Министерство транспорта Российской Федерации  
Федеральное агентство железнодорожного транспорта

Федеральное государственное бюджетное образовательное учреждение  
 высшего образования  
«Дальневосточный государственный университет путей сообщения»

Кафедра «Информационные технологии и системы»

Курсовая работа

Описание предметной области. Описание бизнес-процессов.

дисциплина «Безопасность систем баз данных»

КР 09.03.04. 07.00.БО221ПИН

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Д.К. Журин

*(подпись, дата)*

Руководитель

доцент, к.т.н. Р.А. Ешенко

*(подпись, дата)*

Хабаровск 2023

Вариант 21

Разработать прикладное программное обеспечение для ведения реестра имущества университетского городка. В состав имущества входит несколько зданий. В зданиях располагаются аудитории, кафедры, лаборатории, вычислительные центры, деканаты и т. д. Любое помещение университета относится к какому-либо подразделению. Все движимое имущество, находящееся в помещении, состоит на балансе материально ответственного лица. Каждая аудитория закреплена за определенной кафедрой университета, так же в ведении кафедр находятся и лаборатории. По истечении определенного времени имущество, находящееся в помещениях, списывается. Архив списанного имущества не ведется.

Набор данных к варианту

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **Поле** | **Тип** | **Размер** | **Описание** |
| 1 | Kadastr | **Числовой** | 2 | Регистрационный номер здания |
| 2 | BuildingName | **Текстовый** | 20 | Название здания (корпуса) университета |
| 3 | Land | **Числовой** | Авто | Площадь земельного участка |
| 4 | Address | **Текстовый** | 60 | Адрес здания |
| 5 | Year | **Числовой** | 4 | Год постройки |
| 6 | Material | **Текстовый** | 15 | Материал стен здания |
| 7 | Wear | **Числовой** | 2 | Износ в процентах |
| 8 | Flow | **Числовой** | 2 | Число этажей в здании |
| 9 | Picture | **Поле объекта OLE** | Авто | Фотография здания |
| 10 | Comment | **Поле Memo** | Авто | Дополнительные сведения по зданию |
| 11 | HallID | **Текстовый** | 5 | Номер аудитории |
| 12 | Square | **Числовой** | Авто | Площадь аудитории |
| 13 | Windows | **Числовой** | 1 | Количество окон |
| 14 | Heating | **Числовой** | 3 | Число элементов в батареях отопления |
| 15 | Target | **Текстовый** | 15 | Назначение (лекционная, кафедра и т. д.) |
| 16 | Department | **Текстовый** | 15 | Принадлежность к кафедре (подразделение) |
| 17 | Chief | **Текстовый** | 30 | Материально ответственный за аудиторию |
| 18 | DepartmentID | **Числовой** | 3 | Идентификатор кафедры |
| 19 | DepartmentName | **Текстовый** | 15 | Название кафедры |
| 20 | Boss | **Текстовый** | 40 | Заведующий кафедрой |
| 21 | Phone | **Текстовый** | 10 | Телефон кафедры |
| 22 | OfficeDean | **Текстовый** | 30 | Принадлежность кафедры  к деканату |
| 23 | ChiefID | **Числовой** | 3 | Идентификатор материально ответственного лица |
| 24 | AddressChief | **Текстовый** | 60 | Домашний адрес материально ответственного лица |
| 25 | Experience | **Числовой** | 4 | Год его начала работы в университете |
| 26 | UnitID | **Числовой** | 3 | Идентификатор единицы имущества |
| 27 | UnitName | **Текстовый** | 30 | Название единицы  имущества |
| 28 | DateStart | **Дата/время** | Авто | Дата постановки на учет |
| 29 | Cost | **Денежный** | 15 | Стоимость единицы  имущества |
| 30 | CostYear | **Числовой** | 4 | Год переоценки |
| 31 | CostAfter | **Денежный** | 15 | Стоимость после переоценки |
| 32 | Period | **Числовой** | 4 | Срок службы единицы имущества |

Содержание

[Введение 5](#_Toc131256434)

[1. Описание предметной области и проектирование бизнес-процессов 6](#_Toc131256435)

[2. Реализация базы данных 7](#_Toc131256436)

[2.1. Построение ER-модели 7](#_Toc131256437)

[3. Реализация базы данных 11](#_Toc131256438)

[4. Описание процесса разработки приложения 12](#_Toc131256439)

Введение

Отдел имущественного обеспечение занимается распределением имущества по университету. Отдел устанавливает какие зданию принадлежат аудитории, закрепляют их за подразделением. Подразделение назначает материально ответственного за аудиторию и распределяет имущество между аудиториями.

Материально ответственные в состоянии просматривать всё своё имущество или имущество конкретной аудитории, искать имущество, нуждающееся в переоценке или списании, а также полную стоимость имущества.

# Описание предметной области и проектирование бизнес-процессов

Программное обеспечение для ведения имущества студенческого городка. В состав городка входят здания, которые характеризуются площадью, названием, адресом, материалом, кадастровым номером, комментарием, числом этажей, годом постройки и фотографией.

В зданиях располагаются аудитории, описываемые площадью, материально ответственным, числом окон и батарей, номером, типом, названием здания (кадастром), кафедрой.

Любое помещение соотносится со своим структурным подразделением, основными параметрами которого являются ФИО и номер телефона заведующего, принадлежностью к директорату.

Непосредственно имущество характеризуется названием, датой поставки, стоимостью за одну единицу, датой следующей переоценки, стоимостью после последней переоценки, сроком службы, числом, износом и номером аудитории, в котором имущество установлено.

Материально ответственный, закреплённый за аудитории, описывается годом начала работы, ФИО и адресом проживания.

Во время работы с базой данных пользователи заполняют карточки зданий студенческого городка. Затем, к каждому зданию прикрепляются соответствующие аудитории. Другой сотрудник университета создаёт карточки всех кафедр и заполняет их, а затем распределяет аудитории между кафедрами университета.

Кафедры описывают всё имущество в вверенных им аудиториях и вносят о них запись в базу данных.

Материально ответственные в состоянии следить за каждым имуществом в конкретной аудитории или за имуществом, приписанным к ним.

# Реализация базы данных

## Построение ER-модели

Все поля варианта были выделены в следующие сущности: здания, аудитории, кафедры, имущество, материально ответственные. Для полей «адрес», «материал», «город», «деканат» были созданы отдельные сущности справочники.

Сущность «**Здание**» в себе имеет следующие атрибуты: Кадастр, название здания, площадь здания, город, улица, номер дома, год строительства, материал, этажи, комментарий, фотография. Для атрибутов «улица», «материал», «город» использован соответствующий справочник.

Сущность «**Аудитории**» в своём составе собрала атрибуты: Номер аудитории, название здания, площадь, количество окон, количество элементов в батареях отопления, кафедра, назначение, материально ответственный. Данная Сущность имеет связь с сущностью «Здание» М:1.

Сущность «**Кафедра**» будет иметь связь М:1 с сущностью «Аудитории» и следующие атрибуты: Идентификатор кафедры, название кафедры, заведующий кафедры, телефон кафедры, деканат. Для атрибута «Деканат» использован справочник.

Сущность «**Имущество**» имеет М:1 связь с сущностью «Аудитории» и включать атрибуты: Код имущества, название имущества, дата постановки на учёт, стоимость одного экземпляра, год переоценки, стоимость после переоценки, количество, срок службы, износ.

Сущность «**Материально ответственные**» имеет связь 1:М с сущностью «Аудитории», и связь М:1 со справочниками улиц и городов. Атрибуты: Код материально ответственного, ФИО, город, улица, номер дома, номер квартиры, год начала работы.

Сущность «**Справочник материалов**» имеет связь 1:М с сущностью «Здания» и следующие атрибуты: Код материала, материал.

Сущность «**Справочник деканатов**» имеет связь 1:М с сущностью «Кафедры» и имеет следующие атрибуты: Код деканата, название.

Сущность «**Справочник улиц**» имеет связь 1:М с сущностями «Материально ответственный» и «Здания» и имеет следующие атрибуты: Код улицы, признак адреса, порядок адреса и название.

Сущность «**Справочник городов**» имеет связь 1:М с сущностями «Материально ответственный» и «Здания» и имеет следующие атрибуты: Код города, признак адреса, порядок адреса и название.

Результат построенной ER-модели представлен на рисунке Рисунок 1.

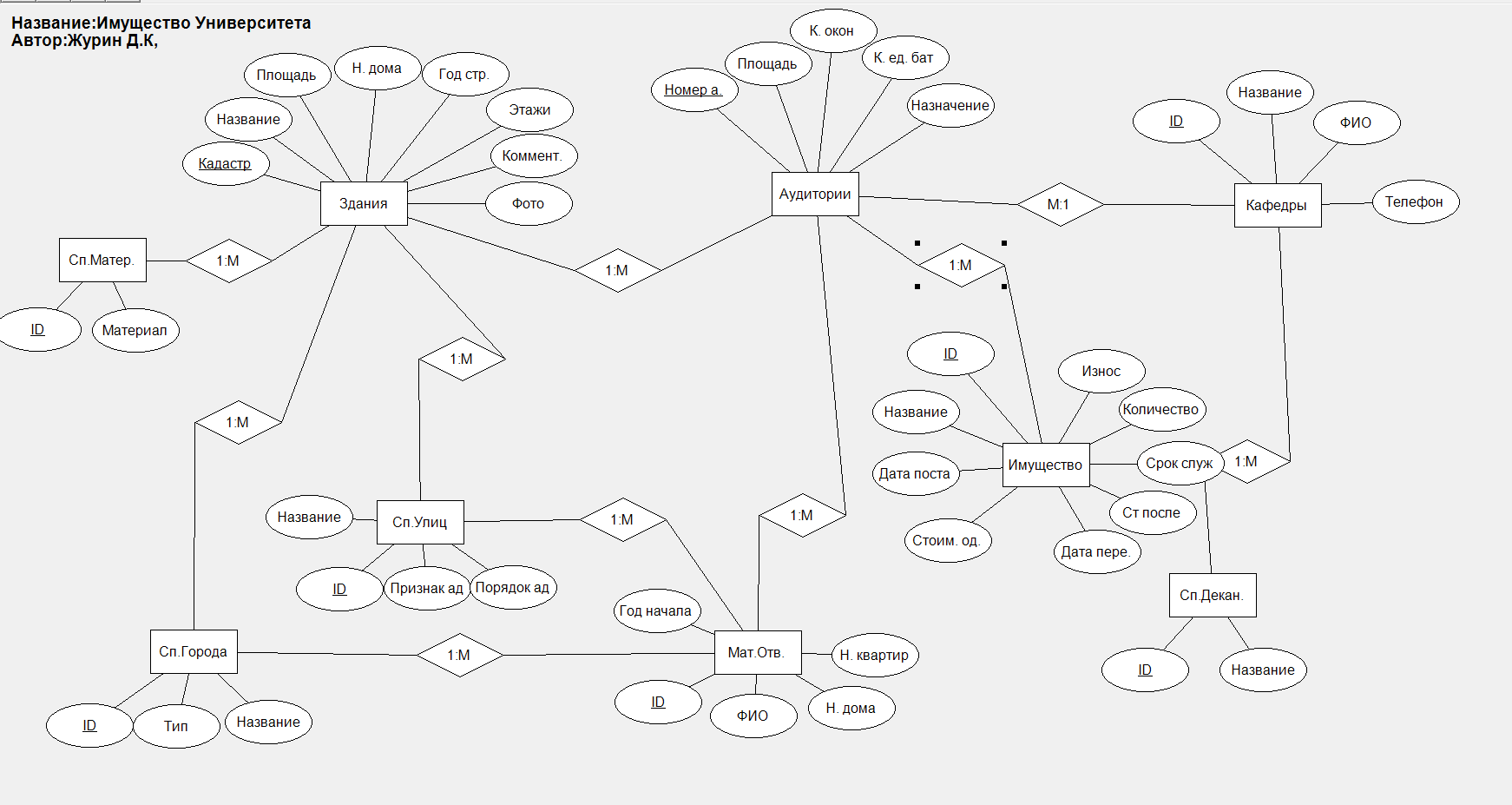


Рисунок 1 — ER-модель

После составления ER-модели было выполнено приведение до третьей нормальной формы, что подразумевает:

* каждый атрибут сущности атомарен;
* любой атрибут сущности, не входящий в состав первичного ключа, функционально полно зависит от первичного ключа;
* любой неключевой атрибут сущности функционально зависит только от первичного ключа.

Результат приведён на рисунке Рисунок 2.

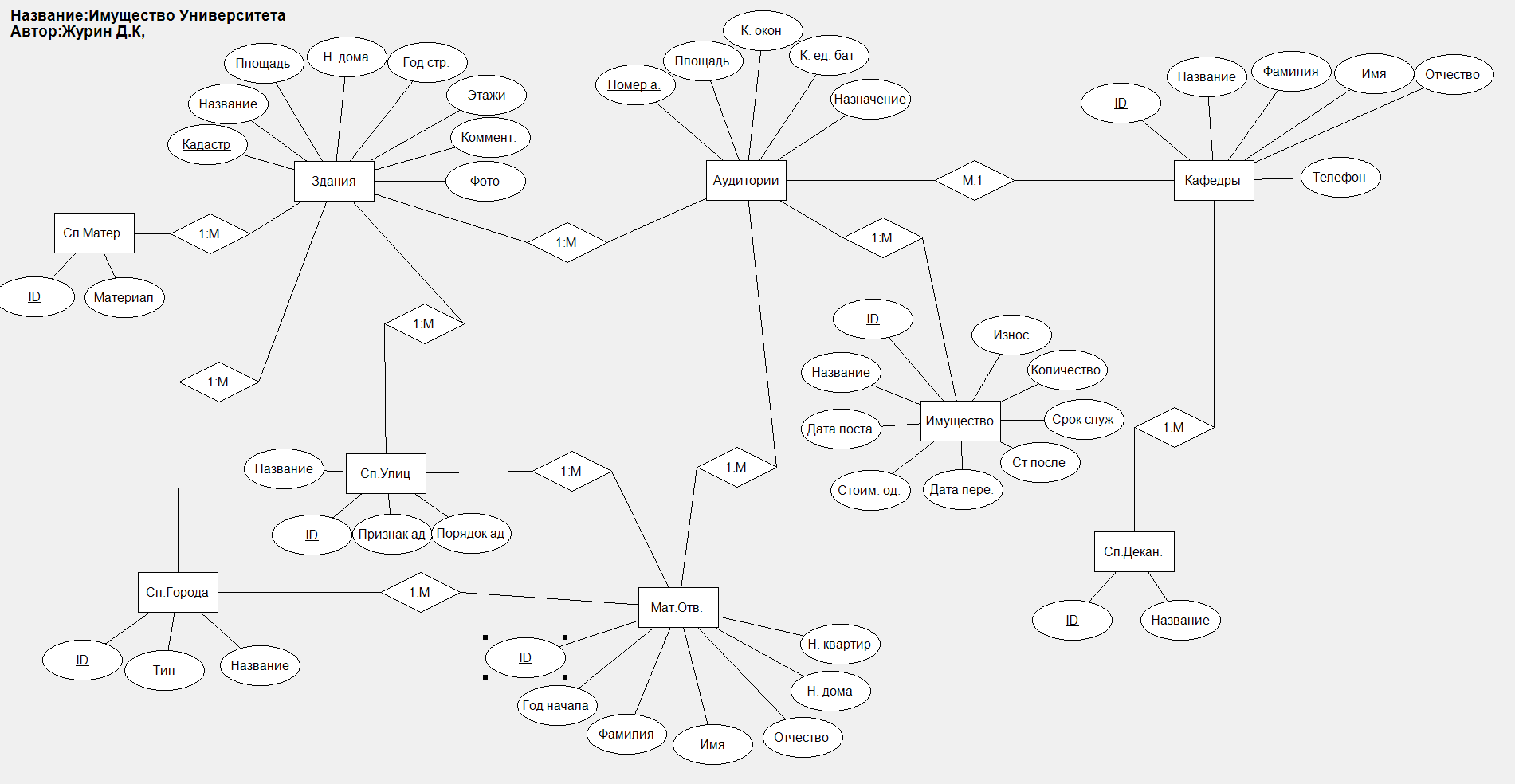


Рисунок 2— ER-модель, приведённая к третьей нормальной форме

## Преобразование ER-модели в реляционную

Сущность преобразуется в отдельную таблицу, полями таблицы становятся все атрибуты сущности. После преобразования полученной модели получились следующие таблицы:

Таблица «**Справочник городов»**:

Cities Handbook (ID, Type, Name)

Таблица «**Справочник улиц**»**:**

Streets Handbook (ID, Address attribute, Address order, Name)

Таблица «**Справочник материала**»**:**

Material Handbook (ID, Material)

Таблица «**Здания**»:

Содержит ссылку на внешние ключи – улица, город и тип материала.

Buildings (Cadastre, Name, Square, House number, Year built, Num of floors, Comment, Photo, Material, City, Address)

Таблица «**Аудитории**»:

Содержит внешние ключи – Название здания, кафедра.

Audience (Aud num, Square, Windows num, Battery num, Type, Name of building, Materially responsible, Department)

Таблица «**Имущество**»:

Содержит внешние ключи – Аудитория, материально ответственный

Property (ID, Name, Delivery date, Cost Per One, Reprice Date, Cost after reprice, Lifetime, Amount, Depreciation, Audience)

Таблица «**Материально ответственные**»:

Содержит внешние ключи – Город, улица

Materially Responsible (ID,Start year, Second name, First name, Fathers name, Num of house, Num of flat, Address, City)

Таблица «**Справочник Деканатов**»:

Deans (ID, Name)

Таблица **Department** содержит внешний ключ – деканат

Department (ID, Name, Second name, First name, Fathers name, Deans, Phone)

Для хранения логинов и паролей была создана отдельная таблица **Login.**

Login (Login, Password)

## Построение ERD-модели

С помощью PgAdmin была составлена ERD-модель базы данных

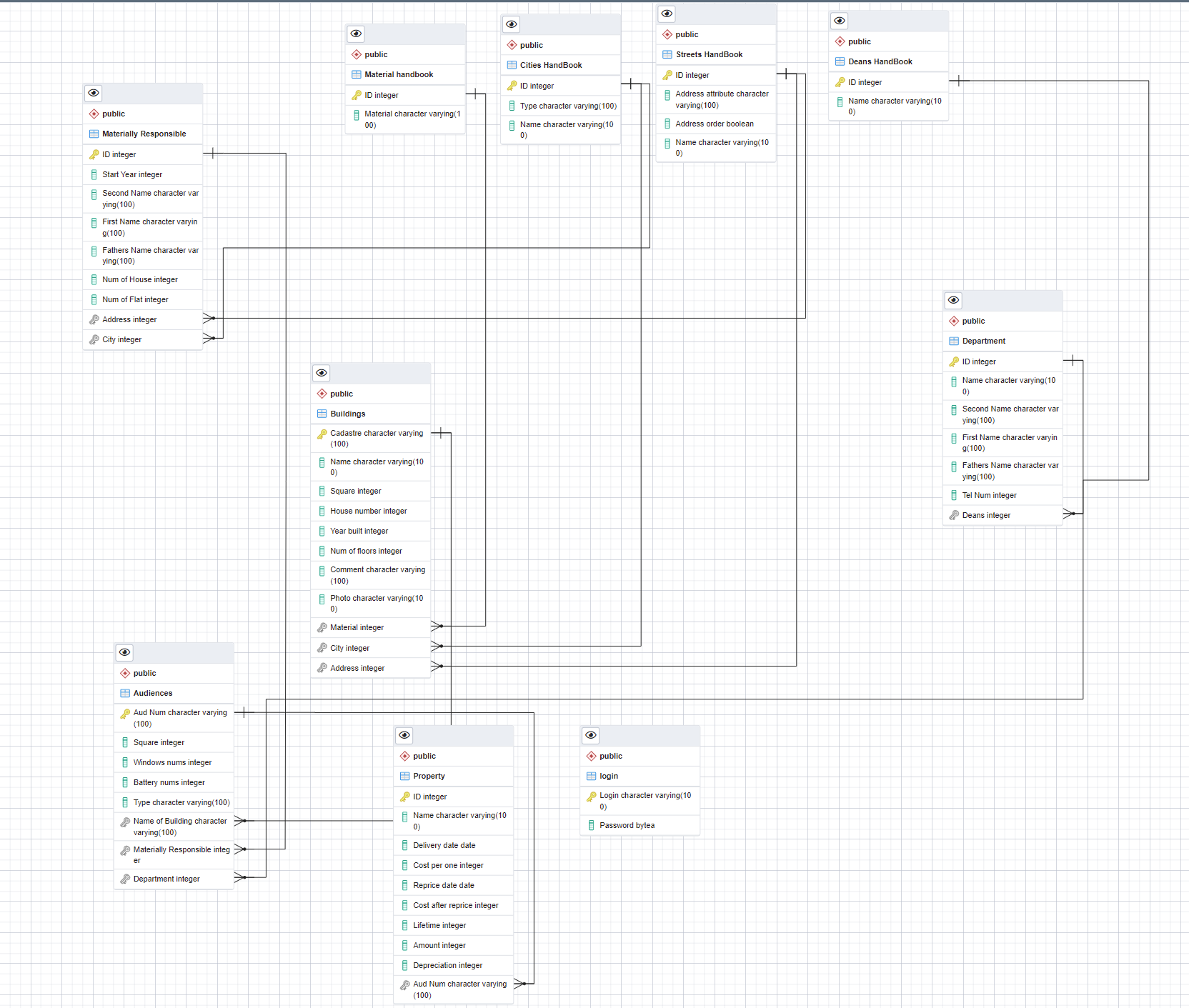


Рисунок 3 — ERD-модель базы данных

# Реализация базы данных

На основе ERD-модели был сгенерирован автоматический SQL код.

Пример кода приведён в листинге Листинг 1. В нём создаётся таблица «Здания», а также устанавливается связи 1 ко многим со справочником улиц.

Листинг 1 — Пример создания таблицы и связи

CREATE TABLE IF NOT EXISTS public.Buildings

(

Cadastre character varying(100) NOT NULL,

Name character varying(100),

Square integer NOT NULL,

House\_number integer NOT NULL,

Year\_built integer NOT NULL,

Num\_of\_floors integer NOT NULL,

Comment character varying(100),

Photo character varying(100),

Material integer NOT NULL,

City integer NOT NULL,

Address integer NOT NULL,

PRIMARY KEY (Cadastre)

);

ALTER TABLE public.Buildings

ADD FOREIGN KEY (Address)

REFERENCES public.Streets\_HandBook (ID)

ON UPDATE CASCADE;

Аналогичным образом были созданы все таблицы. Результат приведён на рисунке Рисунок 4.

Затем все таблицы были заполнены с помощью SQL команды INSERT, синтаксис которой выглядит так: INSERT INTO *название таблицы (столбец 1, столбец 2…) VALUES (значение столбца 1, значение столбца 2…).* Пример приведён на рисунке Рисунок 5.

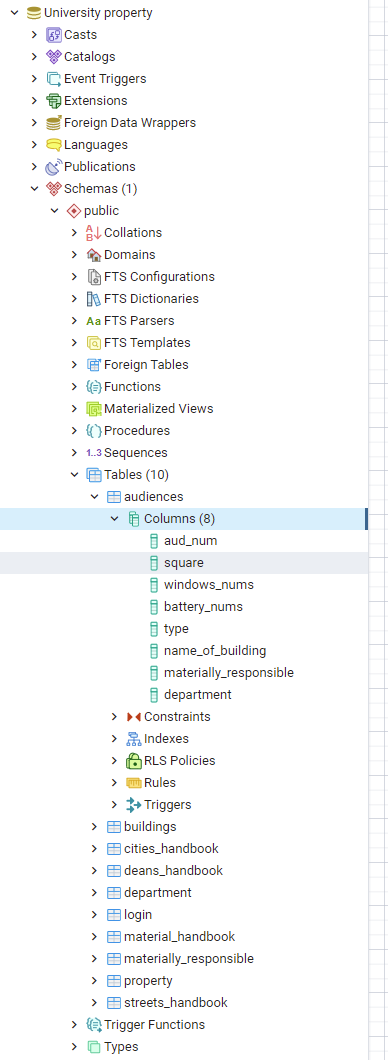


Рисунок 4 — Таблицы в PgAdmin

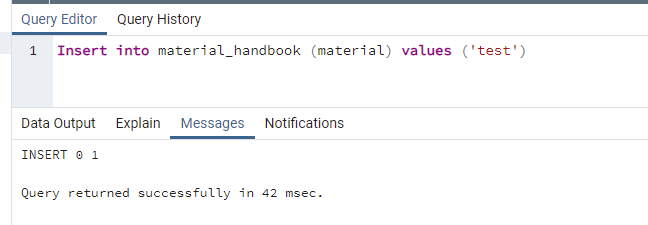


Рисунок 5 — Вставка записи в таблицу

# Описание процесса разработки приложения (руководство разработчика)

## Подключение к Базе данных

Подключение к базе данных PostgreSQL в C# возможно с помощью библиотеки Npgsql.

Для этого была создана переменная типа NpgsqlConnection, в конструктор которой отправляется строка подключения, как показано в листинге Листинг 2.

Листинг 2 — Создание NpgsqlConnection

static readonly string connString = "Server=localhost;Port=5432;User ID=postgres;Password=123;Database=University property;";

readonly NpgsqlConnection conn = new NpgsqlConnection(connString);

После вызова метода Open у NpgsqlConnection соединение открывается и можно работать с БД. Однако требовалось работать только в рамках блоков try catch, поскольку осуществляется работа с внешними данными, и, в случае неполадок в канале связи, возможно повреждение базы данных.

Для работы с базой потребуется её локальное представление – DataSet. Объект DataSet содержит таблицы, которые представлены типом DataTable. Таблица, в свою очередь, состоит из столбцов и строк. Каждый столбец представляет объект DataColumn, а строка - объект DataRow.

Для быстрого считывания информации из PostgreSQL и внесения её в DataSet используются адаптеры – NpgsqlDataAdapter.

NpgsqlDataAdapter — специальный класс, позволяющие эффективно считывать таблицы из базы данных и работать с ними в рамках DataSet.

Для инициализации адаптера потребуется экземпляр класса NpgsqlCommand, который представляет собой команду SQL. Обычно для инициализации используется SQL запрос SELECT.

Для того, чтобы адаптер заполнил DataSet применяется метод Fill. В рамках данной работы для каждой таблицы был создан свой адаптер.

В листинге ***ЗАМЕНИТЬ ФОТКУ*** представлена инициализация соединения с базой данных, а также заполнение DataSet.

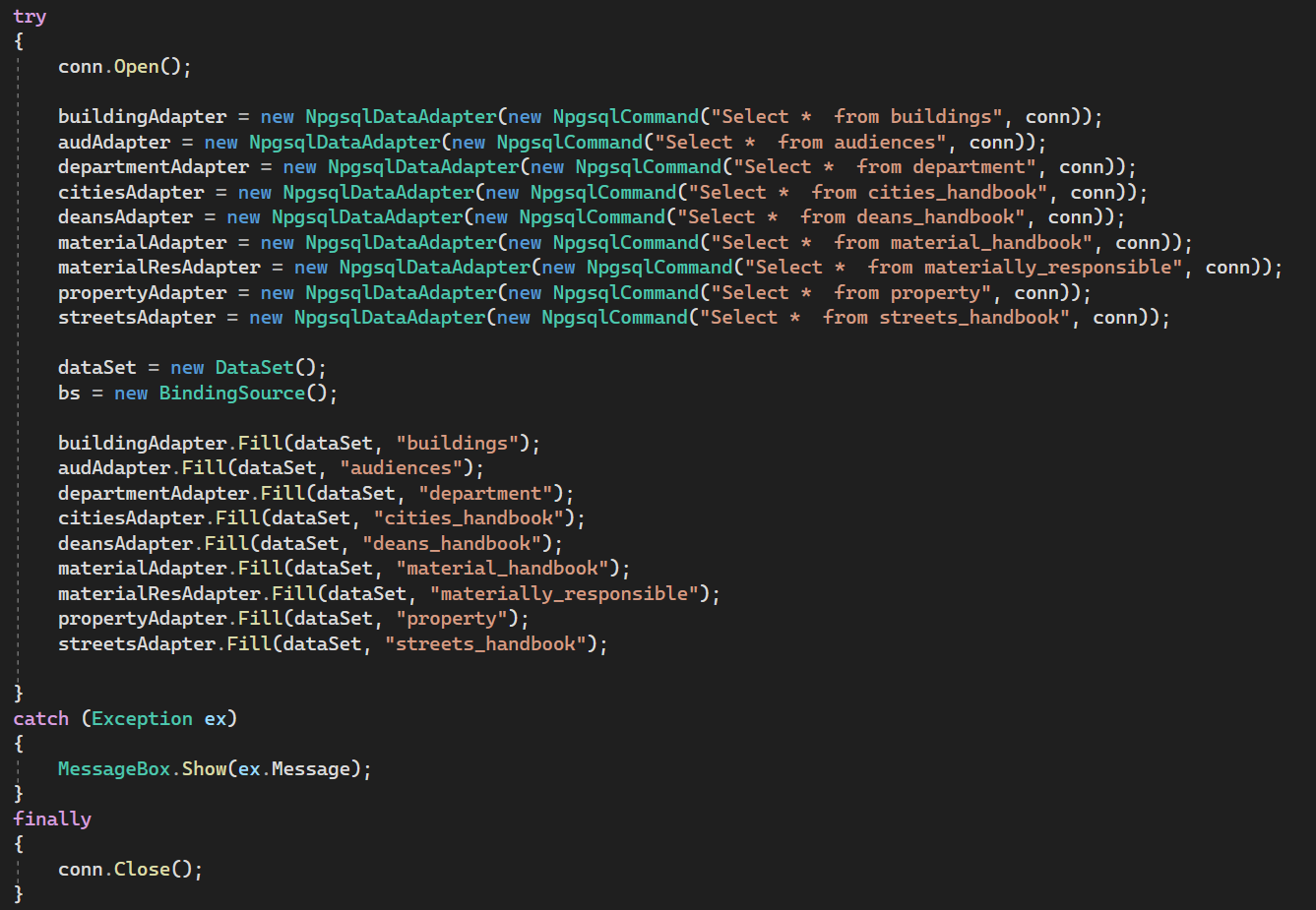


Рисунок 7 — Подключение к базе данных и чтение таблиц

Для эффективной работы с базой данных библиотека Npgsql предоставляет класс NpgsqlDataAdapter и NpgsqlCommand.

С помощью адаптеров мы можем отправлять базе данных запросы и получать ответы в виде таблицы.

Полученные таблицы с помощью метода Fill у адаптеров можно отправить в специализированный класс C# - DataSet.