МИНОБРНАУКИ РОССИИ

САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ

ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)

Кафедра САПР

ОТЧЕТ

по индивидуальному домашнему заданию

по дисциплине «Базы данных»

Тема: «Проектирование базы данных “Почта”»

Студент гр. 1305 _____ Смирнов В.А.

Преподаватель _____ Новакова Н.Е.

Санкт-Петербург

2023

ЗАДАНИЕ

Студент: Смирнов В.А.

Группа: 1305

Тема ИДЗ: «Проектирование и базы данных “Почта”»

Задание:

Спроектировать базу данных (БД) для почтовых работников, используя метод ER-диаграмм.

В БД должны храниться сведения о газетах; о типографиях, выпускающих газеты; о почтовых отделениях, получающих газеты. Сведения о газетах должны включать в себя: название газеты, шифр издания, цену экземпляра газеты, ФИО редактора, номера типографий, где печатается эта газета. Возможно появление новых газет. Для почтового отделения: адрес отделения, название и количество экземпляров, поступающих на каждое почтовое отделение. Для типографий: адрес тип, количество газет данного наименования, печатающихся в этой типографии (в одной типографии может печататься несколько газет). Типография может быть закрыта.

В индивидуальном домашнем задании необходимо выполнить следующие пункты:

1) Краткое описание предметной области

2) Проектирование БД (структура данных)

3) Создание БД

4) Создание таблиц и ограничений целостности

5) Заполнение таблиц данными

6) Разработка объектов промежуточного слоя

7) Разработка стратегии резервного копирования

Содержание пояснительной записки:

«Содержание», «Введение», «Краткое описание предметной области», «Проектирование БД» «Создание БД», «Создание таблиц и ограничений целостности», «Заполнение таблиц данными», «Разработка объектов промежуточного слоя», «Разработка стратегии резервного копирования», «Заключение», «Список использованной литературы».

Предполагаемый объем пояснительной записки:

Не менее 20 страниц

Дата выдачи ИДЗ: 01.09.2023

Дата сдачи ИДЗ: 26.11.2023

Дата защиты ИДЗ: .12.2023

Студент _____ Смирнов В.А.

Преподаватель _____ Новакова Н.Е.

СОДЕРЖАНИЕ

Цель работы

Закрепить теоретические знания, приобретённые в ходе изучения курса «Базы данных», и получить практические навыки в проектировании и создании базы данных, в создании объектов промежуточного слоя, а именно представлений, хранимых процедур и UDF, а также научиться разрабатывать стратегию резервного копирования.

Задачи работы

В ходе выполнения работы сначала кратко обсуждается предметная область, затем база данных проектируется, строится диаграмма связей между таблицами. После этого создаётся база данных с таблицами и ограничениями целостности. Таблицы заполняются тестовыми данными.

Затем создаются объекты промежуточного слоя, а именно представления, хранимые процедуры и UDF (по 2-3 объекта), и разрабатывается стратегия резервного копирования.

1. Краткое описание предметной области

В БД должны храниться сведения о газетах; о типографиях, выпускающих газеты; о почтовых отделениях, получающих газеты.

Сведения о газетах должны включать в себя: название газеты, шифр издания, цену экземпляра газеты, ФИО редактора, номера типографий, где печатается эта газета. Возможно появление новых газет. Для почтового отделения: адрес отделения, название и количество экземпляров, поступающих на каждое почтовое отделение. Для типографий: адрес тип, количество газет данного наименования, печатающихся в этой типографии (в одной типографии может печататься несколько газет). Типография может быть закрыта.

БД «ПОЧТА» проектируем при условии, что:

– в одной типографии может печататься несколько газет;

– одна и та же газета может печататься в разных типографиях.

2. Проектирование базы данных

Проектируем БД с использованием метода ER-диаграмм [1 – 3]. Выделим объекты и их атрибуты:

– ГАЗЕТА (Шифр_издания*, Название, Цена, ФИО_редактора);

– ОТДЕЛЕНИЕ (Индекс*, Адрес, Название, Работоспособность);

– ТИПОГРАФИЯ (№_типографии*, Адрес, Тип, Работоспособность).

Рассмотрим связь между объектами ГАЗЕТА и ОТДЕЛЕНИЕ (рисунок 1).

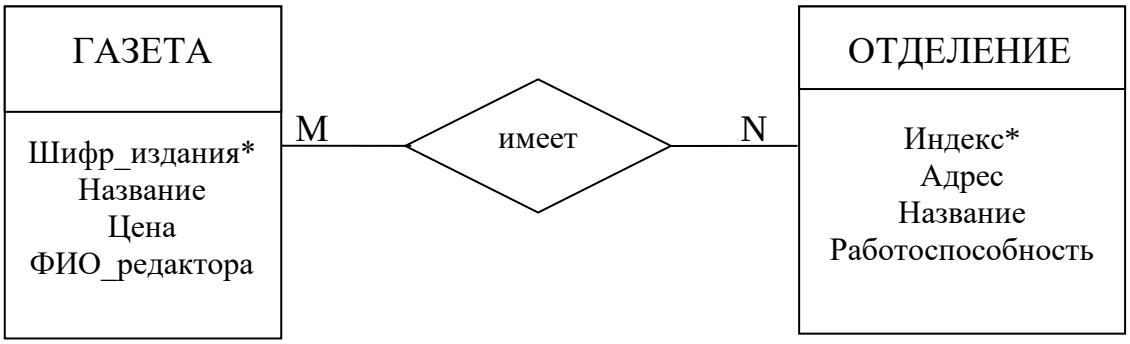

Рисунок 1

Газета определённого названия может находиться в разных почтовых отделениях, а одно почтовое отделение может иметь множество газет разных наименований, следовательно в соответствии с правилом 6 [1 – 3] «Если степень взаимосвязи двух сущностей равна M:N, то независимо от классов принадлежностей сущностей ER-диаграмма преобразуется в три отношения (три таблицы). Схемы первого и второго отношений содержат атрибуты соответствующих сущностей, а в схему третьего отношения включаются ключи обеих сущностей», получим:

– ГАЗЕТА (Шифр_издания*, Название, Цена, ФИО_редактора);

– ОТДЕЛЕНИЕ (Индекс*, Адрес, Название, Работоспособность);

– ГАЗЕТА_ОТДЕЛЕНИЕ (Шифр_издания*, Индекс*, Кол-во_газет).

Рассмотрим связь между объектами ГАЗЕТА и ТИПОГРАФИЯ (рисунок 2).

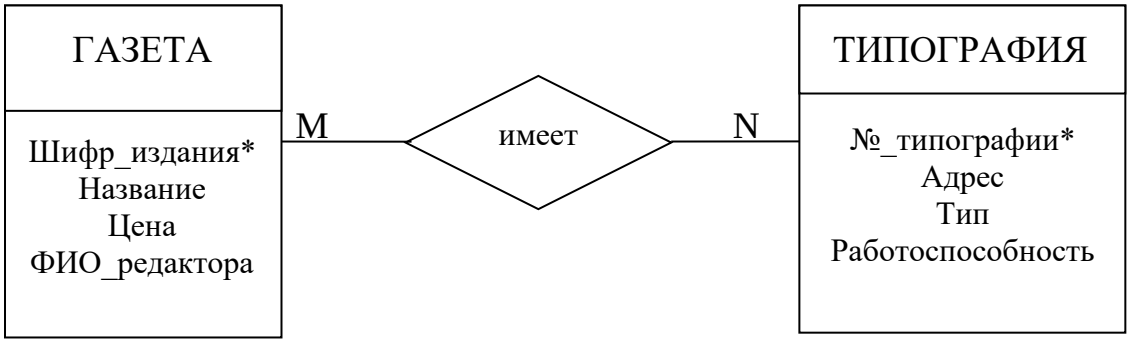

Рисунок 2

Газета определённого названия может печататься в разных типографиях, которые могут находиться в разных городах, с другой стороны, типография выпускает много разных газет, таким образом, в соответствии с правилом 6 [1 – 3] «Если степень взаимосвязи двух сущностей равна M:N, то независимо от классов принадлежностей сущностей ER-диаграмма преобразуется в три отношения (три таблицы). Схемы первого и второго отношений содержат атрибуты соответствующих сущностей, а в схему третьего отношения включаются ключи обеих сущностей», получим:

– ГАЗЕТА (Шифр_издания*, Название, Цена, ФИО_редактора);

– ТИПОГРАФИЯ (№_типографии*, Адрес, Тип, Работоспособность);

– ГАЗЕТА_ТИПОГРАФИЯ (Шифр_издания*, №_типографии*,
Кол-во_газет, Дата_издания).

Таким образом, БД «Почта» включает:

– ГАЗЕТА (Шифр_издания*, Название, Цена, ФИО_редактора);

– ОТДЕЛЕНИЕ (Индекс*, Адрес, Название, Работоспособность);

– ТИПОГРАФИЯ (№_типографии*, Адрес, Тип, Работоспособность);

– ГАЗЕТА_ОТДЕЛЕНИЕ (Шифр_издания*, Индекс*, Кол-во_газет);

– ГАЗЕТА_ТИПОГРАФИЯ (Шифр_издания*, №_типографии*,
Кол-во_газет, Дата_издания).

На рисунке 3 приведена ER-диаграмма для базы данных «Почта»:

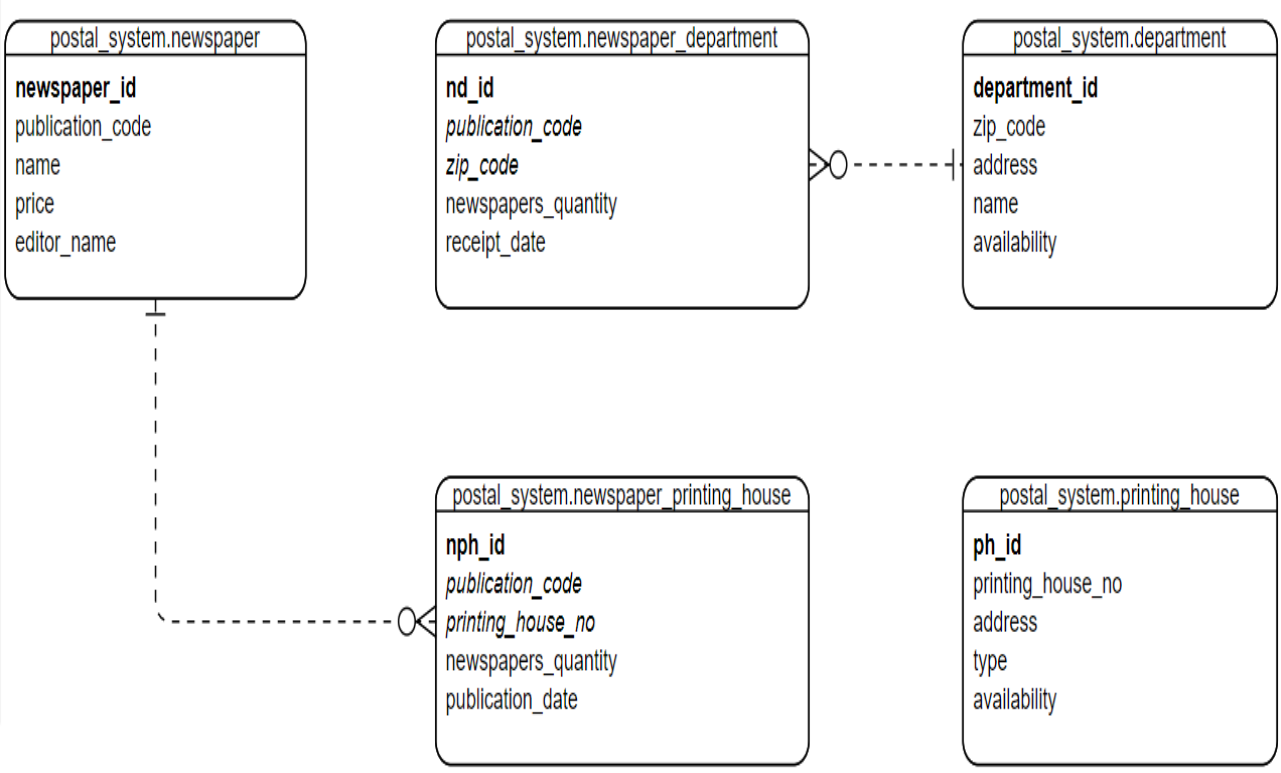

Рисунок 3

На рисунке 4 приведена ER-диаграмма для базы данных «Почта» из SQL Server Management Studio:

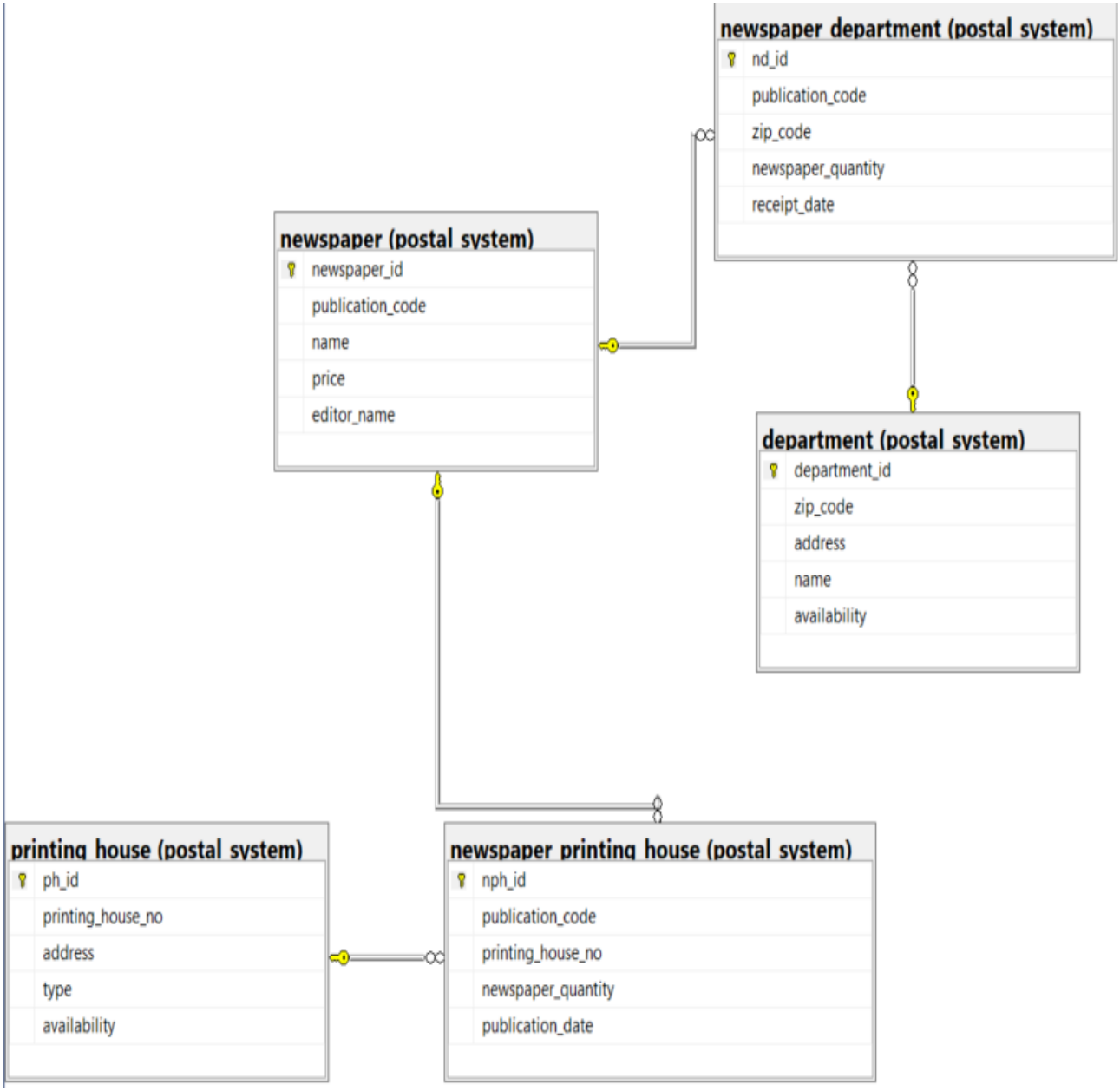

Рисунок 4

3. Создание БД

Запрос 1: создание базы данных Post с использованием основного файла по пути C:\Work\PostDb\Post.mdf, размером 10 Мб, неограниченным максимальным размером и ростом в 10 Мб, а также с использованием файла логирования по пути C:\Work\PostDb\Post_Log.ldf, размером 10 Мб, максимальным размером в 50 Мб и ростом в 10 Мб.

```
CREATE DATABASE Post
ON PRIMARY (NAME = 'Post',
FILENAME = 'C:\Work\PostDb\Post.mdf', SIZE = 10MB, MAXSIZE = 50MB,
FILEGROWTH = 10MB)
LOG ON (NAME = 'Post_Log',
FILENAME = 'C:\Work\PostDb\Post_Log.ldf', SIZE = 10MB, MAXSIZE = 50MB,
FILEGROWTH = 10MB);
```

Результат выполнения приведен на рисунке 5.

```
Commands completed successfully.
Completion time: 2023-11-18T23:37:45.7904125+03:00
```

Рисунок 5

База данных в дереве приведена на рисунке 6.

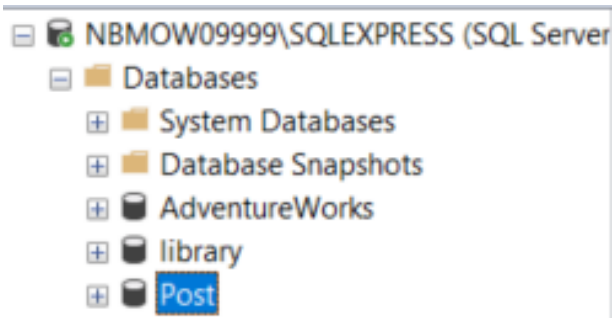

Рисунок 6

4. Создание таблиц и ограничений целостности

Составим структуру для таблицы newspaper:

Описание структуры таблицы БД	Наименование таблицы БД: **Таблица общие параметры газеты**	Имя таблицы: newspaper
Дата разработки: 09.12.2023		
Порядковый номер таблицы: **1**		

№ п/п	Наименование поля	Спецификация данных			
		Имя поля	Тип данных	Ключ	Ограничения целостности
1	Идентификатор	newspaper_id	int	P	NOT NULL
2	Шифр_издания	publication_code	[nvarchar](25)		NOT NULL AND UNIQUE
3	Название	name	[nvarchar](60)		NOT NULL AND UNIQUE
4	Цена	price	smallmoney		NOT NULL AND [price]>=(0)
5	ФИО_редактора	editor_name	[nvarchar](100)		

Составим структуру для таблицы department:

Описание структуры таблицы БД	Наименование таблицы БД: **Таблица общие параметры отделения**	Имя таблицы: department
Дата разработки: 09.12.2023		
Порядковый номер таблицы: **2**		

№ п/п	Наименование поля	Спецификация данных			
		Имя поля	Тип данных	Ключ	Ограничения целостности
1	Идентификатор	department_id	int	P	NOT NULL
2	Индекс	zip_code	[nvarchar](10)		NOT NULL AND UNIQUE
3	Адрес	address	[nvarchar](255)		NOT NULL AND UNIQUE
4	Название	name	[nvarchar](60)		NOT NULL
5	Работоспособность	availability	bit		NOT NULL AND DEFAULT 1

Составим структуру для таблицы printing_house:

Описание структуры таблицы БД	Наименование таблицы БД: **Таблица общие параметры типографии**	Имя таблицы: printing_house
Дата разработки: 09.12.2023		
Порядковый номер таблицы: **3**		

№ п/п	Наименование поля	Спецификация данных			
		Имя поля	Тип данных	Ключ	Ограничения целостности
1	Идентификатор	ph_id	int	P	NOT NULL
2	№_типографии	printing_house_no	int		NOT NULL AND [printing_house_no]>(0) AND UNIQUE
3	Адрес	address	[nvarchar](255)		NOT NULL AND UNIQUE
4	Тип	type	[nvarchar](60)		
5	Работоспособность	availability	bit		NOT NULL AND DEFAULT 1

Составим структуру для таблицы newspaper_department:

Описание структуры таблицы БД	Наименование таблицы БД: **Таблица связи газет с отделением**	Имя таблицы: newspaper_department
Дата разработки: 09.12.2023		
Порядковый номер таблицы: **4**		

№ п/п	Наименование поля	Спецификация данных			
		Имя поля	Тип данных	Ключ	Ограничения целостности
1	Идентификатор	nd_id	int	P	NOT NULL
2	Шифр_издания	publication_code	[nvarchar](25)	F	NOT NULL
3	Индекс	zip_code	[nvarchar](10)	F	NOT NULL
4	Кол-во_газет	newspaper_quantity	int		[newspaper_quantity] >=(0)
5	Дата_поступления	receipt_date	datetime		

Составим структуру для таблицы newspaper_printing_house:

Описание структуры таблицы БД	Наименование таблицы БД: **Таблица связи газет с типографией**	Имя таблицы: newspa-per_printing_house
Дата разработки: 09.12.2023		
Порядковый номер таблицы: **5**		

№ п/п	Наименование поля	Спецификация данных			
		Имя поля	Тип данных	Ключ	Ограничения целостности
1	Идентификатор	nph_id	int	P	NOT NULL
2	Шифр_издания	publication_code	[nvarchar](25)	F	NOT NULL
3	№_типографии	printing_house_no	int	F	NOT NULL AND [printing_house_no]>(0)
4	Кол-во_газет	newspaper_quantity	int		[newspaper_quantity] >=(0)
5	Дата_издания	publication_date	datetime		

Запрос 2: создание схемы postal_system.

```
CREATE SCHEMA postal_system;
```

Результат выполнения приведен на рисунке 7.

```
Commands completed successfully.
Completion time: 2023-12-09T19:22:52.6583749+03:00
```

Рисунок 7

Запрос 3: создание таблицы newspaper в схеме postal_system с полями newspaper_id типа int, которое не может хранить пустые значения, является первичным ключом со счётчиком от 1 с шагом 1, publication_code типа nvarchar(25), которое не может хранить пустые значения и является уникаль-

ным, name типа nvarchar(60), которое не может хранить пустые значения и является уникальным, price типа smallmoney, которое не может хранить пустые значения и не может быть меньше нуля, editor_name типа nvarchar(100).

```
USE Post
CREATE TABLE postal_system.newspaper (
      newspaper_id INT NOT NULL PRIMARY KEY IDENTITY(1, 1),
      publication_code NVARCHAR(25) NOT NULL UNIQUE,
      name NVARCHAR(60) NOT NULL UNIQUE,
      price SMALLMONEY NOT NULL CHECK (Price >= 0),
      editor_name NVARCHAR(100)
);
```

Результат выполнения приведен на рисунке 8.

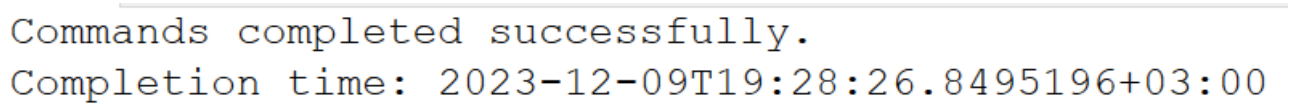

Рисунок 8

Запрос 4: создание таблицы department в схеме postal_system с полями department_id типа int, которое не может хранить пустые значения, является первичным ключом со счётчиком от 1 с шагом 1, zip_code типа nvarchar(10), которое не может хранить пустые значения и является уникальным, address типа nvarchar(255), которое не может хранить пустые значения и является уникальным, name типа nvarchar(60), которое не может хранить пустые значения, availability типа bit, которое не может хранить пустые значения и принимает значение 1 по умолчанию.

```
USE Post
CREATE TABLE postal_system.department (
	department_id INT NOT NULL PRIMARY KEY IDENTITY(1, 1),
	zip_code NVARCHAR(10) NOT NULL UNIQUE,
	address NVARCHAR(255) NOT NULL UNIQUE,
	name NVARCHAR(60) NOT NULL,
	availability BIT NOT NULL DEFAULT 1
);
```

Результат выполнения приведен на рисунке 9.

```
Commands completed successfully.
Completion time: 2023-12-09T19:34:40.9746085+03:00
```

Рисунок 9

Запрос 5: создание таблицы printing_house в схеме postal_system с полями ph_id типа int, которое не может хранить пустые значения, является первичным ключом со счётчиком от 1 с шагом 1, printing_house_no типа int, которое не может хранить пустые значения и значения, меньшие нуля, а также является уникальным, address типа nvarchar(255), которое не может хранить пустые значения и является уникальным, type типа nvarchar(60), availability типа bit, которое не может хранить пустые значения и принимает значение 1 по умолчанию.

```
USE Post
CREATE TABLE postal_system.printing_house (
	ph_id INT NOT NULL PRIMARY KEY IDENTITY(1, 1),
	printing_house_no INT NOT NULL
	CHECK (printing_house_no > 0) UNIQUE,
	address NVARCHAR(255) NOT NULL UNIQUE,
	type NVARCHAR(60),
	availability BIT NOT NULL DEFAULT 1
);
```

Результат выполнения приведен на рисунке 10.

```
Commands completed successfully.
Completion time: 2023-12-09T19:41:20.7408998+03:00
```

Рисунок 10

Запрос 6: создание таблицы newspaper_department в схеме postal_system с полями nd_id типа int, которое не может хранить пустые значения, является первичным ключом со счётчиком от 1 с шагом 1, publication_code типа nvarchar(25), которое не может хранить пустые значения, zip_code типа nvarchar(10), которое не может хранить пустые значения, newspaper_quantity типа int, которое не может хранить значения меньшие нуля, receipt_date типа datetime. Таблица имеет внешние ключи zip_code, который ссылается поле zip_code таблицы department, и publication_code, который ссылается на поле publication_code таблицы newspaper.

```
USE Post
CREATE TABLE postal_system.newspaper_department (
    nd_id INT NOT NULL PRIMARY KEY IDENTITY(1, 1),
    publication_code NVARCHAR(25) NOT NULL,
    zip_code NVARCHAR(10) NOT NULL,
    newspaper_quantity INT CHECK (newspaper_quantity >= 0),
    receipt_date DATETIME,
    FOREIGN KEY (publication_code)
    REFERENCES postal_system.newspaper(publication_code),
    FOREIGN KEY (zip_code) REFERENCES postal_system.department(zip_code)
);
```

Результат выполнения приведен на рисунке 11.

```
Commands completed successfully.
Completion time: 2023-12-09T19:59:14.4163523+03:00
```

Рисунок 11

Запрос 7: создание таблицы newspaper_printing_house в схеме postal_system с полями nph_id типа int, которое не может хранить пустые значения, является первичным ключом со счётчиком от 1 с шагом 1, publication_code типа nvarchar(25), которое не может хранить пустые значения, printing_house_no типа int, которое не может хранить пустые значения и значения меньшие нуля, newspaper_quantity типа int, которое не может хранить значения меньшие нуля, publication_date типа datetime. Таблица имеет внешние ключи printing_house_no, который ссылается поле printing_house_no таблицы printing_house, и publication_code, который ссылается на поле publication_code таблицы newspaper.

```
USE Post
CREATE TABLE postal_system.newspaper_printing_house (
    nph_id INT NOT NULL PRIMARY KEY IDENTITY(1, 1),
    publication_code NVARCHAR(25) NOT NULL,
    printing_house_no INT NOT NULL CHECK (printing_house_no > 0),
    newspaper_quantity INT CHECK (newspaper_quantity >= 0),
    publication_date DATETIME,
    FOREIGN KEY (publication_code)
    REFERENCES postal_system.newspaper(publication_code),
    FOREIGN KEY (printing_house_no)
    REFERENCES postal_system.printing_house(printing_house_no)
);
```

Результат выполнения приведен на рисунке 12.

```
Commands completed successfully.
Completion time: 2023-12-09T20:10:10.9796521+03:00
```

Рисунок 12

5. Заполнение таблиц данными

Запрос 8: вставка значений для полей publication_code, name, price, editor_name таблицы newspaper.

```
USE Post
INSERT INTO postal_system.newspaper
(publication_code, name, price, editor_name)
VALUES
('A1B2C', 'Tech Insight', 2.50, 'Alice Johnson'),
('X7Y8Z', 'Urban Times', 1.80, 'Bob White'),
('P3Q9R', 'Fashion Weekly', 1.75, 'Catherine Black'),
('M5N0O', 'Health Focus', 2.20, 'Daniel Brown'),
('E9F2G', 'Global Digest', 2.00, 'Emily Taylor'),
('K1L6M', 'Science Review', 1.60, 'Frank Miller'),
('S4T7U', 'Business Chronicle', 2.30, 'Grace Davis'),
('H8I2J', 'Entertainment Buzz', 1.95, 'Henry Green'),
('W6X1Y', 'Travel Explorer', 2.10, 'Isabel Carter'),
('D4E7F', 'Food Delight', 1.70, 'Jack Anderson');
```

Результат выполнения приведен на рисунке 13.

```
(10 rows affected)
Completion time: 2023-12-09T20:39:24.1544469+03:00
```

Рисунок 13

Запрос 9: извлечение значения полей publication_code, name, price и editor_name из таблицы newspaper.

```
SELECT publication_code, name, price, editor_name
FROM postal_system.newspaper;
```

Результат выполнения приведен на рисунке 14.

	PublicationCode	Name	Price	EditorName
1	A1B2C	Tech Insight	2,50	Alice Johnson
2	D4E7F	Food Delight	1,70	Jack Anderson
3	E9F2G	Global Digest	2,00	Emily Taylor
4	H8I2J	Entertainment Buzz	1,95	Henry Green
5	K1L6M	Science Review	1,60	Frank Miller
6	M5N0O	Health Focus	2,20	Daniel Brown
7	P3Q9R	Fashion Weekly	1,75	Catherine Black
8	S4T7U	Business Chronicle	2,30	Grace Davis
9	W6X1Y	Travel Explorer	2,10	Isabel Carter
10	X7Y8Z	Urban Times	1,80	Bob White

```
(10 rows affected)
Completion time: 2023-12-09T20:42:34.4666499+03:00
```

Рисунок 14

Запрос 10: вставка значений для полей zip_code, address, name, availability таблицы department.

```
USE Post
INSERT INTO postal_system.department
(zip_code, address, name, availability)
VALUES
('562957', '654 Maple St', 'Central Post Office', 1),
('125258', '876 Birch St', 'Downtown Mail Center', 1),
('843457', '123 Main St', 'Westside Postal Hub', 1),
('376369', '456 Elm St', 'East District Mail Depot', 1),
('237494', '543 Spruce St', 'Southtown Mail Service', 0),
('986540', '321 Pine St', 'North End Mail Center', 1),
('739284', '987 Cedar St', 'City Heights Post Office', 1),
('564832', '210 Walnut St', 'Harborfront Mail Station', 1),
('868455', '789 Oak St', 'Riverside Postal Annex', 1),
('569047', '876 Pineapple St', 'Sunset Valley Mail Office', 0);
```

Результат выполнения приведен на рисунке 15.

```
(10 rows affected)
Completion time: 2023-12-09T20:44:11.7123739+03:00
```

Рисунок 15

Запрос 11: извлечение значения полей ZipCode, Address, Name и Availability из таблицы Department.

```
SELECT zip_code, address, name, availability
FROM postal_system.department;
```

Результат выполнения приведен на рисунке 16.

	ZipCode	Address	Name	Availability
1	125258	876 Birch St	Downtown Mail Center	1
2	237494	543 Spruce St	Southtown Mail Service	0
3	376369	456 Elm St	East District Mail Depot	1
4	562957	654 Maple St	Central Post Office	1
5	564832	210 Walnut St	Harborfront Mail Station	1
6	569047	876 Pineapple St	Sunset Valley Mail Office	0
7	739284	987 Cedar St	City Heights Post Office	1
8	843457	123 Main St	Westside Postal Hub	1
9	868455	789 Oak St	Riverside Postal Annex	1
10	986540	321 Pine St	North End Mail Center	1

```
(10 rows affected)
Completion time: 2023-12-09T20:48:20.9734403+03:00
```

Рисунок 16

Запрос 12: вставка значений для полей printing_house_no, address, type, availability таблицы printing_house.

```
USE Post
INSERT INTO postal_system.printing_house
(printing_house_no, address, type, availability)
VALUES
(1001, '123 Green Street', 'Offset Printing', 1),
(2002, '456 Pine Avenue', 'Digital Printing', 1),
(7007, '789 Meadow Lane', 'Lithographic Printing', 1),
```

```
(3003, '987 Sunflower Street', 'Offset Printing', 1),
(4004, '654 River Boulevard', 'Digital Printing', 0),
(5005, '210 Mountain Road', 'Lithographic Printing', 1),
(6006, '876 Oak Street', 'Offset Printing', 1),
(8008, '543 Sunset Drive', 'Digital Printing', 1),
(9009, '321 Lake Street', 'Lithographic Printing', 1),
(1010, '876 Spring Place', 'Offset Printing', 0);
```

Результат выполнения приведен на рисунке 17.

```
(10 rows affected)
Completion time: 2023-12-09T20:51:20.5963080+03:00
```

Рисунок 17

Запрос 13: извлечение значения полей printing_house_no, address, type и availability из таблицы printing_house.

```
SELECT printing_house_no, address, type, availability
FROM postal_system.printing_house;
```

Результат выполнения приведен на рисунке 18.

	PrintingHouseNo	Address	Type	Availability
1	1001	123 Green Street	Offset Printing	1
2	1010	876 Spring Place	Offset Printing	0
3	2002	456 Pine Avenue	Digital Printing	1
4	3003	987 Sunflower Street	Offset Printing	1
5	4004	654 River Boulevard	Digital Printing	0
6	5005	210 Mountain Road	Lithographic Printing	1
7	6006	876 Oak Street	Offset Printing	1
8	7007	789 Meadow Lane	Lithographic Printing	1
9	8008	543 Sunset Drive	Digital Printing	1
10	9009	321 Lake Street	Lithographic Printing	1

```
(10 rows affected)
Completion time: 2023-12-09T21:55:29.0596347+03:00
```

Рисунок 18

Запрос 14: вставка значений для полей publication_code, zip_code, newspaper_quantity, receipt_date таблицы newspaper_department.

```
USE Post
INSERT INTO postal_system.newspaper_department (publication_code,
zip_code, newspaper_quantity, receipt_date)
VALUES
('A1B2C', '562957', 500, '2023-09-18 10:30:00'),
('A1B2C', '125258', 300, '2023-11-23 15:21:00'),
('X7Y8Z', '125258', 700, '2023-06-16 11:45:00'),
('X7Y8Z', '868455', 900, '2023-07-04 17:26:00'),
('X7Y8Z', '986540', 400, '2023-07-21 10:15:00'),
('P3Q9R', '843457', 300, '2023-08-17 09:15:00'),
('M5N0O', '376369', 450, '2023-05-16 12:20:00'),
('M5N0O', '739284', 600, '2023-05-31 13:52:00'),
('E9F2G', '739284', 600, '2023-06-25 14:00:00'),
('E9F2G', '564832', 500, '2023-07-01 13:07:00'),
('K1L6M', '986540', 350, '2023-11-05 16:30:00'),
('K1L6M', '843457', 900, '2023-11-07 09:23:00'),
('K1L6M', '868455', 245, '2023-10-18 15:30:00'),
('S4T7U', '739284', 800, '2023-10-19 13:10:00'),
('S4T7U', '125258', 950, '2023-09-13 17:18:00'),
('H8I2J', '564832', 250, '2023-10-28 16:45:00'),
('W6X1Y', '868455', 400, '2023-11-14 17:20:00'),
('W6X1Y', '562957', 750, '2023-06-09 11:48:00'),
('D4E7F', '562957', 550, '2023-07-06 18:00:00');
```

Результат выполнения приведен на рисунке 19.

```
(19 rows affected)
Completion time: 2023-12-09T21:57:06.5443092+03:00
```

Рисунок 19

Запрос 15: извлечение значения полей publication_code, zip_code, newspaper_quantity и receipt_date из таблицы newspaper_department.

```
SELECT publication_code, zip_code, newspaper_quantity, receipt_date
FROM postal_system.newspaper_department;
```

Результат выполнения приведен на рисунке 20.

	PublicationCode	ZipCode	NewspaperQuantity	ReceiptDate
1	A1B2C	125258	300	2023-11-23 15:21:00.000
2	A1B2C	562957	500	2023-09-18 10:30:00.000
3	D4E7F	562957	550	2023-07-06 18:00:00.000
4	E9F2G	564832	500	2023-07-01 13:07:00.000
5	E9F2G	739284	600	2023-06-25 14:00:00.000

…

15	W6X1Y	562957	750	2023-06-09 11:48:00.000
16	W6X1Y	868455	400	2023-11-14 17:20:00.000
17	X7Y8Z	125258	700	2023-06-16 11:45:00.000
18	X7Y8Z	868455	900	2023-07-04 17:26:00.000
19	X7Y8Z	986540	400	2023-07-21 10:15:00.000

```
(19 rows affected)
Completion time: 2023-12-09T21:59:20.6379192+03:00
```

Рисунок 20

Запрос 16: вставка значений для полей publication_code, printing_house_no, newspaper_quantity, publication_date таблицы newspaper_printing_house.

```
USE Post
INSERT INTO postal_system.newspaper_printing_house (publication_code,
printing_house_no, newspaper_quantity, publication_date)
VALUES
('A1B2C', 3003, 1500, '2023-01-17 10:30:00'),
('A1B2C', 7007, 1500, '2023-02-25 15:48:00'),
('A1B2C', 5005, 1500, '2023-04-02 12:30:00'),
('X7Y8Z', 2002, 3800, '2023-02-15 11:45:00'),
('P3Q9R', 7007, 1200, '2023-03-24 09:35:00'),
('P3Q9R', 2002, 1200, '2023-03-08 16:42:00'),
('M5N0O', 3003, 2300, '2023-04-03 12:20:00'),
('E9F2G', 5005, 1500, '2023-01-12 14:00:00'),
('E9F2G', 3003, 1500, '2022-12-27 19:47:00'),
('K1L6M', 5005, 2000, '2022-11-14 16:28:00'),
```

```
('K1L6M', 6006, 2000, '2023-04-09 12:30:00'),
('K1L6M', 9009, 2000, '2023-02-28 15:39:00'),
('S4T7U', 6006, 3400, '2023-03-05 13:10:00'),
('H8I2J', 8008, 1000, '2023-02-07 16:45:00'),
('H8I2J', 9009, 1000, '2022-12-11 09:24:00'),
('W6X1Y', 9009, 1750, '2023-04-21 17:20:00'),
('W6X1Y', 8008, 1750, '2023-03-10 15:40:00'),
('D4E7F', 3003, 1350, '2023-04-03 19:00:00');
```

Результат выполнения приведен на рисунке 21.

```
(18 rows affected)
Completion time: 2023-12-09T22:01:50.2329472+03:00
```

Рисунок 21

Запрос 17: извлечение значения полей publication_code, printing_house_no, newspaper_quantity и publication_date из таблицы newspaper_printing_house.

```
SELECT publication_code, printing_house_no, newspaper_quantity,
publication_date
FROM postal_system.newspaper_printing_house;
```

Результат выполнения приведен на рисунке 22.

	PublicationCode	PrintingHouseNo	NewspaperQuantity	PublicationDate
1	A1B2C	3003	1500	2023-01-17 10:30:00.000
2	A1B2C	5005	1500	2023-04-02 12:30:00.000
3	A1B2C	7007	1500	2023-02-25 15:48:00.000
4	D4E7F	3003	1350	2023-04-03 19:00:00.000
5	E9F2G	3003	1500	2022-12-27 19:47:00.000

…

14	P3Q9R	7007	1200	2023-03-24 09:35:00.000
15	S4T7U	6006	3400	2023-03-05 13:10:00.000
16	W6X1Y	8008	1750	2023-03-10 15:40:00.000
17	W6X1Y	9009	1750	2023-04-21 17:20:00.000
18	X7Y8Z	2002	3800	2023-02-15 11:45:00.000

Рисунок 22

```
(18 rows affected)
Completion time: 2023-12-09T22:04:04.3403572+03:00
```

Продолжение рисунка 22

6. Разработка объектов промежуточного слоя

6.1 Разработка представлений

Запрос 18: создание представления postal_system.newspapers_in_departments, которое содержит поля publication_code, name с именем newspaper_name, price, editor_name из таблицы postal_system.newspaper, newspaper_quantity с именем quantity_in_department, receipt_date из таблицы postal_system.newspaper_department, zip_code с именем department_zip_code, address с именем department_address, name с именем department_name из таблицы postal_system.department. Связывание таблиц postal_system.newspaper и postal_system.newspaper_department по условию равенства полей publication_code из обеих таблиц, а также связывание с таблицей postal_system.department по условию равенства полей zip_code из таблиц postal_system.newspaper_department и postal_system.department.

```
CREATE VIEW postal_system.newspapers_in_departments
AS
SELECT
n.publication_code, n.name AS newspaper_name, n.price, n.editor_name,
nd.newspaper_quantity AS quantity_in_department, nd.receipt_date,
d.zip_code AS department_zip_code, d.address AS department_address,
d.name AS department_name
FROM postal_system.newspaper n
INNER JOIN postal_system.newspaper_department nd
ON n.publication_code = nd.publication_code
INNER JOIN postal_system.department d
ON nd.zip_code = d.zip_code;
```

Результат выполнения приведен на рисунке 23.

```
Commands completed successfully.
Completion time: 2023-12-09T22:09:37.3675567+03:00
```

Рисунок 23

Запрос 19: создание представления postal_system.newspapers_in_printing_houses, которое содержит поля publication_code, name с именем newspaper_name, price, editor_name из таблицы postal_system.newspaper, newspaper_quantity с именем quantity_in_printing_house, publication_date из таблицы postal_system.newspaper_printing_house, printing_house_no, address с именем printing_house_address, type с именем printing_house_type из таблицы postal_system.printing_house. Связывание таблиц postal_system.newspaper и postal_system.newspaper_printing_house по условию равенства полей publication_code из обеих таблиц, а также связывание с таблицей postal_system.printing_house по условию равенства полей printing_house_no из таблиц postal_system.printing_house и postal_system.newspaper_printing_house. Накладывается условие на выборку полей такое, что значение поля availability из таблицы printing_house должно быть равно 1.

```
CREATE VIEW postal_system.newspapers_in_printing_houses
AS
SELECT
n.publication_code, n.name AS newspaper_name, n.price, n.editor_name,
nph.newspaper_quantity AS quantity_in_printing_house,
nph.publication_date,
ph.printing_house_no, ph.address AS printing_house_address,
ph.type AS printing_house_type
FROM postal_system.newspaper n
INNER JOIN postal_system.newspaper_printing_house nph
ON nph.publication_code = n.publication_code
INNER JOIN postal_system.printing_house ph
ON ph.printing_house_no = nph.printing_house_no
WHERE ph.availability = 1;
```

Результат выполнения приведен на рисунке 24.

```
Commands completed successfully.
Completion time: 2023-12-09T22:20:36.8523015+03:00
```

Рисунок 24

Запрос 20: создание представления postal_system. printing_houses_addresses_with_type, которое содержит поле type и агрегированное поле address через разделитель “|” с именем Addresses из таблицы postal_system.printing_house с группировкой по полю type.

```
CREATE VIEW postal_system.printing_house_addresses_with_type
AS
SELECT
ph.type, STRING_AGG(ph.address, ' | ') as addresses
FROM postal_system.printing_house ph
GROUP BY ph.type;
```

Результат выполнения приведен на рисунке 25.

```
Commands completed successfully.
Completion time: 2023-12-09T22:40:33.9415198+03:00
```

Рисунок 25

Запрос 21: извлечение значения полей publication_code, newspaper_name, price, editor_name, quantity_in_department, receipt_date, department_zip_code, department_address, department_name из представления postal_system.newspapers_in_departments.

```
SELECT publication_code, newspaper_name, price, editor_name,
quantity_in_department, receipt_date, department_zip_code,
department_address, department_name
FROM postal_system.newspapers_in_departments;
```

Результат выполнения приведен на рисунке 26.

	PublicationCode	NewspaperName	Price	EditorName	QuantityInDepartment	ReceiptDate	DepartmentZipCode	DepartmentAddress	DepartmentName
1	A1B2C	Tech Insight	2,50	Alice Johnson	300	2023-11-23 15:21:00.000	125258	876 Birch St	Downtown Mail Center
2	A1B2C	Tech Insight	2,50	Alice Johnson	500	2023-09-18 10:30:00.000	562957	654 Maple St	Central Post Office
3	D4E7F	Food Delight	1,70	Jack Anderson	550	2023-07-06 18:00:00.000	562957	654 Maple St	Central Post Office
4	E9F2G	Global Digest	2,00	Emily Taylor	500	2023-07-01 13:07:00.000	564832	210 Walnut St	Harborfront Mail Station
5	E9F2G	Global Digest	2,00	Emily Taylor	600	2023-06-25 14:00:00.000	739284	987 Cedar St	City Heights Post Office
6	H8I2J	Entertainment Buzz	1,95	Henry Green	250	2023-10-28 16:45:00.000	564832	210 Walnut St	Harborfront Mail Station
7	K1L6M	Science Review	1,60	Frank Miller	900	2023-11-07 09:23:00.000	843457	123 Main St	Westside Postal Hub

…

14	S4T7U	Business Chronicle	2,30	Grace Davis	800	2023-10-19 13:10:00.000	739284	987 Cedar St	City Heights Post Office
15	W6X1Y	Travel Explorer	2,10	Isabel Carter	750	2023-06-09 11:48:00.000	562957	654 Maple St	Central Post Office
16	W6X1Y	Travel Explorer	2,10	Isabel Carter	400	2023-11-14 17:20:00.000	868455	789 Oak St	Riverside Postal Annex
17	X7Y8Z	Urban Times	1,80	Bob White	700	2023-06-16 11:45:00.000	125258	876 Birch St	Downtown Mail Center
18	X7Y8Z	Urban Times	1,80	Bob White	900	2023-07-04 17:26:00.000	868455	789 Oak St	Riverside Postal Annex
19	X7Y8Z	Urban Times	1,80	Bob White	400	2023-07-21 10:15:00.000	986540	321 Pine St	North End Mail Center

```
(19 rows affected)
Completion time: 2023-12-09T23:00:22.3679366+03:00
```

Рисунок 26

Запрос 22: извлечение значения полей publication_code, newspaper_name, price, editor_name, quantity_in_printing_house, publication_date, printing_house_no, printing_house_address, printing_house_type из представления postal_system.newspapers_in_printing_houses.

```
SELECT publication_code, newspaper_name, price, editor_name,
quantity_in_printing_house, publication_date, printing_house_no,
printing_house_address, printing_house_type
FROM postal_system.newspapers_in_printing_houses;
```

Результат выполнения приведен на рисунке 27.

	PublicationCode	NewspaperName	Price	EditorName	QuantityInPrintingHouse	PublicationDate	PrintingHouseNo	PrintingHouseAddress	PrintingHouseType
1	A1B2C	Tech Insight	2,50	Alice Johnson	1500	2023-01-17 10:30:00.000	3003	987 Sunflower Street	Offset Printing
2	A1B2C	Tech Insight	2,50	Alice Johnson	1500	2023-04-02 12:30:00.000	5005	210 Mountain Road	Lithographic Printing
3	A1B2C	Tech Insight	2,50	Alice Johnson	1500	2023-02-25 15:48:00.000	7007	789 Meadow Lane	Lithographic Printing
4	D4E7F	Food Delight	1,70	Jack Anderson	1350	2023-04-03 19:00:00.000	3003	987 Sunflower Street	Offset Printing
5	E9F2G	Global Digest	2,00	Emily Taylor	1500	2022-12-27 19:47:00.000	3003	987 Sunflower Street	Offset Printing
6	E9F2G	Global Digest	2,00	Emily Taylor	1500	2023-01-12 14:00:00.000	5005	210 Mountain Road	Lithographic Printing
7	H8I2J	Entertainment Buzz	1,95	Henry Green	1000	2023-02-07 16:45:00.000	8008	543 Sunset Drive	Digital Printing
8	H8I2J	Entertainment Buzz	1,95	Henry Green	1000	2022-12-11 09:24:00.000	9009	321 Lake Street	Lithographic Printing

…

Рисунок 27

13	P3Q9R	Fashion Weekly	1,75	Catherine Black	1200	2023-03-08 16:42:00.000	2002	456 Pine Avenue	Digital Printing
14	P3Q9R	Fashion Weekly	1,75	Catherine Black	1200	2023-03-24 09:35:00.000	7007	789 Meadow Lane	Lithographic Printing
15	S4T7U	Business Chronicle	2,30	Grace Davis	3400	2023-03-05 13:10:00.000	6006	876 Oak Street	Offset Printing
16	W6X1Y	Travel Explorer	2,10	Isabel Carter	1750	2023-03-10 15:40:00.000	8008	543 Sunset Drive	Digital Printing
17	W6X1Y	Travel Explorer	2,10	Isabel Carter	1750	2023-04-21 17:20:00.000	9009	321 Lake Street	Lithographic Printing
18	X7Y8Z	Urban Times	1,80	Bob White	3800	2023-02-15 11:45:00.000	2002	456 Pine Avenue	Digital Printing

```
(18 rows affected)
Completion time: 2023-12-09T23:07:35.2382425+03:00
```

Продолжение рисунка 27

Запрос 23: извлечение значения полей type, addresses из представления postal_system.printing_house_addresses_with_type.

```
SELECT type, addresses
FROM postal_system.printing_house_addresses_with_type;
```

Результат выполнения приведен на рисунке 28.

	type	addresses
1	Digital Printing	456 Pine Avenue | 654 River Boulevard | 543 Sunset ...
2	Lithographic Printing	321 Lake Street | 210 Mountain Road | 789 Meadow...
3	Offset Printing	987 Sunflower Street | 876 Oak Street | 876 Spring Pl...

```
(3 rows affected)
Completion time: 2023-12-09T23:09:21.2287348+03:00
```

Рисунок 28

6.2 Разработка хранимых процедур

Запрос 24: создание хранимой процедуры postal_system.transfer_newspaper_to_department с входными параметрами @PublicationCode типа nvarchar(25), @PrintingHouseNo типа int, @DepartmentZipCode типа nvarchar(10), @Quantity типа int. Объявление переменных @NewspaperQuantityInPrintingHouse и @Availability. Если не существует таких элементов, удовлетворяющих запросу вывода 1 из таблицы postal_system.newspaper_printing_house при условии, что publication_code равно значению переменной @PublicationCode, printing_house_no равно значению переменной @PrintingHouseNo и поле availability из таблицы postal_system.printing_house равно 1 при условии, что printing_house_no равно зна-

чению переменной @PrintingHouseNo, то выводим ошибку, что типография отсутствует, закрыта или же не печатает таких газет. Если не существует таких элементов, удовлетворяющих запросу вывода 1 из таблицы postal_system.department при условии, что zip_code равен значению переменной @DepartmentZipCode и availability равно 1, то выводим ошибку, что отделение отсутствует или закрыто. Записываем в переменную @NewspaperQuantityInPrintingHouse значение поля newspaper_quantity из таблицы postal_system.newspaper_printing_house при условии, что publication_code равно значению переменной @PublicationCode и printing_house_no равно значению переменной @PrintingHouseNo. Если значение переменной @Quantity больше, чем значение переменной @NewspaperQuantityInPrintingHouse, то выводим ошибку, что в типографии меньше количества газет, которое необходимо отправить в отделение, иначе обновляем таблицу postal_system.newspaper_printing_house, устанавливаем значение поля newspaper_quantity в newspaper_quantity - @Quantity при условии, что publication_code равно значению переменной @PublicationCode и printing_house_no равно @PrintingHouseNo. Если не существует таких элементов, удовлетворяющих запросу вывода 1 из таблицы postal_system.newspaper_department при условии, что publication_code равно значению переменной @PublicationCode и zip_code равно значению переменной @DepartmentZipCode, то вставляем в таблицу postal_system.newspaper_department в поля publication_code, zip_code, newspaper_quantity, receipt_date значения @PublicationCode, @DepartmentZipCode, @Quantity, GETDATE(), иначе обновляем таблицу postal_system.newspaper_department, устанавливаем значение поля newspaper_quantity в newspaper_quantity + @Quantity и значение поля receipt_date в GETDATE() при условии, что publication_code равно значению переменной @PublicationCode и zip_code равен значению переменной @DepartmentZipCode.

```
USE Post;
GO
CREATE PROCEDURE postal_system.transfer_newspaper_to_department
```

```
    @PublicationCode NVARCHAR(25),
    @PrintingHouseNo INT,
    @DepartmentZipCode NVARCHAR(10),
    @Quantity INT
AS
BEGIN
    DECLARE @NewspaperQuantityInPrintingHouse INT;
    DECLARE @Availability INT;

    IF NOT EXISTS (
        SELECT 1
        FROM postal_system.newspaper_printing_house
        WHERE publication_code = @PublicationCode AND
              printing_house_no = @PrintingHouseNo AND
             (SELECT availability
              FROM postal_system.printing_house
              WHERE printing_house_no = @PrintingHouseNo) = 1
    )
    BEGIN
        RAISERROR('Printing house is absent or closed or don''t print
such newspapers!', -1, 1);
       RETURN;
    END;

    IF NOT EXISTS (
        SELECT 1
        FROM postal_system.department
        WHERE zip_code = @DepartmentZipCode AND
              availability = 1
    )
    BEGIN
        RAISERROR('Department is absent or closed!', -1, 1);
       RETURN;
    END;
```

```
    SELECT @NewspaperQuantityInPrintingHouse = newspaper_quantity
    FROM postal_system.newspaper_printing_house
    WHERE publication_code = @PublicationCode AND
            printing_house_no = @PrintingHouseNo;

     BEGIN TRANSACTION
     IF (@Quantity > @NewspaperQuantityInPrintingHouse)
     BEGIN
          RAISERROR('There are fewer newspapers in the printing house
than needed!', -1, 1);
          ROLLBACK TRANSACTION;
          RETURN;
     END
     ELSE
     BEGIN
          UPDATE postal_system.newspaper_printing_house
        SET newspaper_quantity = newspaper_quantity - @Quantity
        WHERE publication_code = @PublicationCode AND
                  printing_house_no = @PrintingHouseNo;
     END;

    IF NOT EXISTS (
        SELECT 1
        FROM postal_system.newspaper_department
        WHERE publication_code = @PublicationCode AND
                  zip_code = @DepartmentZipCode
    )
    BEGIN
        INSERT INTO postal_system.newspaper_department (publication_code,
zip_code, newspaper_quantity, receipt_date)
        VALUES (@PublicationCode, @DepartmentZipCode, @Quantity,
GETDATE())
    END
```

```
    ELSE
    BEGIN
        UPDATE postal_system.newspaper_department
        SET newspaper_quantity = newspaper_quantity + @Quantity,
            receipt_date = GETDATE()
        WHERE publication_code = @PublicationCode AND
              zip_code = @DepartmentZipCode;
    END;

    COMMIT TRANSACTION;
END;
```

Результат выполнения приведен на рисунке 29.

```
Commands completed successfully.
Completion time: 2023-12-09T23:20:05.0935126+03:00
```

Рисунок 29

Запрос 25: выполнение процедуры postal_system.transfer_newspaper_to_department с параметрами ‘A1B2C’, 1010, ‘125258’, 200.

```
EXEC postal_system.transfer_newspaper_to_department
'A1B2C',
1010,
'125258',
200;
```

Результат выполнения приведен на рисунке 30.

```
Printing house is absent or closed or don't print such newspapers!
Completion time: 2023-12-09T23:33:56.3627743+03:00
```

Рисунок 30

Запрос 26: выполнение процедуры postal_system.transfer_newspaper_to_department с параметрами ‘A1B2C’, 5005, ‘ZIP-CODE’, 200.

```
EXEC postal_system.transfer_newspaper_to_department
'A1B2C',
5005,
'ZIPCODE',
200;
```

Результат выполнения приведен на рисунке 31.

```
Department is absent or closed!
Completion time: 2023-12-09T23:35:06.4419259+03:00
```

Рисунок 31

Запрос 27: выполнение процедуры postal_system.transfer_newspaper_to_department с параметрами ‘A1B2C’, 5005, ‘125258’, 1600.

```
EXEC postal_system.transfer_newspaper_to_department
'A1B2C',
5005,
'125258',
1600;
```

Результат выполнения приведен на рисунке 32.

```
There are fewer newspapers in the printing house than needed!
Completion time: 2023-12-09T23:36:16.2810610+03:00
```

Рисунок 32

Запрос 28: извлечение значения полей publication_code, newspaper_name, price, editor_name, quantity_in_department, receipt_date, department_zip_code, department_address, department_name из представления postal_system.newspapers_in_departments при условии, что значение поля publication_code равно ‘A1B2C’.

```
SELECT publication_code, newspaper_name, price, editor_name,
quantity_in_department, receipt_date, department_zip_code,
department_address, department_name
```

```
FROM postal_system.newspapers_in_departments
WHERE publication_code = 'A1B2C';
```

Результат выполнения приведен на рисунке 33.

	PublicationCode	NewspaperName	Price	EditorName	QuantityInDepartment	ReceiptDate	DepartmentZipCode	DepartmentAddress	DepartmentName
1	A1B2C	Tech Insight	2,50	Alice Johnson	300	2023-11-23 15:21:00.000	125258	876 Birch St	Downtown Mail Center
2	A1B2C	Tech Insight	2,50	Alice Johnson	500	2023-09-18 10:30:00.000	562957	654 Maple St	Central Post Office

```
(2 rows affected)
Completion time: 2023-12-09T23:38:28.0889425+03:00
```

Рисунок 33

Запрос 29: выполнение процедуры postal_system.transfer_newspaper_to_department с параметрами ‘A1B2C’, 5005, ‘125258’, 700.

```
EXEC postal_system.transfer_newspaper_to_department
'A1B2C',
5005,
'125258',
700;
```

Результат выполнения приведен на рисунке 34.

```
(1 row affected)
(1 row affected)
Completion time: 2023-12-09T23:41:07.1689509+03:00
```

Рисунок 34

Запрос 30: извлечение значения полей publication_code, newspaper_name, price, editor_name, quantity_in_department, receipt_date, department_zip_code, department_address, department_name из представления postal_system.newspapers_in_departments при условии, что значение поля publication_code равно ‘A1B2C’.

```
SELECT publication_code, newspaper_name, price, editor_name,
quantity_in_department, receipt_date, department_zip_code,
```

```
department_address, department_name
FROM postal_system.newspapers_in_departments
WHERE publication_code = 'A1B2C';
```

Результат выполнения приведен на рисунке 35.

	PublicationCode	NewspaperName	Price	EditorName	QuantityInDepartment	ReceiptDate	DepartmentZipCode	DepartmentAddress	DepartmentName
1	A1B2C	Tech Insight	2,50	Alice Johnson	1000	2023-11-26 00:45:17.383	125258	876 Birch St	Downtown Mail Center
2	A1B2C	Tech Insight	2,50	Alice Johnson	500	2023-09-18 10:30:00.000	562957	654 Maple St	Central Post Office

```
(2 rows affected)
Completion time: 2023-12-09T23:42:19.9616472+03:00
```

Рисунок 35

Запрос 31: создание хранимой процедуры postal_system.create_newspaper с входными параметрами @PublicationCode типа nvarchar(25), @Name типа nvarchar(60), @Price типа smallmoney, @EditorName типа nvarchar(100). Если существует такой элемент, удовлетворяющий запросу вывода 1 из таблицы postal_system.newspaper при условии, что publication_code равно значению переменной @PublicationCode, то выводим ошибку, что такая газета уже существует. В противном случае вставляем значения переменных @PublicationCode, @Name, @Price и @EditorName в таблицу postal_system.newspaper и возвращаем 0, если не произошло ошибки. Если при вставке возникла ошибка, то выводим на экран значение функции ERROR_MESSAGE() и возвращаем -1.

```
USE Post;
GO
CREATE PROCEDURE postal_system.create_newspaper
    @PublicationCode nvarchar(25),
    @Name nvarchar(60),
    @Price smallmoney,
    @EditorName nvarchar(100)
AS
BEGIN
    IF EXISTS (
        SELECT 1
```

```
            FROM postal_system.newspaper
            WHERE publication_code = @PublicationCode
      )
      BEGIN
            RAISERROR('Such newspaper already exists!', -1, 1);
         RETURN;
      END;

      BEGIN TRY
            INSERT INTO postal_system.newspaper
            (publication_code, name, price, editor_name)
            VALUES
            (@PublicationCode, @Name, @Price, @EditorName);
            RETURN 0;
      END TRY
      BEGIN CATCH
            PRINT ERROR_MESSAGE();
            RETURN -1;
      END CATCH;
END;
```

Результат выполнения приведен на рисунке 36.

```
Commands completed successfully.
Completion time: 2023-12-10T00:10:05.1596786+03:00
```

Рисунок 36

Запрос 32: выполнение процедуры postal_system.create_newspaper с параметрами ‘NNN123’, ‘Test newspaper’, -25, NULL.

```
EXEC postal_system.create_newspaper
'NNN123',
'Test newspaper',
-25,
NULL;
```

Результат выполнения приведен на рисунке 37.

```
(0 rows affected)
The INSERT statement conflicted with the CHECK constraint
"CK__newspaper__price__5BE2A6F2".
The conflict occurred in database "Post", table
"postal_system.newspaper", column 'price'.
Completion time: 2023-12-10T01:01:52.1097040+03:00
```

Рисунок 37

Запрос 33: выполнение процедуры postal_system.create_newspaper с параметрами ‘NNN123’, ‘Test newspaper’, 2, NULL.

```
EXEC postal_system.create_newspaper
'NNN123',
'Test newspaper',
2,
NULL;
```

Результат выполнения приведен на рисунке 38.

```
(1 row affected)
Completion time: 2023-12-10T01:04:38.7904645+03:00
```

Рисунок 38

Запрос 34: создание хранимой процедуры postal_system.delete_newspaper с входным параметром @PublicationCode типа nvarchar(25). Если не существует такого элемента, удовлетворяющего запросу вывода 1 из таблицы postal_system.newspaper при условии, что publication_code равно значению переменной @PublicationCode, то выводим ошибку, что такой газеты не существует. В противном случае удаляем запись из таблицы postal_system.newspaper при условии, что значение поля publication_code равно значению переменной @PublicationCode.

```
USE Post;
GO
CREATE PROCEDURE postal_system.delete_newspaper
```

```
    @PublicationCode nvarchar(25)
AS
BEGIN
    IF NOT EXISTS (
        SELECT 1
        FROM postal_system.newspaper
        WHERE publication_code = @PublicationCode
    )
    BEGIN
        RAISERROR('Such newspaper doesn''t exist!', -1, 1);
      RETURN;
    END;

    DELETE FROM postal_system.newspaper
    WHERE publication_code = @PublicationCode;
END;
```

Результат выполнения приведен на рисунке 39.

```
Commands completed successfully.
Completion time: 2023-12-10T01:08:42.7562100+03:00
```

Рисунок 39

Запрос 35: выполнение процедуры postal_system.delete_newspaper с параметром ‘NNN1230’.

```
EXEC postal_system.delete_newspaper
'NNN1230';
```

Результат выполнения приведен на рисунке 40.

```
Such newspaper doesn't exist!
Completion time: 2023-12-10T01:09:36.7027229+03:00
```

Рисунок 40

Запрос 36: выполнение процедуры postal_system.delete_newspaper с параметром ‘NNN123’.

```
EXEC postal_system.delete_newspaper
'NNN123';
```

Результат выполнения приведен на рисунке 41.

```
(1 row affected)
Completion time: 2023-12-10T01:13:04.9331968+03:00
```

Рисунок 41

6.3 Разработка UDF-ов

Запрос 37: использование базы данных Post для удаления функции postal_system.get_min_operational_delay, если уже имеется, и её создание с параметром @PublicationCode типа varchar(25), которая возвращает значение типа int. Функция создаёт переменную @MinDateDifference и присваивает в неё результат выполнения запроса, который извлекает минимальную разницу в днях между полями publication_date из таблицы postal_system.newspaper_printing_house и receipt_date из таблицы postal_system.newspaper_department, обе таблицы связываются по условию равенства значения полей publication_code при условии, что значение поля publication_code из таблицы postal_system.newspaper_printing_house равно значению переменной @PublicationCode. Возврат значения переменной @MinDateDifference.

```
USE Post;
GO
IF OBJECT_ID('postal_system.get_min_operational_delay') IS NOT NULL
     DROP FUNCTION postal_system.get_min_operational_delay;
GO
CREATE FUNCTION postal_system.get_min_operational_delay (
     @PublicationCode NVARCHAR(25)
)
RETURNS INT
AS BEGIN
     DECLARE @MinDateDifference INT;
```

```
    SELECT @MinDateDifference = MIN(DATEDIFF(DAY,
           nph.publication_date, nd.receipt_date))
    FROM postal_system.newspaper_printing_house nph
    JOIN postal_system.newspaper_department nd
     ON nph.publication_code = nd.publication_code
     WHERE nph.publication_code = @PublicationCode;

    RETURN @MinDateDifference;
END;
GO
```

Результат выполнения приведен на рисунке 42.

```
Commands completed successfully.
Completion time: 2023-12-10T01:20:48.9920034+03:00
```

Рисунок 42

Запрос 38: извлечение значения полей publication_code, publication_date и printing_house_no из представления postal_system.newspapers_in_printing_houses при условии, что значение поля publication_code равно ‘A1B2C’.

```
SELECT publication_code, publication_date, printing_house_no
FROM postal_system.newspapers_in_printing_houses
WHERE publication_code = 'A1B2C';
```

Результат выполнения приведен на рисунке 43.

	PublicationCode	PublicationDate	PrintingHouseNo
1	A1B2C	2023-01-17 10:30:00.000	3003
2	A1B2C	2023-04-02 12:30:00.000	5005
3	A1B2C	2023-02-25 15:48:00.000	7007

```
(3 rows affected)
Completion time: 2023-12-10T01:24:35.6117089+03:00
```

Рисунок 43

Запрос 39: извлечение значения полей publication_code, receipt_date, department_name из представления postal_system.newspapers_in_departments при условии, что значение поля publication_code равно ‘A1B2C’.

```
SELECT publication_code, receipt_date, department_name
FROM postal_system.newspapers_in_departments
WHERE publication_code = 'A1B2C';
```

Результат выполнения приведен на рисунке 44.

	PublicationCode	ReceiptDate	DepartmentName
1	A1B2C	2023-11-26 00:45:17.383	Downtown Mail Center
2	A1B2C	2023-09-18 10:30:00.000	Central Post Office

```
(2 rows affected)
Completion time: 2023-12-10T01:27:44.4995406+03:00
```

Рисунок 44

Запрос 40: извлечение значения, возвращаемого из функции postal_system.get_min_operational_delay с переданным параметром ‘A1B2C’.

```
SELECT postal_system.get_min_operational_delay('A1B2C');
```

Результат выполнения приведен на рисунке 45.

	(No column name)
1	169

```
(1 row affected)
Completion time: 2023-12-10T01:28:53.2240537+03:00
```

Рисунок 45

Запрос 41: использование базы данных Post для удаления функции postal_system.get_price_stat_on_departments, если уже имеется, и её создание без параметров, которая возвращает табличное значение TABLE. Функция возвращает запрос, который извлекает поле name с именем department_name из таблицы postal_system.department, сумму произведения полей price из таблицы postal_system.newspaper и newspaper_quantity из таблицы post-

al_system.newspaper_department с именем total_price. Таблица postal_system.department соединяется с postal_system.newspaper_department на условие равенства полей zip_code из обеих таблиц, далее соединяем с таблицей postal_system.newspaper на условие равенства полей publication_code из обеих таблиц, затем производим группировку по полю name таблицы postal_system.department.

```
USE Post;
GO
IF OBJECT_ID('postal_system.get_price_stat_on_departments') IS NOT NULL
     DROP FUNCTION postal_system.get_price_stat_on_departments;
GO
CREATE FUNCTION postal_system.get_price_stat_on_departments()
RETURNS TABLE
AS
RETURN
(
     SELECT d.name AS department_name,
             SUM(n.price * nd.newspaper_quantity) AS total_price
     FROM postal_system.department d
     JOIN postal_system.newspaper_department nd
     ON d.zip_code = nd.zip_code
     JOIN postal_system.newspaper n
     ON n.publication_code = nd.publication_code
     GROUP BY d.name
);
GO
```

Результат выполнения приведен на рисунке 46.

```
Commands completed successfully.
Completion time: 2023-12-10T01:33:36.9551476+03:00
```

Рисунок 46

Запрос 42: извлечение значения полей newspaper_name, price, quantity_in_department, department_name из представления postal_system.newspapers_in_departments с сортировкой по полю department_name.

```
SELECT newspaper_name, price, quantity_in_department, department_name
FROM postal_system.newspapers_in_departments
ORDER BY department_name;
```

Результат выполнения приведен на рисунке 47.

	NewspaperName	Price	QuantityInDepartment	DepartmentName
1	Tech Insight	2,50	500	Central Post Office
2	Food Delight	1,70	550	Central Post Office
3	Travel Explorer	2,10	750	Central Post Office
4	Health Focus	2,20	600	City Heights Post Office
5	Business Chronicle	2,30	800	City Heights Post Office
6	Global Digest	2,00	600	City Heights Post Office

…

15	Science Review	1,60	245	Riverside Postal Annex
16	Urban Times	1,80	900	Riverside Postal Annex
17	Travel Explorer	2,10	400	Riverside Postal Annex
18	Fashion Weekly	1,75	300	Westside Postal Hub
19	Science Review	1,60	900	Westside Postal Hub

```
(19 rows affected)
Completion time: 2023-12-10T01:36:29.4528225+03:00
```

Рисунок 47

Запрос 43: извлечение значения полей department_name и total_price из функции postal_system.get_price_stat_on_departments.

```
SELECT department_name, total_price
FROM postal_system.get_price_stat_on_departments();
```

Результат выполнения приведен на рисунке 48.

	DepartmentName	TotalPrice
1	Central Post Office	3760,00
2	City Heights Post Office	4360,00
3	Downtown Mail Center	5945,00
4	East District Mail Depot	990,00
5	Harborfront Mail Station	1487,50
6	North End Mail Center	1280,00
7	Riverside Postal Annex	2852,00
8	Westside Postal Hub	1965,00

```
(8 rows affected)
Completion time: 2023-12-10T01:37:53.3541628+03:00
```

Рисунок 48

Запрос 44: использование базы данных Post для удаления функции postal_system.get_price_stat_on_type_printing_house, если уже имеется, и её создание, которая возвращает табличное значение TABLE с полями type типа nvarchar(60), quantity типа int, total_price типа smallmoney. В таблицу @Result происходит вставка результатов запроса, который извлекает значения полей type из таблицы postal_system.printing_house, сумму значения полей newspaper_quantity из таблицы postal_system.newspaper_printing_house с именем quantity и сумму произведения полей price из таблицы postal_system.newspaper и newspaper-quantity из таблицы postal_system.newspaper_printing_house с именем total_price. Соединение таблиц postal_system.printing_house и postal_system.newspaperP

_printing_house на условие равенства полей printing_house_no из обеих таблиц, далее соединение с таблицей postal_system.newspaper на равенство полей publication_code из таблиц postal_system.newspaper и postal_system.newspaper_printing_house. Группировка по полю type таблицы postal_system.printing_house.

```
USE Post;
GO
IF OBJECT_ID('postal_system.get_price_stat_on_type_printing_house') IS
NOT NULL
```

```
	DROP FUNCTION postal_system.get_price_stat_on_type_printing_house;
GO
CREATE FUNCTION postal_system.get_price_stat_on_type_printing_house()
RETURNS @Result TABLE (type NVARCHAR(60), quantity INT, total_price
SMALLMONEY)
AS BEGIN
	INSERT INTO @Result
	SELECT ph.type,
		SUM(nph.newspaper_quantity) AS quantity,
		SUM(n.price * nph.newspaper_quantity) AS total_price
	FROM postal_system.printing_house ph
	JOIN postal_system.newspaper_printing_house nph
	ON ph.printing_house_no = nph.printing_house_no
	JOIN postal_system.newspaper n
	ON n.publication_code = nph.publication_code
	GROUP BY ph.type;
	RETURN;
END;
GO
```

Результат выполнения приведен на рисунке 49.

```
Commands completed successfully.
Completion time: 2023-12-10T01:46:18.7020454+03:00
```

Рисунок 49

Запрос 45: извлечение значения полей printing_house_type, quantity_in_printing_house, price из представления postal_system.newspapers_in_printing_houses с сортировкой по полю printing_house_type.

```
SELECT printing_house_type, quantity_in_printing_house, price
FROM postal_system.newspapers_in_printing_houses
ORDER BY printing_house_type;
```

Результат выполнения приведен на рисунке 50.

	PrintingHouseType	QuantityInPrintingHouse	Price
1	Digital Printing	1200	1,75
2	Digital Printing	3800	1,80
3	Digital Printing	1000	1,95
4	Digital Printing	1750	2,10

…

13	Offset Printing	2000	1,60
14	Offset Printing	3400	2,30
15	Offset Printing	1500	2,50
16	Offset Printing	1350	1,70
17	Offset Printing	1500	2,00
18	Offset Printing	2300	2,20

```
(18 rows affected)
Completion time: 2023-12-10T01:48:36.4107221+03:00
```

Рисунок 50

Запрос 46: извлечение значения полей type, quantity и total_price из таблицы, возвращаемой функцией postal_system.get_price_stat_on_type_printing_house.

```
SELECT Type, quantity, total_price
FROM postal_system.get_price_stat_on_type_printing_house();
```

Результат выполнения приведен на рисунке 51.

	Type	Quantity	TotalPrice
1	Digital Printing	7750	14565,00
2	Lithographic Printing	11750	22875,00
3	Offset Printing	12050	25125,00

```
(3 rows affected)
Completion time: 2023-12-10T01:50:41.5642975+03:00
```

Рисунок 51

7. Разработка стратегии резервного копирования

Запрос 47: полное резервное копирование базы данных Post в файл C:\Work\PostDb\Backup\PostFull.bak. Резервное копирования файла журнала ло-

гирования в файл C:\Work\PostDb\Backup\PostLogs.bak. Резервное копирование всех изменений после последнего полного резервного копирования в файл C:\Work\PostDb\Backup\PostDiff.bak. Резервное копирование файла журнала логирования без перезаписи в файл C:\Work\PostDb\Backup\PostLogs.bak.

```
BACKUP DATABASE Post
TO DISK = 'C:\Work\PostDb\Backup\PostFull.bak';
BACKUP LOG Post
TO DISK = 'C:\Work\PostDb\Backup\PostLogs.bak';
BACKUP DATABASE Post
TO DISK = 'C:\Work\PostDb\Backup\PostDiff.bak'
WITH DIFFERENTIAL;
BACKUP LOG Post
TO DISK = 'C:\Work\PostDb\Backup\PostLogs.bak'
WITH NOINIT;
```

Результат выполнения приведен на рисунке 52.

```
Processed 648 pages for database 'Post', file 'Post' on file 1.
Processed 2 pages for database 'Post', file 'Post_Log' on file 1.
BACKUP DATABASE successfully processed 650 pages in 0.018 seconds (281.901 MB/sec).
Processed 3 pages for database 'Post', file 'Post_Log' on file 1.
BACKUP LOG successfully processed 3 pages in 0.002 seconds (11.718 MB/sec).
Processed 72 pages for database 'Post', file 'Post' on file 1.
Processed 2 pages for database 'Post', file 'Post_Log' on file 1.
BACKUP DATABASE WITH DIFFERENTIAL successfully processed 74 pages
in 0.009 seconds (63.802 MB/sec).
Processed 7 pages for database 'Post', file 'Post_Log' on file 2.
BACKUP LOG successfully processed 7 pages in 0.004 seconds (12.695 MB/sec).

Completion time: 2023-12-10T01:58:59.2062656+03:00
```

Рисунок 52

Результат создания файлов на диске представлен на рисунке 53.

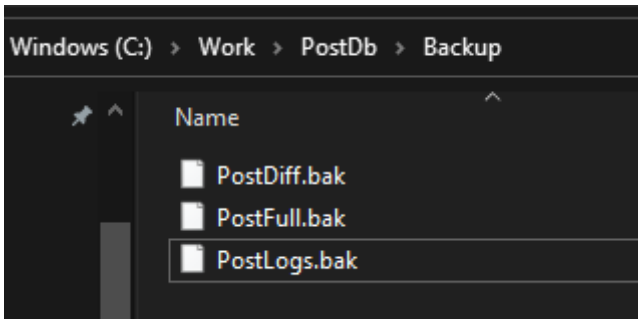

Рисунок 53

После переименования расширения файла базы данных результат поломки базы данных Post представлен на рисунке 54.

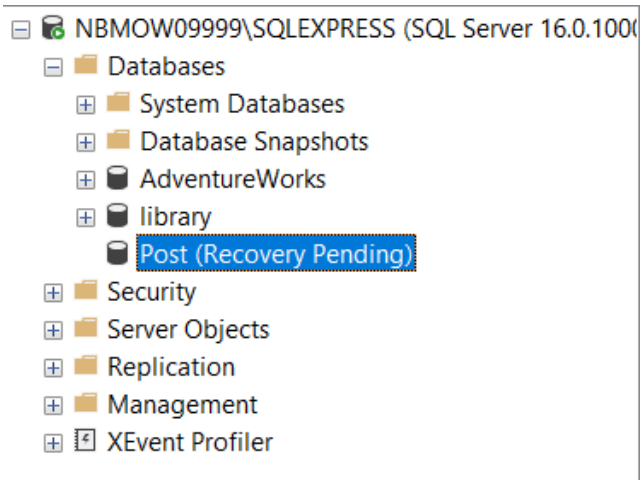

Рисунок 54

Запрос 48: восстановление базы данных Post из файла C:\Work\PostDb\Backup\PostFull.bak с возможностью замены существующей базы данных с продолжением процедуры восстановления. Восстановление дифференциальной резервной копии базы данных Post из файла C:\Work\PostDb\Backup\PostDiff.bak с продолжением процедуры восстановления. Восстановление файла журнала для базы данных Post из файла C:\Work\PostDb\Backup\PostLogs.bak с завершением процедуры восстановления.

```
RESTORE DATABASE Post
FROM DISK = 'C:\Work\PostDb\Backup\PostFull.bak'
```

```
WITH NORECOVERY, REPLACE;
RESTORE DATABASE Post
FROM DISK = 'C:\Work\PostDb\Backup\PostDiff.bak'
WITH NORECOVERY;
RESTORE LOG Post
FROM DISK = 'C:\Work\PostDb\Backup\PostLogs.bak'
WITH RECOVERY;
```

Результат выполнения приведен на рисунке 55.

```
Processed 648 pages for database 'Post', file 'Post' on file 1.
Processed 2 pages for database 'Post', file 'Post_Log' on file 1.
RESTORE DATABASE successfully processed 650 pages in 0.010 seconds (507.421 MB/sec).
Processed 0 pages for database 'Post', file 'Post' on file 1.
Processed 3 pages for database 'Post', file 'Post_Log' on file 1.
RESTORE LOG successfully processed 3 pages in 0.002 seconds (11.718 MB/sec).
Processed 72 pages for database 'Post', file 'Post' on file 1.
Processed 2 pages for database 'Post', file 'Post_Log' on file 1.
RESTORE DATABASE successfully processed 74 pages in 0.004 seconds (143.554 MB/sec).
Completion time: 2023-12-10T02:28:40.9860369+03:00
```

Рисунок 55

Заключение

В результате выполнения индивидуального домашнего задания были закреплены теоретические знания и практические навыки, полученные в ходе изучения курса «Базы данных».

Была спроектирована и создана база данных «Почта», которая содержит информацию о газетах, отделениях и типографиях. Также созданы таблицы с ограничениями целостности, они заполнены тестовыми данными, разработаны представления, хранимые процедуры и UDF, разработана стратегия резервного копирования и созданы резервные копии базы данных.

Список использованных источников

1. Горячев А. В., Новакова Н. Е. Распределенные базы данных. Мет. указания к лаб. работам., СПб. Изд-во СПбГЭТУ «ЛЭТИ», 2008
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. Документация по Microsoft SQL: https://learn.microsoft.com/en-us/sql/?view=sql-server-ver16.
4. Курс «базы данных» на учебном ресурсе moodle: https://vec.etu.ru/moodle/enrol/index.php?id=14314.

Дополнительная литература

5. Системы баз данных. Полный курс. / Гарсия-Молина Гектор, Ульман Джефери Д., Дженифер Уидом : пер. с англ., М.: Издательский дом «Вильямс», 2003, - 1088 с.