Министерство образования и науки Российской Федерации

ФГБО УВО «Севастопольский государственный университет»

Институт информационных технологий и управления в технических системах

Кафедра

информационных технологий

КУРСОВОЙ ПРОЕКТ ПО ДИСЦИПЛИНЕ

«УПРАВЛЕНИЕ ДАННЫМИ»

БАЗА ДАННЫХ ГОСТЕВОЙ СЕТИ «BE MY GUEST»

Пояснительная записка

Листов

ПРОВЕРИЛ

Старший преподаватель кафедры ИТ

\_\_\_\_\_\_\_\_\_Гущин И. К.

«\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_2015г.

РАЗРАБОТАЛИ:

Студент группы ИТ(б)-32д

\_\_\_\_\_\_\_\_\_Серба А.В.

«\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_2015г.

Студент группы ИТ(б)-31д

\_\_\_\_\_\_\_\_\_Малиновский А.А.

«\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_2015г.

Севастополь

2015

АННОТАЦИЯ

В документе описана поэтапная разработка веб-приложения, сеть гостеприимства «Be my guest». В пояснительной записке описывается анализ предметной области, разработка логический схемы базы данных, реализация базы данных, описание программного обеспечения.

ОГЛАВЛЕНИЕ

[ВВЕДЕНИЕ 4](#_Toc419460599)

[1 ПОСТАНОВКА ЗАДАЧИ 5](#_Toc419460600)

[1.1 Техническое задание 5](#_Toc419460601)

[2 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 7](#_Toc419460602)

[2.1 Анализ отношений между объектами 8](#_Toc419460603)

[2.2 Построение ER-диаграммы 9](#_Toc419460604)

[3 РАЗРАБОТКА ЛОГИЧЕСКОЙ СХЕМЫ БАЗЫ ДАННЫХ 10](#_Toc419460605)

[3.1 Обоснование выбора реляционной модели 10](#_Toc419460606)

[3.2 Построение исходной концептуальной модели 11](#_Toc419460607)

[3.3 Нормализация исходной модели 13](#_Toc419460608)

[3.3.1 Первая нормальная форма 13](#_Toc419460609)

[3.3.2 Вторая нормальная форма 14](#_Toc419460610)

[3.3.3 Третья нормальная форма 18](#_Toc419460611)

[4 РЕАЛИЗАЦИЯ БАЗЫ ДАННЫХ 20](#_Toc419460612)

[4.1 Обоснование выбора СУБД и языка программирования 20](#_Toc419460613)

[4.2 Разграничение прав доступа 21](#_Toc419460614)

[4.3 Физическая реализация базы данных 21](#_Toc419460615)

[4.4 Описание интерфейса пользователя 26](#_Toc419460616)

[4.5 Тестирование базы данных 32](#_Toc419460617)

[5. ИССЛЕДОВАНИЕ ИНФОРМАЦИОННЫХ ПАРАМЕТРОВ БД 34](#_Toc419460618)

[ЗАКЛЮЧЕНИЕ 36](#_Toc419460619)

[БИБЛИОГРАФИЧЕСКИЙ СПИСОК 37](#_Toc419460620)

[ПРИЛОЖЕНИЕ А 38](#_Toc419460621)

Т. Контр.

Т. Контр.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

3

КУРСОВОЙ ПРОЕКТ

Разраб.

Серба/Малиновский

Провер.

*Гущин И. К.*

Н. Контр.

Утв.

ПОЯСНИТЕЛЬНАЯ

ЗАПИСКА

Лит.

Листов

Группа ИТ(б)-32д

Группа ИТ(б)-31д

# ВВЕДЕНИЕ

Современные информационные системы, основанные на концепции баз данных, характеризуются большими объемами хранимой информации, их сложной организацией, необходимостью удовлетворять разнообразные требования пользователей. Важным компонентом этой концепции является единая методология проектирования баз данных.

В ходе выполнения курсового проектирования необходимо разработать логическую схему базы данных сети гостеприимства и реализовать веб- приложение, позволяющее путешественникам подготавливаться к путешествиям в незнакомые города, находить ночлег и новых знакомых.

Пояснительная записка состоит из пяти разделов. В разделе «Постановка задачи» содержится техническое задание, согласно которому будет разрабатываться программный продукт. В разделе «Анализ предметной области» выделены сущности, определены атрибуты, построена внешняя модель, ER-диаграмма и EER-диаграмма. В разделе «Разработка логической схема базы данных» описано обоснование выбора реляционной модели и нормализация исходной модели. В разделе «разработка базы данных» описано обоснование выбора СУБД, языка программирования и фреймворка, защита данных, разграничение прав доступа, физическая реализация базы данных, описание интерфейса пользователя и тестирование базы данных. В разделе «Исследование информационных параметров БД» выполнен расчет эксплуатационных характеристик.

# 1 ПОСТАНОВКА ЗАДАЧИ

Целью курсового проектирования является закрепление, систематизация и обобщение знаний по дисциплине «Управление данными», приобретение практических навыков разработки баз данных. Для достижения целей курсового проектирования следующие задачи были разделены между студентами таким образом:

Малиновский Александр:

* Анализ предметной области;
* Разработка логической схемы;
* Разработка база данных;
* Front-end разработка приложения;
* Тестирование физической БД;

Серба Анна:

* Выбор СУБД и средств разработки;
* Разработка приложения физической БД;
* Front-end и Back-end разработка приложения;
* Тестирование физической БД;
* Тестирование приложения;

Поставленные задачи будут решатся в соответствии с техническим заданием.

## Техническое задание

В рамках данного курсового проекта разрабатывается система сети гостеприимства «Be my guest». Данная система будет являться hospitality network он-лайн службой, которая поможет людям с ограниченными средствами из разных уголков планеты планировать свои путешествия. В рамках системы будут разработаны следующие интерфейсы:

1. Интерфейс «Администратор»;

2. Интерфейс «Пользователь»;

В интерфейсе администратора присутствуют функции добавления и редактирование учетных записей менеджеров по продажам и изменение стоимости недвижимости за 1м2.

В интерфейсе менеджера по продажам будут присутствовать функции для обслуживания клиентов, а именно: управление учетными записями клиентов (добавление, редактирование), поиск недвижимости по заданным критериям, добавление недвижимости в базу данных и редактирование характеристик уже присутствующей недвижимости, оформление договоров купли-продажи и аренды недвижимости. При работе с недвижимостью будет возможность рассчитать стоимость аренды и продажи автоматически. Так же будет реализована функция переноса недвижимости из таблицы аренда в таблицу продажа и наоборот.

В интерфейсе клиента будут присутствовать функции поиска недвижимости по заданным критериям и просмотр заключенных договоров купли-продажи и аренды.

В интерфейсе «клиент без авторизации» будет присутствовать возможность поиска недвижимости по заданным критериям. Данный интерфейс предназначен для пользователей, которые еще не являются клиентами фирмы, но желают выполнить поиск доступной недвижимости.

Основной задачей при реализации интерфейсов это повышение скорости работы пользователей системы и автоматическая проверка корректности вводимых данных. Для реализации этой задачи в интерфейсах должна присутствовать продвинутая система поиска недвижимости, позволяющая выполнять поиск сразу по нескольким произвольным параметрам, а интерфейсы для манипулирования данными должны обнаруживать и сообщать об ошибках вводимых данных в режиме реального времени.

Поскольку предметная область предполагает хранение данных о различных взаимосвязанных сущностях возникает естественное ограничение: невозможно удалить данные не нарушив ссылочную целостность. По этой причине база данных будет ориентирована на накопление записей, а их удаление не будет допускаться.

# 2 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

Разрабатываемая база данных содержит сведения о пользователе, переписке между пользователями, фотографии пользователей, мест, достопримечательностей, квартир и их описания, группы пользователей с общими интересами, адреса, договоры о встречах и путешествиях.

Исходя из поставленной задачи выделим сущности. Сущность – любой различимый объект, информацию о котором необходимо хранить в базе данных. Поэтому для решаемой задачи можно определить следующие сущности, которые в дальнейшем будут использоваться для построения внешней модели проектируемой базы данных:

* Пользователь;
* Сообщения;
* Фотоальбом пользователя;
* Группы;
* Фотоальбом групы;
* Событие;
* Участники события;
* Путешествия;
* Места;
* Апартаменты.

На основании назначения системы у сущностей можно определить следующие атрибуты:

* Пользователь (Имя, Фамилия, эл-почта, паспортные данные, телефон, логин, пароль);
* Недвижимость на продажу (стоимость, продавец);
* Недвижимость в аренду (стоимость в месяц, арендодатель);
* Описание недвижимости (тип недвижимости, общая площадь, жилая площадь, площадь приусадебного участка, количество жилых комнат, тип планировки, тип постройки, год постройки, общее количество этажей, этаж, наличие ремонта, санузел, проведен газ, проведена канализация, подача воды, интернет, городской телефон, бытовая техника, отопление, наличие лифта, объекты инфраструктуры, вид из окна);
* Стоимость за м2 (тип планировки, стоимость м2);
* Договор продажи (Номер договора, покупатель, дата подписания);
* Договор аренды (Номер договора, арендатор, дата подписания, дата начала, дата окончания);
* Менеджер по продажам (полное имя, логин, пароль, уволен).

## 2.1 Анализ отношений между объектами

Основными элементами инфологической модели является сущность и связи между ними. Внешнюю модель базы данных можно представить следующим образом (рисунок 2.1).

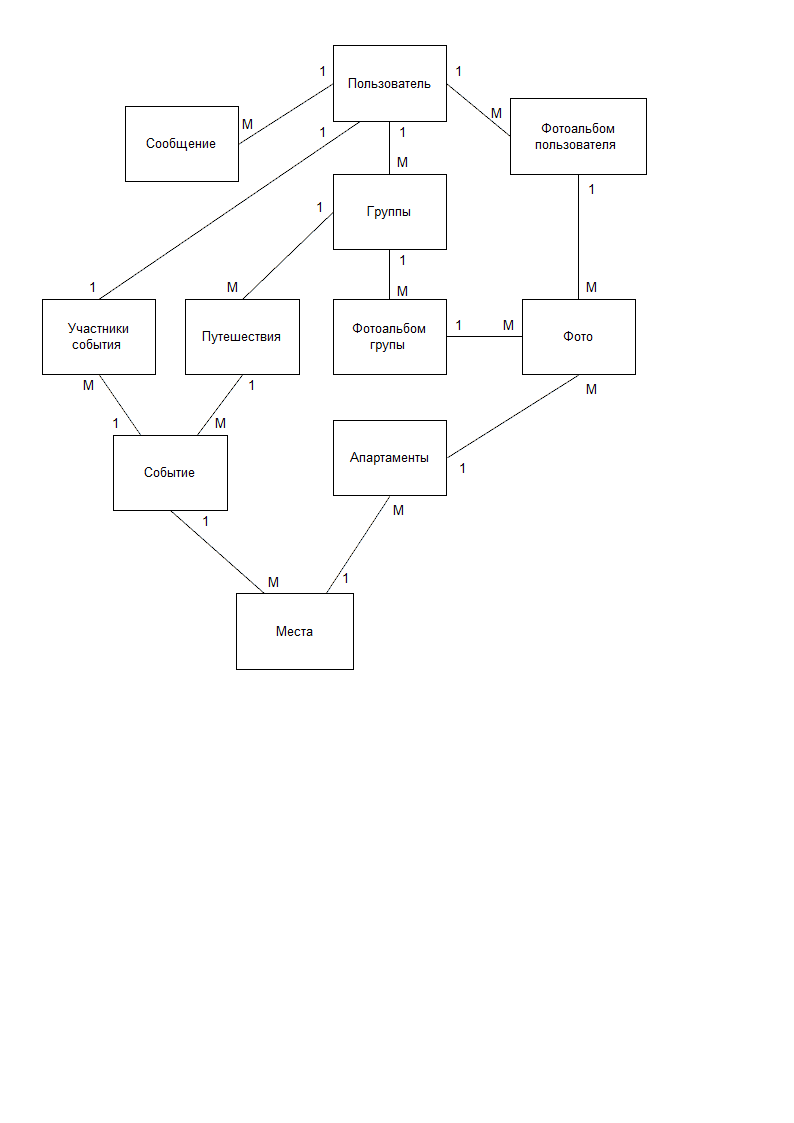


Рисунок 2.1 – Внешняя модель

## 2.2 Построение EER-диаграммы

ER-диаграмма отображает внешнюю модель базы данных. Для сущностей определим свойства и отношения между ними, построив таким образом ER-диаграмму (рисунок 2.2).

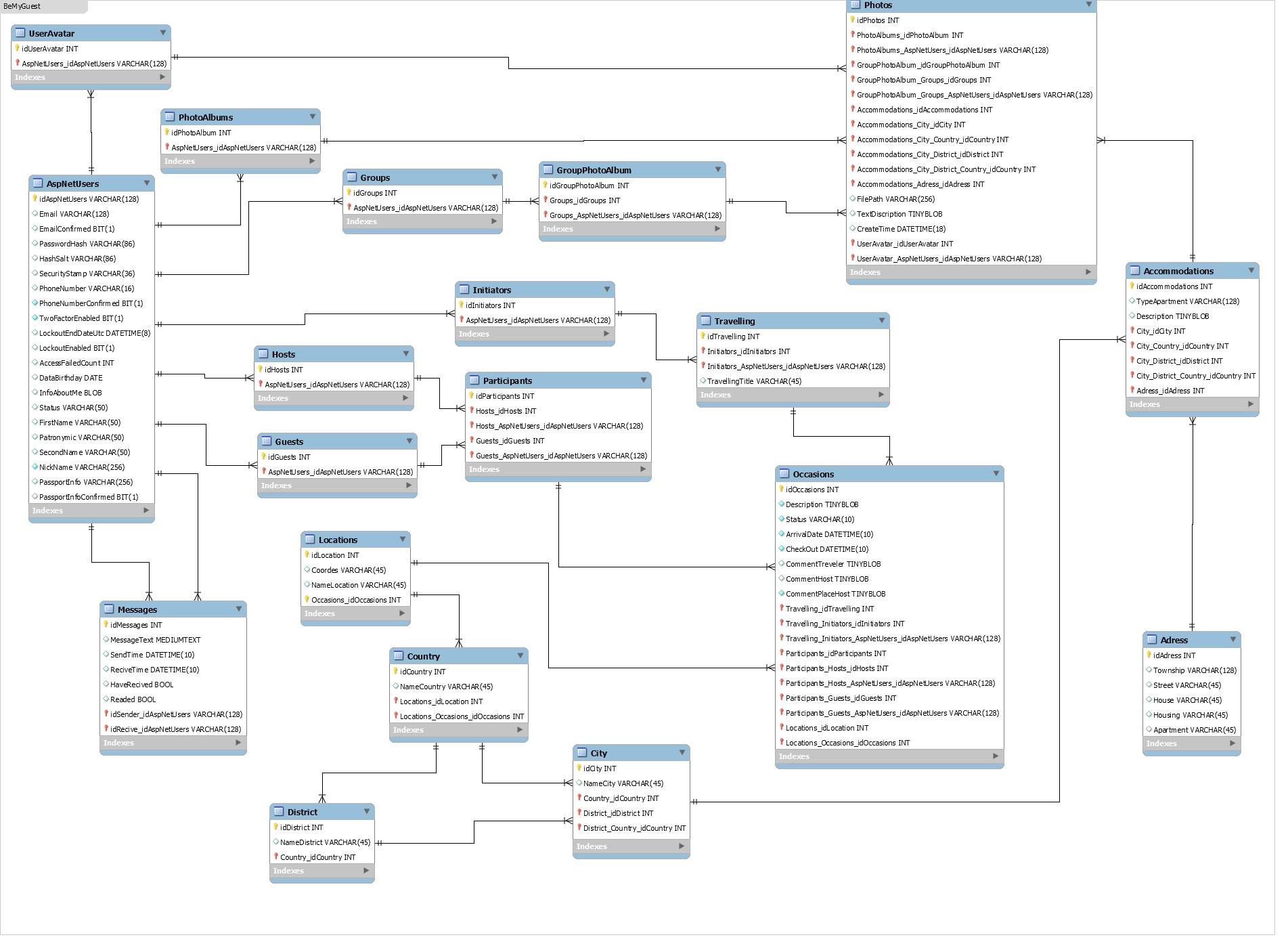


Рисунок 2.2 – EER-диаграмма

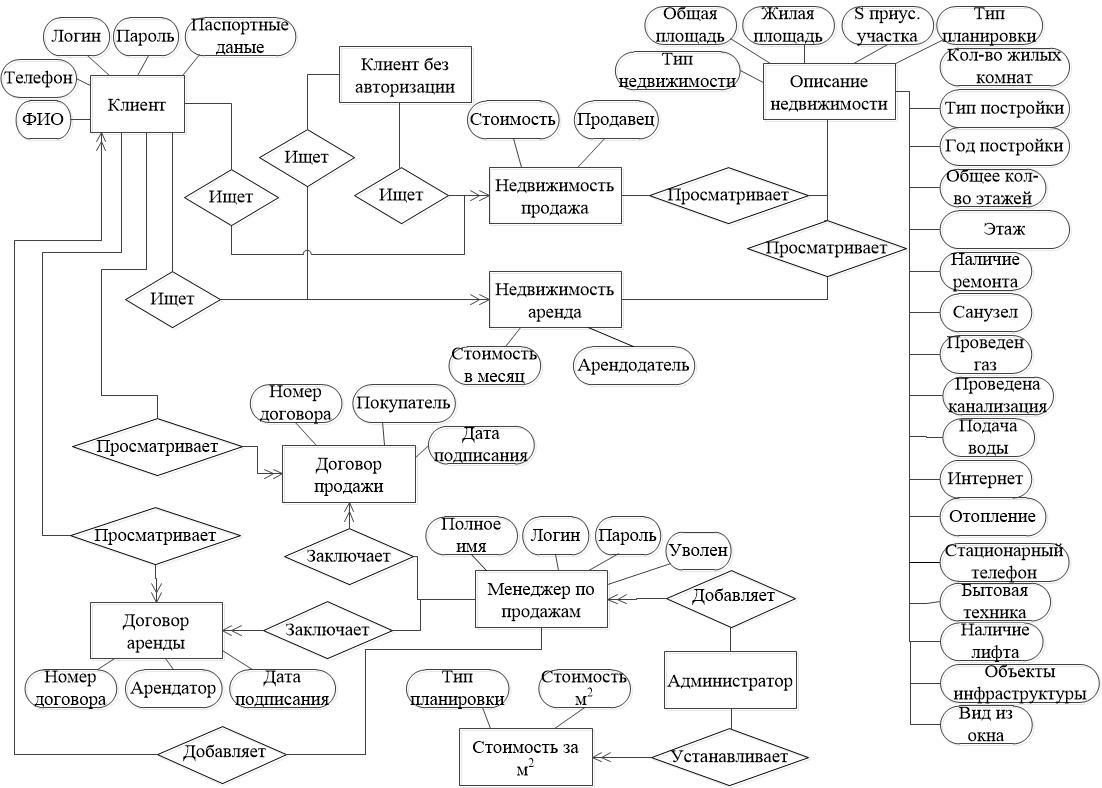


Рисунок 2.2 – ER-диаграмма

# 3 РАЗРАБОТКА ЛОГИЧЕСКОЙ СХЕМЫ БАЗЫ ДАННЫХ

## 3.1 Обоснование выбора реляционной модели

Любую модель данных можно рассматривать как объект, который характеризуется своими свойствами (параметрами). Существуют три основных типа моделей данных – реляционная, иерархическая и сетевая.

Иерархическая модель данных характеризуется высокой избыточностью так как частично дублируются отношения. Как следствие из этого иерархическая модель не имеет механизма поддержки целостности данных.

Основной структурой в сетевых моделях данных является «сеть». При таком представлении существует несколько входов в сеть это приводит к неоднозначности доступа к данным. Так же в сетевой модели возникает сложность в установлении и проверке целостности данных.

Иерархическая и сетевая модель данных оптимизированы на чтение информации из базы данных, но не на запись. Эта особенность обусловлена самой моделью данных.

В отличие от иерархической и сетевой модели, в реляционной модели представление данных не зависит от способа их физической организации. Реляционная модель оперирует с плоскими таблицами, а они являются одним из простейших способов представления данных. Таблица представляет собой множество кортежей. Если таблица имеет n - столбцов, то ее называют отношением степени n. Таблица, состоящая из 2-х столбцов, называется бинарным отношением; из 3-х столбцов - тернарным отношением; из n столбцов - n-арным отношением. С логической точки зрения реляционная база данных – это множество двумерных таблиц с операциями извлечения и объединения столбцов.

При реализации малых баз данных (1000 записей) и средних (10000 записей) используют реляционную модель, а при реализации больших баз данных (100000 записей) используют сетевую или иерархическую структуру. Основываясь на описанных выше преимуществах и недостатках моделей, а также на том что проектируемая база данных относится к малым базам данных было принято решение использовать при реализации БД реляционную модель данных.

## 3.2 Построение исходной концептуальной модели

После построения внешней схемы базы данных переходим к следующему этапу проектирования – построение концептуальной модели. В результате перехода от внешней модели к концептуальной получим структуру изображенную на рисунке 3.1.

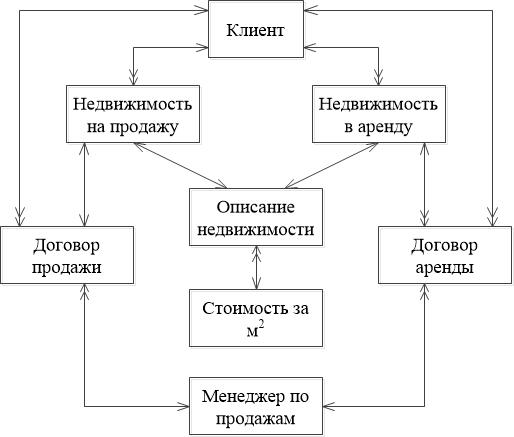


Рисунок 3.1 – Простая сетевая структура

Полученная структура является простой сетевой, так как отсутствуют связи «многие ко многим». Нормализация не может быть произведена в сетевой структуре, поэтому необходимо привести сетевую модель к иерархической. Для этого необходимо чтобы у каждого потомка было по одному родителю. У отношений «Договор продажи» и «Договор аренды» несколько родителей. Введем избыточность, таким образом получим древовидную структуру (рисунок 3.2).

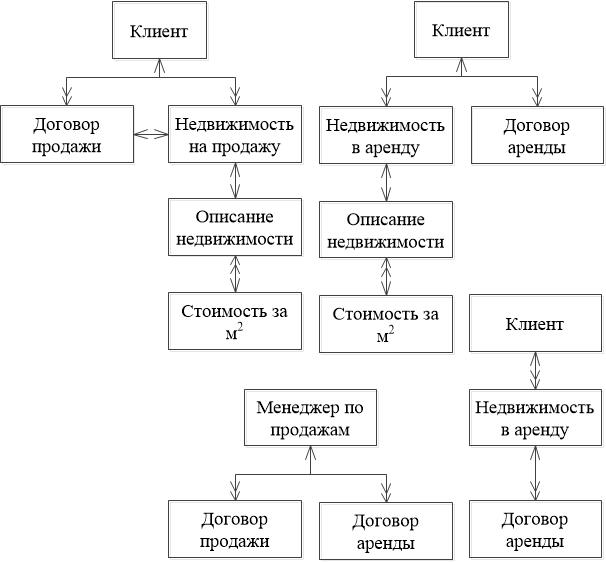


Рисунок 3.2 – Древовидная структура

Полученная структура является древовидной, так как в ней у одного потомка имеется только один родитель.

## 3.3 Нормализация исходной модели

Нормализация – это процесс приведения древовидной структуры к реляционной. Третья нормальная форма является достаточной при решении большинства практических задач, и процесс проектирования реляционной базы данных, как правило, заканчивается приведением к ней. Поэтому выполним приведение таблиц к третьей нормальной форме.

### 3.3.1 Первая нормальная форма

Отношение находится в первой нормальной форме, когда определен первичный ключ, устранены повторяющиеся группы и все поря атомарны. В каждом отношении выделим первичный ключ, устраним повторяющиеся группы и приведем все поля к атомарности.

В отношении «Клиент» добавим атрибут «Id клиента» – это будет первичный ключ. Поля «ФИО» в контексте решаемой задачи будем считать атомарными. Атрибут «Паспортные данные» разделим на два атрибута «Серия паспорта» и «Номер паспорта».

Отношение «Недвижимость на продажу» первичный ключ – «Id недвижимости». Все атрибуты данного отношения атомарны.

В отношении «Недвижимость в аренду» первичный ключ – «Id недвижимости».

Отношение «Описание недвижимости» добавим атрибут «Id описания» – первичный ключ. Атрибут «Адрес» в контексте решаемой задачи будем считать атомарными. В контексте решаемой задачи атрибуты «Бытовая техника», «Объекты инфраструктуры» и «Вид из окна» будем считать атомарными.

В отношении «Договор продажа» выделим первичный ключ «Номер договора». Отношение не содержит не атомарных атрибутов.

В отношении «Договор аренда» выделим первичный ключ «Номер договора». Отношение не содержит не атомарных атрибутов.

В отношении «Стоимость за м2» первичный ключ – «Тип планировки».

В отношение «Менеджер по продажам» добавим атрибут «Id менеджера» – первичный ключ. Атрибут «Полное имя» в контексте решаемой задачи будем считать атомарными.

У всех отношений, входящих в базу данных определены первичные ключи, поля атомарны и нет повторяющихся групп, поэтому отношения находятся в первой нормальной форме.

### 3.3.2 Вторая нормальная форма

Таблица находится во второй нормальной форме, если она находится в первой нормальной форме и все ее атрибуты, не входящие в первичный ключ, связаны полной функциональной зависимостью с первичным ключом.

Таблица «Клиент»



Рисунок 3.3 – Таблица «Клиент»

Таблица «Клиент» находится во второй нормальной форме, т.к. все атрибуты связаны полной функциональной зависимостью с первичным ключом.

Таблица «Недвижимость на продажу»



Рисунок 3.4 – Таблица «Недвижимость на продажу»

Данная таблица находится во второй нормальной форме, т.к. все атрибуты таблицы связаны полной функциональной зависимостью с первичным ключом.

Таблица «Недвижимость в аренду»



Рисунок 3.5 – Таблица «Недвижимость в аренду»

Таблица «Описание недвижимости»



Рисунок 3.6 – Таблица «Описание недвижимости»

Таблица «Описание недвижимости» находится во второй нормальной форме, т.к. первичный ключ однозначно идентифицирует каждый атрибут.

Таблица «Стоимость за м2»



Рисунок 3.7 – Таблица «Стоимость за м2»

Данная таблица находится во второй нормальной форме, т.к. все атрибуты таблицы связаны полной функциональной зависимостью с первичным ключом.

Таблица «Договор аренды»



Рисунок 3.8 – Таблица «Договор аренды»

Таблица «Договор аренды» находится во второй нормальной форме, т.к. все атрибуты таблицы связаны полной функциональной зависимостью с первичным ключом.

Таблица «Договор продажи»



Рисунок 3.9 – Таблица «Договор аренды»

Таблица «Договор аренды» находится во второй нормальной форме, т.к. все атрибуты таблицы связаны полной функциональной зависимостью с первичным ключом.

Таблица «Менеджер по продажам»



Рисунок 3.10 – Таблица «Менеджер по продажам»

Данная таблица находится во второй нормальной форме, т.к. все атрибуты таблицы связаны полной функциональной зависимостью с первичным ключом.

### 3.3.3 Третья нормальная форма

Таблица находится в третьей нормальной форме, если она находится во второй нормальной форме и ни один из ее неключевых атрибутов не связан функциональной зависимостью с любым другим неключевым атрибутом. Все таблицы находятся в третьей нормальной форме.

Таблицы нормализованы построим логическую схему базы данных (рисунок 3.11).

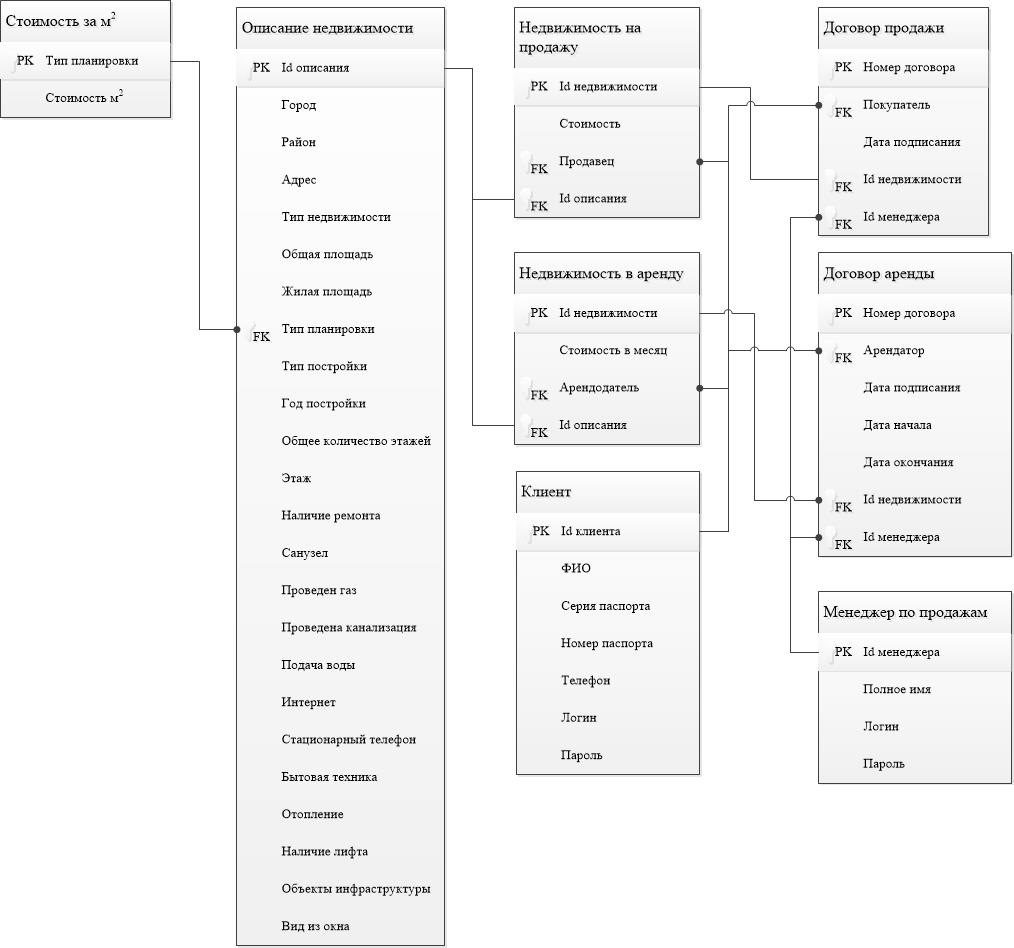


Рисунок 3.11 – Логическая схема базы данных

# 4 РЕАЛИЗАЦИЯ БАЗЫ ДАННЫХ

## 4.1 Обоснование выбора СУБД и языка программирования

В качестве СУБД была выбрана PostgreSQL. Это мощная, бесплатная, высокопроизводительная, кросс-платформенная реляционная база данных. На сегодняшний день PostgreSQL не уступает по функциональности коммерческим СУБД и зачастую превосходит их. Все чаще крупные компании используют PostgreSQL вместо MySQL для организации своих баз данных. Все это послужило причиной выбора в пользу PostgreSQL.

В качестве среды разработки будет использована интегрированная среда разработки Eclipse for Java Developers. Выбор этой среды обусловлен тем, что это бесплатная, полнофункциональная и быстрая среда разработки программ на языке Java. Средой поддерживаются функции для рефакторинга исходного кода, выполнения модульных тестов, авто-дополнение ввода, что сильно упрощает и ускоряет разработку.

Для реализации программной системы был выбран язык программирования Java 8. Язык Java является современным, объектно-ориентированным языком программирования. Программы на языке Java компилируются в байт код и выполняются на виртуальной машине, что дает им кросс-платформенность на уровне бинарных сборок. Платформа Java предоставляет огромное количество стандартных библиотек, в том числе и для создания графического пользовательского интерфейса, взаимодействия с реляционными базами данных, работой с файловой системой, и т.д. Для платформы Java создано огромное количество opensource фреймворков для быстрой разработки сложных программных продуктов. Программы написанные на этом языке легко тестировать и расширять, обладают высокой производительностью и защищенностью.

## 4.2 Разграничение прав доступа

Данный программный продукт содержит четыре уровня доступа: администратор, менеджер по продажам, клиент, клиент без авторизации. В зависимости от этого разграничения с базой данной могу быть выполнены определенные манипуляции.

Администратор может добавлять и редактировать менеджеров по продажам, а также изменять стоимость за м2.

Менеджер по продажам добавляет и редактирует клиентов, добавляет и редактирует недвижимость на продажу и аренду, заключает договора аренды и купли-продажи, выполняет поиск недвижимости по заданным критериям.

Клиент может выполнить поиск недвижимости на продажу и аренду по заданным критериям и просмотреть заключенные с ним договоры.

Клиент без авторизации может выполнить поиск недвижимости на продажу и аренду.

## 4.3 Физическая реализация базы данных

В соответствии со схемой полученой в ходе нормализации были созданы таблицы базы данных и организованы связи между ними.

Таблица «Менеджер»:

CREATE TABLE managers (

id INTEGER NOT NULL PRIMARY KEY DEFAULT nextval('managers\_id\_seq'),

-- Фамилия Имя Отчество

fullname VARCHAR(50) NOT NULL,

-- Логин

login VARCHAR(20) NOT NULL UNIQUE,

-- Пароль

password VARCHAR(20) NOT NULL,

-- Уволен

isFired BOOLEAN NOT NULL DEFAULT FALSE

);

Таблица «Клиент:

CREATE TABLE clients (

id INTEGER NOT NULL PRIMARY KEY DEFAULT nextval('clients\_id\_seq'),

-- Фамилия Имя Отчество

fullname VARCHAR(50) NOT NULL,

-- Серия пасспорта

passportSeries VARCHAR(2) NOT NULL,

-- Номер пасспорта

passportNumber INTEGER NOT NULL,

-- Адрес

phoneNumber VARCHAR(15),

-- Логин

login VARCHAR(20) NOT NULL UNIQUE,

-- Пароль

password VARCHAR(20) NOT NULL

);

Таблица «Описание недвижимости»:

CREATE TABLE realtyDescriptions (

id INTEGER NOT NULL PRIMARY KEY DEFAULT nextval('realtydescriptions\_id\_seq'),

-- Район города

region VARCHAR(30) NOT NULL,

-- Адрес

address VARCHAR(30) NOT NULL,

-- Город

city VARCHAR(20) NOT NULL,

-- Тип недвижимости

realtyType VARCHAR(15) NOT NULL,

-- Общая площадь

totalArea REAL NOT NULL,

-- Жилая площадь

livingArea REAL NOT NULL,

-- Площадь приусадебного участка

smallholdingArea real,

-- Количество жилых комнат

livingRoomsCount INTEGER NOT NULL,

-- Тип планировки layoutType

layoutType VARCHAR(15) NOT NULL REFERENCES square\_meter\_costs(layoutType) ON DELETE CASCADE,

-- Тип постройки

constructionType VARCHAR(15) NOT NULL,

-- Год постройки

yearOfConstruction INTEGER NOT NULL,

-- Этажность

totalFloorCount INTEGER NOT NULL,

-- Номер этажа

floorNumber INTEGER NOT NULL,

-- Наличие ремонта

state VARCHAR(100) NOT NULL,

-- Санузел

toiletType VARCHAR(15) NOT NULL,

-- Подача воды

water VARCHAR(15) NOT NULL,

-- Газ

gas boolean NOT NULL,

-- Проведена канализация

sewerage boolean NOT NULL,

-- Интернет

internet boolean NOT NULL,

-- Стационарный телефон

landlineTelephone boolean NOT NULL,

-- Отопление

heating boolean NOT NULL,

-- Наличие лифта

elevator boolean NOT NULL,

-- Бытовая техника

whiteGoods VARCHAR(200),

-- Объекты инфраструктуры

infrastructure VARCHAR(200),

-- Вид из окна

viewFromWindow VARCHAR(100)

);

Таблица «Стоимость квадратного метра»:

CREATE TABLE square\_meter\_costs (

layoutType VARCHAR(15) NOT NULL PRIMARY KEY,

cost REAL NOT NULL

);

Таблица “Недвижимость на продажу”:

CREATE TABLE realtiesToSale (

id INTEGER NOT NULL PRIMARY KEY DEFAULT nextval('realties\_to\_sale\_id\_seq'),

-- Продавец

sellerId INTEGER NOT NULL REFERENCES clients(id) ON DELETE CASCADE,

-- Стоимость

realtyCost REAL NOT NULL,

-- описание недвижимости

realtyDescriptionId INTEGER NOT NULL REFERENCES realtyDescriptions(id) ON DELETE CASCADE

);

Таблица «Недвижимость в аренду»:

CREATE TABLE realtiesToRent (

id INTEGER NOT NULL PRIMARY KEY DEFAULT nextval('realties\_to\_rent\_id\_seq'),

-- Арендодатель

landLordId INTEGER NOT NULL REFERENCES clients(id) ON DELETE CASCADE,

-- Стоимость

realtyCost REAL NOT NULL,

-- описание недвижимости

realtyDescriptionId INTEGER NOT NULL REFERENCES realtyDescriptions(id) ON DELETE CASCADE,

-- флаг доступности для аренды(false после переноса недвижимости в продажу)

availableForRent BOOLEAN NOT NULL DEFAULT TRUE

);

Таблица «Договор продажи»:

CREATE TABLE saleContract (

id INTEGER NOT NULL PRIMARY KEY DEFAULT nextval('salecontract\_id\_seq'),

-- Клиент покупатель

clientBuyerId INTEGER NOT NULL REFERENCES clients(id) ON DELETE CASCADE,

-- Дата подписания

signingDate DATE NOT NULL,

-- Ссылка на недвижимость

realtyId INTEGER NOT NULL REFERENCES realtiesToSale(id) ON DELETE CASCADE,

-- Менеджер заключающий договор

managerId INTEGER NOT NULL REFERENCES managers(id) ON DELETE CASCADE

);

Таблица «Договор аренды»:

CREATE TABLE rentContract (

id INTEGER NOT NULL PRIMARY KEY DEFAULT nextval('rentcontract\_id\_seq'),

-- Клиент арендатор

clientRenterId INTEGER NOT NULL REFERENCES clients(id) ON DELETE CASCADE,

-- Дата подписания

signingDate DATE NOT NULL,

-- Ссылка на недвижимость

realtyId INTEGER NOT NULL REFERENCES realtiesToRent(id) ON DELETE CASCADE,

-- Срок аренды

beginDate DATE NOT NULL,

endDate DATE NOT NULL,

-- Менеджер

managerId INTEGER NOT NULL REFERENCES managers(id) ON DELETE CASCADE

);

Для упрощения взаимодействия с базой данных были созданы следующие представления:

Представление «Договор продажи»:

CREATE VIEW saleContractView AS

SELECT

saleContract.id,

realtiesToSale.sellerId,

saleContract.clientBuyerId,

saleContract.signingDate,

saleContract.realtyId,

saleContract.managerId

FROM

saleContract JOIN realtiesToSale ON saleContract.realtyId = realtiesToSale.id;

Представление «Договор аренды»:

CREATE VIEW rentContractView AS

SELECT

rentContract.id,

realtiesToRent.landLordId,

rentContract.clientRenterId,

rentContract.realtyId,

rentContract.signingDate,

rentContract.beginDate,

rentContract.endDate,

rentContract.managerId

FROM

rentContract JOIN realtiesToRent ON rentContract.realtyId = realtiesToRent.id;

Представление «Недвижимость на продажу»:

CREATE VIEW realtiesToSaleView AS SELECT

realtiesToSale.id, realtyDescriptionId, sellerId, region, address, city, realtycost, realtytype, totalarea, livingarea, smallholdingArea, livingroomscount, layouttype, constructiontype, yearofconstruction, totalfloorcount, floornumber, state, toilettype, water, gas, sewerage, internet, landlinetelephone, heating, elevator, whitegoods, infrastructure, viewfromwindow

FROM realtiesToSale JOIN realtyDescriptions ON realtiesToSale.realtyDescriptionId = realtyDescriptions.id;

Представление «Недвижимость в аренду»:

CREATE VIEW realtiesToRentView AS SELECT

realtiesToRent.id, realtiesToRent.availableForRent, realtyDescriptionId, landLordId, region, address, city, realtycost, realtytype, totalarea, livingarea, smallholdingArea, livingroomscount, layouttype, constructiontype, yearofconstruction, totalfloorcount, floornumber, state, toilettype, water, gas, sewerage, internet, landlinetelephone, heating, elevator, whitegoods, infrastructure, viewfromwindow

FROM realtiesToRent JOIN realtyDescriptions ON realtiesToRent.realtyDescriptionId = realtyDescriptions.id;

## 4.4 Описание интерфейса пользователя

После запуска программы пользователю предлагается выбрать роль пользователя и ввести данные учетной записи в поля «логин» и «пароль».

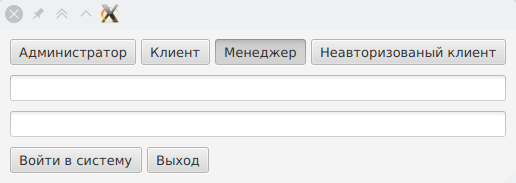


Рисунок 4.1– Авторизация пользователя

При авторизации от имени администратора пользователю предоставляется соответствующий интерфейс. Внешний вид интерфейса представлен на рисунке 4.2.

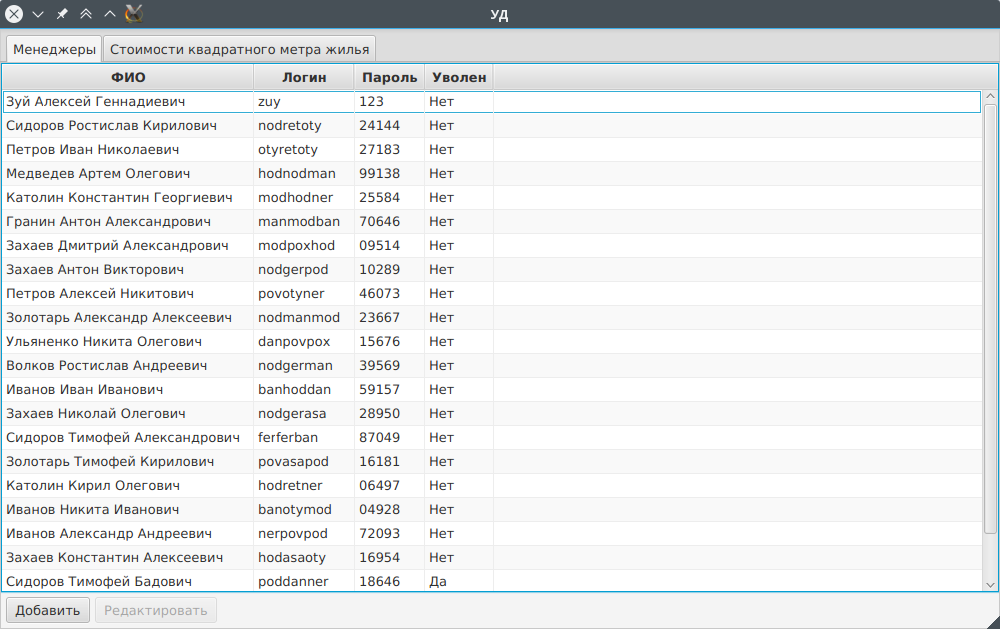


Рисунок 4.2 – Вкладка «Менеджеры»

Интерфейс содержит вкладки «Менеджеры» и «Стоимости квадратного метра жилья». На вкладке менеджера администратор может добавлять и изменять учетные записи менеджеров по продажам (рисунок 4.3).

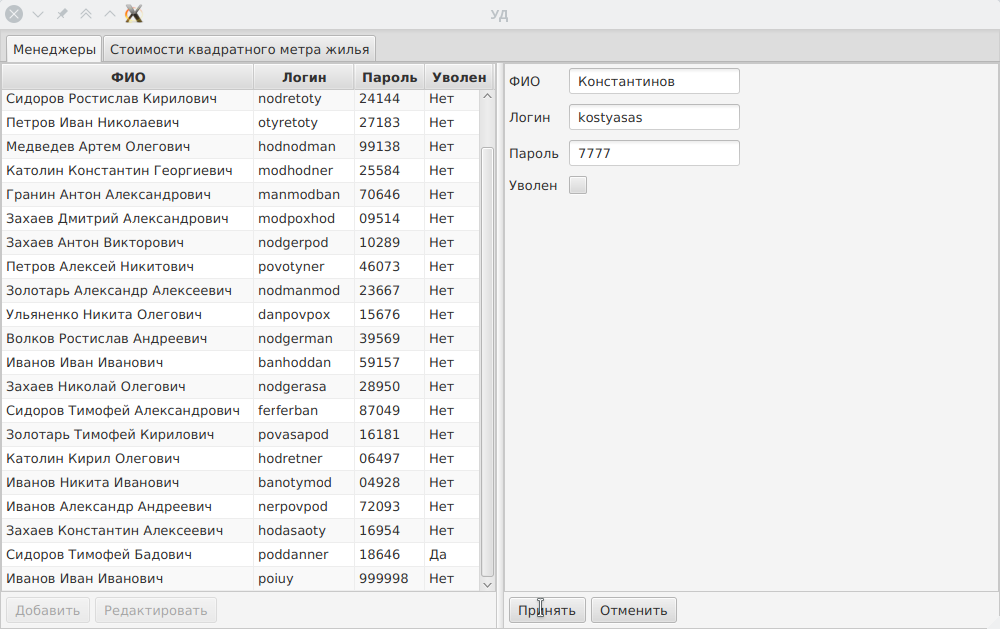


Рисунок4.3 – Добавление менеджера

При нажатии кнопки «Принять» менеджер добавляется в базу. Если при изменении данных о менеджере поставить галочку напротив «Уволен», то этот менеджер не сможет зайти в систему.

На вкладке «Стоимости квадратного метра жилья» администратор может изменить значение стоимости квадратного метра (рисунок 4.4). При нажатии кнопки «Принять» значения сохраняются в базе данных.

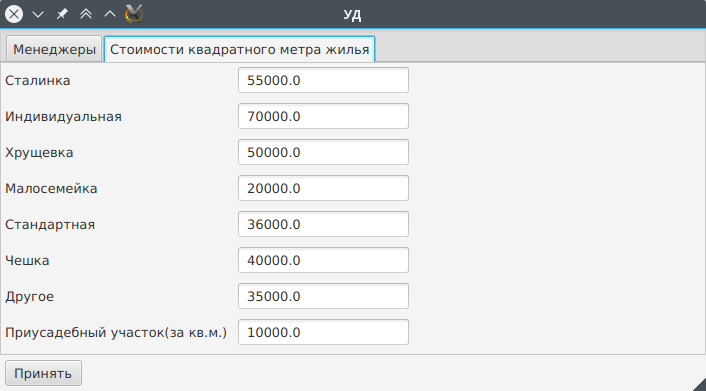


Рисунок 4.4 – Вкладка «Стоимости квадратного метра жилья»

При авторизации от имени менеджера по продажам пользователю предоставляется соответствующий интерфейс. На вкладке «Клиент» менеджер может добавлять и изменять данные учетных записи клиентов (рисунок 4.5).

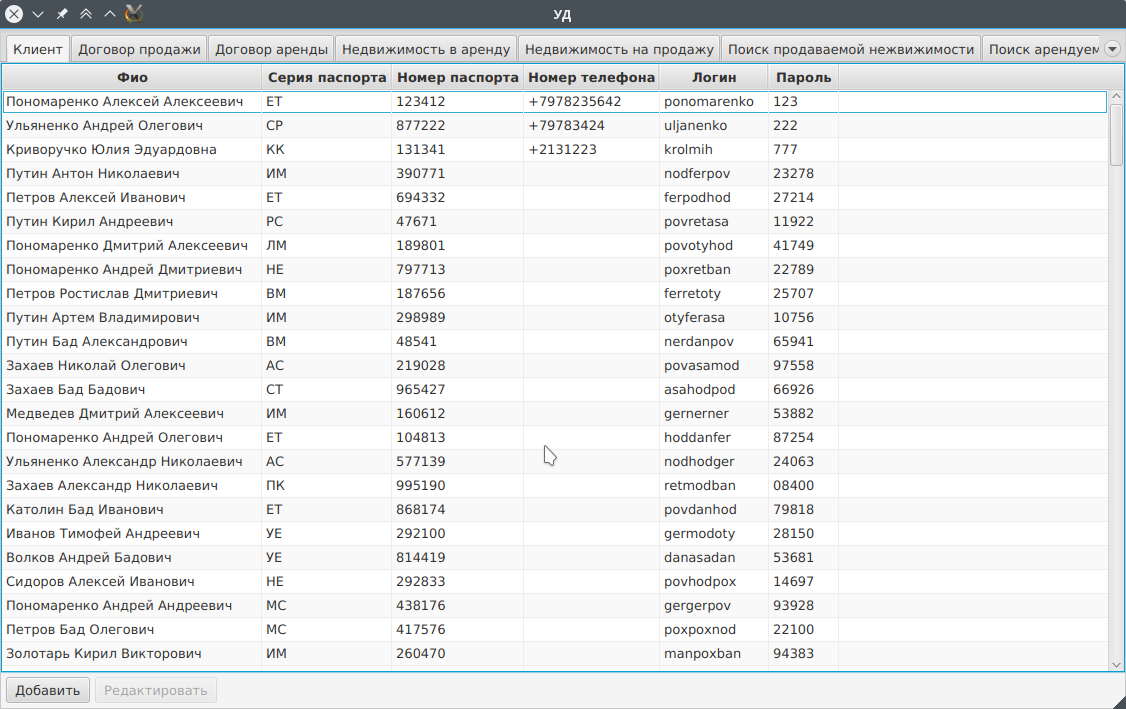


Рисунок 4.5– Вкладка «Клиент»

При добавлении и изменении выполняется автоматическая проверка корректности ввода данных (рисунок 4.6).

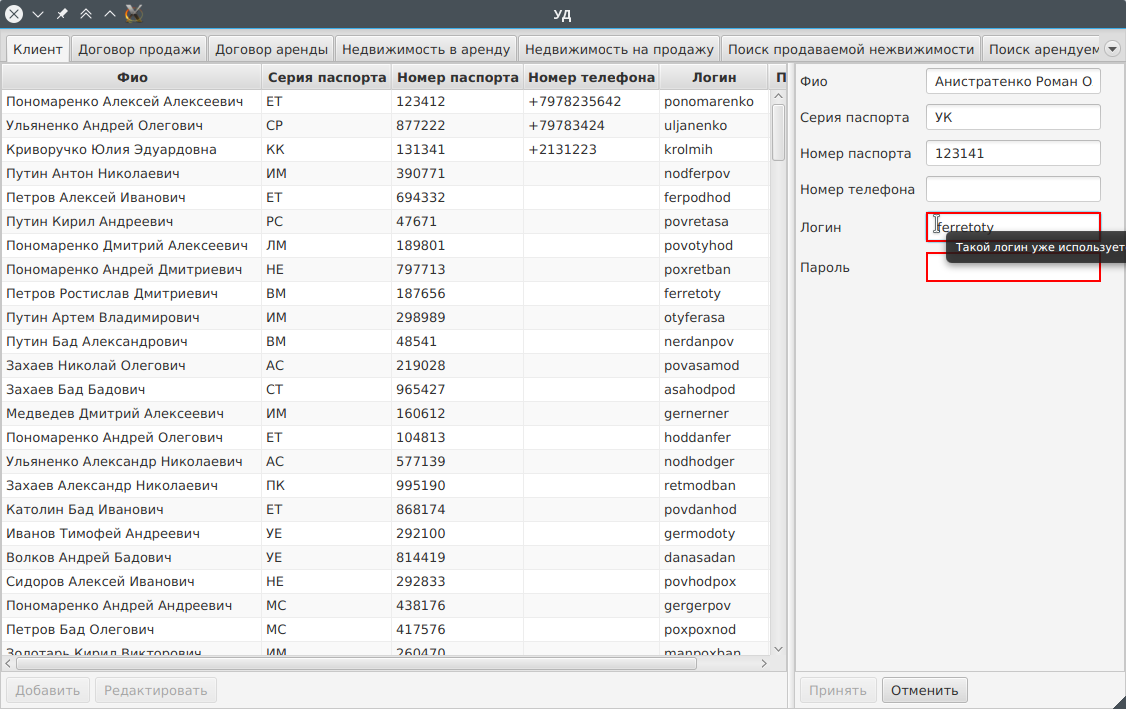


Рисунок 4.6 – Проверка вводимых данных

На вкладке «Договор продажи» менеджер по продажам может добавлять и изменять договоры о продажах. Все вводимые данные проверяются на корректность.

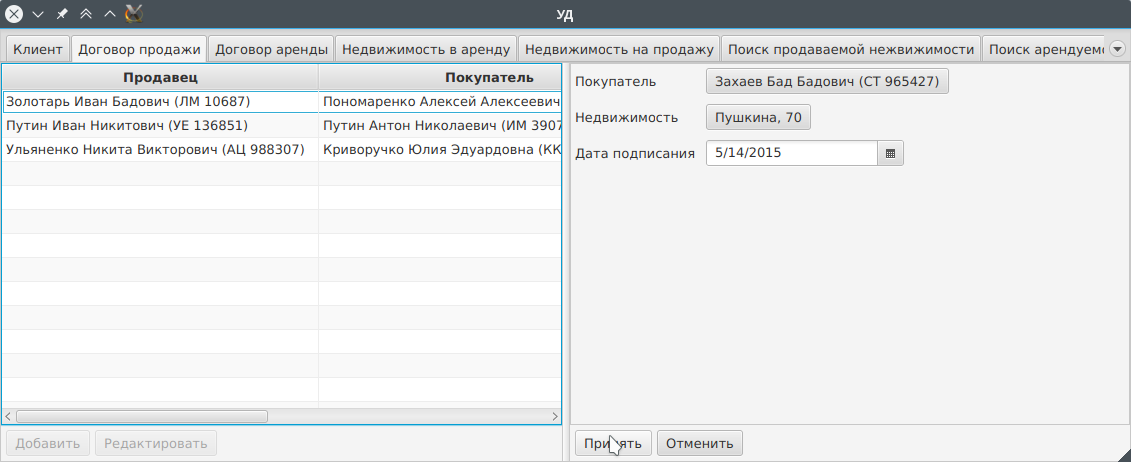


Рисунок 4.7 – Вкладка «Договор продажи»

Вкладка «Договор аренды» содержит интерфейс для добавления и редактирования договоров об аренде. Внешний вид интерфейса схож с интерфейсом «Договор продажи» который приведен на рисунке 4.7.

На вкладке «Недвижимость на аренду» менеджер может добавлять, редактировать недвижимость и дублировать недвижимость в «Недвижимость на продажу». При добавлении и редактировании недвижимости выполняется проверка вводимых данных. При нажатии кнопки «Рассчитать стоимость» автоматически рассчитывается стоимость недвижимости и записывается в поле «Стоимость». При нажатии на кнопку «Принять» выполняется добавление записи в базу данных.

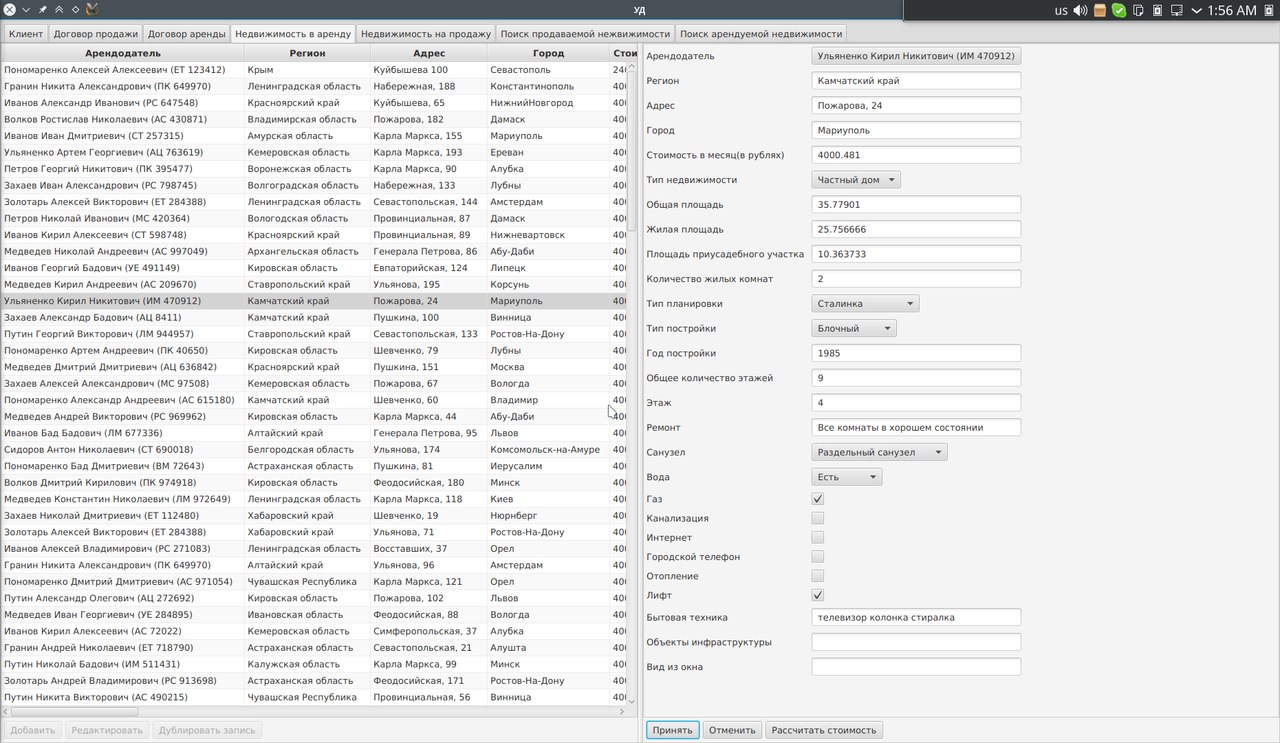


Рисунок 4.8 – Добавление недвижимости на аренду

Вкладка «Недвижимость на продажу» содержит интерфейс для добавления и редактирования недвижимости на продажу. Внешний вид интерфейса схож с интерфейсом «Недвижимости на аренду». Предусмотрено дублирование недвижимости из продажи в аренду.

Вкладка «Поиск арендуемой недвижимости» реализует интерфейс поиска недвижимости по заданным критериям (рисунок 4.9).

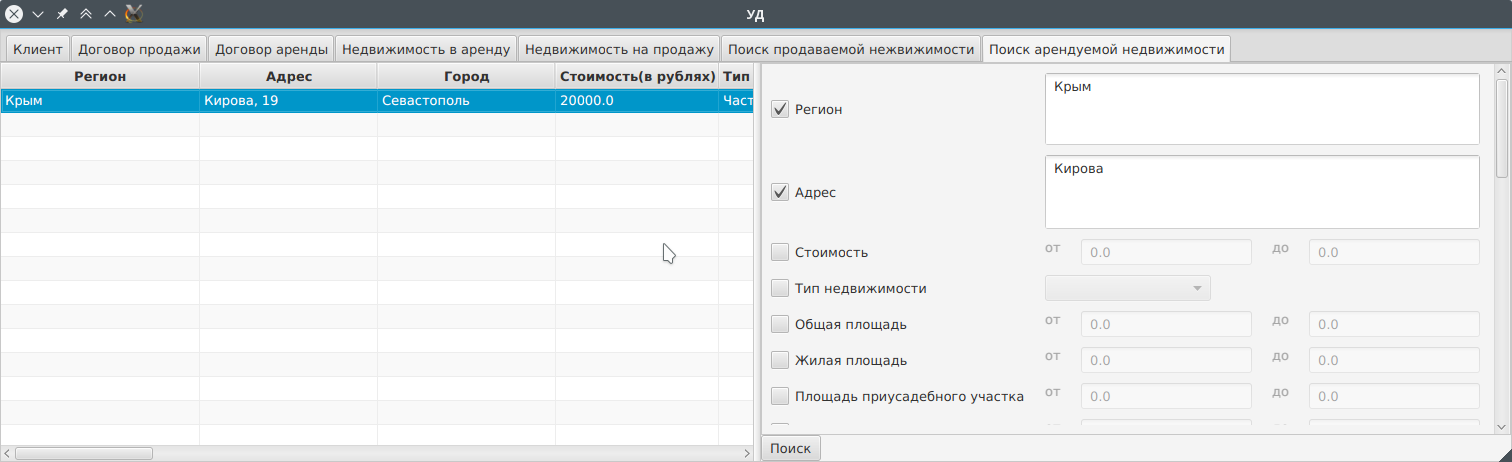


Рисунок 4.9 – Вкладка «Поиск арендуемой недвижимости»

Вкладка «Поиск продаваемой недвижимости» содержит интерфейс для добавления и редактирования договоров об аренде. Внешний вид интерфейса схож с интерфейсом «Поиск арендуемой недвижимости» который приведен на рисунке 4.9.

При авторизации от имени клиента пользователю предоставляется соответствующий интерфейс. На вкладке «Поиск продаваемой недвижимости» клиент может выполнить поиск недвижимости, которая продается (рисунок 4.10).

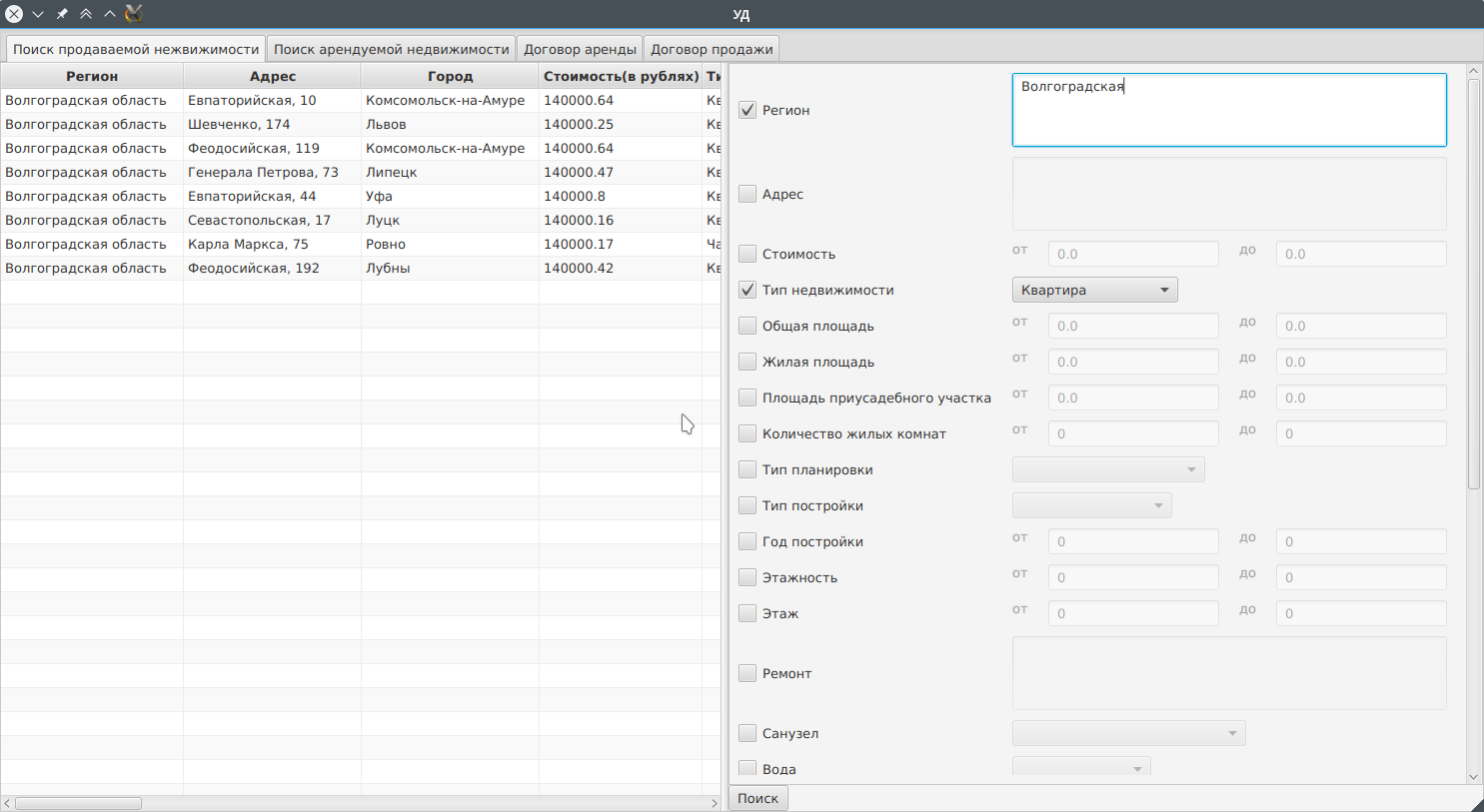


Рисунок 4.10 – Вкладка «Поиск продаваемой недвижимости»

Вкладка «Поиск арендуемой недвижимости» содержит интерфейс для поиска недвижимости, которая арендуется. Внешний вид интерфейса схож с интерфейсом «Поиск продаваемой недвижимости» который приведен на рисунке 4.10.

Вкладка «Договор аренды» содержит информацию о том какие договоры аренды заключал пользователь (рисунок 4.11).

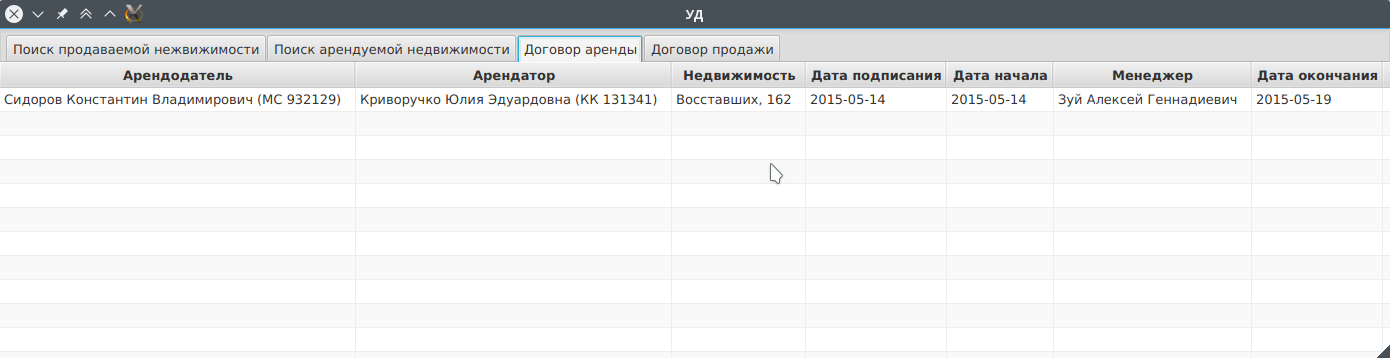


Рисунок 4.11 - Вкладка «Договор аренды»

На вкладке «Договор продажи» отображается информация о договорах продажи которые заключил клиент. Внешний вид интерфейса схож с интерфейсом «Договор аренды» который приведен на рисунке 4.11.

При входе в систему как клиент без авторизации пользователь может осуществлять поиск недвижимости на вкладках «Поиск продаваемой недвижимости» и «Поиск арендуемой недвижимости». Внешний вид интерфейса схож с интерфейсом поиска у клиента который приведен на рисунке 4.10.

## 4.5 Тестирование базы данных

В ходе тестирования базы данных ошибок не было выявлено, что говорит о корректности созданной схемы дынных. В качестве примера приведены результаты добавления данных через представление и “Недвижимость на продажу”.

Запрос на добавление данных:

INSERT INTO realtiestosaleview

(id, sellerId, region, address, city, realtycost, realtytype, totalarea, livingarea, livingroomscount, layouttype, constructiontype, yearofconstruction, totalfloorcount, floornumber, state, toilettype, water, gas, sewerage, internet, landlinetelephone, heating, elevator, whitegoods, infrastructure, viewfromwindow, smallholdingArea)

VALUES

(nextval('realties\_to\_sale\_id\_seq'), 1, 'Крым', 'Севастопольская 15', 'Симферополь', 12000, 'PRIVATE\_REALTY', 120.0, 80.0, 4, 'INDIVIDUALNAYA', 'BRICK', 1990, 3, 1, 'все комнаты в хорошем состоянии', 'SOVMESHENIY', 'YES', true, true, true, false, true, false, 'Стиральная машинка, Холодильник', 'Садик', 'никакой', 25);

Запрос на чтение:

select \* from realtiesToSaleView;

Результат выполнения запроса приведен на рисунке 4.12. Как видно из рисунка данные были успешно добавлены в базу данных.

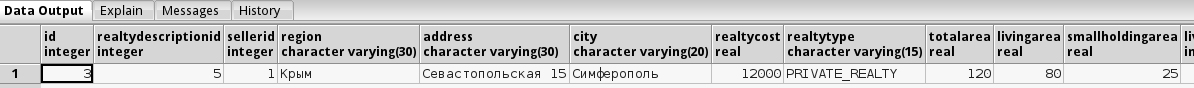


Рисунок 4.12– Добавленная запись

Наиболее характерные запросы для разработаной базы данных:

Получить всю продаваемую недвижимость которая не была продана:

“SELECT \* FROM realtiesToSaleView WHERE id NOT IN (SELECT realtyId FROM saleContract)”

Получить всю доступную для аренды недвижимость:

"SELECT \* FROM realtiesToRentView WHERE NOT EXISTS (SELECT \* FROM rentContract WHERE rentContract.realtyId = realtiesToRentView.id AND rentContract.endDate > CURRENT\_DATE) AND availableForRent = TRUE"

Получить все договоры купли-продажи заключенные клиентом с идентификатором clientId:

“SELECT \* FROM saleContractView where clientBuyerId = clientId OR sellerId = clientId”

Получить все договоры аренды заключенные клиентом с идентификатором clientId:

"SELECT \* FROM rentContractView WHERE landLordId = clientId OR clientRenterId = clientId"

Получить стоимость квадратного метра недвижимости в соответствии с типом планировки type:

"SELECT cost FROM square\_meter\_costs WHERE layoutType = type"

# 5. ИССЛЕДОВАНИЕ ИНФОРМАЦИОННЫХ ПАРАМЕТРОВ БД

Длина логической записи j-ого файла определяется как сумма длин полей, её составляющих и указателей, если они организуются разработчиком. Длина логической записи:

 [байт]

где Mj - число групп полей в записях, lij длина группы [байт], Lj и средняя длины записи m{ Lj }.

 [байт] ;

 [байт] ;

 [байт] ;

 [байт] ;

 [байт] ;

 [байт] ;

 [байт] ;

 [байт] ;

Объем памяти, необходимый для размещения информационного фонда без учёта системных данных и указателей:

 [байт]

где N - количество отношений реляционной базы данных, K j - количество записей j-го файла.

Ниже приведен расчет при следующих количествах записей: менеджер – 20, клиент – 100, Описание недвижимости – 120, Недвижимость на продажу – 60, Недвижимость в аренду – 60, Договор продажи – 20, Договор аренды – 40, стоимость квадратного метра – 7.

[байт]

Приращение информационного фонда:

 [байт-1]

где  - число добавленных типов записей,  - интенсивность добавления записей в файл j -го типа.

[байт-1];

Время заполнения информационного фонда:

 [время] , где

Первоначальный объём памяти, выделенной под развёртывание базы данных: Vобщ. = 100 МБ = 104857600 байт;

Объем программного обеспечения: Vпо = 196096 байт.

.

Время резервного копирования:





# ЗАКЛЮЧЕНИЕ

В данном курсовом проекте была спроектирована база данных предметной области «Агентство недвижимости». На языке Java было написано десктопное приложение, реализующее обращение к базе данных. Разработанная система содержит распределение прав доступа и удобный интерфейс для манипуляции с базой данных.

Для выбранной предметной области была проведена нормализация. Полученная в результате нормализации логическая схема содержит следующие таблицы: клиент, недвижимость на продажу, недвижимость в аренду, стоимость за м2, договор аренды, договор продажи, менеджер по продажам.

В результате выполнения курсового проектирования были закреплены навыки организации баз данных и разработан программный продукт, реализующий частичную автоматизацию деятельности агентства недвижимости.

# БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Дейт К. Дж. Введение в системы баз данных / К. Дж. Дейт; пер. с англ. — 7-е изд. — М.: Издательский дом «Вильямс», 2002. — 1072с.

2. Уорсли Дж. PostgreSQL. Для профессионалов / Дж. Уорсли; пер с англ. — П.: Издательский дом “Питер”, 2003. — 1206с.

3. Шилдт Г. Java. Полное руководство / Г. Шилдт; пер. с англ. — 8-е изд. — М.: Издательский дом “Вильямс”, 2012. — 1104с.

# ПРИЛОЖЕНИЕ А

Листинг программы