

ГУАП

КАФЕДРА № 42

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ
ПРЕПОДАВАТЕЛЬ

Доцент, к.т.н.

А.В. Бржезовский

должность, уч. степень, звание

подпись, дата

инициалы, фамилия

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К КУРСОВОЙ РАБОТЕ

АВТОМАТИЗИРОВАННАЯ ИНФОРМАЦИОННАЯ СИСТЕМА ТИПОГРАФИИ

по курсу: МЕТОДЫ И СРЕДСТВА ПРОЕКТИРОВАНИЯ
ИНФОРМАЦИОННЫХ СИСТЕМ И ТЕХНОЛОГИЙ

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. №

4026

С.М. Николаева

подпись, дата

инициалы, фамилия

Санкт-Петербург 2023

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	3
1. Описание предметной области	4
2. Требования к системе	5
2.1 Определение пользовательских требований	5
2.2 Функциональные требования	11
3. Концептуальная модель БД	18
4. Физическая модель БД	19
5. Интерфейс пользователя	20
ЗАКЛЮЧЕНИЕ	25
ПРИЛОЖЕНИЕ А	26
ПРИЛОЖЕНИЕ Б	35

ВВЕДЕНИЕ

Целью данной курсовой работы является разработка автоматизированной информационной системы типографии для сотрудников предприятия.

Основной задачей разработки является сокращение временных издержек при составлении договора и формировании технических карт, обеспечение контроля наличия материала на складе и отслеживание его потока от склада к производству.

Перед разработкой системы необходимо составить перечень правил, требований, список будущих пользователей и их обязанности, расписать алгоритмы поведения пользователей. Все это указывается в соответствии с требованиями пользователей.

Для разработки использована среда MS Visual Studio, база данных реализована на SQL Server Management Studio.

1. Описание предметной области

1.1 Исходные данные

Система документооборота в типографии в большинстве своем не автоматизирована. Большие временные затраты уходят не только на согласование макета продукта, но и на бумажную волокиту с оформлением договоров и технических карт.

Для согласования всех пунктов для одного даже не крупного заказа менеджер типографии может потратить более суток, а на подписание такого договора уйдет еще больше времени. А в день для одной типографии может прийти десяток таких заказов.

1.2 Возможности бизнеса

Система автоматизации документооборота позволит значительно сократить временные издержки при составлении договора, формировании технических карт для технологов производства, а также контроля наличия материала на складе и отслеживание его потока от склада к производству.

Значительно возрастет показатель эффективности предприятия и снизится монотонная нагрузка с менеджеров типографии. Подобная система позволит собирать статистические данные для последующего планирования ведения бизнеса.

1.3 Бизнес-цели

Финансовые

- Привлечение компаний среднего и большого полиграфического производства к использованию системы Polygraf
- Уменьшение временных затрат на согласование и формирование договора оказания услуг.

Нефинансовые

- Развитие системы автоматизации процесса документооборота для повышения эффективности и минимизации ошибок в работе печатного предприятия
- Извлечение статистических данных для дальнейшего анализа бизнес-аналитиками и координации бизнеса

2. Требования к системе

2.1 Определение пользовательских требований

Для описания взаимодействия между пользователем и системой была создана use case диаграмма в программе Power Designer.

Список пользователей разрабатываемой системы:

Сотрудник типографии (обобщенный актер), в его задачи входит: создание и редактирование договора, просмотр и редактирование статуса заказа и технической карты, закупка материала и просмотр статистики.

Клиент, который может запросить статус договора, создать и отредактировать договор в случае несоответствия полученной тестовой печати с ожиданиями клиента.

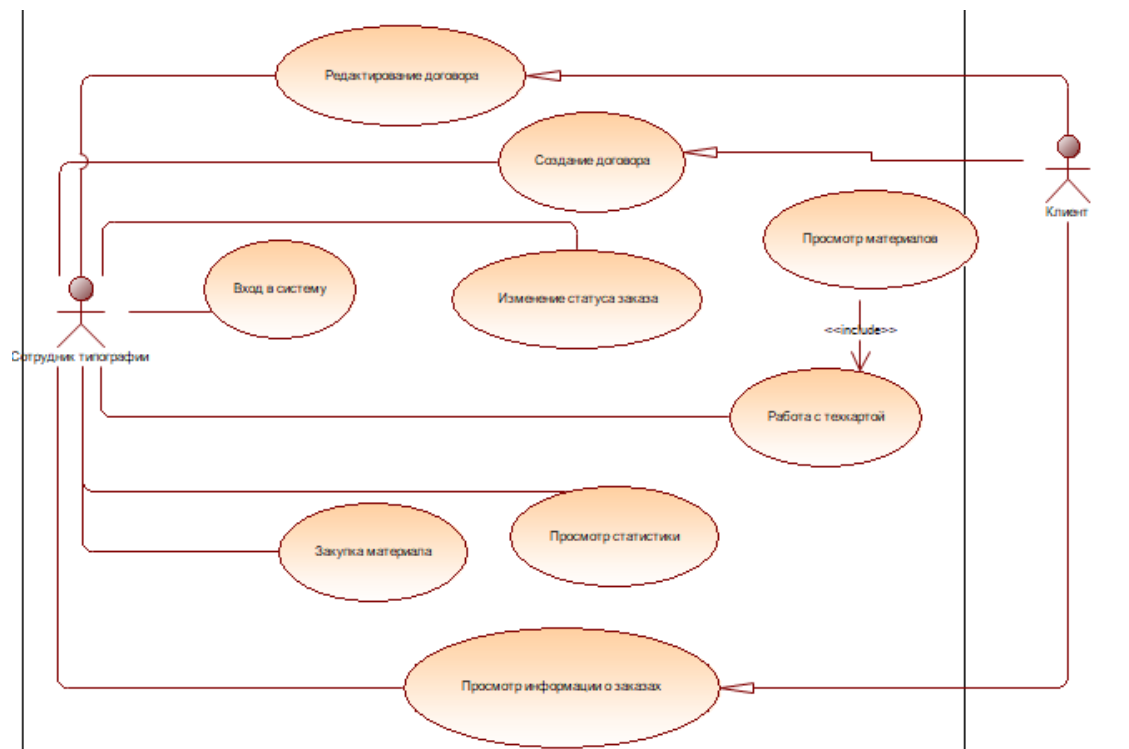


Рисунок 2.1 – Use Case диаграмма

Описание прецедентов

1. Вход в систему

Подразумевается идентификация сотрудника в системе, используется пароль и почта для начала сессии. Из данного прецедента проистекают остальные.

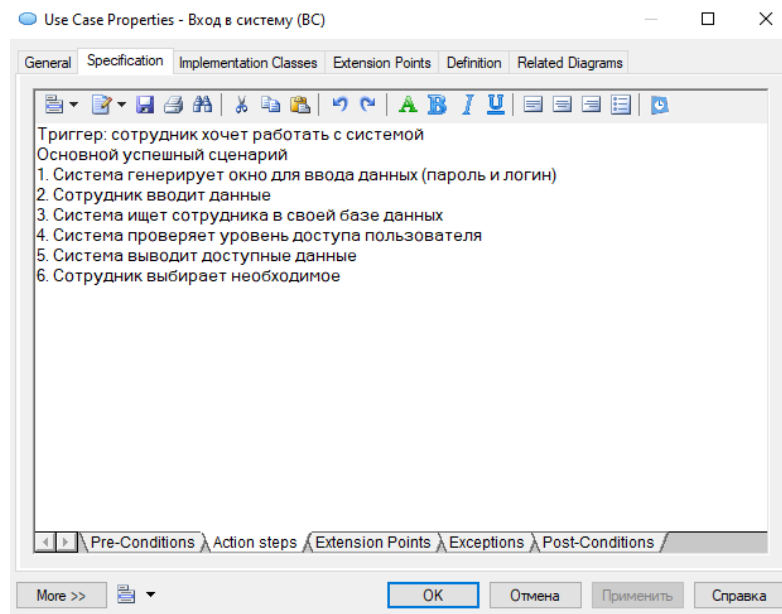


Рисунок 2.2 – Основной сценарий «Вход в систему»

Расширения

За Пользователя не существует

1. Система выводит сообщение о некорректности введенных данных и возвращается к первоначальному состоянию

Постусловие

Система регистрирует время входа сотрудника в систему

2. Создание договора

Подразумевается создание договора и связанной с ним технической карты.

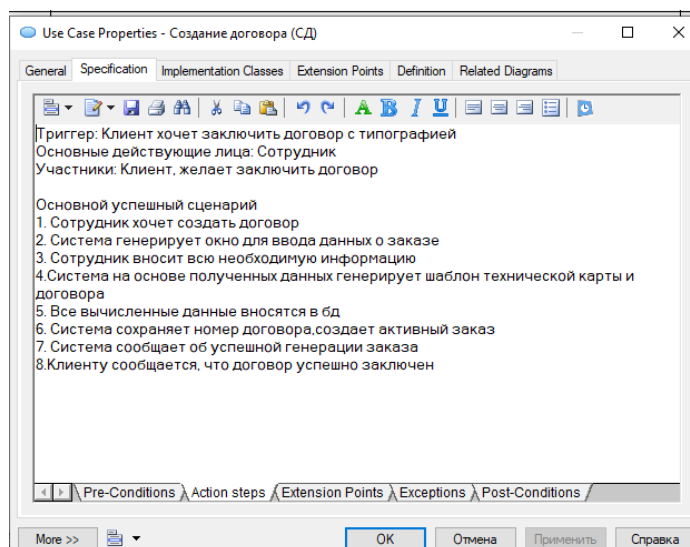


Рисунок 2.3 – Основной сценарий прецедента «Создание договора»

Расширение: 4а Данных недостаточно

1. Система уведомляет что информации о заказе не хватает и возвращается к изначальному состоянию

Предусловие: Сотрудник имеет все необходимые для оформления договора данные о структуре заказа (тз), а также данные о клиенте (номер телефона и фии клиента).

Постусловие: Система сохраняет данные о структуре заказа, данные о договоре (в том числе дату и время его генерации) и технической карте.

3. Редактирование договора

В данном прецеденте осуществляется редактирование договора.

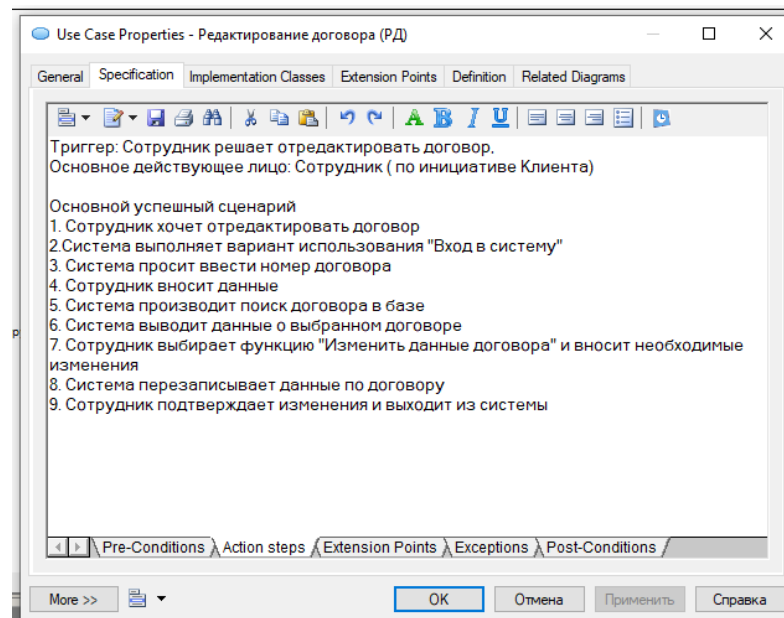


Рисунок 2.4 – Основной успешный сценарий «Редактирование договора»

Расширения

5а некорректный номер договора

1. Система просит ввести данные заново и очищает поле номер договора

Постусловие

Система обновляет данные о договоре и технической карте

4. Работа с технической картой

Триггер: требуется просмотреть и поработать с технической картой

Основные действующие лица: Сотрудник типографии

Основной успешный сценарий:

1. Сотрудник хочет просмотреть техкарту
2. Система выполняет вариант использования "Вход в систему"
3. Система генерирует окно с активными заказами
4. Сотрудник выбирает из перечисленного необходимую карту
5. Система предоставляет данные технической карты
6. Сотрудник решает добавить к техкарте материал
7. Система применяет вариант использования "Просмотр материалов"

Работа с технической картой включает в себя «Просмотр материалов»: ниже представлены основной сценарий и исключительная ситуация нехватки материала на складе.

Просмотр материалов имеет

Предусловие: производится создание договора

Постусловие: Система сохраняет выбранный материал в структуре заказа.

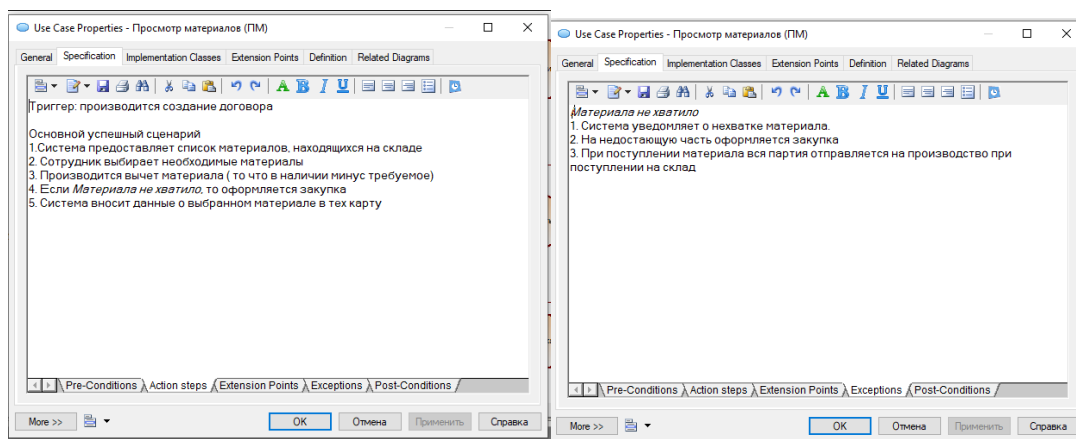


Рисунок 2.5 – Основной сценарий и исключительная ситуация
«Просмотр материалов»

5. Просмотр информации о заказах

Триггер: необходимо проверить статус заказа

Основные действующие лица: Сотрудник и Клиент

Основной успешный сценарий

1. Клиент хочет проверить статус заказа
2. Система выполняет вариант использования "Вход в систему"
3. Сотрудник вводит номер договора
4. Система генерирует на дисплее доступные данные о статусе договора
5. Сотрудник уведомляет клиента о статусе его заказов

Предусловие

Сотрудник знает данные клиента и номер договора

Расширение

3а. Договора не существует

1. Система выводит сообщение о некорректности введенных данных и возвращается к первоначальному состоянию, очищая поле ввода.

4а Активных заказов нет

1. Система уведомляет об отсутствии активных заказов и переходит в начальное состояние

6. Изменение статуса заказа

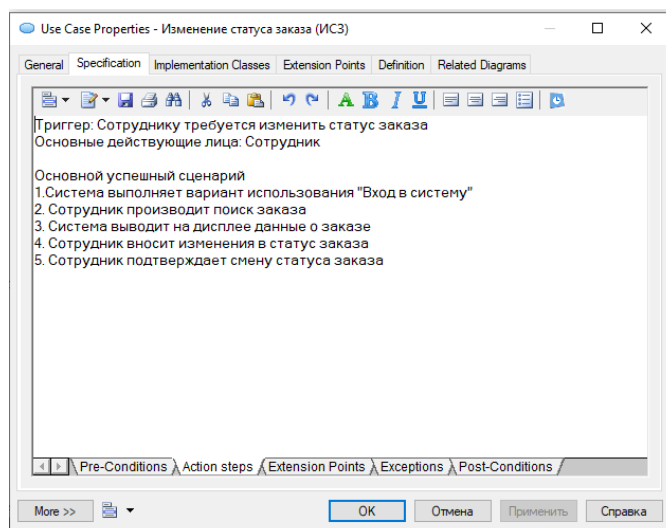


Рисунок 2.6 – Основной успешный сценарий «Изменение статуса заказа»

Предусловие: Сотрудник знает, какой номер заказа и на какой статус необходимо изменить

Постусловие: Система обновляет статус заказа и сохраняет дату его изменения

7. Просмотр статистики

Триггер: Сотруднику бухгалтерии необходимо просмотреть статистику за месяц

Основные действующие лица: Сотрудник бухгалтерии

1. Система выполняет вариант использования "Вход в систему"

2. Система генерирует окно статистики

3. Сотрудник выбирает узнать статистику сотрудников

3.1 Вводит ID сотрудника и нажимает кнопку "Найти сотрудника"

3.2 Система производит подсчет количества договоров или технических карт, закрепленных за выбранным сотрудником, и выводит полученное значение

4. Сотрудник выбирает узнать статистику мобильности материалов за месяц

4.1 Сотрудник выбирает материал и нажимает кнопку "Статистика товара"

4.2 Система производит подсчет количества прибывшего и убывшего материала и выводит полученные данные в соответствующие поля.

Предусловие Сотрудник знает, какие данные статистики ему необходимо получить

8. Закупка материала

Триггер: необходимо произвести закупку

Основной сценарий:

1. Сотрудник хочет произвести закупку

2. Система выполняет вариант использования "Вход в систему"

3. Система генерирует окно для работы с материалами

4. Сотрудник выбирает материал и количество и нажимает кнопку "Добавить в закупку"

5. Система вносит выбранные данные в таблицу закупки и отображает их появление в форме

Предусловие: Сотрудник знает какие материалы и в каком количестве необходимо добавить в закупку

Постусловие: Система обновляет данные по закупкам

2.2 Функциональные требования

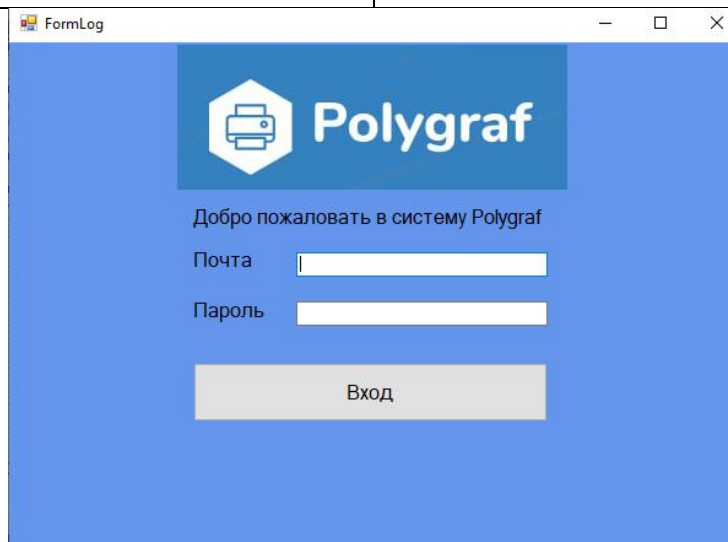
1. Вход в систему

1.1 Описание

Для работы с системой должен быть произведен вход сотрудником по логину и паролю. В зависимости от занимаемой должности сотруднику будет доступен определенный функционал системы Polygraf.

1.2 Функциональные требования

Доступ: Инициализация	Осуществление входа в систему
.Вход	Сотрудник должен войти в систему для дальнейшей работы с ней, если сотрудник ввел некорректные данные, система просит его повторить попытку



Форма входа

2 Работа с договором

2.1 Описание

Сотрудник (менеджер) может добавлять договор, редактировать данные в нем, а также удалять договора при условии отсутствия рабочих технических карт.

2.2 Функциональные требования

Контракт. Выбрать:	Работа с договором
.Создать договор	Добавление нового договора, необходимо указать данные о клиенте, дату подписания и прогнозируемую дату завершения работы с заказом, осуществляется по кнопке «Создать договор», номер договора создается согласно РО-1
.Очистить	Очищает поля Имя и Фамилия Номер телефона клиента для внесения новых данных или полного изменения данных о клиенте
.Удалить	Выбор контракта, который необходимо удалить из системы, осуществляется по кнопке «Удалить». Договор проверяется на соответствие с РО-2
.Изменить данные клиента	Внесение изменений в информацию о контракте, например смена даты завершения работ или контактные данные клиента. Сохранение внесенных изменений осуществляется по кнопке «Изменить данные клиента»

The screenshot displays the 'FormContract' application window. It features a top navigation bar with four buttons: 'Окно договоров' (selected), 'Окно технической карты', 'Окно материалов', and 'Окно статистики'. Below the navigation bar, on the left, is a table with columns 'ContractId', 'ClientId', 'DateOfBegin', and 'DateOfEnd'. The table contains four rows of data. On the right, there is a form titled 'Работа с договором' with input fields for 'Имя' (Name), 'Фамилия' (Surname), and 'Телефон' (Phone). Below these fields are buttons for 'Очистить' (Clear) and 'Изменить данные клиента' (Edit client data). Further down, there are date pickers for 'Дата подписания' (Signing date) and 'Дата окончания' (Completion date), followed by buttons for 'Создать договор' (Create contract) and 'Удалить договор' (Delete contract). The status bar at the bottom left shows 'Статус договора'.

ContractId	ClientId	DateOfBegin	DateOfEnd
1	1	01.07.2023	10.07.2023
2	2	07.07.2023	14.07.2023
3	3	13.08.2023	21.08.2023
4	4	09.09.2023	14.09.2023

Форма работы с договором

3 Работа со статусами

3.1 Описание

Сотрудник (менеджер) имеет доступ к изменениям статуса контракта. Статус контракта изменяется в соответствии с прохождением этапа разработки проекта: от подписания договора до выдачи готовой продукции клиенту.

Всего существует ряд статусов:

- на подписании (для сложных заказов)
- тест-печать
- проверка
- основная печать
- завершен

3.2 Функциональные требования

Статус. Сменить:	Изменение статуса заказа
.выбрать	Сотрудник выбирает статус из списка возможных, по соответствию РО-3
.подтвердить смену статуса	Сохранить выбранный статус с сохранением ФИО сотрудника и времени смены статуса.
.Поиск	Производится поиск договора и его статусов при вводе номера договора и нажатию кнопки «Поиск»

The screenshot shows a web application interface titled "Статус договора" (Contract Status). It features a table with the following columns: Id_Status, ContractId, Data_change, and Status. The table contains six rows of data. To the right of the table is a form with the title "Определить статус" (Determine status). The form includes a text input field labeled "Номер договора" (Contract number) and a "Поиск" (Search) button. Below this, there is a dropdown menu labeled "Новый статус" (New status) and a blue button labeled "Подтвердить изменение статуса" (Confirm status change).

	Id_Status	ContractId	Data_change	Status
▶	1	1	01.07.2023	На подписании...
	2	1	02.07.2023	Тест-печать
	3	1	04.07.2023	Проверка
	4	2	07.07.2023	На подписании...
	5	2	10.07.2023	Тест-печать
	6	2	12.07.2023	Проверка

Определить статус

Номер договора

Новый статус

Форма работы со статусами в договоре

4. Работа с техническими картами

4.1 Описание

Сотрудник – технолог может проводить операции CRUD над техническими картами, а также прикреплять к технической карте материал. В случае если материала недостаточно на складе, выводится сообщение, и операция внесения материала в техническую карту прерывается.

4.2 Функциональные требования

Техкарта .Выбрать	Работа с технической картой
.Добавить техкарту	Сотрудник заполняет данные технической карты (указывает номер контракта, тип печати, объект и тираж) и путем нажатия кнопки «Добавить техкарту» вносит данные в систему
.Изменить	Сохранение внесенных изменений в структуру техкарты
.Удалить	Удаление технической карты
.Добавить материал	Добавление материала к технической карте, выбирается тип материала и вводится количество (в соответствии с РО-4). Если материал не проходит условие РО-5, процесс прерывается с всплывающим сообщением.
.Изменить	Сохранение изменений, внесенных в материал
.Удалить	Удаление выбранного материала из технической карты

Форма работы с техническими картами

5. Работа с материалами

5.1 Описание

Сотрудник склада может вносить материалы в закупку и подтверждать прием материала на склад.

5.2 Функциональные требования

Материал. Выбрать	Работа с материалами на складе
.Внести в закупку	Сотрудник выбирает материал и количество единиц и нажимает кнопку «Внести в закупку»
.Доставлено на склад	При поступлении материала на склад, сотрудник выбирает поступивший материал из списка закупок и нажимает кнопку «Доставлено на склад»
.Очистить список закупок	Все материалы из списка закупок добавляются на склад, список закупок очищается

Форма работы с материалами

6. Работа со статистикой

6.1 Описание

Сотрудник бухгалтер может просмотреть статистику по сотрудникам – у кого сколько договоров или технических карт закреплено. Также можно просматривать статистику по материалам, количеству материала, который сейчас на складе, который забрали в производство и сколько привезли на склад в течение месяца.

6.2 Функциональные требования

Статистика.Выбрать	Работа со статистикой материала и продуктивностью сотрудников
.Найти сотрудника	При вводе ID сотрудника система выводит имя фамилию и должность сотрудника, а также количество договоров/техкарт, закрепленных за ним
.Статистика товара	При выборе материала из возможных система производит подсчет количества материала: сейчас на складе, прибывшего на склад и отправленного в производство за последний месяц.

The screenshot displays the 'FormStatistic' application window. At the top, there are three tabs: 'Окно договоров' (Contracts Window), 'Окно технической карты' (Technical Card Window), and 'Окно материалов' (Materials Window). The main interface is divided into several sections. On the left, there is a form for finding an employee, with fields for 'Номер сотрудника' (Employee ID) containing '2', 'Имя' (Name) containing 'Павел', 'Фамилия' (Surname) containing 'Швецов', 'Должность' (Position) containing 'Менеджер', and 'Количество договоров' (Number of contracts) containing '2'. A blue button labeled 'Найти сотрудника' (Find employee) is below these fields. In the center, there is a section for material statistics, with a dropdown for 'Наименование товара' (Goods name) set to 'Бумага мел' (Fine paper), a button labeled 'Статистика товара' (Goods statistics), and three input fields for 'Сколько на складе' (How much in stock) set to '100', 'Сколько забрали' (How much taken) set to '-50', and 'Сколько привезли' (How much brought) set to '0'. On the right, there is a table showing the history of material changes. The table has columns: 'Id_History', 'Name_M', 'EmployerId', 'Quantity', and 'Data_change_Quantity'. The data rows are as follows:

Id_History	Name_M	EmployerId	Quantity	Data_change_Quantity
1	Бумага мел	3	-20	20.12.2023 23:24
2	Бумага офсет	3	-20	20.12.2023 23:26
3	Бумага картон	4	-10	20.12.2023 23:27
4	Бумага картон	5	-20	20.12.2023 23:32
5	Бумага мел	5	-30	20.12.2023 23:32
6	Бумага офсет	5	-10	20.12.2023 23:32
7	Бумага постерная	6	200	20.12.2023 23:33

Форма работы со статистикой

Бизнес-правила

Идентификатор	Определение	Тип	Статика/ динамика	Источник
PO-1	Идентификационный номер договора, является сквозным, при удалении возможны повторения, редактированию не подлежит	Факт	Статика	Правила целостности проекта
PO-2	Нельзя удалить договор, который имеет статус, то есть запущенный в производство заказ	Ограничение	Статика	Правила целостности договора
PO-3	Статус предыдущего этапа не может быть выбран на следующем шаге (за исключением пары статусов – проверка и тест-печать).	Ограничение	Динамика	Политика системы Polygraf
PO-4	Количество материала не может быть отрицательным или равно 0	Ограничение	Статика	Правила целостности системы
PO-5	Количество указываемого материала должно быть меньше, чем количество материала на складе	Ограничение	Статика	Правила целостности системы

3. Концептуальная модель БД

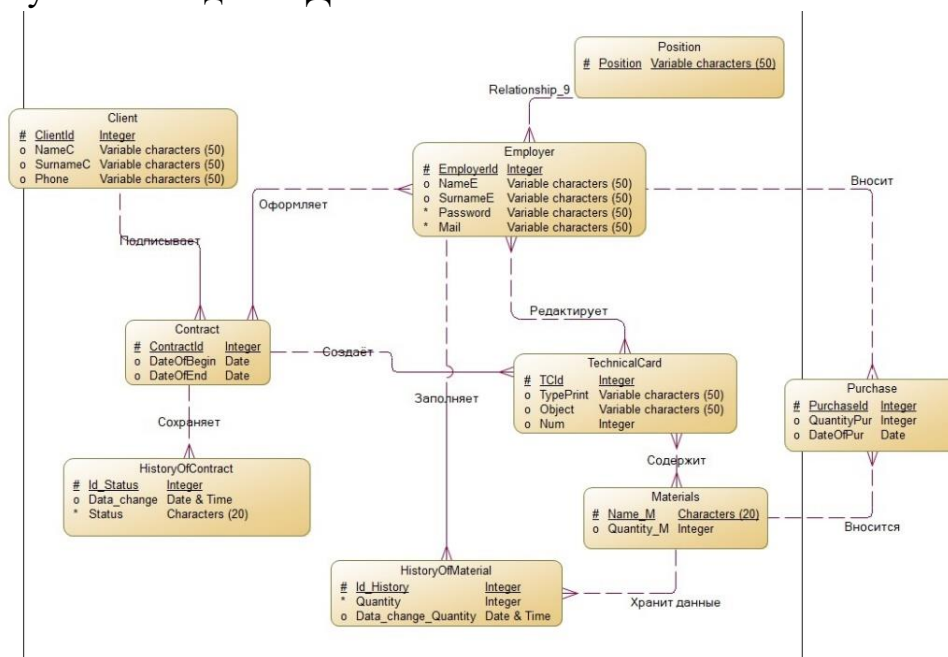


Рисунок 3.1 – Концептуальная модель

Модель содержит в себе ряд таблиц:

- Client – клиенты типографии
- Employer – Сотрудники типографии
- Contract - Договор
- Purchase – Закупка материала
- Material – Материал в настоящее время на складе
- History of Contract - История смена статусов договоров
- History Of Material – История мобильности материала
- Technical Cart - Техническая карта договора
- Position – Должности сотрудников

Так же существуют таблицы, позволяющие реализовать связь многие–ко–многим, это таблицы: «Создает», «Оформляет» и «Редактирует».

4. Физическая модель БД

Код для создания базы данных прикреплен в приложении А. На рисунке 4.1 представлена модель базы данных, для работы с которой были спроектированы формы.

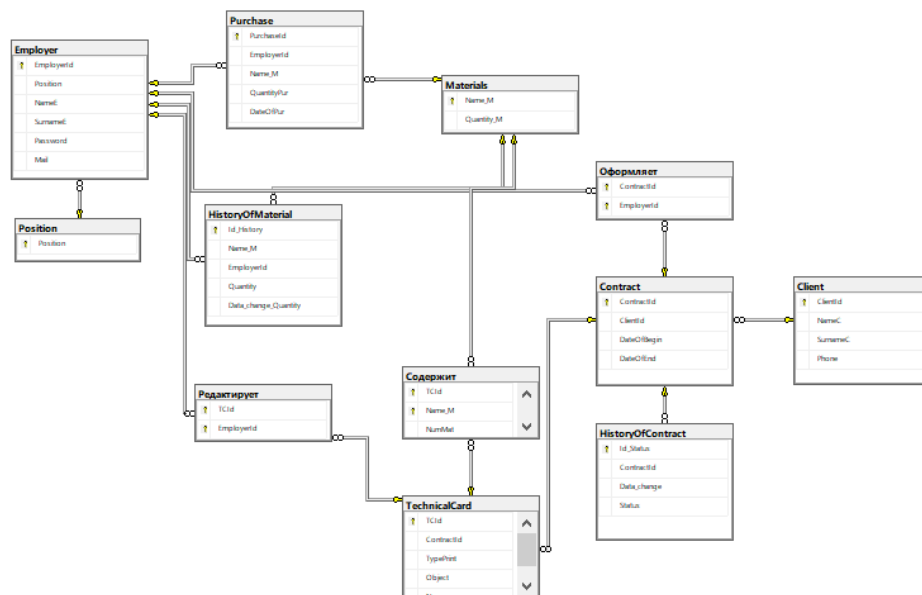


Рисунок 4.1 – Модель базы данных

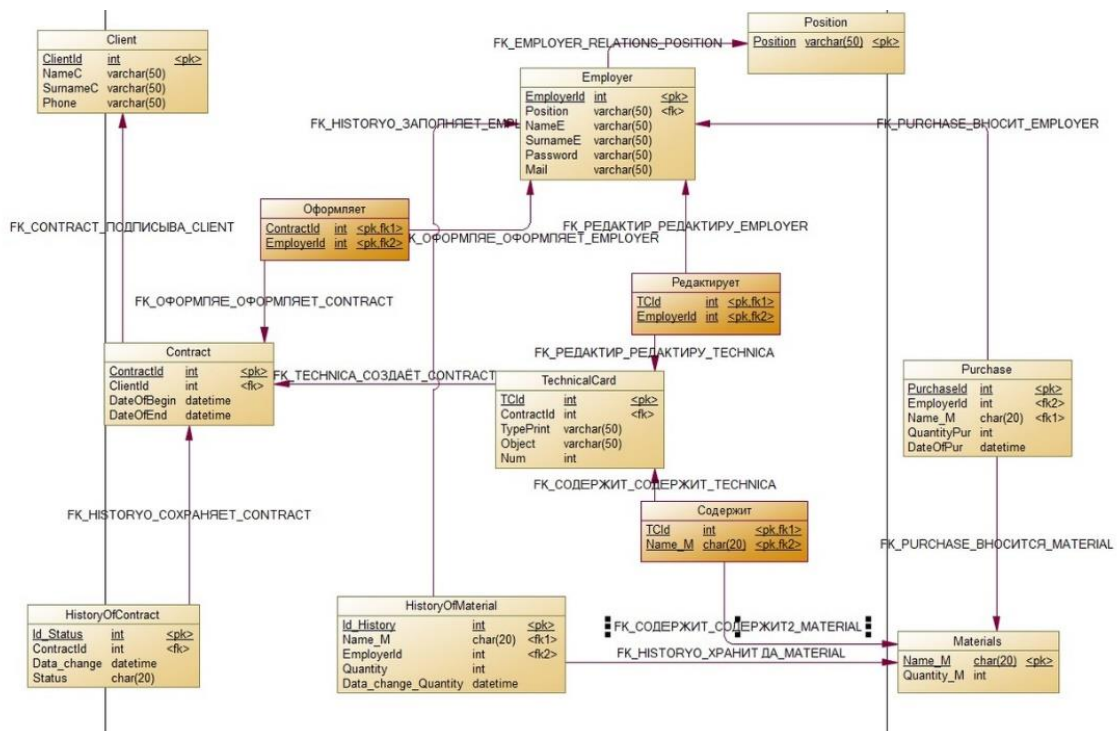


Рисунок 4.2 – Физическая модель базы данных

Хранимые процедуры, триггеры и индексы не использовались в разработке информационной системы типографии «Polygraf».

Ограничения целостности наложены на систему входа, на доступность частей системы конкретному сотруднику в зависимости от занимаемой должности. Таким образом сотрудник имеет доступ только к своей форме.

Организовано каскадное удаление и обновление данных для: технической карты и таблицы «Оформляет» если удален контракт, «Редактирует» и «Содержит» если удаляется техническая карта.

Скрипты для обработки всех действий в системе «Polygraf» приведены в приложении Б.

5. Интерфейс пользователя

Система разработана с помощью языка C# и представляет собой формы ADO.NET.

Первая форма – авторизация сотрудника (рисунок 5.1). Сюда вводится пароль и почта, которые проверяются на соответствие существующим в базе данных логинам и паролям. В случае ошибочного ввода всплывает сообщение (рисунок 5.2).



Рисунок 5.1 – Форма входа

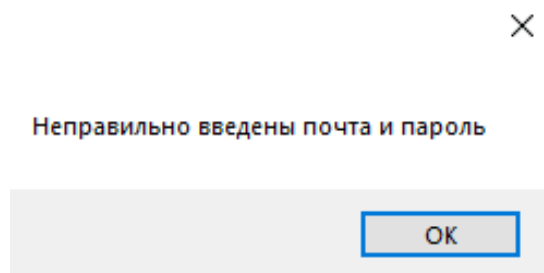
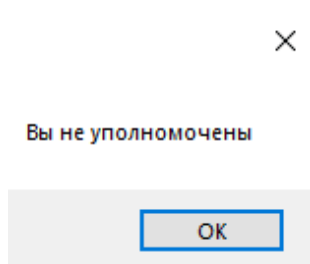


Рисунок 5.2 – Сообщение об ошибке

Таблица с паролями

Id	Должность	Имя	Фамилия	Пароль	Почта
1	Менеджер	Валерия	Муравьева	jknyjhiosn	bkgfngbjkxfgek@mail.ru
2	Менеджер	Павел	Швецов	awefkjotjhop	hfbgkbxfkgnfkk@mail.ru
3	Технолог	Анастасия	Фокина	aweio;fjofu	cfgncfghfcgh@mail.ru
4	Технолог	Дарина	Панина	awefkjhibugt	xfgxfgfhfgr@mail.ru
5	Технолог	Никита	Терентьев	eakulbgjhbul	hgxfxghfghgv@mail.ru
6	Работник склада	Дмитрий	Панкратов	hbfkjhxlirjrg	ghiftjgojhg@mail.ru
7	Бухгалтер	Софья	Николаева	fdfndgnfg	sofya@mail.ru

Далее в зависимости от занимаемой должности сотруднику открывается форма его непосредственных обязанностей. Для менеджера – это окно договоров, для технолога- окно технических карт, для сотрудника склада – окно материалов, а для бухгалтера окно для работы со статистикой. В случае перехода сотрудника к окну другого подразделения и попытке внесения каких-либо изменений система выведет ошибку, представленную ниже. На рисунке 5.3 представлен граф перехода между формами.



Ошибка при нарушении полномочий

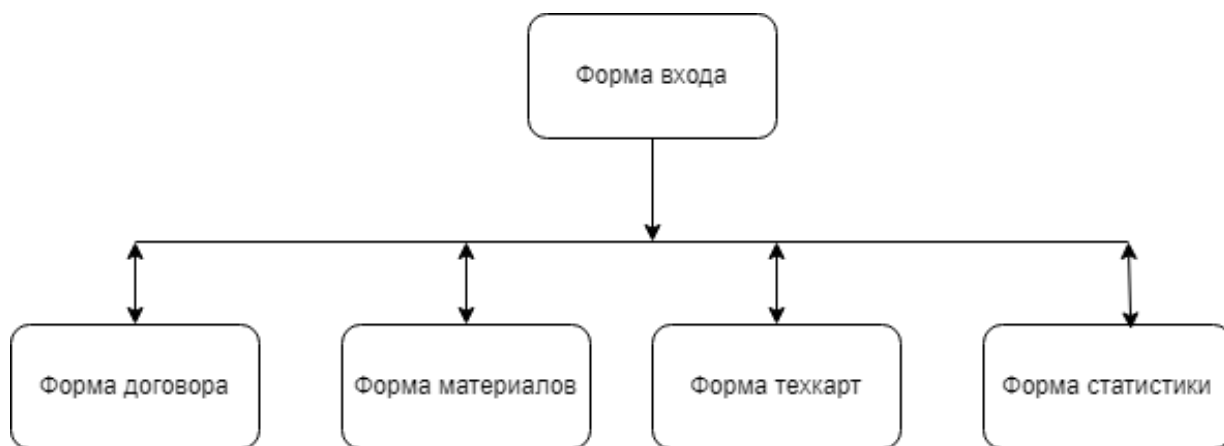


Рисунок 5.3 – Граф перехода форм

Вторая форма – для работы с договором. Менеджер может создавать новые договора, изменять данные о существующих договорах, отмечать статус договоров. По нажатию на логотип системы производится выход из аккаунта и возврат к первой форме. При удалении договора, у которого есть статус (то есть договор заключен и заказ начал создаваться) система прерывает удаление и выводит сообщение, представленное на рисунке 5.5.

ContractId	ClientId	DateOfBegin	DateOfEnd
1	1	01.07.2023	10.07.2023
2	2	07.07.2023	14.07.2023
3	3	13.08.2023	21.08.2023
4	4	09.09.2023	14.09.2023

Id_Status	ContractId	Data_change	Status
1	1	01.07.2023	На подписании...
2	1	02.07.2023	Тест-печать
3	1	04.07.2023	Проверка
4	2	07.07.2023	На подписании...
5	2	10.07.2023	Тест-печать

Рисунок 5.4 – Форма работы с договором

Завершите все технические карты данного контракта!

OK

Рисунок 5.5 – Сообщение об ошибке

Третья форма – для работы с технической картой (рисунок 5.6). Технолог может создавать техническую карту, изменять ее и удалять. К технической карте можно прикрепить материал и указать количество требуемого материала. При добавлении материала он вычитается из находящегося на складе количества. Если материала недостаточно, операция добавления прерывается сообщением, представленным на рисунке 5.7.

Рисунок 5.6 – Форма технической карты

Рисунок 5.7 – Сообщение об ошибке

Четвертая форма создана для работы с материалами (рисунок 5.8). Сотрудник склада может добавить материал в закупку, подтверждать прием на склад как одного материала, так и всего списка закупки.

The FormMaterial application window features a top navigation bar with four buttons: "Окно договоров", "Окно технических карт", "Окно материалов" (highlighted in blue), and "Окно статистики".

Below the navigation bar, the interface is divided into two main sections:

- Список материалов (Materials List):** A table with columns "Name_M" and "Quantity_M". It lists:

Name_M	Quantity_M
Бумага картон	150
Бумага мел	100
Бумага офсет	100
Бумага постерная	150
- Список записей (Records List):** A table with columns "PurchaseId", "EmployerId", "Name_M", "QuantityPur", and "DateOfPur". It shows one record:

PurchaseId	EmployerId	Name_M	QuantityPur	DateOfPur
2	6	Бумага постерная	200	20.12.2023 23:33

Below the "Список материалов" table, there is a section "Работа с материалом" (Material Work) with a dropdown menu for "Наименование" (Name) set to "Бумага картон", an input field for "Количество единиц" (Quantity in units), and a blue button "Добавить в записку" (Add to note).

Below the "Список записей" table, there are two buttons: "Доставлено на склад" (Delivered to warehouse) and "Очистить список записки" (Clear list of notes).

Рисунок 5.8 – Форма работы с материалами

Пятая форма создана для расчета статистики. Она позволяет узнать о количестве договоров/технических карт у выбранного сотрудника (рисунок 5.9) и просмотреть мобильность материалов за месяц.

The FormStatistic application window has a top navigation bar with three buttons: "Окно договоров", "Окно технической карты", and "Окно материалов" (highlighted in blue).

The main interface is divided into two columns:

- Left Column (Employee Selection):**
 - Input field for "Номер сотрудника" (Employee number) with value "2".
 - Blue button "Найти сотрудника" (Find employee).
 - Input field for "Имя" (Name) with value "Павел".
 - Input field for "Фамилия" (Surname) with value "Швецов".
 - Input field for "Должность" (Position) with value "Менеджер".
 - Input field for "Количество договоров" (Number of contracts) with value "2".
- Right Column (Material Statistics):**
 - Dropdown menu for "Наименование товара" (Goods name) set to "Бумага мел".
 - Blue button "Статистика товара" (Goods statistics).
 - Input field for "Сколько на складе" (How many in warehouse) with value "100".
 - Input field for "Сколько забрали" (How many took) with value "-50".
 - Input field for "Сколько привезли" (How many brought) with value "0".

At the bottom, there is a table showing the history of material changes:

	Id_History	Name_M	EmployerId	Quantity	Date_change_Quantity
▶	1	Бумага мел	3	-20	20.12.2023 23:24
	2	Бумага офсет	3	-20	20.12.2023 23:26
	3	Бумага картон	4	-10	20.12.2023 23:27
	4	Бумага картон	5	-20	20.12.2023 23:32
	5	Бумага мел	5	-30	20.12.2023 23:32
	6	Бумага офсет	5	-10	20.12.2023 23:32
	7	Бумага постерная	6	200	20.12.2023 23:33
*					

Рисунок 5.9 – Форма статистики

ЗАКЛЮЧЕНИЕ

В ходе выполнения курсовой работы была разработана система Polygraf, предназначенная для автоматизации документооборота в типографиях.

Разработана база данных системы в SQL Server Management Studio 19, а также ряд форм для комфортной работы с базой данных.

В данной версии система реализует операции CRUD с договорами и техническими картами, производит закупку материалов и выводит базовую статистику по сотрудникам и материалам.

Для полноценной работы в систему необходимо добавить уведомления для сотрудников от других подразделений о необходимости действий, и настроить предварительные уведомления о заканчивающемся материале.

ПРИЛОЖЕНИЕ А

Код для создания бд

Create database Polygraf

go

use Polygraf

go

/*=====*/

/* Table: Client */

/*=====*/

create table Client (

 ClientId int not null,

 NameC varchar(50) null,

 SurnameC varchar(50) null,

 Phone varchar(50) null,

 constraint PK_CLIENT primary key (ClientId)

)

go

/*=====*/

/* Table: Contract */

/*=====*/

create table Contract (

 ContractId int not null,

 ClientId int not null,

 DateOfBegin datetime null,

 DateOfEnd datetime null,

 constraint PK_CONTRACT primary key (ContractId)

)

go

/*=====*/

```

/* Table: Employer                                */
/*=====*/
create table Employer (
    EmployerId      int          not null,
    Position         varchar(50)  null,
    NameE            varchar(50)  null,
    SurnameE         varchar(50)  null,
    Password         varchar(50)  not null,
    Mail             varchar(50)  not null,
    constraint PK_EMPLOYER primary key (EmployerId)
)
go

```

```

/*=====*/
/* Table: HistoryOfMaterial                        */
/*=====*/
create table HistoryOfMaterial (
    Id_History       int          not null,
    Name_M           char(20)     not null,
    EmployerId       int          not null,
    Quantity          int          not null,
    Data_change_Quantity datetime null,
    constraint PK_HISTORYOFMATERIAL primary key (Id_History)
)
go

```

```

/*=====*/
/* Table: HistoryOfContract                        */
/*=====*/
create table HistoryOfContract (
    Id_Status        int          not null,
    ContractId       int          null,

```

```

Data_change    datetime    null,
Status         char(20)    not null,
constraint PK_HISTORYOFCONTRACT primary key (Id_Status)
)
go

```

```

/*=====*/
/* Table: Materials */
/*=====*/
create table Materials (
    Name_M      char(20)    not null,
    Quantity_M  int        null,
    constraint PK_MATERIALS primary key (Name_M)
)
go

```

```

/*=====*/
/* Table: Position */
/*=====*/
create table Position (
    Position    varchar(50) not null,
    constraint PK_POSITION primary key (Position)
)
go

```

```

/*=====*/
/* Table: Purchase */
/*=====*/
create table Purchase (
    PurchaseId  int        not null,
    EmployerId  int        null,
    Name_M      char(20)    null,

```

```

QuantityPur      int          null,
DateOfPur        datetime     null,
constraint PK_PURCHASE primary key (Purchaseld)
)
go

/*=====*/
/* Table: TechnicalCard */
/*=====*/
create table TechnicalCard (
    TCId           int          not null,
    ContractId     int          not null,
    TypePrint      varchar(50)   null,
    Object         varchar(50)   null,
    Num            int          null,
    constraint PK_TECHNICALCARD primary key (TCId)
)
go

/*=====*/
/* Table: Оформляет */
/*=====*/
create table Оформляет (
    ContractId     int          not null,
    EmployerId     int          not null,
    constraint PK_ОФОРМЛЯЕТ primary key (ContractId, EmployerId)
)
go

/*=====*/
/* Table: Редактирует */
/*=====*/

```

```

create table Редактирует (
    TCId          int          not null,
    EmployerId    int          not null,
    constraint PK_РЕДАКТИРУЕТ primary key (TCId, EmployerId)
)
go

/*=====*/
/* Table: Содержит */
/*=====*/

create table Содержит (
    TCId          int          not null,
    Name_M        char(20)     not null,
    NumMat        int          null,
    constraint PK_СОДЕРЖИТ primary key (TCId, Name_M)
)
go

alter table Contract
    add constraint FK_CONTRACT_ПОДПИСЫВА_CLIENT foreign key (ClientId)
        references Client (ClientId) on delete cascade on update cascade
go

alter table Employer
    add constraint FK_EMPLOYER_RELATIONS_POSITION foreign key (Position)
        references Position (Position)
go

alter table HistoryOfMaterial
    add constraint FK_HISTORYO_ЗАПОЛНЯЕТ_EMPLOYER foreign key (EmployerId)
        references Employer (EmployerId)
go

```

```
alter table HistoryOfMaterial  
    add constraint "FK_HISTORYO_ХРАНИТ_ДА_MATERIAL" foreign key (Name_M)  
        references Materials (Name_M)  
go
```

```
alter table HistoryOfContract  
    add constraint FK_HISTORYO_СОХРАНЯЕТ_CONTRACT foreign key (ContractId)  
        references Contract (ContractId)  
go
```

```
alter table Purchase  
    add constraint FK_PURCHASE_ВНОСИТ_EMPLOYER foreign key (EmployerId)  
        references Employer (EmployerId)  
go
```

```
alter table Purchase  
    add constraint FK_PURCHASE_ВНОСИТСЯ_MATERIAL foreign key (Name_M)  
        references Materials (Name_M)  
go
```

```
alter table TechnicalCard  
    add constraint FK_TECHNICA_СОЗДАЁТ_CONTRACT foreign key (ContractId)  
        references Contract (ContractId) on delete cascade on update cascade  
go
```

```
alter table Оформляет  
    add constraint FK_ОФОРМЛЯЕ_ОФОРМЛЯЕТ_CONTRACT foreign key (ContractId)  
        references Contract (ContractId) on delete cascade on update cascade  
go
```

```
alter table Оформляет
```

```
add constraint FK_ОФОРМЛЯЕ_ОФОРМЛЯЕТ_EMPLOYER foreign key (EmployerId)
references Employer (EmployerId)
go
```

```
alter table Редактирует
add constraint FK_РЕДАКТИР_РЕДАКТИРУ_TECHNICA foreign key (TCId)
references TechnicalCard (TCId) on delete cascade on update cascade
go
```

```
alter table Редактирует
add constraint FK_РЕДАКТИР_РЕДАКТИРУ_EMPLOYER foreign key (EmployerId)
references Employer (EmployerId)
go
```

```
alter table Содержит
add constraint FK_СОДЕРЖИТ_СОДЕРЖИТ_TECHNICA foreign key (TCId)
references TechnicalCard (TCId) on delete cascade on update cascade
go
```

```
alter table Содержит
add constraint FK_СОДЕРЖИТ_СОДЕРЖИТ2_MATERIAL foreign key (Name_M)
references Materials (Name_M)
go
```

Код для вставки данных в бд

```
use Polygraf
go
```

```
Insert into Client values(1,'Никита','Щеглов','8461646138');
Insert into Client values(2,'Роман','Егоров','5164853548');
Insert into Client values(3,'Александр','Петров','61568651332');
Insert into Client values(4,'Анна','Комарова','153548513');
```


Insert into Contract values(1,1,'2023-01-07','2023-10-07');

Insert into Contract values(2,2,'2023-07-07','2023-14-07');

Insert into Contract values(3,3,'2023-13-08','2023-21-08');

Insert into Contract values(4,4,'2023-09-09','2023-14-09');

insert into Position values('Менеджер');

insert into Position values('Работник склада');

insert into Position values('Технолог');

insert into Position values('Бухгалтер');

insert into Employer values(1,'Менеджер','Валерия','Муравьева','jknyjhiosn','bkfgnbjkxfgik@mail.ru');

insert into Employer values(2,'Менеджер','Павел','Швецов','awefkjotjhop','hfbgkbxfkgnfkk@mail.ru');

insert into Employer values(3,'Технолог','Анастасия','Фокина','aweio;fjofu','cfnfcghfcgh@mail.ru');

insert into Employer values(4,'Технолог','Дарина','Панина','awefkjhibugt','xfgxfghfghgr@mail.ru');

insert into Employer values(5,'Технолог','Никита','Терентьев','eakulbgjhbul','hgxfxghfghgv@mail.ru');

insert into Employer values(6,'Работник
склада','Дмитрий','Панкратов','hbfkjhlirjrg','ghiftjgojhg@mail.ru');

insert into Employer values(7,'Бухгалтер','Софья','Николаева','fdfndgnfg','sofya@mail.ru');

insert into HistoryOfContract values(1,1,'2023-01-07','На подписании');

insert into HistoryOfContract values(2,1,'2023-02-07','Тест-печать');

insert into HistoryOfContract values(3,1,'2023-04-07','Проверка');

insert into HistoryOfContract values(4,2,'2023-07-07','На подписании');

insert into HistoryOfContract values(5,2,'2023-10-07','Тест-печать');

insert into HistoryOfContract values(6,2,'2023-12-07','Проверка');

insert into Materials values('Бумага офсет',100);

insert into Materials values('Бумага картон',150);

insert into Materials values('Бумага мел',100);

insert into Materials values('Бумага постерная',50);

insert into Purchase values(1,6,'Бумага постерная',100,'2023-12-07');

insert into TechnicalCard values(1,1,'Какой-то','Кружка',1);

insert into TechnicalCard values(2,2,'Какой-то','Кружка',2);

insert into TechnicalCard values(3,3,'Какой-то','Книга',1);

insert into TechnicalCard values(4,3,'Какой-то','Буклеты',10);

insert into TechnicalCard values(5,4,'Какой-то','Бумага',15);

insert into Содержит values(3,'Бумага офсет',50);

insert into Содержит values(4,'Бумага постерная',50);

insert into Содержит values(5,'Бумага офсет',50);

insert into Редактирует values(1,3);

insert into Редактирует values(2,3);

insert into Редактирует values(3,4);

insert into Редактирует values(4,4);

insert into Редактирует values(5,5);

insert into Оформляет values(1,1);

insert into Оформляет values(2,2);

insert into Оформляет values(3,1);

insert into Оформляет values(4,2);

ПРИЛОЖЕНИЕ Б

Код формы входа

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Configuration;
using System.Data;
using System.Data.SqlClient;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Polygraf
{
    public partial class FormLog : Form
    {
        public FormLog()
        {
            InitializeComponent();

            SqlConnection cnn;

            DataSet ds = new DataSet();
            //SqlDataAdapter dataContract = new SqlDataAdapter();
            SqlDataAdapter dataEmployer = new SqlDataAdapter();
            SqlDataAdapter dataPosition = new SqlDataAdapter();
            private void FormLog_Load(object sender, EventArgs e)
            {
                string str = ConfigurationManager.ConnectionStrings["Polygraf"].ConnectionString;
                cnn = new SqlConnection(str);
                cnn.Open();
            }
        }
    }
}
```

```

dataEmployer.SelectCommand = new SqlCommand("select* from Employer", cnn);

dataEmployer.InsertCommand = new SqlCommand("insert into Employer values(@EmployerId,
@Position, @Name, @Surname, @Password, @Mail)", cnn);

dataEmployer.InsertCommand.Parameters.Add("@EmployerId", SqlDbType.Int, 4, "EmployerId");

dataEmployer.InsertCommand.Parameters.Add("@Position", SqlDbType.NVarChar, 50,
"Position");

dataEmployer.InsertCommand.Parameters.Add("@Name", SqlDbType.NVarChar, 50, "NameE");

dataEmployer.InsertCommand.Parameters.Add("@Surname", SqlDbType.NVarChar, 50,
"SurnameE");

dataEmployer.InsertCommand.Parameters.Add("@Password", SqlDbType.NVarChar, 50,
"Password");

dataEmployer.InsertCommand.Parameters.Add("@Mail", SqlDbType.NVarChar, 50, "Mail");

dataEmployer.Fill(ds, "Employer");

bindingSource1.DataSource = ds.Tables["Emlpoyer"];

dataPosition.SelectCommand = new SqlCommand("select* from Position", cnn);
dataPosition.Fill(ds, "Position");

bindingSource2.DataSource = ds.Tables["Position"];
}

private void button1_Click(object sender, EventArgs e)
{
    SqlCommand cmd = new SqlCommand();

    cmd.Connection = cnn;

    cmd.CommandText = "select count(EmployerId) from Employer where Mail = @Mail and
Password = @Password";

    cmd.Parameters.AddWithValue("@Mail", textBox1.Text);

    cmd.Parameters.AddWithValue("@Password", textBox2.Text);

    Console.WriteLine(cmd.ExecuteScalar().ToString());

    if (Convert.ToInt32(cmd.ExecuteScalar()) != 0)
    {

```

```
this.Visible = false;
```

```
SqlCommand cm = new SqlCommand();
```

```
cm.Connection = cnn;
```

```
cm.CommandText = "select EmployerId from Employer where Mail = @Mail and Password = @Password";
```

```
cm.Parameters.AddWithValue("@Mail", textBox1.Text);
```

```
cm.Parameters.AddWithValue("@Password", textBox2.Text);
```

```
SqlCommand sql = new SqlCommand();
```

```
sql.Connection = cnn;
```

```
sql.CommandText = "select Position from Employer where EmployerId = @EmployerId";
```

```
sql.Parameters.AddWithValue("@EmployerId", cm.ExecuteScalar());
```

```
var temp = sql.ExecuteScalar();
```

```
int i = Convert.ToInt32(cm.ExecuteScalar());
```

```
if (temp.ToString() == "Бухгалтер")
```

```
{
```

```
    FormStatistic formStatistic = new FormStatistic(i);
```

```
    formStatistic.ShowDialog();
```

```
    formStatistic.Close();
```

```
}
```

```
else if (temp.ToString() == "Менеджер")
```

```
{
```

```
    FormContract formContract = new FormContract(i);
```

```
    formContract.ShowDialog();
```

```
    formContract.Close();
```

```
}
```

```
else if (temp.ToString() == "Работник склада")
```

```
{
```

```
    FormMaterial formMaterial = new FormMaterial(i);
```

```
    formMaterial.ShowDialog();
```

```
    formMaterial.Close();
```

```
}
```

```

        else if (temp.ToString() == "Технолог")
        {
            FormTechnicalCard formTechnicalCard = new FormTechnicalCard(i);
            formTechnicalCard.ShowDialog();
            formTechnicalCard.Close();
        }
        this.Close();
    }
    else
    {
        MessageBox.Show("Неправильно введены почта или пароль");
        textBox1.Text = "";
        textBox2.Text = "";
    }
}
}
}

```

Код формы договоров

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Configuration;
using System.Data;
using System.Data.Common;
using System.Data.SqlClient;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using static System.Windows.Forms.VisualStyles.VisualStyleElement;

```

```

namespace Polygraf
{
    public partial class FormContract : Form
    {
        public FormContract(int id)
        {
            InitializeComponent();

            this.id = id;

        }

        int id;

        SqlConnection cnn;

        DataSet ds = new DataSet();

        SqlDataAdapter dataContract = new SqlDataAdapter();
        SqlDataAdapter dataClient = new SqlDataAdapter();
        SqlDataAdapter dataTechnical = new SqlDataAdapter();
        SqlDataAdapter dataHistory = new SqlDataAdapter();
        SqlDataAdapter dataRegister = new SqlDataAdapter();

        private void btnFormTechnicalCard_Click(object sender, EventArgs e)
        {
            FormTechnicalCard form = new FormTechnicalCard(id);

            this.Visible = false;

            form.ShowDialog();

            form.Close();

            this.Close();
        }

        private void btnFormMaterials_Click(object sender, EventArgs e)
        {
            FormMaterial form = new FormMaterial(id);

            this.Visible = false;

            form.ShowDialog();
        }
    }
}

```

```

        form.Close();

        Close();
    }

    private void Form1_Load(object sender, EventArgs e)
    {
        string str = ConfigurationManager.ConnectionStrings["Polygraf"].ConnectionString;
        cnn = new SqlConnection(str);
        cnn.Open();

        dataContract.SelectCommand = new SqlCommand("select* from Contract", cnn);

        dataContract.InsertCommand = new SqlCommand("insert into Contract values(@Id, @ClientId, @DateOfBegin, @DateOfEnd)", cnn);
        dataContract.InsertCommand.Parameters.Add("@Id", SqlDbType.Int, 4, "ContractId");
        dataContract.InsertCommand.Parameters.Add("@ClientId", SqlDbType.Int, 4, "ClientId");
        dataContract.InsertCommand.Parameters.Add("@DateOfBegin", SqlDbType.Date, 3, "DateOfBegin");
        dataContract.InsertCommand.Parameters.Add("@DateOfEnd", SqlDbType.Date, 3, "DateOfEnd");

        dataContract.UpdateCommand = new SqlCommand("update Contract set ClientId = @ClientId, " +
            "DateOfBegin = @DateOfBegin, DateOfEnd = @DateOfEnd where ContractId = @Id", cnn);
        dataContract.UpdateCommand.Parameters.Add("@Id", SqlDbType.Int, 4, "ContractId");
        dataContract.UpdateCommand.Parameters.Add("@ClientId", SqlDbType.Int, 4, "ClientId");
        dataContract.UpdateCommand.Parameters.Add("@DateOfBegin", SqlDbType.Date, 3, "DateOfBegin");
        dataContract.UpdateCommand.Parameters.Add("@DateOfEnd", SqlDbType.Date, 3, "DateOfEnd");

        dataContract.DeleteCommand = new SqlCommand("delete from Contract where ContractId = @Id",cnn);
        dataContract.DeleteCommand.Parameters.Add("@Id", SqlDbType.Int, 4, "ContractId");
        dataContract.Fill(ds, "Contract");
        dataClient.SelectCommand = new SqlCommand("select * from Client", cnn);
    }

```



```

dataClient.InsertCommand = new SqlCommand("insert into Client values(@ClientId,@Name,
@Surname, @Phone)", cnn);

dataClient.InsertCommand.Parameters.Add("@ClientId", SqlDbType.Int, 4, "ClientId");

dataClient.InsertCommand.Parameters.Add("@Name", SqlDbType.NVarChar, 50, "NameC");

dataClient.InsertCommand.Parameters.Add("@Surname", SqlDbType.NVarChar, 50, "SurnameC");

dataClient.InsertCommand.Parameters.Add("@Phone", SqlDbType.NVarChar, 50, "Phone");


dataClient.UpdateCommand = new SqlCommand("update Client set NameC = @Name, SurnameC
= @Surname, Phone = @Phone where ClientId = @ClientId", cnn);

dataClient.UpdateCommand.Parameters.Add("@ClientId", SqlDbType.Int, 4, "ClientId");

dataClient.UpdateCommand.Parameters.Add("@Name", SqlDbType.NVarChar, 50, "NameC");

dataClient.UpdateCommand.Parameters.Add("@Surname",      SqlDbType.NVarChar,      50,
"SurnameC");

dataClient.UpdateCommand.Parameters.Add("@Phone", SqlDbType.NVarChar, 50, "Phone");

dataClient.DeleteCommand = new SqlCommand("delete from Client where ClientId = @ClientId",
cnn);

dataClient.DeleteCommand.Parameters.Add("@ClientId", SqlDbType.Int, 4, "ClientId");

dataClient.Fill(ds, "Client");

dataTechnical.SelectCommand = new SqlCommand("select * from TechnicalCard",cnn);

dataTechnical.DeleteCommand = new SqlCommand("delete from TechnicalCard where TCId =
@TCId", cnn);

dataTechnical.DeleteCommand.Parameters.Add("@TCId", SqlDbType.Int, 4, "TCId");

dataTechnical.Fill(ds, "TechnicalCard");

dataHistory.SelectCommand = new SqlCommand("select * from HistoryOfContract", cnn);

dataHistory.InsertCommand = new SqlCommand("insert into HistoryOfContract
values(@Id_Status, @ContractId, @Data_Change, @Status)", cnn);

dataHistory.InsertCommand.Parameters.Add("@Id_Status", SqlDbType.Int, 4, "Id_Status");

dataHistory.InsertCommand.Parameters.Add("@ContractId", SqlDbType.Int, 4, "ContractId");

dataHistory.InsertCommand.Parameters.Add("@Data_Change",      SqlDbType.DateTime,      1,
"Data_Change");

dataHistory.InsertCommand.Parameters.Add("@Status", SqlDbType.VarChar, 50, "Status");

dataHistory.Fill(ds, "HistoryOfContract");

dataRegister.SelectCommand = new SqlCommand("select * from Оформляет", cnn);

```

```

dataRegister.InsertCommand = new SqlCommand("insert into Оформляет values(@ContractId,
@EmployerId)", cnn);

dataRegister.InsertCommand.Parameters.Add("@ContractId", SqlDbType.Int, 4, "ContractId");
dataRegister.InsertCommand.Parameters.Add("@EmployerId", SqlDbType.Int, 4, "EmployerId");
dataRegister.Fill(ds, "Оформляет");


bindingSource1.DataSource = ds.Tables["Contract"];
bindingSource2.DataSource = ds.Tables["Client"];
bindingSource3.DataSource = ds.Tables["TechnicalCard"];
bindingSource4.DataSource = ds.Tables["HistoryOfContract"];


textBox5.DataBindings.Add("text", bindingSource2, "NameC");
textBox6.DataBindings.Add("text", bindingSource2, "SurnameC");
textBox3.DataBindings.Add("text", bindingSource2, "Phone");
dataGridView1.DataSource = bindingSource1;
dataGridView2.DataSource = bindingSource4;
dataGridView1.AutoSizeColumns();
}

private void btnCreate_Click(object sender, EventArgs e)
{
    if (check())
    {
        DataRow rowClient = ds.Tables["Client"].NewRow();

        SqlCommand commandClient = new SqlCommand();
        commandClient.Connection = cnn;
        commandClient.CommandText = "select ClientId from Client order by ClientId desc";

        SqlCommand commandContract = new SqlCommand();
        commandContract.Connection = cnn;
        commandContract.CommandText = "select ContractId from Contract order by ContractId desc";
    }
}

```

```

int idClient = Convert.ToInt32(commandClient.ExecuteScalar()) + 1;
Console.WriteLine(commandClient.ExecuteScalar());
rowClient[0] = idClient;
rowClient[1] = textBox5.Text;
rowClient[2] = textBox6.Text;
rowClient[3] = textBox3.Text;
ds.Tables["Client"].Rows.Add(rowClient);
if (ds.Tables["Client"].GetChanges(DataRowState.Added) != null)
    dataClient.Update(ds.Tables["Client"]);
DataRow rowContract = ds.Tables["Contract"].NewRow();
rowContract[0] = Convert.ToInt32(commandContract.ExecuteScalar()) + 1;
rowContract[1] = Convert.ToInt32(commandClient.ExecuteScalar());
rowContract[2] = dateTimePicker1.Value;
rowContract[3] = dateTimePicker2.Value;
ds.Tables["Contract"].Rows.Add(rowContract);
if (ds.Tables["Contract"].GetChanges(DataRowState.Added) != null)
    dataContract.Update(ds.Tables["Contract"]);
DataRow rowReg = ds.Tables["Оформляет"].NewRow();
rowReg[0] = Convert.ToInt32(commandContract.ExecuteScalar());
rowReg[1] = id;
ds.Tables["Оформляет"].Rows.Add(rowReg);
if (ds.Tables["Оформляет"].GetChanges(DataRowState.Added) != null)
    dataRegister.Update(ds.Tables["Оформляет"]);

bindingSource1.MoveFirst();
bindingSource1.MoveLast();
textBox5.DataBindings.Add("text", bindingSource2, "NameC");
textBox6.DataBindings.Add("text", bindingSource2, "SurnameC");
textBox3.DataBindings.Add("text", bindingSource2, "Phone");
}

```

```
}
```

```
private void btnDelete_Click(object sender, EventArgs e)
{
    if (check())
    {
        try
        {
            bindingSource1.RemoveCurrent();
            if (ds.Tables["Contract"].GetChanges(DataRowState.Deleted) != null)
                dataContract.Update(ds.Tables["Contract"]);
        }
        catch (System.Data.SqlClient.SqlException exp)
        {
            Console.WriteLine(exp.Message);
            ds.Clear();
            dataContract.Fill(ds, "Contract");
            dataHistory.Fill(ds, "HistoryOfContract");
            dataClient.Fill(ds, "Client");
            dataTechnical.Fill(ds, "TechnicalCard");
            dataRegister.Fill(ds, "Оформляет");
            bindingSource1.DataSource = ds.Tables["Contract"];
            bindingSource2.DataSource = ds.Tables["Client"];
            bindingSource3.DataSource = ds.Tables["TechnicalCard"];
            bindingSource4.DataSource = ds.Tables["HistoryOfContract"];
            MessageBox.Show("Завершите все технические карты данного контракта!");
        }
    }
}
```

```
private void btnChange_Click(object sender, EventArgs e)
{

```

```

        dataContract.Update(ds.Tables["Contract"]);
    }

    private void FormContract_FormClosing(object sender, FormClosingEventArgs e)
    {
        bindingSource1.EndEdit();
        if (ds.Tables["Contract"].GetChanges() != null)
            dataContract.Update(ds.Tables["Contract"]);
        if (ds.Tables["HistoryOfContract"].GetChanges() != null)
            dataHistory.Update(ds.Tables["HistoryOfContract"]);
    }

    private void dataGridView1_Click(object sender, EventArgs e)
    {
        bindingSource2.Position = bindingSource1.Position;

        int         rowIndex         =         Convert.ToInt32(dataGridView1[0,Convert.ToInt32(
dataGridView1.CurrentRow.RowIndex)].Value);

        DataTable dataTableHistory = ds.Tables["HistoryOfContract"];

        DataView dataViewHistory = new DataView(dataTableHistory, $"ContractId = {rowIndex}",
"ContractId", DataViewRowState.CurrentRows);

        dataGridView2.DataSource = dataViewHistory;
    }

    private void btnClear_Click(object sender, EventArgs e)
    {
        textBox5.DataBindings.Clear();
        textBox6.DataBindings.Clear();
        textBox3.DataBindings.Clear();
        textBox5.Text = string.Empty;
        textBox6.Text = string.Empty;
        textBox3.Text = string.Empty;
        dateTimePicker1.Value = DateTime.Now;
        dateTimePicker2.Value = DateTime.Now;
    }

```

```

    }

    private void button1_Click(object sender, EventArgs e)
    {
        if (check())
        {
            SqlCommand commandHistory = new SqlCommand();
            commandHistory.Connection = cnn;
            commandHistory.CommandText = "select Id_Status from HistoryOfContract order by Id_Status
desc";
            Console.WriteLine(commandHistory.ExecuteScalar());
            DataRow rowHistory = ds.Tables["HistoryOfContract"].NewRow();
            rowHistory[0] = Convert.ToInt32(commandHistory.ExecuteScalar()) + 1;
            rowHistory[1] = Convert.ToInt32(dataGridView1[0,
Convert.ToInt32(dataGridView1.CurrentCell.RowIndex)].Value);
            rowHistory[2] = DateTime.Now;
            rowHistory[3] = comboBox1.Text;
            ds.Tables["HistoryOfContract"].Rows.Add(rowHistory);
            if (ds.Tables["HistoryOfContract"].GetChanges(DataRowState.Added) != null)
                dataHistory.Update(ds.Tables["HistoryOfContract"]);
        }
    }

    private void button2_Click(object sender, EventArgs e)
    {
        DataTable dataTableContract = ds.Tables["Contract"];
        DataView dataViewContract = new DataView(dataTableContract, $"ContractId = {textBox1.Text}",
"ContractId", DataViewRowState.CurrentRows);
        DataTable dataTableHistory = ds.Tables["HistoryOfContract"];
        DataView dataViewHistory = new DataView(dataTableHistory, $"ContractId = {textBox1.Text}",
"ContractId", DataViewRowState.CurrentRows);
        dataGridView1.DataSource = dataViewContract;
    }

```

```

        dataGridView2.DataSource = dataViewHistory;
    }

    private void pictureBox1_Click(object sender, EventArgs e)
    {
        this.Visible = false;
        FormLog formLog = new FormLog();
        formLog.ShowDialog();
        this.Close();
    }

    private void button3_Click(object sender, EventArgs e)
    {
        this.Visible = false;
        FormStatistic form = new FormStatistic(id);
        form.ShowDialog();
        form.Close();
        Close();
    }

    private bool check()
    {
        SqlCommand sql = new SqlCommand();
        sql.Connection = cnn;
        sql.CommandText = "select Position from Employer where EmployerId = @EmployerId";
        sql.Parameters.AddWithValue("@EmployerId", id);
        if(sql.ExecuteScalar().ToString() == "Менеджер")
            return true;
        else
        {
            MessageBox.Show("Вы не уполномочены");return false;}}}}

```

Код формы материалов

```
namespace Polygraf
{
    public partial class FormMaterial : Form
    {
        public FormMaterial(int id)
        {
            InitializeComponent();

            this.id = id;
        }

        int id;

        SqlConnection cnn;

        DataSet ds = new DataSet();

        SqlDataAdapter dataMaterial = new SqlDataAdapter();
        SqlDataAdapter dataPurchase = new SqlDataAdapter();
        SqlDataAdapter dataHistory = new SqlDataAdapter();

        private void button4_Click(object sender, EventArgs e)
        {
            if (check())
            {
                SqlCommand commandPurchase = new SqlCommand();

                commandPurchase.Connection = cnn;

                commandPurchase.CommandText = "select PurchaseId from Purchase order by PurchaseId
desc";

                DataRow rowPurchase = ds.Tables["Purchase"].NewRow();

                rowPurchase[0] = Convert.ToInt32(commandPurchase.ExecuteScalar()) + 1;

                rowPurchase[1] = id;

                rowPurchase[2] = comboBox1.Text;

                rowPurchase[3] = textBox2.Text;

                rowPurchase[4] = DateTime.Now;

                ds.Tables["Purchase"].Rows.Add(rowPurchase);
            }
        }
    }
}
```



```

        if (ds.Tables["Purchase"].GetChanges(DataRowState.Added) != null)
        {
            dataPurchase.Update(ds.Tables["Purchase"]);
            bindingSource2.MoveFirst();
            bindingSource2.MoveLast();
        }
    }

    private void FormMaterial_Load(object sender, EventArgs e)
    {
        string str = ConfigurationManager.ConnectionStrings["Polygraf"].ConnectionString;
        cnn = new SqlConnection(str);
        cnn.Open();
        dataMaterial.SelectCommand = new SqlCommand("select* from Materials", cnn);
        dataMaterial.InsertCommand = new SqlCommand("insert into Materials values(@Name_M,
@Quantity_M)",cnn);
        dataMaterial.InsertCommand.Parameters.Add("@Name_M", SqlDbType.Char, 20, "Name_M");
        dataMaterial.InsertCommand.Parameters.Add("@Quantity_M", SqlDbType.Int, 4, "Quantity_M");

        dataMaterial.UpdateCommand = new SqlCommand("update Materials set Quantity_M =
@Quantity_M where Name_M = @Name_M", cnn);
        dataMaterial.UpdateCommand.Parameters.Add("@Name_M", SqlDbType.Char, 20, "Name_M");
        dataMaterial.UpdateCommand.Parameters.Add("Quantity_M", SqlDbType.Int, 4, "Quantity_M");

        dataMaterial.Fill(ds, "Materials");
        bindingSource1.DataSource = ds.Tables["Materials"];
        dataPurchase.SelectCommand = new SqlCommand("select* from Purchase", cnn);

        dataPurchase.InsertCommand = new SqlCommand("insert into Purchase values(@PurchaseId,
@EmployerId, @Name_M, @QuantityPur, @DateOfPur)",cnn);
        dataPurchase.InsertCommand.Parameters.Add("@PurchaseId", SqlDbType.Int, 4, "PurchaseId");
        dataPurchase.InsertCommand.Parameters.Add("@EmployerId", SqlDbType.Int, 4, "EmployerId");
        dataPurchase.InsertCommand.Parameters.Add("@Name_M", SqlDbType.Char, 20, "Name_M");
    }
}

```

```

        dataPurchase.InsertCommand.Parameters.Add("@QuantityPur", SqlDbType.Int, 4,
"QuantityPur");

        dataPurchase.InsertCommand.Parameters.Add("@DateOfPur", SqlDbType.DateTime, 4,
"DateOfPur");

        dataPurchase.UpdateCommand = new SqlCommand("update Materials set EmployerId =
@EmployerId, Name_M = @Name_M, QuantityPur = @QuantityPur, DateOfPur = @DateOfPur where
PurchasId = @PurchasId", cnn);

        dataPurchase.UpdateCommand.Parameters.Add("@PurchasId", SqlDbType.Int, 4, "PurchasId");

        dataPurchase.UpdateCommand.Parameters.Add("@EmployerId", SqlDbType.Int, 4,
"EmployerId");

        dataPurchase.UpdateCommand.Parameters.Add("@Name_M", SqlDbType.Char, 20, "Name_M");

        dataPurchase.UpdateCommand.Parameters.Add("@QuantityPur", SqlDbType.Int, 4,
"QuantityPur");

        dataPurchase.UpdateCommand.Parameters.Add("@DateOfPur", SqlDbType.DateTime, 4,
"DateOfPur");

        dataPurchase.DeleteCommand = new SqlCommand("delete Purchase where PurchasId =
@PurchasId",cnn);

        dataPurchase.DeleteCommand.Parameters.Add("@PurchasId", SqlDbType.Int, 4, "PurchasId");

        dataPurchase.Fill(ds, "Purchase");

        bindingSource2.DataSource = ds.Tables["Purchase"];

        dataHistory.SelectCommand = new SqlCommand("select* from HistoryOfMaterial", cnn);

        dataHistory.InsertCommand = new SqlCommand("insert into HistoryOfMaterial
values(@Id_History, @Name_M, @EmployerId, @Quantity, @Data_change_Quantity)", cnn);

        dataHistory.InsertCommand.Parameters.Add("@Id_History", SqlDbType.Int, 4, "Id_History");

        dataHistory.InsertCommand.Parameters.Add("@Name_M", SqlDbType.Char, 20, "Name_M");

        dataHistory.InsertCommand.Parameters.Add("@EmployerId", SqlDbType.Int, 4, "EmployerId");

        dataHistory.InsertCommand.Parameters.Add("@Quantity", SqlDbType.Int, 4, "Quantity");

        dataHistory.InsertCommand.Parameters.Add("@Data_change_Quantity", SqlDbType.DateTime,
4, "Data_change_Quantity");

        dataHistory.Fill(ds, "HistoryOfMaterial");

        dataGridView1.DataSource = bindingSource1;

        dataGridView1.AutoSizeColumns();

        dataGridView2.DataSource = bindingSource2;

        dataGridView2.AutoSizeColumns();

```

```

        comboBox1.DataSource = ds.Tables["Materials"];
        comboBox1.DisplayMember = "Name_M";
        comboBox1.ValueMember = "Name_M";
    }

    private void button5_Click(object sender, EventArgs e)
    {
        if(check())
            ffff(dataGridView2.CurrentRow.RowIndex);
    }

    private void button8_Click(object sender, EventArgs e)
    {
        if(check())
            for(int i = 0; i < dataGridView2.Rows.Count; i++)
                ffff(i);
    }

    private void ffff(int rowIndex)
    {
        DataTable dtM = ds.Tables["Materials"];
        DataView dataViewM = new DataView(dtM);
        var num = Convert.ToInt32(dataGridView2[3, rowIndex].Value);
        var temp = dataGridView2[2, rowIndex].Value;
        int count = 0;
        foreach (DataRow row in dtM.Rows)
        {
            if (row[0].ToString() == temp.ToString())
                break;
            count++;
        }
        num += Convert.ToInt32(dataGridView1[1, count].Value);
        dataViewM[count][1] = num;
    }

```

```

DataTable dtP = ds.Tables["Purchase"];
DataView dataViewP = new DataView(dtP);
dataViewP[rowIndex].Row.Delete();

SqlCommand commandHistory = new SqlCommand();
commandHistory.Connection = cnn;
commandHistory.CommandText = "select Id_History from HistoryOfMaterial order by Id_History
desc";
DataRow rowHistory = ds.Tables["HistoryOfMaterial"].NewRow();
rowHistory[0] = Convert.ToInt32(commandHistory.ExecuteScalar()) + 1;
rowHistory[1] = dataGridView1[0, count].Value;
rowHistory[2] = id;
rowHistory[3] = Convert.ToInt32(dataGridView2[3, rowIndex].Value);
rowHistory[4] = DateTime.Now;

ds.Tables["HistoryOfMaterial"].Rows.Add(rowHistory);
if (ds.Tables["HistoryOfMaterial"].GetChanges(DataRowState.Added) != null)
    dataHistory.Update(ds.Tables["HistoryOfMaterial"]);
if (ds.Tables["Materials"].GetChanges(DataRowState.Modified) != null)
    dataMaterial.Update(ds.Tables["Materials"]);
if (ds.Tables["Purchase"].GetChanges(DataRowState.Deleted) != null)
    dataPurchase.Update(ds.Tables["Purchase"]);
bindingSource2.MoveFirst();
bindingSource2.MoveLast();
}

private void button1_Click(object sender, EventArgs e)
{
    FormContract formContract = new FormContract(id);
    Visible = false;

```

```

        formContract.ShowDialog();
        formContract.Close();
        Close();
    }

```

```

private void button2_Click(object sender, EventArgs e)
{
    FormTechnicalCard form = new FormTechnicalCard(id);
    this.Visible = false;
    form.ShowDialog();
    form.Close();
    this.Close();
}

```

```

private void button6_Click(object sender, EventArgs e)
{
    this.Visible = false;
    FormStatistic form = new FormStatistic(id);
    form.ShowDialog();
    form.Close();
    Close();
}

```

```

private bool check()
{
    SqlCommand sql = new SqlCommand();
    sql.Connection = cnn;
    sql.CommandText = "select Position from Employer where EmployerId = @EmployerId";
    sql.Parameters.AddWithValue("@EmployerId", id);

    if (sql.ExecuteScalar().ToString() == "Работник склада")
        return true;
    else

```

```

        {
            MessageBox.Show("Вы не уполномочены");
            return false;
        }

    }

    private void pictureBox1_Click(object sender, EventArgs e)
    {
        this.Visible = false;
        FormLog formLog = new FormLog();
        formLog.ShowDialog();
        this.Close();
    }
}

```

Код формы статистики

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Configuration;
using System.Data;
using System.Data.SqlClient;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Polygraf
{
    public partial class FormStatistic : Form

```

```

{
    public FormStatistic(int id)
    {
        InitializeComponent();

        this.id = id;
    }

    int id;

    SqlConnection cnn;

    DataSet ds = new DataSet();

    SqlDataAdapter dataContract = new SqlDataAdapter();
    SqlDataAdapter dataMaterial = new SqlDataAdapter();
    SqlDataAdapter dataClient = new SqlDataAdapter();
    SqlDataAdapter dataTechnical = new SqlDataAdapter();
    SqlDataAdapter dataHistoryOfMaterial = new SqlDataAdapter();
    SqlDataAdapter dataEmployer = new SqlDataAdapter();

    private void FormStatistic_Load(object sender, EventArgs e)
    {
        string str = ConfigurationManager.ConnectionStrings["Polygraf"].ConnectionString;
        cnn = new SqlConnection(str);

        cnn.Open();

        dataEmployer.SelectCommand = new SqlCommand("select* from Employer", cnn);
        dataEmployer.Fill(ds, "Employer");

        dataHistoryOfMaterial.SelectCommand = new SqlCommand("select* from HistoryOfMaterial",
cnn);
        dataHistoryOfMaterial.Fill(ds, "HistoryOfMaterial");

        bindingSource1.DataSource = ds.Tables["HistoryOfMaterial"];
        dataGridView1.DataSource = bindingSource1;
        dataGridView1.AutoSizeColumns();

        dataMaterial.SelectCommand = new SqlCommand("select* from Materials", cnn);
        dataMaterial.Fill(ds, "Materials");
    }
}

```

```

        comboBox1.DataSource = ds.Tables["Materials"];
        comboBox1.DisplayMember = "Name_M";
        comboBox1.ValueMember = "Name_M";
    }

```

```

private void btnFormContract_Click(object sender, EventArgs e)
{
    FormContract formContract = new FormContract(id);
    Visible = false;
    formContract.ShowDialog();
    formContract.Close();
    Close();
}

```

```

private void btnFormTechnicalCard_Click(object sender, EventArgs e)
{
    FormTechnicalCard form = new FormTechnicalCard(id);
    this.Visible = false;
    form.ShowDialog();
    form.Close();
    this.Close();
}

```

```

private void btnFormMaterials_Click(object sender, EventArgs e)
{
    FormMaterial form = new FormMaterial(id);
    this.Visible = false;
    form.ShowDialog();
    form.Close();
    Close();
}

```



```

private void button1_Click(object sender, EventArgs e)
{
    if (check())
    {
        DataTable dtE = ds.Tables["Employer"];
        DataView dataViewE = new DataView(dtE);
        int temp;
        int count = 0;
        foreach (DataRow row in dtE.Rows)
        {
            if (Convert.ToInt32(row[0]) == Convert.ToInt32(textBox1.Text))
            {
                break;
                count++;
            }
            textBox2.Text = ds.Tables["Employer"].Rows[count][2].ToString();
            textBox3.Text = ds.Tables["Employer"].Rows[count][3].ToString();
            textBox4.Text = ds.Tables["Employer"].Rows[count][1].ToString();

            if (ds.Tables["Employer"].Rows[count][1].ToString() == "Менеджер")
            {
                label5.Text = "Количество договоров";
                SqlCommand cmd = new SqlCommand();
                cmd.Connection = cnn;
                cmd.CommandText = "select count(ContractId) from Оформляет where EmployerId = @EmployerId";
                cmd.Parameters.AddWithValue("@EmployerId", textBox1.Text);
                textBox5.Text = cmd.ExecuteScalar().ToString();
            }
            else if (ds.Tables["Employer"].Rows[count][1].ToString() == "Технолог")
            {
                label5.Text = "Количество техкарт";
                SqlCommand cmd = new SqlCommand();

```

```

        cmd.Connection = cnn;

        cmd.CommandText = "select count(TCId) from Редактирует where EmployerId =
@EmployerId";

        cmd.Parameters.AddWithValue("@EmployerId", textBox1.Text);

        textBox5.Text = cmd.ExecuteScalar().ToString();

    }

}

}

```

```

private void button2_Click(object sender, EventArgs e)
{
    if (check())
    {
        string name = comboBox1.Text;

        DataTable dtM = ds.Tables["Materials"];

        int temp;

        int count = 0;

        foreach (DataRow row in dtM.Rows)
        {
            if (row[0].ToString() == name)
            {
                break;

                count++;
            }
        }

        textBox6.Text = ds.Tables["Materials"].Rows[count][1].ToString();

        DataTable dtH = ds.Tables["HistoryOfMaterial"];

        int num1 = 0;

        int num2 = 0;

        foreach (DataRow row in dtH.Rows)
        {
            if (row[1].ToString() == name && Convert.ToInt32(row[3]) > 0 &&
Convert.ToDateTime(row[4]) > DateTime.Today.AddMonths(-1))

                num1 += Convert.ToInt32(row[3]);

```

```

        if (row[1].ToString() == name && Convert.ToInt32(row[3]) < 0 &&
Convert.ToDateTime(row[4]) > DateTime.Today.AddMonths(-1))

            num2 += Convert.ToInt32(row[3]);

        }

        textBox7.Text = num1.ToString();

        textBox8.Text = num2.ToString();

    }

}

private bool check()
{
    SqlCommand sql = new SqlCommand();

    sql.Connection = cnn;

    sql.CommandText = "select Position from Employer where EmployerId = @EmployerId";
    sql.Parameters.AddWithValue("@EmployerId", id);

    if (sql.ExecuteScalar().ToString() == "Бухгалтер")
        return true;

    else
    {
        MessageBox.Show("Вы не уполномочены");

        return false;

    }

}

private void pictureBox1_Click(object sender, EventArgs e)
{
    this.Visible = false;

    FormLog formLog = new FormLog();

    formLog.ShowDialog();

    this.Close();

}

}}

```

Код формы технических карт

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Configuration;
using System.Data;
using System.Data.SqlClient;
using System.Drawing;
using System.Linq;
using System.Runtime.InteropServices.ComTypes;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Polygraf
{
    public partial class FormTechnicalCard : Form
    {
        public FormTechnicalCard(int id)
        {
            InitializeComponent();
            this.id = id;
        }

        int id;

        SqlConnection cnn;

        DataSet ds = new DataSet();

        SqlDataAdapter dataTechnicalCard = new SqlDataAdapter();
        SqlDataAdapter dataEdit = new SqlDataAdapter();
        SqlDataAdapter dataInclude = new SqlDataAdapter();
        SqlDataAdapter dataMaterial = new SqlDataAdapter();
        SqlDataAdapter dataHistoryOfMaterial = new SqlDataAdapter();
```

```

private void FormTechnicalCard_Load(object sender, EventArgs e)
{
    string str = ConfigurationManager.ConnectionStrings["Polygraf"].ConnectionString;
    cnn = new SqlConnection(str);
    cnn.Open();

    dataTechnicalCard.SelectCommand = new SqlCommand("select* from TechnicalCard",cnn)

    dataTechnicalCard.InsertCommand = new SqlCommand("insert into TechnicalCard values(@TCId,
@ContractId, @TypePrint, @Object, @Num)",cnn);

    dataTechnicalCard.InsertCommand.Parameters.Add("@TCId", SqlDbType.Int, 4, "TCId");

    dataTechnicalCard.InsertCommand.Parameters.Add("@ContractId",      SqlDbType.Int,      4,
"ContractId");

    dataTechnicalCard.InsertCommand.Parameters.Add("@TypePrint",      SqlDbType.VarChar,      50,
"TypePrint");

    dataTechnicalCard.InsertCommand.Parameters.Add("@Object",      SqlDbType.VarChar,      50,
"Object");

    dataTechnicalCard.InsertCommand.Parameters.Add("@Num", SqlDbType.Int, 4, "Num");


    dataTechnicalCard.UpdateCommand = new SqlCommand("update TechnicalCard set ContractId =
@ContractId, TypePrint = @TypePrint, Object = @Object, Num = @Num where TCId = @TCId", cnn);

    dataTechnicalCard.UpdateCommand.Parameters.Add("@TCId", SqlDbType.Int, 4, "TCId");

    dataTechnicalCard.UpdateCommand.Parameters.Add("@ContractId",      SqlDbType.Int,      4,
"ContractId");

    dataTechnicalCard.UpdateCommand.Parameters.Add("@TypePrint",      SqlDbType.VarChar,      50,
"TypePrint");

    dataTechnicalCard.UpdateCommand.Parameters.Add("@Object",      SqlDbType.VarChar,      50,
"Object");

    dataTechnicalCard.UpdateCommand.Parameters.Add("@Num", SqlDbType.Int, 4, "Num");


    dataTechnicalCard.DeleteCommand = new SqlCommand("delete from TechnicalCard where TCId =
@TCId", cnn);

    dataTechnicalCard.DeleteCommand.Parameters.Add("@TCId", SqlDbType.Int, 4, "TCId");


    dataTechnicalCard.Fill(ds, "TechnicalCard");


    bindingSource1.DataSource = ds.Tables["TechnicalCard"];
}

```

```

dataMaterial.SelectCommand = new SqlCommand("select* from Materials",cnn);
dataMaterial.Fill(ds, "Materials");
bindingSource3.DataSource = ds.Tables["Materials"];
comboBox3.DataSource = bindingSource3;
comboBox3.DisplayMember = "Name_M";
comboBox3.ValueMember = "Name_M";
dataInclude.SelectCommand = new SqlCommand("select* from Содержит", cnn);

dataInclude.InsertCommand = new SqlCommand("insert into Содержит values(@TCId,
@Name_M, @NumMat)", cnn);
dataInclude.InsertCommand.Parameters.Add("@TCId", SqlDbType.Int, 4, "TCId");
dataInclude.InsertCommand.Parameters.Add("@Name_M", SqlDbType.Char, 20, "Name_M");
dataInclude.InsertCommand.Parameters.Add("@NumMat", SqlDbType.Int, 4, "NumMat");

dataInclude.UpdateCommand = new SqlCommand("update Содержит set NumMat = @NumMat
where TCId = @TCId and Name_M = @Name_M", cnn);
dataInclude.UpdateCommand.Parameters.Add("@TCId", SqlDbType.Int, 4, "TCId");
dataInclude.UpdateCommand.Parameters.Add("@Name_M", SqlDbType.Char, 20, "Name_M");
dataInclude.UpdateCommand.Parameters.Add("@NumMat", SqlDbType.Int, 4, "NumMat");

dataInclude.DeleteCommand = new SqlCommand("delete Содержит where TCId = @TCId and
Name_M = @Name_M",cnn);
dataInclude.DeleteCommand.Parameters.Add("@TCId", SqlDbType.Int, 4, "TCId");
dataInclude.DeleteCommand.Parameters.Add("@Name_M", SqlDbType.Char, 20, "Name_M");

dataInclude.Fill(ds, "Содержит");

bindingSource2.DataSource = ds.Tables["Содержит"];

dataEdit.SelectCommand = new SqlCommand("select* from Редактирует", cnn);
dataEdit.InsertCommand = new SqlCommand("insert into Редактирует values(@TCId,
@EmployerId");
dataEdit.InsertCommand.Parameters.Add("@TCId", SqlDbType.Int, 4, "TCId");

```

```

dataEdit.InsertCommand.Parameters.Add("@EmployerId", SqlDbType.Int, 4, "EmployerId");
dataEdit.Fill(ds, "Редактирует");

dataHistoryOfMaterial.SelectCommand = new SqlCommand("select* from HistoryOfMaterial",
cnn);

dataHistoryOfMaterial.InsertCommand = new SqlCommand("insert into HistoryOfMaterial
values(@Id_History, @Name_M, @EmployerId, @Quantity, @Data_change_Quantity)", cnn);

dataHistoryOfMaterial.InsertCommand.Parameters.Add("@Id_History",    SqlDbType.Int,    4,
"Id_History");

dataHistoryOfMaterial.InsertCommand.Parameters.Add("@Name_M",    SqlDbType.Char,    20,
"Name_M");

dataHistoryOfMaterial.InsertCommand.Parameters.Add("@EmployerId",    SqlDbType.Int,    4,
"EmployerId");

dataHistoryOfMaterial.InsertCommand.Parameters.Add("@Quantity",    SqlDbType.Int,    4,
"Quantity");

dataHistoryOfMaterial.InsertCommand.Parameters.Add("@Data_change_Quantity",
SqlDbType.DateTime, 4, "Data_change_Quantity");

dataHistoryOfMaterial.Fill(ds, "HistoryOfMaterial");

dataGridView1.DataSource = bindingSource1;
dataGridView1.AutoSizeColumns();
dataGridView2.DataSource = bindingSource2;
dataGridView2.AutoSizeColumns();
}

private void btnCreate_Click(object sender, EventArgs e)
{
    if (check())
    {
        SqlCommand commandTechnicalCard = new SqlCommand();
        commandTechnicalCard.Connection = cnn;
        commandTechnicalCard.CommandText = "select TCId from TechnicalCard order by TCId desc";

        DataRow rowTechnicalCard = ds.Tables["TechnicalCard"].NewRow();
        rowTechnicalCard[0] = Convert.ToInt32(commandTechnicalCard.ExecuteScalar()) + 1;

```

```

rowTechnicalCard[1] = Convert.ToInt32(textBox1.Text);
rowTechnicalCard[2] = comboBox1.Text;
rowTechnicalCard[3] = comboBox2.Text;
rowTechnicalCard[4] = numericUpDown1.Value;
ds.Tables["TechnicalCard"].Rows.Add(rowTechnicalCard);
if (ds.Tables["TechnicalCard"].GetChanges(DataRowState.Added) != null)
    dataTechnicalCard.Update(ds.Tables["TechnicalCard"]);
bindingSource1.MoveFirst();
bindingSource1.MoveLast();

DataRow rowEdit = ds.Tables["Редактирует"].NewRow();
rowEdit[0] = Convert.ToInt32(commandTechnicalCard.ExecuteScalar()) + 1;
rowEdit[1] = id;
ds.Tables["Редактирует"].Rows.Add(rowEdit);
if (ds.Tables["Редактирует"].GetChanges(DataRowState.Added) != null)
    dataEdit.Update(ds.Tables["Редактирует"]);
}
}

private void btnDelete_Click(object sender, EventArgs e)
{
    if (check())
    {
        try
        {
            bindingSource1.RemoveCurrent();
            if (ds.Tables["TechnicalCard"].GetChanges(DataRowState.Deleted) != null)
                dataTechnicalCard.Update(ds.Tables["TechnicalCard"]);
        }
        catch (System.Data.SqlClient.SqlException exp)
        {
            Console.WriteLine(exp.Message);
        }
    }
}

```



```

        ds.Clear();

        dataTechnicalCard.Fill(ds, "TechnicalCard");

        bindingSource1.DataSource = ds.Tables["TechnicalCard"];

        MessageBox.Show("Завершите все технические карты данного контракта!");
    }
}

private void btnChange_Click(object sender, EventArgs e)
{
    dataTechnicalCard.Update(ds.Tables["TechnicalCard"]);
}

private void FormTechnicalCard_FormClosing(object sender, FormClosingEventArgs e)
{
    bindingSource1.EndEdit();

    if (ds.Tables["TechnicalCard"].GetChanges() != null)
        dataTechnicalCard.Update(ds.Tables["TechnicalCard"]);
}

private void button1_Click(object sender, EventArgs e)
{
    if (check())
    {
        int                rowIndex                =                Convert.ToInt32(dataGridView1[0,
Convert.ToInt32(dataGridView1.CurrentRow.Index)].Value);

        DataTable dataTable = ds.Tables["Materials"];

        DataView dataView = new DataView(dataTable, $"Name_M = '{comboBox3.Text}'", "Name_M",
DataViewRowState.CurrentRows);

        int num = Convert.ToInt32(dataView[0]["Quantity_M"]);

        if (num - Convert.ToInt32(textBox2.Text) >= 0)
        {

```

```

dataView[0]["Quantity_M"] = num - Convert.ToInt32(textBox2.Text);
DataRow rowInclude = ds.Tables["Содержит"].NewRow();
rowInclude[0] = rowIndex;
rowInclude[1] = comboBox3.Text;
rowInclude[2] = textBox2.Text;

ds.Tables["Содержит"].Rows.Add(rowInclude);
if (ds.Tables["Содержит"].GetChanges(DataRowState.Added) != null)
    dataInclude.Update(ds.Tables["Содержит"]);
bindingSource2.MoveFirst();
bindingSource2.MoveLast();

SqlCommand commandHistory = new SqlCommand();
commandHistory.Connection = cnn;
commandHistory.CommandText = "select Id_History from HistoryOfMaterial order by
Id_History desc";

DataRow rowHistory = ds.Tables["HistoryOfMaterial"].NewRow();
rowHistory[0] = Convert.ToInt32(commandHistory.ExecuteScalar()) + 1;
rowHistory[1] = comboBox3.Text;
rowHistory[2] = id;
rowHistory[3] = Convert.ToInt32(textBox2.Text) * (-1);
rowHistory[4] = DateTime.Now;
ds.Tables["HistoryOfMaterial"].Rows.Add(rowHistory);
if (ds.Tables["HistoryOfMaterial"].GetChanges(DataRowState.Added) != null)
    dataHistoryOfMaterial.Update(ds.Tables["HistoryOfMaterial"]);
}
else
{
    MessageBox.Show("На складе нет столько " + comboBox3.Text + "Запрос сотруднику
склада отправлен");
}
}

```

```
}
```

```
private void button3_Click(object sender, EventArgs e)
{
    if (check())
    {
        try
        {
            bindingSource2.RemoveCurrent();

            if (ds.Tables["Содержит"].GetChanges(DataRowState.Deleted) != null)
                dataInclude.Update(ds.Tables["Содержит"]);
        }
        catch (System.Data.SqlClient.SqlException exp)
        {
            Console.WriteLine(exp.Message);
            ds.Clear();
            dataInclude.Fill(ds, "Содержит");
            bindingSource2.DataSource = ds.Tables["Содержит"];
            MessageBox.Show("Завершите все технические карты данного контракта!");
            dataHistoryOfMaterial.Fill(ds, "HistoryOfMaterial");
            dataTechnicalCard.Fill(ds, "TechnicalCard");
            dataMaterial.Fill(ds, "Materials");
            dataEdit.Fill(ds, "Редактирует");
            bindingSource1.DataSource = ds.Tables["TechnicalCard"];
        }
    }
}
```

```
private void button2_Click(object sender, EventArgs e)
{
```

```

        dataInclude.Update(ds.Tables["Содержит"]);
    }

    private void dataGridView1_Click(object sender, EventArgs e)
    {
        bindingSource2.Position = bindingSource1.Position;

        int                rowIndex                =                Convert.ToInt32(dataGridView1[0,
Convert.ToInt32(dataGridView1.CurrentRow.Index)].Value);

        DataTable dataTable = ds.Tables["Содержит"];

        DataView  dataView  =  new  DataView(dataTable,  $"TCId  =  {rowIndex}",  "TCId",
DataViewRowState.CurrentRows);

        dataGridView2.DataSource = dataView;
    }

    private void button4_Click(object sender, EventArgs e)
    {
        this.Visible = false;

        FormStatistic form = new FormStatistic(id);

        form.ShowDialog();

        form.Close();

        Close();
    }

    private void btnFormContract_Click(object sender, EventArgs e)
    {
        FormContract formContract = new FormContract(id);

        Visible = false;

        formContract.ShowDialog();

        formContract.Close();

        Close();
    }

    private void btnFormMaterials_Click(object sender, EventArgs e)
    {

```

```

        FormMaterial form = new FormMaterial(id);

        this.Visible = false;

        form.ShowDialog();

        form.Close();

        Close();
    }

    private bool check()
    {
        SqlCommand sql = new SqlCommand();

        sql.Connection = cnn;

        sql.CommandText = "select Position from Employer where EmployerId = @EmployerId";
        sql.Parameters.AddWithValue("@EmployerId", id);

        if (sql.ExecuteScalar().ToString() == "Технолог")
            return true;

        else
        {
            MessageBox.Show("Вы не уполномочены");

            return false;
        }
    }

}

private void pictureBox1_Click(object sender, EventArgs e)
{
    this.Visible = false;

    FormLog formLog = new FormLog();

    formLog.ShowDialog();

    this.Close();
}
}
}

```