRUE;
       }
  }

  ShowMessageBox("未找到任何结果!");
  return FALSE;
}
void EditShop(gint selected_row, gchar *former_site_id, gchar (*shop_data)[20])  /*编辑店
铺*/
{
  g_message("EditShop");
  g_message("former_site_id=%s, later_site_id=%s", former_site_id, shop_data[0]);
  ShopInfo *shop_edited = NULL;

  if (strcmp(former_site_id, shop_data[0])) /*如果修改了所属网站编号*/
  {
       shop_edited = DelShop(selected_row, former_site_id, shop_data[1], FALSE);

       SearchAndEditShop(shop_edited, selected_row, shop_data);

       shop_amount++;
```

```
        RedrawDealCList(0, 0);
    }
    else
        SearchAndEditShop(shop_edited, selected_row, shop_data);

}
/*找到并编辑店铺*/
void SearchAndEditShop(ShopInfo *shop_edited, gint selected_row, gchar (*shop_data)[20])
{
    g_message("SearchAndEditShop");
    gint index;
    gint site_id = atoi(shop_data[0]);
    gint shop_id = atoi(shop_data[1]);

    SiteInfo *site_current = site_head;
    ShopInfo *shop_current = NULL, *shop_prior;

    for (index = 0; index < SHOP_COLUMN; index++)
        gtk_clist_set_text(GTK_CLIST(clist[1]), selected_row, index, shop_data[index]);

    site_current = GetSite(site_id);
    shop_prior = shop_current = site_current -> shop_head;

    if (shop_edited != NULL)    /*如果修改了所属网站编号*/
    {
        if (shop_current == NULL)
        {
            g_message("shop_current == NULL");
            site_current -> shop_head = shop_edited;
            (shop_edited) -> shop_next = NULL;
        }
        else
        {
            while (shop_current != NULL)
            {
                if (shop_current -> shop_id > shop_id)    /*插在当前节点的前面*/
                {
                    if (shop_current == site_current -> shop_head) /*如果是第一个节点*/
                    {
                        site_current -> shop_head = shop_edited;
                        shop_edited -> shop_next = shop_current;
                    }
                    else
                    {
```

```
                    shop_prior -> shop_next = shop_edited;
                    shop_edited -> shop_next = shop_current;
                }
                break;
            }
            shop_prior = shop_current;
            shop_current = shop_current -> shop_next;
        }

        /*如果须插在最后节点的后面，必须再进行一次判断*/
        if (shop_current == NULL && shop_prior -> shop_id < shop_id)
        {
            shop_prior -> shop_next = shop_edited;
            shop_edited -> shop_next = NULL;
        }

    }
    shop_current = shop_edited;
    UpdateSiteCList(site_current, site_id);
  }
  else /*未修改所属网站编号*/
  {
    g_message("shop_edited == NULL");
    while (shop_current != NULL)
    {
        if (shop_current -> shop_id == shop_id)
            break;
        shop_current = shop_current -> shop_next;
    }
  }

  /*修改对应数据*/
  shop_current -> site_id = atoi(shop_data[0]);
  strcpy(shop_current -> shop_name, shop_data[2]);
  strcpy(shop_current -> seller_name, shop_data[3]);
  strcpy(shop_current -> seller_addr, shop_data[4]);
  strcpy(shop_current -> deposit_bank, shop_data[5]);
}
ShopInfo *DelShop(gint selected_row, gchar *str_site_id, gchar *str_shop_id, gboolean
real_del)     /*删除店铺*/
{
  g_message("DelShop");
  gint site_id = atoi(str_site_id);
  gint shop_id = atoi(str_shop_id);
```

```
    g_message("site_id=%d", site_id);
    g_message("shop_id=%d", shop_id);
    SiteInfo *site_current = site_head;
    ShopInfo *shop_current = NULL, *shop_prior;
    DealInfo *deal_current;

    shop_amount--;

    site_current = GetSite(site_id);

    (site_current -> shop_amount)--;
    shop_prior = shop_current = site_current -> shop_head;

    if (shop_current -> shop_id < shop_id)          /*如果第一个节点的shop_id小于欲删除的商
家的shop_id,就继续往后找*/
        do
        {
            shop_prior = shop_current;
            shop_current = shop_current -> shop_next;
        } while (shop_current -> shop_id < shop_id);

    if (shop_current == site_current -> shop_head)
        site_current -> shop_head = shop_current -> shop_next;
    else
        shop_prior -> shop_next = shop_current -> shop_next ;

    UpdateSiteCList(site_current, site_id);

    if (real_del) /*如果要彻底删除,刚会修改后面商家编号并释放相应内存*/
    {
        gtk_clist_remove(GTK_CLIST(clist[1]), selected_row);
        deal_amount -= shop_current -> deal_amount;

        deal_current = shop_current -> deal_head;
        while (deal_current != NULL)
        {
            free(deal_current);
            deal_current = deal_current -> deal_next;
        }
        free(shop_current);
        UpdateDealData();

        /*修改后面商家的编号*/
        site_current = site_head;
```

```
        while (site_current != NULL)
        {
            shop_current = site_current -> shop_head;
            while (shop_current != NULL)
            {
                if (shop_current -> shop_id > shop_id)
                    (shop_current -> shop_id)--;
                shop_current = shop_current -> shop_next;
            }
            site_current = site_current -> site_next;
        }

        RedrawShopCList(site_id);
        RedrawDealCList(0, 0);

        if (selected_row > shop_amount - 1)
            selected_row = shop_amount - 1;
        gtk_clist_moveto(GTK_CLIST(clist[1]), selected_row, 0, 0.0, 0.0);
        gtk_clist_select_row(GTK_CLIST(clist[1]), selected_row, 0);
        return NULL;
    }
    else
        return shop_current;
}
void RedrawShopCList(gint site_id)/*重新构建ShopCList*/
{
    g_message("RedrawShopCList");
    gchar shop_data[SHOP_COLUMN][20];
    gchar *shop_data_pointer[SHOP_COLUMN];
    gint index = 1;
    SiteInfo *site_current = site_head;
    ShopInfo *shop_current = NULL;

    for (index = 0; index < SHOP_COLUMN; index++)
        shop_data_pointer[index] = shop_data[index];

    gtk_clist_clear(GTK_CLIST(clist[1]));     /*清空CList*/
    gtk_clist_freeze(GTK_CLIST(clist[1]));         /*冻结CList以防止屏幕闪烁*/

    if (expand_all_info)        /*如果是“展开全部”模式*/
    {
        while (site_current != NULL)
        {
            shop_current = site_current -> shop_head;
```

```
            while (shop_current != NULL)
            {
                sprintf(shop_data[0], "%d", shop_current -> site_id);
                sprintf(shop_data[1], "%d", shop_current -> shop_id);
                strcpy(shop_data[2], shop_current -> shop_name);
                strcpy(shop_data[3], shop_current -> seller_name);
                strcpy(shop_data[4], shop_current -> seller_addr);
                strcpy(shop_data[5], shop_current -> deposit_bank);
                sprintf(shop_data[6], "%d", shop_current -> deal_amount);

                gtk_clist_append(GTK_CLIST(clist[1]), shop_data_pointer);
                shop_current = shop_current -> shop_next;
            }
            site_current = site_current -> site_next;
        }
    }
    else     /*“收起其他”模式，只输出所属网站编号为site_id的商家*/
    {
        if (site_id <= 0)
            return;
        for (index = 1; index < site_id; index++)
            site_current = site_current -> site_next;
        shop_current = site_current -> shop_head;

        while (shop_current != NULL)
        {
            sprintf(shop_data[0], "%d", shop_current -> site_id);
            sprintf(shop_data[1], "%d", shop_current -> shop_id);
            strcpy(shop_data[2], shop_current -> shop_name);
            strcpy(shop_data[3], shop_current -> seller_name);
            strcpy(shop_data[4], shop_current -> seller_addr);
            strcpy(shop_data[5], shop_current -> deposit_bank);
            sprintf(shop_data[6], "%d", shop_current -> deal_amount);

            gtk_clist_append(GTK_CLIST(clist[1]), shop_data_pointer);
            shop_current = shop_current -> shop_next;
        }
    }

    gtk_clist_thaw(GTK_CLIST(clist[1]));
}
void UpdateSiteCList(SiteInfo *site_current,gint site_id) /*更新网站clist中site_id对应网站的店铺个数*/
{
```

```
  g_message("UpdateSiteCList");
  gint shop_count = 0, index, row = -1;
  gchar shop_amount_one_site[10], *temp;
  ShopInfo *shop_current = site_current -> shop_head;

  /*计数*/
  while (shop_current != NULL)
  {
       shop_count++;
       shop_current = shop_current -> shop_next;
  }
  site_current -> shop_amount = shop_count;
  sprintf(shop_amount_one_site, "%d", shop_count);

  g_message("shop_amount_one_site=%s", shop_amount_one_site);

  for (index = 0; index < site_amount; index++)
  {
       gtk_clist_get_text(GTK_CLIST(clist[0]), index, 0, &temp);
       if (site_id == atoi(temp))
            row = index;
  }
  if (row != -1)
       gtk_clist_set_text(GTK_CLIST(clist[0]), row, 3, shop_amount_one_site);
}
void UpdateDealData(void)     /*更新店铺下deal的shop_id信息*/
{
  gint count = 0;
  SiteInfo *site_current = site_head;
  ShopInfo *shop_current;
  DealInfo *deal_current;

  while (site_current != NULL)
  {
       shop_current = site_current -> shop_head;
       while (shop_current != NULL)
       {
            deal_current = shop_current -> deal_head;
            while (deal_current != NULL)
            {
                 count++;
                 deal_current -> deal_id = count;
                 deal_current -> shop_id = shop_current -> shop_id;
                 deal_current = deal_current -> deal_next;
```

```
            }
            shop_current = shop_current -> shop_next;
        }
        site_current = site_current -> site_next;
    }
}
/*****************************************************************************************/
/*********************************交易信息相关操作*********************************/
/*****************************************************************************************/
void UpdateShopCList(ShopInfo *shop_current,gint shop_id);              /*更新店铺clist中
shop_id对应的店铺的交易个数*/
void SearchAndEditDeal(DealInfo *deal_edited, gint selected_row, gchar (*deal_data)[20]);
  /*找到并修改交易*/

ShopInfo *GetShop(gint shop_id)         /*根据shop_id找到并返回对应的ShopInfo指针*/
{
  SiteInfo *site_current = site_head;
  ShopInfo *shop_current;
  gboolean result = FALSE;

  while (site_current != NULL && !result)
  {
       shop_current = site_current -> shop_head;
       while (shop_current != NULL)
       {
            if (shop_current -> shop_id == shop_id)
            {
                 result = TRUE;
                 break;
            }
            shop_current = shop_current -> shop_next;
       }
       site_current = site_current -> site_next;
  }

  if (!result)
       return NULL;
  return shop_current;
}
void AddDeal(gchar (*deal_data)[20])        /*增加交易*/
{
  gint shop_id = atoi(deal_data[0]);
  gint new_row, index;
```

```
gchar *deal_data_pointer[DEAL_COLUMN];
ShopInfo *shop_current = NULL;
DealInfo *deal_current = NULL, *new_deal;


shop_current = GetShop(shop_id);
(shop_current -> deal_amount)++;

deal_current = shop_current -> deal_head;

new_deal = (DealInfo *)malloc(sizeof(DealInfo));
new_deal -> shop_id = shop_id;
new_deal -> pay_type = atoi(deal_data[2]);
new_deal -> deal_money = atof(deal_data[3]);
strcpy(new_deal -> deal_date, deal_data[4]);
strcpy(new_deal -> customer_addr, deal_data[5]);

/*为交易编号*/
if (deal_current == NULL)
{
     deal_amount++;

     shop_current -> deal_head = new_deal;
     new_deal -> deal_next = NULL;
}
else
{
     deal_amount++;

     while (deal_current -> deal_next != NULL)
          deal_current = deal_current -> deal_next;
     deal_current -> deal_next = new_deal;
     new_deal -> deal_next = NULL;
}
new_deal -> deal_id = deal_amount;

g_message("AddDeal:deal_id=%d", new_deal -> deal_id);
sprintf(deal_data[1], "%d", new_deal -> deal_id);

for (index = 0; index < DEAL_COLUMN; index++)
     deal_data_pointer[index] = deal_data[index];

new_row = gtk_clist_append(GTK_CLIST(clist[2]), deal_data_pointer);
gtk_clist_moveto(GTK_CLIST(clist[2]), new_row, 0, 0.0, 0.0);
```

```
  gtk_clist_select_row(GTK_CLIST(clist[2]), new_row, 0);

  UpdateShopCList(shop_current, shop_id);
}
gboolean SearchDeal(gint search_key_id, gchar *search_key)          /*查找交易*/
{
  g_message("SearchShop");
  g_message("search_key=%s", search_key);
  g_message("deal_amount=%d", deal_amount);
  gint index, deal_row;
  gchar *temp;

  deal_row = GTK_CLIST(clist[2]) -> rows;

  for (index = 0; index < deal_row; index++)
  {
       gtk_clist_get_text(GTK_CLIST(clist[2]), index, search_key_id + 4, &temp);
       g_message("temp=%s", temp);
       if (!strcmp(search_key, temp))
       {
            g_message("find deal");
            gtk_clist_moveto(GTK_CLIST(clist[2]), index, search_key_id + 4, 0.0, 0.0);
            gtk_clist_select_row(GTK_CLIST(clist[2]), index, search_key_id + 4);
            return TRUE;
       }
  }

  ShowMessageBox("未找到任何结果!");
  return FALSE;
}
void EditDeal(gint selected_row, gchar *former_shop_id, gchar (*deal_data)[20])       /*
编辑交易*/
{
  g_message("EditDeal");
  DealInfo *deal_edited = NULL;

  if (strcmp(former_shop_id, deal_data[0])) /*如果修改了所属商家编号*/
  {
       deal_edited = DelDeal(selected_row, former_shop_id, deal_data[1], FALSE);

       SearchAndEditDeal(deal_edited, selected_row, deal_data);
       deal_amount++;

  }
```

```
  else
       SearchAndEditDeal(deal_edited, selected_row, deal_data);
}
void SearchAndEditDeal(DealInfo *deal_edited, gint selected_row, gchar (*deal_data)[20])
  /*找到修改交易信息*/
{
  g_message("SearchAndEditDeal");
  gint index;
  gint shop_id = atoi(deal_data[0]);
  gint deal_id = atoi(deal_data[1]);
  ShopInfo *shop_current;
  DealInfo *deal_current, *deal_prior;

  for (index = 0; index < DEAL_COLUMN; index++)
       gtk_clist_set_text(GTK_CLIST(clist[2]), selected_row, index, deal_data[index]);

  shop_current = GetShop(shop_id);

  deal_current = shop_current -> deal_head;

  if (deal_edited != NULL)    /*如果修改了所属商家编号*/
  {
       (shop_current -> deal_amount)++;
       if (deal_current == NULL)
       {
            g_message("deal_current == NULL");
            shop_current -> deal_head = deal_edited;
            deal_edited -> deal_next = NULL;
       }
       else
       {
            while (deal_current != NULL)
            {
                 if (deal_current -> deal_id > deal_id)         /*找到插入的地方*/
                 {
                      if (deal_current == shop_current -> deal_head) /*如果是第一个节点*/
                      {
                           shop_current -> deal_head = deal_edited;
                           deal_edited -> deal_next = deal_current;
                      }
                      else
                      {
                           deal_prior -> deal_next = deal_edited;
                           deal_edited -> deal_next = deal_current;
```

```
                    }
                    break;
                }
                deal_prior = deal_current;
                deal_current = deal_current -> deal_next;
            }
            if (deal_current == NULL && deal_prior -> deal_id < deal_id) /*插在最后节点的
后面*/
            {
                deal_prior -> deal_next = deal_edited;
                deal_edited -> deal_next = NULL;
            }
        }
        deal_current = deal_edited;
        UpdateShopCList(shop_current, shop_id);
    }
    else
    {
        if (deal_current -> deal_id < deal_id)         /*如果第一个节点的deal_id小于欲修改
的交易的deal_id,就继续往后找*/
            do
            {
                deal_current = deal_current -> deal_next;
            } while (deal_current -> deal_id < deal_id && deal_current != NULL);
    }

    deal_current -> shop_id = atoi(deal_data[0]);
    deal_current -> pay_type = atoi(deal_data[2]);
    deal_current -> deal_money = atoi(deal_data[3]);
    strcpy(deal_current -> deal_date, deal_data[4]);
    strcpy(deal_current -> customer_addr, deal_data[5]);
  }
  DealInfo *DelDeal(gint selected_row, gchar *str_shop_id, gchar *str_deal_id,
        gboolean real_del)     /*删除交易*/
  {
    g_message("DelShop");
    gint site_id = 0;
    gint shop_id = atoi(str_shop_id);
    gint deal_id = atoi(str_deal_id);
    g_message("shop_id=%d", shop_id);
    g_message("deal_id=%d", deal_id);
    SiteInfo *site_current;
    ShopInfo *shop_current = NULL;
    DealInfo *deal_current = NULL, *deal_prior;
```

```
    deal_amount--;

    shop_current = GetShop(shop_id);
    (shop_current -> deal_amount)--;

    deal_prior = deal_current = shop_current -> deal_head;
    if (deal_current -> deal_id < deal_id)          /*如果第一个节点的deal_id小于欲删除的交
易的deal_id,就继续往后找*/
        do
        {
            deal_prior = deal_current;
            deal_current = deal_current -> deal_next;
        } while (deal_current -> deal_id < deal_id);

    if (deal_current == shop_current -> deal_head)
        shop_current -> deal_head = deal_current -> deal_next;
    else
        deal_prior -> deal_next = deal_current -> deal_next ;

    UpdateShopCList(shop_current, shop_id);

    if (real_del) /*如果要彻底删除,刚会修改后面交易编号并释放相应内存*/
    {
        gtk_clist_remove(GTK_CLIST(clist[2]), selected_row);

        free(deal_current);
        site_current = site_head;
        /*修改后面交易的编号*/
        while (site_current != NULL)
        {
            g_message("site_current != NULL");
            shop_current = site_current -> shop_head;
            while (shop_current != NULL)
            {
                g_message("shop_current != NULL");
                deal_current = shop_current -> deal_head;
                while (deal_current != NULL)
                {
                    if (deal_current -> deal_id > deal_id)
                        (deal_current -> deal_id)--;
                    deal_current = deal_current -> deal_next;
                }
                shop_current = shop_current -> shop_next;
```

```
            }
            site_current = site_current -> site_next;
        }

        RedrawDealCList(site_id, shop_id);
        if (selected_row > deal_amount - 1)
            selected_row = deal_amount - 1;
        gtk_clist_moveto(GTK_CLIST(clist[2]), selected_row, 0, 0.0, 0.0);
        gtk_clist_select_row(GTK_CLIST(clist[2]), selected_row, 0);
        return NULL;
    }
    else
        return deal_current;
}
void RedrawDealCList(gint site_id, gint shop_id)/*重新构建交易页面的clist*/
{
    g_message("RedrawDealCList");
    gchar deal_data[DEAL_COLUMN][20];
    gchar *deal_data_pointer[DEAL_COLUMN];
    gint index = 1;
    SiteInfo *site_current = site_head;
    ShopInfo *shop_current = NULL;
    DealInfo *deal_current = NULL;

    for (index = 0; index < DEAL_COLUMN; index++)
        deal_data_pointer[index] = deal_data[index];

    gtk_clist_clear(GTK_CLIST(clist[2]));
    gtk_clist_freeze(GTK_CLIST(clist[2]));

    if (expand_all_info)        /*"展开全部"模式*/
    {
        while (site_current != NULL)
        {
            shop_current = site_current -> shop_head;
            while (shop_current != NULL)
            {
                deal_current = shop_current -> deal_head;
                while (deal_current != NULL)
                {
                    sprintf(deal_data[0], "%d", deal_current -> shop_id);
                    sprintf(deal_data[1], "%d", deal_current -> deal_id);
                    sprintf(deal_data[2], "%d", deal_current -> pay_type);
                    sprintf(deal_data[3], "%.2f", deal_current -> deal_money);
```

```
                    strcpy(deal_data[4], deal_current -> deal_date);
                    strcpy(deal_data[5], deal_current -> customer_addr);

                    gtk_clist_append(GTK_CLIST(clist[2]), deal_data_pointer);
                    deal_current = deal_current -> deal_next;
               }
               shop_current = shop_current -> shop_next;

          }
          site_current = site_current -> site_next;
     }
}
else     /*"收起其他"模式*/
{
     if (site_id <= 0 || shop_id <= 0)
          return;
     for (index = 1; index < site_id; index++)
          site_current = site_current -> site_next;
     shop_current = site_current -> shop_head;

     while (shop_current != NULL)
     {
          if (shop_current -> shop_id == shop_id)
          {
               deal_current = shop_current -> deal_head;
               while (deal_current != NULL)
               {
                    sprintf(deal_data[0], "%d", deal_current -> shop_id);
                    sprintf(deal_data[1], "%d", deal_current -> deal_id);
                    sprintf(deal_data[2], "%d", deal_current -> pay_type);
                    sprintf(deal_data[3], "%.2f", deal_current -> deal_money);
                    strcpy(deal_data[4], deal_current -> deal_date);
                    strcpy(deal_data[5], deal_current -> customer_addr);

                    gtk_clist_append(GTK_CLIST(clist[2]), deal_data_pointer);
                    deal_current = deal_current -> deal_next;
               }
               break;
          }
          shop_current = shop_current -> shop_next;
     }
}

gtk_clist_thaw(GTK_CLIST(clist[2]));
```

```
}
void UpdateShopCList(ShopInfo *shop_current,gint shop_id) /*更新店铺clist中shop_id对应的店铺的交易个数*/
{
  g_message("UpdateShopCList");
  gint deal_count = 0, index, row;
  gchar deal_amount_one_shop[10], *temp;
  DealInfo *deal_current = shop_current -> deal_head;

  while (deal_current != NULL)
  {
       deal_count++;
       deal_current = deal_current -> deal_next;
  }
  shop_current -> deal_amount = deal_count;
  sprintf(deal_amount_one_shop, "%d", deal_count);

  g_message("deal_amount_one_shop=%s",deal_amount_one_shop);
  for (index = 0; index < shop_amount; index++)
  {
       gtk_clist_get_text(GTK_CLIST(clist[1]), index, 1, &temp);
       if (shop_id == atoi(temp))
            row = index;
  }
  if (row != -1)
       gtk_clist_set_text(GTK_CLIST(clist[1]), row, 6, deal_amount_one_shop);
}
/*****************************************************************************************/
/*************************************文件相关操作*************************************/
/*****************************************************************************************/
gint RealOpen(gchar *site_file_name)        /*读取文件到内存中*/
{
  g_message("RealOpen");
  FILE *site_file, *shop_file, *deal_file;       /*site_file存储site链表的信息，shop_file存储shop链表的信息，deal_file存储deal链表的信息*/
  gchar shop_file_name[55], deal_file_name[55];
  SiteInfo *site_new, *site_current;
  ShopInfo *shop_new, *shop_current;
  DealInfo *deal_new, *deal_current;
  gint index, index2;

  /*将site_file_name后面接上"_shop"作为shop_file的文件名字，接上"_deal"作为deal_file的文件名字*/
  strcpy(shop_file_name, site_file_name);
```

```
    strcpy(deal_file_name, site_file_name);
    strcat(shop_file_name, "_shop");
    strcat(deal_file_name, "_deal");
    g_message("site_file_name=%s\n\tshop_file_name=%s\n\tdeal_file_name=%s", site_file_name,
shop_file_name, deal_file_name);

    site_file = fopen(site_file_name, "r");
    shop_file = fopen(shop_file_name, "r");
    deal_file = fopen(deal_file_name, "r");

    /*如果三个文件有一个打开失败，即退出函数并提示打开失败*/
    if (site_file  == NULL || shop_file == NULL || deal_file == NULL)
    {
        ShowMessageBox("打开文件失败!不支持的数据文件！");
        return 0;
    }

    site_head = NULL;
    expand_all_info = TRUE;
    site_amount = shop_amount = deal_amount = 0;

    while (!feof(site_file))
    {
        site_new = (SiteInfo *)malloc(sizeof(SiteInfo));
        fread(site_new, sizeof(SiteInfo), 1, site_file);
        if(!feof(site_file))
        {
            site_amount++;
            if (site_head == NULL)
                site_current = site_head = site_new;
            else
            {
                site_current -> site_next = site_new;
                site_current = site_new;
            }
            site_current -> shop_head = NULL;
            site_current -> site_next = NULL;

            g_message("site_current -> shop_amount=%d", site_current -> shop_amount);
            for (index = 0; index < (site_current -> shop_amount); index++)
            /*根据site节点里面的成员shop_amount决定读取多少个shop节点*/
            {
                shop_amount++;
                shop_new = (ShopInfo *)malloc(sizeof(ShopInfo));
```

```
                fread(shop_new, sizeof(ShopInfo), 1, shop_file);
                if (site_current -> shop_head == NULL)
                     shop_current = site_current -> shop_head = shop_new;
                else
                {
                     shop_current -> shop_next = shop_new;
                     shop_current = shop_new;
                }
                shop_current -> deal_head = NULL;
                shop_current -> shop_next = NULL;

                g_message("shop_current -> deal_amount=%d", shop_current -> deal_amount);
                /*根据shop节点里面的成员deal_amount决定读取多少个deal节点*/
                for (index2 = 0; index2 < (shop_current -> deal_amount); index2++)
                {
                     deal_amount++;
                     deal_new = (DealInfo *)malloc(sizeof(DealInfo));
                     fread(deal_new, sizeof(DealInfo), 1, deal_file);
                     if (shop_current -> deal_head == NULL)
                          deal_current = shop_current -> deal_head = deal_new;

                     else
                     {
                          deal_current -> deal_next = deal_new;
                          deal_current = deal_new;
                     }
                     deal_current -> deal_next = NULL;
                }
           }
      }
  }

  RedrawAllCList(); /*重构所有CList*/
  fclose(site_file);
  fclose(shop_file);
  fclose(deal_file);

  return 1;
}
gint RealSave(gchar *site_file_name)        /*将数据写入硬盘*/
{
  g_message("RealSave");
  FILE *site_file, *shop_file, *deal_file;
  gchar shop_file_name[55], deal_file_name[55];
```

```
    SiteInfo *site_current = site_head;
    ShopInfo *shop_current;
    DealInfo *deal_current;

    strcpy(shop_file_name, site_file_name);
    strcpy(deal_file_name, site_file_name);
    strcat(shop_file_name, "_shop");
    strcat(deal_file_name, "_deal");
    g_message("site_file_name=%s\n\tshop_file_name=%s\n\tdeal_file_name=%s", site_file_name,
shop_file_name, deal_file_name);

    site_file = fopen(site_file_name, "w");
    shop_file = fopen(shop_file_name, "w");
    deal_file = fopen(deal_file_name, "w");

    if (site_file  == NULL || shop_file == NULL || deal_file == NULL)
    {
        ShowMessageBox("创建文件失败!");
        return 0;
    }

    while (site_current != NULL)
    {
        fwrite(site_current, sizeof(SiteInfo), 1, site_file);
        shop_current = site_current -> shop_head;
        while (shop_current != NULL)
        {
            fwrite(shop_current, sizeof(ShopInfo), 1, shop_file);
            deal_current = shop_current -> deal_head;
            while (deal_current != NULL)
            {
                fwrite(deal_current, sizeof(DealInfo), 1, deal_file);
                deal_current = deal_current -> deal_next;
            }
            shop_current = shop_current -> shop_next;
        }
        site_current = site_current -> site_next;
    }

    fclose(site_file);
    fclose(shop_file);
    fclose(deal_file);

    return 1;
```

```
}

void RedrawAllCList(void)          /*重新构建所有页面的clist*/
{
  g_message("RedrawAllCList");
  SiteInfo *site_current = site_head;
  ShopInfo *shop_current;
  DealInfo *deal_current;
  gint index;
  gchar site_data[SITE_COLUMN][50];
  gchar shop_data[SHOP_COLUMN][20];
  gchar deal_data[DEAL_COLUMN][20];
  gchar *site_data_pointer[SITE_COLUMN];
  gchar *shop_data_pointer[SHOP_COLUMN];
  gchar *deal_data_pointer[DEAL_COLUMN];

  for (index = 0; index < 3; index++)
  {
       gtk_clist_clear(GTK_CLIST(clist[index]));
       gtk_clist_freeze(GTK_CLIST(clist[index]));
  }

  /*初始化三个指针数组*/
  for (index = 0; index < SITE_COLUMN; index++)
       site_data_pointer[index] = site_data[index];
  for (index = 0; index < SHOP_COLUMN; index++)
       shop_data_pointer[index] = shop_data[index];
  for (index = 0; index < DEAL_COLUMN; index++)
       deal_data_pointer[index] = deal_data[index];

  while (site_current != NULL)
  {
       g_message("site_current != NULL");
       /*增加一条site信息*/
       sprintf(site_data[0], "%d", site_current -> site_id);
       strcpy(site_data[1], site_current -> site_name);
       strcpy(site_data[2], site_current -> site_addr);
       sprintf(site_data[3],"%d",site_current -> shop_amount);
       gtk_clist_append(GTK_CLIST(clist[0]), site_data_pointer);

       shop_current = site_current -> shop_head;
       while (shop_current != NULL)
       {
            g_message("shop_current != NULL");
```

```
            /*增加一条shop信息*/
            sprintf(shop_data[0], "%d", shop_current -> site_id);
            sprintf(shop_data[1], "%d", shop_current -> shop_id);
            strcpy(shop_data[2], shop_current -> shop_name);
            strcpy(shop_data[3], shop_current -> seller_name);
            strcpy(shop_data[4], shop_current -> seller_addr);
            strcpy(shop_data[5], shop_current -> deposit_bank);
            sprintf(shop_data[6], "%d", shop_current -> deal_amount);

            gtk_clist_append(GTK_CLIST(clist[1]), shop_data_pointer);

            deal_current = shop_current -> deal_head;
            while (deal_current != NULL)
            {
                g_message("deal_current != NULL");
                /*增加一条deal信息*/
                sprintf(deal_data[0], "%d", deal_current -> shop_id);
                sprintf(deal_data[1], "%d", deal_current -> deal_id);
                sprintf(deal_data[2], "%d", deal_current -> pay_type);
                sprintf(deal_data[3], "%.2f", deal_current -> deal_money);
                strcpy(deal_data[4], deal_current -> deal_date);
                strcpy(deal_data[5], deal_current -> customer_addr);

                gtk_clist_append(GTK_CLIST(clist[2]), deal_data_pointer);
                deal_current = deal_current -> deal_next;
            }
            shop_current = shop_current -> shop_next;

        }
        site_current = site_current -> site_next;
    }

    for (index = 0; index < 3; index++)
    {
        gtk_clist_thaw(GTK_CLIST(clist[index]));
    }

    gtk_clist_moveto(GTK_CLIST(clist[0]), 0, 0, 0.0, 0.0);
    gtk_clist_select_row(GTK_CLIST(clist[0]), 0, 0);

}

/*****************************************************************************/
/*************************************统计模块*********v**************************/
```

```
/*********************************************************************************************/
#define STATI_SITE 0
#define STATI_SHOP 1
#define STATI_DEAL 2

void ShowStatiSiteResult(gchar (*stati_result)[5][20], gchar *date);    /*输出统计网站后的结果*/
void JudgeStatiDealResult(gchar *stati_result, gfloat balance);              /*根据统计后结果判断贸易关系(顺差或逆差)*/

void StatiSite(gchar *date)        /*统计：网站排行榜*/
{
  gchar date_temp[7];
  gint deal_amount_one_site[site_amount];        /*记录每个网站下的交易次数*/
  gint site_id[site_amount];                     /*记录网站编号，与上个数组的数据同步，即元素编号一致的代表同个网站*/
  gint index, index2, temp;
  gchar stati_result[5][5][20];                  /*记录统计结果，即5个商家的所有相关数据*/
  SiteInfo *site_current = site_head;
  ShopInfo *shop_current;
  DealInfo *deal_current;

  for (index = 0; index < site_amount; index++)
  {
      deal_amount_one_site[index] = 0;
      site_id[index] = index + 1;
  }

  index = 0;
  while (site_current != NULL)
  {
      shop_current = site_current -> shop_head;
      while (shop_current != NULL)
      {
          deal_current = shop_current -> deal_head;
          while (deal_current != NULL)
          {
              strncpy(date_temp, deal_current -> deal_date, 6);  /*截取日期的前6位作为月份*/
              date_temp[6] = '\0';

              if (!strcmp(date_temp, date))
                  deal_amount_one_site[index]++;
```

```
                deal_current = deal_current -> deal_next;
            }
            shop_current = shop_current -> shop_next;
        }
        site_current = site_current -> site_next;
        index++;
    }

    /*排序，同时更新数组site_id使之与数组deal_amount_one_site同步*/
    for (index = 0; index < site_amount; index++)
    {
        for (index2 = index + 1; index2 < site_amount; index2++)
        {
            if (deal_amount_one_site[index] < deal_amount_one_site[index2])
            {
                temp = deal_amount_one_site[index];
                deal_amount_one_site[index] = deal_amount_one_site[index2];
                deal_amount_one_site[index2] = temp;

                temp = site_id[index];
                site_id[index] = site_id[index2];
                site_id[index2] = temp;
            }
        }
    }

    /*存储统计结果*/
    for (index = 0; index < site_amount && index < 5; index++)
    {
        site_current = GetSite(site_id[index]);
        sprintf(stati_result[index][0], "%d", index + 1);
        sprintf(stati_result[index][1], "%d", site_id[index]);
        strcpy(stati_result[index][2], site_current -> site_name);
        strcpy(stati_result[index][3], date);
        sprintf(stati_result[index][4], "%d", deal_amount_one_site[index]);
    }
    ShowStatiSiteResult(stati_result, date);
}
void StatiShop(gchar *year)        /*统计：店铺交易情况*/
{
    g_message("StatiShop:year=%s", year);
    gchar date_temp[6];
    gint deal_amount_one_shop = 0, index;
    gfloat deal_money_amount = 0;
```

```
    gchar stati_result[5][20];
    gchar *stati_result_pointer[5];
    SiteInfo *site_current = site_head;
    ShopInfo *shop_current;
    DealInfo *deal_current;

    CreateStatiNotebook(STATI_SHOP);
    for (index = 0; index < 5; index++)
        stati_result_pointer[index] =  stati_result[index];

    while (site_current != NULL)
    {
        shop_current = site_current -> shop_head;
        while (shop_current != NULL)
        {
            deal_current = shop_current -> deal_head;
            while (deal_current != NULL)
            {
                strncpy(date_temp, deal_current -> deal_date, 4);  /*截取日期前4位作为年
份*/
                date_temp[4] = '\0';

                if (!strcmp(date_temp, year))
                {
                    deal_amount_one_shop++;
                    deal_money_amount += deal_current -> deal_money;
                }

                deal_current = deal_current -> deal_next;
            }
            sprintf(stati_result[0], "%d", shop_current -> shop_id);
            strcpy(stati_result[1], shop_current -> shop_name);
            strcpy(stati_result[2], year);
            sprintf(stati_result[3], "%d", deal_amount_one_shop);
            sprintf(stati_result[4], "%.2f", deal_money_amount);

            gtk_clist_append(GTK_CLIST(clist[3]), stati_result_pointer);

            deal_amount_one_shop = 0;
            deal_money_amount = 0;
            shop_current = shop_current -> shop_next;
        }
        site_current = site_current -> site_next;
    }
```

```
}
void StatiDeal(gchar (*stati_data)[20])
{
  gchar stati_result[5][20];
  gchar *stati_result_pointer[5];
  gfloat area_deal_money[2] = {0, 0};  /*第一个元素记录地区1资金流入金额，第二个记录地区1
资金流出金额*/
  gchar date_temp[7];
  SiteInfo *site_current = site_head;
  ShopInfo *shop_current;
  DealInfo *deal_current;
  gboolean area1_exist = FALSE, area2_exist = FALSE;      /*标识是否存在地区1、2的信息*/
  gint index;

  for (index = 0; index < 5; index++)
       stati_result_pointer[index] =  stati_result[index];

  while (site_current != NULL)
  {
       shop_current = site_current -> shop_head;
       while (shop_current != NULL)
       {
            /*如果找到地区1的商家*/
            if (!strcmp((gchar *)stati_data, shop_current -> seller_addr))
            {
                 deal_current = shop_current -> deal_head;
                 while (deal_current != NULL)
                 {
                      strncpy(date_temp, deal_current -> deal_date, 6);
                      date_temp[6] = '\0';
                      if (!strcmp((gchar *)stati_data[1], deal_current -> customer_addr)
                           && !strcmp((gchar *)stati_data[2], date_temp))
                      {
                           /*地区2的买家*/
                           area_deal_money[0] += deal_current -> deal_money;
                           area2_exist = TRUE;
                      }
                      deal_current = deal_current -> deal_next;
                 }
                 area1_exist = TRUE;
            }
            else if (!strcmp((gchar *)stati_data[1], shop_current -> seller_addr))
            /*如果找到地区2的商家*/
            {
```

```
                deal_current = shop_current -> deal_head;
                while (deal_current != NULL)
                {
                     strncpy(date_temp, deal_current -> deal_date, 6);
                     date_temp[6] = '\0';
                     if (!strcmp((gchar *)stati_data, deal_current -> customer_addr)
                          && !strcmp((gchar *)stati_data[2], date_temp))
                     {
                          /*地区1的买家*/
                          area1_exist = TRUE;
                          area_deal_money[1] += deal_current -> deal_money;
                     }

                     deal_current = deal_current -> deal_next;
                }
                area2_exist = TRUE;
           }

           shop_current = shop_current -> shop_next;
      }
      site_current = site_current -> site_next;
}

if (!area1_exist || !area2_exist)
{
     ShowMessageBox("地区1、2之间在指定月份无贸易关系！");
     return;
}

CreateStatiNotebook(STATI_DEAL);

/*输出地区1的情况*/
strcpy(stati_result[0], stati_data[0]);
strcpy(stati_result[1], stati_data[2]);
sprintf(stati_result[2], "%.2f", area_deal_money[0]);
sprintf(stati_result[3], "%.2f", area_deal_money[1]);
JudgeStatiDealResult(stati_result[4], area_deal_money[0]-area_deal_money[1]);
gtk_clist_append(GTK_CLIST(clist[3]), stati_result_pointer);

/*输出地区2的情况*/
strcpy(stati_result[0], stati_data[1]);
sprintf(stati_result[2], "%.2f", area_deal_money[1]);
sprintf(stati_result[3], "%.2f", area_deal_money[0]);
JudgeStatiDealResult(stati_result[4], area_deal_money[1]-area_deal_money[0]);
```

```
    gtk_clist_append(GTK_CLIST(clist[3]), stati_result_pointer);
  }

  void ShowStatiSiteResult(gchar (*stati_result)[5][20], gchar *date)          /*输出统计网
站后的结果*/
  {
    gchar *stati_result_pointer[5];
    gint index, index2;

    CreateStatiNotebook(STATI_SITE);
    for (index = 0; index < 5 && index < site_amount; index++)
    {
        for (index2 = 0; index2 < 5; index2++)
            stati_result_pointer[index2] = *(*(stati_result + index) + index2);

        gtk_clist_append(GTK_CLIST(clist[3]), stati_result_pointer);
    }
  }
  void JudgeStatiDealResult(gchar *stati_result, gfloat balance)     /*根据统计后结果判断贸
易关系(顺差或逆差)*/
  {
    if (balance == 0)
        strcpy(stati_result, "平衡");
    else if (balance > 0)
        sprintf(stati_result, "顺差: %.2f", balance);
    else
        sprintf(stati_result, "逆差: %.2f", balance);
  }
  /*****************************************************************************************/
  /*************************************其他函数*************************************/
  /*****************************************************************************************/
  void ClearList(void)              /*删除整个十字交叉链表*/
  {
    g_message("ClearList");
    SiteInfo *site_current = site_head;
    ShopInfo *shop_current;
    DealInfo *deal_current;

    while (site_current != NULL)
    {
        shop_current = site_current -> shop_head;
        while (shop_current != NULL)
        {
            deal_current = shop_current -> deal_head;
```

```
            while (deal_current != NULL)
            {
                free(deal_current);

                g_message("deal_current != NULL");
                deal_current = deal_current -> deal_next;
            }
            free(shop_current);
            g_message("shop_current != NULL");
            shop_current = shop_current -> shop_next;
        }
        free(site_current);
        g_message("site_current != NULL");
        site_current = site_current -> site_next;
    }
    site_head = NULL;
    site_amount = 0;
    shop_amount = 0;
    deal_amount = 0;

}
void ChangeShowMode(gboolean expand_all)         /*修改输出模式*/
{
    expand_all_info = expand_all;
}
void ShowDealOfSite(SiteInfo *site_current)      /*输出某个网站下面的所有交易信息*/
{
    gchar deal_data[DEAL_COLUMN][20];
    gchar *deal_data_pointer[DEAL_COLUMN];
    gint index;
    ShopInfo *shop_current;
    DealInfo *deal_current;

    if (site_current == NULL)
        return ;
    for (index = 0; index < DEAL_COLUMN; index++)
        deal_data_pointer[index] = deal_data[index];

    gtk_clist_clear(GTK_CLIST(clist[2]));
    gtk_clist_freeze(GTK_CLIST(clist[2]));

    shop_current = site_current -> shop_head;
    while (shop_current != NULL)
    {
```

```
        deal_current = shop_current -> deal_head;
        while (deal_current != NULL)
        {
            sprintf(deal_data[0], "%d", deal_current -> shop_id);
            sprintf(deal_data[1], "%d", deal_current -> deal_id);
            sprintf(deal_data[2], "%d", deal_current -> pay_type);
            sprintf(deal_data[3], "%.2f", deal_current -> deal_money);
            strcpy(deal_data[4], deal_current -> deal_date);
            strcpy(deal_data[5], deal_current -> customer_addr);

            gtk_clist_append(GTK_CLIST(clist[2]), deal_data_pointer);
            deal_current = deal_current -> deal_next;
        }
        shop_current = shop_current -> shop_next;
    }
    gtk_clist_thaw(GTK_CLIST(clist[2]));

    gtk_clist_moveto(GTK_CLIST(clist[2]), 0, 0, 0.0, 0.0);
}
void RealCheckData(void) /*数据排错：更新所有编号，以及网站下店铺个数，店铺下交易次数*/
{
    g_message("CheckData");
    SiteInfo *site_current = site_head;
    ShopInfo *shop_current;
    DealInfo *deal_current;
    gint site_count = 0, shop_count = 0, deal_count = 0;
    gint shop_amount_one_site, deal_amount_one_shop;

    while (site_current != NULL)
    {
        site_count++;
        site_current -> site_id = site_count;

        shop_current = site_current -> shop_head;
        shop_amount_one_site = 0;
        while (shop_current != NULL)
        {
            shop_count++;
            shop_current -> shop_id = shop_count;
            shop_current -> site_id = site_count;

            deal_current = shop_current -> deal_head;
            deal_amount_one_shop = 0;
            while (deal_current != NULL)
```

```
            {
                deal_count++;
                deal_current -> deal_id = deal_count;
                deal_current -> shop_id = shop_count;
                deal_amount_one_shop++;
                deal_current = deal_current -> deal_next;
            }

            shop_amount_one_site++;
            shop_current -> deal_amount = deal_amount_one_shop;
            shop_current = shop_current -> shop_next;
        }
        site_current -> shop_amount = shop_amount_one_site;
        site_current = site_current -> site_next;
    }

    site_amount = site_count;
    shop_amount = shop_count;
    deal_amount = deal_count;

    RedrawAllCList();
}
```

vii. handle_data.h

```
#ifndef __HANDLE_DATA_H__
#define __HANDLE_DATA_H__

typedef struct deal_information {
    gint shop_id;                       /*所属店铺编号*/
    gint deal_id;                       /*交易编号*/
    gint pay_type;                      /*支付类型*/
    gfloat deal_money;                  /*交易金额*/
    gchar deal_date[20];                /*交易日期*/
    gchar customer_addr[20];            /*客户所在地*/
    struct deal_information *deal_next; /*指向下个节点*/
} DealInfo;
typedef struct shop_information {
    gint site_id;                       /*所属网站编号*/
    gint shop_id;                       /*店铺编号*/
    gchar shop_name[20];                /*店铺名字*/
    gchar seller_name[10];              /*负责人名字*/
    gchar seller_addr[20];              /*店铺名字*/
    gchar deposit_bank[20];             /*开户银行*/
```

```
  gint deal_amount;                    /*交易次数*/
  struct shop_information *shop_next;  /*指向下个节点*/
  DealInfo *deal_head;                 /*指向交易链*/
} ShopInfo;
typedef struct site_information {
  gint site_id;                        /*网站编号*/
  gchar site_name[20];                 /*网站名称*/
  gchar site_addr[50];                 /*网站地址*/
  gint shop_amount;                    /*店铺总数*/
  struct site_information *site_next;  /*指向下个节点*/
  ShopInfo *shop_head;                 /*指向店铺链*/
} SiteInfo;


void AddSite(gchar (*site_data)[50]);                                  /*添加网站*/
gboolean SearchSite(gint search_key_id, gchar *search_key);            /*查找网站*/
void EditSite(gint selected_row, gchar (*site_data)[50]);              /*编辑网站*/
void DelSite(gint selected_row, gchar *site_id);                       /*删除网站*/

void AddShop(gchar (*shop_data)[20]);                                  /*增加店铺*/

gboolean SearchShop(gint search_key_id, gchar *search_key);            /*查找店铺*/
void EditShop(gint selected_row, gchar *former_site_id,
      gchar (*shop_data)[20]);                                         /*编辑店铺*/
ShopInfo *DelShop(gint selected_row, gchar *str_site_id,
      gchar *str_shop_id, gboolean real_del);                          /*删除店铺*/

void AddDeal(gchar (*deal_data)[20]);                                  /*增加交易*/
gboolean SearchDeal(gint search_key_id, gchar *search_key);            /*查找交易*/
void EditDeal(gint selected_row, gchar *former_shop_id,
      gchar (*deal_data)[20]);                                         /*编辑交易*/
DealInfo *DelDeal(gint selected_row, gchar *str_shop_id,
      gchar *str_deal_id, gboolean real_del);                  /*删除交易*/

gint RealOpen(gchar *filename);                                        /*读取文件到内存中*/
gint RealSave(gchar *filename);                                        /*将数据写入硬盘*/

void StatiSite(gchar *date);                                           /*统计：网站排行榜*/
void StatiShop(gchar *year);                                           /*统计：店铺交易情况*/
void StatiDeal(gchar (*area_name)[20]);                                /*统计：地区贸易关系*/

SiteInfo *GetSite(gint site_id);           /*根据site_id查找并返回对应的SiteInfo指针*/
ShopInfo *GetShop(gint shop_id);           /*根据shop_id查找并返回对应的ShopInfo指针*/
```

```
void RedrawShopCList(gint site_id);                          /*重新构建店铺页面的clist*/
void RedrawDealCList(gint site_id, gint shop_id);            /*重新构建交易页面的clist*/
void RedrawAllCList(void);                                   /*重新构建所有页面的clist*/

void ClearList(void);                                        /*删除整个十字交叉链表*/

void ChangeShowMode(gboolean expand_all);                    /*修改输出模式*/
void ShowDealOfSite(SiteInfo *site_current);            /*输出某个网站下面的所有交易信息*/
void RealCheckData(void);/*数据排错：更新所有编号，以及网站下店铺个数，店铺下交易次数*/

#endif
```

viii. Makefile

```
CC = gcc
all:main.o create_window.o get_user_data.o handle_data.o
  $(CC) -Wall -o app main.o create_window.o get_user_data.o handle_data.o `pkg-config --libs
gtk+-2.0 libgnomeui-2.0`
main.o:main.c create_window.h
  $(CC) -Wall -c main.c `pkg-config --cflags --libs gtk+-2.0`
create_window.o:create_window.c get_user_data.h
  $(CC) -Wall -c create_window.c `pkg-config --cflags --libs gtk+-2.0 libgnomeui-2.0`
get_user_data.o:get_user_data.c create_window.h handle_data.h
  $(CC) -Wall -c get_user_data.c `pkg-config --cflags --libs gtk+-2.0 libgnomeui-2.0`
handle_data.o:handle_data.c create_window.h handle_data.h
  $(CC) -Wall -c handle_data.c `pkg-config --cflags --libs gtk+-2.0 libgnomeui-2.0`
clean:
  rm -f *.o
```