Министерство образования и науки Кыргызской Республики

Кыргызский государственный технический университет им И.Раззакова

Институт информационных технологий

Кафедра «Программное обеспечение компьютерных систем»

**КУРСОВАЯ РАБОТА**

по дисциплине: «**Система управления базами данных»**

на тему: «Разработка Базы Данных для предприятия по производству кондитерских изделий»

Выполнил: студент группы ИБ-1-22

Омурбеков Эрмек

Проверил: Раматов К.С.

Бишкек – 2023

# **Содержание**

[**Содержание** 2](#_Toc136125093)

[**Описание предметной области** 3](#_Toc136125094)

[**Введение** 3](#_Toc136125095)

[**Спецификация проблемы** 3](#_Toc136125096)

[**Функциональные требования** 3](#_Toc136125097)

[**Глава 1. Серверная часть.** 4](#_Toc136125098)

[**1.1. Таблицы, описание таблиц и назначения** 5](#_Toc136125099)

[**1.2. Хранимые процедуры, описание каждой хранимки:** 8](#_Toc136125100)

[**Глава 2. Клиентская часть** 19](#_Toc136125101)

[**Описание клиентской части** 19](#_Toc136125102)

# **Описание предметной области**

## **Введение**

В современном мире правильная организация данных становится ключом к успешному управлению бизнесом. Это особенно важно для предприятий, производящих потребительские товары, где точный учет ресурсов напрямую связан с эффективностью работы. В моей курсовой работе я сосредоточился на разработке базы данных, которая поможет в учете всех операций на предприятии, занимающемся производством кондитерских изделий. Моя цель - создать систему, которая не только будет хранить данные, но и поможет в их анализе и управлении, улучшая тем самым общую эффективность предприятия.

Для решения задач, поставленных в моей курсовой работе, я выбрал SQL Server в качестве СУБД из-за его высокой производительности, надежности и возможности масштабирования. Это позволяет мне эффективно обрабатывать и хранить большие объемы данных.

Особенностью моего проекта стал выбор Microsoft Access для разработки клиентского приложения. Благодаря его простому и понятному интерфейсу для создания форм и отчетов, я смог быстро и эффективно разработать приложение для работы с базой данных. Это приложение помогает в управлении данными и повседневных операциях на предприятии.

## 

## **Спецификация проблемы**

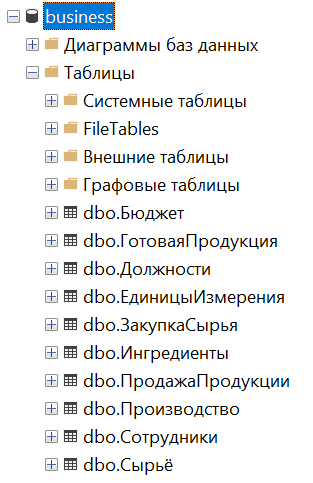
* Учет и контроль запасов.
* Затраты времени.
* Управление производственными процессами.

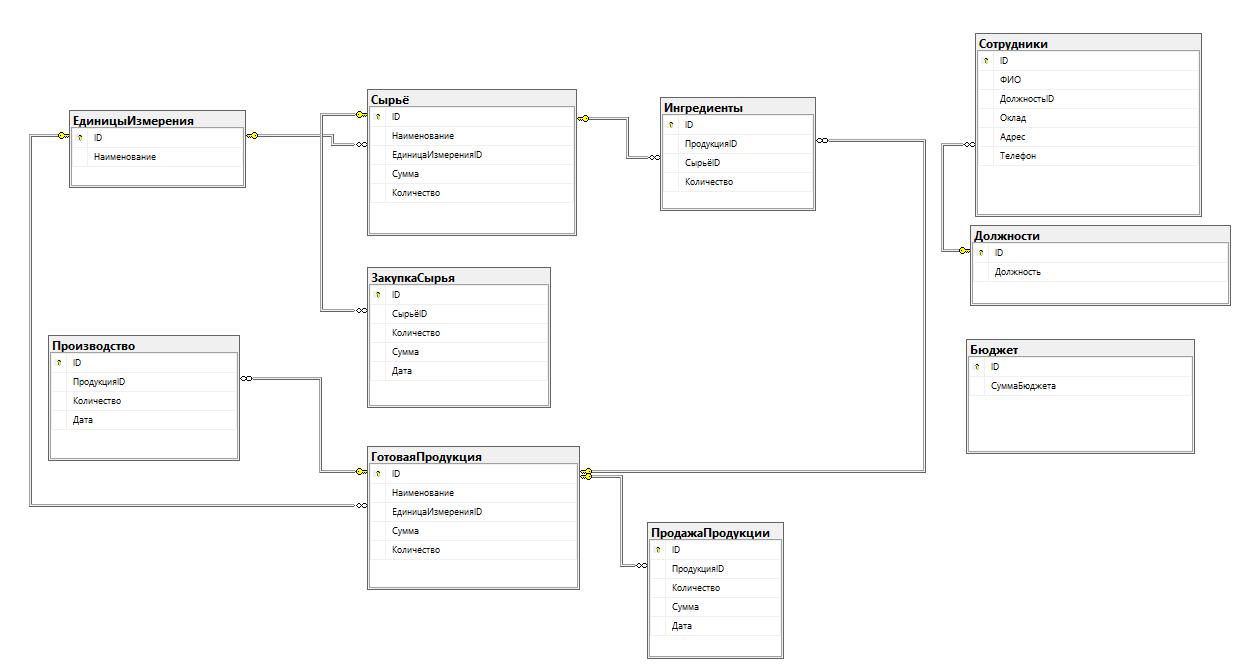
## 

## **Функциональные требования**

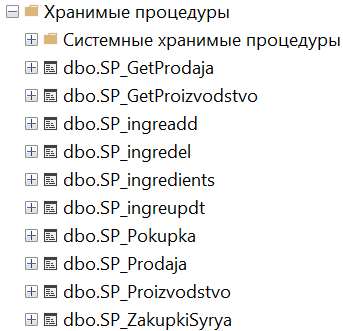
* Управление производственными процессами
* Учет товаров
* Контроль качество
* Учет сотрудников
* Учет продажи товаров
* Учет закупки

# **Глава 1. Серверная часть.**

База данных состоит из 10 таблиц, 1 диаграмма. 10 хранимых процедур, 

*Рис.1. Таблицы БД*

*Рис.2. Диаграмма БД*

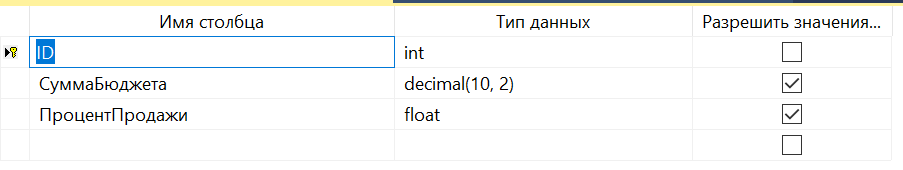


*Рис.3. Хранимые процедуры БД*

## **1.1. Таблицы, описание таблиц и назначения**

**Таблица “Бюджет”**

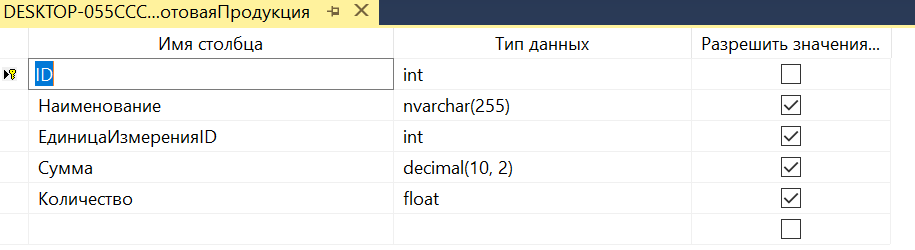
Эта таблица предназначена для отслеживания общего бюджета предприятия и его доступности для различных операций и расходов.



*Рис.4. Структура таблицы «*Бюджет*»*

**Таблица “** **Готовая Продукция”**

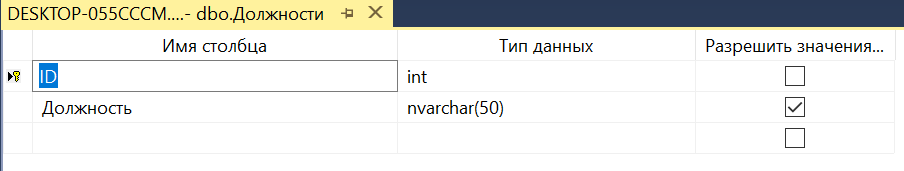
Эта таблица используется для учета всех готовых изделий, их стоимости и наличия на складе.



*Рис.5. Структура таблицы «ГотоваяПродукция»*

**Таблица “** **Должности”**

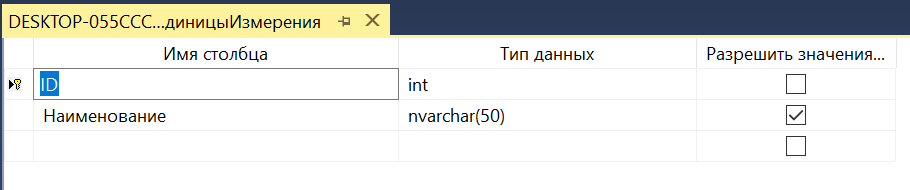
Учитывает различные должности в предприятии и служит для связи с данными сотрудников, определяя их роли и обязанности.



*Рис.6. Структура таблицы «Должности»*

**Таблица “** **Ингредиенты”**

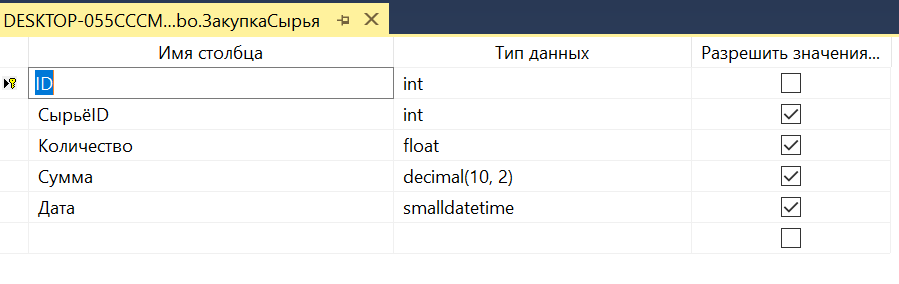
Содержит информацию об единицах измерения, используемых для товаров и материалов, обеспечивая стандартизацию данных по всей базе.



*Рис.7. Структура таблицы «Ингредиенты»*

**Таблица “** **Закупка Сырья”**

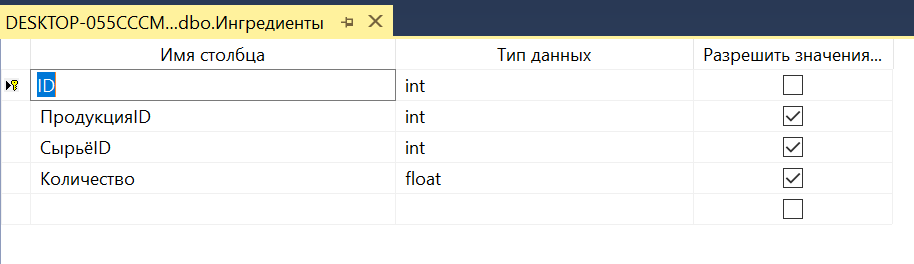
Отслеживает закупки сырья, включая количество, стоимость и даты поставок, что важно для управления запасами и финансами.



*Рис.8. Структура таблицы «ЗакупкаСырья»*

**Таблица “** **Ингредиенты ”**

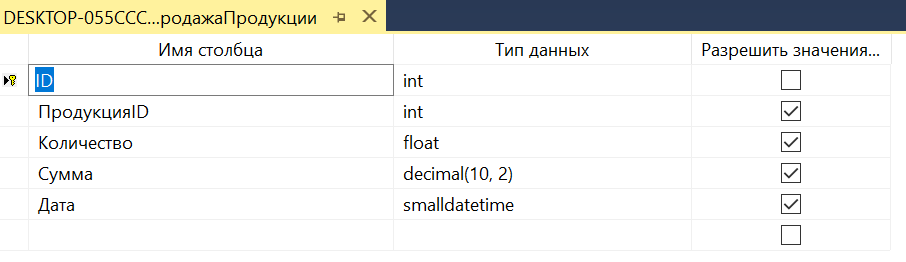
Содержит информацию о компонентах, используемых в производстве готовой продукции, позволяя контролировать потребление сырья и расчет себестоимости продуктов.

****

*Рис.9. Структура таблицы* «*Ингредиенты»*

**Таблица “** **Продажа Продукции”**

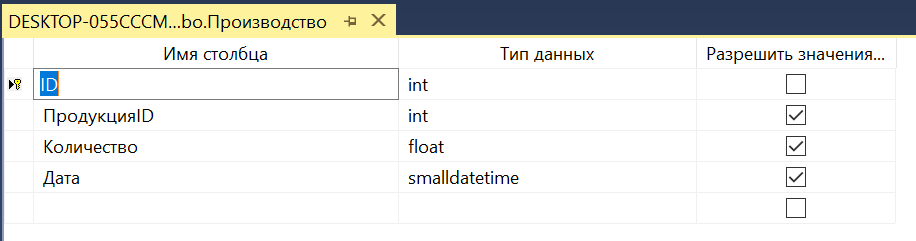
Фиксирует все сделки по продаже продукции, отражая доходы и помогая анализировать рыночный спрос.



*Рис.10. Структура таблицы «ПродажаПродукции»*

**Таблица “** **Производство”**

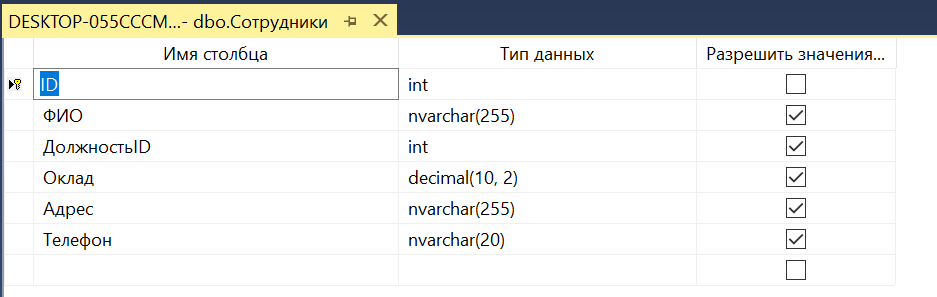
Отображает процесс производства товаров, включая использование сырья и выпуск готовой продукции, что критично для оперативного планирования.



*Рис.11. Структура таблицы «Производство»*

**Таблица “** **Сотрудники”**

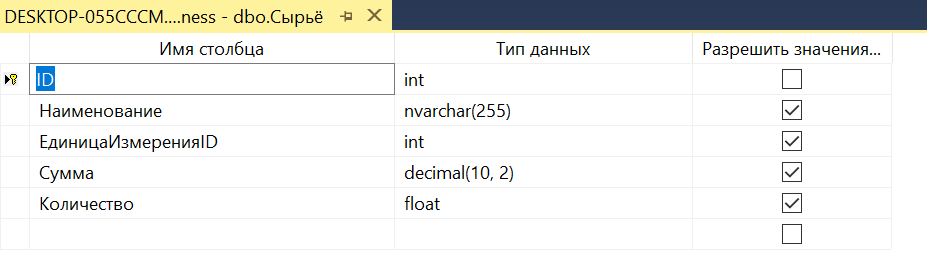
Включает данные о сотрудниках предприятия, их должностях и личной информации, обеспечивая управление персоналом.



*Рис.12. Структура таблицы «Сотрудники»*

**Таблица “** **Сырьё”**

Содержит информацию о сырьевых материалах на складе, их количестве и стоимости, что необходимо для бухгалтерского учета и планирования закупок.



*Рис.13. Структура таблицы «Сырьё»*

## **1.2. Хранимые процедуры, описание каждой хранимки:**

|  |  |
| --- | --- |
| ***Код*** | ***Описание*** |
| ALTER PROCEDURE [dbo].[SP\_ingredients]  @ProductID INT  AS  BEGIN  SELECT \* FROM Ингредиенты WHERE ПродукцияID = @ProductID;  END; | Эта процедура возвращает список всех ингредиентов, используемых в указанной продукции, принимая @ProductID в качестве параметра для определения продукции. |
| ALTER PROCEDURE [dbo].[SP\_ingreadd]  @ПродукцияID INT,  @СырьёID INT,  @Количество float  AS  BEGIN  SET NOCOUNT ON;  IF EXISTS (SELECT 1 FROM Ингредиенты WHERE ПродукцияID = @ПродукцияID AND СырьёID = @СырьёID)  BEGIN  UPDATE Ингредиенты  SET Количество = Количество + @Количество  WHERE ПродукцияID = @ПродукцияID AND СырьёID = @СырьёID;  END  ELSE  BEGIN  INSERT INTO Ингредиенты (ПродукцияID, СырьёID, Количество)  VALUES (@ПродукцияID, @СырьёID, @Количество);  END  SET NOCOUNT OFF;  END | Эта процедура добавляет новую запись об ингредиенте или обновляет существующую в зависимости от того, есть ли уже в базе данных сочетание конкретной продукции (@ПродукцияID) и сырья (@СырьёID). Если такая комбинация уже существует, процедура увеличивает количество сырья на указанное значение. Если такой записи нет, создается новая запись с указанными значениями продукции, сырья и количества. |
| ALTER PROCEDURE [dbo].[SP\_ingreupdt]  @ПродукцияID INT,  @СырьёID INT,  @НовоеКоличество float  AS  BEGIN  SET NOCOUNT ON;  IF EXISTS (SELECT 1 FROM Ингредиенты WHERE ПродукцияID = @ПродукцияID AND СырьёID = @СырьёID)  BEGIN  UPDATE Ингредиенты  SET Количество = @НовоеКоличество  WHERE ПродукцияID = @ПродукцияID AND СырьёID = @СырьёID;  END  ELSE  BEGIN  THROW 50000, 'Сырье для данной продукции не найдено.', 1;  END  SET NOCOUNT OFF;  END | Это процедура обновляет количество сырья (@СырьёID) для заданной продукции (@ПродукцияID). Если запись существует, обновляет количество на @НовоеКоличество. В случае отсутствия записи выдает ошибку о ненайденном сырье. |
| ALTER PROCEDURE [dbo].[SP\_GetProizvodstvo]  AS  BEGIN  SELECT \* FROM Производство;  END; | Это процедура показывает все записи из таблицы Производство. |
| ALTER PROCEDURE [dbo].[SP\_GetProdaja]  AS  BEGIN  SELECT \* FROM ПродажаПродукции;  END; | Это процедура показывает все записи о продажах продукции. |
| ALTER PROCEDURE [dbo].[SP\_ingredel]  @ПродукцияID INT,  @СырьёID INT  AS  BEGIN  SET NOCOUNT ON;  IF EXISTS (SELECT 1 FROM Ингредиенты WHERE ПродукцияID = @ПродукцияID AND СырьёID = @СырьёID)  BEGIN  DELETE FROM Ингредиенты  WHERE ПродукцияID = @ПродукцияID AND СырьёID = @СырьёID;  END  ELSE  BEGIN  THROW 50000, 'Запись с указанными ПродукцияID и СырьёID ненайдена.', 1;  END  SET NOCOUNT OFF;  END | Эта процедура удаляет запись о сочетании определенного сырья (@СырьёID) и продукции (@ПродукцияID). При наличии такой записи в таблице Ингредиенты, она удаляется. Если соответствующая запись отсутствует, процедура возвращает сообщение об ошибке. |
| ALTER PROCEDURE [dbo].[SP\_Pokupka]  @СырьёID INT,  @Количество FLOAT,  @ЦенаЗаЕдиницу DECIMAL(10, 2)  AS  BEGIN  SET NOCOUNT ON;  DECLARE @ДоступныйБюджет DECIMAL(10, 2);  DECLARE @ОбщаяСтоимость DECIMAL(10, 2);  SET @ОбщаяСтоимость = @Количество \* @ЦенаЗаЕдиницу;  SELECT @ДоступныйБюджет = СуммаБюджета FROM Бюджет WHERE ID = 1;  IF @ДоступныйБюджет >= @ОбщаяСтоимость  BEGIN  UPDATE Бюджет SET СуммаБюджета = СуммаБюджета - @ОбщаяСтоимость WHERE ID = 1;  UPDATE Сырьё  SET Количество = Количество + @Количество,  Сумма = Сумма + @ОбщаяСтоимость  WHERE ID = @СырьёID;  INSERT INTO ЗакупкаСырья (СырьёID, Количество, Сумма, Дата)  VALUES (@СырьёID, @Количество, @ОбщаяСтоимость, GETDATE());  END  ELSE  BEGIN  THROW 50000, 'Недостаточно средств в бюджете для закупки сырья.', 1;  END  SET NOCOUNT OFF;  END | Эта процедура реализует процесс покупки сырья, проверяя достаточность бюджета и обновляя соответствующие записи. Она принимает идентификатор сырья (@СырьёID), количество купленного сырья (@Количество) и цену за единицу (@ЦенаЗаЕдиницу). Процедура сначала рассчитывает общую стоимость закупки и проверяет, достаточно ли средств в бюджете для ее осуществления. Если средств достаточно, процедура обновляет информацию о сырье на складе и вычитает стоимость покупки из бюджета. В противном случае процедура возвращает ошибку о недостаточности средств. |
| ALTER PROCEDURE [dbo].[SP\_ZakupkiSyrya]  AS  BEGIN  SELECT \* FROM ЗакупкаСырья;  END; | Эта процедура показывает список закупок сырья. |
| ALTER PROCEDURE [dbo].[SP\_Proizvodstvo]  @ПродукцияID INT,  @КоличествоПродукции FLOAT  AS  BEGIN  SET NOCOUNT ON;  BEGIN TRANSACTION;  DECLARE @СырьёID INT;  DECLARE @ТребуемоеКоличествоСырья FLOAT;  DECLARE @ДоступноеКоличествоСырья FLOAT;  DECLARE @СтоимостьСырья DECIMAL(10, 2);  DECLARE @ОбщаяСтоимостьСырья DECIMAL(10, 2) = 0;  -- Курсор для перебора всех ингредиентов продукции  DECLARE ing\_cursor CURSOR FOR  SELECT СырьёID, Количество  FROM Ингредиенты  WHERE ПродукцияID = @ПродукцияID;  OPEN ing\_cursor;  FETCH NEXT FROM ing\_cursor INTO @СырьёID, @ТребуемоеКоличествоСырья;  WHILE @@FETCH\_STATUS = 0  BEGIN  SELECT @ДоступноеКоличествоСырья = Количество, @СтоимостьСырья = Сумма / Количество  FROM Сырьё  WHERE ID = @СырьёID;  -- Умножаем требуемое количество сырья на количество производимой продукции  SET @ТребуемоеКоличествоСырья \*= @КоличествоПродукции;  -- Проверка на отрицательный остаток сырья после списания  IF @ДоступноеКоличествоСырья < @ТребуемоеКоличествоСырья  BEGIN  -- Недостаточно сырья, отменяем транзакцию  CLOSE ing\_cursor;  DEALLOCATE ing\_cursor;  ROLLBACK TRANSACTION;  RAISERROR('Недостаточно сырья на складе для производства.', 16, 1);  RETURN;  END  -- Списание сырья со склада и расчет стоимости  UPDATE Сырьё  SET Количество = Количество - @ТребуемоеКоличествоСырья,  Сумма = Сумма - (@ТребуемоеКоличествоСырья \* @СтоимостьСырья)  WHERE ID = @СырьёID;  -- Накопление общей стоимости сырья  SET @ОбщаяСтоимостьСырья += @ТребуемоеКоличествоСырья \* @СтоимостьСырья;  FETCH NEXT FROM ing\_cursor INTO @СырьёID, @ТребуемоеКоличествоСырья;  END  CLOSE ing\_cursor;  DEALLOCATE ing\_cursor;  -- Обновление готовой продукции  UPDATE ГотоваяПродукция  SET Количество = Количество + @КоличествоПродукции,  Сумма = Сумма + @ОбщаяСтоимостьСырья  WHERE ID = @ПродукцияID;  -- Запись в таблицу Производство  INSERT INTO Производство (ПродукцияID, Количество, Дата)  VALUES (@ПродукцияID, @КоличествоПродукции, GETDATE());  COMMIT TRANSACTION;  SET NOCOUNT OFF;  END | Эта хранимая процедура проверяет запасы сырья перед производством продукции, списывает его при достаточном количестве и увеличивает запасы готовой продукции. При нехватке сырья операция отклоняется с ошибкой. |
| ALTER PROCEDURE [dbo].[SP\_Prodaja]  @ПродукцияID INT,  @КоличествоПродукции FLOAT,  @ПроцентНаЦенку FLOAT  AS  BEGIN  SET NOCOUNT ON;  BEGIN TRANSACTION;  DECLARE @ДоступноеКоличество FLOAT;  DECLARE @СтоимостьПродукции DECIMAL(10, 2);  DECLARE @СуммаПродажи DECIMAL(10, 2);  -- Проверяем наличие продукции на складе  SELECT @ДоступноеКоличество = Количество, @СтоимостьПродукции = Сумма / Количество  FROM ГотоваяПродукция  WHERE ID = @ПродукцияID;  -- Если продукции достаточно для продажи  IF @ДоступноеКоличество >= @КоличествоПродукции  BEGIN  -- Рассчитываем сумму продажи с учетом заданной наценки  SET @СуммаПродажи = (@КоличествоПродукции \* @СтоимостьПродукции) \* (1 + @ПроцентНаЦенку / 100.0);  -- Уменьшаем количество продукции на складе  UPDATE ГотоваяПродукция  SET Количество = Количество - @КоличествоПродукции,  Сумма = Сумма - (@КоличествоПродукции \* @СтоимостьПродукции)  WHERE ID = @ПродукцияID;  -- Увеличиваем бюджет на сумму продажи  UPDATE Бюджет  SET СуммаБюджета = СуммаБюджета + @СуммаПродажи  WHERE ID = 1;  -- Добавляем запись о продаже в таблицу ПродажаПродукции  INSERT INTO ПродажаПродукции (ПродукцияID, Количество, Сумма, Дата)  VALUES (@ПродукцияID, @КоличествоПродукции, @СуммаПродажи, GETDATE());  COMMIT TRANSACTION;  END  ELSE  BEGIN  -- Если продукции недостаточно, отменяем транзакцию и возвращаем ошибку  ROLLBACK TRANSACTION;  THROW 50000, 'Недостаточно продукции на складе для реализации.', 1;  END  SET NOCOUNT OFF;  END | Эта хранимая процедура управляет продажей продукции, проверяя наличие необходимого количества на складе и так же процент продажи берем с таблицы бюджет. При достаточном количестве продукции она рассчитывает сумму продажи, обновляет запасы и бюджет. В случае недостатка продукции на складе, процедура отменяется с сообщением об ошибке. |

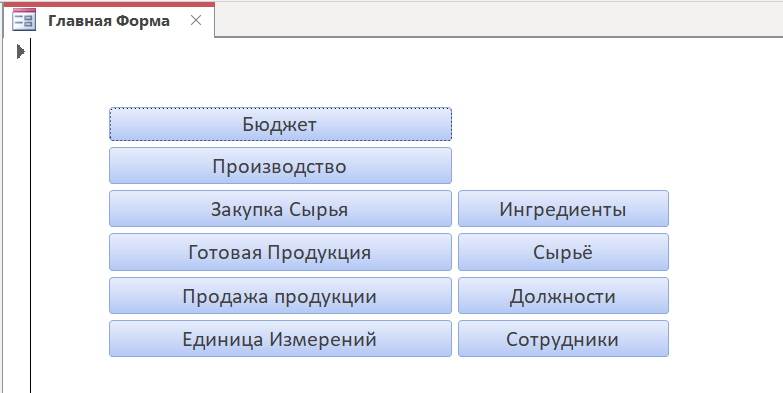
# **Глава 2. Клиентская часть**

Описание клиентской части:

В проекте я настроил подключение к серверу через специальный модуль, разработал интерфейс различных форм в Access и создал главную форму для навигации по приложению. Каждая форма была оптимизирована для удобства использования, обеспечивая эффективный доступ к функциям и данным.

**Форма “** **Главная Форма”**

С помощью этой формы я могу переходить к различным функциям управления базой данных: отслеживанию бюджета, процессам производства, закупке сырья, учету готовой продукции, продажам и управлению персоналом. Каждый раздел легко доступен благодаря кнопкам, что делает использование базы данных понятным и эффективным.

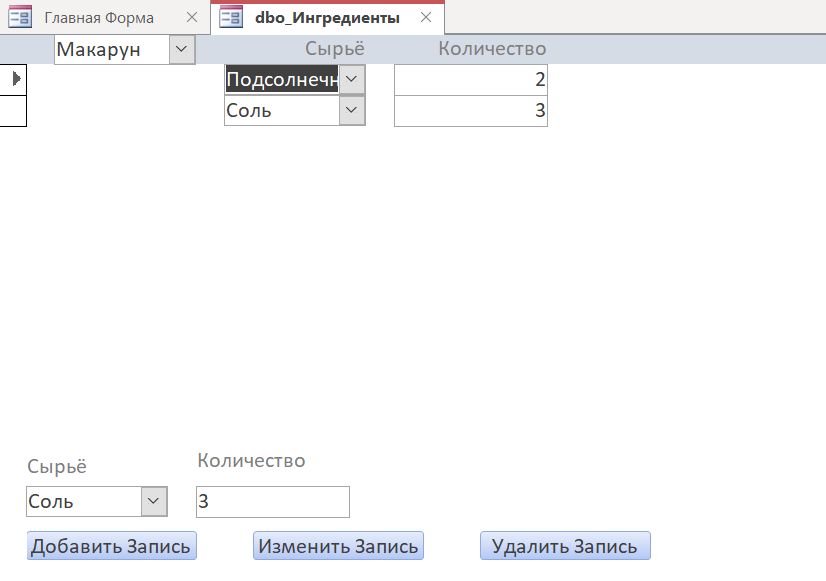


*Рис.14. Интерфейс формы «Главная Форма»*

**Форма “** **Ингредиенты”**

В этой форме я могу выбрать продукцию из выпадающего списка в верхней части, а затем вносить и редактировать ингредиенты, используемые в этом продукте. Есть поля для выбора типа сырья и ввода количества, которое требуется. В центральной таблице отображается список уже добавленных ингредиентов. Под таблицей расположены кнопки, позволяющие мне добавлять новые записи, вносить изменения в существующие или удалять их.

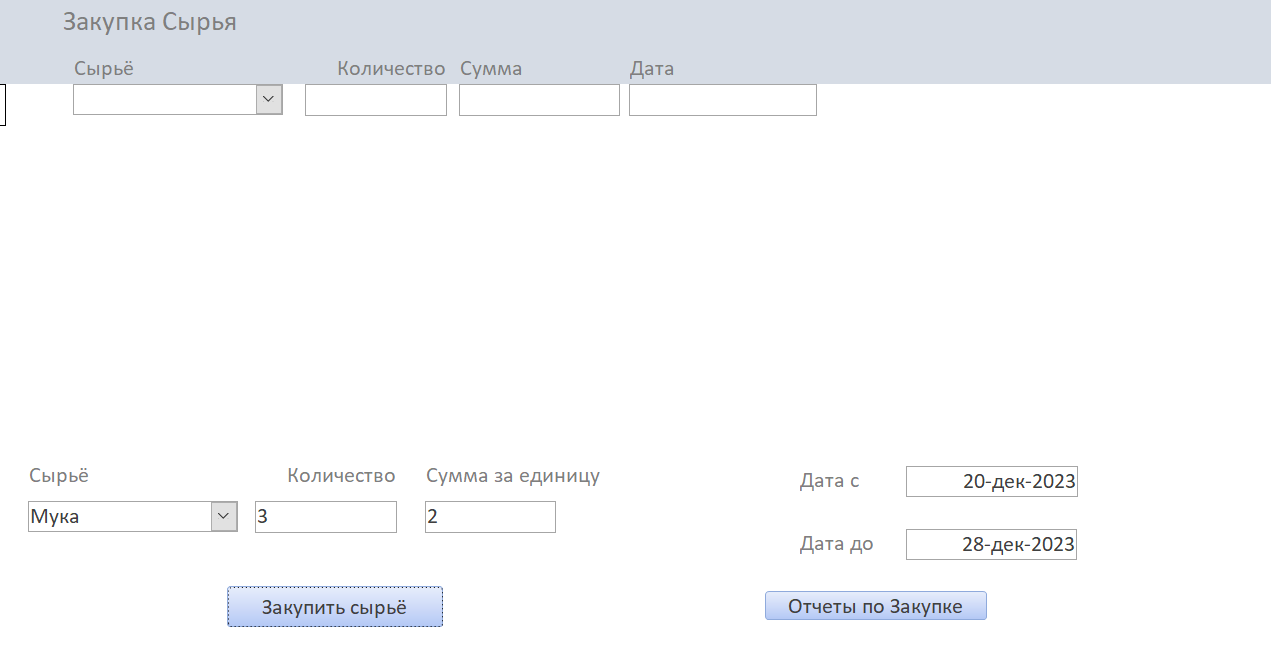
Каждая кнопка связанна с Хранимыми процедурами.



*Рис.15. Интерфейс формы «Ингредиенты»*

**Форма “** **Закупка Сырья”**

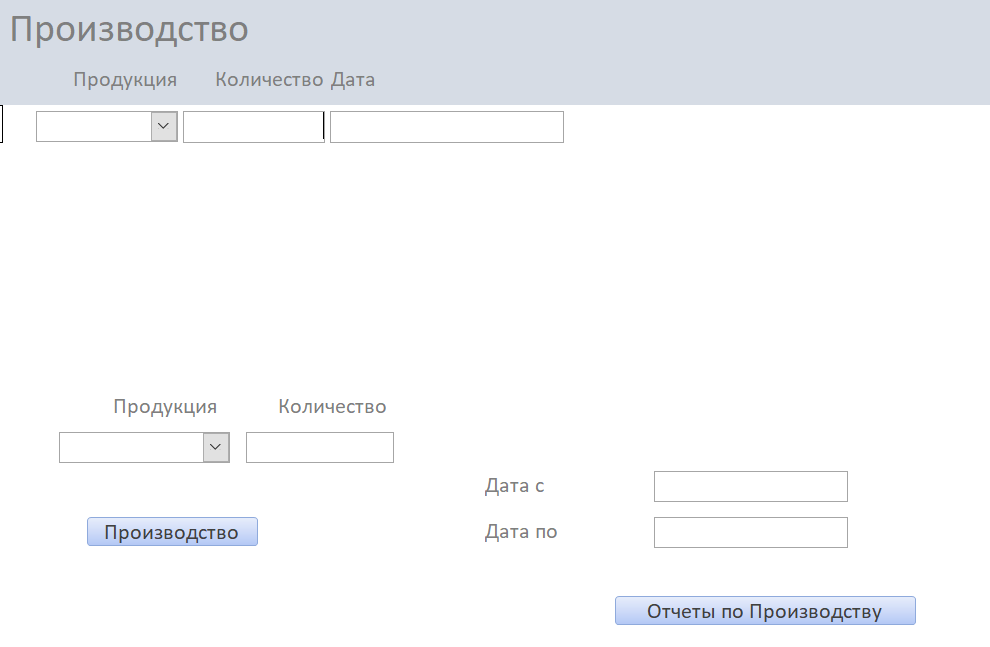
На форме "Закупка Сырья" я могу выбрать тип сырья из выпадающего списка и ввести необходимое количество и стоимость за единицу. Внизу формы есть кнопка для подтверждения закупки. Также есть поля для фильтрации данных по дате, что позволяет мне просматривать закупки за определенный период времени и показывать отчеты по закупке. Так же после закупки сразу же обновляется данные формы. Связана с хранимой процедурой SP\_Pokupka и SP\_ZakupkiSyrya.

****

*Рис.16. Интерфейс формы «Закупка Сырья»*

**Форма “** **Производство”**

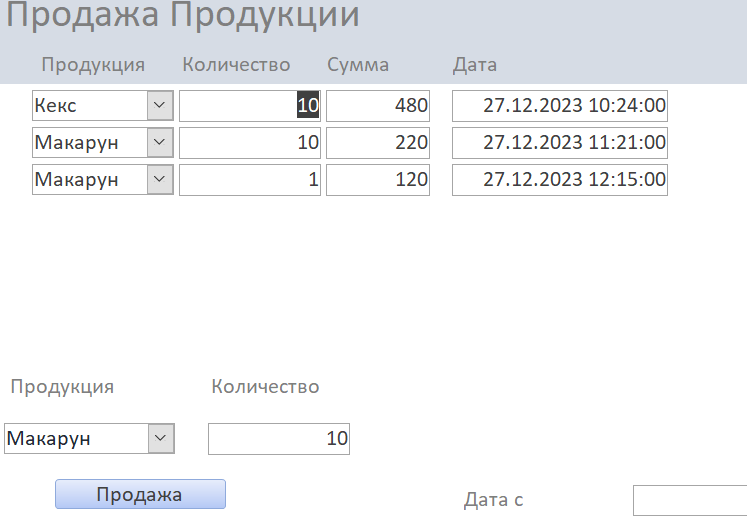
В этой форме можно реализовать производство продукции. Так же посмотреть отчеты по Производствам. Связан с SP\_Proizvodstvo и SP\_Getproivodstvo.



*Рис.17. Интерфейс формы «Производство»*

**Форма “** **Продажа Продукции”**

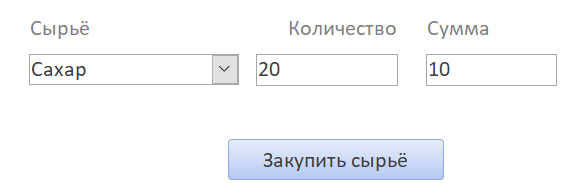
В этой форме мы реализуем продажи продукции, так же просмотр отчета Продаж по выбранной дате. Связан с SP\_Prodaja и SP\_GetProdaja.



*Рис.18. Интерфейс формы «Продажа Продукции»*

Проверка всех процессов:

Начнем с Закупки Сырья при закупке у нас сразу же обновляется Склад Сырья и так же бюджет. Так же в Складе Сырья у нас обновляется общая сумму.

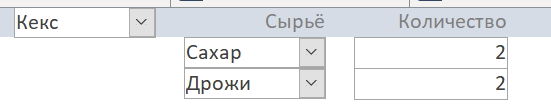




*Рис.19. Проверка формы «Закупки Сырья»*

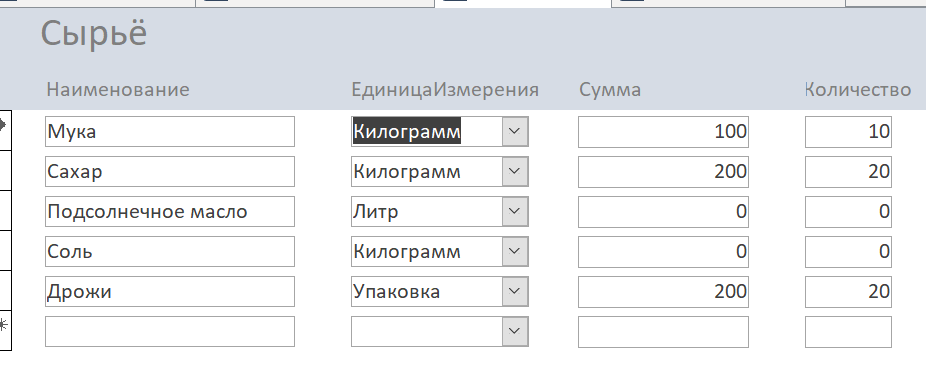
**Проверка формы Производство**

При производстве продукта у нас проверяется ингредиенты продукта допустим для Кекса нам нужно 2 колличества Дрожи и Сахара.



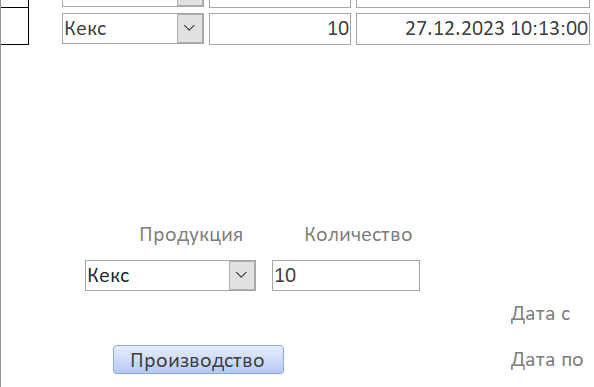
*Рис.19. Проверка формы «Ингредиенты»*

А в складе сырья у нас по 20 штук на эти сырья.



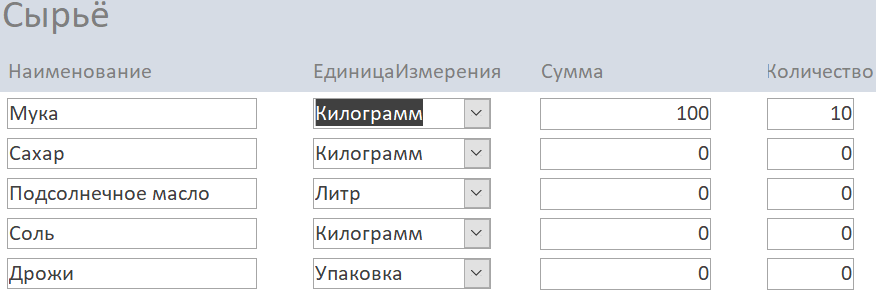
*Рис.20. Проверка формы «Сырьё»*

Проверка формы производство при производстве у нас сразу же обновляется форма Производство.



*Рис.21. Проверка формы «Производство»*

После производство у нас обновляется в складе сырья количество сырья и так же готовая продукция сумма и количество.

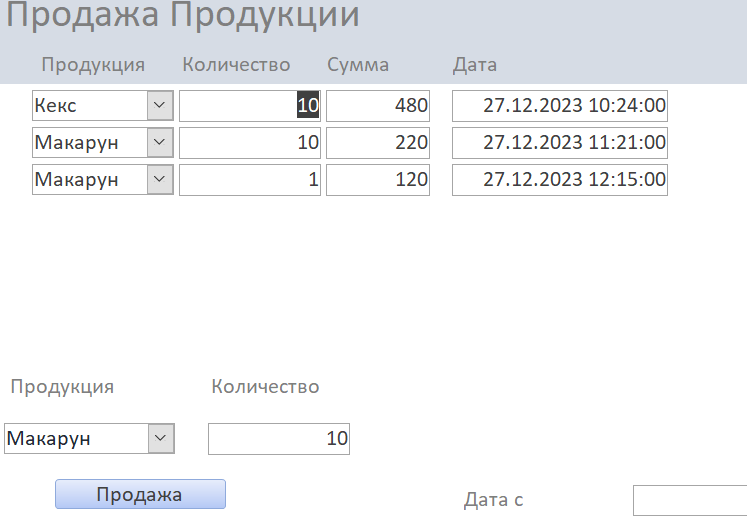


*Рис.22. Проверка формы «Сырьё»*



*Рис.23. Проверка формы «Готовая Продукция»*

Проверка продажи продукции при продаже продукци у нас сразу же обновляется форма и общая сумма продажа и мы взяли с таблицы бюджет процент продажи при продаже у нас 1 количество на 20% дороже продается и так же обновляется бюджет на эту сумму и списывает количество и сумму в форме Готовая Продукция.



*Рис.23. Проверка формы «Продажа Продукции»*

Заключение: Подводя итог своему проекту, я укрепил свои знания работы с базами данных. Создание базы данных и приложения для учета ресурсов помогло мне лучше понять, как важно правильно управлять данными и создавать интерфейсы, удобные для пользователя. Этот проект стал отличной практикой, которая будет полезна в моей будущей карьере.