МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
имени Н.Э. БАУМАНА

Учебное пособие

Учебно-методический комплект

по дисциплине Инструментальные средства систем автоматизированного проектирования Методические указания по выполнению практических занятий

МГТУ имени Н.Э. Баумана

УДК 681.3.06(075.8)
ББК 32.973-018
И201

Методические указания по выполнению домашних заданий по единому комплексному заданию по блоку дисциплины «Инструментальные средства САПР» / Коллектив авторов –
М.: МГТУ им. Н.Э. Баумана, 2012. – XX с.: ил.

В методических указаниях рассмотрены основные этапы, их последовательность и содержание по выполнению домашних заданий по единому комплексному заданию по блоку дисциплин «Инструментальные средства САПР».

Ил. 39. Табл. 5. Библиогр. 7 назв.

УДК 681.3.06(075.8)

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

При разработке информационной модели будут выявлены следующие сущности: каталоги (dir), компоненты (part), поставщики (sup).На их основе разработан словарь сущностей и атрибутов.На следующем шаге была разработана логическая, а затем физическая модель БД.Далее на основе физической модели БД были получены SQL-скрипты для СУБД SQLServer.

1 РАЗРАБОТКА БАЗЫ ДАННЫХ

Для создания системы учета радиоэлементов или любой другой системы необходимо разработать логическую модель базы данных, которая будет отражать зависимости между сущностями по определенным атрибутам, которые мы ей зададим [1].

1.1 Логическая информационная модель базы данных

Для создания системы учета радиоэлементов была разработана логическая модель базы данных, которая отражает зависимости между сущностями по определенным атрибутам.
Структура логической модели представлена на рис. 1.

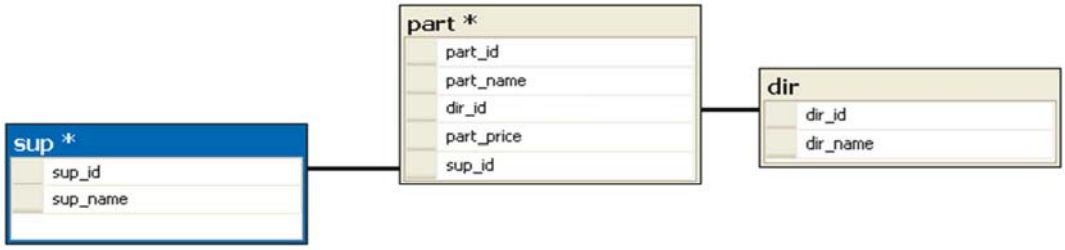

Рисунок 1 – Логическая информационная модель

Логический уровень модели данных является универсальным и никак не связан с конкретной реализацией СУБД.

1.2 Разработка физической информационной моделиБД

В процессе создания физической информационной модели каждому атрибуту сущности был присвоен определенный тип данных в соответствии с возможностями выбранной СУБД SQLServer. Результатом работы является физическая информационная модель, представленная на рис. 2.

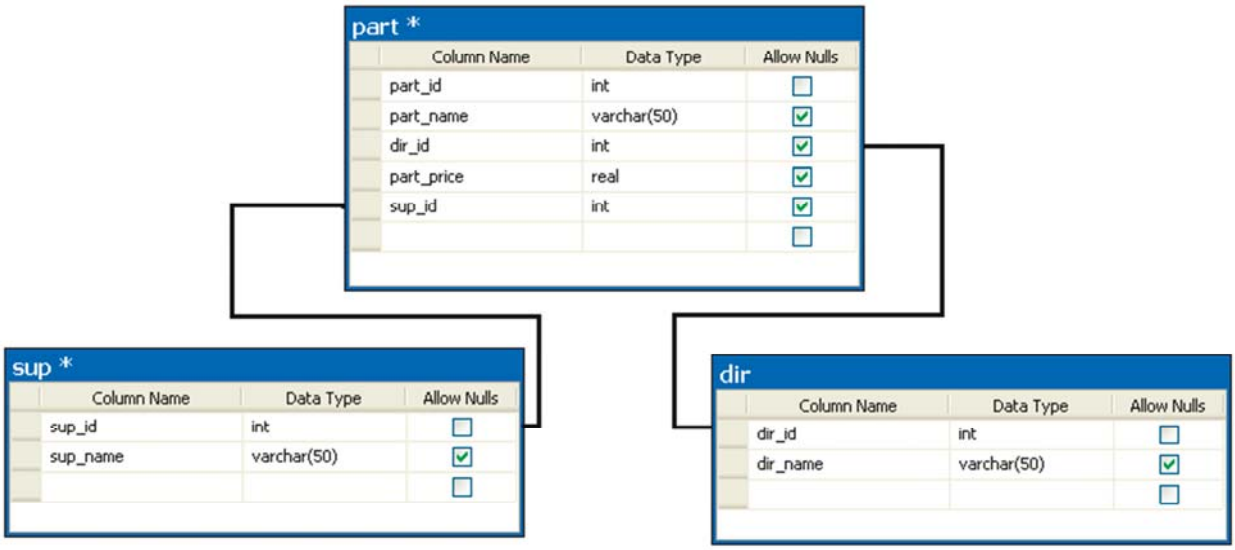

Рисунок 2 - Физическая информационная модель

Спецификация физической информационной модели представлена в табл. 1.

Таблица 1 – Спецификация физической модели

Наименование атрибута	**Таблица**	**Тип данных**	**Nullопция**
sup_id	sup	integer	NOT NULL
sup_name	sup	varchar(50)	NULL
dir_id	dir	integer	NOT NULL
dir_name	dir	varchar(50)	NULL
part_id	part	integer	NOT NULL
part_name	part	varchar(50)	NULL
dir_id	part	integer	NULL
part_price	part	real	NULL
sup_id	part	integer	NULL

Разработанная физическая модель соответствует третьей нормальной форме, поскольку каждый ее атрибут атомарен и находится в прямой зависимости от соответствующего первичного ключа [2].

1.3 Инсталляционный комплект в виде SQL кода

Для развертывания базы данных разработанной структуры на предприятии, был написан код SQLскриптов для создания объектов базы данных в СУБД Microsoft SQL Server [3]. Листинг данного инсталляционного комплекта представлен в табл. 2.

Таблица 2 – Листинг кода создания таблиц

Код	Комментарий

`PROMPT Созданиетаблицыdir`
`CREATE TABLE`
`dir  (`
`[dir_id] int IDENTITY(1,1) NOT NULL,`
`[dir_name] varchar(50) NULL,`
`CONSTRAINT [PK_dir] PRIMARY KEY ([dir_id])`
`)`
	Создание таблицы dir

`PROMPT Созданиетаблицыpart`
`CREATE TABLE`
`part  (`
`[part_id] int IDENTITY(1,1) NOT NULL,`
`[part_name] varchar(50) NULL,`
`[dir_id] int NULL,`
`[part_price] real NULL,`
`[sup_id] int NULL,`
`CONSTRAINT [PK_part] PRIMARY KEY`
`([part_id])`
`)`
	Создание таблицы part

`PROMPT Созданиетаблицыsup`
`CREATE TABLE`
`sup  (`
`[sup_id] int IDENTITY(1,1) NOT NULL,`
`[sup_name] varchar(50) NULL,`
`CONSTRAINT [PK_sup] PRIMARY KEY ([sup_id])`
`)`
	Создание таблицы sup

Данные SQLскрипты используются в СУБД SQLServer для создания объектов баз данных.

1.4 Класс для работы с БД

Листинг кода класса для работы с базой данных приведен ниже:

```
package WindowsApplication1;
import System.Data.*;
import System.Data.SqlClient.*;

// Класс работы с базой данных
public class database
{
	// переменная идентификатор подключения
	SqlConnection conn;

	//конструктор класса, производит подключение к базе
	public database()
	{
		conn = new SqlConnection("Data Source=SKIF\\SQLEXPRESS;Initial
Catalog=mag;Integrated Security=SSPI");
		conn.Open();
	}

	// функция отправки запросов
	public void sql_query(String queryString)
	{
		SqlCommand command = new SqlCommand(queryString, conn);
		command.ExecuteNonQuery();
	}

	// закрываем коннект
	public void Close()
	{
		conn.Close();
	}

	// запрос -> данные типа DataTableдля вывода в гридах
	public System.Data.DataTable sql_table(String query)
	{
		System.Data.SqlClient.SqlCommand myCommand = new
System.Data.SqlClient.SqlCommand(query, conn);
		System.Data.SqlClient.SqlDataAdapter myDataAdapter = new
System.Data.SqlClient.SqlDataAdapter(myCommand);
		DataTable myDataTable = new DataTable();
		myDataAdapter.Fill(myDataTable);
		myCommand.Dispose();
		myDataAdapter.Dispose();
		return myDataTable;
	}
}
```

ВЫВОДЫ

При разработке класса работы с базой данных были описаны три метода database(), sql_query(query), close(), sql_table(query) и описана одна переменная класса, хранящая строку подключения к базе данных.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1 Власов А.И. Основы функционального моделирования // Курс лекций – М.: МГТУ им. Н. Э. Баумана, 2004, 50 с.
2 Власов А.И., Лыткин С.Л., Яковлев В.Л. Краткое практическое руководство по языку PL/SQL. – М.: Машиностроение - 2000. 64 с
3 Норенков И. П. Основы автоматизированного проектирования. - М.: МГТУ им. Н. Э. Баумана, 2000. - 360 с. ил.

В левой части окна программы находятся три кнопки, позволяющие переключать режим отображения и изменения данных.

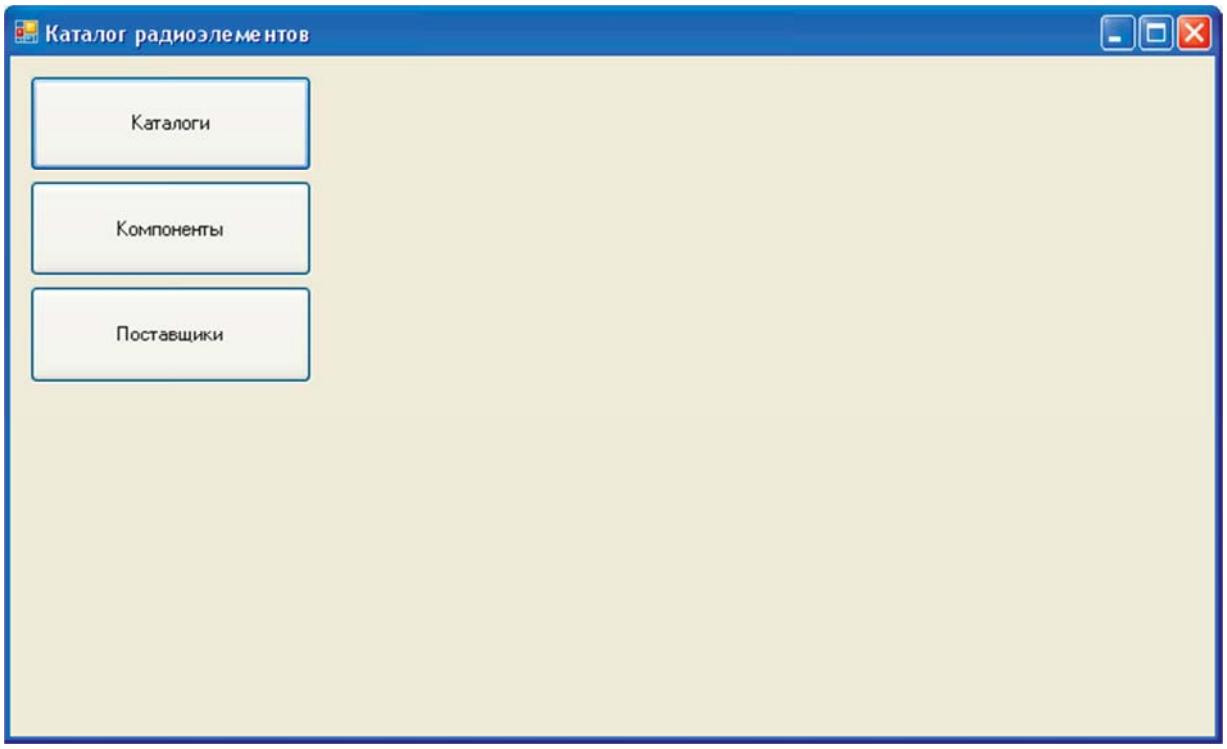

Рисунок 4 – Внешний вид программы

Нажатие на кнопку каталоги позволяет просматривать существующие каталоги, а также создавать новые.

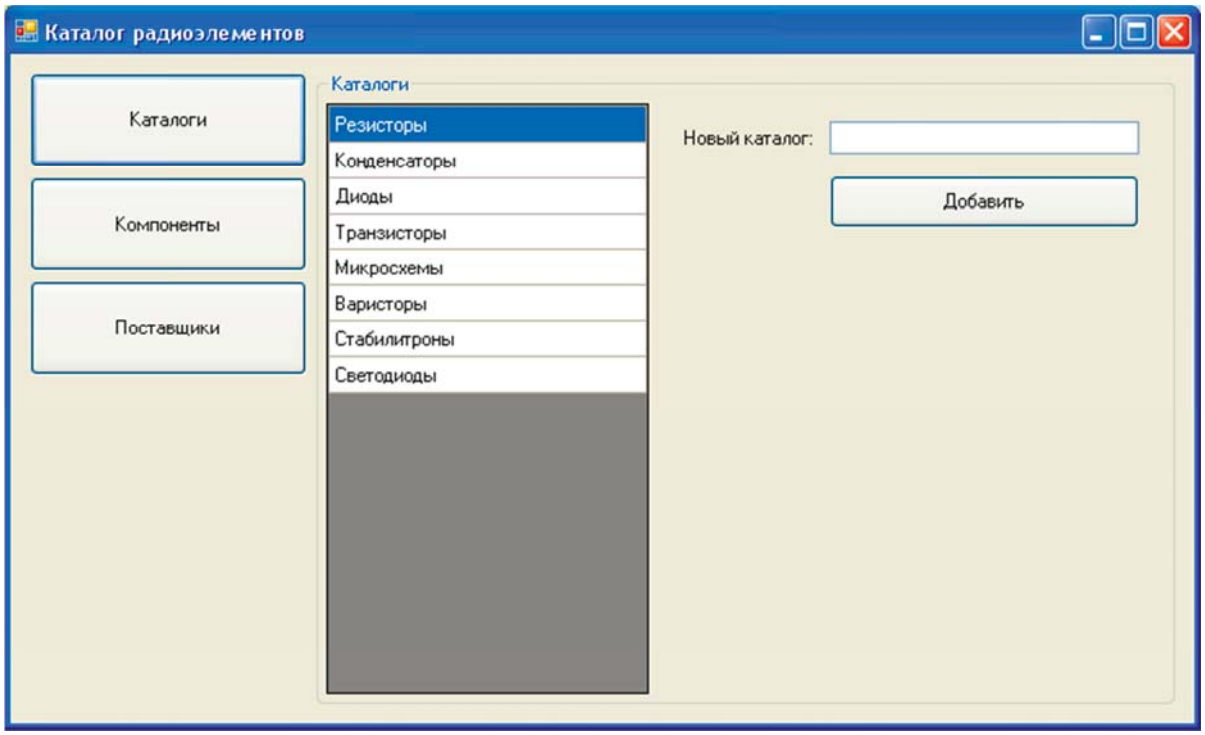

Рисунок 5 – Управление каталогами

В режиме просмотра списка компонентов необходимо выбрать, из какого каталога требуется отобразить компоненты. После выбора нужного каталога из выпадающего списка, компоненты загружаются автоматически.

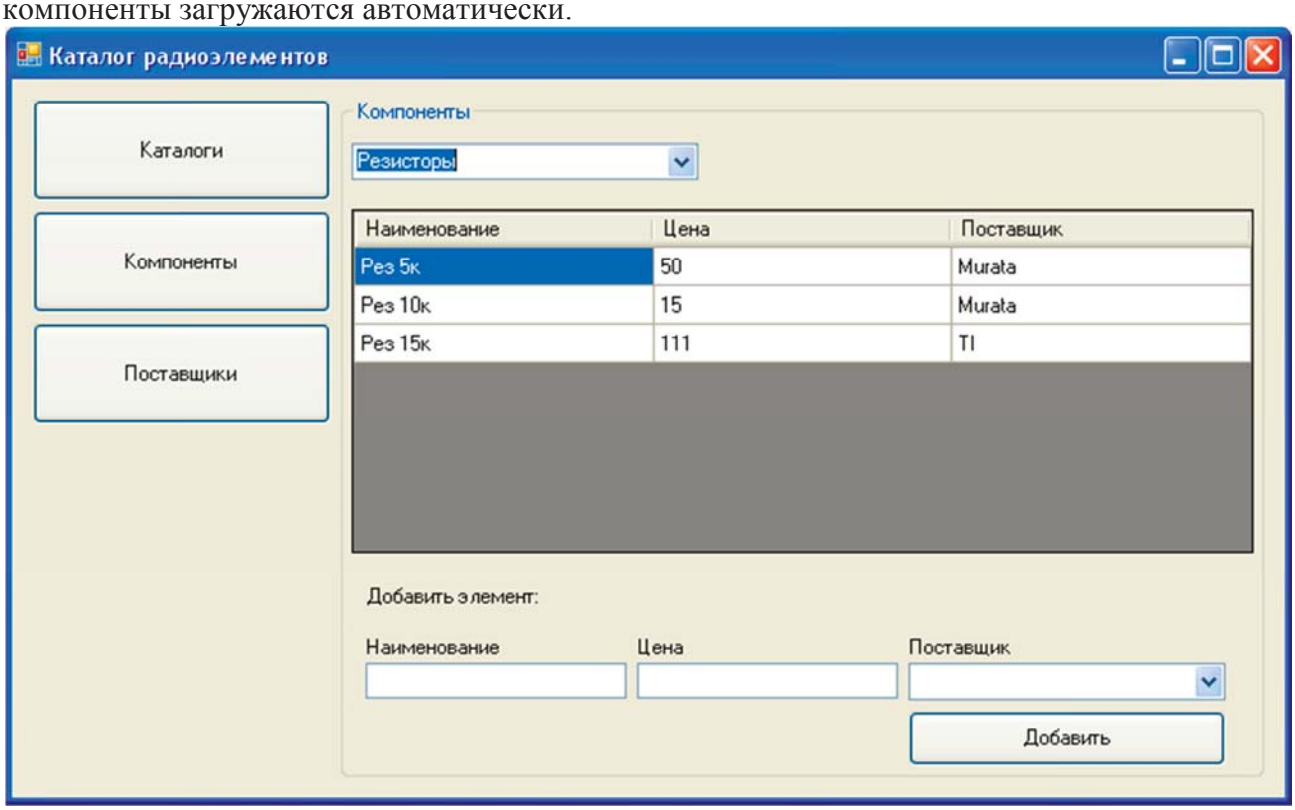

Рисунок 6 – Просмотр компонентов

Имеется возможность добавления новых компонентов. При этом осуществляется контроль правильности заполнения полей. Если они заполнены не полностью - отображается предупреждающее окно.

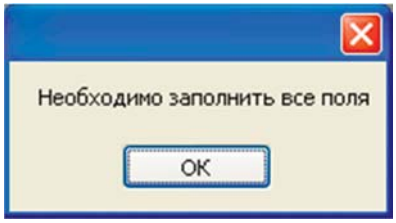

Рисунок 7 – Сообщение о неправильно заполненных полях

Раздел поставщиков аналогичен разделу с каталогами.

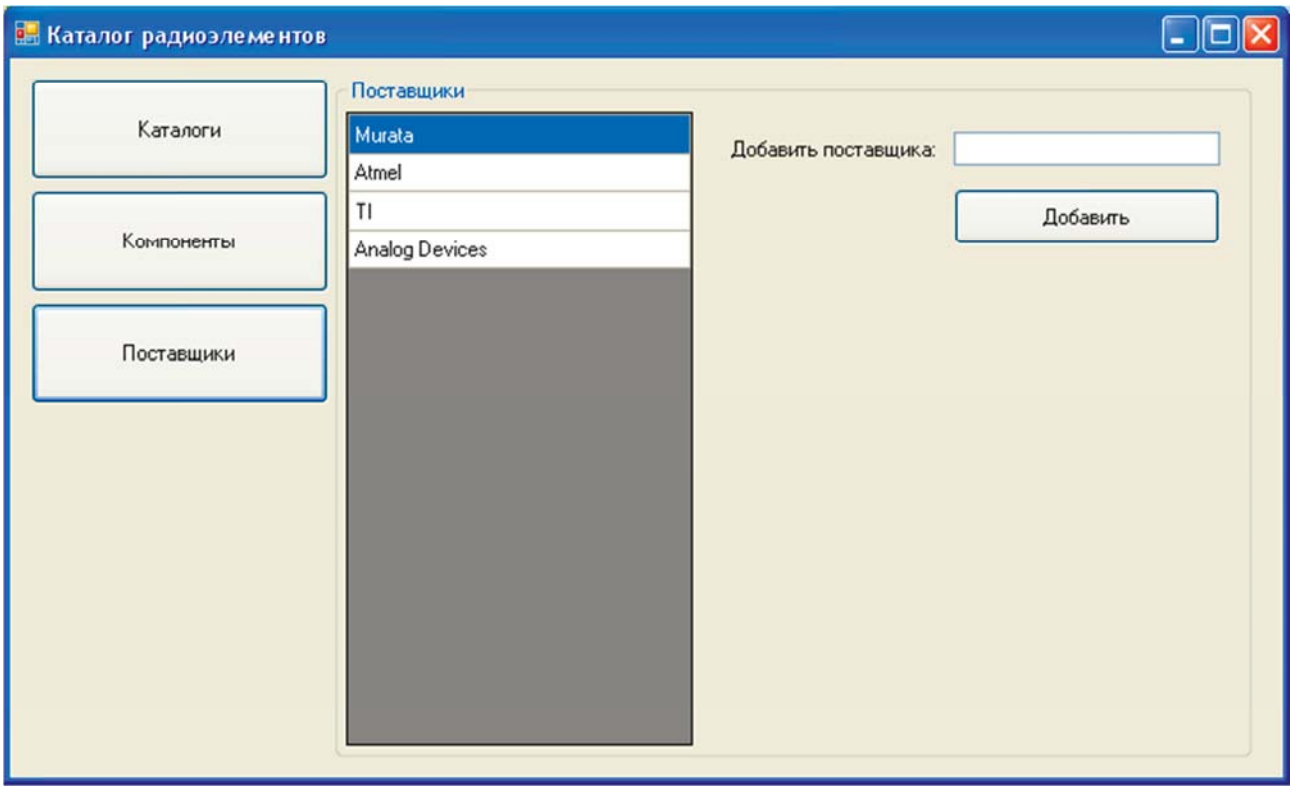

Рисунок 8 – Раздел просмотра и редактирования поставщиков

В таблице 3 приведен листинг части кода программы, отвечающей за обработку нажатий кнопок и прочих действий со стороны пользователя

Таблица 3 – Листинг кода программы

Код

```
 // Загрузка списка каталогов из БД
     public void dir_load()
     {
          database conn = new database();
          String query = "SELECT dir_name FROM dir";
          System.Data.DataTable Table = conn.sql_table(query);
          dir.set_DataSource(Table);
          conn.Close();
     }

// Загрузка списка компонентов из БД
     public void comp_load()
     {
          database conn = new database();
          String query = "SELECT dir_id,dir_name FROM dir";
          System.Data.DataTable Table = conn.sql_table(query);

          int dir =
Integer.parseInt(Table.get_Rows().get_Item(comboBox1.get_SelectedIndex()).get_Item(0).toString());

          query = "SELECT part_name AS 'Наименование',part_price AS 'Цена',sup_name AS 'Поставщик'
FROM part,sup WHERE part.sup_id=sup.sup_id AND dir_id=" + dir;
          Table = conn.sql_table(query);
          conn.Close();
          comp.set_DataSource(Table);
     }

// Загрузка списка поставщиков из БД
     public void supl_load()
     {
          database conn = new database();
          String query = "SELECT sup_name FROM sup";
          System.Data.DataTable Table = conn.sql_table(query);
          sup_grid.set_DataSource(Table);
```

```
          conn.Close();
     }


// Активация одного из трех основных рабочих полей программы
     public void activate(int num)
     {
          groupBox1.set_Visible(false);
          groupBox2.set_Visible(false);
          groupBox3.set_Visible(false);

          switch (num)
          {
                   case 1:
                    groupBox1.set_Visible(true);
                    break;
                    case 2:
                    groupBox2.set_Visible(true);
                    break;
                    case 3:
                    groupBox3.set_Visible(true);
                    break;
          }
     }

// Нажатие на кнопку КАТАЛОГ
     private void button1_Click(Object sender, System.EventArgs e)
     {
          activate(1);
          dir_load();
     }

// Добавление нового элемента в каталог
     private void button2_Click(Object sender, System.EventArgs e)
     {
          if (!(textBox1.get_Text().get_Length() <= 1))
          {
               groupBox1.set_Visible(true);
               database conn = new database();
               String query = "INSERT INTO dir (dir_name) values('" + textBox1.get_Text() + "')";
               conn.sql_table(query);
               conn.Close();
               dir_load();
          }
          else
          {
               MessageBox.Show("Необходимо заполнить все поля");
          }
     }

// Нажатие на кнопку КОМПОНЕНТЫ
     private void button3_Click(Object sender, System.EventArgs e)
     {
          activate(2);

          database conn = new database();
          String query = "SELECT dir_id,dir_name FROM dir";
          System.Data.DataTable Table = conn.sql_table(query);

        comboBox1.get_Items().Clear();
          for (int i = 0; i < Table.get_Rows().get_Count(); i++)
               comboBox1.get_Items().AddRange(new Object[] {
Table.get_Rows().get_Item(i).get_Item(1).toString() });

          query = "SELECT sup_id,sup_name FROM sup";
          Table = conn.sql_table(query);
          sup.get_Items().Clear();
          for (int i = 0; i < Table.get_Rows().get_Count(); i++)
          sup.get_Items().AddRange(new Object[] {
Table.get_Rows().get_Item(i).get_Item(1).toString() });


          conn.Close();

     }


// Изменение выбранного пункта в выпадающем списке каталогов
     private void comboBox1_SelectedIndexChanged(Object sender, System.EventArgs e)
     {
```

```
          comp_load();
     }

// Добавление нового компонента
     private void button4_Click(Object sender, System.EventArgs e)
     {
          database conn = new database();

          if (!(part_name.get_Text().get_Length() <=1 && part_price.get_Text().get_Length() <= 1))
          {
               String query = "SELECT sup_id,sup_name FROM sup";
               System.Data.DataTable Table = conn.sql_table(query);
               int supl =
Integer.parseInt(Table.get_Rows().get_Item(sup.get_SelectedIndex()).get_Item(0).toString());

               query = "SELECT dir_id,dir_name FROM dir";
               Table = conn.sql_table(query);
               int dir =
Integer.parseInt(Table.get_Rows().get_Item(comboBox1.get_SelectedIndex()).get_Item(0).toString());


               query = "INSERT INTO part (part_name,part_price,sup_id,dir_id) VALUES ('" +
part_name.get_Text() + "','" + part_price.get_Text() + "','" + supl + "','" + dir + "')";
               conn.sql_query(query);
               part_name.set_Text("");
               part_price.set_Text("");
               conn.Close();
               comp_load();

          }
          else
          {
               MessageBox.Show("Необходимо заполнить все поля");
          }
     }

// Нажатие на кнопку ПОСТАВЩИКИ
     private void button5_Click(Object sender, System.EventArgs e)
     {
          activate(3);
          supl_load();
     }

// Добавление нового поставщика
     private void button7_Click(Object sender, System.EventArgs e)
     {
          if (!(textBox4.get_Text().get_Length() <= 1))
          {
               groupBox1.set_Visible(true);
               database conn = new database();
               String query = "INSERT INTO sup (sup_name) values('" + textBox4.get_Text() + "')";
               conn.sql_table(query);
               conn.Close();
               supl_load();
          }
          else
          {
               MessageBox.Show("Необходимо заполнить все поля");
          }
     }
}
```