ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ профессиональное

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

**Пермский химико-технологический техникум**

Специальность 09.02.07

«Информационные системы и программирование»

**КУРСОВОЙ ПРОЕКТ**

**Междисциплинарный курс:** «МДК.11.01 Технология разработки и защиты баз данных»

**Тема:** «Проектирование базы данных для салона по ремонту оргтехники, подсистема «Обслуживание клиента»

**Выполнил** студент гр.ИС-22-11

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (К.М.Гладких)

**Проверил(а)**:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_(Е.В.Юшкова)

*20\_\_*

Содержание

[**Введение** 3](#_Toc170237290)

[**Описательная часть** 5](#_Toc170237291)

[**1.** **Описание предметной области.** 5](#_Toc170237292)

[**2.** **Постановка задачи (функциональная диаграмма, диаграмма прецедентов)** 6](#_Toc170237293)

[**3.** **Выбор средств/методологии проектирования. Выбор СУБД** 8](#_Toc170237294)

[**4.** **Построение инфологической (концептуальной) модели предметной области** 10](#_Toc170237295)

[**5.** **Проектирование логической структуры БД: выявление полного перечня ограниченной целостности, присущего данной предметной области. Определение перечня ограничений целостности, которые будут контролироваться в данном КП. Выбор способа реализации контроля цеостности для каждого из ограничений** 11](#_Toc170237296)

[**Практическая часть** 14](#_Toc170237297)

[**6.** **Проектирование физической структуры базы данных** 14](#_Toc170237298)

[**7.** **Организация ввода-вывода данных в БД** 14](#_Toc170237299)

[**8.** **Организация манипулирования данными БД** 16](#_Toc170237300)

[**9.** **Реализация запросов, получение отчетов;** 19](#_Toc170237301)

[**10.** **Администрирование БД (использование механизма триггеров)** 20](#_Toc170237302)

[**11.** **Разработка интерфейса приложения** 21](#_Toc170237303)

[**Заключение** 41](#_Toc170237304)

[**Литература** 42](#_Toc170237305)

[**Приложение** 43](#_Toc170237306)

**ВВЕДЕНИЕ**

Современный мир высоких технологий и стремительно развивающегося бизнес-процесса навязывает свои требования каждому сферам активности. В частности, даже малые предприятия всё чаще прибегают к помощи программных продуктов для оптимизации своих продаж, упрощения ведения учета и сбора данных. И приложения с подключением к базам данных становятся одним из самых практичных и полезных вложений, сокращающие расходы времени, минимизирующие риск ошибок и обеспечивающие безопасное хранение информации.

Данный проект представляет собой проект, цель которого – разработать базу данных для салона по ремонту оргтехники «ProfitKomp», подсистема «Обслуживание клиента» и создать приложение для работы с этой базой данных. Для того чтобы упростить управление и оптимизацию данных на предприятии. Это включает в себя создание системы для работы с информацией, а также управление договорами и учёт материалов.

Основные задачи проекта включают в себя следующие пункты:

* Анализ предметной области;
* Разработка логической и концептуальной модели БД;
* Создание приложения для работы с БД;
* Организация системы для автоматической генерации чеков по окончанию выполненных работ.
* Разработка технической документации.

Таким образом, данный проект сфокусирован на предоставлении решений, способствующих повышению продуктивности работы предприятия, оптимизации бизнес-процессов и обеспечении качественного контроля информации.

# **Описательная часть**

**Предметная область: Салон по ремонту оргтехники «ProfitKomp», подсистема «Обслуживание клиента».**

## **Описание предметной области.**

Салон «ProfitKomp» занимается ремонтом оргтехники. Клиент обращается в салон к мастеру с сломанной оргтехникой, мастер осматривает устройство, предоставляет перечень услуг в котором указана их примерная стоимость. Мастер, с согласия клиента, вписывает в заказ необходимые услуги и материалы.

Мастер может приобрести материалы с различных ресурсов, которые отправятся на хранение, которые после будут использоваться в работе.

Клиент оформляет заказ: по необходимости указывает адрес, оставляет свои контактные данные, по которым мастер может связаться с ним после завершения работы.

После получения заказа мастер согласовывает с клиентом примерные сроки выполнения работ и окончательную стоимость. После окончания работы по ремонту мастер отправляет клиенту рабочую технику и помечает заказ как «Выполненный».

По запросу от клиента мастер может сформировать и распечатать чек, в котором указаны: Название предприятия, дата, оказанная услуга, сумма заказа, ФИО мастера.

Исходя из предметной области были выделены следующие сущности:

* Клиент, с атрибутами: адрес, контактные данные, ФИО.
* Мастер, с атрибутами: ФИО.
* Услуги, с атрибутами: название, примерная стоимость.
* Материал, с атрибутами: Название
* Заказ, с атрибутами: ФИО мастера, ФИО клиента, «выполненность», итоговая стоимость, начало и конец работы, номер плана работы, материал.
* План работы, с атрибутами: Название услуги, материал, количество материала, стоимость работы.
* Поступление материала, с атрибутами: Материал, количество
* Чек, с атрибутами: название предприятия, дата, оказанная услуга, сумма заказа, ФИО мастера, ФИО клиента

## **Постановка задачи (функциональная диаграмма, диаграмма прецедентов)**

**Требования к функциональным характеристикам:**

В программном продукте должны быть реализованы:

* Просмотр услуг, создание заказа для клиента.
* Просмотр, добавление и изменение заявок, добавление, редактирование, удаление и просмотр услуг, создание чека для мастера.
* Добавление и списание материалов
* Поиск заявок по параметру «Выполненный» и «Не выполненный»
* Удобный и понятный интерфейс

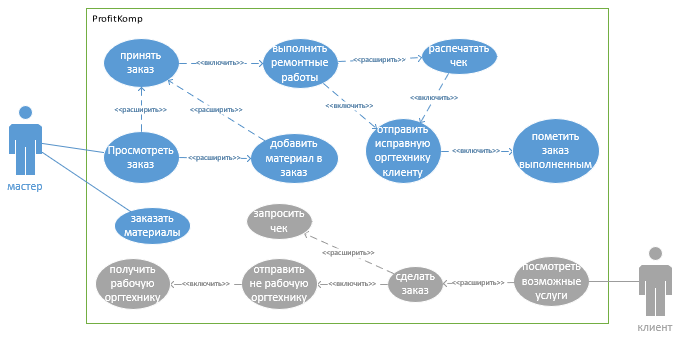


Рисунок 1. Диаграмма прецедентов

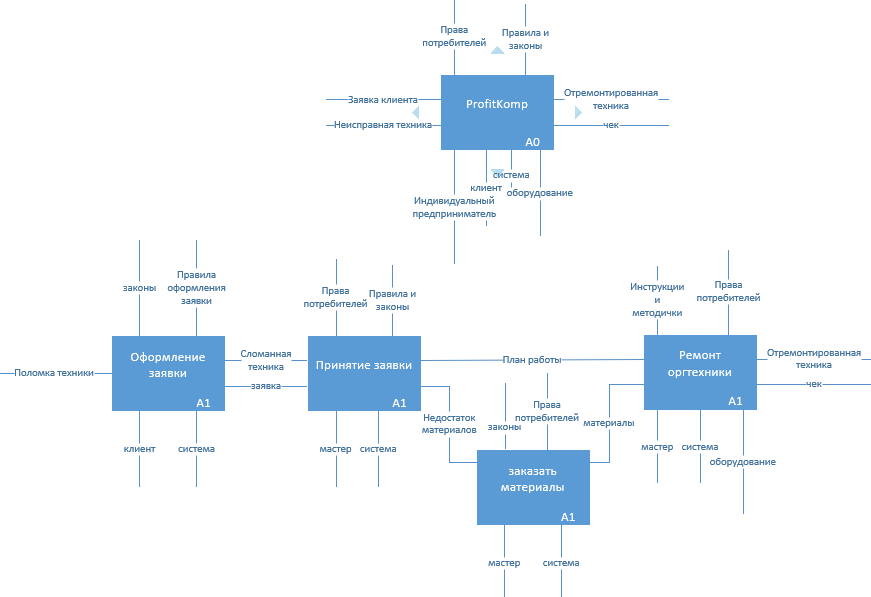


Рисунок 2. Функциональная диаграмма

## **Выбор средств/методологии проектирования. Выбор СУБД**

Два наиболее распространенных метода, используемых для проектирования базы данных, включают в себя:

**Нормализация:** Таблицы организованы таким образом, чтобы уменьшить избыточность и зависимость данных. Таблицы большего размера делятся на таблицы меньшего размера и связываются друг с другом с помощью отношений.

**Моделирование сущностей-связей (ER):**Подход к графическому проектированию базы данных моделирует атрибуты сущностей и определяет отношения между ними для обозначения реальных объектов. Сущность — это любой предмет реального мира, отличающийся или уникальный от окружающей среды.

Воспользовавшись технологией моделирования сущностей-связей, можно наглядно показать зависимости атрибутов сущностей. Также данная технология не требует больших затрат, поэтому я воспользуюсь именно ей.

Система управления базами данных (СУБД) – это комплекс программно-языковых средств, позволяющих создать базы данных и управлять данными.

**Наиболее популярные СУБД:**

Microsoft SQL Server- фирменная СУБД, разработанная Microsoft. Система позволяет синхронизироваться с другими программными продуктами компании Microsoft, а также обеспечивает надежную защиту данных и простой интерфейс, однако отличается высокой стоимостью лицензии и повышенным потреблением ресурсов.

Файловая СУБД- Файловая СУБД разработана фирмой «1С» и является частью платформы. Файловая СУБД хранит все данные в одном файле — файловой базе данных. Этот формат хранения данных разработан фирмой «1С» специально для прикладных решений 1С:Предприятия 8

PhpMyAdmin- приложение написанное на PHP и обеспечивающее полноценную, работу с базами данных MySQL через браузер. Так как phpMyAdmin позволяет во многих случаях обойтись без непосредственного ввода команд SQL, то работа с базами данных становится вполне посильной задачей даже для человека весьма поверхностно знакомого с MySQL.

Таблица 1. Сравнительная таблица СУБД

|  |  |  |  |
| --- | --- | --- | --- |
|  | Microsoft SQL Server | Файловая СУБД | phpMyAdmin |
| Цена | Платно | Бесплатно | Бесплатно |
| Сложность настройки | Высокая | Низкая | Средняя |
| Скорость чтения | Высокая | Средняя | Средняя |
| Скорость записи | Высокая | Высокая | Средняя |
| Требуется сервер | Да | Нет | Да |
| Поддержка параллельной работы | Да | Да | Да |
| Возможность резервного копирования | Да | Да | Да |
| Безопасность | Высокая | Средняя | Высокая |
| Простота обучения | Сложное | Простое | Средне |

Файловая СУБД- разработана фирмой «1С» и является частью платформы, поэтому она удобна в работе, не нужно ставить сторонний софт и проводить какие-либо сложные манипуляции, низкие вложения средств и мало времени на установку. Также большой плюс этого варианта, то, что подключаться к базе и работать с ней можно из любой точки земного шара.

Создание проекта проходит с использованием встроенного языка программирования программного обеспечения «1C:Предприятие», оптимизированный для работы на платформе.

## **Построение инфологической (концептуальной) модели предметной области**

Нотации – графические модели, которые используются, чтобы фиксировать бизнес-процессы, анализировать их и оптимизировать.

*Нотация Чена*- **очень трудоемка**, читать диаграмму очень утомительно, ведь на ней очень много элементов, однако она неплохо отражает общую идею диаграммы.

*Нотация диаграммы классов UML*-Диаграмма классов активно используется при объектно-ориентированном проектировании и для ее построения существует множество инструментов. читать диаграмму легко.

*Нотация IDEF1X*- Схожа с нотацей диаграммы классов UML, но в стандарте IDEF1X используются символы и типы связей, не поддерживаемые многими инструментальными средствами.

Использовать нотацию диаграммы классов UML будет целесообразнее всего.

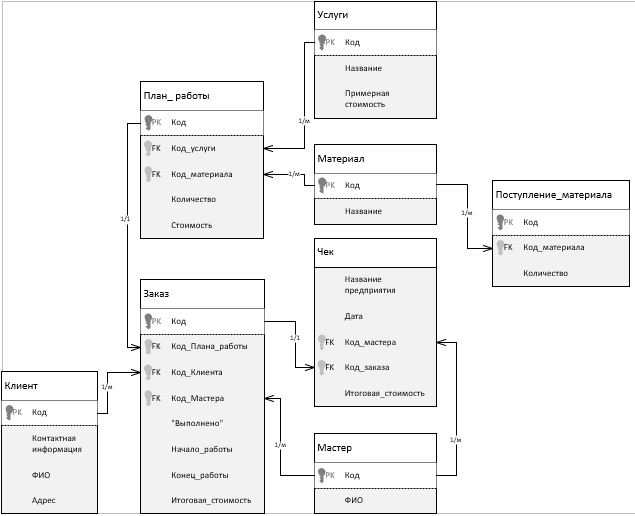


Рисунок 3. Концептуальная модель

## **Проектирование логической структуры БД: выявление полного перечня ограниченной целостности, присущего данной предметной области. Определение перечня ограничений целостности, которые будут контролироваться в данном КП. Выбор способа реализации контроля целостности для каждого из ограничений**

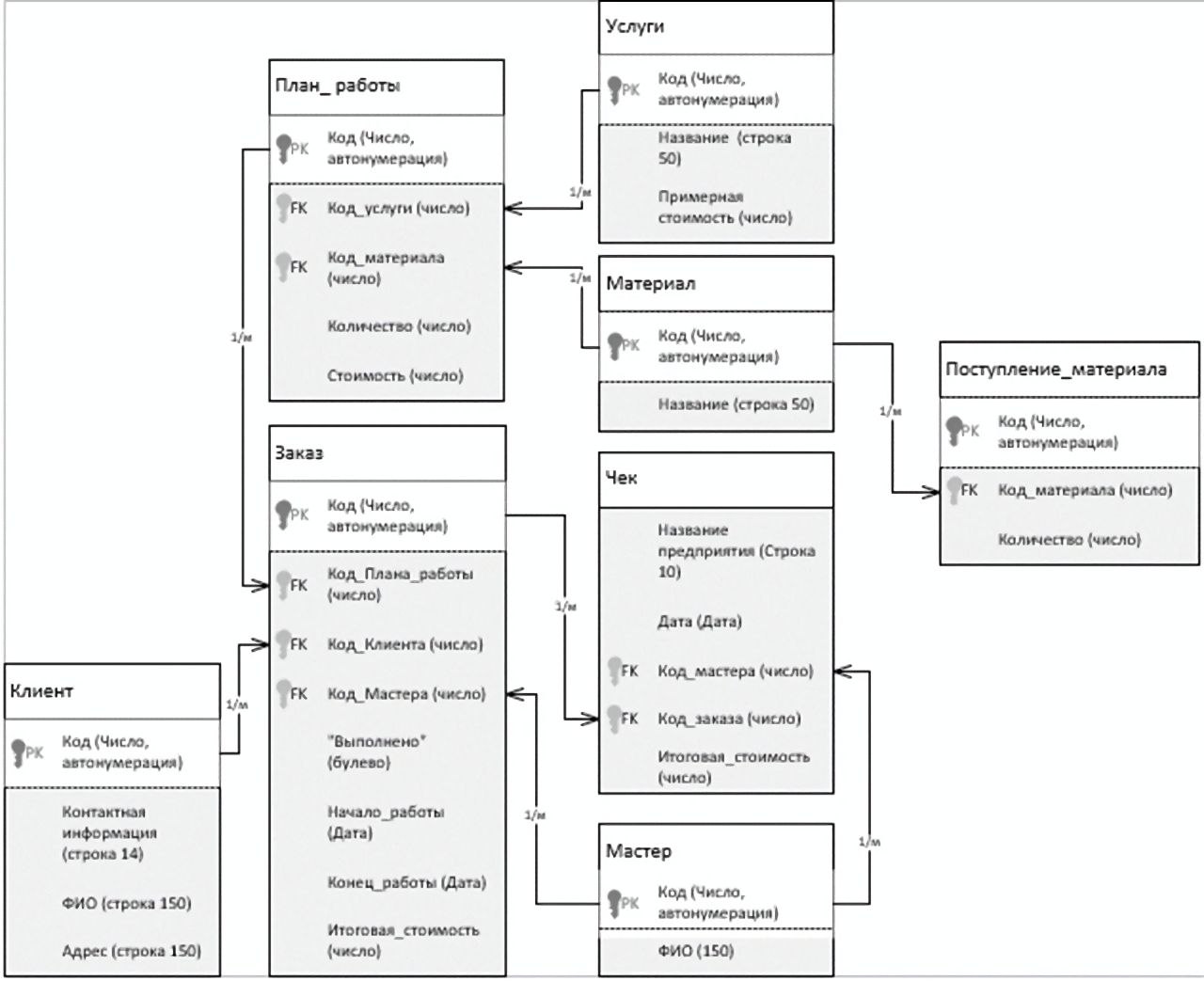


Рисунок 4. Логическая структура БД

Контроль целостности будет включать в себя следующие проверки:

Уникальность ключевых идентификаторов: от каждого клиента, мастера, сервиса, материала и заказа ожидается уникальный идентификатор, чтобы предотвратить дублирование и обеспечить целостность данных.

Проверка зависимостей: необходимо обеспечить, что каждый заказ связан только с одним клиентом и одним мастером. Также, каждый заказ должен содержать только валидные сервисы и материалы.

Проверка бизнес-правил: нужно гарантировать, что все бизнес-правила соблюдаются. Например, заказ не может быть создан без клиента, и чек не может быть выдан без соответствующего заказа.

Проверка дат: Необходимо обеспечить, что дата начала работы всегда предшествует дате окончания работы.

Все эти ограничения целостности можно реализовать, используя внутренние механизмы СУБД и программный код. Например, уникальность идентификаторов можно обеспечить, используя уникальные ключи и ограничения базы данных. Зависимости между атрибутами можно поддерживать, используя внешние ключи. Правила бизнес-логики и последовательность дат можно реализовать через триггеры и процедуры СУБД.

# **ПРАКТИЧЕСКАЯ ЧАСТЬ**

## **Проектирование физической структуры базы данных**

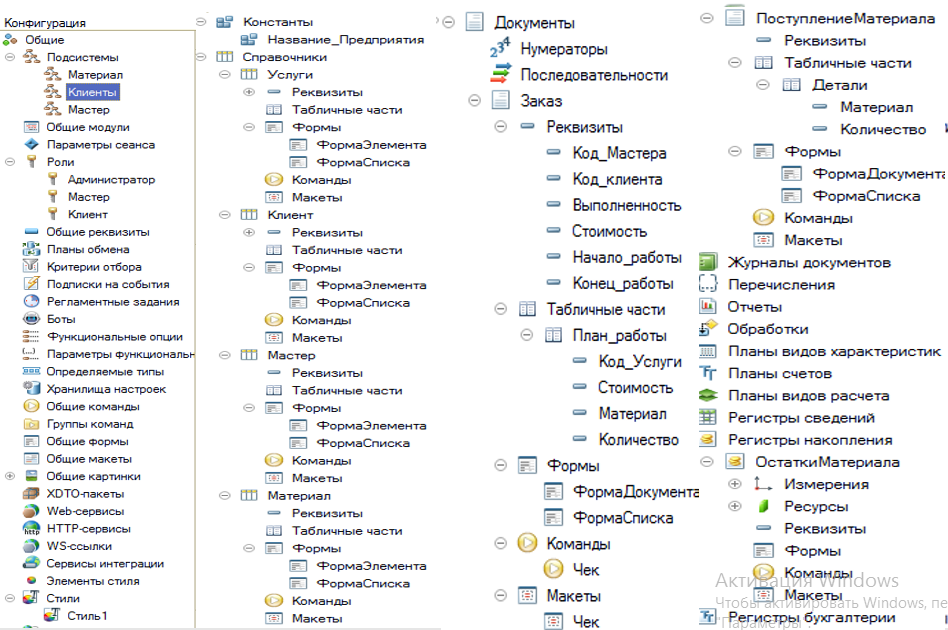


Рисунок 5. Физическая структура БД

## **Организация ввода-вывода данных в БД**

Платформа 1C облегчает процесс организации ввода-вывода данных, благодаря встроенной файловой СУБД. Она автоматизирует многие процессы, связанные с управлением данными, что способствует более эффективной и экономной работе пользователей.

Для ввода данных в базу данных используются формы элемента и формы документа, которые отображают заполняемые поля – реквизиты, а также реквизиты табличных частей

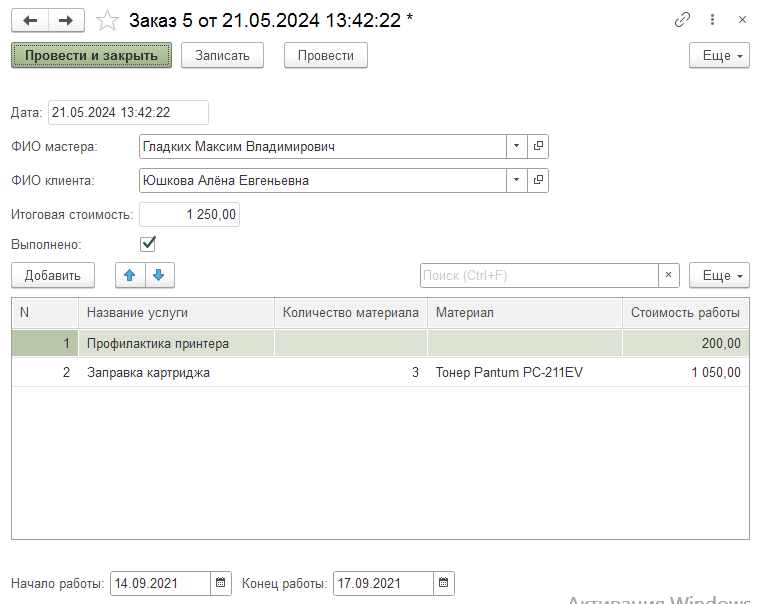


Рисунок 6. Внешний вид формы элемента справочника «Клиент» с заполненными полями – реквизитами

Когда вносятся данные в базу, при нажатии кнопки «записать и закрыть», выводится все записи в базе данных в окне формы списка. Если дважды щёлкнуть по одной из записей, откроется окно с формой элемента справочника, где уже автоматически будут заполнены все необходимые поля из базы данных.

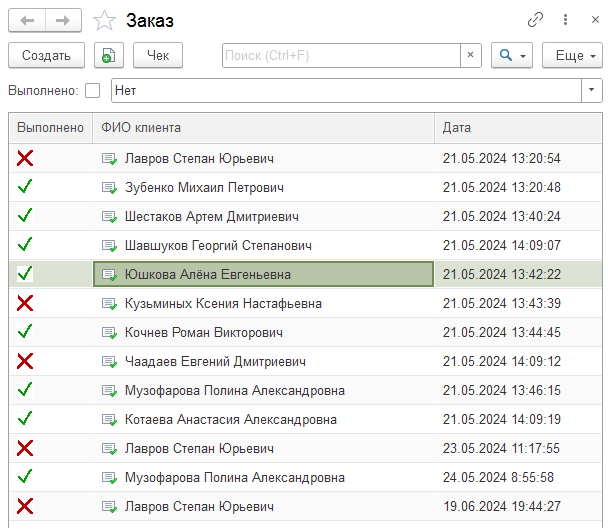


Рисунок 7. Внешний вид формы списка справочника «Клиент»

## **Организация манипулирования данными БД**

Платформа 1с предоставляет автоматическую работу с манипулированием данными из БД.

Для того чтобы изменить существующую запись, необходимо:

1. Найти ее в окне формы списка.
2. Навестись на нее мышкой и открыть двойным щелчком.

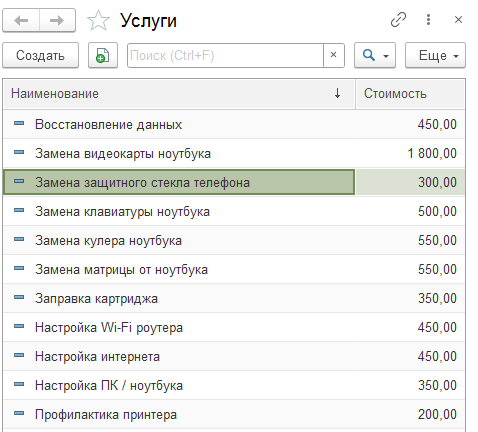


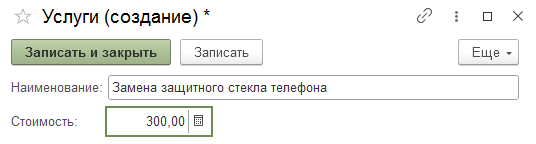
Рисунок 8. Внешний вид формы списка справочника «Услуги», с выделенной записью  


Рисунок 9. Внешний вид формы элемента справочника «Услуги» с заполненными полями до изменений

После открытия нужной записи автоматически будут заполнены все необходимые поля из базы данных. После изменения нужных полей необходимо нажать кнопку «Записать» или «Записать и закрыть» для сохранения изменённых данных.

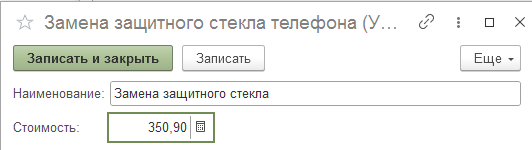


Рисунок 10. Внешний вид формы элемента справочника «Услуги» с заполненными полями после изменений

После успешного сохранения изменений, если открыть форму списка, то в нем будет отображаться измененная запись

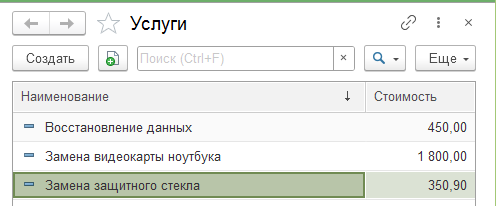


Рисунок 11. Внешний вид формы списка справочника «Услуги», с выделенной измененной записью.

Для удаления записи из базы данных необходимо:

1. Найти ее в окне формы списка.
2. Навестись на нее мышкой и выделить щелчком.
3. В командной панели нажать на кнопку «Еще»
4. Нажать на пункт из выпадающего списка «Удалить»
5. На всплывающем окне с текстом «Удалить “Название выбранной записи”» нажать кнопку «Да»

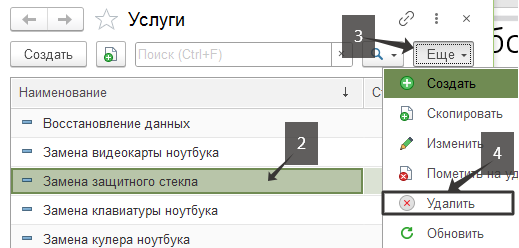


Рисунок 12. Внешний вид формы списка справочника «Услуги», с выделенной записью и отмеченными пунктами действий

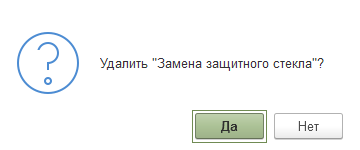


Рисунок 13. Всплывающее окно, открывающееся при удалении записи

## **Реализация запросов, получение отчетов;**

Для реализации печати чека по выполненному заказу из документа «Заказ» можно воспользоваться встроенным конструктором печати, после выбора необходимых для вывода данных нужно: добавить на открывшемся окне макета не хватающие данные, по необходимости назначить имя для отдельных строк, настроить в модуле менеджера код так, чтобы автоматически считалась и выводилась итоговая стоимость услуг, и из константы выводилось название предприятия.

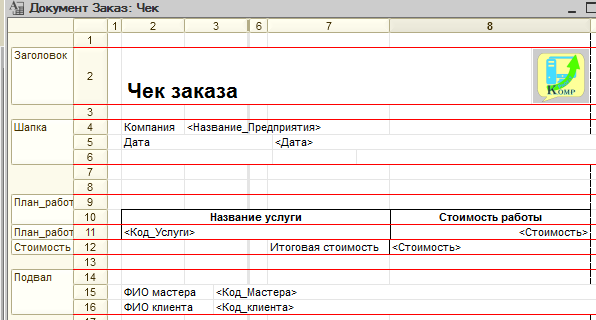


Рисунок 14. Макет конструктора печати

## **Администрирование БД (использование механизма триггеров)**

В платформе 1с предусмотрена автоматическое администрирование БД, для регулирования возможности администрирования необходимо добавить роли с необходимыми правами и добавить пользователей. Система ролей и доступа позволяет настраивать права доступа на уровне каждого отдельного пользователя, а регистрация событий и обработчиков событий помогает контролировать исполнение этих прав.

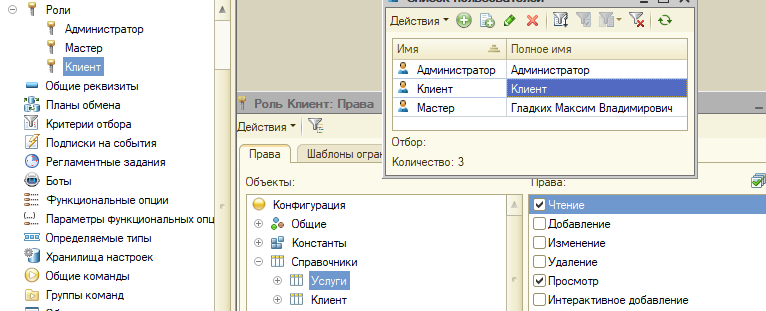


Рисунок 15. Пользователи с настроенными ролями

В программе 1С:Предприятие нет интегрированной среды для создания триггеров, поэтому администрирование базы данных заключается в разделении прав у ролей пользователя.

## **Разработка интерфейса приложения**

Чтобы начать пользоваться приложением, необходимо пройти процедуру авторизации, выбрав необходимого пользователя из выпадающего списка

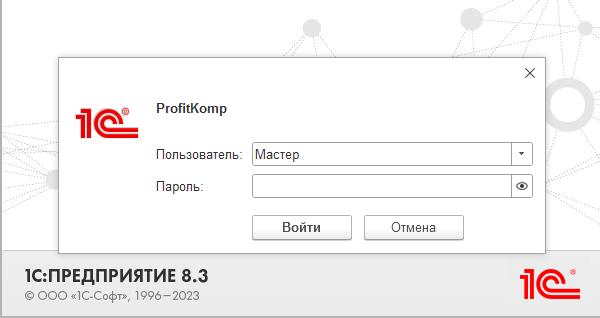


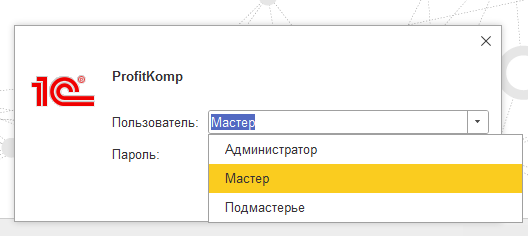
Рисунок 16. Внешний вид начального окна авторизации с выбранным пользователем  


Рисунок 17. Внешний вид начального окна авторизации показывающее выпадающий списком с пользователями

После успешного входа, оператору открывается окно, в котором можно увидеть в правом верхнем углу – полное имя вошедшего в систему пользователя, а ниже подписанные ярлыки подсистем

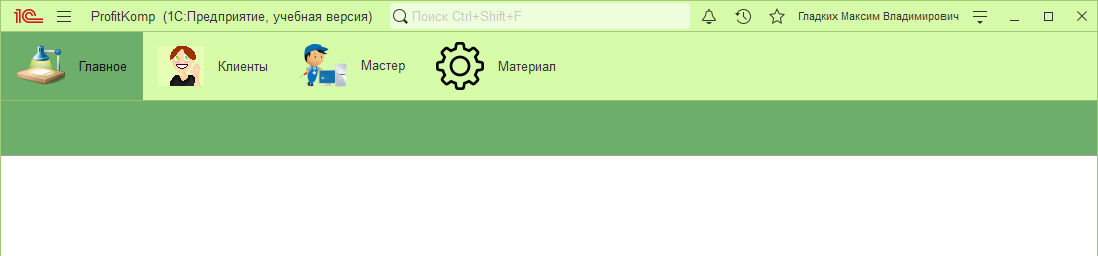


Рисунок 18. Внешний вид начального окна с подсистемами

Для работы необходимы мастера, чтобы добавить мастера необходимо:

1. Открыть вкладку «Мастер»
2. В открывшейся вкладке открыть справочник мастеров, нажав на надпись «Мастер»
3. Нажать на кнопку «Создать»

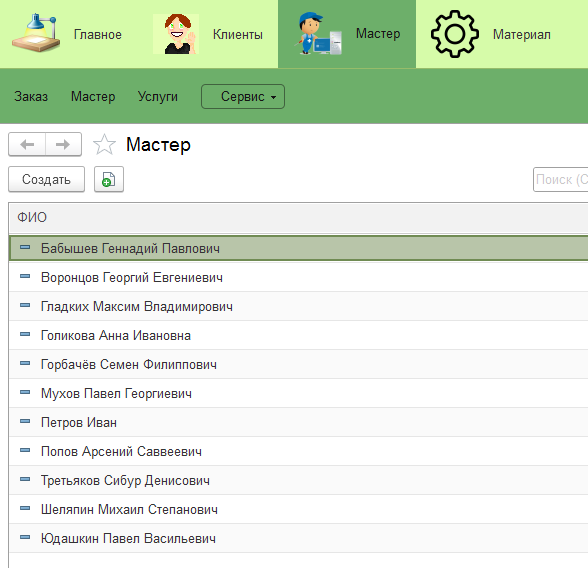


Рисунок 19. Внешний вид формы списка справочника «Мастер»

После этого откроется окно создания нового мастера, в котором будет поле ввода фамилии, имени и отчества (при наличии), это поле обязательно к заполнению

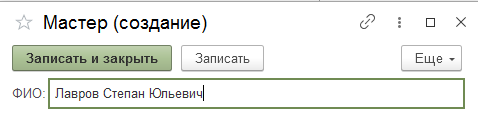


Рисунок 20. Внешний вид формы элемента справочника «Мастер» с заполненным полем ФИО

После того как оператор ввел ФИО необходимо нажать кнопку «Записать и закрыть», после этого действия, если данные были заполнены правильно, информация запишется в базу данных, а новую запись можно будет найти в форме списка справочника «Мастер»

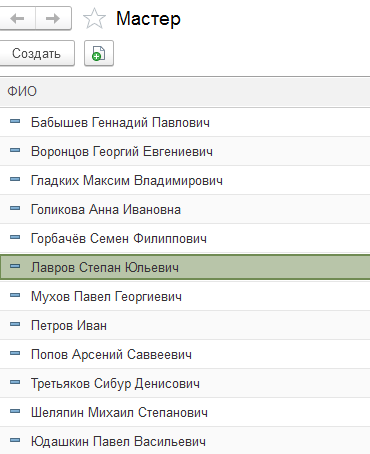


Рисунок 21. Внешний вид формы списка справочника «Мастер» после создания новой записи

Находясь на этой-же вкладке оператор может добавить название предприятия, нажав на кнопку «Сервис» и из выпавшего списка выбрать «Название предприятия»

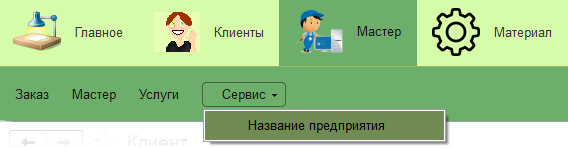


Рисунок 22. Кнопка сервис на панели ячейки «Мастер» с выпадающим списком

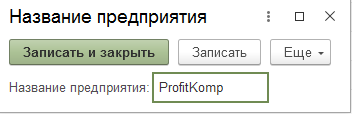
После этого откроется окно, в которое можно вписать название предприятия  


Рисунок 23. Внешний вид формы константы с заполненным полем

После добавления мастера и названия предприятия необходимо добавить услуги для этого на той-же вкладке мастера откроем справочник услуг, нажав на надпись «Услуги» и нажмем кнопку «Создать»

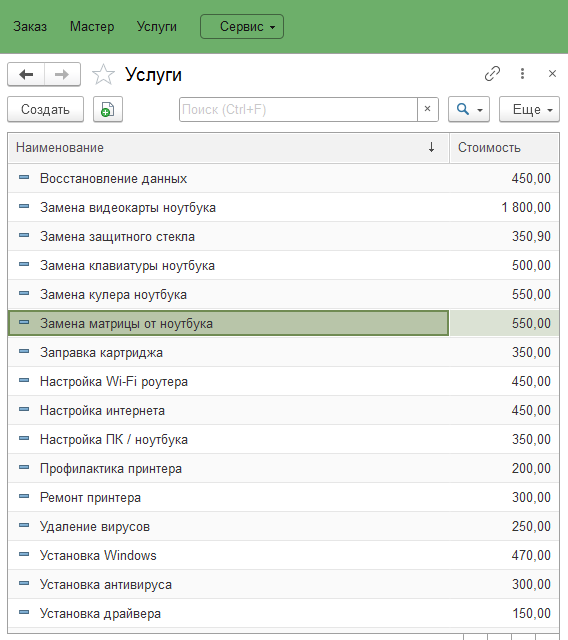


Рисунок 24. Внешний вид формы списка справочника «Услуги»

После этого откроется окно создания новой услуги, в котором будет поля ввода Наименования и стоимости, оба этих поля обязательны к заполнению

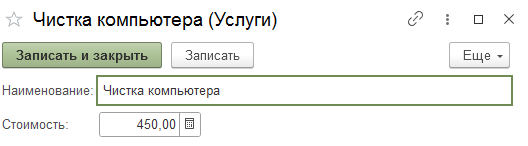


Рисунок 25. Внешний вид формы элемента справочника «Мастер» с заполненными полями

После того как оператор заполнил оба поля необходимо нажать кнопку «Записать и закрыть», после этого действия, если данные были заполнены правильно, информация запишется в базу данных, а новую запись можно будет найти в форме списка справочника «Услуги»

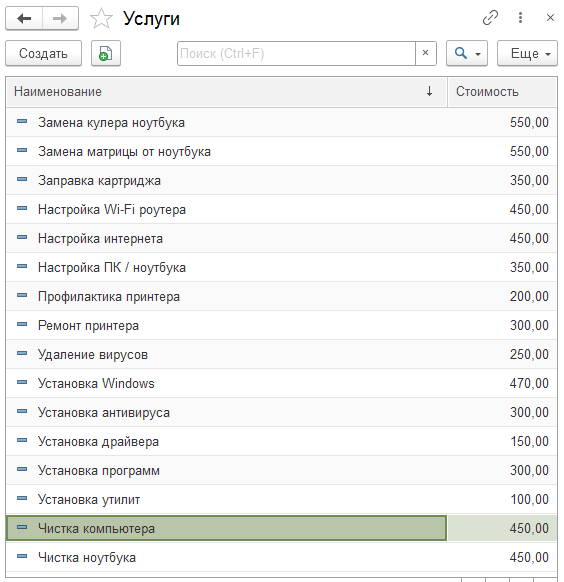


Рисунок 26. Внешний вид формы списка справочника «Услуги» после создания новой записи

После добавления услуги необходимо добавить материалы для этого переходим на вкладку «Материал» и откроем справочник материалов, нажав на надпись «Материал» и нажмем кнопку «Создать»

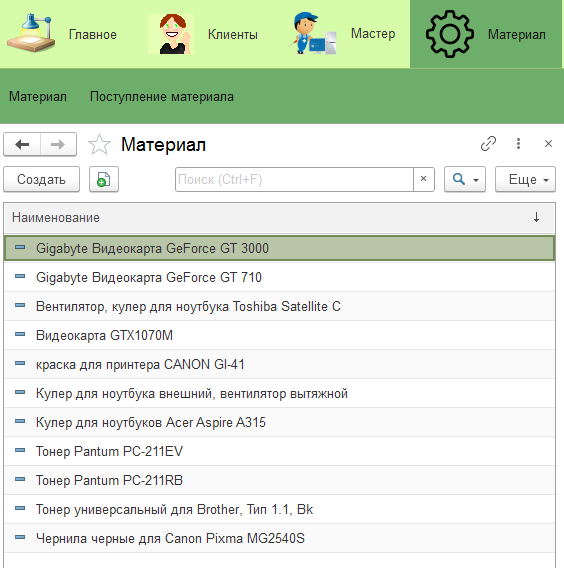


Рисунок 27. Внешний вид формы списка справочника «Материал»

После этого откроется окно создания нового материала, в котором будет поле ввода «Наименование», это поле обязательно к заполнению

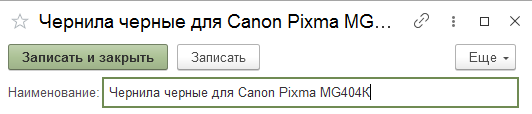


Рисунок 28. Внешний вид формы элемента справочника «Материал» с заполненным полем

После того как оператор заполнил поле необходимо нажать кнопку «Записать и закрыть», после этого действия, если данные были заполнены правильно, информация запишется в базу данных, а новую запись можно будет найти в форме списка справочника «Материал»

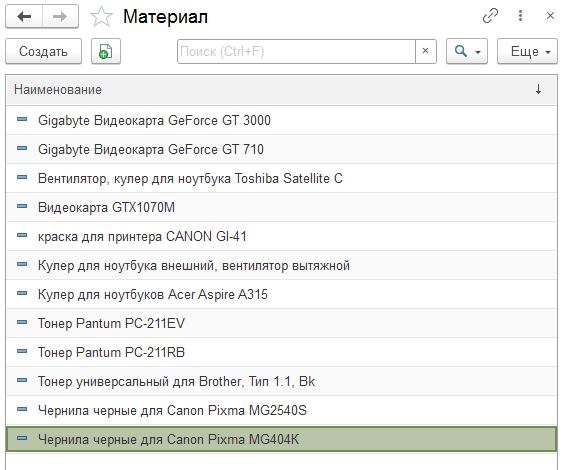
****

Рисунок 29. Внешний вид формы списка справочника «Материал» после создания новой записи

После добавления материала необходимо добавить материал на склад для этого на той-же вкладке откроем документ поступление материалов, нажав на надпись «Поступление материала» и нажмем кнопку «Создать»

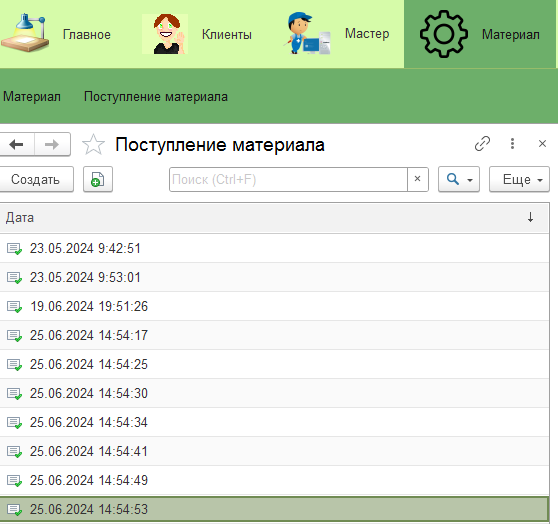


Рисунок 30. Внешний вид формы списка документа «Поступление материала»

После этого откроется окно поступления материала, в котором будет табличная часть обязательная к заполнению, в ней поля «Материал» и «Количество», поле материала заполняется путем выбора элемента из выпадающего списка

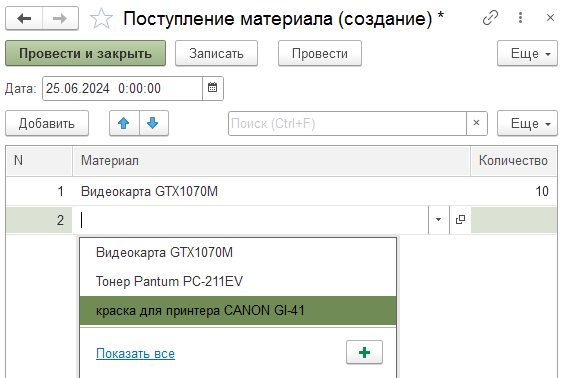


Рисунок 31. Внешний вид формы документа «Материал» с заполненной строкой табличной части

После того как оператор заполнил необходимое количество строк табличной части необходимо нажать кнопку «Записать и закрыть», после этого действия, если данные были заполнены правильно, информация запишется в базу данных, а новую запись можно будет найти в форме списка справочника «Материал» в виде даты проведения документа

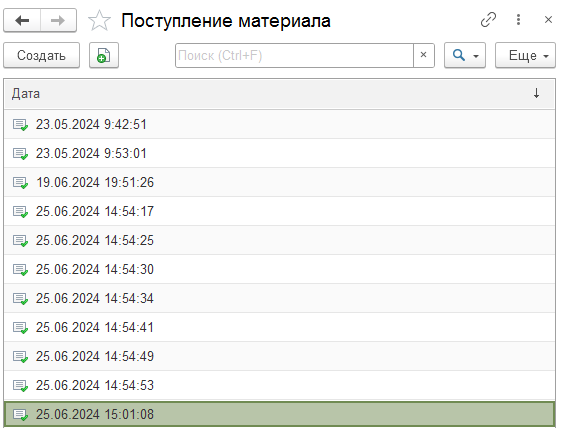
****

Рисунок 32. Внешний вид формы списка справочника «Материал» после создания новой записи

После добавления на склад материала необходимо добавить клиентов в базу данных для этого переходим на вкладку «Клиенты» и откроем справочник клиентов, нажав на надпись «Клиент» и нажмем кнопку «Создать»

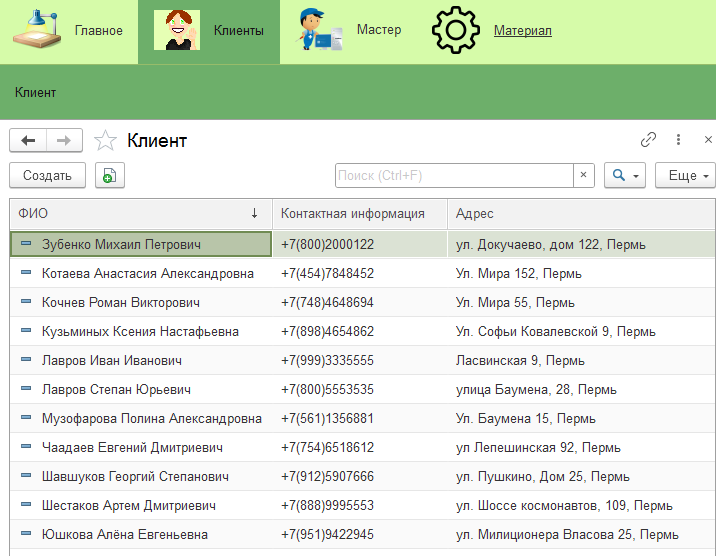


Рисунок 33. Внешний вид формы списка справочника «Клиент»

После этого откроется окно создания нового клиента, в котором будут поля ввода «ФИО», «Контактная информация» и «Адрес», поля «ФИО» и «Контактная информация» обязательны к заполнению

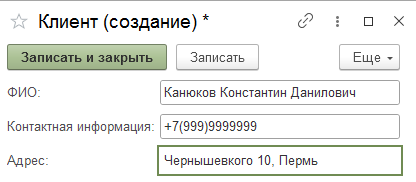


Рисунок 34. Внешний вид формы элемента справочника «Клиент» с заполненными полями

После того как оператор заполнил поля необходимо нажать кнопку «Записать и закрыть», после этого действия, если данные были заполнены правильно, информация запишется в базу данных, а новую запись можно будет найти в форме списка справочника «Клиент»

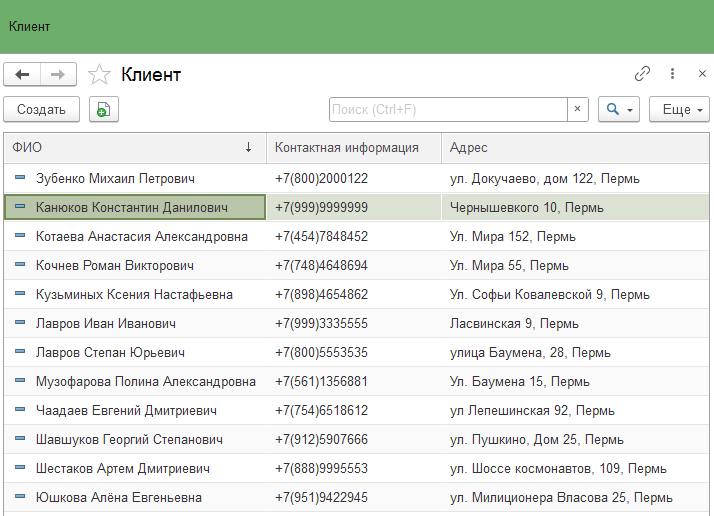
****

Рисунок 35. Внешний вид формы списка справочника «Материал» после создания новой записи

После добавления на клиента необходимо добавить клиентов в базу данных для этого переходим на вкладку «Клиенты» и откроем справочник клиентов, нажав на надпись «Клиент» и нажмем кнопку «Создать»

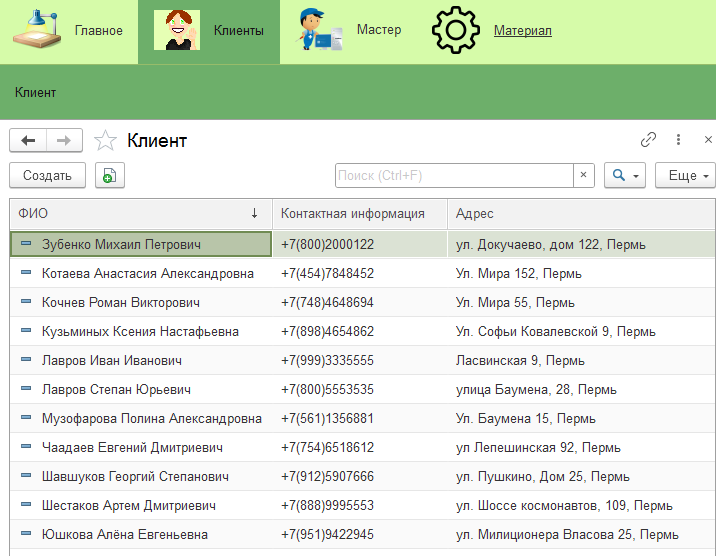


Рисунок 36. Внешний вид формы списка справочника «Клиент»

После этого откроется окно создания нового клиента, в котором будут поля ввода «ФИО», «Контактная информация» и «Адрес», поля «ФИО» и «Контактная информация» обязательны к заполнению

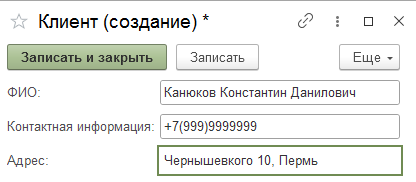


Рисунок 37. Внешний вид формы элемента справочника «Клиент» с заполненными полями

После того как оператор заполнил поля необходимо нажать кнопку «Записать и закрыть», после этого действия, если данные были заполнены правильно, информация запишется в базу данных, а новую запись можно будет найти в форме списка справочника «Клиент»

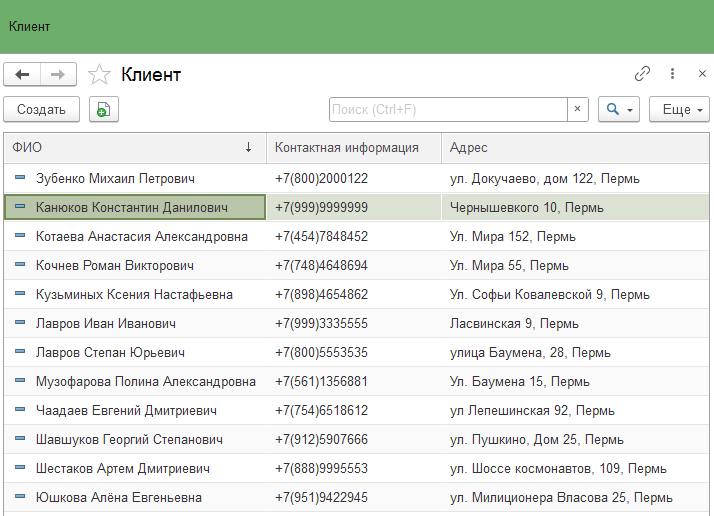
****

Рисунок 38. Внешний вид формы списка справочника «Клиент» после создания новой записи

После всех прошлых действий можно добавить заказ для этого переходим на вкладку «Мастер» и откроем справочник заказов, нажав на надпись «Заказ» и нажмем кнопку «Создать»

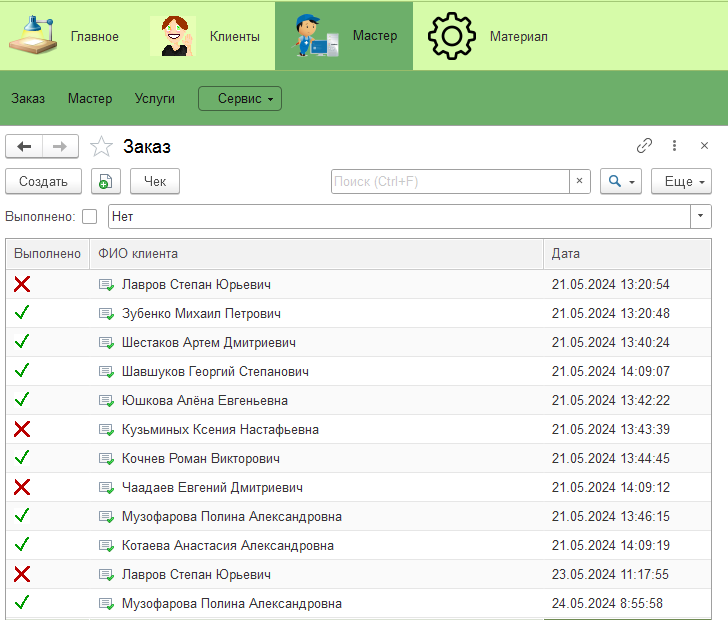
****

Рисунок 39. Внешний вид формы списка документа «Заказ»

После этого откроется окно создания нового заказа, в котором будут поля ввода «ФИО мастера», «ФИО клиента» и «Итоговая стоимость», пункт «Выполнено», «Начало работы» и «Конец работы», также здесь есть табличная часть с полями «Название услуги», «Количество материала», «Материал», «Стоимость работы», поля «ФИО клиента» и табличная часть обязательны к заполнению, «Итоговая стоимость» не доступна для изменения. Стоимость работы вводится автоматически после выбора «Название услуги».

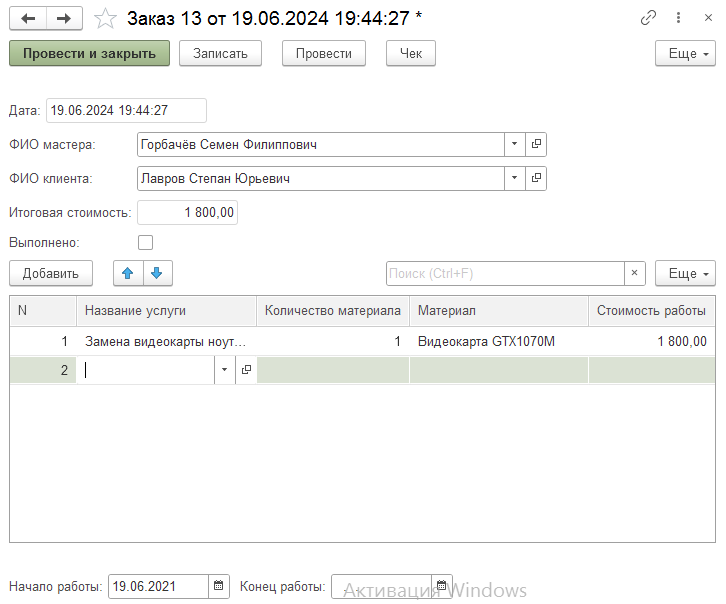


Рисунок 40. Внешний вид формы документа «Заказ» с заполненными полями и строкой табличной части

После того как оператор заполнил поля необходимо нажать кнопку «Записать и закрыть», после этого действия, если данные были заполнены правильно, информация запишется в базу данных, а новую запись можно будет найти в форме списка справочника «Заказ»

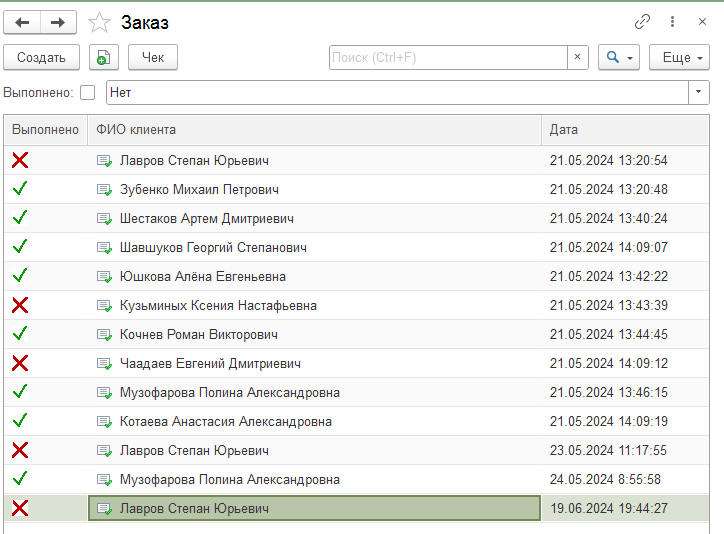
****

Рисунок 41. Внешний вид формы списка документа «Заказ» после создания новой записи

Если в документе заполнить поле «Конец работы», то пункт «Выполнено» будет отмечен автоматически. Также при изменении количества заполненных сток в табличной части будет пересчитываться сумма Итоговой стоимости

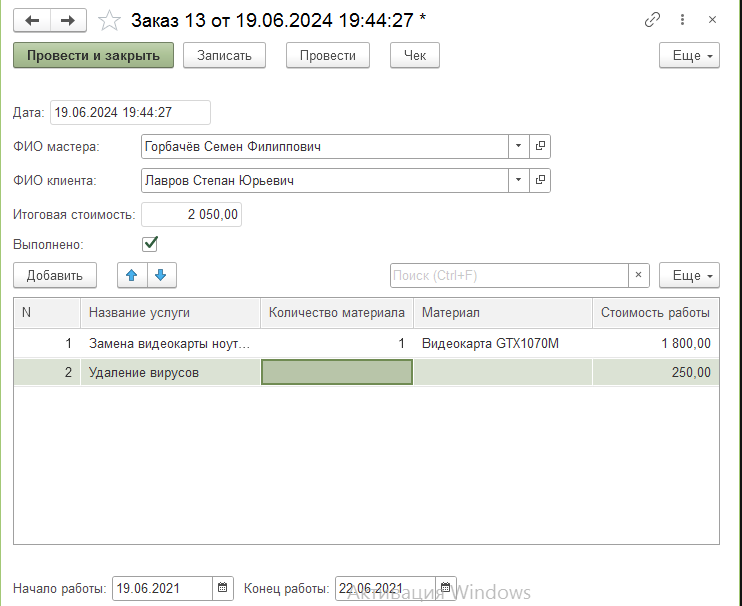
****

Рисунок 42. Внешний вид формы документа «Заказ» с заполненным полем «Конец работы» и добавленной строкой табличной части

При проведении и закрытии документа, для сохранения изменений, в форме списка справочника «Клиент» картинка крестика сменится картинкой галочки, означая что данный заказ выполнен, также на этой форме можно сделать сортировку по выполненным и не выполненным заказам

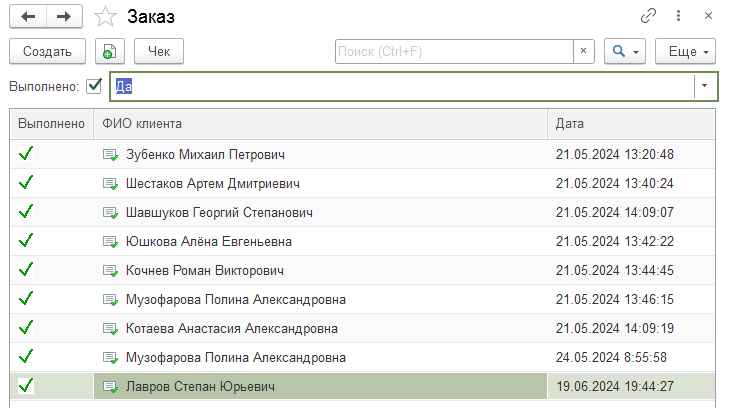


Рисунок 43. Внешний вид формы списка документа «Заказ» после изменений документа с выполняемой сортировкой

Для получения чека необходимо нажать на кнопку «Чек» в любой из двух форм документа «Заказ», после чего автоматически сгенерируется чек готовый к печати.



Рисунок 44. Внешний вид печатаемого чека

Ссылка на Gogs с загруженной выгрузкой проекта: http://10.0.10.250:3000/kefirona/ProfitKomp.git

# **ЗАКЛЮЧЕНИЕ**

В рамках данной работы было проведено глубокое исследование в области проектирования баз данных. Начиная с выбора подходящих средств и методологии проектирования, получивших основу в моделировании сущностей-связей (ER). Это позволило наглядно представить зависимости атрибутов сущностей без значительных материальных затрат.

Был обоснован выбор системы управления базами данных (СУБД) - файловой СУБД от "1С" - на основе сравнительного анализа. Эта СУБД выигрывает в удобстве использования и согласуется с платформой проекта.

Построение инфологической (концептуальной) модели предметной области позволило визуализировать и структурировать данные с использованием нотации диаграммы классов UML.

Заключительный этап включал проектирование логической структуры базы данных с целью выявления и управления ограничениями целостности.

Вся проект был проделан с применением теоретических знаний и практических навыков в области проектирования баз данных. Все поставленные задачи были выполнены. Полученные результаты положительно свидетельствуют о готовности к реализации более сложных проектов.

Однако стоит отметить, что дальнейшее совершенствование навыков в этой области всегда необходимо, учитывая быстрый темп развития технологий. Благодаря непрерывному изучению новых методов и подходов можно обеспечить наиболее эффективное использование инструментов баз данных и достижение впечатляющих результатов в будущих проектах.

Разработанный пример базы данных можно использовать как прототип или отправную точку для создания похожих проектов в области технического обслуживания и ремонта или использоваться в качестве учебного материала для обучения в области проектирования БД, а также в демонстрационных целях.

# **Литература**

1. Заполнение "пустых" строк табличной части - <https://fastcode.im/Templates/7079/zapolnenie-pustyx-strok-tablichnoj-chasti>
2. Определение логической структуры базы данных - <https://studfile.net/preview/6144887/page:4/>
3. Проверка на пустой результат выполнения запроса - <https://its.1c.ru/db/v8std/content/438/hdoc>
4. Файловая база 1С или SQL – что лучше? - https://programmist1s.ru/1s-faylovaya-ili-sql/
5. Функциональные диаграммы - <https://multiurok.ru/files/funktsionalnye-diagrammy.html>

# **ПРИЛОЖЕНИЕ**

*Код формы документа «Заказ»:*

&НаКлиенте

Процедура УслугиПриИзменении(Элемент)

Объект.Стоимость=Объект.План\_работы.Итог("Стоимость"); //подсчет итоговой стоимости

КонецПроцедуры

&НаКлиенте

Процедура Конец\_работыПриИзменении(Элемент)

Если Объект.Начало\_работы<=Объект.Конец\_работы тогда

Объект.Выполненность=Истина;

Иначе

Сообщить("Вы ошиблись датой");

Объект.Выполненность=Ложь;

Возврат;

КонецЕсли;

КонецПроцедуры

&НаСервере

Функция ПолучЦена(Ссылка)

Возврат Ссылка.Стоимость;

конецфункции

&НаКлиенте

Процедура УслугиМатериалПриИзменении(Элемент)

СтрокаТЧ= Элементы.Услуги.ТекущиеДанные;

Если ПроверьСклад(СтрокаТЧ.Материал)<>0 тогда

Если ПроверьСклад(СтрокаТЧ.Материал)<СтрокаТЧ.Количество тогда

х=СтрокаТЧ.Количество-ПроверьСклад(СтрокаТЧ.Материал);

Сообщить("На складе не хватает "+х);

Возврат;

Иначе

СтрокаТЧ= Элементы.Услуги.ТекущиеДанные; //текущие данные строки табличной части

Если ЗначениеЗаполнено(СтрокаТЧ.Количество) тогда

СтрокаТЧ.Стоимость=ПолучЦена(СтрокаТЧ.Код\_Услуги)\*СтрокаТЧ.Количество;

КонецЕсли;

КонецЕсли;

Иначе

Сообщить("На складе никогда не было материалов для выполнения данной услуги");

Возврат;

КонецЕсли;

КонецПроцедуры

&НаСервере

функция ПроверьСклад(Материалы)

//{{КОНСТРУКТОР\_ЗАПРОСА\_С\_ОБРАБОТКОЙ\_РЕЗУЛЬТАТА

// Данный фрагмент построен конструктором.

// При повторном использовании конструктора, внесенные вручную изменения будут утеряны!!!

Запрос = Новый Запрос;

Запрос.Текст =

"ВЫБРАТЬ

| ОстаткиМатериалаОстатки.Материалы КАК Материалы,

| ОстаткиМатериалаОстатки.КоличествоОстаток КАК Количество

|ИЗ

| РегистрНакопления.ОстаткиМатериала.Остатки КАК ОстаткиМатериалаОстатки

|ГДЕ

| ОстаткиМатериалаОстатки.Материалы = &Материалы";

Запрос.УстановитьПараметр("Материалы", Материалы);

РезультатЗапроса = Запрос.Выполнить();

Если НЕ РезультатЗапроса.Пустой() Тогда

ВыборкаДетальныеЗаписи = РезультатЗапроса.Выбрать();

ВыборкаДетальныеЗаписи.Следующий();

Возврат ВыборкаДетальныеЗаписи.Количество;

Иначе

Возврат 0;

КонецЕсли;

//}}КОНСТРУКТОР\_ЗАПРОСА\_С\_ОБРАБОТКОЙ\_РЕЗУЛЬТАТА

конецфункции

&НаКлиенте

Процедура УслугиКод\_УслугиПриИзменении(Элемент)

СтрокаТЧ= Элементы.Услуги.ТекущиеДанные; //текущие данные строки табличной части

СтрокаТЧ.Стоимость=ПолучЦена(СтрокаТЧ.Код\_Услуги);

КонецПроцедуры

&НаКлиенте

Процедура УслугиКоличествоПриИзменении(Элемент)

СтрокаТЧ= Элементы.Услуги.ТекущиеДанные; //текущие данные строки табличной части

Если ЗначениеЗаполнено(СтрокаТЧ.Материал) тогда

Если ПроверьСклад(СтрокаТЧ.Материал)<>0 тогда

Если ПроверьСклад(СтрокаТЧ.Материал)<СтрокаТЧ.Количество тогда

х=СтрокаТЧ.Количество-ПроверьСклад(СтрокаТЧ.Материал);

Сообщить("На складе не хватает "+х);

Возврат;

Иначе

СтрокаТЧ= Элементы.Услуги.ТекущиеДанные; //текущие данные строки табличной части

Если ЗначениеЗаполнено(СтрокаТЧ.Количество) тогда

СтрокаТЧ.Стоимость=ПолучЦена(СтрокаТЧ.Код\_Услуги)\*СтрокаТЧ.Количество;

КонецЕсли;

КонецЕсли;

Иначе

Сообщить("На складе никогда не было материалов для выполнения данной услуги");

Возврат;

КонецЕсли;

КонецЕсли;

КонецПроцедуры

*Код модуля объекта «Заказ»:*

Процедура ОбработкаПроведения(Отказ, Режим)

//{{\_\_КОНСТРУКТОР\_ДВИЖЕНИЙ\_РЕГИСТРОВ

// Данный фрагмент построен конструктором.

// При повторном использовании конструктора, внесенные вручную изменения будут утеряны!!!

// регистр ОстаткиМатериала Расход

Движения.ОстаткиМатериала.Записывать = Истина;

Для Каждого ТекСтрокаУслуги Из План\_работы Цикл

Движение = Движения.ОстаткиМатериала.Добавить();

Движение.ВидДвижения = ВидДвиженияНакопления.Расход;

Движение.Период = Дата;

Движение.Материалы = ТекСтрокаУслуги.Материал;

Движение.Количество = ТекСтрокаУслуги.Количество;

КонецЦикла;

//}}\_\_КОНСТРУКТОР\_ДВИЖЕНИЙ\_РЕГИСТРОВ

КонецПроцедуры

*Код модуля менеджера документа «Заказ»*

Процедура Чек(ТабДок, Ссылка) Экспорт

//{{\_КОНСТРУКТОР\_ПЕЧАТИ(Чек)

Макет = Документы.Заказ.ПолучитьМакет("Чек");

Запрос = Новый Запрос;

Запрос.Текст =

"ВЫБРАТЬ

| Заказ.Дата,

| Заказ.Код\_клиента,

| Заказ.Код\_Мастера,

| Заказ.Стоимость,

| Заказ.План\_работы.(

| Код\_Услуги,

| Стоимость

| )

|ИЗ

| Документ.Заказ КАК Заказ

|ГДЕ

| Заказ.Ссылка В (&Ссылка)";

Запрос.Параметры.Вставить("Ссылка", Ссылка);

Выборка = Запрос.Выполнить().Выбрать();

ОбластьЗаголовок = Макет.ПолучитьОбласть("Заголовок");

Шапка = Макет.ПолучитьОбласть("Шапка");

Стоимость = Макет.ПолучитьОбласть("Стоимость");

ОбластьПлан\_работыШапка = Макет.ПолучитьОбласть("План\_работыШапка");

ОбластьПлан\_работы = Макет.ПолучитьОбласть("План\_работы");

Подвал = Макет.ПолучитьОбласть("Подвал");

Шапка.Параметры.Название\_Предприятия = Константы.Название\_Предприятия.Получить();

ТабДок.Очистить();

ВставлятьРазделительСтраниц = Ложь;

Пока Выборка.Следующий() Цикл

Если ВставлятьРазделительСтраниц Тогда

ТабДок.ВывестиГоризонтальныйРазделительСтраниц();

КонецЕсли;

ТабДок.Вывести(ОбластьЗаголовок);

Шапка.Параметры.Заполнить(Выборка);

Стоимость.Параметры.Заполнить(Выборка);

ТабДок.Вывести(Шапка, Выборка.Уровень());

ТабДок.Вывести(ОбластьПлан\_работыШапка);

ВыборкаПлан\_работы = Выборка.План\_работы.Выбрать();

Пока ВыборкаПлан\_работы.Следующий() Цикл

ОбластьПлан\_работы.Параметры.Заполнить(ВыборкаПлан\_работы);

ТабДок.Вывести(ОбластьПлан\_работы, ВыборкаПлан\_работы.Уровень());

КонецЦикла;

ТабДок.Вывести(Стоимость);

Подвал.Параметры.Заполнить(Выборка);

ТабДок.Вывести(Подвал);

ВставлятьРазделительСтраниц = Истина;

КонецЦикла;

//}}

КонецПроцедуры

*Команда модуля команды «Чек» в документе «Заказ»*

&НаКлиенте

Процедура ОбработкаКоманды(ПараметрКоманды, ПараметрыВыполненияКоманды)

//{{\_КОНСТРУКТОР\_ПЕЧАТИ(Чек)

ТабДок = Новый ТабличныйДокумент;

Чек(ТабДок, ПараметрКоманды);

ТабДок.ОтображатьСетку = Ложь;

ТабДок.Защита = Ложь;

ТабДок.ТолькоПросмотр = Ложь;

ТабДок.ОтображатьЗаголовки = Ложь;

ТабДок.Показать();

//}}

КонецПроцедуры

&НаСервере

Процедура Чек(ТабДок, ПараметрКоманды)

Документы.Заказ.Чек(ТабДок, ПараметрКоманды);

КонецПроцедуры