

Содержание

Введение.....	3
1 Постановка задачи	4
2 Разработка модели базы данных.....	8
3 Разработка необходимых объектов	14
3.1 Таблицы.....	14
3.1 Роли. Пользователи. Имена для входа.....	17
3.2 Представления	18
3.2 Триггеры.....	19
3.3 Процедуры	19
3.3.1 Процедуры для получения данных	19
3.3.2 Процедуры для добавления данных	20
3.3.3 Процедуры для обновления данных.....	21
3.3.4 Процедуры для удаления данных.....	21
3.3.5 Процедуры для создания отчетов.....	21
3.4 Индексы.....	22
4 Описание процедур импорта и экспорта	23
4.1 Процедуры для импорта.....	23
4.2 Процедуры для экспорта	23
5 Тестирование производительности	24
6 Описание технологии	26
7 Руководство пользователя.....	31
Заключение	35
Список использованных источников.....	36
Приложение А.....	37
Приложение Б	38
Приложение В.....	39
Приложение Г	40
Приложение Д.....	41
Приложение Е	42

Введение

Курсовой проект посвящён разработке базы данных для программного средства «Организация монтажных работ».

Сфера выполнения заявок на какие-либо виды услуг (к которым относятся и монтажные работы) является весьма развитой. Однако, вместе с этим, она является недостаточно автоматизированной. Практически не существует аналогов, предоставляющих широкий ряд операций над заявками такого типа. Соответственно, разработанная под такие нужды базы данных будет востребована в сфере выполнения заявок на монтаж, а также в смежных сферах.

Основной целью курсового проекта станет проектирование базы данных для сервисов, управляющих организацией монтажных работ, а также изучение технологии SQL Server Reporting Services.

Проектируемая база данных должна поддерживать несколько ролей: администратор, бригада, клиент, а это подразумевает наличие также соответствующих имен для входа и пользователей.

Доступ к данным будет осуществляться через процедуры, привилегии на исполнение которых будет выданы соответствующим ролям.

Роль клиента предполагает возможность оставить заявку, а также получить информацию о статусе заявки по номеру (идентификатору). Она не предусматривает наличия аккаунта.

Роль бригады должна поддерживать просмотр информации о бригаде, ее работниках, просмотр назначенных ей заявок, обновление заявки (изменение ее статуса), управление вовлеченными в работу над заявкой сотрудниками бригады, просмотр отчетов с количеством своих заявок на неделю.

Наибольшее количество функций программного средства поддерживает роль администратора.

Помимо описанных выше объектов будут созданы также необходимые представления, триггеры и индексы.

При разработке базы данных планируется использовать СУБД Microsoft SQL Server.

Будет проведен импорт данных из JSON файлов, а также экспорт данных в JSON файлы.

Для создания отчетов будет использована технология SQL Server Reporting Services (SSRS).

Программное средство для демонстрации работоспособности базы данных будет представлено приложением на языке C#, с использованием технологий WPF, ADO.NET.

1 Постановка задачи

Для постановки задачи необходимо произвести анализ существующих прототипов разрабатываемого продукта.

Зачастую сервисы для организации заявок на монтаж имеют довольно ограниченный функционал, предоставляя клиенту лишь возможность оставить контактный номер телефона для связи с сотрудником для консультации (рисунок 1.1).

The screenshot shows a website with a dark background and orange text. The header includes the company logo, navigation links (Услуги, Наши проекты, Контакты, Сертификаты), social media icons, and a phone number (+375 29 176-29-24). The main content area features a large orange headline: **Монтаж отопления и водоснабжения в частном доме**. Below the headline is a list of services and guarantees in white text. On the right, there is a white box containing a form for a free consultation, with fields for 'Имя' and 'Телефон', and an orange 'Заказать консультацию' button. A chat icon is visible in the bottom right corner.

ОТОПЛЕНИЕ ВОДОСНАБЖЕНИЕ ДОМА.БЕЛ

Услуги Наши проекты Контакты Сертификаты

+375 29 176-29-24

Монтаж отопления и водоснабжения в частном доме

- Минск и Минская область;
- Более 10-ти лет опыта;
- Гарантия от 3 до 5 лет в зависимости от оборудования;
- Внимательное и уважительное отношение к вам;
- Отчитываемся на каждом этапе работы;
- За все излишки материалов вернем деньги.

Закажите бесплатную консультацию о выборе системы отопления или водоснабжения

Мы вам перезвоним

Имя

Телефон

Заказать консультацию

Рисунок 1.1 Малоинформативная форма заявки на монтаж.

Существенным минусом такого клиентского интерфейса является отсутствие возможности указать в форме необходимые для обработки заявки данные, такие, как адрес или оборудование.

Так, структура заявки на рисунке 1.2 будет более полной, чем рассмотренная выше, так как содержит адрес объекта, комментарий к заявке. Такая информация, к тому же, является абстрагированной от технической, поэтому доступна к заполнению для любого клиента.

Однако и в этой структуре не хватает некоторых важнейших данных, а именно монтируемого оборудования и стадии отделки (черновая, чистовая). Эта информация влияет на, на необходимые для выполнения заявки сотрудниками монтажной бригады инструменты и комплектующие, сроки исполнения работ – то есть, является основополагающей при автоматизации процесса обработки подобных заявок.

Сформируйте свой запрос.

Контактное лицо

Email

Телефон

Адрес объекта

Комментарии

Отправить заявку ↗

Рисунок 1.2 Более полная форма заявки на монтаж.

Поэтому к проектируемой базе данных следует предъявить следующие требования.

Структура заявки клиента должна включать следующие компоненты:

- поля для личной информации о контактном лице (заказчике работ);
- поля для адреса выполнения работ;
- количество и типы монтируемого оборудования;
- стадия отделки;
- статус исполнения заявки;
- дата регистрации заявки;
- дата выполнения работ;
- комментарий клиента к заявке;
- общую стоимость всех необходимых для выполнения заявки комплектующих.

Полностью заполненная структура такого образца позволит компании не держать в штате сотрудников-менеджеров, предоставляющих каждому клиенту примерно одинаковую информацию при консультации по телефону, а сэкономить компании как время, так и деньги, автоматизировав процесс оставления и обработки заявки.

Для обработки подобной заявки потребуются администратор, непосредственно бригада-исполнитель, а также предварительно внесенные в базу данные о стадиях отделки, необходимых инструментах и типах оборудования.

Задачей курсового проекта станет разработка базы данных для программного средства, предназначенного для автоматизации процесса обработки заявок на монтаж, а также для выполнения ряда дополнительных функций: добавления бригад, редактирования клиентов, заявок и сотрудников и других.

При работе над курсовым проектом будут задействованы:

- СУБД Microsoft SQL Server;
- технология SQL Server Reporting Services;

- язык программирования C#, технологии WPF и ADO.NET;

Для реализации поставленной задачи база данных должна предоставлять пользователям, в зависимости от их роли, следующие функциональные возможности.

Функции администратора:

- отобразить информацию о заявках (вместе с информацией об адресе выполнения работ);
- отобразить информацию о бригадах (а также их работниках);
- отобразить информацию о клиентах;
- отобразить информацию о работниках;
- просмотреть содержимое склада (инструменты, комплектующие, оборудование);
- произвести импорт данных в базу (инструменты, комплектующие) из файла формата JSON;
- произвести экспорт данных из базы (расписания для конкретного работника, а также экспорт из любой таблицы) в файл формата JSON;
- добавить бригады (вместе с которой добавляется соответствующий пользователь);
- добавить работников;
- добавить содержимое на склад (инструменты, комплектующие, оборудование);
- редактировать заявку (изменить статус, назначенную бригаду);
- редактировать бригаду (назначить бригадира);
- редактировать клиента (изменить личную информацию);
- редактировать работника (изменить личную информацию, назначить бригаде);
- редактировать содержимое склада;
- удалить бригаду;
- удалить клиента (вместе с удалением всех его заявок);
- удалить работника;
- удалить содержимое склада (инструменты, комплектующие, оборудование);
- просмотреть отчеты: о количестве заявок всех бригад на текущую неделю (в виде таблицы), о количестве заявок для конкретной бригады на текущую неделю (в виде диаграммы), отчет по заявке (включающий информацию о клиенте, адресе, оборудовании, комплектующих, необходимых инструментах).

Функции клиента:

- оставить заявку;
- просмотреть информацию о своей заявке (по номеру заявки).

Функции бригады:

- отобразить информацию о своих заявках (вместе с информацией об адресе выполнения работ);
- редактировать заявку (сменить статус);
- редактировать вовлеченных в работу над заявкой сотрудников бригады;
- просмотреть отчеты: свое расписание на текущую неделю (в виде диаграммы).

2 Разработка модели базы данных

Для хранения данных программного средства «Организация монтажных работ» была спроектирована модель базы данных TheBureau.

База данных состоит из 16 таблиц, ее схема приведена на рисунке в приложении А.

Ниже – описание назначения таблиц базы данных TheBureau.

Таблица User представляет собой описание пользователя приложения, с данными для входа. Содержит следующие поля:

- id, целочисленного типа, автоинкрементируемое, первичный ключ — уникальный номер записи в таблице;
- login, набор символов, максимальная длина 20, не допускает null значений — логин пользователя;
- password, набор символов, максимальная длина 70, не допускает null значений — пароль для входа в аккаунт;
- isDeleted, логический тип, не допускает null значений, значение по умолчанию 0 (или false) — метка для удаленных записей таблицы.

Таблица Status хранит возможные статусы обработки заявки (В обработке, В процессе, Готова). Содержит следующие поля:

- id, целочисленного типа, автоинкрементируемое, первичный ключ — уникальный номер записи в таблице;
- status, набор символов, максимальная длина 12, не допускает null значений — значение статуса.

Таблица Stage хранит возможные стадии монтажных работ для заявки (черновая отделка, чистовая отделка, обе отделки (чистовая + черновая)). Содержит следующие поля:

- id, целочисленного типа, автоинкрементируемое, первичный ключ — уникальный номер записи в таблице;
- stage, набор символов, максимальная длина 12, не допускает null значений — значение стадии монтажных работ.

Таблица Mounting хранит возможные способы монтажа оборудования (на пол, на стену). Содержит следующие поля:

- id, целочисленного типа, автоинкрементируемое, первичный ключ — уникальный номер записи в таблице;
- mounting, набор символов, максимальная длина 12, не допускает null значений — значение способа монтажа.

Таблица Tool хранит все возможные инструменты для выполнения монтажных работ на разных стадиях отделки. Содержит следующие поля:

- id, целочисленного типа, автоинкрементируемое, первичный ключ — уникальный номер записи в таблице;
- name, набор символов, максимальная длина 30, не допускает null значений

- наименование инструмента;
- stageId, целочисленного типа, не допускает null значений, используется для связи с полем id таблицы Stage — идентификатор стадии, на которой применяется инструмент;
- isDeleted, логический тип, не допускает null значений, значение по умолчанию 0 (или false) — метка для удаленных записей таблицы.

Таблица Equipment хранит всё возможное оборудование для монтажа. Содержит следующие поля:

- id, набор символов, максимальная длина 3, первичный ключ — уникальный номер записи в таблице;
- type, набор символов, максимальная длина 30, не допускает null значений — наименование инструмента;
- mountingId, целочисленного типа, не допускает null значений, используется для связи с полем id таблицы Mounting — идентификатор типа монтажа для оборудования;
- isDeleted, логический тип, не допускает null значений, значение по умолчанию 0 (или false) — метка для удаленных записей таблицы.

Таблица Accessory хранит все возможные комплектующие для разных типов оборудования.

- id, целочисленного типа, автоинкрементируемое, первичный ключ — уникальный номер записи в таблице;
- art, набор символов, максимальная длина 15, допускает null значения — артикул комплектующего;
- equipmentId, набор символов, максимальная длина 3, не допускает null значений, используется для связи с полем id таблицы Equipment — идентификатор оборудования, для монтажа которого необходимо комплектующее;
- name, набор символов, максимальная длина 150, не допускает null значений — наименование комплектующего;
- price, число с фиксированной точностью (точность 6, масштаб 2), не допускает null значений, значение по умолчанию 0 — стоимость за единицу комплектующего;
- isDeleted, логический тип, не допускает null значений, значение по умолчанию 0 (или false) — метка для удаленных записей таблицы.

Таблица Brigade хранит бригаду с указанием на её аккаунт для входа в приложение.

- id, целочисленного типа, автоинкрементируемое, первичный ключ — уникальный номер записи в таблице;
- userId, целочисленного типа, используется для связи с полем id таблицы User, допускает null значения — идентификатор пользователя, под которым бригада будет входить в систему;
- brigadierId, целочисленного типа, используется для связи с полем id

таблицы Employee, допускает null значения — идентификатор бригадира (главы) бригады;

- creationDate, дата, не допускает null значений — дата формирования бригады;
- isDeleted, логический тип, не допускает null значений, значение по умолчанию 0 (или false) — метка для удаленных записей таблицы.

Таблица Employee хранит информацию о работнике бригады.

- id, целочисленного типа, автоинкрементируемое, первичный ключ — уникальный номер записи в таблице;
- firstname, набор символов, максимальная длина 20, не допускает null значений — имя работника;
- patronymic, набор символов, максимальная длина 20, не допускает null значений — отчество работника;
- surname, набор символов, максимальная длина 20, не допускает null значений — фамилия работника;
- email, набор символов, максимальная длина 255, допускает null значения — адрес электронной почты работника;
- contactNumber, набор символов, максимальная длина 12, допускает null значения — контактный номер работника;
- brigadeId, целочисленного типа, допускает null значения, используется для связи с полем id таблицы Brigade — идентификатор бригады, в которую входит работник;
- isDeleted, логический тип, не допускает null значений, значение по умолчанию 0 (или false) — метка для удаленных записей таблицы.

Таблица Client хранит информацию о клиенте, сделавшем заявку.

- id, целочисленного типа, автоинкрементируемое, первичный ключ — уникальный номер записи в таблице;
- firstname, набор символов, максимальная длина 20, не допускает null значений — имя клиента;
- patronymic, набор символов, максимальная длина 20, не допускает null значений — отчество клиента;
- surname, набор символов, максимальная длина 20, не допускает null значений — фамилия клиента;
- email, набор символов, максимальная длина 255, допускает null значения — адрес электронной почты клиента;
- contactNumber, набор символов, максимальная длина 12, допускает null значения — контактный номер клиента;
- isDeleted, логический тип, не допускает null значений, значение по умолчанию 0 (или false) — метка для удаленных записей таблицы.

Таблица Address хранит информацию об адресе, на котором планируется осуществление монтажных работ.

- id, целочисленного типа, автоинкрементируемое, первичный ключ —

- уникальный номер записи в таблице;
- country, набор символов, максимальная длина 30, значение по умолчанию 'Belarus' — страна;
- city, набор символов, максимальная длина 30, не допускает null значений — город;
- street, набор символов, максимальная длина 30, не допускает null значений — улица;
- house, целочисленного типа, не допускает null значений — номер дома;
- corpus, набор символов, максимальная длина 10, допускает null значения — номер корпуса;
- flat, целочисленного типа, допускает null значения — номер квартиры;
- isDeleted, логический тип, не допускает null значений, значение по умолчанию 0 (или false) — метка для удаленных записей таблицы.

Таблица Request служит для хранения заявки на монтаж.

- id, целочисленного типа, автоинкрементируемое, первичный ключ — уникальный номер записи в таблице;
- clientId, целочисленного типа, не допускает null значений, используется для связи с полем id таблицы Client — идентификатор клиента, оставившего заявку;
- addressId, целочисленного типа, не допускает null значений, используется для связи с полем id таблицы Address — идентификатор адреса выполнения работ;
- stageId, целочисленного типа, не допускает null значений, используется для связи с полем id таблицы Stage — идентификатор стадии проводимых на объекте монтажных работ;
- statusId, целочисленного типа, не допускает null значений, используется для связи с полем id таблицы Status — идентификатор статуса обработки заявки;
- registerDate, дата, не допускает null значений — дата регистрации заявки;
- mountingDate, дата, не допускает null значений — дата проведения монтажных работ;
- comment, набор символов, максимальная длина 200, допускает null значения — комментарий к заявке;
- brigadeId, целочисленного типа, допускает null значения, используется для связи с полем id таблицы Brigade — идентификатор бригады, назначенной на выполнение заявки;
- proceeds, число с фиксированной точностью (точность 6, масштаб 2), допускает null значения — вычисляемое поле, хранящее общую стоимость комплектующих для заявки;
- isDeleted, логический тип, не допускает null значений, значение по умолчанию 0 (или false) — метка для удаленных записей таблицы.

Таблица RequestEquipment хранит информацию о монтируемом оборудовании для заявки.

- id, целочисленного типа, автоинкрементируемое, первичный ключ — уникальный номер записи в таблице;
- requestId, целочисленного типа, не допускает null значений, используется для связи с полем id таблицы Request — идентификатор заявки;
- equipmentId, набор символов, максимальная длина 3, не допускает null значений, используется для связи с полем id таблицы Equipment — идентификатор оборудования;
- quantity, целочисленного типа, — количество оборудования одного вида в заявке;
- isDeleted, логический тип, не допускает null значений, значение по умолчанию 0 (или false) — метка для удаленных записей таблицы.

Таблица RequestTool хранит информацию о необходимых для выполнения монтажных работ на заявке инструментах.

- id, целочисленного типа, автоинкрементируемое, первичный ключ — уникальный номер записи в таблице;
- requestId, целочисленного типа, не допускает null значений, используется для связи с полем id таблицы Request — идентификатор заявки;
- toolId, целочисленного типа, не допускает null значений, используется для связи с полем id таблицы Tool — идентификатор инструмента;
- isDeleted, логический тип, не допускает null значений, значение по умолчанию 0 (или false) — метка для удаленных записей таблицы.

Таблица RequestAccessory хранит информацию о используемых для оборудования заявки комплектующих.

- id, целочисленного типа, автоинкрементируемое, первичный ключ — уникальный номер записи в таблице;
- requestId, целочисленного типа, не допускает null значений, используется для связи с полем id таблицы Request — идентификатор заявки;
- accessoryId, целочисленного типа, не допускает null значений, используется для связи с полем id таблицы Accessory — идентификатор комплектующего;
- quantity, целочисленного типа, не допускает null значений, значение по умолчанию 1 — количество комплектующих одного типа для заявки;
- isDeleted, логический тип, не допускает null значений, значение по умолчанию 0 (или false) — метка для удаленных записей таблицы.

Таблица Schedule хранит о работающих и работавших в прошлом на заявке работников.

- id, целочисленного типа, автоинкрементируемое, первичный ключ — уникальный номер записи в таблице;
- employeeId, целочисленного типа, не допускает null значений, используется для связи с полем id таблицы Employee — идентификатор работника;
- requestId, целочисленного типа, не допускает null значений, используется

- для связи с полем id таблицы Request — идентификатор заявки;
- modified, дата и время, не допускает null значений — дата и время добавления записи об участии работника в заявке;
 - isDeleted, логический тип, не допускает null значений, значение по умолчанию 0 (или false) — метка для удаленных записей таблицы.

3 Разработка необходимых объектов

Для хранения данных программного средства «Организация монтажных работ – “Бюро Монтажника”» была создана база данных TheBuerau. Для этого использовалась система управления реляционными базами данных Microsoft SQL Server 2019.

3.1 Таблицы

Ниже представлено описание реализации описанных ранее таблиц базы данных TheBuerau в СУБД Microsoft SQL Server 2019.

Таблица User представляет собой описание пользователя приложения, с данными для входа. Содержит следующие поля:

- id int identity(1,1) primary key — уникальный номер записи в таблице;
- login nvarchar(20) not null — логин пользователя;
- password nchar(70) not null — пароль для входа в аккаунт;
- isDeleted bit not null, значение по умолчанию 0 — метка для удаленных записей таблицы.

Таблица Status хранит возможные статусы обработки заявки (В обработке, В процессе, Готова). Содержит следующие поля:

- id int identity(1,1) primary key — уникальный номер записи в таблице;
- status nvarchar(12) not null — значение статуса.

Таблица Stage хранит возможные стадии монтажных работ для заявки (черновая отделка, чистовая отделка, обе отделки (чистовая + черновая)). Содержит следующие поля:

- id int identity(1,1) primary key — уникальный номер записи в таблице;
- stage nvarchar(12) not null — значение стадии монтажных работ.

Таблица Mounting хранит возможные способы монтажа оборудования (на пол, на стену). Содержит следующие поля:

- id int identity(1,1) primary key — уникальный номер записи в таблице;
- mounting nvarchar(12) not null — значение способа монтажа.

Tool хранит все возможные инструменты для выполнения монтажных работ на разных стадиях отделки. Содержит следующие поля:

- id int identity(1,1) primary key — уникальный номер записи в таблице;
- name nvarchar(30) not null — наименование инструмента;
- stageId int not null foreign key references Stage(id) — идентификатор стадии, на которой применяется инструмент;
- isDeleted bit not null, значение по умолчанию 0 — метка для удаленных записей таблицы.

Таблица Equipment хранит всё возможное оборудование для монтажа.

Содержит следующие поля:

- id nvarchar(3) primary key — уникальный номер записи в таблице;
- type nvarchar(30) not null — наименование инструмента;
- mountingId int not null foreign key references Mounting(id) — идентификатор типа монтажа для оборудования;
- isDeleted bit not null, значение по умолчанию 0 — метка для удаленных записей таблицы.

Таблица Accessory хранит все возможные комплектующие для разных типов оборудования.

- id int identity(1,1) primary key — уникальный номер записи в таблице;
- art nvarchar(15) null — артикул комплектующего;
- equipmentId nvarchar(3) not null foreign key references Equipment(id) — идентификатор оборудования, для монтажа которого необходимо комплектующее;
- name nvarchar(150) not null — наименование комплектующего;
- price decimal(6,2) not null default 0 — стоимость за единицу комплектующего;
- isDeleted bit not null, значение по умолчанию 0 — метка для удаленных записей таблицы.

Таблица Brigade хранит бригаду с указанием на её аккаунт для входа в приложение.

- id int identity(1,1) primary key — уникальный номер записи в таблице;
- userId int foreign key references [User](id) null — идентификатор пользователя, под которым бригада будет входить в систему;
- brigadierId int foreign key references [Employee](id) null — идентификатор бригадира (главы) бригады;
- creationDate date not null — дата формирования бригады;
- isDeleted bit not null, значение по умолчанию 0 — метка для удаленных записей таблицы.

Таблица Employee хранит информацию о работнике бригады.

- id int identity(1,1) primary key — уникальный номер записи в таблице;
- firstname nvarchar(20) not null — имя работника;
- patronymic nvarchar(20) not null — отчество работника;
- surname nvarchar(20) not null — фамилия работника;
- email nvarchar(255) null — адрес электронной почты работника;
- contactNumber nvarchar(12) null — контактный номер работника;
- brigadeId int null foreign key references Brigade(id) — идентификатор бригады, в которую входит работник;
- isDeleted bit not null, значение по умолчанию 0 — метка для удаленных записей таблицы.

Таблица Client хранит информацию о клиенте, сделавшем заявку.

- id int identity(1,1) primary key — уникальный номер записи в таблице;
- firstname nvarchar(20) not null — имя клиента;
- patronymic nvarchar(20) not null — отчество клиента;
- surname nvarchar(20) not null — фамилия клиента;
- email nvarchar(255) null — адрес электронной почты клиента;
- contactNumber nvarchar(12) null — контактный номер клиента;
- isDeleted bit not null, значение по умолчанию 0 — метка для удаленных записей таблицы.

Таблица Address хранит информацию об адресе, на котором планируется осуществление монтажных работ.

- id int identity(1,1) primary key — уникальный номер записи в таблице;
- country nvarchar(30), значение по умолчанию 'Belarus' — страна;
- city nvarchar(30) not null — город;
- street nvarchar(30) not null — улица;
- house int not null — номер дома;
- corpus nvarchar(10) null — номер корпуса;
- flat int null — номер квартиры;
- isDeleted bit not null, значение по умолчанию 0 — метка для удаленных записей таблицы.

Таблица Request служит для хранения заявки на монтаж.

- id int identity(1,1) primary key — уникальный номер записи в таблице;
- clientId int not null foreign key references Client(id) — идентификатор клиента, оставившего заявку;
- addressId int not null foreign key references Address(id) — идентификатор адреса выполнения работ;
- stageId int not null foreign key references Stage(id) — идентификатор стадии проводимых на объекте монтажных работ;
- statusId int not null foreign key references [Status](id) — идентификатор статуса обработки заявки;
- registerDate date not null — дата регистрации заявки;
- mountingDate date not null — дата проведения монтажных работ;
- comment nvarchar(200) null — комментарий к заявке;
- brigadeId int null foreign key references Brigade(id) — идентификатор бригады, назначенной на выполнение заявки;
- proceeds decimal(6,2) null — вычисляемое поле, хранящее общую стоимость комплектующих для заявки;
- isDeleted bit not null, значение по умолчанию 0 — метка для удаленных записей таблицы.

Таблица RequestEquipment хранит информацию о монтируемом оборудовании для заявки.

- id int identity(1,1) primary key — уникальный номер записи в таблице;
- requestId int not null foreign key references Request(id) — идентификатор

заявки;

- equipmentId nvarchar(3) not null foreign key references Equipment(id) — идентификатор оборудования;
- quantity int — количество оборудования одного вида в заявке;
- isDeleted bit not null, значение по умолчанию 0 — метка для удаленных записей таблицы.

Таблица RequestTool хранит информацию о необходимых для выполнения монтажных работ на заявке инструментах.

- id int identity(1,1) primary key — уникальный номер записи в таблице;
- requestId int not null foreign key references Request(id) — идентификатор заявки;
- toolId int not null foreign key references Tool(id) — идентификатор инструмента;
- isDeleted bit not null, значение по умолчанию 0 — метка для удаленных записей таблицы.

Таблица RequestAccessory хранит информацию о используемых для оборудования заявки комплектующих.

- id int identity(1,1) primary key — уникальный номер записи в таблице;
- requestId int not null foreign key references Request(id) — идентификатор заявки;
- accessoryId int not null foreign key references Accessory(id) — идентификатор комплектующего;
- quantity int not null, значение по умолчанию 1 — количество комплектующих одного типа для заявки;
- isDeleted bit not null, значение по умолчанию 0 — метка для удаленных записей таблицы.

Таблица Schedule хранит о работающих и работавших в прошлом на заявке работников.

- id int identity(1,1) primary key — уникальный номер записи в таблице;
- employeeId int not null foreign key references Employee(id) — идентификатор работника;
- requestId int not null foreign key references Request(id) — идентификатор заявки;
- modified datetime not null — дата и время добавления записи об участии работника в заявке;
- isDeleted bit not null, значение по умолчанию 0 — метка для удаленных записей таблицы.

3.1 Роли. Пользователи. Имена для входа

Для удовлетворения требований проектируемой базы данных понадобилось создание трех ролей.

Роль администратора имеет привилегии на выполнение всех хранимых процедур базы данных.

Роли клиента доступно выполнение процедур для оставления заявки (LeaveRequest, AddRequestEquipment) и поиска заявки по идентификатору (GetRequestsForClientById), а также некоторых других.

Роль бригады наделена привилегиями для выполнения процедур, некоторые из которых:

- отобразить информацию о своих заявках (вместе с информацией об адресе выполнения работ) (GetAllRequestsForBrigade, GetRequestEquipmentByRequestId);
- редактировать заявку (UpdateRequestByBrigade);
- редактировать вовлеченных в работу над заявкой сотрудников бригады (SetScheduleRecord, ClearRequestSchedule, GetEmployeesForBrigade).

Для обеспечения доступа под несколькими ролями были созданы соответствующие имена для входа и пользователи базы данных. Пример для бригады приведен в листинге ниже:

```
create login [brigade_login] with password=N'brigade', default_database=[TheBureau],
default_language=[русский],
check_expiration=off, check_policy=on
go
create user [thebureau_brigade_user] for login [brigade_login];
go
```

Затем были созданы роли. Им выданы необходимые разрешения на выполнение процедур. Пример для бригады приведен в листинге ниже:

```
create role thebureau_brigade_role;
alter role thebureau_brigade_role add member thebureau_brigade_user;
go

grant execute on GetStatusIdByName to thebureau_brigade_role;
grant execute on GetBrigadeByUserId to thebureau_brigade_role;
grant execute on GetAllRequestsForBrigade to thebureau_brigade_role;
grant execute on GetRequestEquipmentByRequestId to thebureau_brigade_role;
grant execute on GetEmployeesForBrigade to thebureau_brigade_role;
grant execute on ClearRequestSchedule to thebureau_brigade_role;
grant execute on SetScheduleRecord to thebureau_brigade_role;
grant execute on UpdateRequestByBrigade to thebureau_brigade_role;
go
```

3.2 Представления

В структуре практически всех таблиц имеется поле «isDeleted», представляющее собой метку, отображающую, удалена ли запись из таблицы.

Значение поля по умолчанию — 0 (false).

Установка значения в 1 (true) используется вместо удаления данных из таблиц.

Для таблиц, содержащих эту метку, были созданы представления, позволяющие получать из таблиц только актуальные (не удаленные) данные. Дальнейшая работа с таблицами осуществляется через эти представления.

Для представлений, для которых в дальнейшем будут созданы индексы, при

создании указывается опция SCHEMABINDING.

Пример представления BrigadeView, отображающего данные из таблицы Brigade, приведен в листинге ниже:

```
create or alter view [BrigadeView]
with schemabinding
as select id, userId, brigadierId, creationDate from [dbo].[Brigade] where isDeleted = 0; --если
не "удалено" из таблицы
go
```

Также были созданы представления для получения данных в результате запроса к нескольким таблицам. Пример представления ForBrigadeView, отображающего бригаду с работниками, приведен в листинге ниже:

```
--отобразить бригаду вместе с работниками
create or alter view ForBrigadeView
as
select [dbo].[BrigadeView].id, [dbo].[BrigadeView].userId,
[dbo].[BrigadeView].brigadierId, --информация о бригаде
[dbo].[EmployeeView].firstname, [dbo].[EmployeeView].surname,
[dbo].[EmployeeView].patronymic, [dbo].[EmployeeView].email,
[dbo].[EmployeeView].contactNumber --информация о работниках
from [dbo].[BrigadeView] left join -- в бригаде может не быть работников
[dbo].[EmployeeView] on [dbo].[EmployeeView].brigadeId = [dbo].[BrigadeView].id
go
```

3.2 Триггеры

Для заполнения таких таблиц, как RequestTool, RequestAccessory, созданы триггеры.

Так, триггер SetToolsByRequestTrigger срабатывает после добавления в таблицу Request новой заявки и заносит соответствующие заявке необходимые инструменты в таблицу RequestTool. Код триггера SetToolsByRequestTrigger приведен в листинге:

```
create or alter trigger SetToolsByRequestTrigger
on [dbo].[Request]
after insert
as
begin try
    declare @requestId int = (select id from inserted); --id новой заявки
    declare @stageId int = (select stageId from inserted); --стадия выполнения
работ новой заявки
    insert into [dbo].[RequestToolView] (requestId, toolId) --добавить все
соответствующие инструменты
    select @requestId, id from [dbo].[ToolView] where stageId = @stageId;
end try
begin catch
    print 'Error: ' + cast(error_number() as varchar(6)) + ': ' + error_message();
end catch
Go
```

3.3 Процедуры

3.3.1 Процедуры для получения данных

Разработаны и реализованы процедуры для получения данных из таблиц

(преимущественно посредством обращения к соответствующим представлениям).

Пример процедуры для получения всех актуальных записей из таблицы Brigade приведен в листинге ниже.

```
create or alter procedure GetAllBrigades
as begin
    begin try
        select id, userId, brigadierId, creationDate from [dbo].[BrigadeView]
    end try
    begin catch
        print 'Error: ' + cast(error_number() as varchar(6)) + ': ' + error_message();
    end catch
end;
go
```

Пример процедуры для получения актуальной записи из таблицы Brigade по идентификатору приведен в листинге ниже.

```
create or alter procedure GetBrigade(@id int)
as begin
    begin try
        select id, userId, brigadierId, creationDate from [dbo].[BrigadeView] where id =
@id
    end try
    begin catch
        print 'Error: ' + cast(error_number() as varchar(6)) + ': ' + error_message();
    end catch
end;
go
```

Пример процедуры для получения количества невыполненных заявок приведен в листинге ниже.

```
create or alter procedure GetRequestsCountByStatusId(@statusId int)
as begin
    begin try
        select count(id) as rqcount from [dbo].[RequestView] where statusId = @statusId;
    end try
    begin catch
        print 'Error: ' + cast(error_number() as varchar(6)) + ': ' + error_message();
    end catch
end;
go
```

3.3.2 Процедуры для добавления данных

Разработаны и реализованы процедуры для добавления данных в базу. Пример процедуры для создания новой бригады приведен в листинге ниже.

```
create or alter procedure AddBrigade(@password nchar(70))
as begin
    begin try
        begin tran
            --добавить новую бригаду, изначально без пользователя для входа и без бригадира
            --установить для бригады время создания - сейчас
            insert into [dbo].[BrigadeView] (userId, brigadierId, creationDate) values(NULL,
NULL, getdate());
            declare @brigadeId int = SCOPE_IDENTITY(); --получить id созданной бригады
            declare @login nvarchar(20) = concat('brigade', @brigadeId); --создать имя
пользователя для бригады как brigade+id
        end tran
    end try
    begin catch
        print 'Error: ' + cast(error_number() as varchar(6)) + ': ' + error_message();
    end catch
end;
go
```

```

--добавить нового пользователя с переданным как параметр процедуры паролем
insert into [dbo].[UserView] ([login], [password]) values(@login, @password);
declare @userId int = SCOPE_IDENTITY(); --получить id созданного пользователя
--обновить данные о пользователе для бригады
update [dbo].[BrigadeView] set userId = @userId where id = @brigadeId;
select * from [dbo].[BrigadeView] where id = brigadeId; --вернуть id бригады
commit;
end try
begin catch
    if @@trancount > 0 rollback;
    print 'Error: ' + cast(error_number() as varchar(6)) + ': ' + error_message();
end catch
end;
go

```

3.3.3 Процедуры для обновления данных

Разработаны и реализованы процедуры для обновления данных в базе. Пример процедуры для обновления единицы оборудования приведен в листинге ниже.

```

create or alter procedure UpdateEquipment(@id nvarchar(3), @type nvarchar(30), @mountingId int)
as begin
    begin try
        update [dbo].[EquipmentView] set [type] = @type, mountingId = @mountingId where id = @id;
    end try
    begin catch
        print 'Error: ' + cast(error_number() as varchar(6)) + ': ' + error_message();
    end catch
end;
Go

```

3.3.4 Процедуры для удаления данных

Разработаны и реализованы процедуры для удаления данных из базы (пометить как удаленное). Пример процедуры для удаления комплектующего приведен в листинге ниже.

```

create or alter procedure DeleteAccessory (@accessoryId int)
as begin
    begin try
        begin tran
            --пометить как удаленные комплектующие в зависимых заявках
            update [dbo].[RequestAccessory] set isDeleted = 1 where [accessoryId] = @accessoryId;
            --пометить как удаленные комплектующее
            update [dbo].[Accessory] set isDeleted = 1 where [id] = @accessoryId;
        commit;
    end try
    begin catch
        if @@trancount > 0 rollback;
        print 'Error: ' + cast(error_number() as varchar(6)) + ': ' + error_message();
    end catch
end;

```

3.3.5 Процедуры для создания отчетов

Разработаны и реализованы процедуры, на основе которых строятся отчеты SQL Server Reporting Services. Пример процедуры, отображающей количество

заявок бригады на текущую неделю, приведен в листинге ниже.

```
create or alter procedure BrigadeScheduleReport(@brigadeId int)
as begin
    begin try
        --все заявки для этой бригады, которые надо выполнять в эту неделю, (еще не завершены)
        select [понедельник], [вторник], [среда], [четверг], [пятница], [суббота], [воскресенье]
        from (
            select [dbo].[RequestView].id as id, datename(weekday, mountingdate) as wday
            from [dbo].[RequestView] join [Status] on [dbo].[RequestView].statusId =
[Status].id
            where
                brigadeId = @brigadeId and --бригада передается как параметр процедуры
                datename(week, mountingdate) = datename(week, getdate()) and --текущая неделя
                status in('InProgress', 'InProcessing') --заявка еще не завершена
            ) s
        pivot (count(id) for wday in([понедельник],[вторник],[среда],[четверг], [пятница],
[суббота], [воскресенье])) p;
    end try
    begin catch
        print 'Error: ' + cast(error_number() as varchar(6)) + ': ' + error_message();
    end catch
end;
```

3.4 Индексы

Для работы с наиболее часто используемыми таблицами и представлениями были разработаны индексы по наиболее используемым выборкам. Для примера представлен некластеризованный индекс покрытия для таблицы Accessory. Его код представлен в листинге:

```
create nonclustered index nocl_Accessory_index on [dbo].[Accessory] (id) include(name, art,
equipmentId, price);
```

4 Описание процедур импорта и экспорта

4.1 Процедуры для импорта

В качестве примера для процедур импорта представлена процедура ImportTools для добавления инструментов из JSON-файла в таблицу Tool базы данных. Код процедуры ImportTools приведен в листинге ниже:

```
--Параметр процедуры файл (.json), содержащий инструменты
create or alter procedure ImportTools(@filename nvarchar(max))
as begin
    begin try
        declare @json varchar(max)                --импорт из передаваемого файла
        declare @command nvarchar(max)= N'select @json = bulkcolumn from openrowset(BULK ''' +
@filename + ''', SINGLE_CLOB) Imported
        insert into [dbo].[Tool]([name], [stageId])
        output inserted.id as [id], inserted.name as [name], inserted.stageId as [stageId]
        select * from openjson(@json) with ([name] nvarchar(30), [stageId] int)';
        exec sp_executesql @command, N'@filename nvarchar(max), @json varchar(max) output',
@filename, @json output;
    end try
    begin catch
        print 'Error: ' + cast(error_number() as varchar(6)) + ': ' + error_message();
    end catch
end;
go
```

4.2 Процедуры для экспорта

В качестве процедуры экспорта реализована процедура ExportTable для сохранения содержимого таблицы (представления), название которой передается с параметром, в JSON файл. Код процедуры ExportTable приведен в листинге ниже:

```
--Pass folder path to create file (TableYYYYMMDD-HHMMSS.json)
create or alter procedure ExportTable(@table varchar(30), @path varchar(260))
as begin
    begin try
        declare @result int;
        --текущая дата в формате YYYYMMDD-HHMMSS
        declare @date varchar(100) = (select (replace(convert(varchar, getdate(), 23), '-', '')) +
'-' + replace(convert(varchar, getdate(), 8), ':', '')) );
        --создать имя файла в формате TableYYYYMMDD-HHMMSS.json
        declare @filename varchar(255) = (select concat(@path, '\', @table, @date, '.json'));
        --параметр процедуры - имя таблицы на экспорт
        declare @command varchar(700)= 'bcp "SELECT * FROM [TheBureau].[dbo].['+@table+'] FOR
JSON AUTO, INCLUDE_NULL_VALUES;" queryout ''' + @filename + ''' -t, -k -c -C UTF-8 -S
PASYAGITKAASUS -T'
        exec @result = xp_cmdshell @command, no_output
        select @filename as [filename], @result as [success] --result = 0 -> success
    end try
    begin catch
        select ('Error: ' + cast(error_number() as varchar(6)) + ': ' + error_message()) as
error;
    end catch
end;
go
```

5 Тестирование производительности

Для примера тестирования производительности базы данных были выбрана таблица Accessory. В нее было добавлено 106 855 строк (рисунок 5.1).

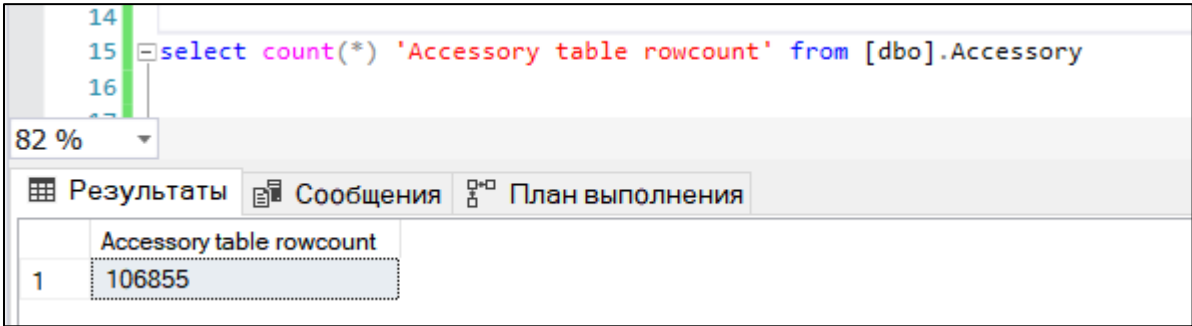


Рисунок 5.1 Количество строк в таблице Accessory.

Изначально для таблицы Accessory имеется лишь кластеризованный индекс, созданный при добавлении первичного ключа для поля id.

Все запросы будут выполняться для соответствующего таблице Accessory представления AccessoryView.

Первоначально при выполнении запроса выборки из представления AccessoryView задействуется кластеризованный индекс таблицы Accessory. План выполнения представлен на рисунке 5.2.

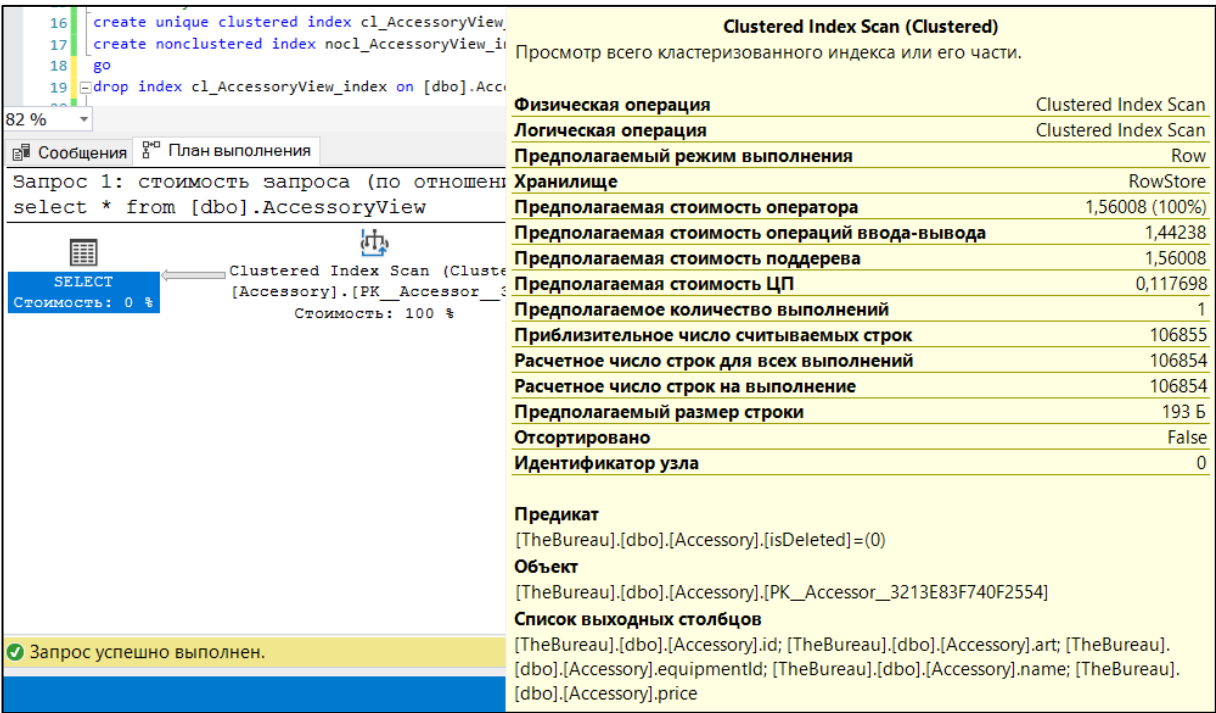


Рисунок 5.2 План выполнения запроса до создания индекса.

Показатели указывают на необходимость внесения изменений в структуру. Создадим индексы для представления.

Уникальный кластеризованный индекс cl_AccessoryView_index,

представленный в листинге ниже:

```
create unique clustered index cl_AccessoryView_index on [dbo].AccessoryView (id);
```

Некластеризованный индекс покрытия nocl_AccessoryView_index, представленный в листинге ниже:

```
create nonclustered index nocl_AccessoryView_index on [dbo].AccessoryView (id) include ([name], art, equipmentId, price);
```

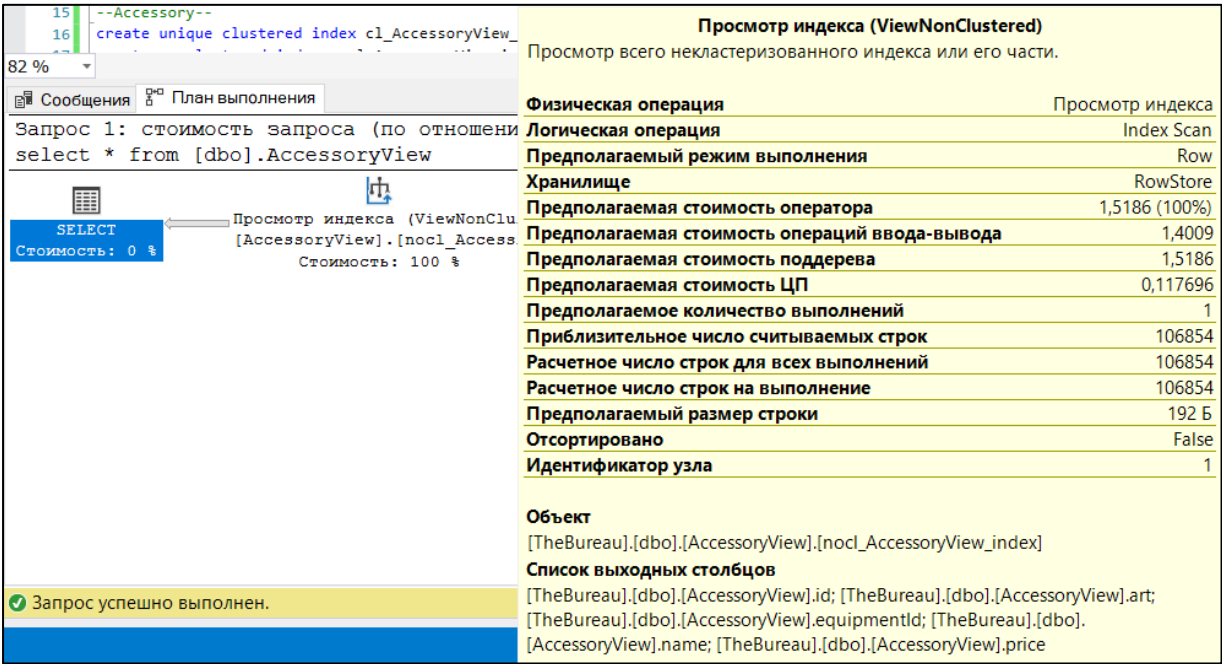


Рисунок 5.3 План выполнения запроса после создания индекса.

План выполнения запроса с использованием некластеризованного индекса для представления представлен на рисунке 5.3.

6 Описание технологии

SQL Server Reporting Services (SSRS) – это службы для разработки, построения, доставки и просмотра отчетов.

Технология гармонично вписывается в концепцию организации монтажных работ, как как позволяет визуализировать расписание бригады на неделю, а также представить в удобном формате полную информацию о заявке, включая информацию о клиенте, адресе, необходимых инструментах, номенклатуре оборудования и комплектующих.

Работа с технологией будет описана на примере разработки отчета с расписанием бригад на неделю.

Для начала работы необходимо создать проект по шаблону Reporting Project в Visual Studio 2019. Он предусматривает наличие трёх видов компонентов: источники данных, наборы данных и непосредственно отчёты.

Необходимо начать с создания источника данных. На этом этапе требуется указать имя сервера, проверку подлинности, имя базы данных (рисунок 6.1).

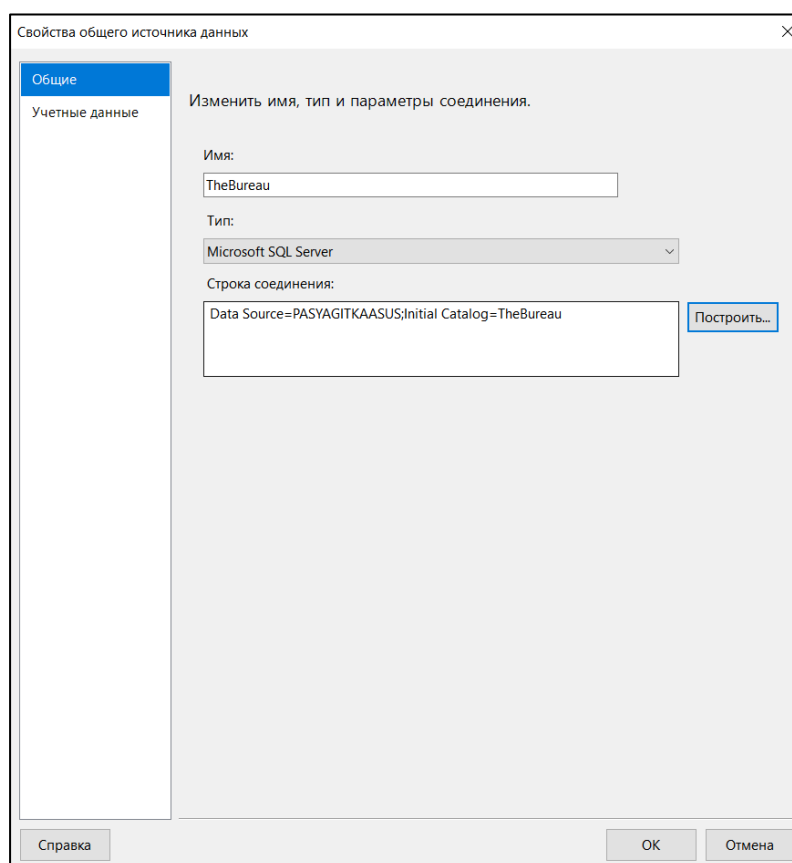


Рисунок 6.1 Создание источника данных.

Следующим шагом будет создание набора данных (DataSet) на основе созданного источника. На этом этапе есть возможность указать как тип запроса предварительно созданную хранимую процедуру (рисунок 6.2). В данном случае BrigadeScheduleReport, описание которой было приведено ранее.

Свойства общего набора данных

Запрос

Выбор источника данных и создание запроса

Имя:
ScheduleDataSet

Источник данных:
TheBureau Создать...

Тип запроса:
☐ Текст
 ☐ Таблица
 ☒ Хранимая процедура

Выберите или введите имя хранимой процедуры:
 BrigadeScheduleReport

Конструктор запросов... Обновить поля

Время ожидания (сек.):
0

Справка ОК Отмена

Рисунок 6.2 Создание набора данных. Запрос.

После добавления процедуры вкладка «Поля» обновится, и отобразятся данные запроса процедуры (рисунок 6.3).

Свойства общего набора данных

Поля

Изменение запроса и вычисляемых полей.

Добавить Удалить ↕ ↗

Имя поля	Источник поля
понедельник	понедельник
вторник	вторник
среда	среда
четверг	четверг
пятница	пятница
суббота	суббота
воскресенье	воскресенье

Справка ОК Отмена

Рисунок 6.3 Создание набора данных. Поля.

На вкладке «Параметры», в свою очередь, можно будет увидеть принимаемый процедурой параметр (рисунок 6.4).

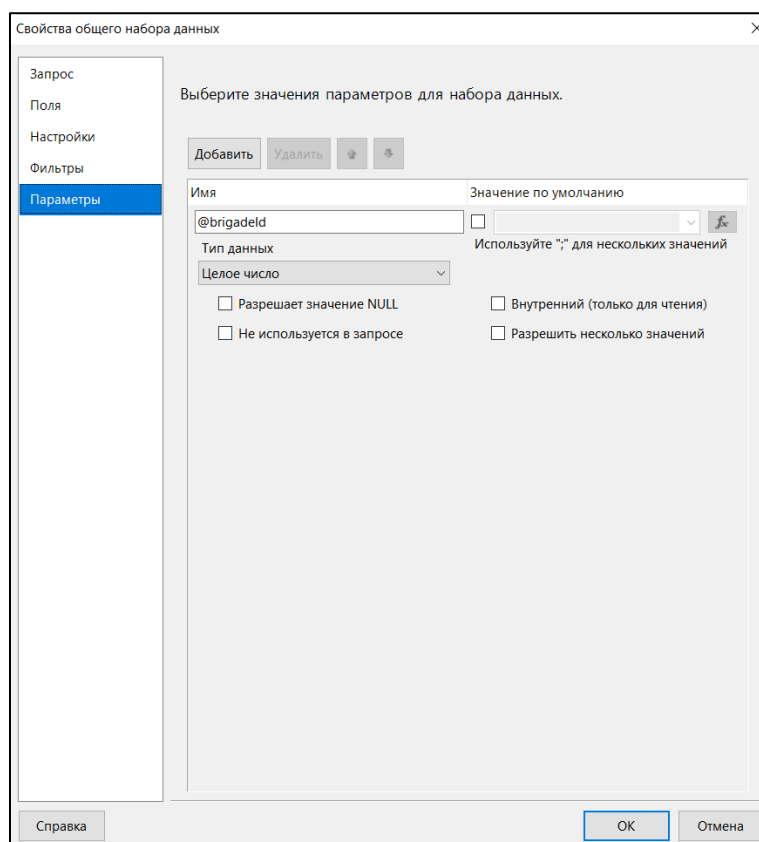


Рисунок 6.4 Создание набора данных. Параметры.

Следующим шагом станет создание отчета и добавление источником его данных созданного ранее набора данных ScheduleDataSet. На созданный отчет необходимо добавить элементы для отображения данных из набора. Их можно найти на вкладке «Панель элементов» (рисунок 6.5).

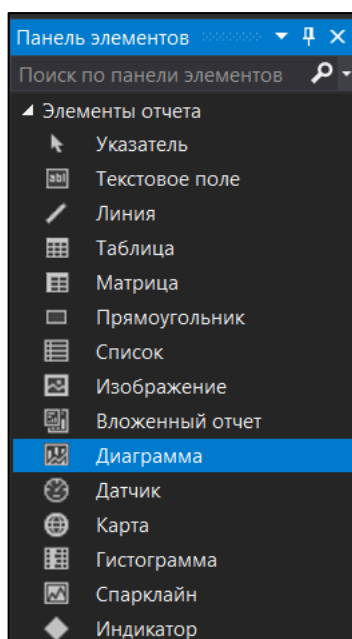


Рисунок 6.5 Панель элементов.

В случае с отчетом для расписания бригады был выбран элемент «диаграмма» (рисунок 6.6).

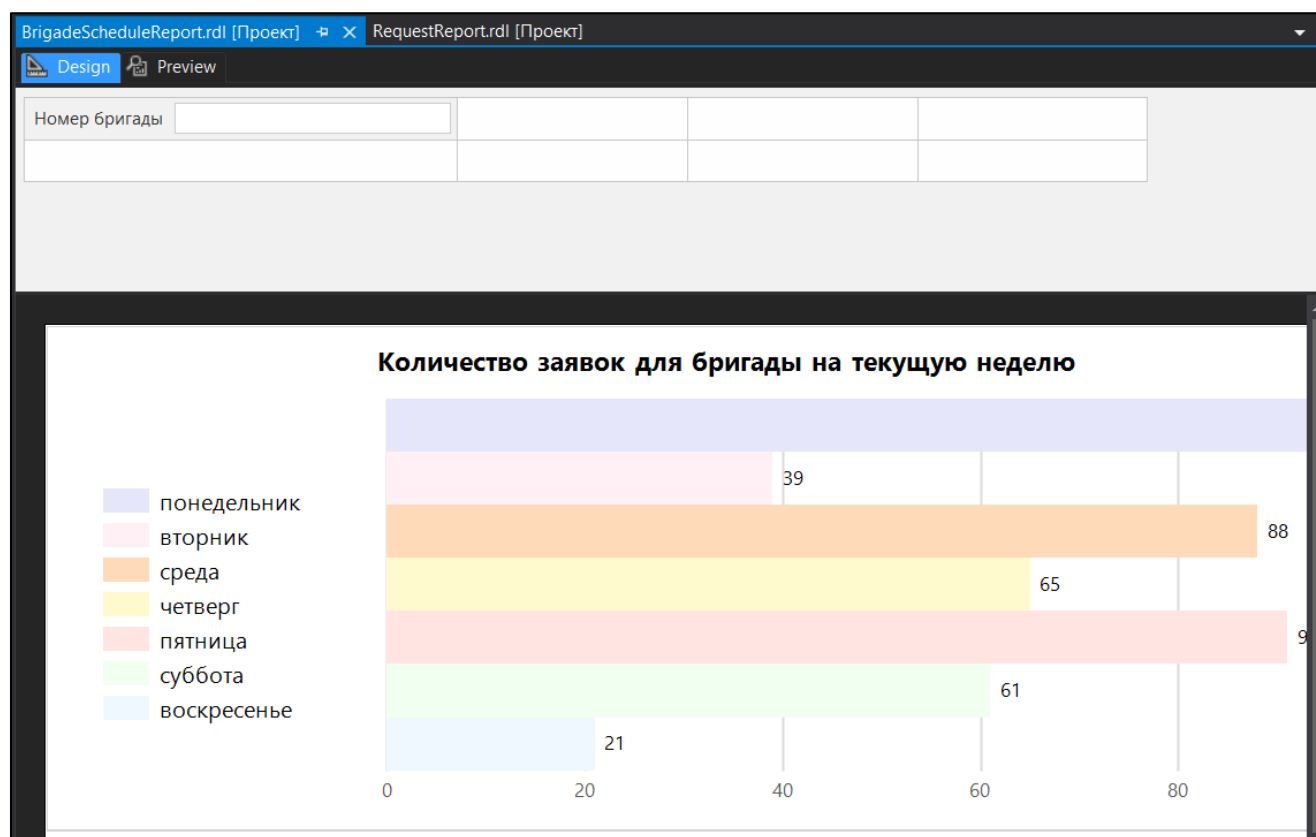


Рисунок 6.6 Отчет с диаграммой.

В результате готовый отчет, интегрированный в приложение для демонстрации, выглядит как на рисунке 6.7.

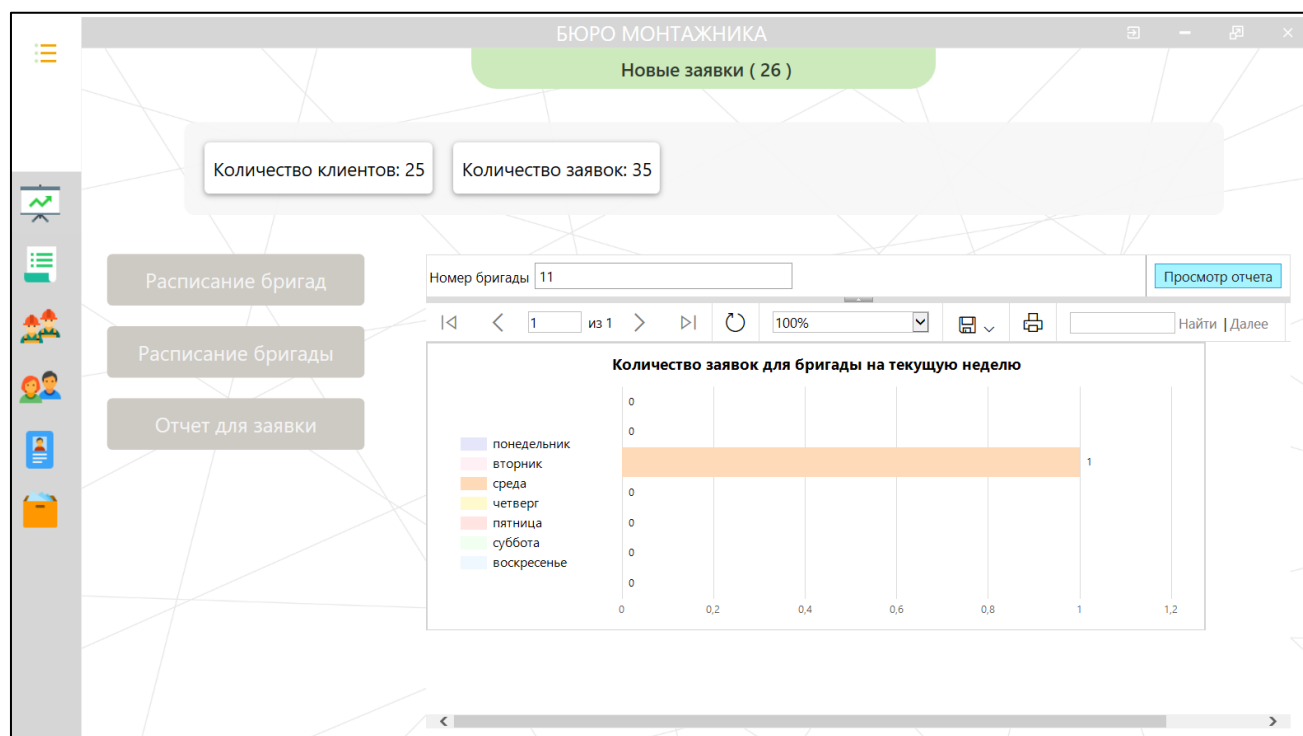


Рисунок 6.7 Отчет для бригады номер 11.

Вложенный отчет для инструментов представлен в приложении Б.
Вложенный отчет для комплектующих представлен в приложении В.
Отчет для всех бригад представлен в приложении Г.
Отчет для бригады представлен в приложении Д.
Отчет для заявки представлен в приложении Е.

7 Руководство пользователя

После запуска приложения в первую очередь открывается окно, в котором содержатся две вкладки: клиент и вход. Для входа под клиентом необходимо нажать на кнопку «Войти как клиент».

Если нужно войти как бригада или администратор, необходимо нажать на кнопку «Вход» и перейти на вкладку со входом, представленную на рисунке 7.1.

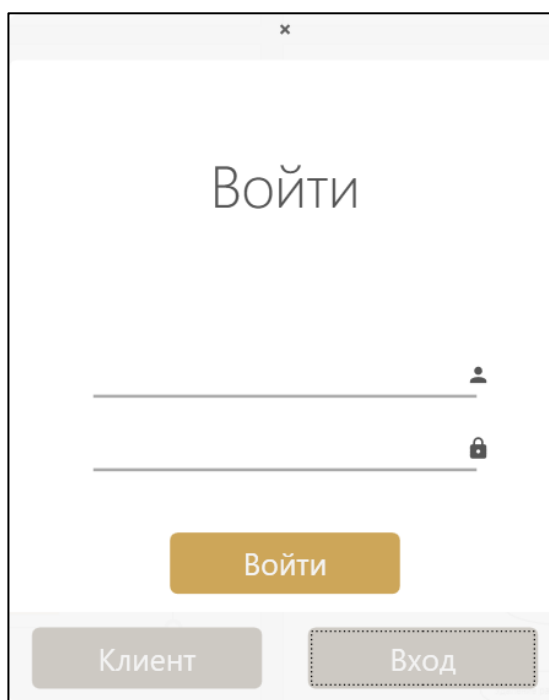


Рисунок 7.1 Вход в аккаунт бригады или администратора.

Ниже представлено окно клиента. Если необходимо оставить заявку, нужно нажать на кнопку «Оставить заявку», после чего появится форма (рисунок 7.2).

Заказчик

Зинович 375445686953
 (Фамилия) (Контактный телефон)

Елизавета liza@gmail.com
 (Имя) (Электронная почта)

Игоревна
 (Отчество)

Заявка #5
 Зинович Елизавета Игоревна, #6
 Бригада

Беларусь, г.Минск, ул.С...
 Комментарий
 Чистовая отделка

Адрес монтажа

Беларусь 0 0 0
 (Страна) (Улица) (Корпус)

0 0 0
 (Город) (Дом) (Квартира)

Выберите количество приборов по типам подключения

☐ Радиаторы ☐ Конвекторы

Пол Стена Пол Стена
 0 0 0 0

☐ Внутрипольные конвекторы
 Пол
 0

☒ Черновая отделка ☐ Чистовая отделка

Комментарий к заявке (до 200 символов)

Выберите дату монтажа: 13.12.2021

Отправить

Рисунок 7.2 Окно клиента.

При входе в аккаунт бригады открывается главное окно бригады, представленное на рисунке 7.3. В нем можно просматривать информацию о бригаде, об оборудовании конкретной заявки, редактировать статус заявки, задействованных работников, а также просмотреть расписание бригады на текущую неделю (рисунок 7.4).

БРИГАДА

Иванов
 Андреев
 Алексеев
 Семенов

Оборудование Способ монтажа Количество

радиатор	Стена	1
напольный конвектор	Стена	2

Заявка #73 В процессе

Клиент Име Отч, #53 Беларусь, г.Минск, ул.Мира, д.6, к.1, кв.22
 Бригада 11 Код от домофона - 1111.

Чистовая и черновая отделки 15.12.2021

Заявка #72 Готово

Рисунок 7.3 Окно бригады.

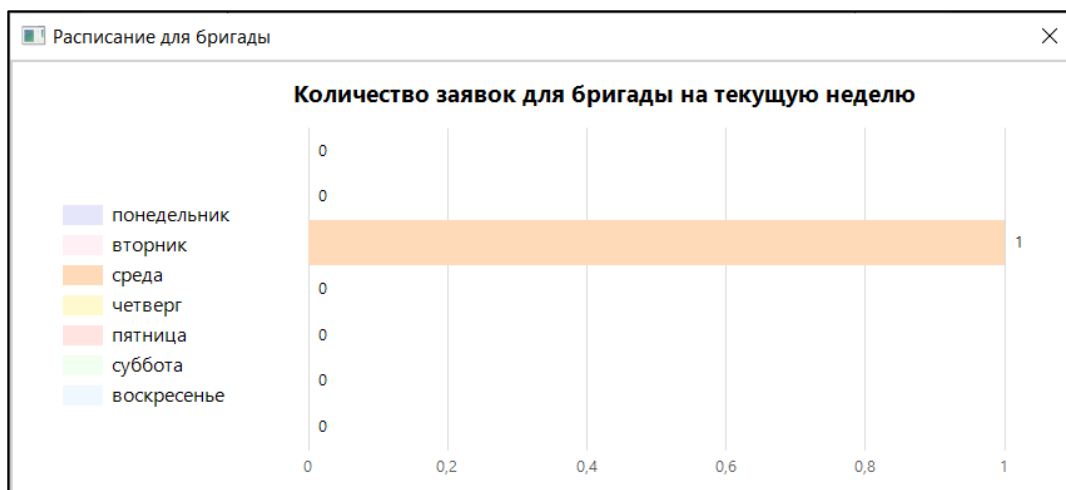


Рисунок 7.4 Количество заявок для бригады.

При входе в аккаунт администратора открывается главное окно администратора, представленное на рисунке 7.5 с активной вкладкой «Статистика». На ней можно ознакомиться с отчетами по количеству заявок для бригад на каждый день недели, а также составить отчет для заявки.

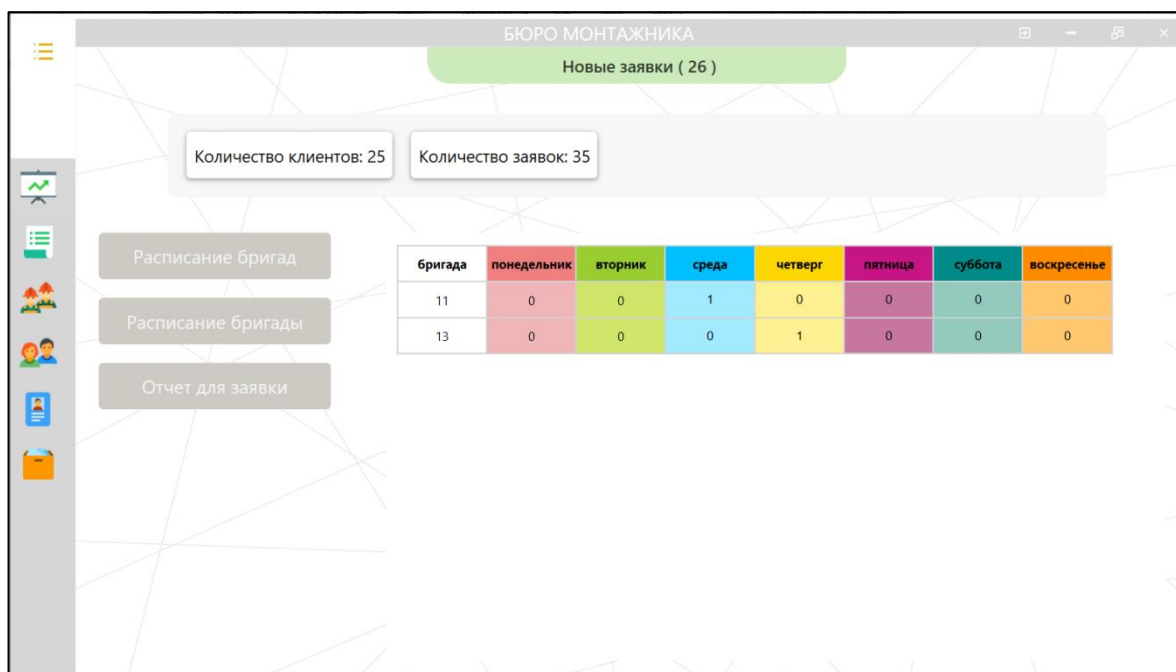


Рисунок 7.5 Окно администратора, вкладка «Статистика».

Если перейти на вкладку «Заявки», можно совершить редактирование заявки (изменения её статуса, назначение бригаде) во всплывающем окне. Изменения можно сохранить нажатием на зеленую кнопку с галкой или отменить нажатием на красный крест.

На вкладке «Бригады» (рисунок 7.6) можно добавить бригаду нажатием кнопки со знаком плюс или удалить выделенную бригаду нажатием на красный крест.



Рисунок 7.6 Окно администратора, вкладка «Бригады».

На вкладке «Клиенты» можно редактировать клиента нажатием на карандаш или удалить выделенного клиента.

Вкладка «Работники» практически аналогична вкладке «Клиенты».

На вкладке «Склад», изображенной на рисунке 7.7, можно редактировать, просматривать список инструментов, комплектующих и приборов, а также производить экспорт и импорт номенклатуры.

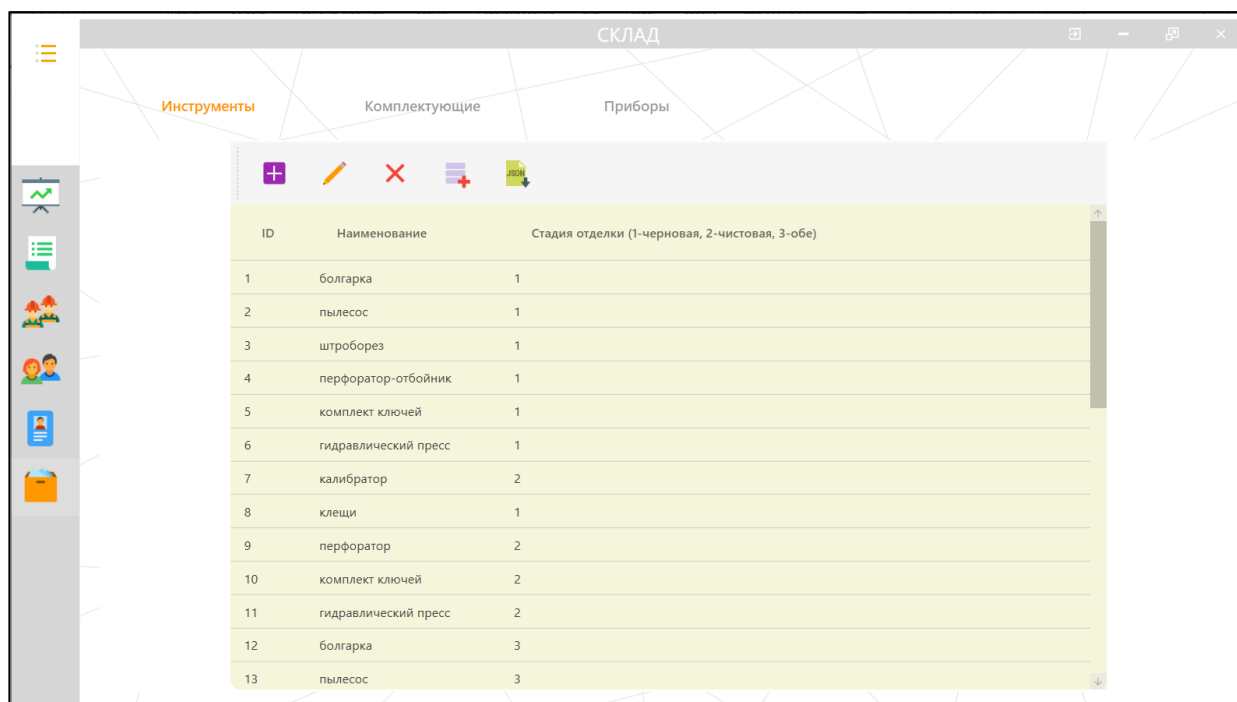


Рисунок 7.7 Окно администратора, вкладка «Склад».

Заключение

В ходе курсовой работы поставленные задачи были выполнены в полном объеме.

Спроектирована база данных для программного средства «Организация монтажных работ» в СУБД Microsoft SQL Server.

Созданы необходимые объекты: имена для входа, пользователи, роли, таблицы, представления, процедуры, индексы, триггеры.

Проведен импорт данных из JSON файлов в таблицы Tool, Accessory, а также экспорт данных в JSON файлы (из любой таблицы или представления).

Освоена и применена технология SQL Server Reporting Services (SSRS) для создания следующих отчетов:

- отчет в виде диаграммы с количеством заявок для бригады на текущую неделю;
- отчет в виде таблицы с количеством заявок для всех бригад (у которых есть незавершенные заявки) на текущую неделю;
- отчет с полной информацией о заявке (включающей данные клиента, адрес, комплектующие, оборудование, инструменты), использующий вложенные отчеты.

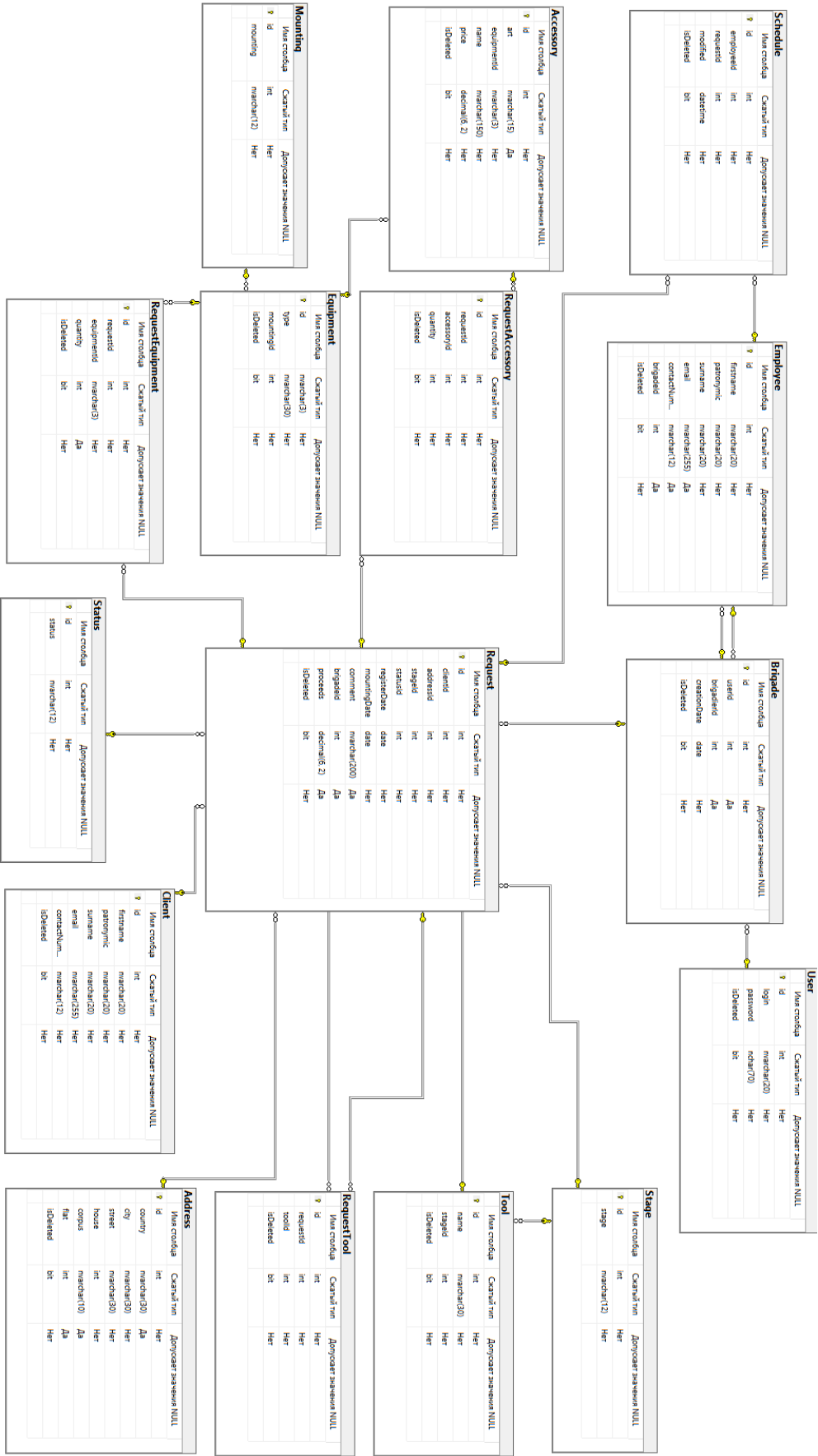
Для демонстрации работоспособности базы данных создано программное средство TheBureau (C#, с использованием технологий WPF, ADO.NET).

Список использованных источников

1. MSDN сеть разработчиков в Microsoft [Электронный ресурс] / Режим доступа: <https://docs.microsoft.com/en-us/sql/reporting-services/create-deploy-and-manage-mobile-and-paginated-reports?view=sql-server-ver15/>. Дата доступа: 13.12.2021
2. SQL Server technical documentation [Электронный ресурс] / Режим доступ: <https://docs.microsoft.com/en-us/sql/sql-server/?view=sql-server-ver15>. Дата доступа: 13.12.2021
3. Блинова, Е. А. Базы данных [Электронный ресурс]: учебно-методическое пособие к выполнению курсовой работы для студентов специальности 1-40 01 01 "Программное обеспечение информационных технологий" / Е. А. Блинова, Л. С. Мороз. - Минск : БГТУ, 2018. - 35, 36 с.
4. METANIT.COM Сайт о программировании [Электронный ресурс] / Режим доступа: <https://metanit.com>. Дата доступа: 14.12.2021
5. ProfessorWeb .NET & Web Programming [Электронный ресурс] / Режим доступа: <https://professorweb.ru> – Дата доступа: 15.12.2021.

Приложение А

Рисунок 1 Схема базы данных.



Приложение Б

Рисунок 2 Отчет по инструментам.

request Id	<input type="text" value="1"/>	Просмотр отчета
<div><div><div>1</div> из 1</div><div>100%</div><div><div>Най</div></div></div>		
<div><div>българка</div><div>пылесос</div><div>штроборез</div><div>перфоратор-отбойник</div><div>комплект ключей</div><div>гидравлический пресс</div><div>калибратор</div><div>клещи</div><div>ключ</div><div>щипцы</div></div>		

Приложение В

Рисунок 3 Отчет по комплектующим.

request Id <input type="text" value="1"/>		Просмотр отчета	
<div> ⏪ ⏴ 1 из 1 ⏵ ⏩ ↺ 100% ⏴ ⏩ 🖨 🔍 </div>			
Количество	Наименование	Цена	Артикул
1	Клапан радиаторный запорно-регулирующий Raditec проходной, резьба Rp1/2"x1/2", DN15, Kvs=1,36м3/ч, PN10, Tmax=95C, латунь, Италия	20,00	0382-02.000
	Клапан термостатический с преднастройкой Calypso Exact проходной, резьба Rp 1/2", DN15, Kvs=0,86м3/ч, PN10, Tmax=120C, AMETAL, Германия	35,00	3452-02.000
	Крюк двойной L=80мм (5шт)	1,00	
	Мешки (5шт)	5,00	
	Пресс-муфта 16x2-16x2, из латуни, Австрия (2шт)	30,00	P701600
			P701600
			P701600
	Пресс-муфта 20x2-20x2, из латуни, Австрия (2шт)		P702000
			P702000
			P702000

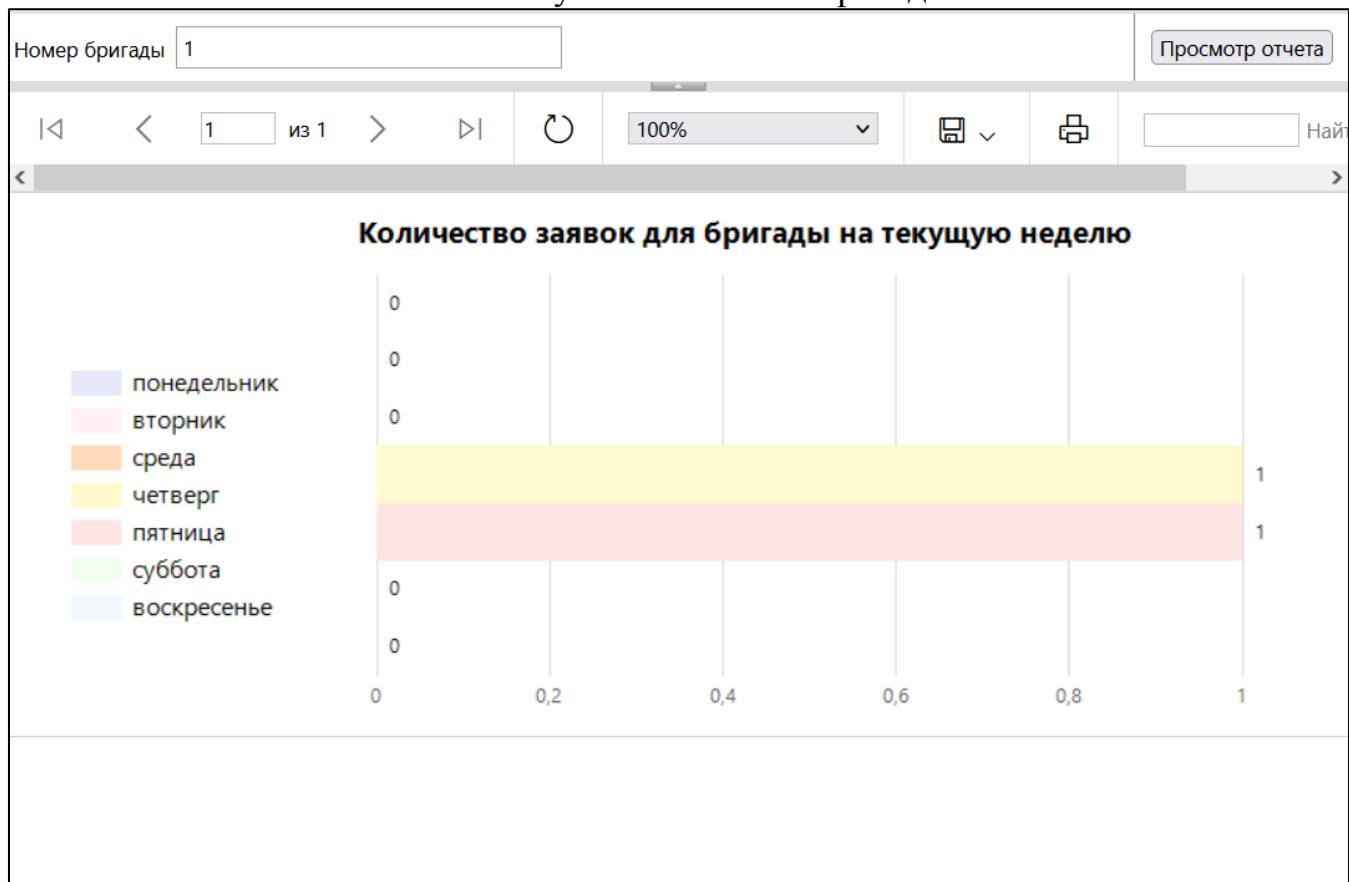
Приложение Г

Рисунок 4 Отчет по всем бригадам.

⏪	<	1	из 1	>	⏩	↺	100%	⏴	⏵	Най
бригада	понедельник	вторник	среда	четверг	пятница	суббота	воскресенье			
1	0	0	0	1	1	0	0			
2	0	1	0	0	0	0	0			
4	0	0	0	0	0	1	0			

Приложение Д

Рисунок 4 Отчет по бригаде.



Приложение Е

Рисунок 4 Отчет по заявке.

Номер заявки

1

Просмотр отчета

⏪

⏩

1

из 2 ?

⏴

⏵


↺

75%

💾

🖨

Заявка номер 1



Черновая и чистовая отделки

комментарий:

Частный дом.
Добрая собака.

зарегистрирована:

14.12.2021

монтаж назначен на:

15.12.2021

Адрес

Беларусь, г. Минск, ул. Черниговская

дом

5

корпус

1

квартира

1

Информация о клиенте

ФИО:

Зинович Злата Игоревна

E-mail:

zlatazinovich@gmail.com

Телефон:

375295634337

Монтируемое оборудование

Наименование	Тип	Способ монтажа	Количество
HP	напольный конвектор	Пол	2
HS	напольный конвектор	Стена	1
RP	радиатор	Пол	1
RS	радиатор	Стена	2
VP	внутрипольный конвектор	Пол	1

Комплектующие