**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**

**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

**Кафедра ИС**

**ОТЧЕТ**

**по заданию №2-2**

**по дисциплине «Методы и средства проектирования информационных систем»**

**Тема: Задача 2-2. Приемы моделирования маршрутной технологии**

|  |  |  |
| --- | --- | --- |
| Студент гр. 1374  Студент гр. 1374  Студент гр. 1374 |  | Архипов Д.А.  Дюков Н.В.  Хлебников А. |
| Преподаватель |  | Дубенецкий В.А. |

Санкт-Петербург

2024г.

**Содержание**

[**1. Задание на выполнение работы 3**](#_heading=h.30j0zll)

[1.1 Функциональные требования 3](#_heading=h.1fob9te)

[1.2 Содержание работы 3](#_heading=h.3dy6vkm)

[1.3 Исходные данные для задачи 4](#_heading=h.1t3h5sf)

[1.4 Используемые инструменты 4](#_heading=h.4d34og8)

[1.5 Используемые документы 4](#_heading=h.2s8eyo1)

[**2. Описание выполненной работы 5**](#_heading=h.17dp8vu)

[2.1 Анализ исходных данных 5](#_heading=h.3rdcrjn)

[2.2 Разработка функциональных требований к подсистеме 6](#_heading=h.26in1rg)

[2.3 Разработка проектной модели классов 7](#_heading=h.lnxbz9)

[2.4 Разработка модели хранения в среде СУБД 7](#_heading=h.35nkun2)

[2.5 Создание базы данных 8](#_heading=h.1ksv4uv)

[2.6 Разработка основных процедур 11](#_heading=h.44sinio)

[2.7 Тестирование разработки 20](#_heading=h.2jxsxqh)

[**Вывод 24**](#_heading=h.3j2qqm3)

**Задача №1**

**Приемы моделирования классификатора изделий**

# Задание на выполнение работы

Разработать проект каркаса для работы с маршрутной

технологией изготовления изделий

## 1.1 Функциональные требования

Разработать проект каркаса для работы с маршрутной

технологией изготовления изделий

## Обеспечить:

## Ведение справочника изделий различных типов, используемых в составе изделий;

## Формирование строк маршрутной технологии с указанием классов технологических операций, требуемого оборудования, инструмента, норм расхода материалов и комплектующих;

## Поиск всех строк технологического маршрута изделия на всю глубину вложенности;

## Расчет сводных норм расхода компонентов маршрутной спецификации с указанием потребности в трудовых и материальных ресурсов заданного класса.

## 1.2 Содержание работы

|  |  |  |
| --- | --- | --- |
| **№** | **Этап выполнения** | **Результат** |
|  | Разработать функциональные требования к проектируемой подсистеме | Диаграммы вариантов использования. Описание компонентов диаграмм |
|  | Разработать модель классов для выделенных процессов | Диаграммы классов с операциями и атрибутами. Описание компонентов диаграмм. |
|  | Разработать модель хранения в среде СУБД для решения задач управления материальной спецификацией | Диаграммы ER. Скрипты метаданных с комментариями. |
|  | Разработка основных SQL-процедур для работы с классификатором | Скрипты SQL-процедур с комментариями. |
|  | Тестирование процедур | Описание тестов. Скрипты исходных данных для тестов.  Скриншоты результатов тестирования. |

## 1.3 Исходные данные для задачи

|  |  |  |
| --- | --- | --- |
| № вар. | Наименование изделия | Источник (для просмотра использовать вьюер ViewRep.exe) |
|  | Самостоятельный выбор примера маршрутной технологии |  |

## 1.4 Используемые инструменты

РедакторVisio. МодельDatabase.Entity relationship

IDE Datagrip

СУБДPostgreSQL

Текстовыйредактор Word

## 1.5 Используемые документы

1. Data Definition Guide : файл DataDef.pdf
2. Language Reference Guide: файл LangRef.pdf
3. Архитектура информационных систем. Учебник. М.,Издательский центр «Академия», 2012
4. Информационное обеспечение жизненного цикла изделий. Учебное пособие: СПб, Издательство Политехнического университета, 2012
5. Приемы объектно-ориентированного проектирования. Паттерны проектирования: Изд-во «ПИТЕР», 2008
6. Проектирование корпоративных информационных систем.: СПб, Изд-во СПбГЭТУ «ЛЭТИ», 2013

# Описание выполненной работы

## 2.1 Маршрутная технология и маршрутные карты

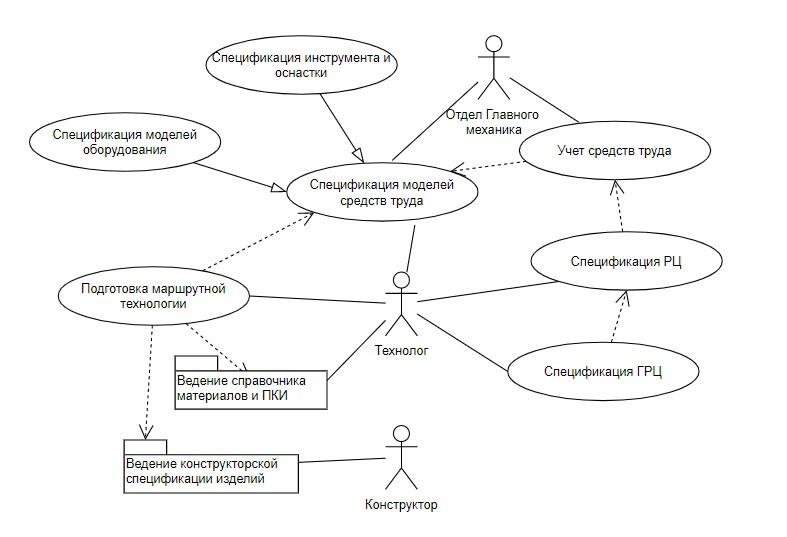
Правила описания маршрутных карт изложены в ЕСТД. Единая Система Технологической Документации (ЕСТД) — комплекс межгосударственных стандартов и рекомендаций, устанавливающих взаимосвязанные правила и положения по порядку разработки, комплектации, оформления и обращения технологической документации, применяемой при изготовлении, контроле, приемке и ремонте (модернизации) изделий (включая сбор и сдачу технологических отходов).

## 2.2 Разработка функциональных требований к подсистеме

Для разработки концептуального проекта будем использовать редактор VISIO.

Создадим диаграмму вариантов использования с именем Ведение маршрутной спецификации изделий.

На рис. 1 приведена диаграмма прецедентов Ведение маршрутной спецификации изделий.



*Рисунок 1 – Диаграмма прецедентов Ведение маршрутной спецификации изделий*

Описание компонентов диаграммы.

Создадим следующие варианты использования:

*- Спецификация моделей средств труда;*

*- Учет средств труда;*

*- Подготовка маршрутной технологии;*

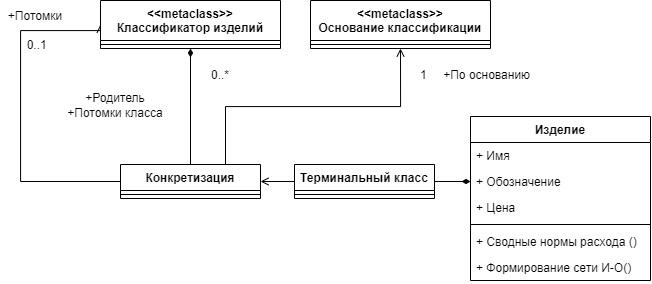
Используем пакеты:

*- Конструкторская подготовка производства;*

*- Ведение данных по трудовым ресурсам;*

*- Ведение справочника материалов и покупных изделий.*

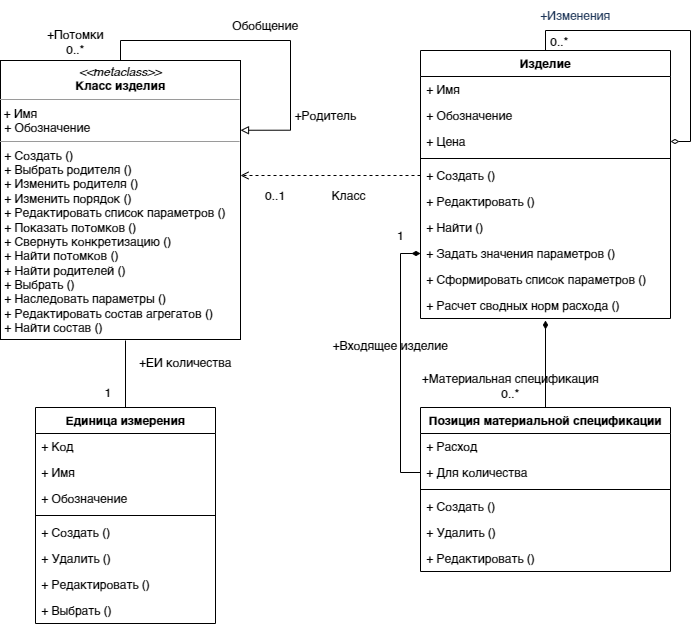
## Разработка модели классов этапа анализа и проектирования



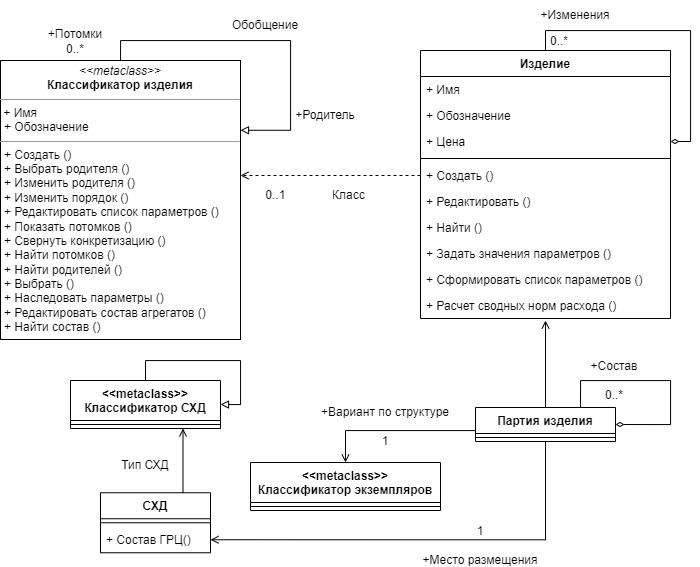
*Рисунок 6 - Модель для описания много альтернативной классификации*

Для решения задач ведения маршрутной технологии воспользуемся результатами проекта ведения материальных спецификаций и доработаем их.

Модель классов для ведения материальной спецификации представлена на рис. 7.

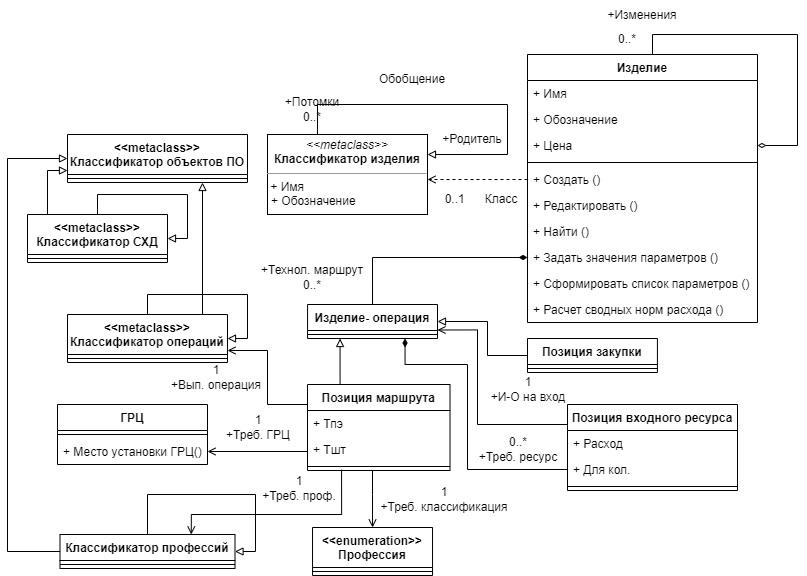


*Рисунок 7 - Базовое решение для описания материальной спецификации*



*Рисунок 8 - Модель для описания структуры СХД, РЦ и ГРЦ*

Модель классов для прецедента *Ведение маршрутной спецификации изделий* представлена на рис. 9.



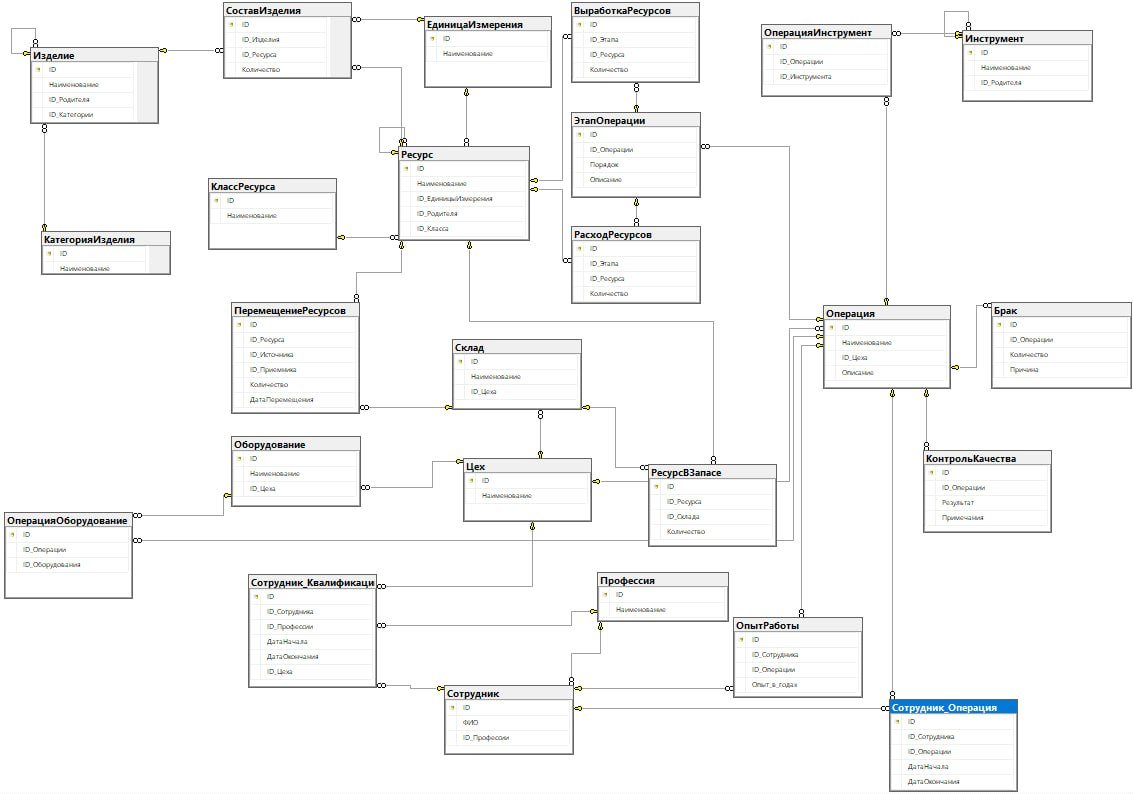
*Рисунок 9 - Модель классов для прецедента Ведение маршрутной спецификации изделий*

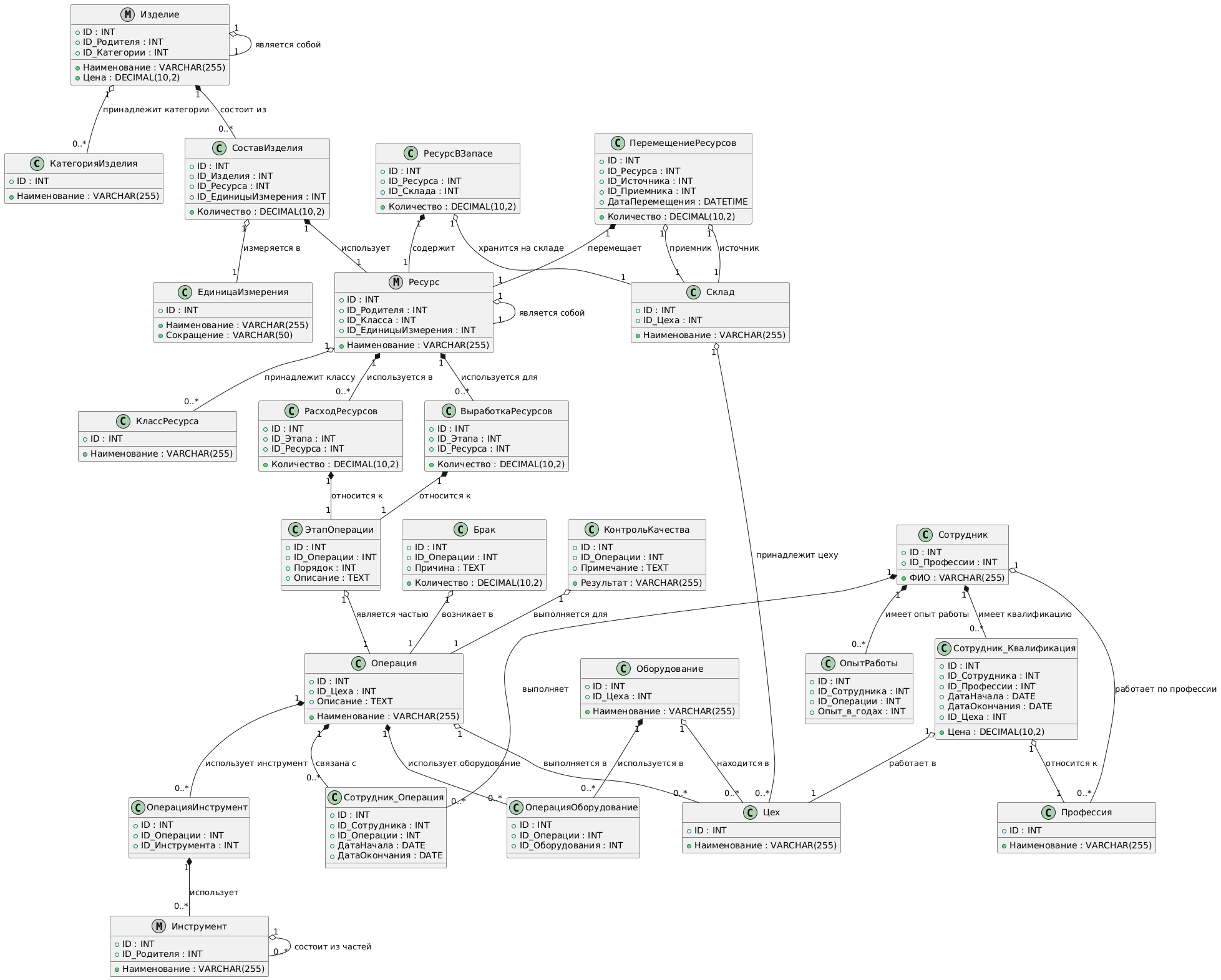
Описание компонентов диаграммы

- *Изделие-Операция* – введен для описания всех точек маршрута каждого изделия (изделие с точки зрения его изготовления описывается совокупностью взаимосвязанных состояний в соответствии с выполненной операцией);

- *Позиция входного ресурса* – это список других изделий-операций, которые расходуются для получения требуемого изделия –операции.

## Разработка модели хранения





*Рисунок 10 - ERD для работы с маршрутной технологией*

## Создание базы данных

-- Создание таблиц

CREATE TABLE КатегорияИзделия (

ID INT PRIMARY KEY,

Наименование VARCHAR(255)

);

CREATE TABLE Изделие (

ID INT PRIMARY KEY,

Наименование VARCHAR(255),

ID\_Родителя INT,

ID\_Категории INT,

Цена DECIMAL(10, 2),

FOREIGN KEY (ID\_Категории) REFERENCES КатегорияИзделия(ID)

);

CREATE TABLE ЕдиницаИзмерения (

ID INT PRIMARY KEY,

Наименование VARCHAR(255),

Сокращение VARCHAR(50)

);

CREATE TABLE Ресурс (

ID INT PRIMARY KEY,

Наименование VARCHAR(255),

ID\_Родителя INT,

ID\_Класса INT,

ID\_ЕдиницыИзмерения INT,

FOREIGN KEY (ID\_ЕдиницыИзмерения) REFERENCES ЕдиницаИзмерения(ID)

);

CREATE TABLE КлассРесурса (

ID INT PRIMARY KEY,

Наименование VARCHAR(255)

);

CREATE TABLE СоставИзделия (

ID INT PRIMARY KEY,

ID\_Изделия INT,

ID\_Ресурса INT,

Количество DECIMAL(10,2),

ID\_ЕдиницыИзмерения INT,

FOREIGN KEY (ID\_Изделия) REFERENCES Изделие(ID),

FOREIGN KEY (ID\_Ресурса) REFERENCES Ресурс(ID),

FOREIGN KEY (ID\_ЕдиницыИзмерения) REFERENCES ЕдиницаИзмерения(ID)

);

CREATE TABLE Профессия (

ID INT PRIMARY KEY,

Наименование VARCHAR(255)

);

CREATE TABLE Сотрудник (

ID INT PRIMARY KEY,

ФИО VARCHAR(255),

ID\_Профессии INT,

FOREIGN KEY (ID\_Профессии) REFERENCES Профессия(ID)

);

CREATE TABLE ОпытРаботы (

ID INT PRIMARY KEY,

ID\_Сотрудника INT,

ID\_Операции INT,

Опыт\_в\_годах INT,

FOREIGN KEY (ID\_Сотрудника) REFERENCES Сотрудник(ID)

);

CREATE TABLE Инструмент (

ID INT PRIMARY KEY,

Наименование VARCHAR(255),

ID\_Родителя INT

);

CREATE TABLE ОперацияИнструмент (

ID INT PRIMARY KEY,

ID\_Операции INT,

ID\_Инструмента INT,

FOREIGN KEY (ID\_Операции) REFERENCES Операция(ID),

FOREIGN KEY (ID\_Инструмента) REFERENCES Инструмент(ID)

);

CREATE TABLE Операция (

ID INT PRIMARY KEY,

Наименование VARCHAR(255),

ID\_Цеха INT,

Описание TEXT

);

CREATE TABLE Цех (

ID INT PRIMARY KEY,

Наименование VARCHAR(255)

);

CREATE TABLE Оборудование (

ID INT PRIMARY KEY,

Наименование VARCHAR(255),

ID\_Цеха INT,

FOREIGN KEY (ID\_Цеха) REFERENCES Цех(ID)

);

CREATE TABLE ОперацияОборудование (

ID INT PRIMARY KEY,

ID\_Операции INT,

ID\_Оборудования INT,

FOREIGN KEY (ID\_Операции) REFERENCES Операция(ID),

FOREIGN KEY (ID\_Оборудования) REFERENCES Оборудование(ID)

);

CREATE TABLE Склад (

ID INT PRIMARY KEY,

Наименование VARCHAR(255),

ID\_Цеха INT,

FOREIGN KEY (ID\_Цеха) REFERENCES Цех(ID)

);

CREATE TABLE РесурсВЗапасе (

ID INT PRIMARY KEY,

ID\_Ресурса INT,

ID\_Склада INT,

Количество DECIMAL(10,2),

FOREIGN KEY (ID\_Ресурса) REFERENCES Ресурс(ID),

FOREIGN KEY (ID\_Склада) REFERENCES Склад(ID)

);

CREATE TABLE ПеремещениеРесурсов (

ID INT PRIMARY KEY,

ID\_Ресурса INT,

ID\_Источника INT,

ID\_Приемника INT,

Количество DECIMAL(10,2),

ДатаПеремещения DATETIME,

FOREIGN KEY (ID\_Ресурса) REFERENCES Ресурс(ID)

);

CREATE TABLE Брак (

ID INT PRIMARY KEY,

ID\_Операции INT,

Количество DECIMAL(10,2),

Причина TEXT,

FOREIGN KEY (ID\_Операции) REFERENCES Операция(ID)

);

CREATE TABLE ЭтапОперации (

ID INT PRIMARY KEY,

ID\_Операции INT,

Порядок INT,

Описание TEXT,

FOREIGN KEY (ID\_Операции) REFERENCES Операция(ID)

);

CREATE TABLE КонтрольКачества (

ID INT PRIMARY KEY,

ID\_Операции INT,

Результат VARCHAR(255),

Примечание TEXT,

FOREIGN KEY (ID\_Операции) REFERENCES Операция(ID)

);

CREATE TABLE РасходРесурсов (

ID INT PRIMARY KEY,

ID\_Этапа INT,

ID\_Ресурса INT,

Количество DECIMAL(10,2),

FOREIGN KEY (ID\_Этапа) REFERENCES ЭтапОперации(ID),

FOREIGN KEY (ID\_Ресурса) REFERENCES Ресурс(ID)

);

CREATE TABLE ВыработкаРесурсов (

ID INT PRIMARY KEY,

ID\_Этапа INT,

ID\_Ресурса INT,

Количество DECIMAL(10,2),

FOREIGN KEY (ID\_Этапа) REFERENCES ЭтапОперации(ID),

FOREIGN KEY (ID\_Ресурса) REFERENCES Ресурс(ID)

);

CREATE TABLE Сотрудник\_Операция (

ID INT PRIMARY KEY IDENTITY(1,1),

ID\_Сотрудника INT NOT NULL,

ID\_Операции INT NOT NULL,

ДатаНачала DATE NOT NULL,

ДатаОкончания DATE,

CONSTRAINT FK\_Сотрудник\_Операция\_Сотрудник FOREIGN KEY (ID\_Сотрудника) REFERENCES Сотрудник(ID),

CONSTRAINT FK\_Сотрудник\_Операция\_Операция FOREIGN KEY (ID\_Операции) REFERENCES Операция(ID)

);

CREATE TABLE Сотрудник\_Квалификация (

ID INT PRIMARY KEY IDENTITY(1,1),

ID\_Сотрудника INT NOT NULL,

ID\_Профессии INT NOT NULL,

ДатаНачала DATE NOT NULL,

ДатаОкончания DATE,

ID\_Цеха INT,

Цена DECIMAL(10, 2),

CONSTRAINT FK\_Сотрудник\_Квалификация\_Сотрудник FOREIGN KEY (ID\_Сотрудника) REFERENCES Сотрудник(ID),

CONSTRAINT FK\_Сотрудник\_Квалификация\_Профессия FOREIGN KEY (ID\_Профессии) REFERENCES Профессия(ID),

CONSTRAINT FK\_Сотрудник\_Квалификация\_Цех FOREIGN KEY (ID\_Цеха) REFERENCES Цех(ID)

);

## Разработка основных процедур

***Процедура: ПолучитьПодробностиИзделия***

• Описание: Получение подробной информации об изделии, включая его составные части и технологический маршрут.

• Параметры:

\* @ProductID INT - ID изделия.

• Логика:

1. Получает информацию об изделии из таблицы Изделие.

2. Получает составные части изделия из таблицы СоставИзделия.

3. Рекурсивно вызывает себя для составных частей изделия, если они тоже являются изделиями.

4. Получает этапы операций для каждого изделия.

CREATE PROCEDURE ПолучитьПодробностиИзделия

@ProductID INT

AS

BEGIN

-- Основная информация об изделии

SELECT

I.ID AS ProductID,

I.Наименование AS ProductName,

I.ID\_Родителя AS ParentProductID,

I.Цена,

C.Наименование AS Category

FROM Изделие I

LEFT JOIN КатегорияИзделия C ON I.ID\_Категории = C.ID

WHERE I.ID = @ProductID;

-- Составные части изделия

SELECT

CI.ID\_Ресурса AS ResourceID,

R.Наименование AS ResourceName,

CI.Количество AS Quantity,

EI.Наименование AS UnitName

FROM СоставИзделия CI

JOIN Ресурс R ON CI.ID\_Ресурса = R.ID

JOIN ЕдиницаИзмерения EI ON CI.ID\_ЕдиницыИзмерения = EI.ID

WHERE CI.ID\_Изделия = @ProductID;

-- Этапы операций

SELECT

EO.Порядок AS StageNumber,

EO.Описание AS StageDescription,

O.Наименование AS OperationName,

O.Описание AS OperationDescription,

O.ID AS OperationId

FROM ЭтапОперации EO

JOIN Операция O ON EO.ID\_Операции = O.ID

WHERE O.ID IN ( SELECT ID\_Операции

FROM ОпытРаботы

WHERE ID\_Операции IN ( SELECT ID FROM Операция ))

END;

-- Пример использования

-- EXEC ПолучитьПодробностиИзделия @ProductID = 1;

1. sp\_GetProductTypes

***Процедура: ПолучитьВсеКатегории***

• Описание: Получение списка всех типов изделий (категорий).

• Параметры: Нет

• Логика: Выбирает все записи из таблицы КатегорияИзделия.

CREATE PROCEDURE ПолучитьВсеКатегории

AS

BEGIN

SELECT ID, Наименование

FROM КатегорияИзделия;

END;

-- Пример использования

-- EXEC ПолучитьВсеКатегории;

***Процедура: РасчетСводныхНорм***

• Временные таблицы:

• #ResourcesToProcess содержит ресурсы для обработки, а так же родительский Id и количество, которое используется для составных ресурсов

• #FinalResources содержит конечные ресурсы и их количество.

• Цикл WHILE: Осуществляет итеративный обход иерархии ресурсов.

• Логика обработки: В цикле происходит обработка ресурса. Если у ресурса есть дочерние элементы, то они добавляются в таблицу на обработку, если нет, то ресурс добавляется в таблицу конечных ресурсов. Так же происходит проверка, если такой ресурс уже добавлен в конечную таблицу, то мы обновляем его количество.

• Возврат результатов: Выводится таблица с конечными ресурсами.

CREATE PROCEDURE РасчетСводныхНорм

@ProductID INT

AS

BEGIN

SET NOCOUNT ON;

-- Временная таблица для хранения ресурсов, которые нужно обработать

CREATE TABLE #ResourcesToProcess (

ResourceID INT,

ParentID INT NULL,

Quantity DECIMAL(10,2)

);

-- Временная таблица для хранения конечных ресурсов

CREATE TABLE #FinalResources (

ResourceID INT,

ResourceName VARCHAR(255),

TotalQuantity DECIMAL(10,2),

UnitName VARCHAR(50)

);

-- Добавляем ресурсы для данного изделия в #ResourcesToProcess

INSERT INTO #ResourcesToProcess (ResourceID, Quantity)

SELECT

ID\_Ресурса,

Количество

FROM СоставИзделия

WHERE ID\_Изделия = @ProductID;

-- Цикл для обхода иерархии ресурсов

WHILE (SELECT COUNT(\*) FROM #ResourcesToProcess) > 0

BEGIN

DECLARE @CurrentResourceID INT;

DECLARE @CurrentParentID INT;

DECLARE @CurrentQuantity DECIMAL(10,2);

-- Выбор первого ресурса из #ResourcesToProcess

SELECT TOP 1

@CurrentResourceID = ResourceID,

@CurrentParentID = ParentID,

@CurrentQuantity = Quantity

FROM #ResourcesToProcess;

IF @CurrentParentID IS NULL

BEGIN

-- Проверяем, есть ли дочерние ресурсы

IF EXISTS(SELECT 1 FROM Ресурс WHERE ID\_Родителя = @CurrentResourceID)

BEGIN

INSERT INTO #ResourcesToProcess (ResourceID, ParentID, Quantity)

SELECT

ID,

@CurrentResourceID,

@CurrentQuantity

FROM Ресурс

WHERE ID\_Родителя = @CurrentResourceID;

END

ELSE

BEGIN

-- Если дочерних ресурсов нет, добавляем в финальную таблицу

IF EXISTS (SELECT 1 FROM #FinalResources WHERE ResourceID = @CurrentResourceID)

BEGIN

UPDATE #FinalResources SET TotalQuantity = TotalQuantity + @CurrentQuantity WHERE ResourceID = @CurrentResourceID;

END

ELSE

BEGIN

INSERT INTO #FinalResources (ResourceID, ResourceName, TotalQuantity, UnitName)

SELECT R.ID, R.Наименование, @CurrentQuantity, EI.Наименование FROM Ресурс R

JOIN ЕдиницаИзмерения EI ON R.ID\_ЕдиницыИзмерения = EI.ID

WHERE R.ID = @CurrentResourceID;

END;

END

END

ELSE

BEGIN

-- Проверяем, есть ли дочерние ресурсы

IF EXISTS(SELECT 1 FROM Ресурс WHERE ID\_Родителя = @CurrentResourceID)

BEGIN

INSERT INTO #ResourcesToProcess (ResourceID, ParentID, Quantity)

SELECT

ID,

@CurrentResourceID,

@CurrentQuantity

FROM Ресурс

WHERE ID\_Родителя = @CurrentResourceID;

END

ELSE

BEGIN

-- Если дочерних ресурсов нет, добавляем в финальную таблицу

IF EXISTS (SELECT 1 FROM #FinalResources WHERE ResourceID = @CurrentResourceID)

BEGIN

UPDATE #FinalResources SET TotalQuantity = TotalQuantity + @CurrentQuantity WHERE ResourceID = @CurrentResourceID;

END

ELSE

BEGIN

INSERT INTO #FinalResources (ResourceID, ResourceName, TotalQuantity, UnitName)

SELECT R.ID, R.Наименование, @CurrentQuantity, EI.Наименование FROM Ресурс R

JOIN ЕдиницаИзмерения EI ON R.ID\_ЕдиницыИзмерения = EI.ID

WHERE R.ID = @CurrentResourceID;

END;

END;

END;

-- Удаляем обработанный ресурс из очереди

DELETE FROM #ResourcesToProcess WHERE ResourceID = @CurrentResourceID AND (ParentID = @CurrentParentID OR ParentID IS NULL);

END;

-- Вывод результатов

SELECT

ResourceName,

TotalQuantity,

UnitName

FROM #FinalResources;

-- Удаляем временные таблицы

DROP TABLE #FinalResources;

DROP TABLE #ResourcesToProcess;

END;

-- Пример использования

-- EXEC РасчетСводныхНорм @ProductID = 1;

***Процедура: ПолучитьТехнологическийМаршрут***

Работа процедуры (пошагово):

1. Создание временных таблиц:

\* #TechnologyRoute: Временная таблица для хранения результатов (технологического маршрута).

\* #ProcessedProducts: Временная таблица для отслеживания уже обработанных изделий в процессе рекурсивного обхода.

2. Инициализация:

\* Начальный уровень (@CurrentLevel) устанавливается в 0.

\* Счетчик обработанных строк (@RowsAffected) инициализируется нулём.

\* В таблицу #ProcessedProducts добавляется начальное изделие (предположительно жестко заданное или взятое из глобальной переменной/таблицы). Level у этого изделия равен 0.

3. Основной цикл WHILE:

\* Цикл выполняется, пока есть новые обработанные строки (@RowsAffected > 0). Этот цикл используется для рекурсивной обработки вложенных изделий.

\* Вставка данных в #TechnologyRoute:

\* Используется INSERT INTO #TechnologyRoute для сбора информации о ресурсах и операциях.

\* Извлекаются данные из таблиц Изделие, СоставИзделия, Ресурс, КлассРесурса, ЭтапОперации, Операция, Операция\_Оборудование, Оборудование, Операция\_Инструмент, Инструмент, Сотрудник\_Операция, Сотрудник, Профессия, Цех

\* Данные вставляются во временную таблицу #TechnologyRoute с текущим уровнем (pp.Level) и прочей информацией.

\* Добавление дочерних продуктов:

\* Используется INSERT INTO #ProcessedProducts для добавления новых дочерних продуктов, которые еще не были обработаны (WHERE NOT EXISTS...). Здесь и происходит рекурсивный обход

\* Здесь также проверяется условие AND res\_p.ID <> prod.ID, что исключает возможность рекурсии на одном и том же продукте.

\* Новые изделия добавляются с уровнем на 1 больше, чем у родительского изделия.

\* Обновление счетчиков:

\* Обновляется счетчик обработанных строк (@RowsAffected) с помощью @@ROWCOUNT.

\* Увеличивается текущий уровень (@CurrentLevel).

4. Вывод результата:

\* Выбираются данные из таблицы #TechnologyRoute.

\* Добавляется условие WHERE NOT (РасходРесурса IS NULL AND ВыходнойРесурс IS NULL), исключающее строки без фактического расхода или выработки ресурсов.

\* Результат сортируется по уровню (Level).

**Как передаются ресурсы в процедуру:**

- Явно ресурсы в процедуру не передаются, а используются данные из таблиц Изделие, СоставИзделия и Ресурс.

- Входным параметром процедуры является ID изделия, для которого нужно построить технологический маршрут.

- Затем процедура, используя связи в таблицах, определяет необходимые ресурсы.

CREATE PROCEDURE [dbo].[ПолучитьТехнологическийМаршрут]

@ProductID INT,

@CurrentDate DATE = NULL

AS

BEGIN

IF @CurrentDate IS NULL

SET @CurrentDate = GETDATE()

-- Таблица для хранения результатов

CREATE TABLE #TechnologyRoute (

Level INT,

Операция NVARCHAR(255),

Инструмент NVARCHAR(255),

Оборудование NVARCHAR(255),

Цех NVARCHAR(255),

Мастер NVARCHAR(255),

Квалификация NVARCHAR(255),

РасходРесурса NVARCHAR(255),

РасходРесурсаКоличество DECIMAL(18,2),

ВыходнойРесурс NVARCHAR(255),

ВыходнойРесурсКоличество DECIMAL(18,2),

ДочернийПродукт NVARCHAR(255),

КлассРесурса NVARCHAR(255),

Количество INT,

ProductID INT

);

-- Вспомогательная таблица для отслеживания обработки

CREATE TABLE #ProcessedProducts (

ProductID INT,

Level INT,

ParentProductID INT

);

-- Добавляем корневой продукт, с уровнем 0

INSERT INTO #ProcessedProducts (ProductID, Level, ParentProductID)

VALUES (@ProductID, 0, NULL)

DECLARE @CurrentLevel INT = 0;

DECLARE @RowsAffected INT = 1;

-- Итерация по уровням, пока есть что обрабатывать

WHILE @RowsAffected > 0

BEGIN

INSERT INTO #TechnologyRoute (

Level,

Операция,

Инструмент,

Оборудование,

Цех,

Мастер,

Квалификация,

РасходРесурса,

РасходРесурсаКоличество,

ВыходнойРесурс,

ВыходнойРесурсКоличество,

ДочернийПродукт,

КлассРесурса,

Количество,

ProductID

)

SELECT

pp.Level +1,

o.Наименование AS Операция,

i.Наименование AS Инструмент,

eq.Наименование AS Оборудование,

wh.Наименование AS Цех,

s.ФИО AS Мастер,

p.Наименование AS Квалификация,

res\_used.Наименование AS РасходРесурса,

ISNULL(rr.Количество,0) AS РасходРесурсаКоличество,

res\_made.Наименование AS ВыходнойРесурс,

ISNULL(wr.Количество,0) AS ВыходнойРесурсКоличество,

res\_p.Наименование AS ДочернийПродукт,

cr.Наименование AS КлассРесурса,

si.Количество AS Количество,

prod.ID AS ProductID

FROM

#ProcessedProducts pp

INNER JOIN Изделие prod ON pp.ProductID = prod.ID

INNER JOIN СоставИзделия si ON prod.ID = si.ID\_Изделия

INNER JOIN Ресурс res\_p ON si.ID\_Ресурса = res\_p.ID

INNER JOIN КлассРесурса cr ON res\_p.ID\_Класса = cr.ID

INNER JOIN ЭтапОперации ep ON 1=1

INNER JOIN Операция o ON ep.ID\_Операции = o.ID

LEFT JOIN ОперацияИнструмент oi ON o.ID = oi.ID\_Операции

LEFT JOIN Инструмент i ON oi.ID\_Инструмента = i.ID

LEFT JOIN ОперацияОборудование oe ON o.ID = oe.ID\_Операции

LEFT JOIN Оборудование eq ON oe.ID\_Оборудования = eq.ID

LEFT JOIN Цех wh ON o.ID\_Цеха = wh.ID

LEFT JOIN РасходРесурсов rr ON ep.ID = rr.ID\_Этапа and rr.ID\_Ресурса = res\_p.ID

LEFT JOIN ВыработкаРесурсов wr ON ep.ID = wr.ID\_Этапа and wr.ID\_Ресурса = res\_p.ID

LEFT JOIN Ресурс res\_used ON rr.ID\_Ресурса = res\_used.ID

LEFT JOIN Ресурс res\_made ON wr.ID\_Ресурса = res\_made.ID

-- Связываем таблицы для получения имени мастера и его квалификации

LEFT JOIN Сотрудник\_Операция so ON o.ID = so.ID\_Операции

LEFT JOIN Сотрудник s ON so.ID\_Сотрудника = s.ID

LEFT JOIN Сотрудник\_Квалификации sk ON s.ID = sk.ID\_Сотрудника AND o.ID\_Цеха = sk.ID\_Цеха

LEFT JOIN Профессия p ON o.ID\_Профессии = p.ID

WHERE NOT EXISTS (SELECT 1 FROM #ProcessedProducts pp2 WHERE pp2.ProductID = res\_p.ID AND pp2.Level > pp.Level)

AND res\_p.ID <> prod.ID

AND so.ДатаНачала <= @CurrentDate AND (so.ДатаОкончания IS NULL OR so.ДатаОкончания >= @CurrentDate)

AND sk.ДатаНачала <= @CurrentDate AND (sk.ДатаОкончания IS NULL OR sk.ДатаОкончания >= @CurrentDate)

-- Добавляем новые дочерние продукты для обработки на следующем уровне

INSERT INTO #ProcessedProducts (ProductID, Level, ParentProductID)

SELECT res\_p.ID, pp.Level +1 , pp.ProductID

FROM

#ProcessedProducts pp

INNER JOIN Изделие prod ON pp.ProductID = prod.ID

INNER JOIN СоставИзделия si ON prod.ID = si.ID\_Изделия

INNER JOIN Ресурс res\_p ON si.ID\_Ресурса = res\_p.ID

WHERE NOT EXISTS (SELECT 1 FROM #ProcessedProducts pp2 WHERE pp2.ProductID = res\_p.ID)

AND res\_p.ID <> prod.ID

-- Обновляем счетчик

SET @RowsAffected = @@ROWCOUNT;

SET @CurrentLevel = @CurrentLevel + 1;

END;

-- Выводим результат, добавляем условие where

SELECT

Level,

Операция,

Инструмент,

Оборудование,

Цех,

Мастер,

Квалификация,

РасходРесурса,

РасходРесурсаКоличество,

ВыходнойРесурс,

ВыходнойРесурсКоличество,

ДочернийПродукт,

КлассРесурса,

Количество,

ProductID

FROM

#TechnologyRoute

WHERE NOT (РасходРесурса IS NULL AND ВыходнойРесурс IS NULL)

ORDER BY Level

-- Удаляем временные таблицы

DROP TABLE #TechnologyRoute;

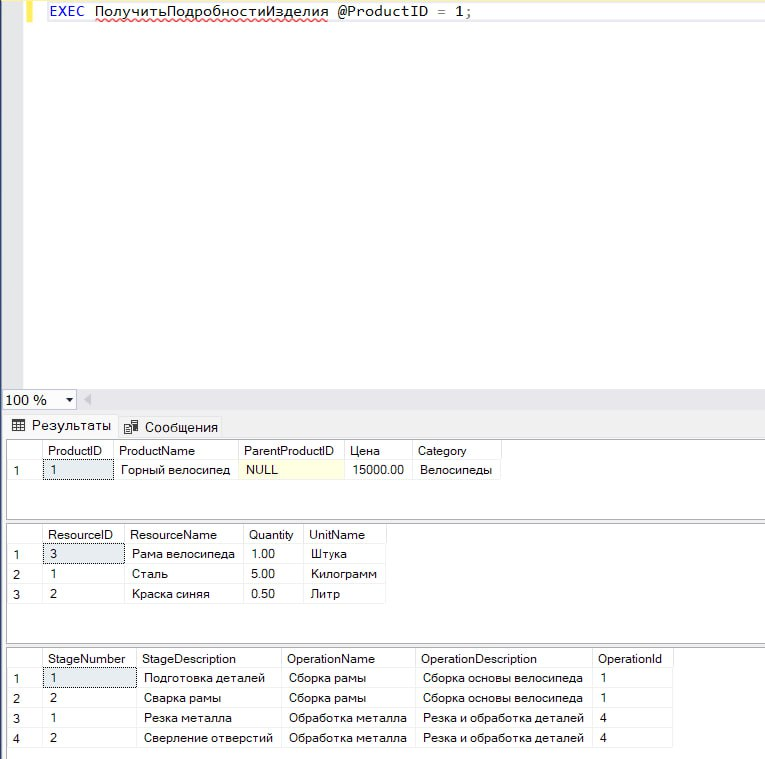
DROP TABLE #ProcessedProducts;

END;

## Тестирование разработки

1. Ведение справочника изделий различных типов, используемых в составе изделий

*Таблица 1 – Результат работы процедуры ПолучитьПодробностиИзделия*



2. Получение всех категорий заданного изделия

*Таблица 2 – Результат работы процедуры ПолучитьВсеКатегории*



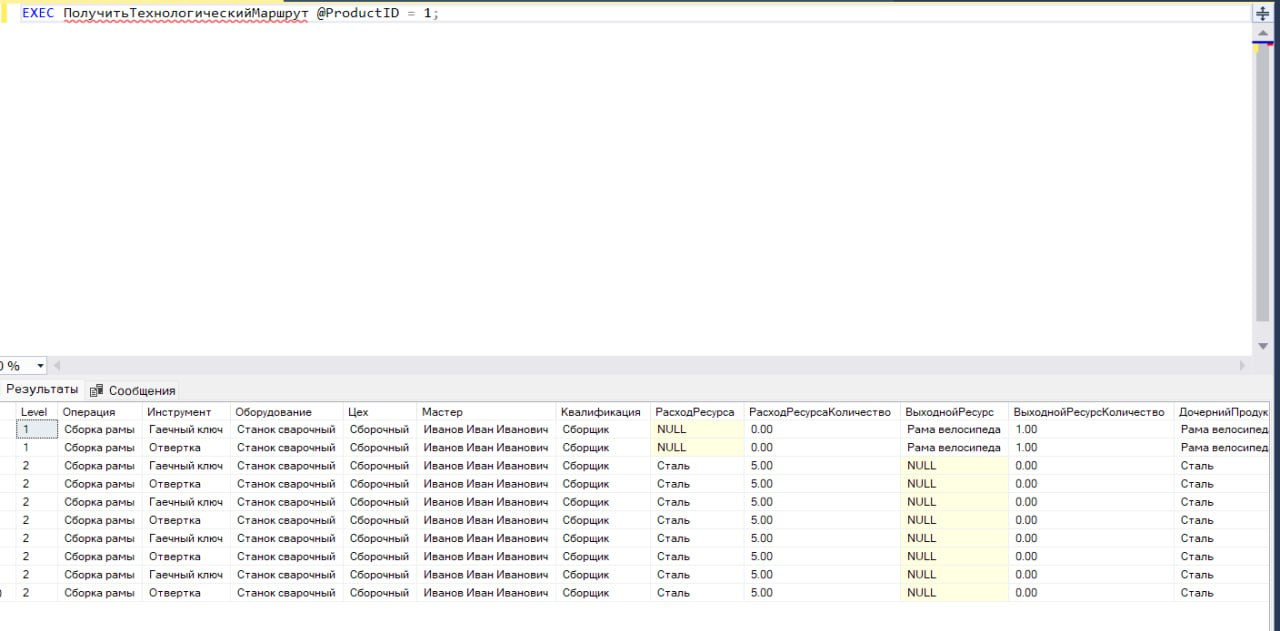
## 3. Расчет сводных норм расхода компонентов маршрутной спецификации с указанием потребности в трудовых и материальных ресурсов заданного класса.

*Таблица 3 - Результат работы процедуры РасчетСводныхНорм*

## 

## 4. Поиск всех строк технологического маршрута изделия на всю глубину вложенности;

*Таблица 4 - Результат работы процедуры ПолучитьТехнологическийМаршрут*



# Вывод

В данной работе был разработан проект каркаса для работы с маршрутной технологией изготовления изделий. Реализация произведена в базе данных MS SQL. Проведено тестирование, показывающее корректную работу реализованных функций для работы с данными.