

**МИНОБРНАУКИ РОССИИ**  
**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**  
**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**  
**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**  
**Кафедра САПР**

**ИНДИВИДУАЛЬНОЕ ДОМАШНЕЕ ЗАДАНИЕ**  
**по дисциплине «Базы данных»**  
**ТЕМА: ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ**

Студентка гр. 1308

\_\_\_\_\_

Черникова П.В.

Преподаватель

\_\_\_\_\_

Новакова Н.Е.

Санкт-Петербург

2023

## ИНДИВИДУАЛЬНОЕ ДОМАШНЕЕ ЗАДАНИЕ

Студентка Черникова П.В.

Группа 1308

Тема работы: Проектирование базы данных

Исходные данные:

Разработать БД для контроля использования веществ-прекурсоров в химической лаборатории.

Содержание пояснительной записки: «Введение», «Цель работы», «Задание», «Предметная область», «Проектирование базы данных», «Создание базы данных», «Создание таблиц», «Заполнение таблиц данными», «Разработка объектов промежуточного слоя», «Разработка стратегии резервного копирования», «Заключение», «Список использованных источников».

Предполагаемый объем пояснительной записки:

Не менее 20 страниц.

Дата выдачи задания: 28.09.2023

Дата сдачи задания: 00.12.2023

Дата защиты задания: 00.12.2023

Студентка

\_\_\_\_\_

Черникова П.В.

Преподаватель

\_\_\_\_\_

Новакова Н.Е.

## СОДЕРЖАНИЕ

Введение	4
Цель работы	5
1. Задание	6
2. Предметная область	6
3. Проектирование базы данных	6
4. Создание базы данных	9
5. Создание таблиц	10
6. Заполнение таблиц данными	17
7. Разработка объектов промежуточного слоя	24
Хранимые процедуры	24
UDF	31
Представления	35
8. Разработка стратегии резервного копирования	40
9. Заключение	43
10. Список использованных источников	44

## **Введение**

Для выполнения данного индивидуального домашнего задания требуется спроектировать, а затем создать базу данных согласно выбранной предметной области, а также разработать для неё объекты промежуточного слоя и продемонстрировать полученные результаты.

Выполнение данного задания позволит на практике закрепить знания и навыки, полученные на лекционных и практических занятиях в рамках пройденной дисциплины “Базы данных”, а также продемонстрировать их применение в процессе проектирования базы.

## **Цель работы**

Целью данного индивидуального домашнего задания является проектирование и создание базы данных, соответствующей выбранной предметной области. Помимо создания самой базы, схемы, таблиц и их заполнения в ходе выполнения задания необходимо разработать разнообразные виды объектов промежуточного слоя, таких как хранимые процедуры, UDF и представления. Привести иллюстрации, подтверждающие корректное выполнение запросов. Кроме того, в данной работе следует также предусмотреть и описать стратегию резервного копирования базы данных.

## **1. Задание**

Разработать БД для контроля использования веществ-прекурсоров в химической лаборатории в ходе следующих этапов работы:

1. Проектирование БД (структура данных)
2. Создание БД
3. Создание таблиц и ограничений целостности
4. Заполнение таблиц данным
5. Разработка объектов промежуточного слоя (представлений, хранимых процедур, UDF-ов)
6. Разработка стратегии резервного копирования

## **2. Предметная область**

Прекурсоры — вещества, часто используемые при производстве, изготовлении, переработке наркотических средств, включенных в Перечень наркотических средств, психотропных веществ и их прекурсоров, подлежащих контролю в Российской Федерации. Однако, эти вещества также часто требуется задействовать при проведении реакций для выполнения широкого ряда анализов в химических лабораториях различного направления. В связи с этим, в лабораториях необходимо обеспечивать тщательный контроль расходов веществ-прекурсоров, чтобы закупаемые объемы таких реактивов были использованы исключительно по своему назначению, описанному в действующих методиках химических анализов.

## **3. Проектирование базы данных**

В базе данных представлены следующие таблицы:

1. ReagentSuppliers — таблица, содержащая информацию о поставщиках реактивов для удобства их своевременного заказа. Таблица содержит: название компании, контактный телефон и электронную почту. Первичный ключ — SupplierID.
2. Reagents — таблица, содержащая информацию о прекурсорах как о реактивах, имеющих характеристики, соответствующие требованиям в

действующих методиках выполнения анализов. Таблица содержит: название химического вещества прекурсора, чистоту реактива и его концентрацию. Чистота реактива для веществ в виде ампул стандарт-титров не указывается. Первичный ключ — ReagentID.

3. Precursors — таблица, содержащая информацию о физических экземплярах прекурсоров в тарах и их характеристиках. Таблица содержит: дату изготовления, дату истечения срока годности, количество прекурсора (г/мл/шт. (для ампул)), единицы измерения этого количества и статус использования: “израсх.” — сообщаемая пользователем метка о том, что прекурсор в таре израсходован полностью и далее требует подтверждения соответствия расхода, “соотв.” — метка о том, что количество прекурсора, использованное для анализов, соответствует изначальному и “не соотв.”, если иначе. Статус использования незавершенного прекурсора не указывается. Первичным ключом является PrecursorID. Внешние ключи: SupplierID и ReagentID.

4. Analyses — таблица, содержащая информацию о химических анализах, в которых используются прекурсоры. Таблица содержит: краткое название анализа, зачастую — это определяемое вещество (напр. содержание сульфатов в исследуемом образце) или исследуемое свойство образца (напр. его перманганатная окисляемость), а также номер действующей методики исполнения анализа. Первичным ключом является AnalysisID.

5. Chemists — таблица, содержащая информацию о химиках, работающих в лаборатории и исполняющих анализы в соответствии с методиками. Таблица содержит: фамилию, имя и отчество сотрудника, которое может быть не указано. Первичным ключом является EmployeeID.

6. PrecursorConsumptions — таблица, содержащая информацию о расходах прекурсоров. Таблица содержит: количество взятого для анализа вещества и дату использования. Первичным ключом является ConsumptionID. Внешние ключи: PrecursorID, AnalysisID и EmployeeID.

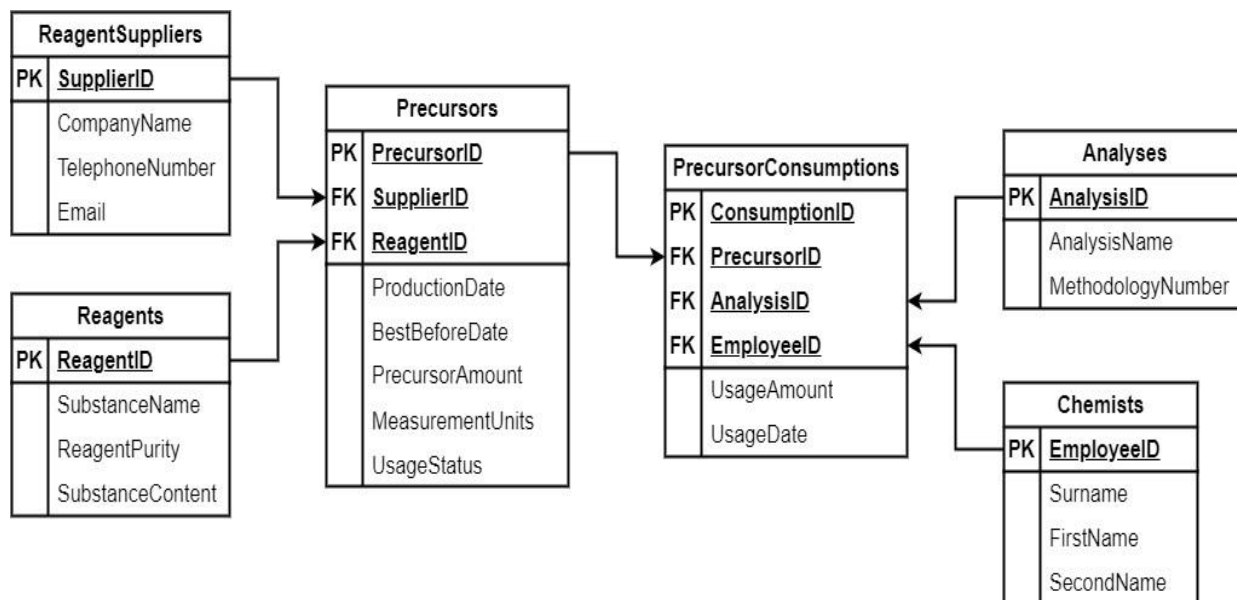


Рисунок 1. Диаграмма сущность—связь.



#### 4. Создание базы данных

Создадим базу данных PrecursorControl с определенными параметрами файлов данных и журнала транзакций. Файл для хранения данных и файл для хранения журнала транзакций в ходе работы будут располагаться на одном диске C:\, однако в примере ниже продемонстрируем возможность создания этих файлов с использованием нескольких дисков: C:\ и D:\.

```
CREATE DATABASE PrecursorControl ON  
(  
name = PrecursorControl_dat,  
filename = 'C:\BD_IDZ\PrecursorControl.mdf',  
size = 5MB,  
maxsize = 50MB,  
filegrowth = 5MB  
)  
log ON  
(  
name = PrecursorControl_log,  
filename = 'D:\BD_IDZ\PrecursorControl.ldf',  
size = 5MB,  
maxsize = 50MB,  
filegrowth = 5MB  
);
```

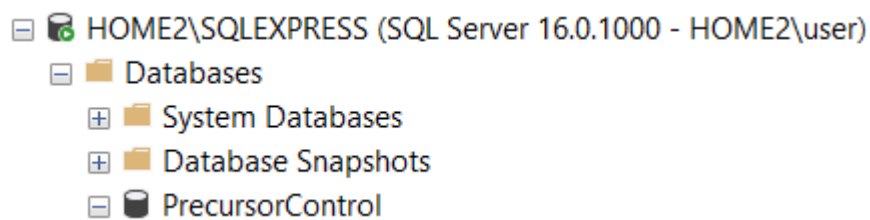


Рисунок 2. Результат создания базы данных.

Создадим отдельную схему — precursor control (pc).

```
USE PrecursorControl  
GO  
CREATE SCHEMA pc
```

## 5. Создание таблиц

Таблица 1. Описание таблицы ReagentSuppliers.

Описание структуры таблицы БД		Наименование таблицы БД: Таблица поставщики реактивов		Имя таблицы: ReagentSuppliers		
Дата разработки: 00.12.2023						
Порядковый номер таблицы: 1						
№ п/п	Наименование поля	Спецификация данных				
		Имя поля	Тип данных	Ключ	Ограничения целостности	
	1	Идентификатор поставщика	SupplierID	INT	P	NOT NULL
	2	Название компании	CompanyName	NVARCHAR(50)		NOT NULL UNIQUE
	3	Телефон	TelephoneNumber	NVARCHAR(11)		NOT NULL UNIQUE
	4	Электронная почта	Email	NVARCHAR(50)		NOT NULL UNIQUE

Таблица 2. Описание таблицы Reagents.

Описание структуры таблицы БД	Наименование таблицы БД: Таблица химические реактивы	Имя таблицы: Reagents
Дата разработки: 00.12.2023		
Порядковый номер таблицы: 2		

№ п/п	Наименование поля	Спецификация данных			
		Имя поля	Тип данных	Ключ	Ограничения целостности
1	Идентификатор реактива	ReagentID	INT	P	NOT NULL
2	Название вещества	SubstanceName	NVARCHAR(40)		NOT NULL
3	Содержание вещества	SubstanceContent	NVARCHAR(10)		NOT NULL
4	Чистота реактива	ReagentPurity	NVARCHAR(3)		NULL

Таблица 3. Описание таблицы Precursors.

Описание структуры таблицы БД		Наименование таблицы БД: <b>Таблица прекурсоры</b>	Имя таблицы: Precursors
Дата разработки: 00.12.2023			
Порядковый номер таблицы: <b>3</b>			

№ п/п	Наименование поля	Спецификация данных			
		Имя поля	Тип данных	Ключ	Ограничения целостности
1	Идентификатор прекурсора	PrecursorID	INT	P	NOT NULL
2	Идентификатор поставщика	SupplierID	INT	F	NOT NULL
3	Идентификатор реактива	ReagentID	INT	F	NOT NULL
4	Дата изготовления	ProductionDate	DATE		NOT NULL
5	Дата истечения срока годности	BestBeforeDate	DATE		NOT NULL
6	Количество вещества в упаковке (г/мл/шт.)	PrecursorAmount	INT		NOT NULL PrecursorAmount > 0
7	Статус использования	UsageStatus	NVARCHAR(10)		NULL

Таблица 4. Описание таблицы Analyses.

Описание структуры таблицы БД		Наименование таблицы БД: Таблица химические анализы		Имя таблицы: Analyses	
Дата разработки: 00.12.2023					
Порядковый номер таблицы: 4					
№ п/п	Наименование поля	Спецификация данных			
		Имя поля	Тип данных	Ключ	Ограничения целостности
1	Идентификатор анализа	AnalysisID	INT	P	NOT NULL
2	Краткое название анализа	AnalysisName	NVARCHAR(100)		NOT NULL
3	Номер методики выполнения анализа	MethodologyNumber	NVARCHAR(40)		NOT NULL UNIQUE

Таблица 5. Описание таблицы Chemists.

Описание структуры таблицы БД	Наименование таблицы БД: <b>Таблица химики</b>	Имя таблицы: Chemists
Дата разработки: 00.12.2023		
Порядковый номер таблицы: <b>5</b>		

№ п/п	Наименование поля	Спецификация данных			
		Имя поля	Тип данных	Ключ	Ограничения целостности
1	Идентификатор сотрудника	EmployeeID	INT	P	NOT NULL
2	Фамилия	Surname	NVARCHAR(30)		NOT NULL
3	Имя	FirstName	NVARCHAR(30)		NOT NULL
4	Отчество	SecondName	NVARCHAR(30)		NULL

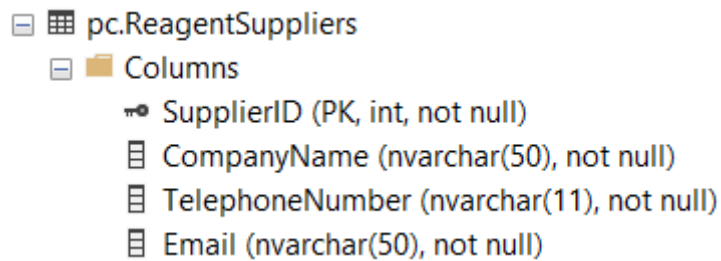
Таблица 6. Описание таблицы PrecursorConsumptions.

Описание структуры таблицы БД		Наименование таблицы БД: Таблица расходы прекурсоров		Имя таблицы: PrecursorConsumptions	
Дата разработки: 00.12.2023					
Порядковый номер таблицы: 6					
№ п/п	Наименование поля	Спецификация данных			
		Имя поля	Тип данных	Ключ	Ограничения целостности
1	Идентификатор расхода	ConsumptionID	INT	P	NOT NULL
2	Идентификатор прекурсора	PrecursorID	INT	F	NOT NULL
3	Идентификатор анализа	AnalysisID	INT	F	NOT NULL
4	Идентификатор химика- исполнителя анализа	EmployeeID	INT	F	NOT NULL
5	Использованное количество (г/мл/шт.)	UsageAmount	INT		NOT NULL UsageAmount > 0
6	Дата использования	UsageDate	Date		NOT NULL

В таблицах ограничением целостности данных преимущественно является NOT NULL, однако в некоторых столбцах необходимо предусмотреть возможное отсутствие информации. Также используем ключевое слово UNIQUE для определения ограничения у некоторых столбцов таблиц, чтобы все значения в таких столбцах были уникальными в пределах таблицы. Применим к некоторым столбцам ограничение CHECK с условием, чтобы значения в них были строго больше нуля.

Создадим спроектированные таблицы.

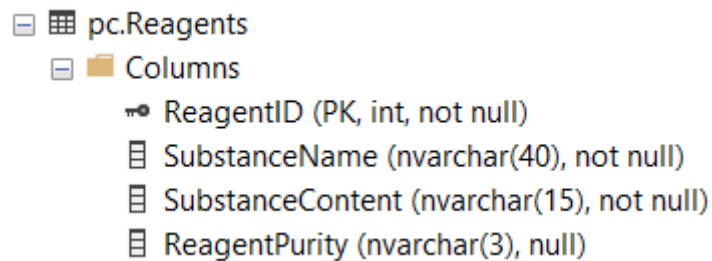
```
CREATE TABLE pc.ReagentSuppliers(
SupplierID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
CompanyName NVARCHAR(50) NOT NULL UNIQUE,
TelephoneNumber NVARCHAR(11) NOT NULL UNIQUE,
Email NVARCHAR(50) NOT NULL UNIQUE);
```



pc.ReagentSuppliers
Columns
SupplierID (PK, int, not null)
CompanyName (nvarchar(50), not null)
TelephoneNumber (nvarchar(11), not null)
Email (nvarchar(50), not null)

Рисунок 3. Результат создания таблицы ReagentSuppliers.

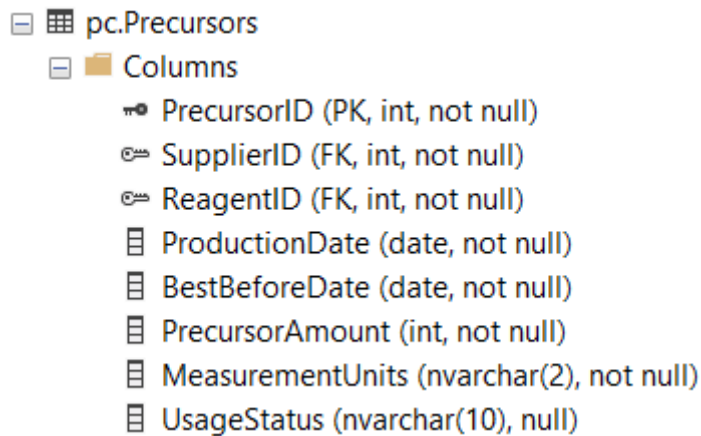
```
CREATE TABLE pc.Reagents(
ReagentID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
SubstanceName NVARCHAR(40) NOT NULL,
SubstanceContent NVARCHAR(15) NOT NULL,
ReagentPurity NVARCHAR(3) NULL);
```



pc.Reagents
Columns
ReagentID (PK, int, not null)
SubstanceName (nvarchar(40), not null)
SubstanceContent (nvarchar(15), not null)
ReagentPurity (nvarchar(3), null)

Рисунок 4. Результат создания таблицы Reagents.

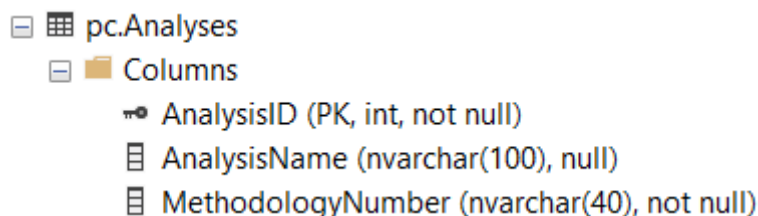
```
CREATE TABLE pc.Precursors(
PrecursorID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
SupplierID INT
REFERENCES pc.ReagentSuppliers(SupplierID) NOT NULL,
ReagentID INT REFERENCES pc.Reagents(ReagentID) NOT NULL,
ProductionDate DATE NOT NULL,
BestBeforeDate DATE NOT NULL,
PrecursorAmount INT NOT NULL CHECK (PrecursorAmount > 0),
MeasurementUnits NVARCHAR(2) NOT NULL,
UsageStatus NVARCHAR(10) NULL);
```



pc.Precursors
Columns
PrecursorID (PK, int, not null)
SupplierID (FK, int, not null)
ReagentID (FK, int, not null)
ProductionDate (date, not null)
BestBeforeDate (date, not null)
PrecursorAmount (int, not null)
MeasurementUnits (nvarchar(2), not null)
UsageStatus (nvarchar(10), null)

Рисунок 5. Результат создания таблицы Precursors.

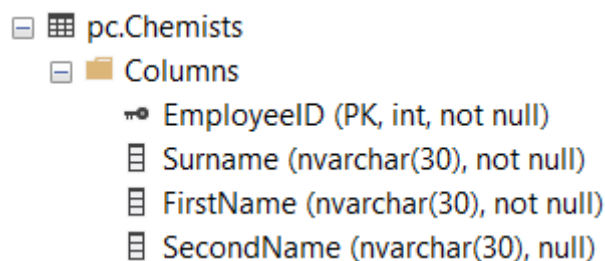
```
CREATE TABLE pc.Analyses(
AnalysisID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
AnalysisName NVARCHAR(40) NOT NULL,
MethodologyNumber NVARCHAR(40) NOT NULL UNIQUE);
```



pc.Analyses
Columns
AnalysisID (PK, int, not null)
AnalysisName (nvarchar(100), null)
MethodologyNumber (nvarchar(40), not null)

Рисунок 6. Результат создания таблицы Analyses.

```
CREATE TABLE pc.Chemists(
EmployeeID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
Surname NVARCHAR(30) NOT NULL,
FirstName NVARCHAR(30) NOT NULL,
SecondName NVARCHAR(30) NULL);
```



pc.Chemists
Columns
EmployeeID (PK, int, not null)
Surname (nvarchar(30), not null)
FirstName (nvarchar(30), not null)
SecondName (nvarchar(30), null)

Рисунок 7. Результат создания таблицы Chemists.

```
CREATE TABLE pc.PrecursorConsumptions(
ConsumptionID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
PrecursorID INT REFERENCES pc.Precursors(PrecursorID) NOT NULL,
AnalysisID INT REFERENCES pc.Analyses(AnalysisID) NOT NULL,
EmployeeID INT REFERENCES pc.Chemists(EmployeeID) NOT NULL,
```

```
UsageAmount INT NOT NULL CHECK (UsageAmount > 0),  
UsageDate DATE NOT NULL);
```

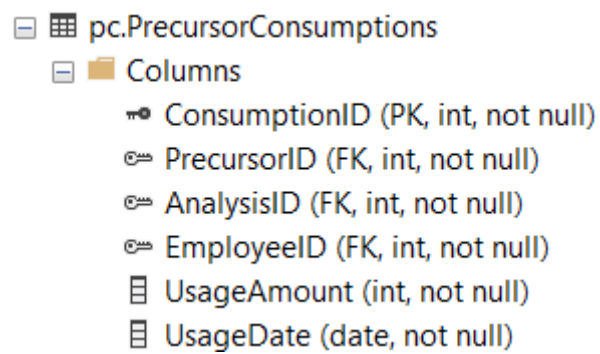


Рисунок 8. Результат создания таблицы `PrecursorConsumptions`.



## 6. Заполнение таблиц данными

```
INSERT pc.ReagentSuppliers (CompanyName, TelephoneNumber, Email)
VALUES ('НеваРеактив', '88123254065', 'office@nevareaktiv.ru');
INSERT pc.ReagentSuppliers (CompanyName, TelephoneNumber, Email)
VALUES ('ЛенРеактив', '88122239637', 'info@lenreactiv.ru');
INSERT pc.ReagentSuppliers (CompanyName, TelephoneNumber, Email)
VALUES ('Формула', '88129493420', 'Store@formula-re.ru');
INSERT pc.ReagentSuppliers (CompanyName, TelephoneNumber, Email)
VALUES ('Ареолаб', '89217780442', 'areolab@yandex.ru');
INSERT pc.ReagentSuppliers (CompanyName, TelephoneNumber, Email)
VALUES ('БалтХим', '88124541466', 'mail@baltchim.ru');
```

```
SELECT SupplierID, CompanyName, TelephoneNumber, Email
FROM pc.ReagentSuppliers;
```

	SupplierID	CompanyName	TelephoneNumber	Email
1	1	НеваРеактив	88123254065	office@nevareaktiv.ru
2	2	ЛенРеактив	88122239637	info@lenreactiv.ru
3	3	Формула	88129493420	Store@formula-re.ru
4	4	Ареолаб	89217780442	areolab@yandex.ru
5	5	БалтХим	88124541466	mail@baltchim.ru

Рисунок 9. Результат заполнения таблицы ReagentSuppliers.

```
INSERT pc.Reagents (SubstanceName, SubstanceContent, ReagentPurity)
VALUES ('Серная кислота', '94%', 'хч');
INSERT pc.Reagents (SubstanceName, SubstanceContent, ReagentPurity)
VALUES ('Серная кислота', '0,1Н', '');
INSERT pc.Reagents (SubstanceName, SubstanceContent, ReagentPurity)
VALUES ('Соляная кислота', '0,1Н', '');
INSERT pc.Reagents (SubstanceName, SubstanceContent, ReagentPurity)
VALUES ('Ацетон', '99,75%', 'осч');
INSERT pc.Reagents (SubstanceName, SubstanceContent, ReagentPurity)
VALUES ('Толуол', '99%', 'тех');
INSERT pc.Reagents (SubstanceName, SubstanceContent, ReagentPurity)
VALUES ('1,4-Бутандиол', '99%', 'ч');
```

```

INSERT pc.Reagents (SubstanceName, SubstanceContent, ReagentPurity)
VALUES ('Диэтиловый эфир', '99,5%', 'чда');
INSERT pc.Reagents (SubstanceName, SubstanceContent, ReagentPurity)
VALUES ('Соляная кислота', '35%', 'хч');
INSERT pc.Reagents (SubstanceName, SubstanceContent, ReagentPurity)
VALUES ('Пиперидин', '99%', 'хч');
INSERT pc.Reagents (SubstanceName, SubstanceContent, ReagentPurity)
VALUES ('Кислота серная', '99%', 'осч');
INSERT pc.Reagents (SubstanceName, SubstanceContent, ReagentPurity)
VALUES ('Калий марганцовокислый', '99,5%', 'чда');

SELECT ReagentID, SubstanceName, SubstanceContent, ReagentPurity
FROM pc.Reagents;

```

	ReagentID	SubstanceName	SubstanceContent	ReagentPurity
1	1	Серная кислота	94%	хч
2	2	Серная кислота	0,1Н	
3	3	Соляная кислота	0,1Н	
4	4	Ацетон	99,75%	осч
5	5	Толуол	99%	тех
6	6	1,4-Бутандиол	99%	ч
7	7	Диэтиловый эфир	99,5%	чда
8	8	Соляная кислота	35%	хч
9	9	Пиперидин	99%	хч
10	10	Кислота серная	99%	осч
11	11	Калий марганцовокислый	99,5%	чда

Рисунок 10. Результат заполнения таблицы Reagents.

```

INSERT pc.Analyses (AnalysisName, MethodologyNumber)
VALUES ('Нитраты', 'ПНД Ф 14.1 2 4.4-95');
INSERT pc.Analyses (AnalysisName, MethodologyNumber)
VALUES ('Сульфаты', 'ПНД Ф 14.1 2 3.108-97');
INSERT pc.Analyses (AnalysisName, MethodologyNumber)
VALUES ('Перманганатная окисляемость', 'ПНД Ф 14.1 2 4.154-99');
INSERT pc.Analyses (AnalysisName, MethodologyNumber)
VALUES ('Ацетон и метанол', 'ПНД Ф 14.1:2:4.201-03');
INSERT pc.Analyses (AnalysisName, MethodologyNumber)
VALUES ('Пиперидин', 'МУК 4.1.153-96');
INSERT pc.Analyses (AnalysisName, MethodologyNumber)

```

```

VALUES ('Фосфаты', 'ПНД Ф 16.1:2:2.2:3.52-08');
INSERT pc.Analyses (AnalysisName, MethodologyNumber)
VALUES ('Мышьяк', 'ПНД Ф 16.1:2:2.3.16-98');
INSERT pc.Analyses (AnalysisName, MethodologyNumber)
VALUES ('Гидрокарбонаты', 'ПНДФ 14.1:2:3.99-97');
INSERT pc.Analyses (AnalysisName, MethodologyNumber)
VALUES ('Бензол', 'МУК 4.1.3167– 14');
INSERT pc.Analyses (AnalysisName, MethodologyNumber)
VALUES ('Ацетон', 'РД 52.24.506-2009');

SELECT AnalysisID, AnalysisName, MethodologyNumber
FROM pc.Analyses;

```

	AnalysisID	AnalysisName	MethodologyNumber
1	1	Нитраты	ПНД Ф 14.1 2 4.4-95
2	2	Сульфаты	ПНД Ф 14.1 2 3.108-97
3	3	Перманганатная окисляемость	ПНД Ф 14.1 2 4.154-99
4	4	Ацетон и метанол	ПНД Ф 14.1:2:4.201-03
5	5	Пиперидин	МУК 4.1.153-96
6	6	Фосфаты	ПНД Ф 16.1:2:2.2:3.52-08
7	7	Мышьяк	ПНД Ф 16.1:2:2.3.16-98
8	8	Гидрокарбонаты	ПНДФ 14.1:2:3.99-97
9	9	Бензол	МУК 4.1.3167– 14
10	10	Ацетон	РД 52.24.506-2009

Рисунок 11. Результат заполнения таблицы Analyses.

```

INSERT pc.Chemists(Surname, FirstName, SecondName)
VALUES ('Кузнецова', 'Ирина', 'Олеговна');
INSERT pc.Chemists(Surname, FirstName, SecondName)
VALUES ('Садриева', 'Вера', '');
INSERT pc.Chemists(Surname, FirstName, SecondName)
VALUES ('Жукова', 'Марьяна', '');
INSERT pc.Chemists(Surname, FirstName, SecondName)
VALUES ('Белозеров', 'Филипп', 'Аркадьевич');
INSERT pc.Chemists(Surname, FirstName, SecondName)
VALUES ('Кузьмин', 'Александр', 'Валерьевич');
INSERT pc.Chemists(Surname, FirstName, SecondName)
VALUES ('Дакуо', 'Кристиан', '');
INSERT pc.Chemists(Surname, FirstName, SecondName)

```

```
VALUES ('Рахматова', 'Барно', '');
INSERT pc.Chemists(Surname, FirstName, SecondName)
VALUES ('Петрова', 'Ксения', 'Александровна');

SELECT EmployeeID, Surname, FirstName, SecondName
FROM pc.Chemists;
```

	EmployeeID	Surname	FirstName	SecondName
1	1	Кузнецова	Ирина	Олеговна
2	2	Садриева	Вера	
3	3	Жукова	Марьяна	
4	4	Белозеров	Филипп	Аркадьевич
5	5	Кузьмин	Александр	Валерьевич
6	6	Дакуо	Кристиан	
7	7	Рахматова	Барно	
8	8	Петрова	Ксения	Александровна

Рисунок 12. Результат заполнения таблицы Chemists.

```
INSERT pc.Precursors(SupplierID, ReagentID,
ProductionDate, BestBeforeDate, PrecursorAmount, MeasurementUnits,
UsageStatus)
VALUES (1, 1, '2023-06-29', '2025-06-29', 1000, 'мл', '');
INSERT pc.Precursors(SupplierID, ReagentID,
ProductionDate, BestBeforeDate, PrecursorAmount, MeasurementUnits,
UsageStatus)
VALUES (1, 1, '2023-06-29', '2025-06-29', 1000, 'мл', '');
INSERT pc.Precursors(SupplierID, ReagentID,
ProductionDate, BestBeforeDate, PrecursorAmount, MeasurementUnits,
UsageStatus)
VALUES (1, 1, '2023-06-29', '2025-06-29', 1000, 'мл', '');
INSERT pc.Precursors(SupplierID, ReagentID,
ProductionDate, BestBeforeDate, PrecursorAmount, MeasurementUnits,
UsageStatus)
VALUES (2, 2, '2022-11-06', '2023-11-06', 10, 'шт', '');
INSERT pc.Precursors(SupplierID, ReagentID,
ProductionDate, BestBeforeDate, PrecursorAmount, MeasurementUnits,
UsageStatus)
VALUES (3, 3, '2023-10-11', '2024-04-11', 5, 'шт', 'израсх.');
```

```

INSERT pc.Precursors(SupplierID, ReagentID,
ProductionDate, BestBeforeDate, PrecursorAmount, MeasurementUnits,
UsageStatus)
VALUES (3, 3, '2023-10-11', '2024-04-11', 5, 'шт', '');
INSERT pc.Precursors(SupplierID, ReagentID,
ProductionDate, BestBeforeDate, PrecursorAmount, MeasurementUnits,
UsageStatus)
VALUES (4, 4, '2023-10-10', '2028-10-10', 1000, 'мл', '');
INSERT pc.Precursors(SupplierID, ReagentID,
ProductionDate, BestBeforeDate, PrecursorAmount, MeasurementUnits,
UsageStatus)
VALUES (4, 4, '2023-11-22', '2023-11-22', 5000, 'мл', '');
INSERT pc.Precursors(SupplierID, ReagentID,
ProductionDate, BestBeforeDate, PrecursorAmount, MeasurementUnits,
UsageStatus)
VALUES (5, 5, '2023-08-17', '2030-08-17', 5000, 'мл', '');
INSERT pc.Precursors(SupplierID, ReagentID,
ProductionDate, BestBeforeDate, PrecursorAmount, MeasurementUnits,
UsageStatus)
VALUES (1, 6, '2022-12-12', '2025-12-12', 1000, 'г', '');
INSERT pc.Precursors(SupplierID, ReagentID,
ProductionDate, BestBeforeDate, PrecursorAmount, MeasurementUnits,
UsageStatus)
VALUES (2, 7, '2023-05-01', '2026-05-01', 1000, 'мл', '');
INSERT pc.Precursors(SupplierID, ReagentID,
ProductionDate, BestBeforeDate, PrecursorAmount, MeasurementUnits,
UsageStatus)
VALUES (3, 7, '2023-09-20', '2026-09-20', 1000, 'мл', '');
INSERT pc.Precursors(SupplierID, ReagentID,
ProductionDate, BestBeforeDate, PrecursorAmount, MeasurementUnits,
UsageStatus)
VALUES (4, 8, '2023-02-04', '2028-02-04', 1000, 'мл', '');
INSERT pc.Precursors(SupplierID, ReagentID,

```

```

ProductionDate, BestBeforeDate, PrecursorAmount, MeasurementUnits,
UsageStatus)
VALUES (4, 8, '2023-06-02', '2028-06-02', 10000, 'мл', '');
INSERT pc.Precursors(SupplierID, ReagentID,
ProductionDate, BestBeforeDate, PrecursorAmount, MeasurementUnits,
UsageStatus)
VALUES (5, 9, '2022-12-01', '2027-12-01', 1000, 'мл', '');
INSERT pc.Precursors(SupplierID, ReagentID,
ProductionDate, BestBeforeDate, PrecursorAmount, MeasurementUnits,
UsageStatus)
VALUES (1, 10, '2023-03-25', '2024-03-25', 1000, 'мл', 'израсх.');
```

```

INSERT pc.Precursors(SupplierID, ReagentID,
ProductionDate, BestBeforeDate, PrecursorAmount, MeasurementUnits,
UsageStatus)
VALUES (2, 11, '2023-11-29', '2024-11-29', 500, 'г', '');

SELECT PrecursorID, SupplierID, ReagentID,
ProductionDate, BestBeforeDate, PrecursorAmount, MeasurementUnits,
UsageStatus
FROM pc.Precursors;
```

	PrecursorID	SupplierID	ReagentID	ProductionDate	BestBeforeDate	PrecursorAmount	UsageStatus
1	1	1	1	2023-06-29	2025-06-29	1000	
2	2	1	1	2023-06-29	2025-06-29	1000	
3	3	1	1	2023-06-29	2025-06-29	1000	
4	4	2	2	2022-11-06	2023-11-06	10	
5	5	3	3	2023-10-11	2024-04-11	5	израсх.
6	6	3	3	2023-10-11	2024-04-11	5	
7	7	4	4	2023-10-10	2028-10-10	1000	
8	8	4	4	2023-11-22	2023-11-22	5000	
9	9	5	5	2023-08-17	2030-08-17	5000	
10	10	1	6	2022-12-12	2025-12-12	1000	
11	11	2	7	2023-05-01	2026-05-01	1000	
12	12	3	7	2023-09-20	2026-09-20	1000	
13	13	4	8	2023-02-04	2028-02-04	1000	
14	14	4	8	2023-06-02	2028-06-02	10000	
15	15	5	9	2022-12-01	2027-12-01	1000	
16	16	1	10	2023-03-25	2024-03-25	1000	израсх.
17	17	2	11	2023-11-29	2024-11-29	500	

Рисунок 13. Результат заполнения таблицы Precursors.

```

INSERT pc.PrecursorConsumptions(PrecursorID, AnalysisID, EmployeeID,
UsageAmount, UsageDate)
VALUES (14, 1, 1, 60, '2023-06-29');
INSERT pc.PrecursorConsumptions(PrecursorID, AnalysisID, EmployeeID,
UsageAmount, UsageDate)
VALUES (14, 2, 2, 250, '2023-06-30');
INSERT pc.PrecursorConsumptions(PrecursorID, AnalysisID, EmployeeID,
UsageAmount, UsageDate)
VALUES (14, 1, 3, 150, '2023-10-28');
INSERT pc.PrecursorConsumptions(PrecursorID, AnalysisID, EmployeeID,
UsageAmount, UsageDate)
VALUES (15, 3, 3, 25, '2023-12-16');
INSERT pc.PrecursorConsumptions(PrecursorID, AnalysisID, EmployeeID,
UsageAmount, UsageDate)
VALUES (15, 3, 4, 175, '2023-12-16');
INSERT pc.PrecursorConsumptions(PrecursorID, AnalysisID, EmployeeID,
UsageAmount, UsageDate)
VALUES (15, 3, 5, 325, '2023-12-17');
INSERT pc.PrecursorConsumptions(PrecursorID, AnalysisID, EmployeeID,
UsageAmount, UsageDate)
VALUES (15, 3, 5, 472, '2023-12-17');
INSERT pc.PrecursorConsumptions(PrecursorID, AnalysisID, EmployeeID,
UsageAmount, UsageDate)
VALUES (7, 9, 6, 500, '2023-11-29');
INSERT pc.PrecursorConsumptions(PrecursorID, AnalysisID, EmployeeID,
UsageAmount, UsageDate)
VALUES (7, 9, 6, 250, '2023-12-01');
INSERT pc.PrecursorConsumptions(PrecursorID, AnalysisID, EmployeeID,
UsageAmount, UsageDate)
VALUES (11, 4, 8, 75, '2023-10-29');
INSERT pc.PrecursorConsumptions(PrecursorID, AnalysisID, EmployeeID,
UsageAmount, UsageDate)
VALUES (11, 4, 8, 450, '2023-11-29');
INSERT pc.PrecursorConsumptions(PrecursorID, AnalysisID, EmployeeID,

```

```

UsageAmount, UsageDate)
VALUES (4, 7, 7, 3, '2023-11-29');
INSERT pc.PrecursorConsumptions(PrecursorID, AnalysisID, EmployeeID,
UsageAmount, UsageDate)
VALUES (4, 7, 7, 2, '2023-12-03');

SELECT ConsumptionID, PrecursorID, AnalysisID, EmployeeID,
UsageAmount, UsageDate
FROM pc.PrecursorConsumptions;

```

	ConsumptionID	PrecursorID	AnalysisID	EmployeeID	UsageAmount	UsageDate
1	1	14	1	1	60	2023-06-29
2	2	14	2	2	250	2023-06-30
3	3	14	1	3	150	2023-10-28
4	4	15	3	3	25	2023-12-16
5	5	15	3	4	175	2023-12-16
6	6	15	3	5	325	2023-12-17
7	7	15	3	5	472	2023-12-17
8	8	7	9	6	500	2023-11-29
9	9	7	9	6	250	2023-12-01
10	10	11	4	8	75	2023-10-29
11	11	11	4	8	450	2023-11-29
12	12	5	7	7	3	2023-11-29
13	13	5	7	7	2	2023-12-03
14	14	16	7	7	900	2023-12-03

Рисунок 14. Результат заполнения таблицы PrecursorConsumptions.

## 7. Разработка объектов промежуточного слоя

### Хранимые процедуры

1. Изменение статуса использованного прекурсора. Процедура позволяет изменять статус прекурсора с “израсх.” на “соотв.”, если количество использованного при анализах реактива соответствует его количеству в упаковке, и на “не соотв.”, если иначе. Статус изменяется для всех элементов, у которых в таблице прекурсоров пользователем была установлена отметка о том, что упаковка была закончена.

```

CREATE PROCEDURE pc.UsageStatusUpdate
AS
BEGIN
    DECLARE @PrecursorID INT;

```



```

DECLARE @MeasurementUnits NVARCHAR(10);
DECLARE @PrecursorAmount INT;
DECLARE @Amount INT;
DECLARE @Diff INT;
WHILE EXISTS (SELECT 1 FROM pc.Precursors
WHERE UsageStatus = 'израсх.')
BEGIN
    SELECT TOP 1
        @PrecursorID = PrecursorID,
        @MeasurementUnits = MeasurementUnits,
        @PrecursorAmount = PrecursorAmount
    FROM
        pc.Precursors
    WHERE
        UsageStatus = 'израсх.';
    SELECT @Amount = SUM(UsageAmount)
    FROM pc.PrecursorConsumptions
    WHERE PrecursorID = @PrecursorID;
    SET @Diff = @PrecursorAmount - @Amount;
    IF (@MeasurementUnits IN ('г', 'мл') AND
@Diff <= @PrecursorAmount * 0.01)
        OR (@MeasurementUnits NOT IN ('г', 'мл') AND @Diff = 0)
    BEGIN
        UPDATE pc.Precursors
        SET UsageStatus = 'коотв.'
        WHERE PrecursorID = @PrecursorID;
    END
    ELSE
    BEGIN
        UPDATE pc.Precursors
        SET UsageStatus = 'не коотв.'
        WHERE PrecursorID = @PrecursorID;
    END
END
END

```

END;

Проверим корректность обновления данных в таблице прекурсоров:

```
EXEC pc.UsageStatusUpdate;
```

```
SELECT PrecursorID, SupplierID, ReagentID,  
ProductionDate, BestBeforeDate, PrecursorAmount, MeasurementUnits,  
UsageStatus  
FROM pc.Precursors;
```

	PrecursorID	SupplierID	ReagentID	ProductionDate	BestBeforeDate	PrecursorAmount	MeasurementUnits	UsageStatus
1	1	1	1	2023-06-29	2025-06-29	1000	мл	
2	2	1	1	2023-06-29	2025-06-29	1000	мл	
3	3	1	1	2023-06-29	2025-06-29	1000	мл	
4	4	2	2	2022-11-06	2023-11-06	10	шт	
5	5	3	3	2023-10-11	2024-04-11	5	шт	соотв.
6	6	3	3	2023-10-11	2024-04-11	5	шт	
7	7	4	4	2023-10-10	2028-10-10	1000	мл	
8	8	4	4	2023-11-22	2023-11-22	5000	мл	
9	9	5	5	2023-08-17	2030-08-17	5000	мл	
10	10	1	6	2022-12-12	2025-12-12	1000	г	
11	11	2	7	2023-05-01	2026-05-01	1000	мл	
12	12	3	7	2023-09-20	2026-09-20	1000	мл	
13	13	4	8	2023-02-04	2028-02-04	1000	мл	
14	14	4	8	2023-06-02	2028-06-02	10000	мл	
15	15	5	9	2022-12-01	2027-12-01	1000	мл	
16	16	1	10	2023-03-25	2024-03-25	1000	мл	не соотв.
17	17	2	11	2023-11-29	2024-11-29	500	г	

Рисунок 15. Результат вызова хранимой процедуры UsageStatusUpdate.

Видно, что строки 5 и 16 изменились по сравнению с той же таблицей на рис. 13.

2. Добавление нового прекурсора в таблицу Precursors.

```
CREATE PROCEDURE pc.AddPrecursor  
    @SupplierID INT,  
    @ReagentID INT,  
    @ProductionDate DATE,  
    @BestBeforeDate DATE,  
    @PrecursorAmount INT,  
    @MeasurementUnits NVARCHAR(2),  
    @UsageStatus NVARCHAR(10) = NULL  
AS  
BEGIN  
    IF @UsageStatus IS NOT NULL AND @UsageStatus != ''  
    BEGIN
```

```

        THROW 50000, 'Invalid value for UsageStatus. Allowed value is
an empty string.', 1;
    RETURN;
END
IF @PrecursorAmount < 0
BEGIN
    THROW 50001, 'PrecursorAmount cannot be negative.', 1;
    RETURN;
END
INSERT INTO pc.Precursors (SupplierID, ReagentID, ProductionDate,
BestBeforeDate, PrecursorAmount, MeasurementUnits, UsageStatus)
VALUES (@SupplierID, @ReagentID, @ProductionDate, @BestBeforeDate,
@PrecursorAmount, @MeasurementUnits, @UsageStatus);
END;

```

Проверим, что при соблюдении всех условий для ввода данных, строка будет добавлена в таблицу:

```

EXEC pc.AddPrecursor
    @SupplierID = 5,
    @ReagentID = 6,
    @ProductionDate = '2023-01-08',
    @BestBeforeDate = '2025-01-08',
    @PrecursorAmount = 2000,
    @MeasurementUnits = 'г',
    @UsageStatus = '';

SELECT PrecursorID, SupplierID, ReagentID,
ProductionDate, BestBeforeDate, PrecursorAmount, MeasurementUnits,
UsageStatus
FROM pc.Precursors;

```

	PrecursorID	SupplierID	ReagentID	ProductionDate	BestBeforeDate	PrecursorAmount	MeasurementUnits	UsageStatus
1	1	1	1	2023-06-29	2025-06-29	1000	мл	
2	2	1	1	2023-06-29	2025-06-29	1000	мл	
3	3	1	1	2023-06-29	2025-06-29	1000	мл	
4	4	2	2	2022-11-06	2023-11-06	10	шт	
5	5	3	3	2023-10-11	2024-04-11	5	шт	соотв.
6	6	3	3	2023-10-11	2024-04-11	5	шт	
7	7	4	4	2023-10-10	2028-10-10	1000	мл	
8	8	4	4	2023-11-22	2023-11-22	5000	мл	
9	9	5	5	2023-08-17	2030-08-17	5000	мл	
10	10	1	6	2022-12-12	2025-12-12	1000	г	
11	11	2	7	2023-05-01	2026-05-01	1000	мл	
12	12	3	7	2023-09-20	2026-09-20	1000	мл	
13	13	4	8	2023-02-04	2028-02-04	1000	мл	
14	14	4	8	2023-06-02	2028-06-02	10000	мл	
15	15	5	9	2022-12-01	2027-12-01	1000	мл	
16	16	1	10	2023-03-25	2024-03-25	1000	мл	не соотв.
17	17	2	11	2023-11-29	2024-11-29	500	г	
18	18	5	6	2023-01-08	2025-01-08	2000	г	

Рисунок 16. Результат вызова хранимой процедуры AddPrecursor.

Заметим, что ввод информации о новом прекурсоре с непустой строкой в поле UsageStatus или с отрицательным числом в поле PrecursorAmount вызовет ошибку с соответствующим сообщением.

**EXEC** pc.AddPrecursor

```
@SupplierID = 5,
@ReagentID = 6,
@ProductionDate = '2023-01-08',
@BestBeforeDate = '2025-01-08',
@PrecursorAmount = 2000,
@MeasurementUnits = 'г',
@UsageStatus = 'соотв.';
```

Msg 50000, Level 16, State 1, Procedure pc.AddPrecursor, Line 13 [Batch Start Line 0]  
Invalid value for UsageStatus. Allowed value is an empty string.

Рисунок 17. Результат вызова хранимой процедуры AddPrecursor. Вывод сообщения об ошибке 50000.

**EXEC** pc.AddPrecursor

```
@SupplierID = 5,
@ReagentID = 6,
@ProductionDate = '2023-01-08',
@BestBeforeDate = '2025-01-08',
@PrecursorAmount = -1,
@MeasurementUnits = 'г',
@UsageStatus = '';
```

Msg 50001, Level 16, State 1, Procedure pc.AddPrecursor, Line 19 [Batch Start Line 0]  
PrecursorAmount cannot be negative.

Рисунок 18. Результат вызова хранимой процедуры AddPrecursor. Вывод сообщения об ошибке 50001.

2. Удаление прекурсора из таблицы Precursors и соответствующих строк с его расходами в таблице PrecursorConsumptions.

```
CREATE PROCEDURE pc.DeletePrecursor
    @PrecursorID INT
AS
BEGIN
    DELETE FROM pc.PrecursorConsumptions
    WHERE PrecursorID = @PrecursorID;
    DELETE FROM pc.Precursors
    WHERE PrecursorID = @PrecursorID;
END;
```

Проверим корректность удаления:

```
EXEC pc.DeletePrecursor @PrecursorID = 11;
```

	PrecursorID	SupplierID	ReagentID	ProductionDate	BestBeforeDate	PrecursorAmount	MeasurementUnits	UsageStatus
1	1	1	1	2023-06-29	2025-06-29	1000	мл	
2	2	1	1	2023-06-29	2025-06-29	1000	мл	
3	3	1	1	2023-06-29	2025-06-29	1000	мл	
4	4	2	2	2022-11-06	2023-11-06	10	шт	
5	5	3	3	2023-10-11	2024-04-11	5	шт	соотв.
6	6	3	3	2023-10-11	2024-04-11	5	шт	
7	7	4	4	2023-10-10	2028-10-10	1000	мл	
8	8	4	4	2023-11-22	2023-11-22	5000	мл	
9	9	5	5	2023-08-17	2030-08-17	5000	мл	
10	10	1	6	2022-12-12	2025-12-12	1000	г	
11	12	3	7	2023-09-20	2026-09-20	1000	мл	
12	13	4	8	2023-02-04	2028-02-04	1000	мл	

Рисунок 19. Результат вызова хранимой процедуры DeletePrecursor. Удалена строка с PrecursorID = 11 из таблицы Precursors.

	ConsumptionID	PrecursorID	AnalysisID	EmployeeID	UsageAmount	UsageDate
1	1	14	1	1	60	2023-06-29
2	2	14	2	2	250	2023-06-30
3	3	14	1	3	150	2023-10-28
4	4	15	3	3	25	2023-12-16
5	5	15	3	4	175	2023-12-16
6	6	15	3	5	325	2023-12-17
7	7	15	3	5	472	2023-12-17
8	8	7	9	6	500	2023-11-29
9	9	7	9	6	250	2023-12-01
10	12	5	7	7	3	2023-11-29
11	13	5	7	7	2	2023-12-03
12	14	16	7	7	900	2023-12-03

Рисунок 20. Результат вызова хранимой процедуры DeletePrecursor. Удалена строка PrecursorID = 11 из таблицы PrecursorConsumptions.

3. Установка статуса использования прекурсора “израсх.” для сообщения, что его упаковка закончилась. Применима только к тем строкам таблицы Precursors, у которых в поле UsageStatus пустая строка.

```
CREATE PROCEDURE pc.SetUsageStatus
    @PrecursorID INT
AS
BEGIN
    DECLARE @CurrentStatus NVARCHAR(10);
    SELECT @CurrentStatus = UsageStatus
    FROM pc.Precursors
    WHERE PrecursorID = @PrecursorID;
    IF @CurrentStatus = ''
    BEGIN
        UPDATE pc.Precursors
        SET UsageStatus = 'израсх.'
        WHERE PrecursorID = @PrecursorID;
    END
    ELSE
    BEGIN
        THROW 50000,
        'Invalid operation. UsageStatus is already set.', 1;
    END;
END;
```

Проверим корректность установки статуса для элемента 17 таблицы Precursors, у которого на рис. 13 статус отсутствует:

```
EXEC pc.SetUsageStatus @PrecursorID = 17;
```

16	16	1	10	2023-03-25	2024-03-25	1000	мл	не соотв.
17	17	2	11	2023-11-29	2024-11-29	500	г	израсх.
18	19	5	6	2023-01-08	2025-01-08	2000	г	

Рисунок 21. Результат вызова хранимой процедуры SetUsageStatus.

Попробуем выполнить данную процедуру для строки 16 и получим сообщение об ошибке:

```
EXEC pc.SetUsageStatus @PrecursorID = 16;
```

Msg 50000, Level 16, State 1, Procedure pc.UpdateUsageStatus, Line 19 [Batch Start Line 0]  
Invalid operation. UsageStatus is already set.

Рисунок 22. Результат вызова хранимой процедуры SetUsageStatus. Вывод сообщения об ошибке 50000.

## UDF

1. Вычисление количества прекурсора, оставшегося во всех таррах таблицы Precursors, по его идентификатору. Для вычисления точного остатка реактива, необходимо, чтобы его статус использования был подтвержден значением “соотв.” в поле UsageStatus таблицы Precursors или не имел значения вовсе будучи еще используемым.

```
CREATE FUNCTION pc.CalculateReagentResidue(@ReagentIDParam INT)
RETURNS INT
AS
BEGIN
    DECLARE @Amount1 INT;
    DECLARE @Amount2 INT;
    SELECT @Amount1 = SUM(PrecursorAmount)
    FROM pc.Precursors
    WHERE ReagentID = @ReagentIDParam
        AND NOT EXISTS (
            SELECT 1
            FROM pc.Precursors AS P2
            WHERE P2.ReagentID = @ReagentIDParam
                AND (P2.UsageStatus <> '' AND P2.UsageStatus <> 'соотв.')
        );
    IF @Amount1 IS NULL
    BEGIN
        RETURN NULL;
    END
    SELECT @Amount2 = SUM(UsageAmount)
    FROM pc.PrecursorConsumptions
    WHERE PrecursorID IN (SELECT PrecursorID FROM pc.Precursors
    WHERE ReagentID = @ReagentIDParam);
    DECLARE @Result INT;
```

```

SET @Result = @Amount1 - @Amount2;
RETURN @Result;
END;

```

Проверим вычисление остатка прекурсора 3:

```
SELECT pc.CalculateReagentResidue(3)
```

	(No column name)
1	NULL

Рисунок 23. Результат вычисления остатка прекурсора 3.

Видно, что для данного реактива не было найдено подходящих для расчета элементов прекурсоров, что подтверждается на рис. 13 наличием у одной из тар реактива статуса “израсх.”.

Проверим вычисление остатка прекурсора 8:

```
SELECT pc.CalculateReagentResidue(8)
```

	(No column name)
1	10540

Рисунок 24. Результат вычисления остатка прекурсора 8.

Исходя из рисунков 13 и 14 можно вычислить разность начального количества реактива прекурсора с использованным:  $11000 - 60 - 250 - 150 = 10540$ .

2. Вывод списка всех заканчивающихся прекурсоров и их остатков для их дозаказа.

```

CREATE FUNCTION pc.PrecursorsForOrder()
RETURNS TABLE
AS
RETURN
(

```

```

    SELECT DISTINCT
        r.ReagentID,
        r.SubstanceName,
        r.SubstanceContent,
        r.ReagentPurity,

```



```

        Residue.ReagentResidue AS ReagentResidue,
        p.MeasurementUnits
FROM pc.Reagents r
OUTER APPLY (
    SELECT
        COALESCE(SUM(p1.PrecursorAmount), 0) AS Amount1
    FROM pc.Precursors p1
    WHERE p1.ReagentID = r.ReagentID
) AS Amount
OUTER APPLY (
    SELECT
        COALESCE(SUM(p2.UsageAmount), 0) AS Amount2
    FROM pc.PrecursorConsumptions p2
    WHERE p2.PrecursorID IN (SELECT p3.PrecursorID
FROM pc.Precursors p3 WHERE p3.ReagentID = r.ReagentID)
) AS UsageAmount
OUTER APPLY (
    SELECT
        Amount.Amount1 - UsageAmount.Amount2 AS ReagentResidue
) AS Residue
LEFT JOIN pc.Precursors p ON p.ReagentID = r.ReagentID
WHERE Residue.ReagentResidue IS NOT NULL
    AND Residue.ReagentResidue <= Amount.Amount1 * 0.5
);

```

Проверим работу данной функции:

```

SELECT ReagentID, SubstanceName, SubstanceContent, ReagentPurity,
ReagentResidue, MeasurementUnits
FROM pc.PrecursorsForOrder();

```

	ReagentID	SubstanceName	SubstanceContent	ReagentPurity	ReagentResidue	MeasurementUnits
1	3	Соляная кислота	0,1Н		5	шт
2	9	Пиперидин	99%	хч	3	мл
3	10	Кислота серная	99%	осч	100	мл

Рисунок 25. Результат определения заканчивающихся прекурсоров и их остатков.

3. Вывод списка прекурсоров, их предположительные остатки в упаковках и оставшийся на момент выполнения функции срок годности с датой его истечения.

```
CREATE FUNCTION pc.PrecursorsStoragePeriodLeft()
RETURNS TABLE
AS
RETURN
(
    SELECT DISTINCT
        r.ReagentID,
        r.SubstanceName,
        r.SubstanceContent,
        r.ReagentPurity,
        p.PrecursorID,
        (p.PrecursorAmount - COALESCE(UsageAmount.Amount1, 0))
    AS ReagentResidue,
        p.MeasurementUnits,
        p.BestBeforeDate,
        DATEDIFF(YEAR, SYSDATETIME(), p.BestBeforeDate)
    AS YearsDifference,
        DATEDIFF(MONTH, SYSDATETIME(), p.BestBeforeDate)
    AS MonthsDifference,
        DATEDIFF(DAY, SYSDATETIME(), p.BestBeforeDate)
    AS DaysDifference
    FROM pc.Reagents r
    OUTER APPLY (
        SELECT
            pc.Precursors.PrecursorID,
            pc.Precursors.PrecursorAmount,
            MAX(pc.Precursors.MeasurementUnits) AS MeasurementUnits,
            MAX(pc.Precursors.BestBeforeDate) AS BestBeforeDate
        FROM pc.Precursors
        WHERE pc.Precursors.ReagentID = r.ReagentID
```

```

GROUP BY pc.Precursors.PrecursorID,
pc.Precursors.PrecursorAmount
) AS p
OUTER APPLY (
SELECT
COALESCE(SUM(pc1.UsageAmount), 0) AS Amount1
FROM pc.PrecursorConsumptions pc1
WHERE pc1.PrecursorID = p.PrecursorID
) AS UsageAmount
);

```

Проверим работу данной функции:

```

SELECT ReagentID, SubstanceName, SubstanceContent, ReagentPurity,
PrecursorID, ReagentResidue, MeasurementUnits, BestBeforeDate,
YearsDifference, MonthsDifference, DaysDifference
FROM pc.PrecursorsStoragePeriodLeft();

```

	ReagentID	SubstanceName	SubstanceContent	ReagentPurity	PrecursorID	ReagentResidue	MeasurementUnits	BestBeforeDate	YearsDifference	MonthsDifference	DaysDifference
1	1	Серная кислота	94%	хч	1	1000	мл	2025-06-29	2	18	555
2	1	Серная кислота	94%	хч	2	1000	мл	2025-06-29	2	18	555
3	1	Серная кислота	94%	хч	3	1000	мл	2025-06-29	2	18	555
4	2	Серная кислота	0,1Н		4	10	шт	2023-11-06	0	-1	-46
5	3	Соляная кислота	0,1Н		5	0	шт	2024-04-11	1	4	111
6	3	Соляная кислота	0,1Н		6	5	шт	2024-04-11	1	4	111
7	4	Ацетон	99,75%	осч	7	250	мл	2028-10-10	5	58	1754
8	4	Ацетон	99,75%	осч	8	5000	мл	2023-11-22	0	-1	-30
9	5	Толуол	99%	тех	9	5000	мл	2030-08-17	7	80	2430
10	6	1,4-Бутандиол	99%	ч	10	1000	г	2025-12-12	2	24	721
11	6	1,4-Бутандиол	99%	ч	19	2000	г	2025-01-08	2	13	383
12	7	Диэтиловый эфир	99,5%	чда	11	475	мл	2026-05-01	3	29	861
13	7	Диэтиловый эфир	99,5%	чда	12	1000	мл	2026-09-20	3	33	1003
14	8	Соляная кислота	35%	хч	13	1000	мл	2028-02-04	5	50	1505
15	8	Соляная кислота	35%	хч	14	9540	мл	2028-06-02	5	54	1624
16	9	Пиперидин	99%	хч	15	3	мл	2027-12-01	4	48	1440
17	10	Кислота серная	99%	осч	16	100	мл	2024-03-25	1	3	94
18	11	Калий марганцовокислый	99,5%	чда	17	500	г	2024-11-29	1	11	343

Рисунок 26. Результат вывода списка прекурсоров, их остатков и оставшегося срока годности на момент выполнения функции.

## Представления

1. Вывод для каждого прекурсора перечня компаний-поставщиков и контактной информации для их заказа.

```

CREATE VIEW pc.ReagentSuppliersForOrder AS

```

SELECT

```

r.ReagentID,
r.SubstanceName,
r.SubstanceContent,
r.ReagentPurity,
p.SupplierID,
ps.CompanyName,
ps.TelephoneNumber,
ps.Email

```

FROM pc.Reagents r

JOIN pc.Precursors p ON r.ReagentID = p.ReagentID

JOIN pc.ReagentSuppliers ps ON p.SupplierID = ps.SupplierID;

Проверим работу данного представления:

```

SELECT ReagentID, SubstanceName, SubstanceContent, ReagentPurity,
SupplierID, CompanyName, TelephoneNumber, Email
FROM pc.ReagentSuppliersForOrder;

```

	ReagentID	SubstanceName	SubstanceContent	ReagentPurity	SupplierID	CompanyName	TelephoneNumber	Email
1	1	Серная кислота	94%	хч	1	НеваРеактив	88123254065	office@nevareaktiv.ru
2	1	Серная кислота	94%	хч	1	НеваРеактив	88123254065	office@nevareaktiv.ru
3	1	Серная кислота	94%	хч	1	НеваРеактив	88123254065	office@nevareaktiv.ru
4	2	Серная кислота	0,1Н		2	ЛенРеактив	88122239637	info@lenreactiv.ru
5	3	Соляная кислота	0,1Н		3	Формула	88129493420	Store@formula-re.ru
6	3	Соляная кислота	0,1Н		3	Формула	88129493420	Store@formula-re.ru
7	4	Ацетон	99,75%	осч	4	Ареолаб	89217780442	areolab@yandex.ru
8	4	Ацетон	99,75%	осч	4	Ареолаб	89217780442	areolab@yandex.ru
9	5	Толуол	99%	тех	5	БалтХим	88124541466	mail@baltchim.ru
10	6	1,4-Бутандиол	99%	ч	1	НеваРеактив	88123254065	office@nevareaktiv.ru
11	7	Диэтиловый эфир	99,5%	чда	2	ЛенРеактив	88122239637	info@lenreactiv.ru
12	7	Диэтиловый эфир	99,5%	чда	3	Формула	88129493420	Store@formula-re.ru
13	8	Соляная кислота	35%	хч	4	Ареолаб	89217780442	areolab@yandex.ru
14	8	Соляная кислота	35%	хч	4	Ареолаб	89217780442	areolab@yandex.ru
15	9	Пиперидин	99%	хч	5	БалтХим	88124541466	mail@baltchim.ru
16	10	Кислота серная	99%	осч	1	НеваРеактив	88123254065	office@nevareaktiv.ru
17	11	Калий марганцовокислый	99,5%	чда	2	ЛенРеактив	88122239637	info@lenreactiv.ru

Рисунок 27. Результат работы представления ReagentSuppliersForOrder.

2. Вывод таблицы с информацией обо всех расходах прекурсоров для обзора истории использования реактивов: даты, исполнителя, анализа, типа прекурсора и его израсходованного количества.

```

CREATE VIEW pc.DatePrecursorConsumptions
AS

```

SELECT

```
pc.UsageDate,
c.EmployeeID,
c.Surname,
c.FirstName,
c.SecondName,
a.AnalysisName,
r.SubstanceName,
pc.UsageAmount,
p.MeasurementUnits
```

FROM

```
pc.PrecursorConsumptions pc
```

JOIN

```
pc.Chemists AS c ON pc.EmployeeID = c.EmployeeID
```

JOIN

```
pc.Precursors AS p ON pc.PrecursorID = p.PrecursorID
```

JOIN

```
pc.Analyses AS a ON a.AnalysisID = pc.AnalysisID
```

JOIN

```
pc.Reagents AS r ON p.ReagentID = r.ReagentID;
```

Проверим работу данного представления:

```
SELECT UsageDate, EmployeeID, Surname, FirstName, SecondName,
AnalysisName, SubstanceName, UsageAmount, MeasurementUnits
FROM pc.DatePrecursorConsumptions;
```

	UsageDate	EmployeeID	Surname	FirstName	SecondName	AnalysisName	SubstanceName	UsageAmount	MeasurementUnits
1	2023-06-29	1	Кузнецова	Ирина	Олеговна	Нитраты	Соляная кислота	60	мл
2	2023-06-30	2	Садриева	Вера		Сульфаты	Соляная кислота	250	мл
3	2023-10-28	3	Жукова	Марьяна		Нитраты	Соляная кислота	150	мл
4	2023-12-16	3	Жукова	Марьяна		Перманганатная окисляемость	Пиперидин	25	мл
5	2023-12-16	4	Белозеров	Филипп	Аркадьевич	Перманганатная окисляемость	Пиперидин	175	мл
6	2023-12-17	5	Кузьмин	Александр	Валерьевич	Перманганатная окисляемость	Пиперидин	325	мл
7	2023-12-17	5	Кузьмин	Александр	Валерьевич	Перманганатная окисляемость	Пиперидин	472	мл
8	2023-11-29	6	Дакуо	Кристиан		Бензол	Ацетон	500	мл
9	2023-12-01	6	Дакуо	Кристиан		Бензол	Ацетон	250	мл
10	2023-10-29	8	Петрова	Ксения	Александровна	Ацетон и метанол	Дизтиловый эфир	75	мл
11	2023-11-29	8	Петрова	Ксения	Александровна	Ацетон и метанол	Дизтиловый эфир	450	мл
12	2023-11-29	7	Рахматова	Барно		Мышьяк	Соляная кислота	3	шт
13	2023-12-03	7	Рахматова	Барно		Мышьяк	Соляная кислота	2	шт
14	2023-12-03	7	Рахматова	Барно		Мышьяк	Кислота серная	900	мл

Рисунок 28. Результат работы представления DatePrecursorConsumptions.

3. Вывод таблицы с информацией обо всех прекурсорах, чьи расходы не совпали с их изначальным количеством в упаковке, с вышедшим остатком ReagentResidue — разницей здесь в закупленном и использованном количествах.

```
CREATE VIEW pc.WrongUsedReagents1
AS
SELECT
    r.ReagentID,
    r.SubstanceName,
    r.SubstanceContent,
    r.ReagentPurity,
    p.PrecursorID,
    p.PrecursorAmount,
    Residue.ReagentResidue,
    p.MeasurementUnits,
    p.UsageStatus
FROM
    pc.Precursors p
JOIN
    pc.Reagents r ON p.ReagentID = r.ReagentID
LEFT JOIN
    (
        SELECT
            pc1.PrecursorID,
            pc1.PrecursorAmount - COALESCE(SUM(pc2.UsageAmount), 0)
        AS ReagentResidue
        FROM
            pc.Precursors pc1
        LEFT JOIN
            pc.PrecursorConsumptions pc2
        ON pc1.PrecursorID = pc2.PrecursorID
        WHERE
            pc1.UsageStatus = 'COOTB.'
        GROUP BY
            pc1.PrecursorID, pc1.PrecursorAmount
    ) AS Residue ON p.PrecursorID = Residue.PrecursorID
WHERE
    p.UsageStatus = 'COOTB.';
```

Проверим работу данного представления:

```
SELECT ReagentID, SubstanceName, ReagentPurity, SubstanceContent,
ReagentResidue, MeasurementUnits, UsageStatus
FROM pc.WrongUsedReagents;
```

	ReagentID	SubstanceName	ReagentPurity	SubstanceContent	PrecursorID	PrecursorAmount	ReagentResidue	MeasurementUnits	UsageStatus
1	10	Кислота серная	осч	99%	16	1000	100	мл	не соотв.

Рисунок 29. Результат работы представления WrongUsedReagents.

Рисунок 15 подтверждает, что список сомнительно израсходованных прекурсоров, показанный на рисунке 29, был представлен верно.

## 8. Разработка стратегии резервного копирования

Резервное копирование базы данных представляет собой процесс создания копии данных, хранящихся в базе, с целью обеспечения их сохранности и восстановления в случае потери, повреждения или непредвиденных сбоев. В данной работе будет сделана полная резервная копия базы данных, а также дифференциальная копия базы данных, которая учитывает изменения последнего копирования, и журнал транзакций базы.

Изменим для начала модель восстановления базы данных на «Полную».

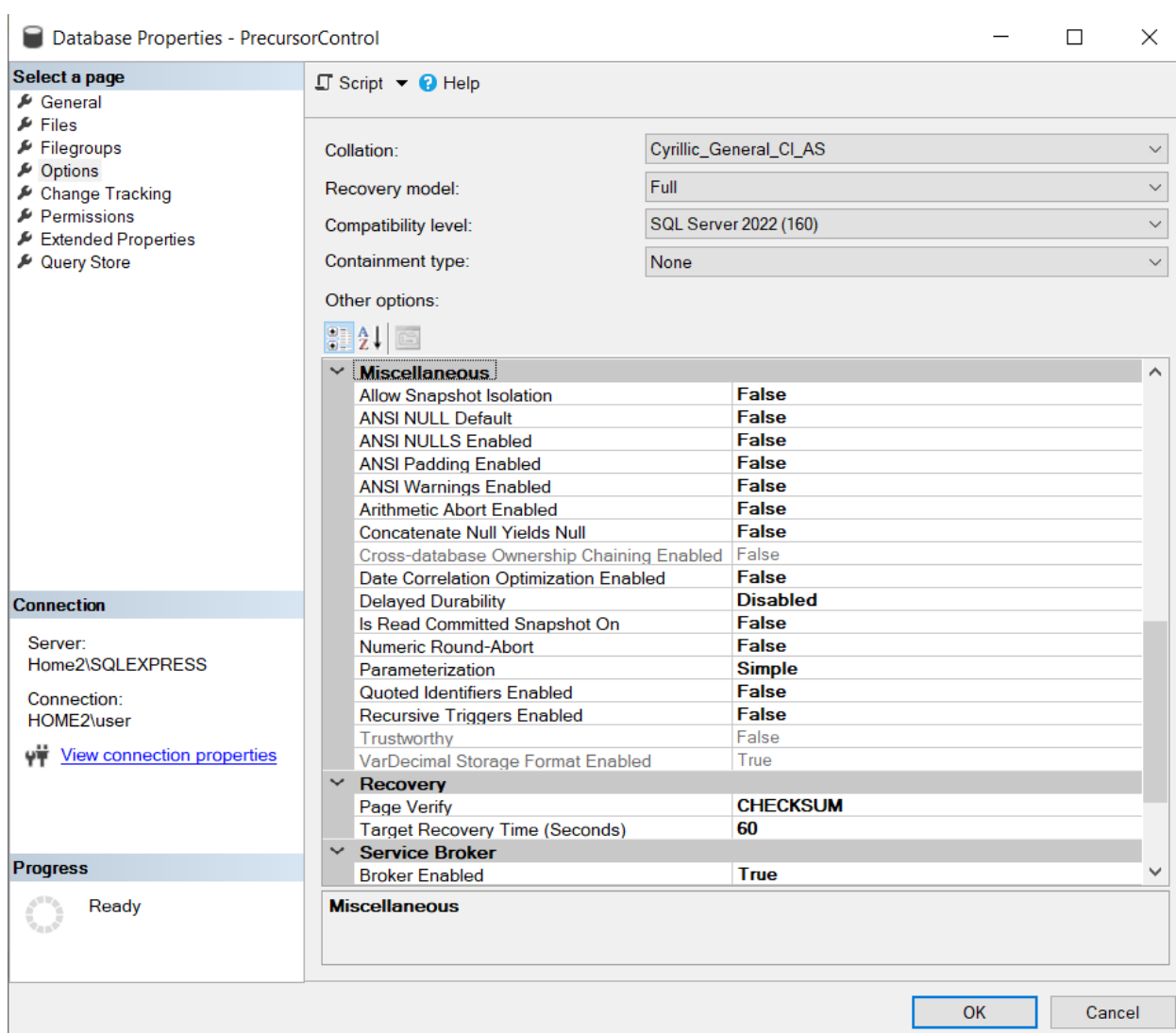


Рисунок 30. Изменение модели восстановления БД.

В случае базы данных небольшого объема может выполняться полное резервное копирование базы ежедневно в определённое время.



Создадим полную резервную копию базы данных PrecursorControl:

```
BACKUP DATABASE PrecursorControl  
TO DISK = 'C:\BD_IDZ\PrecursorControlReserve.bak'  
WITH INIT
```

Обработано 688 страниц для базы данных "PrecursorControl", файл "PrecursorControl\_dat" для файла 1.  
Обработано 2 страниц для базы данных "PrecursorControl", файл "PrecursorControl\_log" для файла 1.  
BACKUP DATABASE успешно обработал 690 страниц за 0.040 секунд (134.667 МБ/сек).

Рисунок 31. Изменение модели восстановления БД.

Для большого объема данных полное резервное копирование может выполняться еженедельно. Ежедневное полное резервное копирование может быть заменено разностным резервным копированием.

```
BACKUP DATABASE PrecursorControl  
TO DISK = 'C:\BD_IDZ\PrecursorControlReserveRAS.bak'  
WITH  
DIFFERENTIAL, INIT
```

Обработано 72 страниц для базы данных "PrecursorControl", файл "PrecursorControl\_dat" для файла 1.  
Обработано 2 страниц для базы данных "PrecursorControl", файл "PrecursorControl\_log" для файла 1.  
BACKUP DATABASE WITH DIFFERENTIAL успешно обработал 74 страниц за 0.013 секунд (44.170 МБ/сек).

Рисунок 32. Изменение модели восстановления БД.

Кроме этого, нужно выполнить резервное копирование журнала транзакций.

```
BACKUP LOG PrecursorControl  
TO DISK = 'C:\BD_IDZ\PrecursorControlReserveLOG.bak'
```

Обработано 44 страниц для базы данных "PrecursorControl", файл "PrecursorControl\_log" для файла 1.  
BACKUP LOG успешно обработал 44 страниц за 0.011 секунд (31.250 МБ/сек).

Рисунок 33. Изменение модели восстановления БД.

Чтобы восстановить резервную копию базы данных и журналов транзакций, а также, чтобы иметь возможность применить несколько инструкций RESTORE, следует указать параметр WITH NORECOVERY для всех инструкций, исключая последнюю.

```
RESTORE DATABASE PrecursorControl  
FROM DISK='C:\BD_IDZ\PrecursorControlReserve.bak'  
WITH NORECOVERY
```

```
RESTORE DATABASE PrecursorControl
FROM DISK='C:\BD_IDZ\PrecursorControlReserveRAS.bak'
WITH NORECOVERY
```

```
RESTORE LOG PrecursorControl
FROM DISK='C:\BD_IDZ\PrecursorControlReserveLOG.bak'
WITH RECOVERY
```

## 9. Заключение

При работе над данным индивидуальным домашним заданием была спроектирована и создана база данных для контроля использования веществ-прекурсоров в химической лаборатории, содержащая в себе шесть таблиц.

В процессе выполнения задания были разработаны такие объекты промежуточного слоя, как:

1. Хранимые процедуры для добавления и удаления прекурсора, а также изменения статуса использования прекурсоров.
2. Пользовательские функции для вычисления: количества прекурсора, оставшегося во всех тарах, вывода списка заканчивающихся прекурсоров для дозаказа и таблицы для контроля сроков годности их остатков.
3. Представления для вывода таблиц, содержащих: список компаний-поставщиков реактивов, информацию об истории расходов прекурсоров, а также сведения обо всех прекурсорах, чьи расходы не совпали с их изначальным количеством в упаковке.

Помимо всего перечисленного, в данной работе была предложена стратегия резервного копирования базы данных.

Таким образом, выполнение представленного индивидуального домашнего задания позволило продемонстрировать и закрепить все знания и навыки, полученные в результате практических и лекционных занятий по курсу дисциплины “Базы данных”.

## 10.Список использованных источников

1. Распределенные базы данных: Методические указания к лабораторным работам / Сост.: А. В. Горячев, Н. Е. Новакова. СПб.: Изд-во СПбГЭТУ «ЛЭТИ», 2008. 32 с
2. Горячев А. В, Новакова Н.Е. Особенности разработки и администрирования приложений баз данных: учеб. пособие. СПб.: Изд-во СПбГЭТУ «ЛЭТИ», 2016. 68 с.
3. Изучаем SQL. Генерация, выборка и обработка данных, 3-е изд./ Алан Болье; пер. с англ. И.В. Красикова. — Киев. : “Диалектика”, 2021. — 402 с.: ил. — Парал. тит . англ.
4. Осипов Д. Л. Технологии проектирования баз данных. – М.: ДМК Пресс, 2019. – 498 с.: ил.
5. Курс «Базы данных» / Moodle — Open-source learning platform [Электронный ресурс] Режим доступа: <https://vec.etu.ru/moodle/course/view.php?id=14314> (дата обращения 01.12.2023).
6. Справочник по Transact-SQL / Microsoft Learn [Электронный ресурс] Режим доступа: <https://learn.microsoft.com/ru-ru/sql/t-sql/> (дата обращения 09.12.2023).