**Федеральное агентство связи**

**Ордена Трудового Красного Знамени**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Московский технический университет связи и информатики»**

Кафедра Интеллектуальные системы в управлении и автоматизации

****

**Отчет по курсовой работе**

по предмету «Технологии баз данных»

на тему:

«БД продажа квартир»

Выполнил:

студент группы БВТ1802

Руководитель:

Вовик А. Г.

Москва 2021

**Оглавление**

[**Введение** 2](#_Toc73822343)

[**ГЛАВА 1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ** 4](#_Toc73822344)

[**1.1** **Постановка задачи проекта** 4](#_Toc73822345)

[**ГЛАВА 2. ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ “Продажа ЖД билетов”** 5](#_Toc73822346)

[**2.1** **Разработка концепции автоматизации продажи билетов** 5](#_Toc73822347)

[**Модель БД** 5](#_Toc73822348)

[Рисунок 1 – модель БД 5](#_Toc73822349)

[**2.2** **Процесс работы** 6](#_Toc73822350)

[**2.3** **Структура таблиц** 6](#_Toc73822351)

[**Вывод** 8](#_Toc73822352)

[**ГЛАВА 3. ПРОГРАММНАЯ РЕАЛИЗАЦИЯ** 9](#_Toc73822353)

[**3.1** **Разработка контрольного примера** 9](#_Toc73822354)

[**3.2 Разработка алгоритмов и текстов программ** 12](#_Toc73822355)

[**3.3 Создание таблиц** 12](#_Toc73822356)

[**3.4 Создание представлений** 15](#_Toc73822357)

[**3.5 Создание процедуры** 16](#_Toc73822358)

[**3.6 Основной код интерфейса приложения** 17](#_Toc73822359)

[**3.6 Разработка оконных форм взаимодействия с конечным пользователем** 19](#_Toc73822360)

[**Вывод** 21](#_Toc73822361)

[**ЗАКЛЮЧЕНИЕ** 21](#_Toc73822362)

# **Введение**

Так как увеличивается спрос на недвижимость – замечается рост популярности продаж квартир. Люди хотят получить собственное жильё, это приводит к росту спроса на недвижимость. В целях облегчения работы продаж недвижимости для клиентов была создана БД продажа квартир.

Сотрудникам трудно выполнять бумажную работу с множеством документов. Необходимо улучшить и ускорить работу в агентствах по продажам недвижимости.

Современному человеку трудно представить работу данных учреждения без компьютера, управление экономикой на всех ее уровнях, научные исследования. Целью данной проектной работы будет автоматизация системы управления и создания базы данных для агентства по недвижимости.

В итоге данной научно-исследовательской работы была спроектирована база данных для автоматизированной системы управления продажами квартир. Построена логическая и физическая структура данных и база данных. Внедрена автоматизированная система по учету продаж, что существенно ускорило продажу квартир в меньшие сроки.

# **ГЛАВА 1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ**

Агентство постоянно обслуживают большой поток клиентов, так что скорость обслуживания играет не малую роль. Необходимо быстро заполнять нужные поля для регистрации продажи квартиры, однако в бумажном виде эта работа может занимать очень много времени.

Для этого была создана БД продажа квартир для сети риелторских фирм, чтобы быстро мониторить квартиры, типы квартир, подсчитывать цену.

Так как подсчет цены может значительно понизить скорость обслуживания – разработка БД продажа квартир необходима, так как большая часть процессов станет автоматизирована.

Целью проекта является разработка БД для риелторских фирм для автоматизации продаж квартир, увеличить количество клиентов и прибыль.

# **Постановка задачи проекта**

Для достижения поставленной цели определим следующие задачи проекта:

– провести системный анализ предметной области, изучить полностью процесс работы данной сети риелторских фирм и разработать соответствующую концепцию будущей БД;

– разработать логическую и физическую модель БД, модель, которая призвана облегчить работу по созданию БД, что позволит уменьшить время разработки;

– осуществить реализацию БД на выбранной СУБД, реализация будет осуществляться с помощью СУБД Microsoft Server SQL и языка программирования C#, а также будет представлен код необходимых объектов на языке SQL.

**Вывод**

В данной главе была доказана актуальность разрабатываемой АСУ БД продажи квартир, в связи с долгим временем обслуживания клиентов.

# **ГЛАВА 2. ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ “Продажа ЖД билетов”**

# **Разработка концепции автоматизации продажи билетов**

Разработаем следующую концепцию БД по продаже квартир.

основные таблицы:

– клиенты, хранит информацию о клиентах;

– свойства клиентов, хранит дополнительные свойства о клиентах;

– сотрудники, хранит информацию о сотрудниках, их должности;

– должности сотрудников хранит информацию о различных видах должностей;

– квартиры, хранит информацию о квартире, типе квартиры и свойстве квартиры;

– типы квартир, хранит информацию о типах квартир;

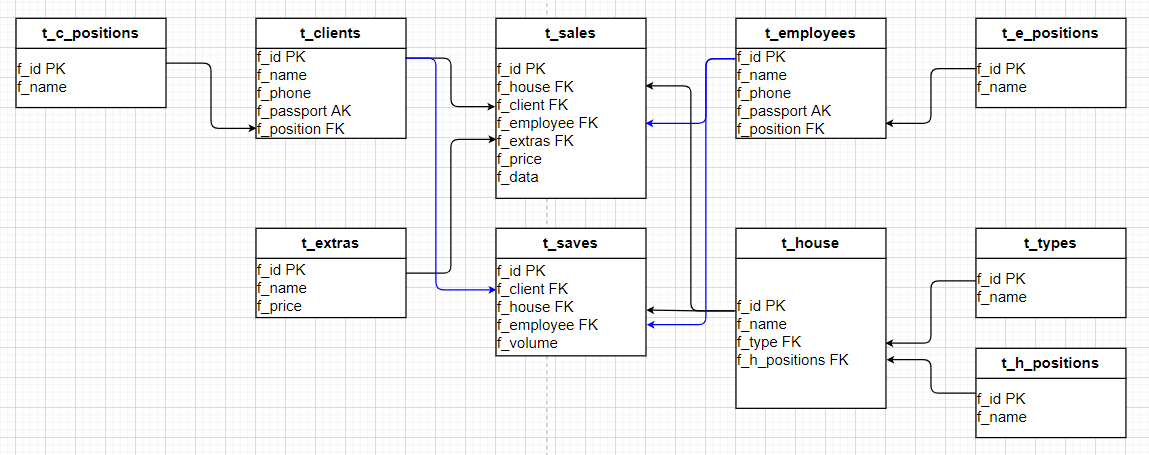
– свойства квартир, хранит информацию о свойствах квартир;

– дополнительные услуги, хранит информацию о дополнительных услугах;

– продажи, хранит информацию о продажах;

– сумма, хранит информацию о всей полученной прибыли.

# **Модель БД**



# Рисунок 1 – модель БД

# **Процесс работы**

При продаже квартиры клиенту заполняются поля: данные о клиенте, номер квартиры, данные о сотруднике, который обслуживал клиента, данные о доп. услуге, цена. По завершению транзакции ставится дата, и считается итоговая цена.

После продажи данные заносятся в таблицу «продажи», где хранится вся история транзакций, и в таблицу «сумма», где хранится информация о прибыли.

# **Структура таблиц**

Структура таблицы включает, что каждый объект должен содержать определенный тип данных. Опишем структуру таблиц.

Сотрудники

|  |  |  |  |
| --- | --- | --- | --- |
| Название поля | Тип данных | Пояснение | Ключи |
| Ключ | INT | Первичный ключ | Primary key |
| Фамилия | NVARCHAR(50) |  |  |
| Имя | NVARCHAR(50) |  |  |
| Отчество | NVARCHAR(50) |  |  |
| Телефон | NVARCHAR(10) |  |  |
| Паспорт | NVARCHAR(10) | Уникальный ключ | Unique key |
| Должность | INT | Внешний ключ | Foreign key |

Должности сотрудников

|  |  |  |  |
| --- | --- | --- | --- |
| Название поля | Тип данных | Пояснение | Ключи |
| Ключ | INT | Первичный ключ | Primary key |
| Название | NVARCHAR(50) | Уникальный ключ | Unique key |

Клиенты

|  |  |  |  |
| --- | --- | --- | --- |
| Название поля | Тип данных | Пояснение | Ключи |
| Ключ | INT | Первичный ключ | Primary key |
| Фамилия | NVARCHAR(50) |  |  |
| Имя | NVARCHAR(50) |  |  |
| Отчество | NVARCHAR(50) |  |  |
| Телефон | NVARCHAR(10) |  |  |
| Паспорт | NVARCHAR(10) | Уникальный ключ | Unique key |
| Должность | INT | Внешний ключ | Foreign key |

Свойства клиентов

|  |  |  |  |
| --- | --- | --- | --- |
| Название поля | Тип данных | Пояснение | Ключи |
| Ключ | INT | Первичный ключ | Primary key |
| Название | NVARCHAR(50) | Уникальный ключ | Unique key |

Типы квартир

|  |  |  |  |
| --- | --- | --- | --- |
| Название поля | Тип данных | Пояснение | Ключи |
| Ключ | INT | Первичный ключ | Primary key |
| Название | NVARCHAR(50) | Уникальный ключ | Unique key |

Свойства квартир

|  |  |  |  |
| --- | --- | --- | --- |
| Название поля | Тип данных | Пояснение | Ключи |
| Ключ | INT | Первичный ключ | Primary key |
| Название | NVARCHAR(50) | Уникальный ключ | Unique key |

Квартиры

|  |  |  |  |
| --- | --- | --- | --- |
| Название поля | Тип данных | Пояснение | Ключи |
| Ключ | INT | Первичный ключ | Primary key |
| Название | NVARCHAR(50) |  |  |
| Свойство | INT | Внешний ключ | Foreign key |
| Тип | INT | Внешний ключ | Foreign key |

Дополнительные услуги

|  |  |  |  |
| --- | --- | --- | --- |
| Название поля | Тип данных | Пояснение | Ключи |
| Ключ | INT | Первичный ключ | Primary key |
| Название | NVARCHAR(50) | Уникальный ключ | Unique key |

Сумма

|  |  |  |  |
| --- | --- | --- | --- |
| Название поля | Тип данных | Пояснение | Ключи |
| Ключ | INT | Первичный ключ | Primary key |
| Клиент | INT | Внешний ключ | Foreign key |
| Сотрудник | INT | Внешний ключ | Foreign key |
| Количество | MONEY |  |  |

Продажи

|  |  |  |  |
| --- | --- | --- | --- |
| Название поля | Тип данных | Пояснение | Ключи |
| Ключ | INT | Первичный ключ | Primary key |
| Квартира | INT | Внешний ключ | Foreign key |
| Клиент | INT | Внешний ключ | Foreign key |
| Сотрудник | INT | Внешний ключ | Foreign key |
| Доп. услуги | INT | Внешний ключ | Foreign key |
| Цена | MONEY |  |  |
| Дата | DATE |  |  |

# **Вывод**

В данной главе были разработаны все основные и дополнительных таблицы, которые используется в разрабатываемой БД.

# **ГЛАВА 3. ПРОГРАММНАЯ РЕАЛИЗАЦИЯ**

# **Разработка контрольного примера**

Для проверки возможности представлять информацию о продажах квартир в спроектированной структуре данных, разработаем контрольный пример.

Введем информацию во все поля и посмотрим правильность выполнения работы

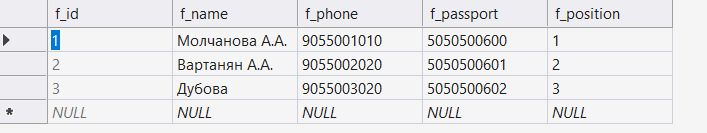


Рисунок 2 – таблица Сотрудники

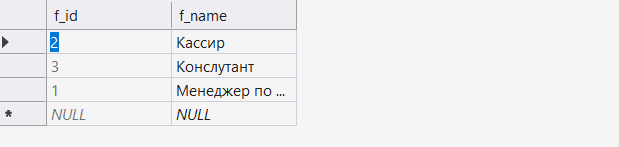


Рисунок 3 – таблица Должности сотрудников

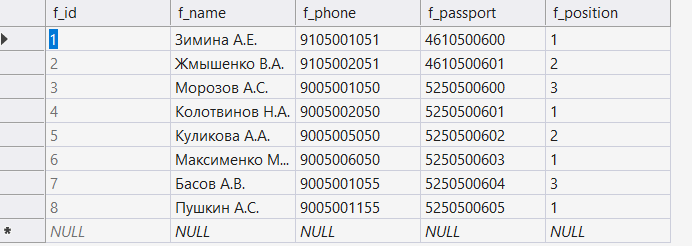


Рисунок 4 – таблица Клиенты

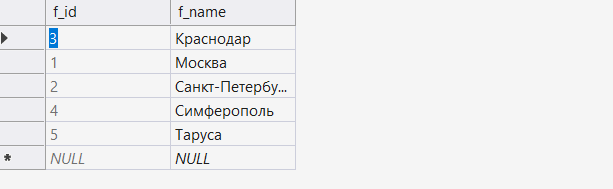


Рисунок 5 – таблица Свойства клиентов



Рисунок 6 – таблица Типы квартир

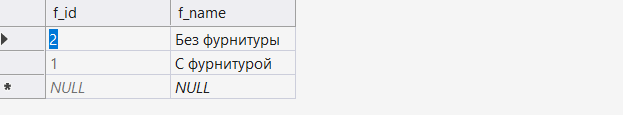


Рисунок 7 – таблица Свойства квартир

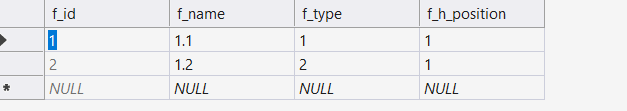


Рисунок 8 – таблица Квартиры



Рисунок 9 – таблица Дополнительные услуги

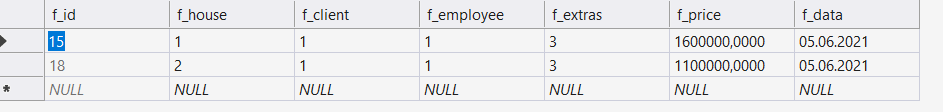


Рисунок 10 – таблица Продажи

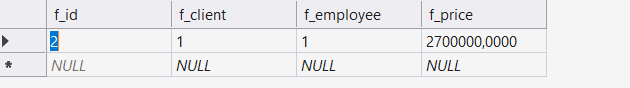


Рисунок 11 – таблица Сумма

Этот контрольный пример был заполнен в программе Visual Studio 2019 c помощью Microsoft Server SQL.

# **3.2 Разработка алгоритмов и текстов программ**

Разработка базы данных должна начинаться с построения структуры таблиц и определения первичных, внешних и альтернативных ключей.

Все ключи в таблицах именуются следующим образом: Первичный ключ = Primary key, Внешний ключ = Foreign key, уникальный ключ = Unique key.

Создадим БД «FLAT»

# **3.3 Создание таблиц**

Создадим таблицу «Сотрудники».

CREATE TABLE [dbo].[t\_employees] (

[f\_id] INT IDENTITY (1, 1) NOT NULL,

[f\_name] NVARCHAR (50) NOT NULL,

[f\_phone] NVARCHAR (50) NULL,

[f\_passport] NVARCHAR (10) NOT NULL,

[f\_position] INT NULL,

PRIMARY KEY CLUSTERED ([f\_id] ASC),

CONSTRAINT [c\_fk\_t\_employees\_f\_position] FOREIGN KEY ([f\_position]) REFERENCES [dbo].[t\_e\_positions] ([f\_id])

);

GO

CREATE UNIQUE NONCLUSTERED INDEX [idx\_employees\_passport]

ON [dbo].[t\_employees]([f\_passport] ASC);

Создадим таблицу «Должности сотрудников».

CREATE TABLE [dbo].[t\_e\_positions] (

[f\_id] INT IDENTITY (1, 1) NOT NULL,

[f\_name] NVARCHAR (50) NOT NULL,

PRIMARY KEY CLUSTERED ([f\_id] ASC)

);

GO

CREATE UNIQUE NONCLUSTERED INDEX [idx\_e\_positions\_name]

ON [dbo].[t\_e\_positions]([f\_name] ASC);

Создадим таблицу «Клиенты» с 1 триггером: на контроль записей, номер телефона клиента не должен повторяться.

CREATE TABLE [dbo].[t\_clients] (

[f\_id] INT IDENTITY (1, 1) NOT NULL,

[f\_name] NVARCHAR (50) NOT NULL,

[f\_phone] NVARCHAR (50) NULL,

[f\_passport] NVARCHAR (10) NOT NULL,

[f\_position] INT NULL,

PRIMARY KEY CLUSTERED ([f\_id] ASC),

CONSTRAINT [c\_fk\_t\_clients\_f\_position] FOREIGN KEY ([f\_position]) REFERENCES [dbo].[t\_c\_positions] ([f\_id])

);

GO

CREATE UNIQUE NONCLUSTERED INDEX [idx\_clients\_passport]

ON [dbo].[t\_clients]([f\_passport] ASC);

GO

CREATE TRIGGER tr\_clients

ON t\_clients

INSTEAD OF INSERT

AS

DECLARE

@row\_id int = 0,

@row\_phone NVARCHAR(50),

@row\_count int = 0

BEGIN

SET @row\_id = (select f\_id from inserted);

SET @row\_phone = (select f\_phone from inserted);

SET @row\_count = (SELECT count(f\_id) from t\_clients where f\_phone = @row\_phone);

IF @row\_count = 0

BEGIN

INSERT INTO

t\_clients(f\_name, f\_phone, f\_passport, f\_position)

VALUES

((select f\_name from inserted), (select f\_phone from inserted), (select f\_passport from inserted), (select f\_position from inserted))

END

END

Создадим таблицу «Свойства клиентов».

CREATE TABLE [dbo].[t\_c\_positions] (

[f\_id] INT IDENTITY (1, 1) NOT NULL,

[f\_name] NVARCHAR (50) NOT NULL,

PRIMARY KEY CLUSTERED ([f\_id] ASC)

);

GO

CREATE UNIQUE NONCLUSTERED INDEX [idx\_c\_positions\_name]

ON [dbo].[t\_c\_positions]([f\_name] ASC);

Создадим таблицу «Типы квартир».

CREATE TABLE [dbo].[t\_types] (

[f\_id] INT IDENTITY (1, 1) NOT NULL,

[f\_name] NVARCHAR (50) NOT NULL,

PRIMARY KEY CLUSTERED ([f\_id] ASC)

);

GO

CREATE UNIQUE NONCLUSTERED INDEX [idx\_t\_types]

ON [dbo].[t\_types]([f\_name] ASC);

Создадим таблицу «Свойства квартир».

CREATE TABLE [dbo].[t\_h\_positions] (

[f\_id] INT IDENTITY (1, 1) NOT NULL,

[f\_name] NVARCHAR (50) NOT NULL,

PRIMARY KEY CLUSTERED ([f\_id] ASC)

);

GO

CREATE UNIQUE NONCLUSTERED INDEX [idx\_t\_h\_positions]

ON [dbo].[t\_h\_positions]([f\_name] ASC);

Создадим таблицу «Квартиры».

CREATE TABLE [dbo].[t\_houses] (

[f\_id] INT IDENTITY (1, 1) NOT NULL,

[f\_name] NVARCHAR (50) NOT NULL,

[f\_type] INT NULL,

[f\_h\_position] INT NULL,

PRIMARY KEY CLUSTERED ([f\_id] ASC),

CONSTRAINT [c\_fk\_t\_houses\_f\_type] FOREIGN KEY ([f\_type]) REFERENCES [dbo].[t\_types] ([f\_id]),

CONSTRAINT [c\_fk\_t\_houses\_f\_h\_position] FOREIGN KEY ([f\_h\_position]) REFERENCES [dbo].[t\_h\_positions] ([f\_id])

);

GO

CREATE UNIQUE NONCLUSTERED INDEX [idx\_houses\_name]

ON [dbo].[t\_houses]([f\_name] ASC);

Создадим таблицу «Дополнительные услуги».

CREATE TABLE [dbo].[t\_extras] (

[f\_id] INT IDENTITY (1, 1) NOT NULL,

[f\_name] NVARCHAR (50) NOT NULL,

[f\_price] MONEY NOT NULL,

PRIMARY KEY CLUSTERED ([f\_id] ASC)

);

GO

CREATE UNIQUE NONCLUSTERED INDEX [idx\_t\_extras]

ON [dbo].[t\_extras]([f\_name] ASC);

Создадим таблицу «Продажи квартир» с 2 триггерами: на подсчет цены с учетом доп. услуг, на подсчёт общей прибыли в таблице «Сумма»

CREATE TABLE [dbo].[t\_sales] (

[f\_id] INT IDENTITY (1, 1) NOT NULL,

[f\_house] INT NOT NULL,

[f\_client] INT NOT NULL,

[f\_employee] INT NOT NULL,

[f\_extras] INT NULL,

[f\_price] MONEY NULL,

[f\_data] DATE NULL,

PRIMARY KEY CLUSTERED ([f\_id] ASC),

CONSTRAINT [c\_fk\_t\_sales\_f\_client] FOREIGN KEY ([f\_client]) REFERENCES [dbo].[t\_clients] ([f\_id]),

CONSTRAINT [c\_fk\_t\_sales\_f\_employee] FOREIGN KEY ([f\_employee]) REFERENCES [dbo].[t\_employees] ([f\_id]),

CONSTRAINT [c\_fk\_t\_sales\_f\_extras] FOREIGN KEY ([f\_extras]) REFERENCES [dbo].[t\_extras] ([f\_id])

);

GO

CREATE UNIQUE NONCLUSTERED INDEX [idx\_t\_sales\_house]

ON [dbo].[t\_sales]([f\_house] ASC);

Создадим таблицу «Сумма».

CREATE TABLE [dbo].[t\_saves] (

[f\_id] INT IDENTITY (1, 1) NOT NULL,

[f\_client] INT NOT NULL,

[f\_employee] INT NOT NULL,

[f\_price] MONEY NULL,

PRIMARY KEY CLUSTERED ([f\_id] ASC),

CONSTRAINT [c\_fk\_t\_sum\_f\_client] FOREIGN KEY ([f\_client]) REFERENCES [dbo].[t\_clients] ([f\_id]),

CONSTRAINT [c\_fk\_t\_sum\_f\_employee] FOREIGN KEY ([f\_employee]) REFERENCES [dbo].[t\_employees] ([f\_id])

);

# **3.4 Создание представлений**

Создадим представление «Сотрудники»

CREATE VIEW [dbo].[v\_employees]

AS SELECT

t\_employees.f\_id ID,

t\_employees.f\_name ФИО,

t\_employees.f\_phone Телефон,

t\_employees.f\_passport Паспорт,

t\_e\_positions.f\_name Должность

FROM [t\_employees],[t\_e\_positions]

WHERE t\_employees.f\_position = t\_e\_positions.f\_id;

Создадим представление «Клиенты»

CREATE VIEW [dbo].[v\_clients]

AS SELECT

t\_clients.f\_id ID,

t\_clients.f\_name ФИО,

t\_clients.f\_phone Телефон,

t\_clients.f\_passport Паспорт,

t\_c\_positions.f\_name Статус

FROM [t\_clients],[t\_c\_positions]

WHERE t\_clients.f\_position = t\_c\_positions.f\_id;

Создадим представление «Свойства клиентов»

CREATE VIEW [dbo].[v\_c\_positions]

AS SELECT

t\_c\_positions.f\_id ID,

t\_c\_positions.f\_name Название

FROM [t\_c\_positions]

Создадим представление «Квартиры»

CREATE VIEW [dbo].[v\_houses]

AS SELECT

t\_houses.f\_id ID,

t\_houses.f\_name НомерКвартиры,

t\_types.f\_name Тип,

t\_h\_positions.f\_name Свойство

FROM [t\_houses],[t\_types],[t\_h\_positions]

WHERE t\_houses.f\_type = t\_types.f\_id and

t\_houses.f\_h\_position = t\_h\_positions.f\_id;

Создадим представление «Типы квартир»

CREATE VIEW [dbo].[v\_types]

AS SELECT

t\_types.f\_id ID,

t\_types.f\_name Название

FROM [t\_types]

Создадим представление «Свойства квартир»

CREATE VIEW [dbo].[v\_h\_positions]

AS SELECT

t\_h\_positions.f\_id ID,

t\_h\_positions.f\_name Название

FROM [t\_h\_positions]

Создадим представление «Дополнительные услуги»

CREATE VIEW [dbo].[v\_extras]

AS SELECT

t\_extras.f\_id ID,

t\_extras.f\_name Название

FROM [t\_extras]

Создадим представление «Продажи»

CREATE VIEW [dbo].[v\_sales]

AS SELECT

t\_sales.f\_id ID,

t\_houses.f\_name Дом,

t\_clients.f\_name Клиент,

t\_employees.f\_name Сотрудник,

t\_extras.f\_name ДопУслуга,

t\_sales.f\_price Цена,

t\_sales.f\_data Дата

FROM [t\_sales],[t\_houses],[t\_clients],[t\_employees],[t\_extras]

WHERE t\_sales.f\_house = t\_houses.f\_id and

t\_sales.f\_client = t\_clients.f\_id and

t\_sales.f\_employee = t\_employees.f\_id and

t\_sales.f\_extras = t\_extras.f\_id;

Создадим представление «Сумма»

CREATE VIEW [dbo].[v\_saves]

AS SELECT

t\_saves.f\_id ID,

t\_clients.f\_name Клиент,

t\_employees.f\_name Сотрудник,

t\_saves.f\_price КоличествоСредств

FROM [t\_saves],[t\_clients],[t\_employees]

WHERE t\_saves.f\_client = t\_clients.f\_id and

t\_saves.f\_employee = t\_employees.f\_id;

# **3.5 Создание процедуры**

Создадим процедуру «Клиенты»

CREATE PROCEDURE [dbo].[p\_clients]

@new\_name NVARCHAR(50),

@new\_phone NVARCHAR(50),

@new\_passport NVARCHAR(10),

@new\_position INT

AS

BEGIN

INSERT INTO

t\_clients(f\_name, f\_phone,f\_passport,f\_position)

VALUES

(@new\_name,@new\_phone,@new\_passport,@new\_position);

END

Создадим процедуру «Продажи»

CREATE PROCEDURE [dbo].[p\_sales]

@new\_house INT,

@new\_client INT,

@new\_employee INT,

@new\_extras INT,

@new\_price MONEY

AS

BEGIN

INSERT INTO

t\_sales(f\_house,f\_client,f\_employee, f\_extras, f\_price, f\_data)

VALUES

(@new\_house, @new\_client,@new\_employee, @new\_extras, @new\_price, SYSDATETIME());

END

# **3.6 Основной код интерфейса приложения**

using System;

using System.Windows.Forms;

using System.Configuration;

using System.Data;

using System.Data.SqlClient;

namespace TBD\_HOUSE {

public partial class Form1 : Form

{

private SqlConnection sqlConnection = null;

private SqlDataAdapter dataAdapter = null;

private DataSet dataSet = null;

private SqlCommand command = null;

private void view(string str)

{

dataAdapter = new SqlDataAdapter(str, sqlConnection);

dataSet = new DataSet();

dataAdapter.Fill(dataSet);

dataGridView1.DataSource = dataSet.Tables[0];

dataAdapter.Dispose();

dataSet.Dispose();

}

private void procedure(string str)

{

command = new SqlCommand(str, sqlConnection);

if (command.ExecuteNonQuery().ToString() == "-1")

{

MessageBox.Show("Ошибка!");

}

else

{

MessageBox.Show("Успешно!");

}

command.Dispose();

}

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

sqlConnection = new SqlConnection(ConfigurationManager.ConnectionStrings["FLATDB"].ConnectionString);

sqlConnection.Open();

if (sqlConnection.State == ConnectionState.Open)

{

MessageBox.Show("Подключение к БД установлено");

}

}

private void button1\_Click(object sender, EventArgs e)

{

procedure($"EXECUTE p\_sales @new\_house = N'{textBox1.Text}', @new\_client = N'{textBox2.Text}', @new\_employee = N'{textBox3.Text}', @new\_extras = N'{textBox4.Text}', @new\_price = N'{textBox5.Text}';");

}

private void button2\_Click(object sender, EventArgs e)

{

procedure($"EXECUTE p\_clients @new\_name = N'{textBox6.Text}', @new\_phone = N'{textBox7.Text}', @new\_passport = N'{textBox8.Text}', @new\_position = N'{textBox9.Text}';");

}

private void button4\_Click(object sender, EventArgs e)

{

view("select \* from v\_clients");

}

private void button5\_Click(object sender, EventArgs e)

{

view("select \* from v\_houses");

}

private void button6\_Click(object sender, EventArgs e)

{

view("select \* from v\_employees");

}

private void button7\_Click(object sender, EventArgs e)

{

view("select \* from v\_types");

}

private void button8\_Click(object sender, EventArgs e)

{

view("select \* from v\_h\_positions");

}

private void button9\_Click(object sender, EventArgs e)

{

view("select \* from v\_c\_positions");

}

private void button10\_Click(object sender, EventArgs e)

{

view("select \* from v\_extras");

}

private void button11\_Click(object sender, EventArgs e)

{

view("select \* from v\_sales");

}

private void button12\_Click(object sender, EventArgs e)

{

view("select \* from v\_saves");

}

}

}

# **3.6 Разработка оконных форм взаимодействия с конечным пользователем**

Продажа квартир описывается оконной формой взаимодействия с пользователем.

Запись продажи квартиры.

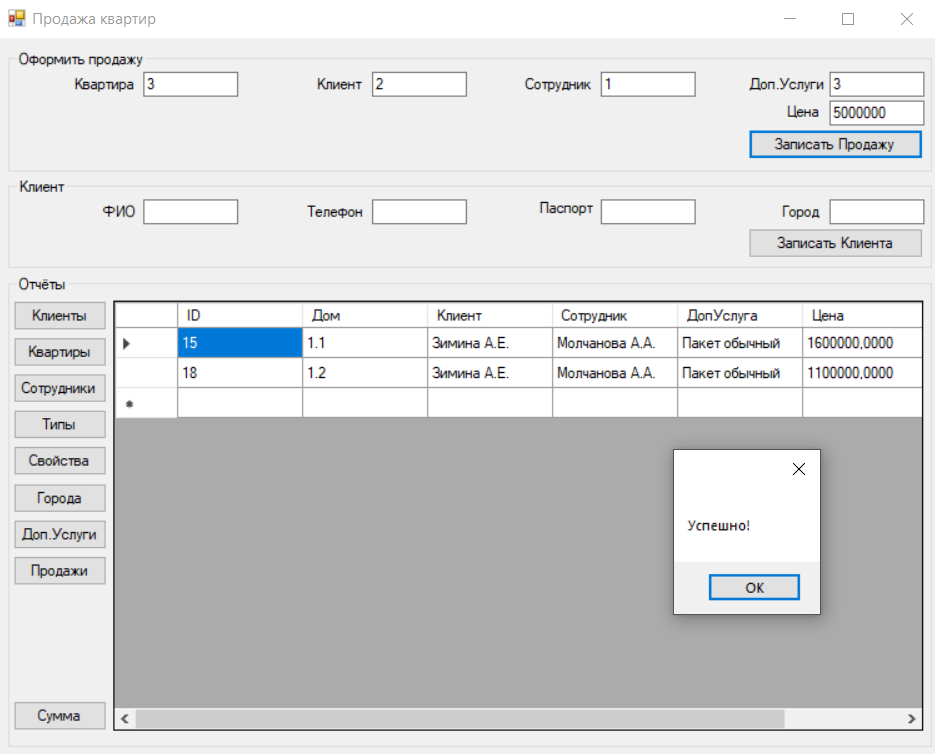


Рисунок 1 – Форма «БД продажи квартир»

Просмотр отчётов.

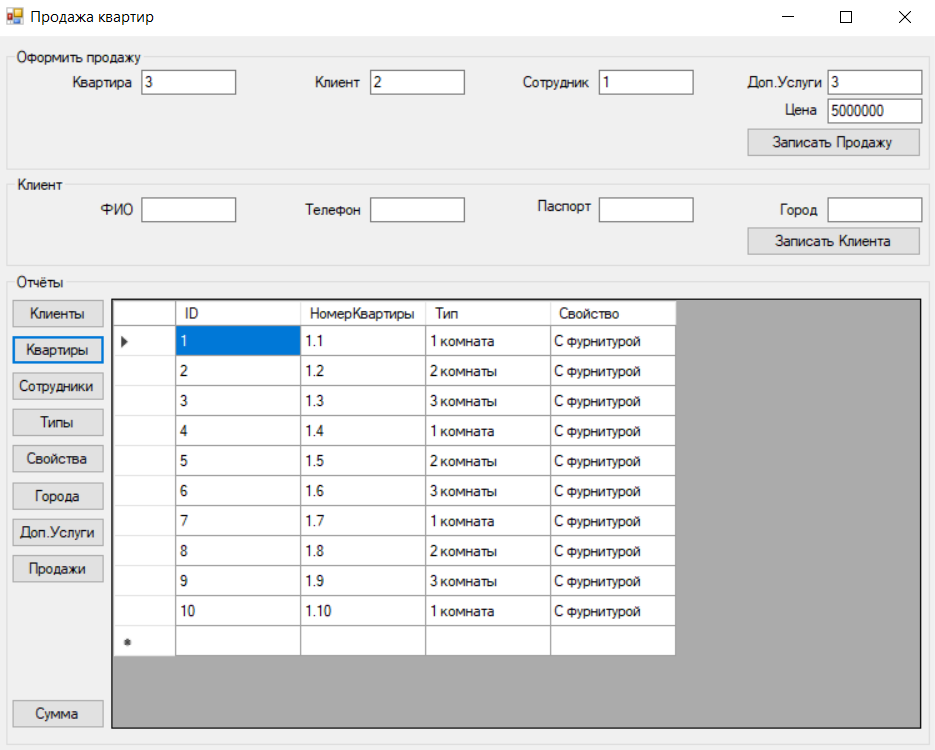
****

Рисунок 2 – Форма «БД продажи квартир»

Просмотр суммы.

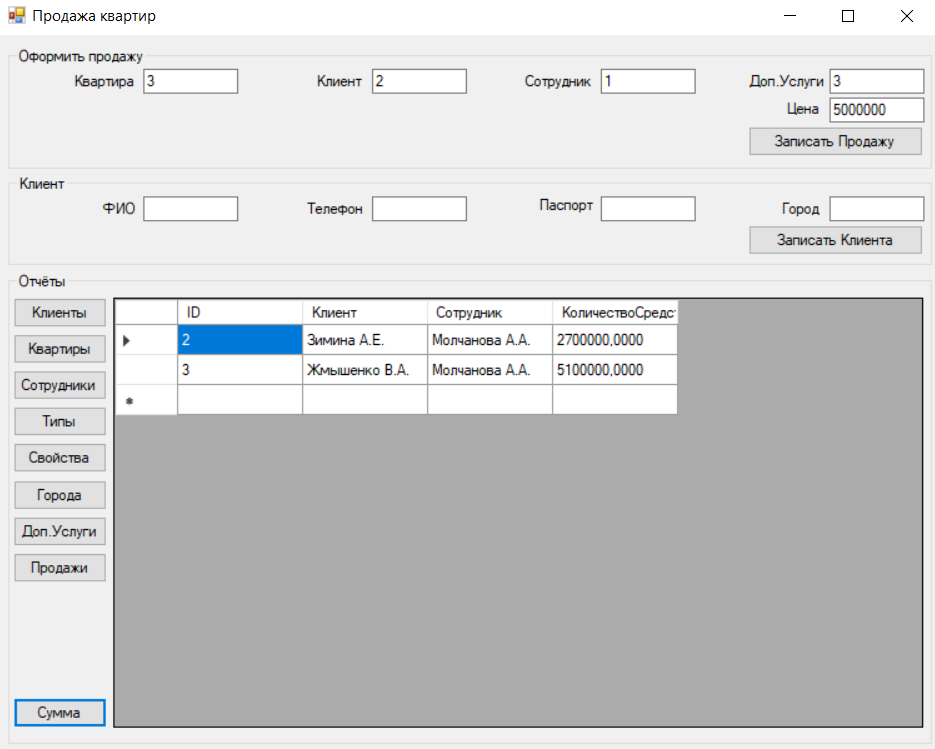


Рисунок 3 – Форма «БД продажи квартир»

# **Вывод**

В этой главе реализовали оконную форму для конечного пользователя, написали программу на C# и MSQL.

# **ЗАКЛЮЧЕНИЕ**

Разработали базу данных по продаже квартир, автоматизировали подсчет цены, реализовали оконную форму для конечного пользователя. Использовали исходный код MSQL для создания таблиц, представлений, процедур и триггеров. Разработали логическую модель БД.

Автоматизировали процесс подсчета цены.