# Отчет за квалификационный экзамен студента группы ПКС-304 Абдюшева Артема

# База данных

Логическая модель

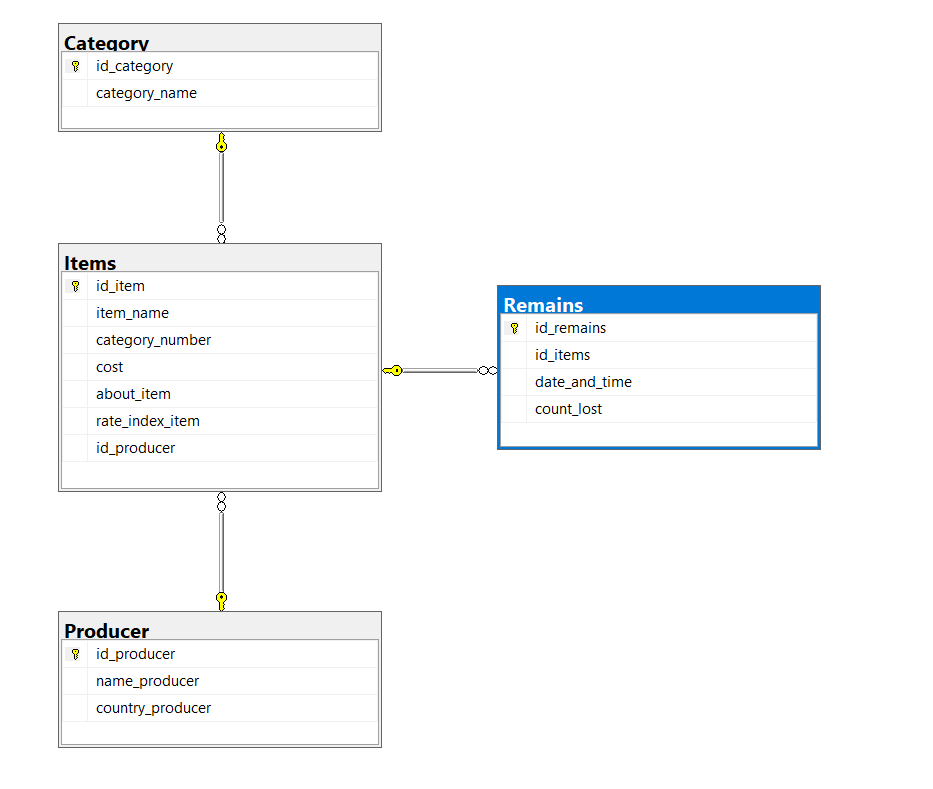


Таблица Category(категории)

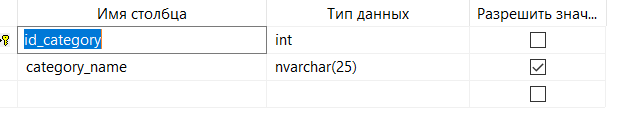
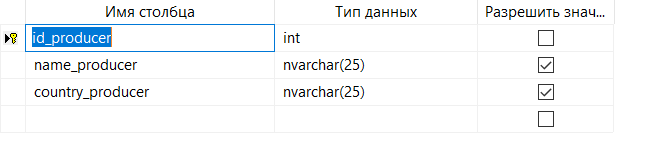




Таблица Producer(Производитель)



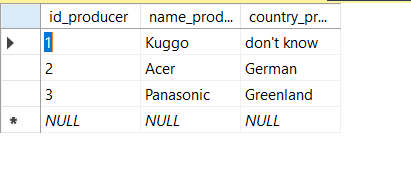
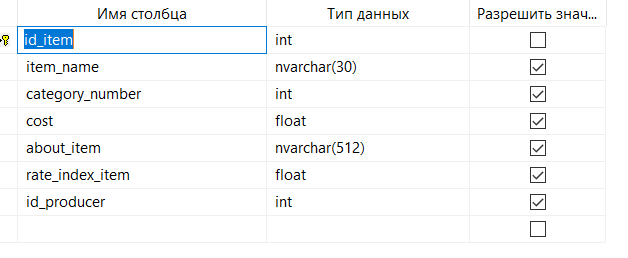


Таблица Items(товары)



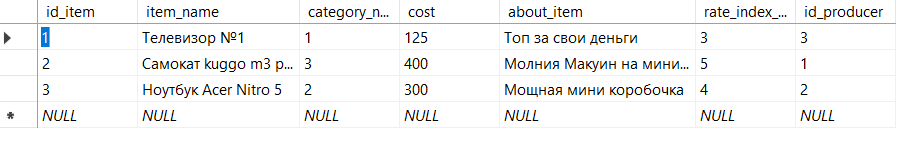
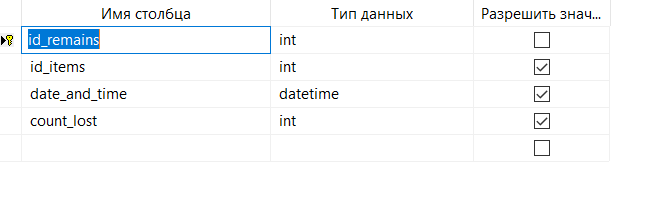
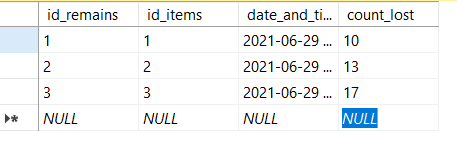


Таблица Remains(Остатки товаров)





USE [master]

GO

/\*\*\*\*\*\* Object: Database [KvalEkzamen] Script Date: 29.06.2021 10:03:20 \*\*\*\*\*\*/

CREATE DATABASE [KvalEkzamen]

CONTAINMENT = NONE

ON PRIMARY

( NAME = N'KvalEkzamen', FILENAME = N'D:\Program Files\Microsoft SQL Server\MSSQL14.MSSQLSERVER\MSSQL\DATA\KvalEkzamen.mdf' , SIZE = 8192KB , MAXSIZE = UNLIMITED, FILEGROWTH = 65536KB )

LOG ON

( NAME = N'KvalEkzamen\_log', FILENAME = N'D:\Program Files\Microsoft SQL Server\MSSQL14.MSSQLSERVER\MSSQL\DATA\KvalEkzamen\_log.ldf' , SIZE = 8192KB , MAXSIZE = 2048GB , FILEGROWTH = 65536KB )

GO

ALTER DATABASE [KvalEkzamen] SET COMPATIBILITY\_LEVEL = 140

GO

IF (1 = FULLTEXTSERVICEPROPERTY('IsFullTextInstalled'))

begin

EXEC [KvalEkzamen].[dbo].[sp\_fulltext\_database] @action = 'enable'

end

GO

ALTER DATABASE [KvalEkzamen] SET ANSI\_NULL\_DEFAULT OFF

GO

ALTER DATABASE [KvalEkzamen] SET ANSI\_NULLS OFF

GO

ALTER DATABASE [KvalEkzamen] SET ANSI\_PADDING OFF

GO

ALTER DATABASE [KvalEkzamen] SET ANSI\_WARNINGS OFF

GO

ALTER DATABASE [KvalEkzamen] SET ARITHABORT OFF

GO

ALTER DATABASE [KvalEkzamen] SET AUTO\_CLOSE OFF

GO

ALTER DATABASE [KvalEkzamen] SET AUTO\_SHRINK OFF

GO

ALTER DATABASE [KvalEkzamen] SET AUTO\_UPDATE\_STATISTICS ON

GO

ALTER DATABASE [KvalEkzamen] SET CURSOR\_CLOSE\_ON\_COMMIT OFF

GO

ALTER DATABASE [KvalEkzamen] SET CURSOR\_DEFAULT GLOBAL

GO

ALTER DATABASE [KvalEkzamen] SET CONCAT\_NULL\_YIELDS\_NULL OFF

GO

ALTER DATABASE [KvalEkzamen] SET NUMERIC\_ROUNDABORT OFF

GO

ALTER DATABASE [KvalEkzamen] SET QUOTED\_IDENTIFIER OFF

GO

ALTER DATABASE [KvalEkzamen] SET RECURSIVE\_TRIGGERS OFF

GO

ALTER DATABASE [KvalEkzamen] SET DISABLE\_BROKER

GO

ALTER DATABASE [KvalEkzamen] SET AUTO\_UPDATE\_STATISTICS\_ASYNC OFF

GO

ALTER DATABASE [KvalEkzamen] SET DATE\_CORRELATION\_OPTIMIZATION OFF

GO

ALTER DATABASE [KvalEkzamen] SET TRUSTWORTHY OFF

GO

ALTER DATABASE [KvalEkzamen] SET ALLOW\_SNAPSHOT\_ISOLATION OFF

GO

ALTER DATABASE [KvalEkzamen] SET PARAMETERIZATION SIMPLE

GO

ALTER DATABASE [KvalEkzamen] SET READ\_COMMITTED\_SNAPSHOT OFF

GO

ALTER DATABASE [KvalEkzamen] SET HONOR\_BROKER\_PRIORITY OFF

GO

ALTER DATABASE [KvalEkzamen] SET RECOVERY SIMPLE

GO

ALTER DATABASE [KvalEkzamen] SET MULTI\_USER

GO

ALTER DATABASE [KvalEkzamen] SET PAGE\_VERIFY CHECKSUM

GO

ALTER DATABASE [KvalEkzamen] SET DB\_CHAINING OFF

GO

ALTER DATABASE [KvalEkzamen] SET FILESTREAM( NON\_TRANSACTED\_ACCESS = OFF )

GO

ALTER DATABASE [KvalEkzamen] SET TARGET\_RECOVERY\_TIME = 60 SECONDS

GO

ALTER DATABASE [KvalEkzamen] SET DELAYED\_DURABILITY = DISABLED

GO

ALTER DATABASE [KvalEkzamen] SET QUERY\_STORE = OFF

GO

USE [KvalEkzamen]

GO

/\*\*\*\*\*\* Object: Table [dbo].[Category] Script Date: 29.06.2021 10:03:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Category](

[id\_category] [int] NOT NULL,

[category\_name] [nvarchar](25) NULL,

CONSTRAINT [PK\_Category] PRIMARY KEY CLUSTERED

(

[id\_category] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[Items] Script Date: 29.06.2021 10:03:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Items](

[id\_item] [int] NOT NULL,

[item\_name] [nvarchar](30) NULL,

[category\_number] [int] NULL,

[cost] [float] NULL,

[about\_item] [nvarchar](512) NULL,

[rate\_index\_item] [float] NULL,

[id\_producer] [int] NULL,

CONSTRAINT [PK\_Items] PRIMARY KEY CLUSTERED

(

[id\_item] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[Producer] Script Date: 29.06.2021 10:03:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Producer](

[id\_producer] [int] NOT NULL,

[name\_producer] [nvarchar](25) NULL,

[country\_producer] [nvarchar](25) NULL,

CONSTRAINT [PK\_Producer] PRIMARY KEY CLUSTERED

(

[id\_producer] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[Remains] Script Date: 29.06.2021 10:03:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Remains](

[id\_remains] [int] NOT NULL,

[id\_items] [int] NULL,

[date\_and\_time] [datetime] NULL,

[count\_lost] [int] NULL,

CONSTRAINT [PK\_Remains] PRIMARY KEY CLUSTERED

(

[id\_remains] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO

ALTER TABLE [dbo].[Items] WITH CHECK ADD CONSTRAINT [FK\_Items\_Category] FOREIGN KEY([category\_number])

REFERENCES [dbo].[Category] ([id\_category])

GO

ALTER TABLE [dbo].[Items] CHECK CONSTRAINT [FK\_Items\_Category]

GO

ALTER TABLE [dbo].[Items] WITH CHECK ADD CONSTRAINT [FK\_Items\_Producer] FOREIGN KEY([id\_producer])

REFERENCES [dbo].[Producer] ([id\_producer])

GO

ALTER TABLE [dbo].[Items] CHECK CONSTRAINT [FK\_Items\_Producer]

GO

ALTER TABLE [dbo].[Remains] WITH CHECK ADD CONSTRAINT [FK\_Remains\_Items] FOREIGN KEY([id\_items])

REFERENCES [dbo].[Items] ([id\_item])

GO

ALTER TABLE [dbo].[Remains] CHECK CONSTRAINT [FK\_Remains\_Items]

GO

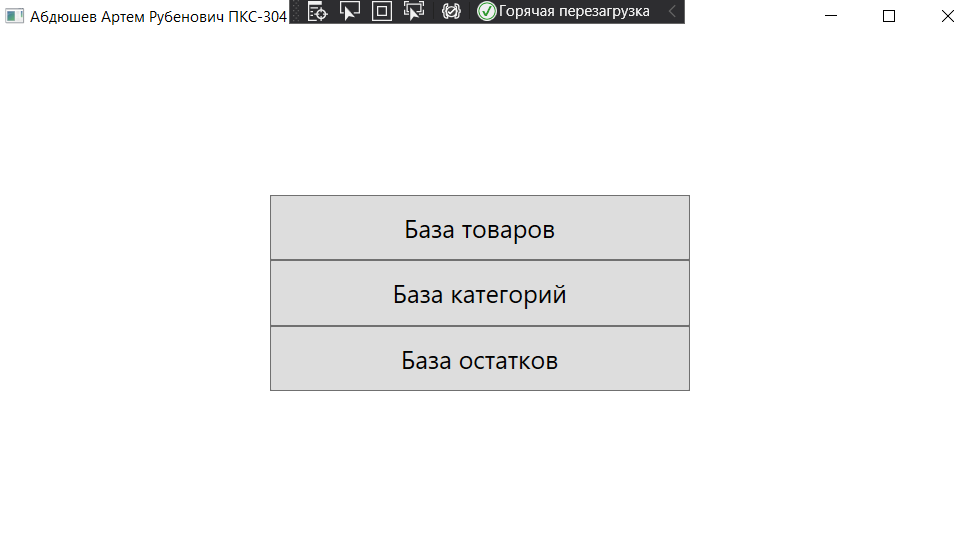
USE [master]

GO

ALTER DATABASE [KvalEkzamen] SET READ\_WRITE

GO

# Главное меню



Код

private void OpenItemsPage(object sender, RoutedEventArgs e)

{

Manager.frame.Navigate(new ItemPage());

}

private void OpenCategoryPage(object sender, RoutedEventArgs e)

{

Manager.frame.Navigate(new CategoryPage());

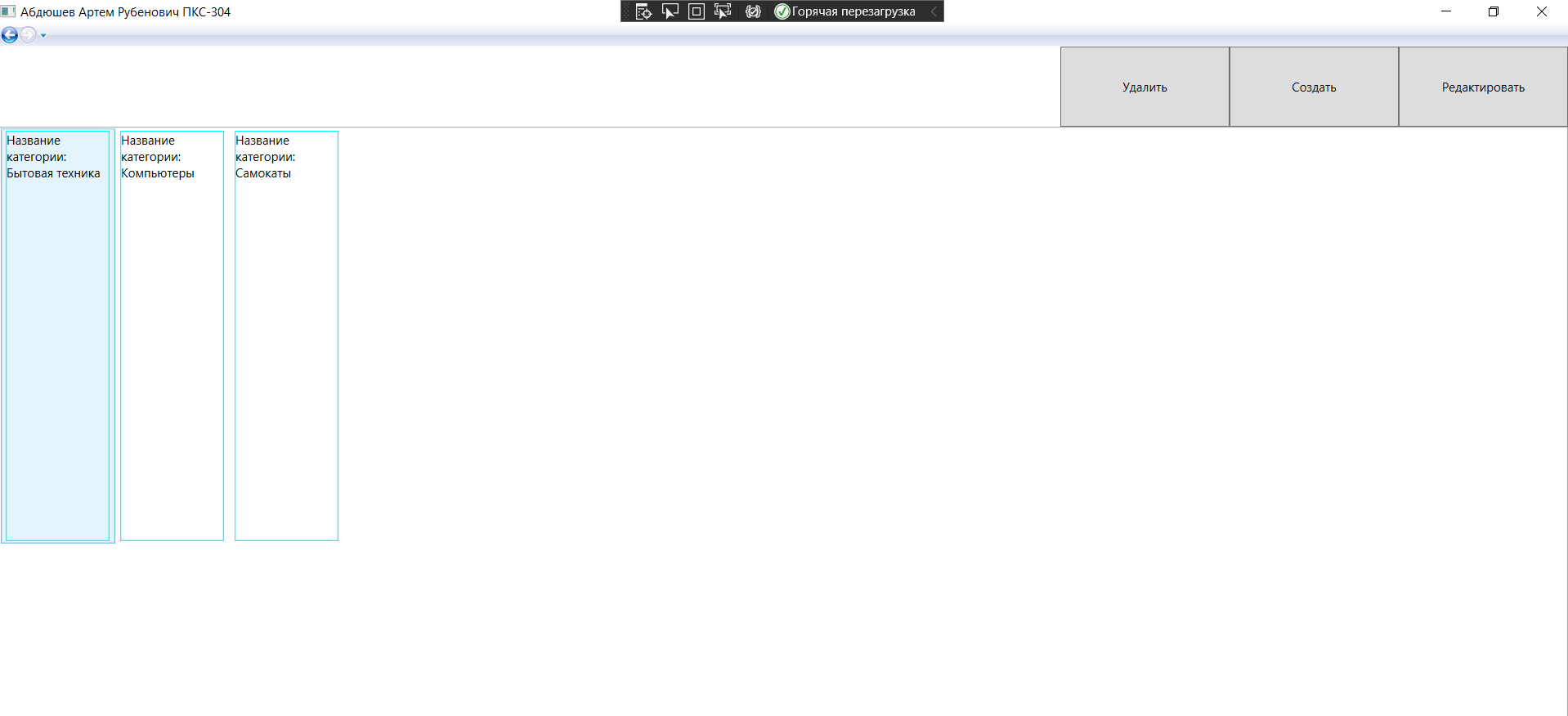
}

private void OpenRemainsPage(object sender, RoutedEventArgs e)

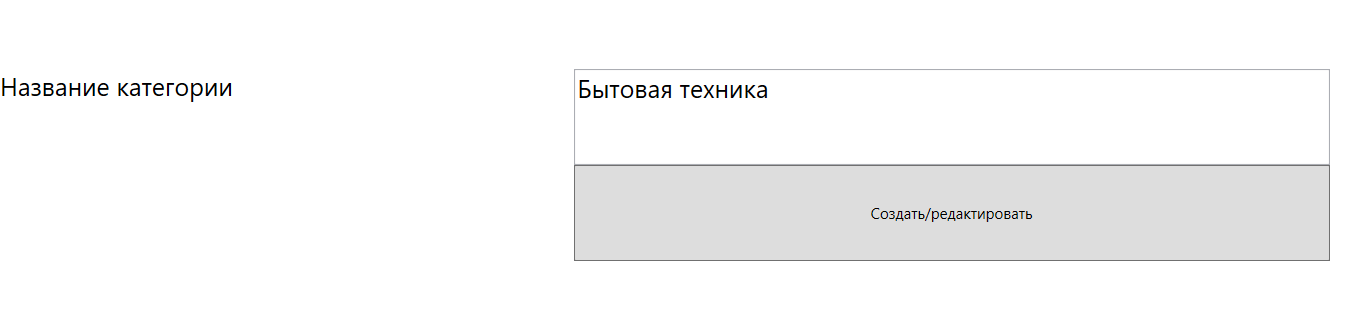
{

}

# Работа с категориями







public CategoryPage()

{

InitializeComponent();

using (SqlConnection connection = new SqlConnection(Manager.connectionInfo))

{

try

{

connection.Open();

string sqlExpression = "SELECT \* FROM Category";

using (SqlCommand command = new SqlCommand(sqlExpression, connection))

{

SqlDataReader reader = command.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

CategoryList.Items.Add(new Category

{

id\_category = reader.GetInt32(0),

category\_name = reader.GetString(1)

});

}

}

else

{

MessageBox.Show("Категории отсутствуют...");

}

}

}

catch (SqlException except)

{

MessageBox.Show(except.ToString());

}

finally

{

connection.Close();

}

}

}

private void btnRemove(object sender, RoutedEventArgs e)

{

if (CategoryList.SelectedIndex != -1)

{

using (SqlConnection connection = new SqlConnection(Manager.connectionInfo))

{

try

{

connection.Open();

using (SqlCommand command = new SqlCommand("DELETE FROM Category Where id\_category = @id\_category", connection))

{

command.Parameters.AddWithValue("@id\_category", ((Category)CategoryList.SelectedItem).id\_category);

int number = -1;

number = command.ExecuteNonQuery();

if (number == -1) MessageBox.Show("Неудачное удаление");

else MessageBox.Show("Успешное удаление");

}

}

catch (SqlException except)

{

MessageBox.Show(except.ToString());

}

finally

{

connection.Close();

}

}

}

else

{

MessageBox.Show("Вы не выбрали категорию товара");

}

}

private void btnCreate(object sender, RoutedEventArgs e)

{

Manager.frame.Navigate(new AddEditCategoryPage());

}

private void btnEdit(object sender, RoutedEventArgs e)

{

if (CategoryList.SelectedIndex != -1)

{

Manager.frame.Navigate(new AddEditCategoryPage((Category)CategoryList.SelectedItem));

} else

{

MessageBox.Show("Вы не выбрали категорию для редактирования");

}

}

private Category category1 { get; set; }

private bool isEdit { get; set; }

public AddEditCategoryPage() //создание

{

InitializeComponent();

isEdit = false;

}

public AddEditCategoryPage(Category category)

{

InitializeComponent();

isEdit = true;

category1 = category;

CategoryInput.Text = category1.category\_name;

}

private void AddEdit(object sender, RoutedEventArgs e)

{

Regex regex = new Regex("^([А-Я]{1}[а-яё]{3,54})$");

MatchCollection matches = regex.Matches(CategoryInput.Text);

if (matches.Count > 0)

{

using (SqlConnection connection = new SqlConnection(Manager.connectionInfo))

{

try

{

connection.Open();

string sqlExpression = null;

if (isEdit) //редактирование

{

sqlExpression = "UPDATE [dbo].[Category] SET [category\_name] = @category\_name WHERE id\_category = @id\_category";

using (SqlCommand command = new SqlCommand(sqlExpression, connection))

{

command.Parameters.AddWithValue("category\_name", CategoryInput.Text);

command.Parameters.AddWithValue("@id\_category", category1.id\_category);

command.ExecuteNonQuery();

}

}

else //создание

{

sqlExpression = "BEGIN DECLARE @index int; set @index = (SELECT MAX(id\_category) AS Expr1 FROM dbo.Category)+1; IF(@index is NULL) SET @index = 1 INSERT INTO dbo.Category VALUES (@index, @category\_name)";

using (SqlCommand command = new SqlCommand(sqlExpression, connection))

{

command.Parameters.AddWithValue("@category\_name", CategoryInput.Text);

command.ExecuteNonQuery();

}

}

}

catch (SqlException except)

{

MessageBox.Show(except.ToString());

}

finally

{

connection.Close();

Manager.frame.Navigate(new Menu());

}

}

} else

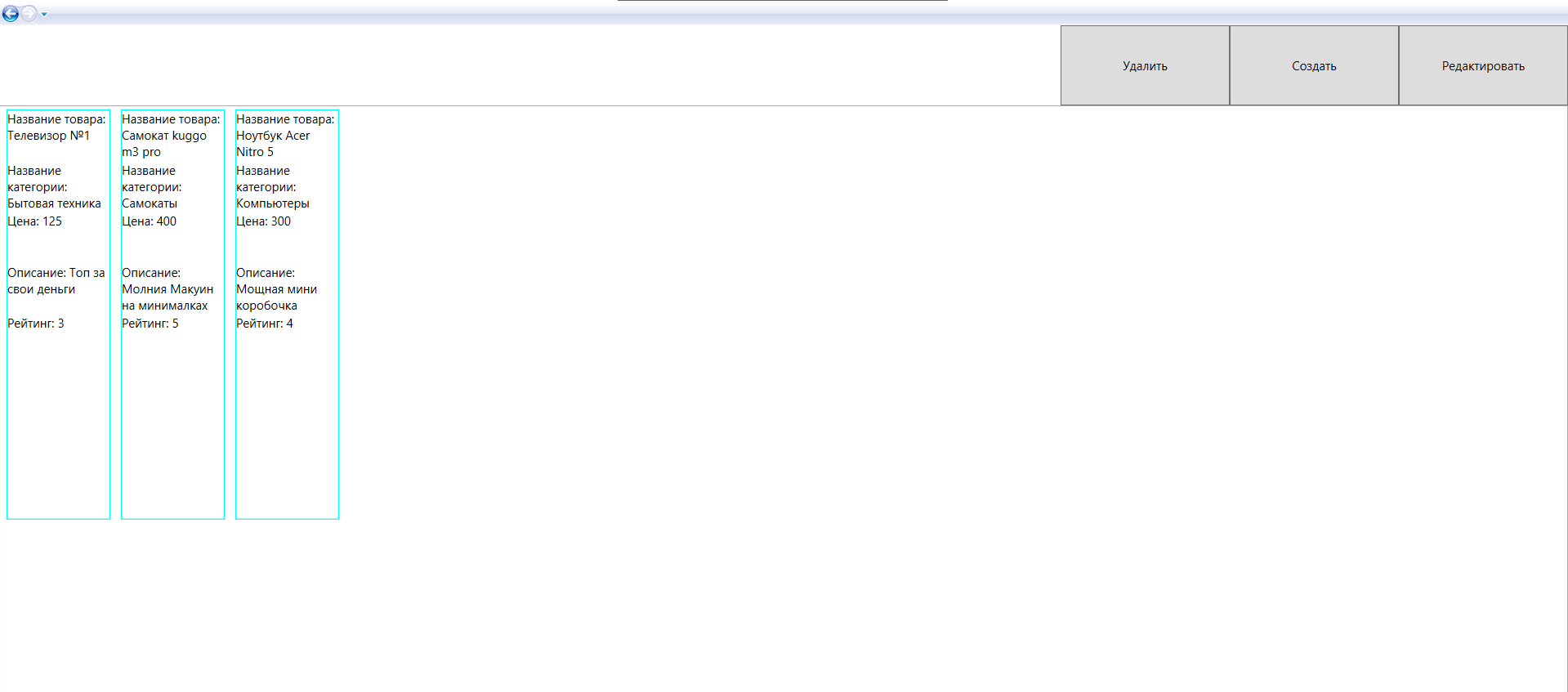
{

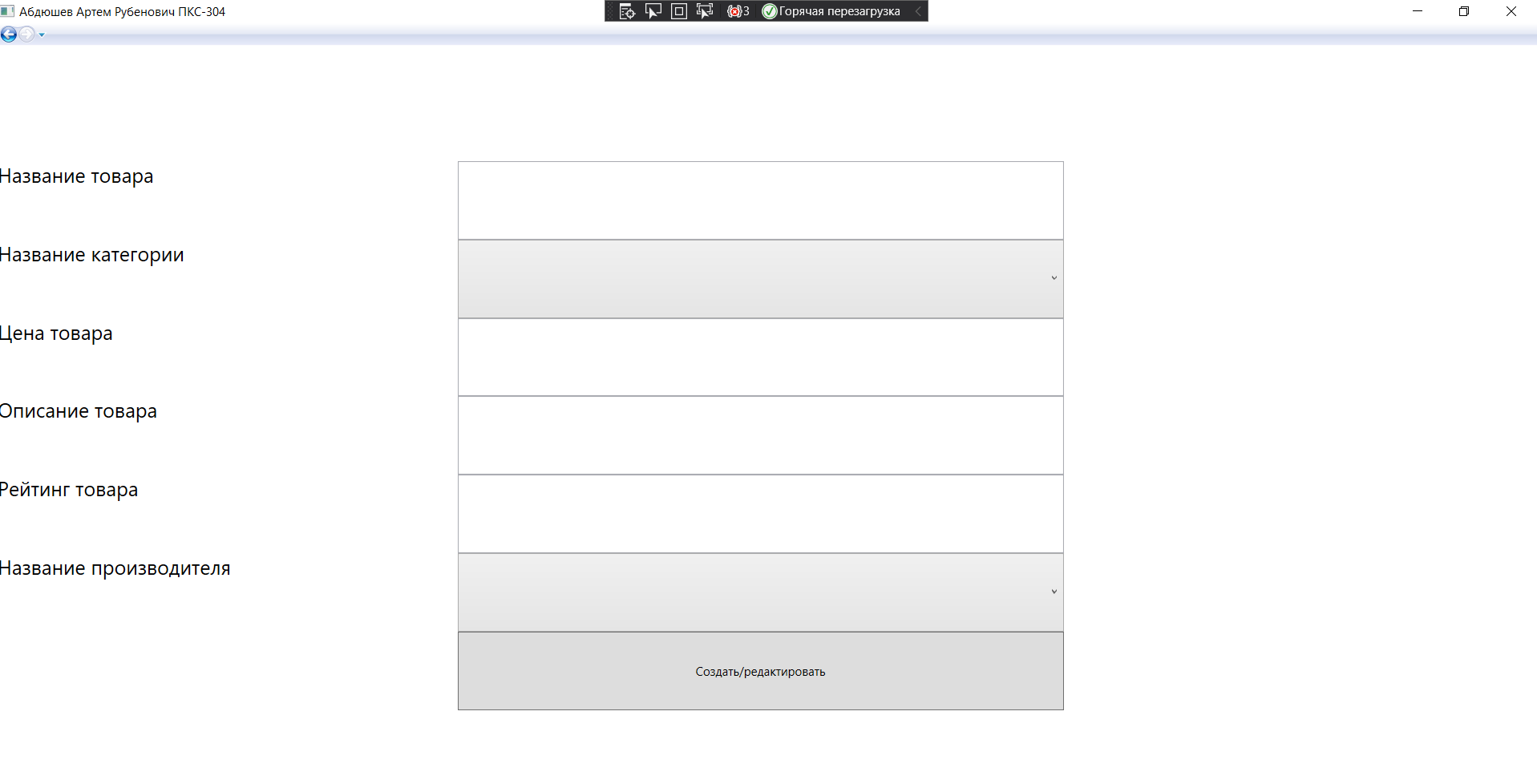
MessageBox.Show("Некорректный ввод категории");

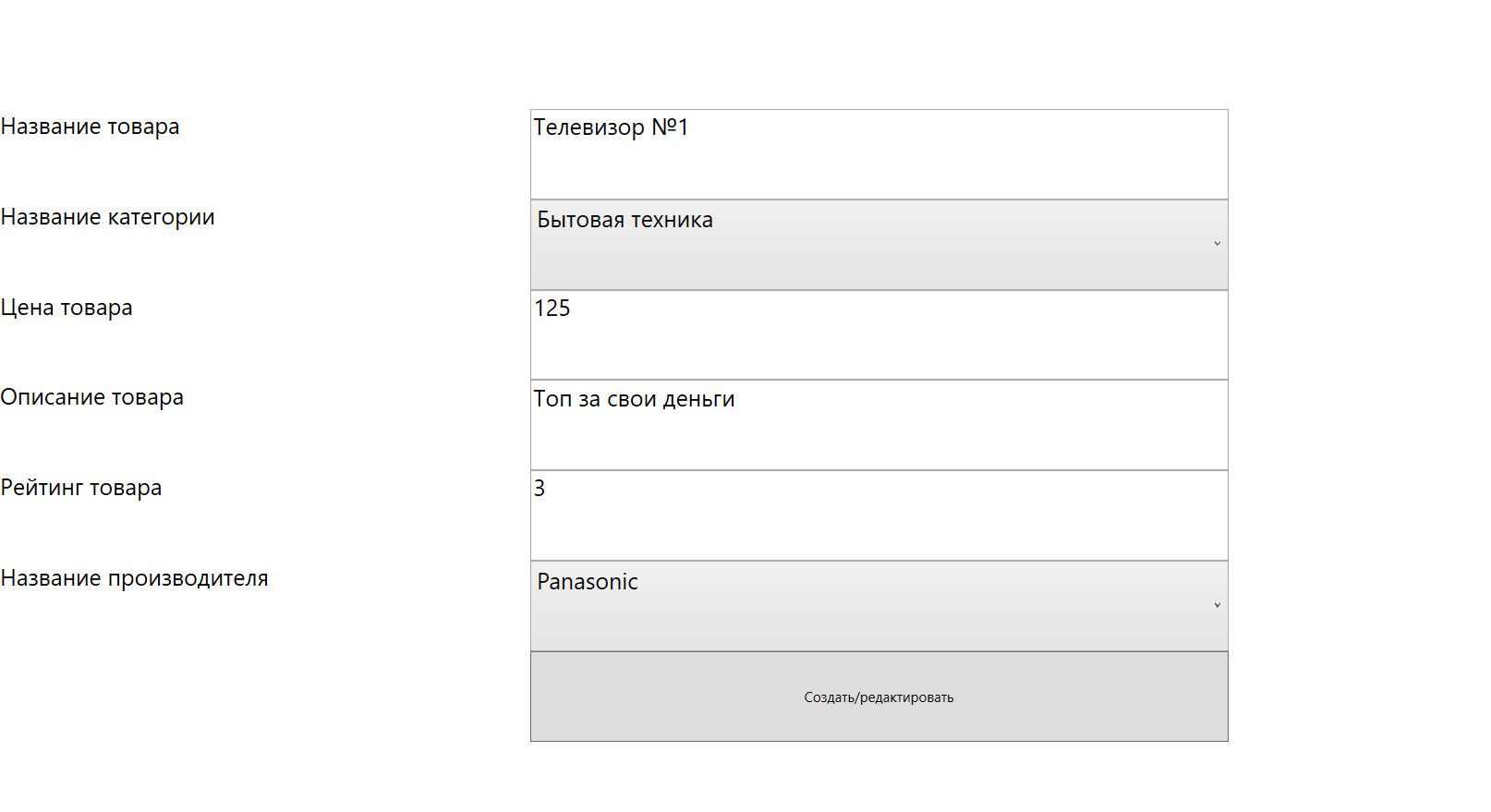
}

}

# Работа с товарами







public ItemPage()

{

InitializeComponent();

using (SqlConnection connection = new SqlConnection(Manager.connectionInfo))

{

try

{

connection.Open();

string sqlExpression = "SELECT dbo.Items.id\_item, dbo.Items.item\_name, dbo.Items.category\_number, dbo.Items.cost, dbo.Category.category\_name, dbo.Items.about\_item, dbo.Items.rate\_index\_item, dbo.Items.id\_producer, dbo.Producer.name\_producer FROM dbo.Items INNER JOIN dbo.Category ON dbo.Items.category\_number = dbo.Category.id\_category INNER JOIN dbo.Producer ON dbo.Items.id\_producer = dbo.Producer.id\_producer";

using (SqlCommand command = new SqlCommand(sqlExpression, connection))

{

SqlDataReader reader = command.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

ItemList.Items.Add(new Items

{

id\_item = reader.GetInt32(0),

item\_name = reader.GetString(1),

category\_number = reader.GetInt32(2),

cost = reader.GetDouble(3),

category\_name = reader.GetString(4),

about\_item = reader.GetString(5),

rate\_index\_item = reader.GetDouble(6),

id\_producer = reader.GetInt32(7),

producer\_name = reader.GetString(8)

});

}

}

else

{

MessageBox.Show("Товары отсутствуют...");

}

}

}

catch (SqlException except)

{

MessageBox.Show(except.ToString());

}

finally

{

connection.Close();

}

}

}

private void btnRemove(object sender, RoutedEventArgs e)

{

if (ItemList.SelectedIndex != -1)

{

using (SqlConnection connection = new SqlConnection(Manager.connectionInfo))

{

try

{

connection.Open();

using (SqlCommand command = new SqlCommand("DELETE FROM Items Where id\_item = @id\_item", connection))

{

command.Parameters.AddWithValue("@id\_item", ((Items)ItemList.SelectedItem).id\_item);

int number = -1;

number = command.ExecuteNonQuery();

if (number == -1) MessageBox.Show("Неудачное удаление");

else MessageBox.Show("Успешное удаление");

}

}

catch (SqlException except)

{

MessageBox.Show(except.ToString());

}

finally

{

connection.Close();

}

}

}

else

{

MessageBox.Show("Вы не выбрали товар");

}

}

private void btnCreate(object sender, RoutedEventArgs e)

{

Manager.frame.Navigate(new AddEditItemPage());

}

private void btnEdit(object sender, RoutedEventArgs e)

{

if (ItemList.SelectedIndex != -1)

{

Manager.frame.Navigate(new AddEditItemPage((Items)ItemList.SelectedItem));

} else

{

MessageBox.Show("Вы не выбрали товар");

}

}

private Items items1 { get; set; }

private bool isEdit { get; set; }

private void fillCombo()

{

using (SqlConnection connection = new SqlConnection(Manager.connectionInfo))

{

try

{

connection.Open();

string sqlExpression = "SELECT name\_producer FROM dbo.Producer GROUP BY name\_producer";

using (SqlCommand command = new SqlCommand(sqlExpression, connection))

{

SqlDataReader reader = command.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

producerInput.Items.Add(reader.GetString(0));

}

}

}

}

catch (SqlException except)

{

MessageBox.Show(except.ToString());

}

finally

{

connection.Close();

}

try

{

connection.Open();

string sqlExpression = "SELECT category\_name FROM dbo.Category GROUP BY category\_name";

using (SqlCommand command = new SqlCommand(sqlExpression, connection))

{

SqlDataReader reader = command.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

CategoryInput.Items.Add(reader.GetString(0));

}

}

}

}

catch (SqlException except)

{

MessageBox.Show(except.ToString());

}

finally

{

connection.Close();

}

}

}

public AddEditItemPage()

{

InitializeComponent();

fillCombo();

isEdit = false;

}

public AddEditItemPage(Items items)

{

InitializeComponent();

fillCombo();

isEdit = true;

items1 = items;

ItemInput.Text = items.item\_name;

CategoryInput.Text = items.category\_name;

CostInput.Text = items.cost.ToString();

InfoInput.Text = items.about\_item;

RateInput.Text = items.rate\_index\_item.ToString();

CategoryInput.SelectedItem = items.category\_name;

producerInput.SelectedItem = items.producer\_name;

}

private void AddEdit(object sender, RoutedEventArgs e)

{

if (producerInput.SelectedIndex != -1)

{

if (CategoryInput.SelectedIndex != -1)

{

if (ItemInput.Text != "")

{

if (CostInput.Text != "")

{

if (InfoInput.Text != "")

{

if (RateInput.Text != "")

{

//тут редактирование и удаление))))))) типо оно есть)))))

}

}

}

}

}

} else

{

MessageBox.Show("Вы не выбрали производителя");

}

}