|  |
| --- |
| МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ  федеральное г**о**сударственное АВТОНОМНОЕ образовательное учреждение высшего образования  «Национальный исследовательский ядерный университет «МИФИ» |
| **Димитровградский инженерно-технологический институт –**  филиал федерального государственного автономного образовательного учреждения высшего образования «Национальный исследовательский ядерный университет «МИФИ»  **(ДИТИ НИЯУ МИФИ)** |

**Специальность 09.02.07 Информационные системы и программирование**

**ОТЧЁТ**

**по учебной практике**

|  |  |
| --- | --- |
| **Наименование практики** | УП.11.01 Учебная практика (по профилю специальности) |
| **Профессиональный модуль** | ПМ.11. Разработка, администрирование и защита баз данных |
| **Обучающийся** | Рыбалев Денис Александрович |
| **Группа** | 431 |
| **Место проведения практики:** | ДИТИ НИЯУ МИФИ, учебный корпус №6 (Куйбышева, 300), лаборатория информационных технологий и мастерская по наладке технологического оборудования по профилю выбираемой рабочей профессии, ауд. №36 |
| **Сроки прохождения практики:** | с 12.01.2026 по 07.02.2026 |
| **Руководитель/ли практики от образовательной организации:** | Фролова Валерия Витальевна |
| **Итоговая оценка** |  |
| **Подпись руководителя практики** |  |
| **Дата** | 07.02.2026 |

Димитровград, 2026

**СОДЕРЖАНИЕ**

[**Инструкционная карта №1** 3](#_Toc220948021)

[**Инструкционная карта №2** 4](#_Toc220948022)

[**Инструкционная карта №3** 14](#_Toc220948023)

[**Инструкционная карта №4** 23](#_Toc220948024)

[**Инструкционная карта №5** 35](#_Toc220948025)

[**Инструкционная карта №6** 41](#_Toc220948026)

[**Инструкционная карта №7** 49](#_Toc220948027)

[**Инструкционная карта №8** 53](#_Toc220948028)

[**Инструкционная карта №9** 61](#_Toc220948029)

[**Инструкционная карта №10** 64](#_Toc220948030)

[**Инструкционная карта №11** 69](#_Toc220948031)

[**Инструкционная карта №12** 78](#_Toc220948032)

[**Инструкционная карта №13** 82](#_Toc220948033)

[**Инструкционная карта №14** 92](#_Toc220948034)

[**Инструкционная карта №15** 97](#_Toc220948035)

[**Инструкционная карта №16** 105](#_Toc220948036)

[**Инструкционная карта №17** 113](#_Toc220948037)

[**Инструкционная карта №18** 123](#_Toc220948038)

[**Инструкционная карта №19** 131](#_Toc220948039)

[**Инструкционная карта №20** 138](#_Toc220948040)

[**Инструкционная карта №21** 140](#_Toc220948041)

[**Инструкционная карта №22** 153](#_Toc220948042)

[**ЗАКЛЮЧЕНИЕ** 154](#_Toc220948043)

**Инструкционная карта №1**

**Тема: «Знакомство с MS SQL Server».**

**Цель работы:** изучить интерфейс среды управления SQL Server Management Studio.

**Ход работы:**

1. Были установлены SQL Server Developer Edition и SQL Server Management Studio, после чего SSMS был запущен. В диалоговом окне Соединение с сервером были заданы определённые параметры.
2. Была создана новая учётная запись с именем GROUP431, результат работы представлен на рисунке 1.1.

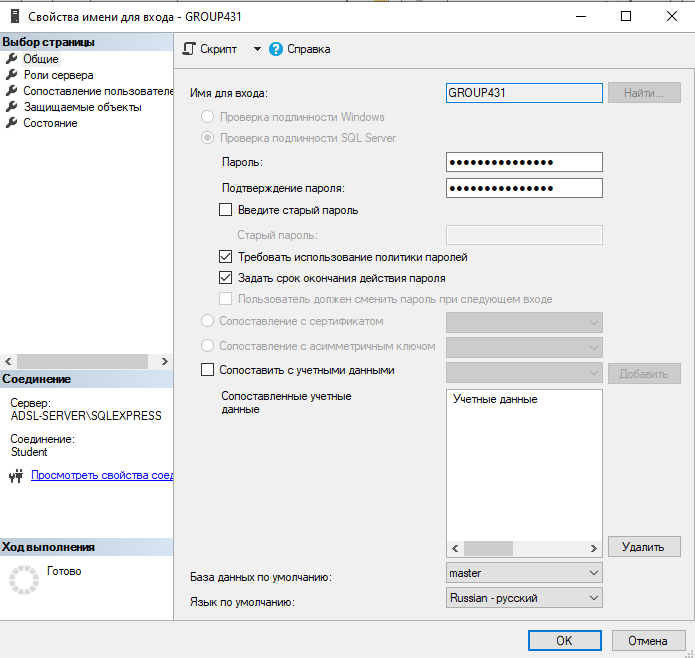


Рисунок 1.1 – Созданная учетная запись

1. Была создана База данных с именем «Учебная\_Рыбалев». В данную базу данных был добавлен новый пользователь «GROUP431» с правами пользователя. После чего была создана таблица с именем «TABL1». Результат работы представлен на рисунке 1.2.

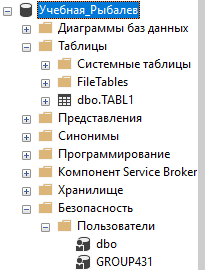


Рисунок 1.2 – Созданная БД, добавленный пользователь и созданная таблица

1. На диске С была создана папка «ARXIV\_db», после чего в созданную папку было добавлено резервное копирование с названием «PZ01-01».

**Рефлексия:** Я изучил интерфейс среды управления SQL Server Management Studio.

**Инструкционная карта №2**

**Тема: «Типы данных и встроенные функции».**

**Цель работы:**

1. Изучение основ выборки данных.
2. Изучение конструкции DISTINCT | ALL.
3. Изучение сортировки данных.
4. Изучение конструкции TOP.
5. Изучение конструкции OFFSET и FETCH.

**Ход работы:**

1. Была создана база данных «ПЗ02\_Рыбалев», в ней была создана таблица «Академики».
2. Таблица «Академики» была заполнена данными. Результат работы представлен на рисунке 2.1.

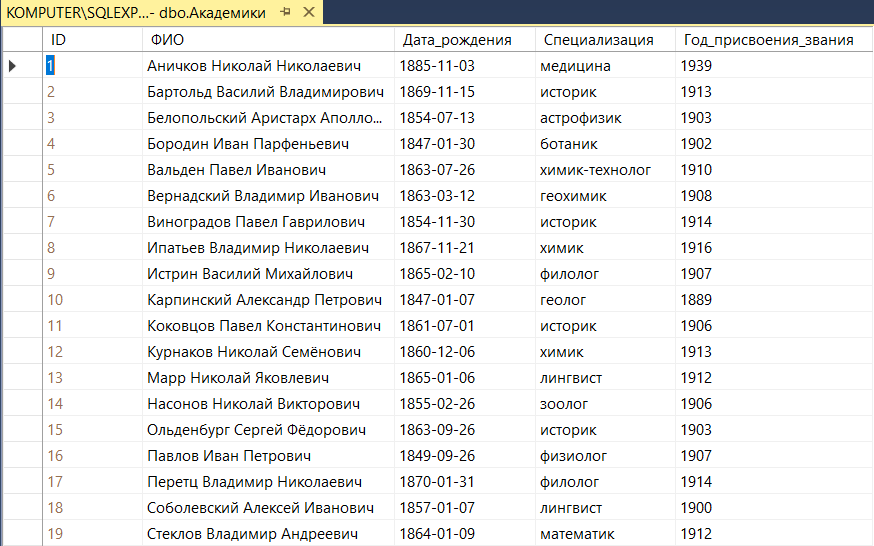


Рисунок 2.1 – Заполненная таблица

1. Были выполнены следующие примеры:

Пример 1. Вывести список академиков. Выполненный пример представлен на рисунке 2.2.

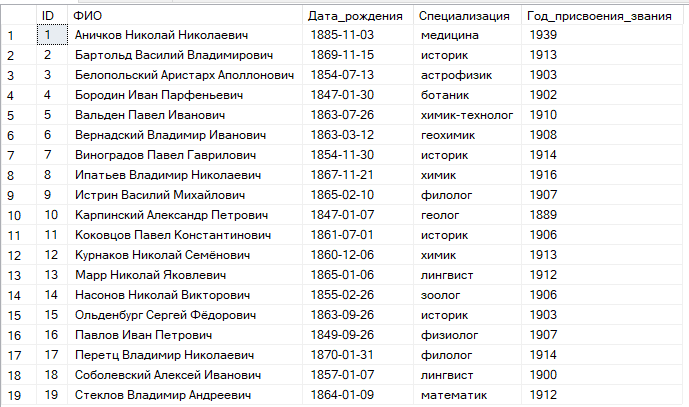


Рисунок 2.2 – Результат примера 1

Пример 2. Вывести список академиков. Выполненный пример представлен на рисунке 2.3.

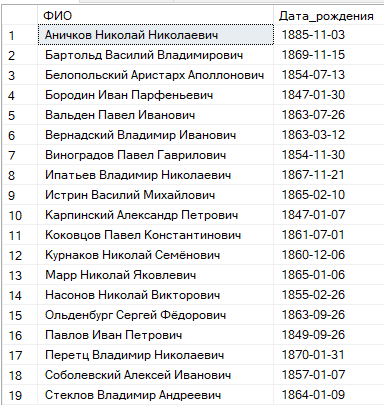


Рисунок 2.3 – Результат примера 2

Пример 3: Создайте вычисляемое поле «Информация», содержащее информацию об академиках в таком виде: «Академик Петров Петр Петрович, специализация: математика». Выполненный пример представлен на рисунке 2.4.

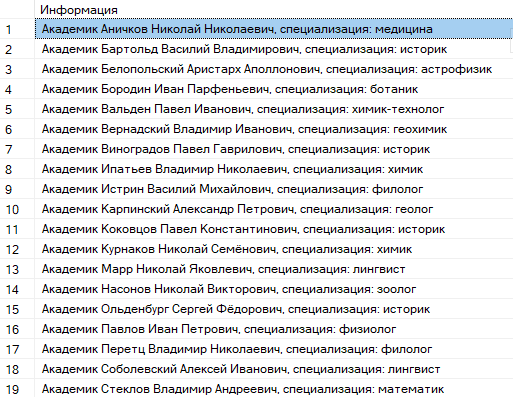


Рисунок 2.4 – Результат примера 3

Пример 4: Вывести ФИО академиков и номер следующего года после присвоения звания. Выполненный пример представлен на рисунке 2.5.



Рисунок 2.5 – Результат примера 4

Пример 5: Выведите список специализаций, убрав дубликаты. Выполненный пример представлен на рисунке 2.6.

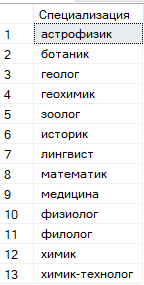


Рисунок 2.6 – Результат примера 5

Пример 6: вывести список академиков, отсортированный по возрастанию года присвоения звания. Выполненный пример представлен на рисунке 2.7.

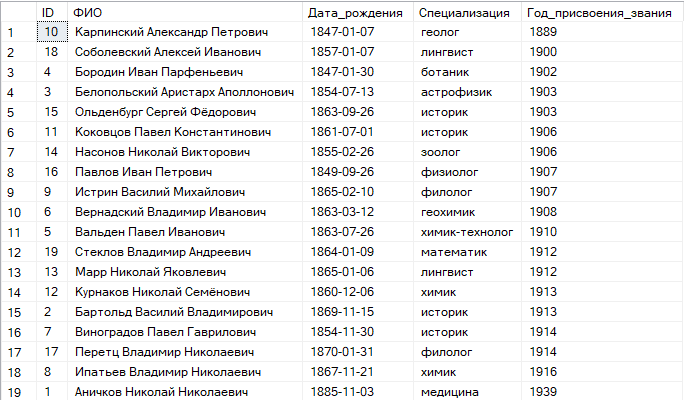


Рисунок 2.7 – Результат примера 6

Пример 7: вывести список академиков, отсортированный в обратном алфавитном порядке по полю «Специализация» и в алфавитном порядке по полю «ФИО». Выполненный пример представлен на рисунке 2.8.

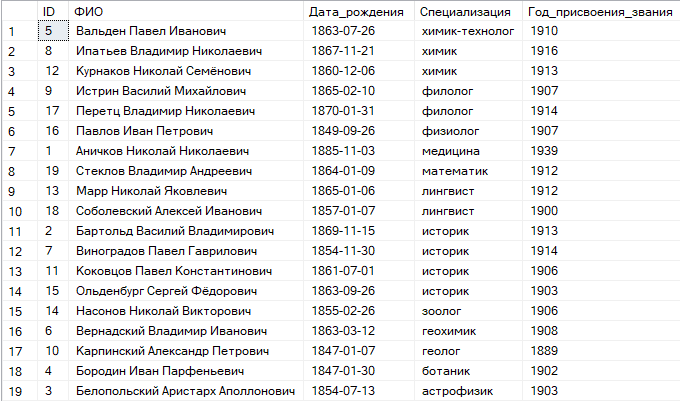


Рисунок 2.8 – Результат примера 7

Пример 8: вывести первые две строки из списка академиков, отсортированного в алфавитном порядке по полю «ФИО». Выполненный пример представлен на рисунке 2.9.



Рисунок 2.9 – Результат примера 8

Пример 9: вывести первые 30% строк из списка академиков, отсортированного по возрастанию года присвоения звания. Выполненный пример представлен на рисунке 2.10.



Рисунок 2.10 – Результат примера 9

Пример 10: вывести из таблицы «Академики», отсортированной по возрастанию года присвоения звания, список академиков, у которых год присвоения звания – один из первых четырех в отсортированной таблице. Выполненный пример представлен на рисунке 2.11.

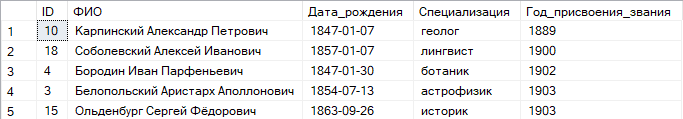


Рисунок 2.11 – Результат примера 10

Пример 11: вывести, начиная с третьего, список академиков, отсортированный в алфавитном порядке ФИО. Выполненный пример представлен на рисунке 2.12.

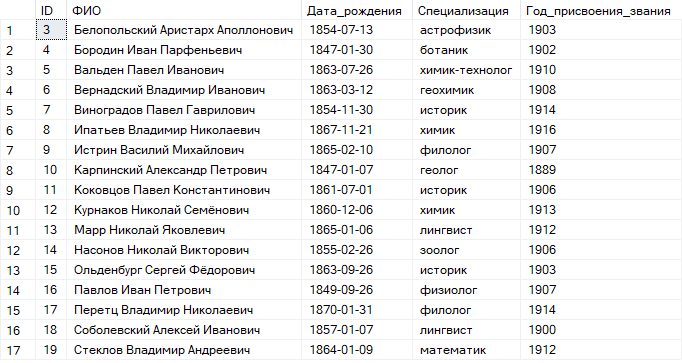


Рисунок 2.12 – Результат примера 11

Пример 12: вывести, начиная с третьего и до десятого, список академиков, отсортированный в алфавитном порядке ФИО. Выполненный пример представлен на рисунке 2.13.

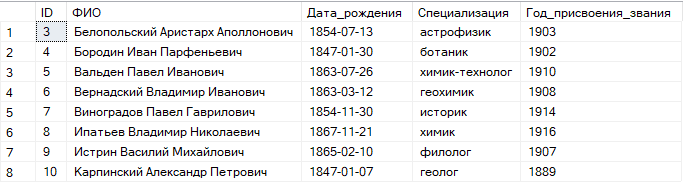


Рисунок 2.13 – Результат примера 12

**Самостоятельное задание**

1. Вывести ФИО, специализацию и дату рождения всех академиков.

Код выполнения задания представлен ниже:

SELECT ФИО, Специализация, Дата\_рождения

FROM Академики;

Выполненное задание представлено на рисунке 2.14.

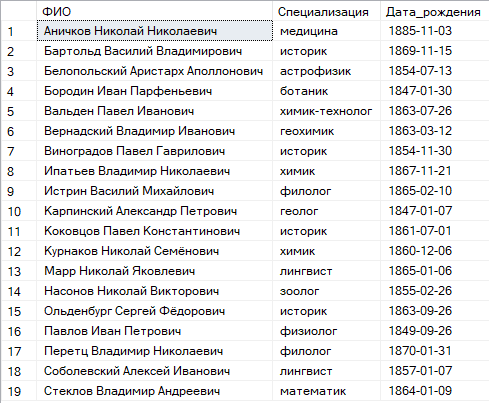


Рисунок 2.14 – Выполненное задание 1

1. Создать вычисляемое поле «О присвоении звания», которое содержит информацию об академиках в виде: «Петров Петр Петрович получил звание в 1974».

Код выполнения задания представлен ниже:

SELECT

CONCAT(ФИО, ' получил звание в ', Год\_присвоения\_звания) AS "О присвоении звания"

FROM Академики;

Выполненное задание представлено на рисунке 2.15.

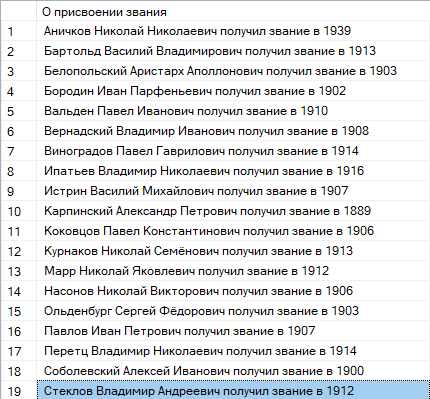


Рисунок 2.15 – Выполненное задание 2

1. Вывести ФИО академиков и через пробел вычисляемое поле «Через 5 лет после присвоения звания». Через 5 лет после присвоения звания; Карпинский Александр Петрович 1894; Бартольд Василий Владимирович 1918.

Код выполнения задания представлен ниже:

SELECT

CONCAT(ФИО, ' ', Год\_присвоения\_звания + 5) AS "Через 5 лет после присвоения звания"

FROM Академики;

Выполненное задание представлено на рисунке 2.16.

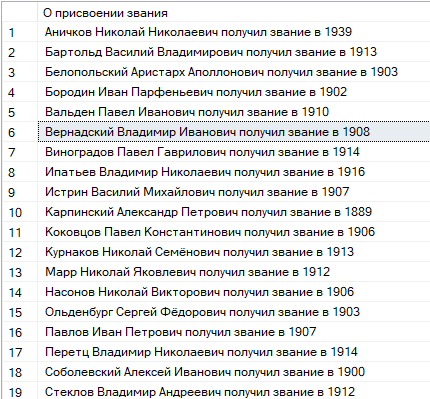


Рисунок 2.16 – Выполненное задание 3

1. Вывести список годов присвоения званий, убрав дубликаты.

Код выполнения задания представлен ниже:

SELECT DISTINCT Год\_присвоения\_звания

FROM Академики

ORDER BY Год\_присвоения\_звания;

Выполненное задание представлено на рисунке 2.17.

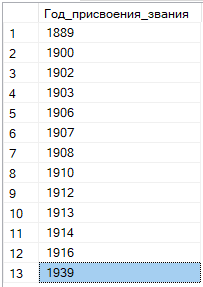


Рисунок 2.17 – Выполненное задание 4

1. Вывести список академиков, отсортированный по убыванию даты рождения.

Код выполнения задания представлен ниже:

SELECT \*

FROM Академики

ORDER BY Дата\_рождения DESC;

Выполненное задание представлено на рисунке 2.18.

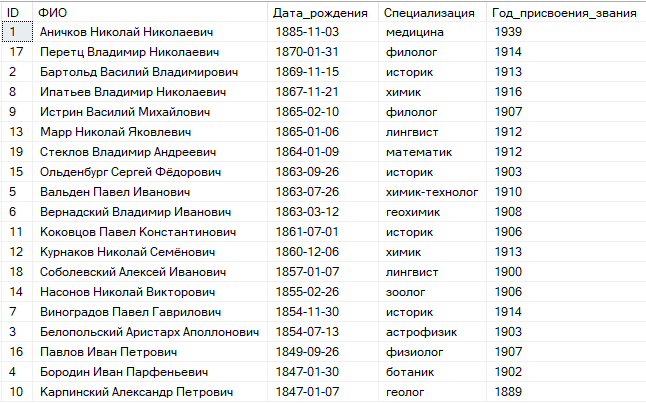


Рисунок 2.18 – Выполненное задание 5

1. Вывести список академиков, отсортированный в обратном алфавитном порядке специализаций, по убыванию года присвоения звания, и в алфавитном порядке ФИО.

Код выполнения задания представлен ниже:

SELECT \*

FROM Академики

ORDER BY

Специализация DESC,

Год\_присвоения\_звания DESC,

ФИО ASC;

Выполненное задание представлено на рисунке 2.19.



Рисунок 2.19 – Выполненное задание 6

1. Вывести первую строку из списка академиков, отсортированного в обратном алфавитном порядке ФИО.

Код выполнения задания представлен ниже:

SELECT TOP 1 \*

FROM Академики

ORDER BY ФИО DESC;

Выполненное задание представлено на рисунке 2.20.



Рисунок 2.20 – Выполненное задание 7

1. Вывести фамилию академика, который раньше всех получил звание.

Код выполнения задания представлен ниже:

SELECT TOP 1

ФИО,

Год\_присвоения\_звания

FROM Академики

ORDER BY Год\_присвоения\_звания ASC

Выполненное задание представлено на рисунке 2.21.



Рисунок 2.21 – Выполненное задание 8

1. Вывести первые 10% строк из списка академиков, отсортированного в алфавитном порядке ФИО.

Код выполнения задания представлен ниже:

SELECT TOP 10 PERCENT \*

FROM Академики

ORDER BY ФИО ASC;

Выполненное задание представлено на рисунке 2.22.

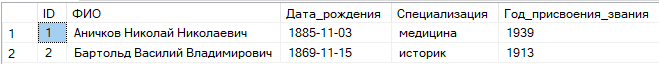


Рисунок 2.22 – Выполненное задание 9

1. Вывести из таблицы «Академики», отсортированной по возрастанию года присвоения звания, список академиков, у которых год присвоения звания – один из первых пяти в отсортированной таблице.

Код выполнения задания представлен ниже:

SELECT TOP 5 \*

FROM Academics

ORDER BY Год\_присвоения\_звания ASC;

Выполненное задание представлено на рисунке 2.23.

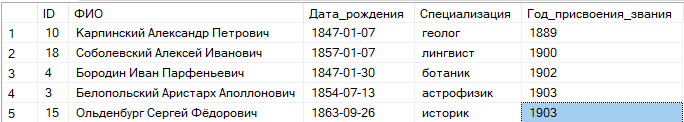


Рисунок 2.23 – Выполненное задание 10

1. Вывести, начиная с десятого, список академиков, отсортированный по возрастанию даты рождения.

Код выполнения задания представлен ниже:

SELECT \*

FROM Академики

ORDER BY Дата\_рождения ASC

OFFSET 9 ROWS;

Выполненное задание представлено на рисунке 2.24.



Рисунок 2.24 – Выполненное задание 11

1. Вывести девятую и десятую строку из списка академиков, отсортированного в алфавитном порядке ФИО.

Код выполнения задания представлен ниже:

SELECT \*

FROM Академики

ORDER BY ФИО ASC

OFFSET 8 ROWS

FETCH NEXT 2 ROWS ONLY;

Выполненное задание представлено на рисунке 2.25.

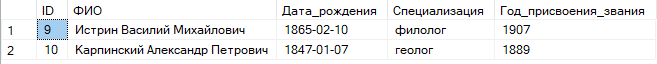


Рисунок 2.25 – Выполненное задание 12

**Рефлексия:** Яизучил основы выборки данных, конструкции DISTINCT | ALL, сортировки данных, конструкции TOP, конструкции OFFSET и FETCH.

**Инструкционная карта №3**

**Тема: «Фильтрация данных».**

**Цель работы:**

1. Изучение основ фильтрации данных.
2. Изучение операций сравнения.
3. Изучение логических операторов.
4. Изучение BETWEEN.
5. Изучение LIKE.
6. Изучение NULL.
7. Изучение IN.

**Ход работы:**

1. Была создана база данных «ПЗ03\_Рыбалев», в ней была создана таблица «Страны» и заполнена данными. Заполненная таблица «Страны» показана на рисунке 3.1.

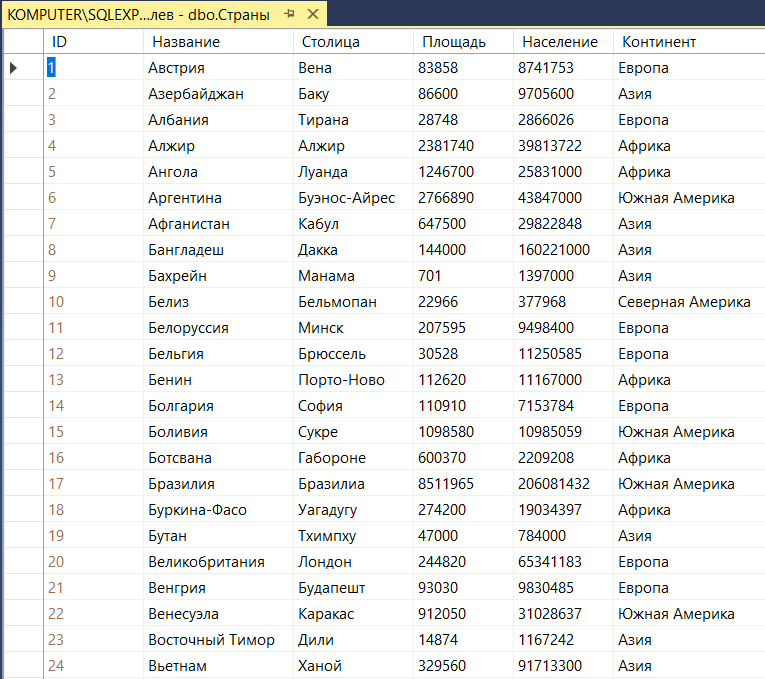


Рисунок 3.1 – Заполненная таблица «Страны»

1. Были выполнены следующие примеры:

Пример 1: вывести список стран, площадь которых больше 1 млн. кв. км. Выполнение примера представлен на рисунке 3.2.

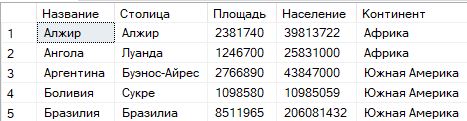


Рисунок 3.2 – Выполнение примера 1

Пример 2: вывести список стран, население которых не больше 1 млн. чел. Выполнение примера представлен на рисунке 3.3.

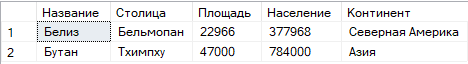


Рисунок 3.3 – Выполнение примера 2

Пример 3: вывести список африканских стран. Выполнение примера представлен на рисунке 3.4.



Рисунок 3.4 – Выполнение примера 3

Пример 4: вывести список всех стран, кроме европейских. Выполнение примера представлен на рисунке 3.5.

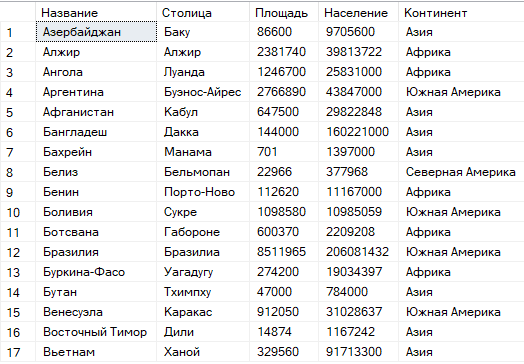


Рисунок 3.5 – Выполнение примера 4

Пример 5: вывести список стран, население которых больше 1 млн. чел., а площадь меньше 100 тыс. кв. км. Выполнение примера представлен на рисунке 3.6.



Рисунок 3.6 – Выполнение примера 5

Пример 6: вывести список стран, которые находятся в Европе и их население больше 10 млн. чел., или находятся в Азии, а население больше 50 млн. чел. Выполнение примера представлен на рисунке 3.7.

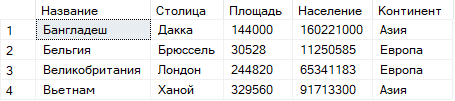


Рисунок 3.7 – Выполнение примера 6

Пример 7: вывести список стран, население которых от 10 до 100 млн. чел., а площадь от 100 до 200 тыс. кв. км. Выполнение примера представлен на рисунке 3.8.



Рисунок 3.8 – Выполнение примера 7

Пример 8: вывести отсортированный в алфавитном порядке список стран от Бенина до Ватикана. Выполнение примера представлен на рисунке 3.9.



Рисунок 3.9 – Выполнение примера 8

Пример 9: вывести список стран, название которых начинается с буквы «С». Выполнение примера представлен на рисунке 3.10.

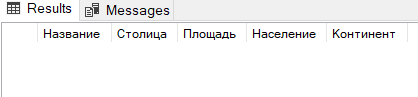


Рисунок 3.10 – Выполнение примера 9

Пример 10: вывести список стран, в названии которых вторая буква – «а», а последняя – я». Выполнение примера представлен на рисунке 3.11.



Рисунок 3.11 – Выполнение примера 10

Пример 11: вывести список стран, в названии которых третья буква – «а», «о» или «у». Выполнение примера представлен на рисунке 3.12.



Рисунок 3.12 – Выполнение примера 11

Пример 12: вывести список стран, название которых начинается с буквы от «А» до «Г». Выполнение примера представлен на рисунке 3.13.

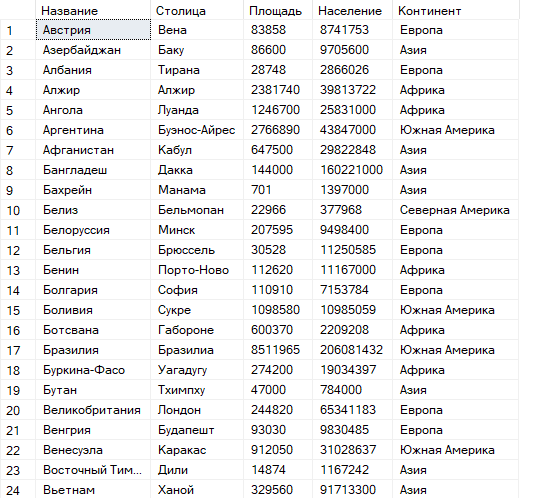


Рисунок 3.13 – Выполнение примера 12

Пример 13: вывести список стран, название которых не начинается с буквы от «А» до «Г» или с буквы «С». Выполнение примера представлен на рисунке 3.14.



Рисунок 3.14 – Выполнение примера 13

Пример 14: вывести список стран, столицы которых не введены в базу. Выполнение примера представлен на рисунке 3.15.



Рисунок 3.15 – Выполнение примера 14

Пример 15: вывести список европейских, азиатских и африканских стран. Выполнение примера представлен на рисунке 3.16.

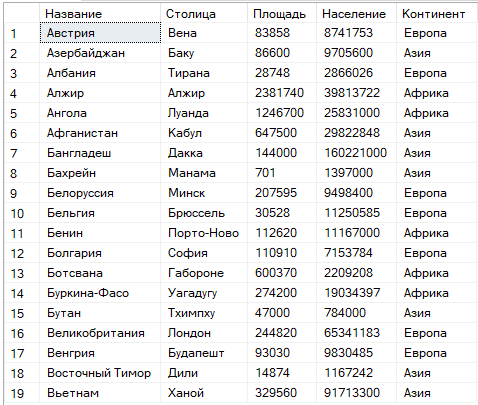


Рисунок 3.16 – Выполнение примера 15

**Самостоятельное задание**

Были выполнены следующие задания:

1. Вывести названия и столицы пяти наибольших стран по площади.

Код выполнения задания представлен ниже:

SELECT TOP 5 Название, Столица, Площадь

FROM Страны

ORDER BY Площадь DESC;

Выполненное задание представлено на рисунке 3.17.

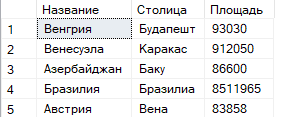


Рисунок 3.17 – Выполненное задание

1. Вывести список африканских стран, население которых не превышает 1 млн. чел.

Код выполнения задания представлен ниже:

SELECT Название, Столица, Население

FROM Страны

WHERE Континент = 'Африка' AND Население <= 1000000;

Выполненное задание представлено на рисунке 3.18.



Рисунок 3.18 – Выполненное задание

1. Вывести список стран, население которых больше 5 млн. чел., а площадь меньше 100 тыс. кв. км, и они расположены не в Европе.

Код выполнения задания представлен ниже:

SELECT Название, Столица, Площадь, Население, Континент

FROM Страны

WHERE Население > 5000000

AND Площадь < 100000

AND Континент != 'Европа';

Выполненное задание представлено на рисунке 3.19.



Рисунок 3.19 – Выполненное задание

1. Вывести список стран Северной и Южной Америки, население которых больше 20 млн. чел., или стран Африки, у которых население больше 30 млн. чел.

Код выполнения задания представлен ниже:

SELECT Название, Столица, Континент, Население

FROM Страны

WHERE (Континент IN ('Северная Америка', 'Южная Америка') AND Население > 20000000)

OR (Континент = 'Африка' AND Население > 30000000);

Выполненное задание представлено на рисунке 3.20.

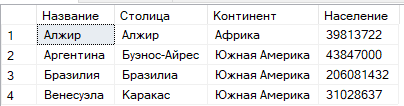


Рисунок 3.20 – Выполненное задание

1. Вывести список стран, население которых составляет от 10 до 100 млн. чел., а площадь не больше 500 тыс. кв. км.

Код выполнения задания представлен ниже:

SELECT Название, Столица, Площадь, Население

FROM Страны

WHERE Население BETWEEN 10000000 AND 100000000

AND Площадь <= 500000;

Выполненное задание представлено на рисунке 3.21.



Рисунок 3.21 – Выполненное задание

1. Вывести список стран, названия которых не начинаются с буквы «К».

Код выполнения задания представлен ниже:

SELECT Название, Столица

FROM Страны

WHERE Название NOT LIKE 'К%';

Выполненное задание представлено на рисунке 3.22.

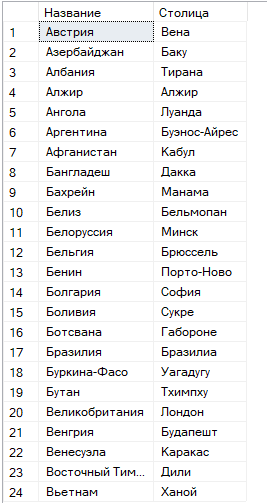


Рисунок 3.22 – Выполненное задание

1. Вывести список стран, в названии которых третья буква – «а», а предпоследняя – «и».

Код выполнения задания представлен ниже:

SELECT Название, Столица

FROM Страны

WHERE SUBSTRING(Название, 3, 1) = 'а'

AND SUBSTRING(Название, LEN(Название)-1, 1) = 'и';

Выполненное задание представлено на рисунке 3.23.



Рисунок 3.23 – Выполненное задание

1. Вывести список стран, в названии которых вторая буква – гласная.

Код выполнения задания представлен ниже:

SELECT Название, Столица

FROM Страны

WHERE LOWER(SUBSTRING(Название, 2, 1)) IN ('а', 'е', 'ё', 'и', 'о', 'у', 'ы', 'э', 'ю', 'я');

Выполненное задание представлено на рисунке 3.24.



Рисунок 3.24 – Выполненное задание

1. Вывести список стран, названия столиц которых начинаются с букв от «К» до «П» отсортированный в алфавитном порядке по полю Столица

Код выполнения задания представлен ниже:

SELECT Название, Столица

FROM Страны

WHERE Столица >= 'К' AND Столица < 'Р'

ORDER BY Столица;

Выполненное задание представлено на рисунке 3.25.

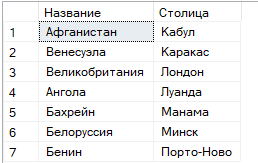


Рисунок 3.25 – Выполненное задание

1. Вывести список стран, названия которых начинаются с букв от «А» до «Г», но не с буквы «Б».

Код выполнения задания представлен ниже:

SELECT Название, Столица

FROM Страны

WHERE Название >= 'А' AND Название < 'Д'

AND Название NOT LIKE 'Б%';

Выполненное задание представлено на рисунке 3.26.

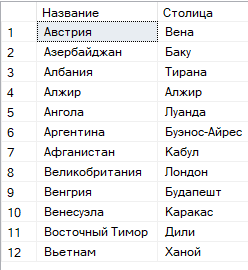


Рисунок 3.26 – Выполненное задание

1. Вывести список стран, столицы которых есть в базе.

Код выполнения задания представлен ниже:

SELECT DISTINCT s1.Название, s1.Столица

FROM Страны s1

WHERE s1.Столица IN (SELECT s2.Название FROM Страны s2);

Выполненное задание представлено на рисунке 3.27.



Рисунок 3.27 – Выполненное задание

1. Вывести список стран Африки, Северной и Южной Америки.

Код выполнения задания представлен ниже:

SELECT Название, Столица, Континент

FROM Страны

WHERE Континент IN ('Африка', 'Северная Америка', 'Южная Америка')

ORDER BY Континент, Название;

Выполненное задание представлено на рисунке 3.28.

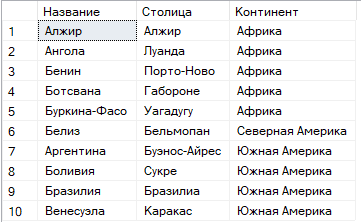


Рисунок 3.28 – Выполненное задание

**Рефлексия:** Я изучил: основы фильтрации данных; операции сравнения; логические операторы; BETWEEN; LIKE; NULL; IN.

**Инструкционная карта №4**

**Тема: «Типы данных и встроенные функции».**

**Цель работы:**

1. Изучить основные типы данных.
2. Изучить встроенные функции для работы со строками.
3. Изучить встроенные функции для работы с числами.
4. Изучить встроенные функции для работы с датами и временем.
5. Изучить встроенные функции преобразования данных.
6. Изучить CASE и IIF.

**Ход работы:**

1. Была создана база данных «ПЗ04\_Рыбалев», в ней была создана таблица «Академики», которая была заполнена данными. Заполненная таблица «Академики» представлена на рисунке 4.1.

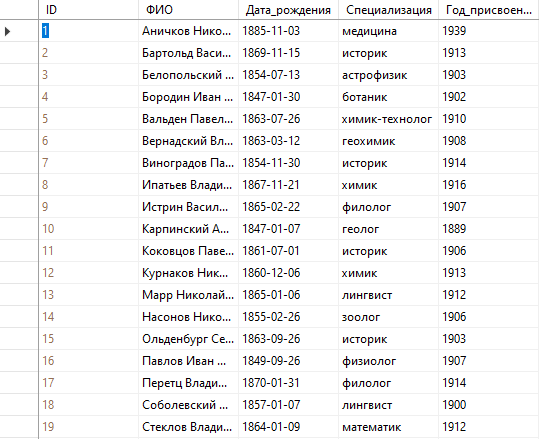


Рисунок 4.1 – Заполненная таблица «Академики»

1. Были выполнены следующие примеры:

Пример 1: Вывести ФИО академиков и длину ФИО. Выполнение примера представлено на рисунке 4.2.

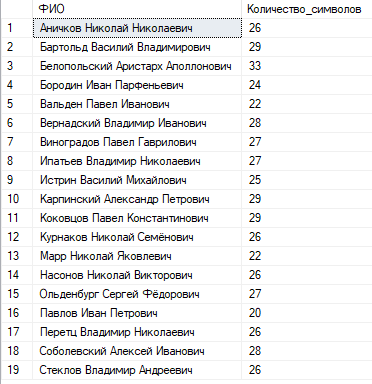


Рисунок 4.2 – Выполнение примера 1

Пример 2: вывести список академиков, убрать лишние пробелы в ФИО. Выполнение примера представлено на рисунке 4.3.

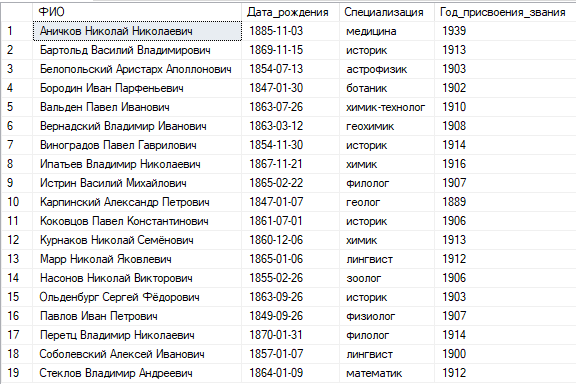


Рисунок 4.3 – Выполнение примера 2

Пример 3: найти позиции буквы «о» в ФИО каждого академика. Вывести ФИО и позицию. Выполнение примера представлено на рисунке 4.4.



Рисунок 4.4 – Выполнение примера 3

Пример 4: Вывести ФИО и первые три буквы специализации каждого академика. Выполнение примера представлено на рисунке 4.5.



Рисунок 4.5 – Выполнение примера 4

Пример 5: Вывести ФИО и от второй до пятой буквы специализации каждого академика. Выполнение примера представлено на рисунке 4.6.

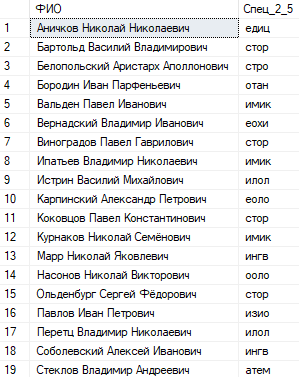


Рисунок 4.6 – Выполнение примера 5

Пример 6: вывести список академиков, заменить специализацию «лингвист» на «языковед». Выполнение примера представлено на рисунке 4.7.

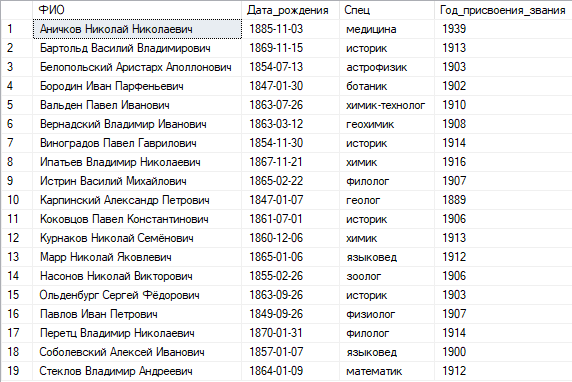


Рисунок 4.7 – Выполнение примера 6

Пример 7: вывести список академиков, специализацию на верхнем регистре. Выполнение примера представлено на рисунке 4.8.

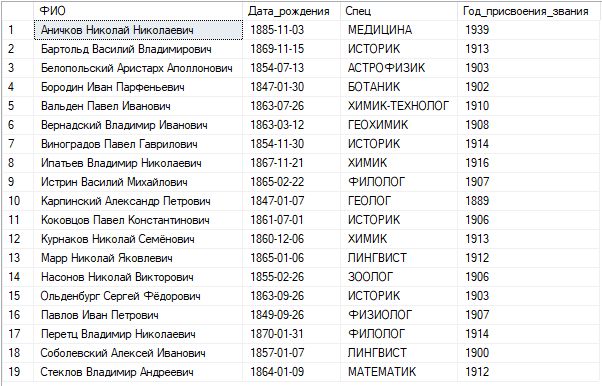


Рисунок 4.8 – Выполнение примера 7

Пример 8: Вывести ФИО академиков в правильном и обратном виде. Выполнение примера представлено на рисунке 4.9.

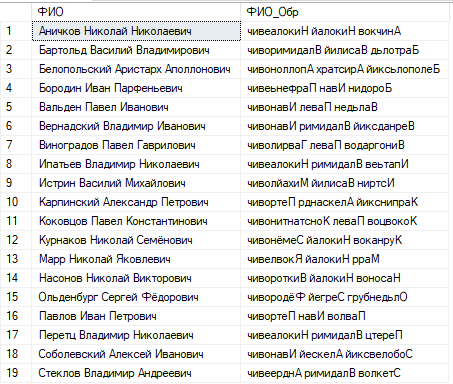


Рисунок 4.9 – Выполнение примера 8

Пример 9: вывести каждую специализацию 4 раза в одной строке. Убрать дубликаты. Выполнение примера представлено на рисунке 4.10.

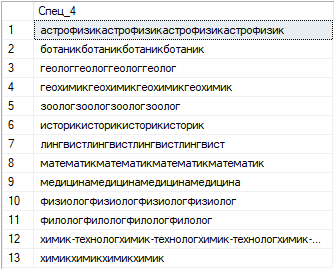


Рисунок 4.10 – Выполнение примера 9

Пример 10: вывести абсолютное значение тригонометрических функций на точке π. Выполнение примера представлено на рисунке 4.11.



Рисунок 4.11 – Выполнение примера 10

Пример 11: вывести число 132.456, округленное с точностью от 3 до -3. Выполнение примера представлено на рисунке 4.12.



Рисунок 4.12 – Выполнение примера 11

Пример 12: вывести наименьшее целое число, которое больше или равно 123.456, и наибольшее целое число, которое меньше или равно 123.456. Выполнение примера представлено на рисунке 4.13.



Рисунок 4.13 – Выполнение примера 12

Пример 13: вывести квадратный корень, квадрат и куб числа 25. Выполнение примера представлено на рисунке 4.14.



Рисунок 4.14 – Выполнение примера 13

Пример 14: вывести день, месяц, год, час, минуту, секунду, номер квартала, номер недели, день года, день недели для текущей даты и времени. Выполнение примера представлено на рисунке 4.15.



Рисунок 4.15 – Выполнение примера 14

Пример 15: Академик Игорь Евгеньевич Тамм родился 8 июля 1895 года. И.Е. Тамм скончался 12 апреля 1971 года. Вывести количество прожитых дней. Выполнение примера представлено на рисунке 4.16.



Рисунок 4.16 – Выполнение примера 15

Пример 16: Вывести ФИО и время года рождения каждого академика. Выполнение примера представлено на рисунке 4.17.

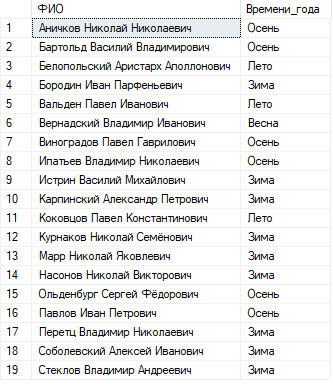


Рисунок 4.17 – Выполнение примера 16

Пример 17: вывести список академиков. Для каждого академика, в зависимости от возраста, при присвоении звания вывести «молодой» или «старый» в дополнительном столбце. Выполнение примера представлено на рисунке 4.18.



Рисунок 4.18 – Выполнение примера 17

**Самостоятельное задание**

Были выполнены следующие задания:

1. Вывести список академиков, отсортированный по количеству символов в ФИО.

Код выполнения задания представлен ниже:

SELECT ФИО, LEN(ФИО) as Длина\_ФИО

FROM Академики

ORDER BY LEN(ФИО);

Выполненное задание представлено на рисунке 4.19.

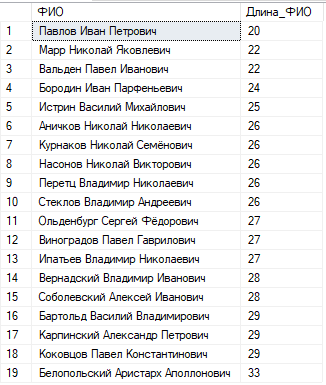


Рисунок 4.19– Выполненное задание

1. Вывести список академиков, убрать лишние пробелы в ФИО.

Код выполнения задания представлен ниже:

SELECT LTRIM(RTRIM(ФИО)) as ФИО\_без\_пробелов,

ФИО as Оригинал

FROM Академики;

Выполненное задание представлено на рисунке 4.20.

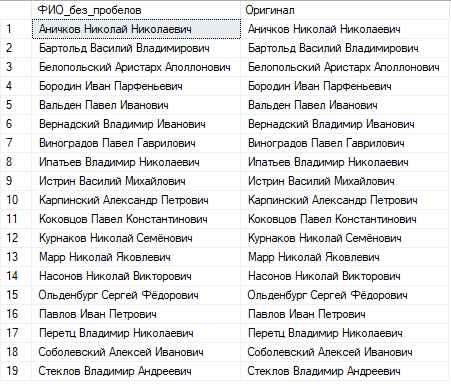


Рисунок 4.20 – Выполненное задание

1. Найти позиции «ов» в ФИО каждого академика. Вывести ФИО и номер позиции.

Код выполнения задания представлен ниже:

SELECT ФИО,

CHARINDEX('ов', ФИО) as Позиция\_ов

FROM Академики

WHERE CHARINDEX('ов', ФИО) > 0;

Выполненное задание представлено на рисунке 4.21.

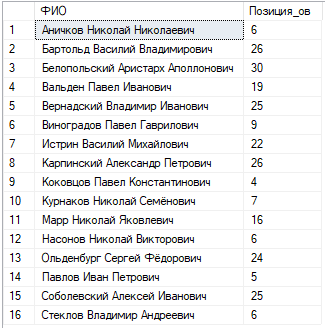


Рисунок 4.21 – Выполненное задание

1. Вывести ФИО и последние две буквы специализации для каждого академика.

Код выполнения задания представлен ниже:

SELECT ФИО,

Специализация,

RIGHT(Специализация, 2) as Последние\_две\_буквы

FROM Академики;

Выполненное задание представлено на рисунке 4.22.

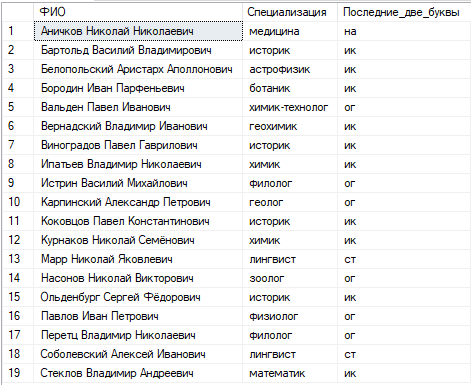


Рисунок 4.22 – Выполненное задание

1. Вывести список академиков, ФИО в формате Фамилия и Инициалы.

Код выполнения задания представлен ниже:

SELECT

CONCAT(

LEFT(ФИО, CHARINDEX(' ', ФИО) - 1), ' ', -- Фамилия

LEFT(SUBSTRING(ФИО, CHARINDEX(' ', ФИО) + 1, LEN(ФИО)), 1), '. ', -- Инициал имени

LEFT(REVERSE(LEFT(REVERSE(ФИО), CHARINDEX(' ', REVERSE(ФИО)) - 1)), 1), '.' -- Инициал отчества

) as Фамилия\_Инициалы,

ФИО as Полное\_ФИО

FROM Академики;

Выполненное задание представлено на рисунке 4.23.



Рисунок 4.23 – Выполненное задание

1. Вывести список специализаций в правильном и обратном виде. Убрать дубликаты.

Код выполнения задания представлен ниже:

SELECT DISTINCT

Специализация,

REVERSE(Специализация) as Обратная\_специализация

FROM Академики

ORDER BY Специализация;

Выполненное задание представлено на рисунке 4.24.

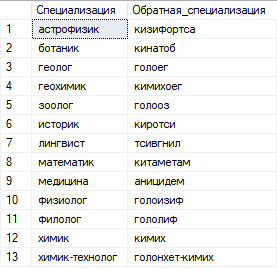


Рисунок 4.24 – Выполненное задание

1. Вывести свою фамилию в одной строке столько раз, сколько вам лет.

Код выполнения задания представлен ниже:

DECLARE @Фамилия NVARCHAR(50) = 'Рыбалев';

DECLARE @Возраст INT = 19;

SELECT REPLICATE(@Фамилия, @Возраст) as Фамилия\_повтор;

Выполненное задание представлено на рисунке 4.25.



Рисунок 4.25 – Выполненное задание

1. Вывести количество дней до конца семестра.

Код выполнения задания представлен ниже:

DECLARE @КонецСеместра DATE = '2024-12-31';

SELECT DATEDIFF(day, GETDATE(), @КонецСеместра) as Дней\_до\_конца\_семестра;

Выполненное задание представлено на рисунке 4.26.



Рисунок 4.26 – Выполненное задание

1. Вывести количество месяцев от дня рождения А.С. Пушкина

Код выполнения задания представлен ниже:

SELECT DATEDIFF(month, '1799-06-06', GETDATE()) as Месяцев\_с\_рождения\_Пушкина;

Выполненное задание представлено на рисунке 4.27.



Рисунок 4.27 – Выполненное задание

1. Вывести ФИО и название дня недели даты рождения каждого академика.

Код выполнения задания представлен ниже:

SELECT ФИО,

Дата\_рождения,

DATENAME(weekday, Дата\_рождения) as День\_недели\_рождения

FROM Академики;

Выполненное задание представлено на рисунке 4.28.

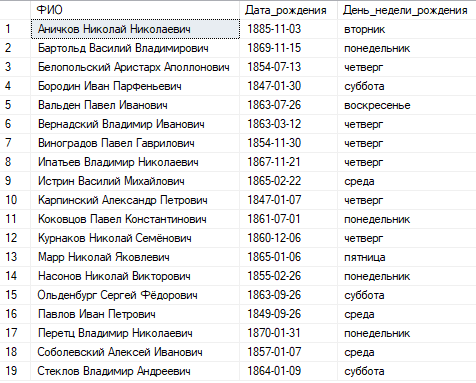


Рисунок 4.28 – Выполненное задание

1. Вывести список специализаций без повторений. Для каждой специализации вывести «длинный» или «короткий», в зависимости от количества символов. (>6 длинный, <=короткий).

Код выполнения задания представлен ниже:

SELECT DISTINCT

Специализация,

CASE

WHEN LEN(Специализация) > 6 THEN 'длинный'

ELSE 'короткий'

END as Длина\_специализации,

LEN(Специализация) as Количество\_символов

FROM Академики

ORDER BY Специализация;

Выполненное задание представлено на рисунке 4.29.



Рисунок 4.29 – Выполненное задание

**Рефлексия:** Я изучил: основные типы данных; встроенные функции для работы со строками; встроенные функции для работы с числами; встроенные функции для работы с датами и временем; встроенные функции преобразования данных; CASE и IIF.

**Инструкционная карта №5**

**Тема: «Агрегатные функции».**

**Цель работы:**

1. Изучить основы агрегации данных.
2. Изучить функцию MAX.
3. Изучить функцию MIN.
4. Изучить функцию SUM.
5. Изучить функцию AVG.
6. Изучить функцию COUNT.
7. Изучить группировки данных.
8. Изучить применение фильтрации в группировке данных.

**Ход работы:**

1. Была создана база данных «ПЗ05\_Рыбалев», в которой была создана таблица «Страны» и заполнена данными. Заполненная таблица «Страны» представлена на рисунке 5.1.



Рисунок 5.1 – Заполненная таблица «Страны»

1. Были выполнены следующие примеры:

Пример 1: вывести максимальную площадь стран. Выполнение примера представлено на рисунке 5.2.



Рисунок 5.2 – Выполнение примера 1

Пример 2: вывести наименьшее население стран в Африке. Выполнение примера представлено на рисунке 5.3.



Рисунок 5.3 – Выполнение примера 2

Пример 3: вывести суммарное население стран Северной и Южной Америки. Выполнение примера представлено на рисунке 5.4.



Рисунок 5.4 – Выполнение примера 3

1. Был выполнен обучающий тест. Результат тестирования представлен на рисунке 5.5.

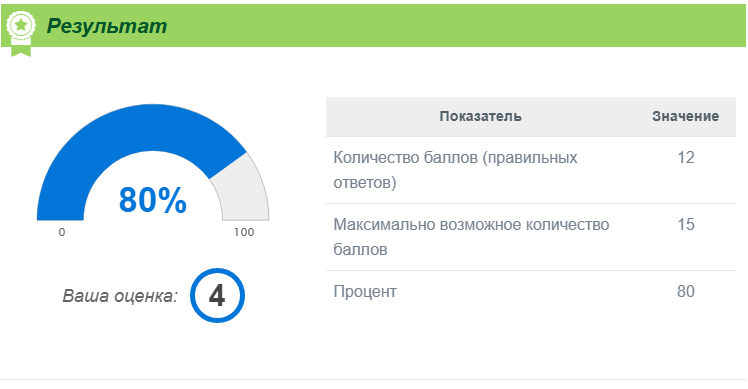


Рисунок 5.5 – Результат тестирования

**Самостоятельное задание**

Были выполнены следующие задания:

1. Вывести минимальную площадь стран.

Код выполнения задания представлен ниже:

SELECT MIN(Площадь) AS Мин\_Площадь

FROM Страны;

Выполненное задание представлено на рисунке 5.6.



Рисунок 5.6– Выполненное задание

1. Вывести наибольшую по населению страну в Северной и Южной Америке.

Код выполнения задания представлен ниже:

SELECT TOP 1 Название, Население

FROM Страны

WHERE Континент IN ('Северная Америка', 'Южная Америка')

ORDER BY Население DESC;

Выполненное задание представлено на рисунке 5.7.



Рисунок 5.7 – Выполненное задание

1. Вывести среднее население стран. Результат округлить до одного знака.

Код выполнения задания представлен ниже:

SELECT ROUND(AVG(CAST(Население AS FLOAT)), 1) AS Среднее\_Население

FROM Страны;

Выполненное задание представлено на рисунке 5.8.



Рисунок 5.8 – Выполненное задание

1. Вывести количество стран, у которых название заканчивается на «ан», кроме стран, у которых название заканчивается на «стан».

Код выполнения задания представлен ниже:

SELECT COUNT(\*) AS Количество

FROM Страны

WHERE Название LIKE '%ан'

AND Название NOT LIKE '%стан';

Выполненное задание представлено на рисунке 5.9.



Рисунок 5.9 – Выполненное задание

1. вести количество континентов, где есть страны, название которых начинается с буквы «Р».

Код выполнения задания представлен ниже:

SELECT COUNT(DISTINCT Континент) AS Количество\_Континентов

FROM Страны

WHERE LEFT(Название, 1) = 'Р';

Выполненное задание представлено на рисунке 5.10.



Рисунок 5.10 – Выполненное задание

1. Сколько раз страна с наибольшей площадью больше, чем страна с наименьшей площадью?

Код выполнения задания представлен ниже:

SELECT

MAX(Площадь) / MIN(Площадь) AS Во\_Сколько\_Раз\_Больше

FROM Страны;

Выполненное задание представлено на рисунке 5.11.



Рисунок 5.11 – Выполненное задание

1. Вывести количество стран с населением больше, чем 100 млн. чел. на каждом континенте. Результат отсортировать по количеству стран по возрастанию.

Код выполнения задания представлен ниже:

SELECT

Континент,

COUNT(\*) AS Количество\_Стран

FROM Страны

WHERE Население > 100000000

GROUP BY Континент

ORDER BY Количество\_Стран ASC;

Выполненное задание представлено на рисунке 5.12.

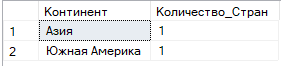


Рисунок 5.12– Выполненное задание

1. Вывести количество стран по количеству букв в названии. Результат отсортировать по убыванию.

Код выполнения задания представлен ниже:

SELECT

LEN(Название) AS Количество\_Букв,

COUNT(\*) AS Количество\_Стран

FROM Страны

GROUP BY LEN(Название)

ORDER BY Количество\_Стран DESC;

Выполненное задание представлено на рисунке 5.13.

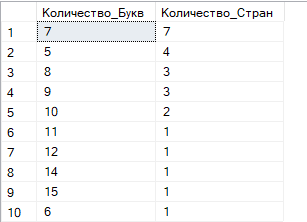


Рисунок 5.13 – Выполненное задание

1. Ожидается, что через 20 лет население мира вырастет на 10%. Вывести список континентов с прогнозируемым населением.

Код выполнения задания представлен ниже:

SELECT

Континент,

SUM(Население \* 1.1) AS Прогнозируемое\_Население

FROM Страны

GROUP BY Континент;

Выполненное задание представлено на рисунке 5.14.

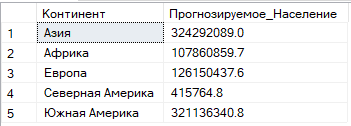


Рисунок 5.14 – Выполненное задание

1. Вывести список континентов, где разница по площади между наибольшими и наименьшими странами не более в 10000 раз.

Код выполнения задания представлен ниже:

SELECT

Континент

FROM Страны

GROUP BY Континент

HAVING MAX(Площадь) <= 10000 \* MIN(Площадь);

Выполненное задание представлено на рисунке 5.15.

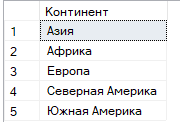


Рисунок 5.15 – Выполненное задание

1. Вывести среднюю длину названий Африканских стран.

Код выполнения задания представлен ниже:

SELECT

AVG(CAST(LEN(Название) AS FLOAT)) AS Средняя\_Длина\_Названия

FROM Страны

WHERE Континент = 'Африка';

Выполненное задание представлено на рисунке 5.16.



Рисунок 5.16 – Выполненное задание

1. Вывести список континентов, у которых средняя плотность среди стран с населением более 1 млн. чел. больше, чем 30 чел. на кв. км.

Код выполнения задания представлен ниже:

SELECT

Континент,

AVG(CAST(Население AS FLOAT) / Площадь) AS Средняя\_Плотность

FROM Страны

WHERE Население > 1000000

GROUP BY Континент

HAVING AVG(CAST(Население AS FLOAT) / Площадь) > 30;

Выполненное задание представлено на рисунке 5.17.

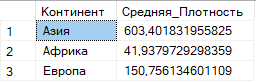


Рисунок 5.17 – Выполненное задание

**Рефлексия:** Я изучил: основы агрегации данных; функция MAX; функция MIN; функция SUM; функция AVG; функция COUNT; группировки данных; применение фильтрации в группировке данных.

**Инструкционная карта №6**

**Тема: «Соединение таблиц».**

**Цель работы:**

1. Изучить неявные соединения таблиц.
2. Изучить явные соединения таблиц.
3. Изучить внутреннее соединение таблиц.
4. Изучить внешнее соединение таблиц.
5. Изучить соединения таблиц со своей копией.

**Ход работы:**

1. Была создана база данных «ПЗ06\_Рыбалев», в ней были созданы следующие таблицы: Факультет, Кафедра, Сотрудники, Специальность, Дисциплина, Заявка, Зав\_кафедрой, Инженер, Преподаватель, Студент, Экзамен. Созданные таблицы представлены на рисунке 6.1.

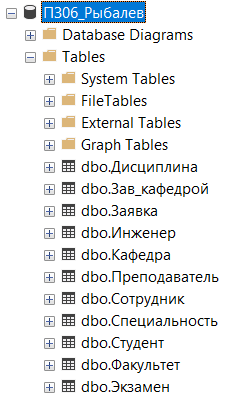


Рисунок 6.1 – Созданные таблицы

1. Все таблицы были заполнены соответствующими данными.
2. Были выполнены следующие примеры:

Пример 1: выбрать факультет и кафедры, используя неявное соединение. Результат отсортировать по алфавиту. Выполнение примера представлено на рисунке 6.2.

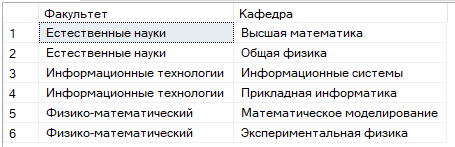


Рисунок 6.2 – Выполнение примера 1

Пример 2: выбрать факультет и кафедры, используя явное соединение. Результат отсортировать по алфавиту. Выполнение примера представлено на рисунке 6.3.



Рисунок 6.3 – Выполнение примера 2

Пример 3: выбрать все факультеты и их кафедры, если существуют. Результат отсортировать по алфавиту. Выполнение примера представлено на рисунке 6.4.

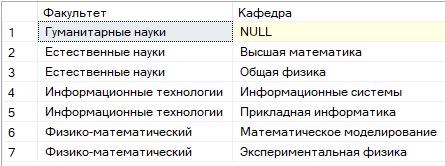


Рисунок 6.4 – Выполнение примера 3

**Самостоятельное задание**

Были выполнены следующие задания:

1. Вывести из таблиц «Кафедра», «Специальность» и «Студент» данные о студентах, которые обучаются на данном факультете.

Код выполнения задания представлен ниже:

SELECT

s.Рег\_номер,

s.Фамилия AS Фамилия\_студента,

sp.Номер AS Номер\_специальности,

sp.Направление,

k.Название AS Кафедра,

f.Название AS Факультет

FROM Студент s

INNER JOIN Специальность sp ON s.Номер = sp.Номер

INNER JOIN Кафедра k ON sp.Шифр = k.Шифр

INNER JOIN Факультет f ON k.Факультет = f.Аббревиатура

WHERE f.Аббревиатура = 'Ит'

ORDER BY s.Фамилия;

Выполненное задание представлено на рисунке 6.5.

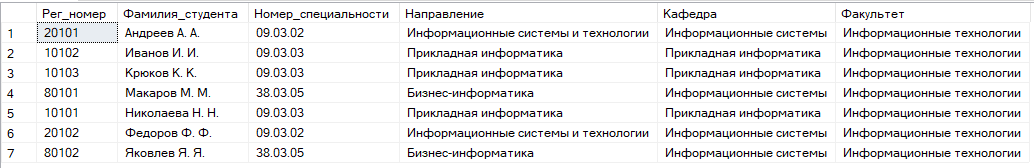


Рисунок 6.5– Выполненное задание

1. Вывести из таблиц «Кафедра», «Специальность» и «Сотрудник» данные о выпускающих кафедрах (факультет, шифр, название, фамилию заведующего). Выпускающей считается та кафедра, на которую есть ссылки в таблице «Специальность».

Код выполнения задания представлен ниже:

SELECT DISTINCT

f.Название AS Факультет,

k.Шифр,

k.Название AS Кафедра,

s.Фамилия AS Заведующий

FROM Кафедра k

INNER JOIN Факультет f ON k.Факультет = f.Аббревиатура

INNER JOIN Специальность sp ON k.Шифр = sp.Шифр

INNER JOIN Сотрудник s ON k.Шифр = s.Шифр AND s.Должность = 'зав. кафедрой'

ORDER BY f.Название, k.Название;

Выполненное задание представлено на рисунке 6.6.

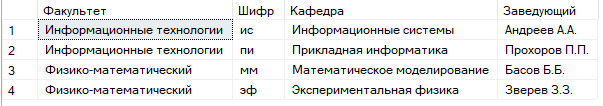


Рисунок 6.6 – Выполненное задание

1. Вывести в запросе для каждого сотрудника номер и фамилию его непосредственного руководителя. Для заведующих кафедрами поле руководителя оставить пустым.

Код выполнения задания представлен ниже:

SELECT

s1.Таб\_номер,

s1.Фамилия AS Сотрудник,

s1.Должность,

CASE

WHEN s1.Таб\_номер = s1.Шеф THEN NULL

ELSE s2.Фамилия

END AS Руководитель,

CASE

WHEN s1.Таб\_номер = s1.Шеф THEN 'Зав. кафедрой'

ELSE 'Подчиненный'

END AS Статус

FROM Сотрудник s1

LEFT JOIN Сотрудник s2 ON s1.Шеф = s2.Таб\_номер

ORDER BY s1.Таб\_номер;

Выполненное задание представлено на рисунке 6.7.

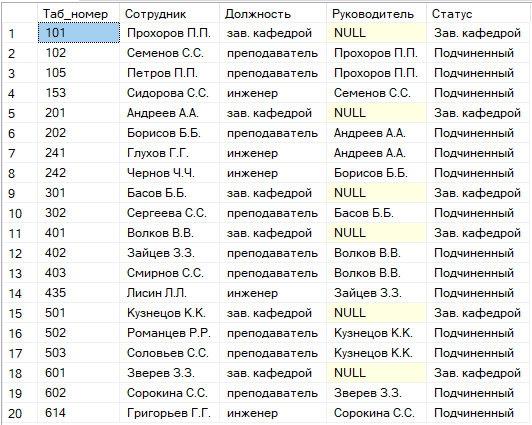


Рисунок 6.7 – Выполненное задание

1. Вывести список студентов, сдавших минимум два экзамена.

Код выполнения задания представлен ниже:

SELECT

st.Рег\_номер,

st.Фамилия,

COUNT(e.Код) AS Количество\_экзаменов

FROM Студент st

INNER JOIN Экзамен e ON st.Рег\_номер = e.Рег\_номер

GROUP BY st.Рег\_номер, st.Фамилия

HAVING COUNT(e.Код) >= 2

ORDER BY Количество\_экзаменов DESC;

Выполненное задание представлено на рисунке 6.8.

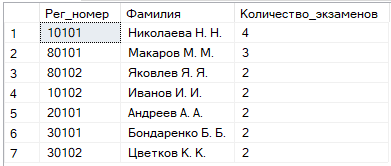


Рисунок 6.8 – Выполненное задание

1. Вывести список инженеров с зарплатой, меньшей 20000 руб.

Код выполнения задания представлен ниже:

SELECT

s.Таб\_номер,

s.Фамилия,

s.Зарплата,

k.Название AS Кафедра,

i.Специальность

FROM Сотрудник s

INNER JOIN Инженер i ON s.Таб\_номер = i.Таб\_номер

INNER JOIN Кафедра k ON s.Шифр = k.Шифр

WHERE s.Зарплата < 20000

ORDER BY s.Зарплата;

Выполненное задание представлено на рисунке 6.9.

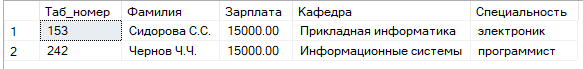


Рисунок 6.9 – Выполненное задание

1. Вывести список студентов, сдавших экзамены в заданной аудитории.

Код выполнения задания представлен ниже:

SELECT DISTINCT

st.Рег\_номер,

st.Фамилия,

sp.Направление,

e.Аудитория,

e.Дата,

d.Название AS Дисциплина

FROM Студент st

INNER JOIN Экзамен e ON st.Рег\_номер = e.Рег\_номер

INNER JOIN Дисциплина d ON e.Код = d.Код

INNER JOIN Специальность sp ON st.Номер = sp.Номер

WHERE e.Аудитория = 'т505' -- пример аудитории

ORDER BY e.Дата;

Выполненное задание представлено на рисунке 6.10.

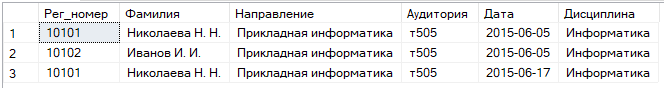


Рисунок 6.10 – Выполненное задание

1. Вывести из таблиц «Студент» и «Экзамен» учетные номера и фамилии студентов, а также количество сданных экзаменов и средний балл для каждого студента только для тех студентов, у которых средний балл не меньше заданного.

Код выполнения задания представлен ниже:

SELECT

st.Рег\_номер,

st.Фамилия,

COUNT(e.Код) AS Количество\_экзаменов,

AVG(CAST(e.Оценка AS FLOAT)) AS Средний\_балл

FROM Студент st

INNER JOIN Экзамен e ON st.Рег\_номер = e.Рег\_номер

GROUP BY st.Рег\_номер, st.Фамилия

HAVING AVG(CAST(e.Оценка AS FLOAT)) >= 4.0

ORDER BY Средний\_балл DESC;

Выполненное задание представлено на рисунке 6.11.



Рисунок 6.11 – Выполненное задание

1. Вывести список заведующих кафедрами и их зарплаты, и степень.

Код выполнения задания представлен ниже:

SELECT

s.Таб\_номер,

s.Фамилия,

k.Название AS Кафедра,

s.Зарплата,

p.Степень,

z.Стаж

FROM Сотрудник s

INNER JOIN Кафедра k ON s.Шифр = k.Шифр

INNER JOIN Зав\_кафедрой z ON s.Таб\_номер = z.Таб\_номер

LEFT JOIN Преподаватель p ON s.Таб\_номер = p.Таб\_номер

WHERE s.Должность = 'зав. кафедрой'

ORDER BY s.Фамилия;

Выполненное задание представлено на рисунке 6.12.

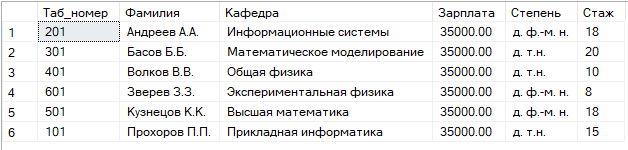


Рисунок 6.12 – Выполненное задание

1. Вывести список профессоров.

Код выполнения задания представлен ниже:

SELECT

s.Таб\_номер,

s.Фамилия,

k.Название AS Кафедра,

p.Звание,

p.Степень,

s.Зарплата

FROM Сотрудник s

INNER JOIN Кафедра k ON s.Шифр = k.Шифр

INNER JOIN Преподаватель p ON s.Таб\_номер = p.Таб\_номер

WHERE p.Звание = 'профессор'

ORDER BY s.Фамилия;

Выполненное задание представлено на рисунке 6.13.

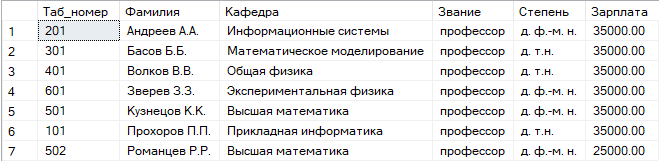


Рисунок 6.13 – Выполненное задание

1. Вывести название дисциплины, фамилию, должность и степень преподавателя, дату и место проведения экзаменов в хронологическом порядке в заданном интервале даты.

Код выполнения задания представлен ниже:

SELECT

d.Название AS Дисциплина,

s.Фамилия AS Преподаватель,

s.Должность,

p.Степень,

e.Дата,

e.Аудитория,

st.Фамилия AS Студент,

e.Оценка

FROM Экзамен e

INNER JOIN Дисциплина d ON e.Код = d.Код

INNER JOIN Сотрудник s ON e.Таб\_номер = s.Таб\_номер

INNER JOIN Студент st ON e.Рег\_номер = st.Рег\_номер

LEFT JOIN Преподаватель p ON s.Таб\_номер = p.Таб\_номер

WHERE e.Дата BETWEEN '2015-06-05' AND '2015-06-10'

ORDER BY e.Дата, e.Аудитория;

Выполненное задание представлено на рисунке 6.14.

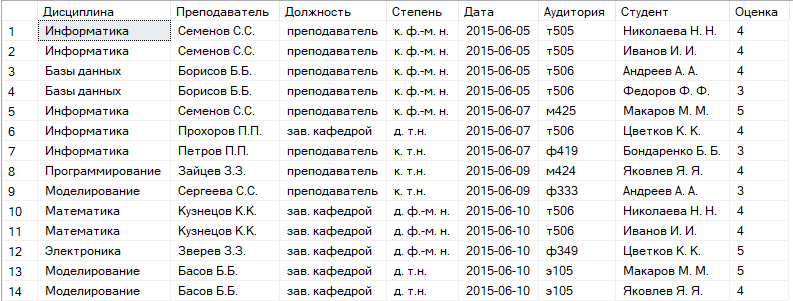


Рисунок 6.14 – Выполненное задание

1. Вывести фамилию преподавателей, принявших более трех экзаменов.

Код выполнения задания представлен ниже:

SELECT

s.Таб\_номер,

s.Фамилия,

k.Название AS Кафедра,

COUNT(e.Код) AS Количество\_экзаменов

FROM Сотрудник s

INNER JOIN Экзамен e ON s.Таб\_номер = e.Таб\_номер

INNER JOIN Кафедра k ON s.Шифр = k.Шифр

WHERE s.Должность = 'преподаватель' OR s.Должность = 'зав. кафедрой'

GROUP BY s.Таб\_номер, s.Фамилия, k.Название

HAVING COUNT(e.Код) > 3

ORDER BY Количество\_экзаменов DESC;

Выполненное задание представлено на рисунке 6.15.

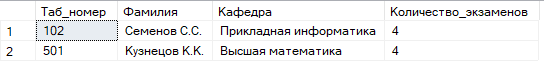


Рисунок 6.15 – Выполненное задание

1. Вывести список студентов, не сдавших ни одного экзамена в указанной дате.

Код выполнения задания представлен ниже:

SELECT

st.Рег\_номер,

st.Фамилия,

sp.Направление

FROM Студент st

INNER JOIN Специальность sp ON st.Номер = sp.Номер

WHERE NOT EXISTS (

SELECT 1

FROM Экзамен e

WHERE e.Рег\_номер = st.Рег\_номер

AND e.Дата = '2015-06-05' -- указанная дата

)

ORDER BY st.Фамилия;

Выполненное задание представлено на рисунке 6.16.

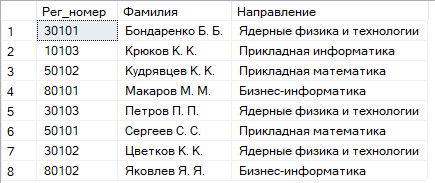


Рисунок 6.16 – Выполненное задание

**Рефлексия:** Я изучил: неявные соединения таблиц; явные соединения таблиц; внутреннее соединение таблиц; внешнее соединение таблиц; соединения таблиц со своей копией.

**Инструкционная карта №7**

**Тема: «Объединение результатов нескольких запросов».**

**Цель работы:**

1. Изучить основы объединения результатов запросов.
2. Изучить UNION.
3. Изучить EXCEPT.
4. Изучить INTERSECT.

**Ход работы:**

1. Была создана база данных «ПЗ07\_Рыбалев», в ней была создана таблица «Страны». Таблица была заполнена данными.
2. Были выполнены следующие примеры:

Пример 1: вывести объединенный результат выполнения запросов, которые выбирают страны с площадью больше 1 млн. кв. км и с населением больше 100 млн. чел. Выполнение примера представлено на рисунке 7.1.



Рисунок 7.1– Выполнение примера 1

**Самостоятельное задание**

Были выполнены следующие задания:

1. Вывести объединенный результат выполнения запросов, которые выбирают страны с площадью меньше 5000 кв. км и с площадью больше 500000 кв. км.

Код выполнения задания представлен ниже:

SELECT \* FROM Страны

WHERE Площадь < 5000

UNION

SELECT \* FROM Страны

WHERE Площадь > 500000

Выполненное задание представлено на рисунке 7.1.



Рисунок 7.2– Выполненное задание

1. Вывести список стран с площадью больше 100 000. кв. км, исключить страны с населением меньше 100 млн. чел.

Код выполнения задания представлен ниже:

SELECT \* FROM Страны

WHERE Площадь > 100000

EXCEPT

SELECT \* FROM Страны

WHERE Население < 100000000

Выполненное задание представлено на рисунке 7.3.

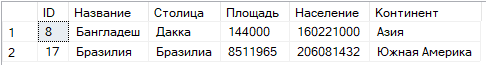


Рисунок 7.3 – Выполненное задание

1. Вывести список стран с площадью меньше 50000 кв. км и с населением меньше 1 млн. чел.

Код выполнения задания представлен ниже:

SELECT \* FROM Страны

WHERE Площадь < 50000

INTERSECT

SELECT \* FROM Страны

WHERE Население < 1000000

Выполненное задание представлено на рисунке 7.4.

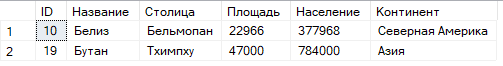


Рисунок 7.4 – Выполненное задание

1. Вывести объединенный результат выполнения запросов, которые выбирают академиков с длинной ФИО больше 26 (включая пробелы) и исключая химиков.

Код выполнения задания представлен ниже:

SELECT \* FROM Академики

WHERE LEN(ФИО) > 26

INTERSECT

SELECT \* FROM Академики

WHERE Специализация != 'химик'

Выполненное задание представлено на рисунке 7.5.



Рисунок 7.5 – Выполненное задание

1. Вывести общие строки результата запросов, которые выбирают академиков со словосочетанием «ов» в ФИО присвоением звания в 20 веке.

Код выполнения задания представлен ниже:

SELECT \* FROM Академики

WHERE CHARINDEX('ов',ФИО,1) > 0

intersect

SELECT \* FROM Академики

WHERE Год\_присвоения\_звания >1900

Выполненное задание представлено на рисунке 7.6.



Рисунок 7.6 – Выполненное задание

1. Вывести объединенный результат работы запросов, один из которых выводит список академиков с ФИО начинающимся на А, а другой выводит список академиков с ФИО начинающимся на Б.

Код выполнения задания представлен ниже:

SELECT \* FROM Академики

WHERE LEFT(ФИО,1)='А'

UNION

SELECT \* FROM Академики

WHERE LEFT(ФИО,1)='Б'

Выполненное задание представлено на рисунке 7.7.

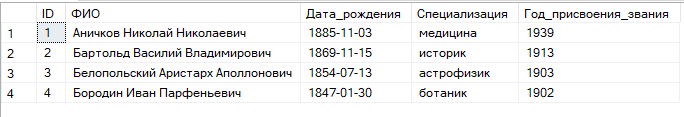


Рисунок 7.7 – Выполненное задание

1. Вывести объединенный результат работы запросов, один из которых выводит список академиков с ФИО начинающимся на А, а другой выводит список академиков с ФИО начинающимся на Б. Из объединенного списка на исключить всех Астрофизиков.

Код выполнения задания представлен ниже:

SELECT \* FROM Академики

WHERE LEFT(ФИО,1)='А'

UNION

SELECT \* FROM Академики

WHERE LEFT(ФИО,1)='Б'

EXCEPT

SELECT \* FROM Академики

WHERE Специализация='Астрофизик'

Выполненное задание представлено на рисунке 7.8.



Рисунок 7.8 – Выполненное задание

1. Вывести список академиков, родившихся в прошлом веке или с первой буквой А в ФИО.

Код выполнения задания представлен ниже:

SELECT \* FROM Академики

WHERE YEAR(Дата\_рождения)>=1901

UNION

SELECT \* FROM Академики

WHERE LEFT(ФИО,1) ='А'

Выполненное задание представлено на рисунке 7.9.



Рисунок 7.9 – Выполненное задание

1. Вывести список академиков, родившихся в прошлом веке и с первой буквой А в ФИО.

Код выполнения задания представлен ниже:

SELECT \* FROM Академики

WHERE YEAR(‘Дата\_рождения’)>=1901

INTERSECT

SELECT \* FROM Академики

WHERE LEFT(ФИО,1)='А'

В результате выполнения задания, таких академиков не было.

1. Вывести список академиков, получивших звание в прошлом веке и без первой букв А в ФИО.

Код выполнения задания представлен ниже:

SELECT \* FROM Академики

WHERE Год\_присвоения\_звания<=1901

EXCEPT

SELECT \* FROM Академики

WHERE LEFT(ФИО,1)='А'

Выполненное задание представлено на рисунке 7.10.



Рисунок 7.10 – Выполненное задание

**Рефлексия:** Я изучил: основы объединения результатов запросов; UNION; EXCEPT; INTERSECT.

**Инструкционная карта №8**

**Тема: «Подзапросы».**

**Цель работы:**

1. Изучить виды вложенных запросов.
2. Изучить некоррелирующие подзапросы.
3. Изучить коррелирующие подзапросы.
4. Изучить применение конструкции IN к подзапросам.
5. Изучить конструкцию ALL.
6. Изучить конструкцию ANY/SOME.
7. Изучить конструкцию EXISTS.

**Ход работы:**

1. Была создана база данных «ПЗ08\_Рыбалев», в ней была создана таблица «Страны». Таблица была заполнена данными.
2. Были выполнены следующие примеры:

Пример 1: вывести список стран и процентное соотношение их населения к суммарному населению мира. Выполнение примера представлено на рисунке 8.1.

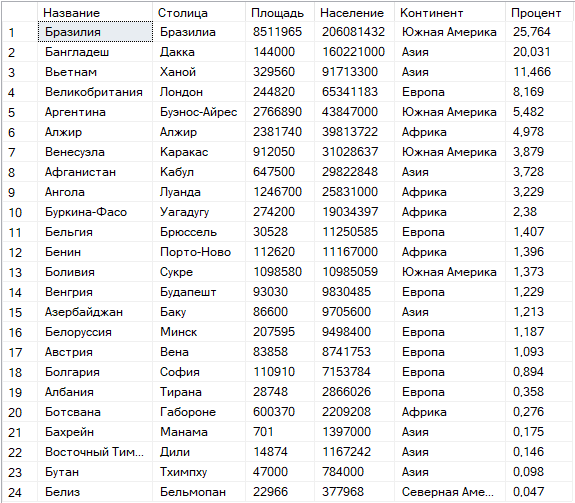


Рисунок 8.1– Выполнение примера 1

Пример 2: вывести список стран мира, население которых больше, чем среднее население всех стран мира. Выполнение примера представлено на рисунке 8.2.

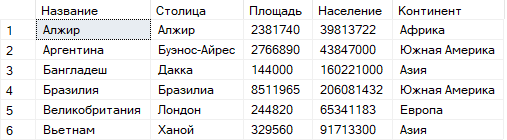


Рисунок 8.2– Выполнение примера 2

Пример 3: с помощью подзапроса вывести список африканских стран, население которых больше 50 млн. чел. В результате выполнения запроса, такой информации не оказалось в таблице.

**Самостоятельное задание**

Были выполнены следующие задания:

1. Вывести список стран и процентное соотношение площади каждой из них к общей площади всех стран мира.

Код выполнения задания представлен ниже:

SELECT

Название,

Площадь,

CAST(Площадь \* 100.0 / (SELECT SUM(Площадь) FROM Страны) AS DECIMAL(10,2)) AS [Процент от мировой площади]

FROM Страны

ORDER BY Площадь DESC;

Выполненное задание представлено на рисунке 8.3.



Рисунок 8.3– Выполненное задание

1. Вывести список стран мира, плотность населения которых больше, чем средняя плотность населения всех стран мира.

Код выполнения задания представлен ниже:

WITH Плотность AS (SELECT Население\*1.0/NULLIF(Площадь,0) AS Пл FROM Страны)

SELECT Название, Население, Площадь,

CAST(Население\*1.0/NULLIF(Площадь,0) AS DECIMAL(10,2)) AS Плотность

FROM Страны

WHERE Население\*1.0/NULLIF(Площадь,0) > (SELECT AVG(Пл) FROM Плотность);

Выполненное задание представлено на рисунке 8.4.

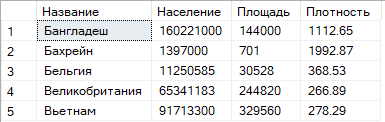


Рисунок 8.4 – Выполненное задание

1. С помощью подзапроса вывести список европейских стран, население которых меньше 5 млн. чел.

Код выполнения задания представлен ниже:

SELECT Название, Население FROM Страны

WHERE Континент = N'Европа' AND Население < 5000000;

Выполненное задание представлено на рисунке 8.5.



Рисунок 8.5 – Выполненное задание

1. Вывести список стран и процентное соотношение их площади к суммарной площади той части мира, где они находятся.

Код выполнения задания представлен ниже:

SELECT Название, Континент, Площадь,

CAST(Площадь\*100.0/SUM(Площадь) OVER(PARTITION BY Континент) AS DECIMAL(10,2)) AS [% континента]

FROM Страны;

Выполненное задание представлено на рисунке 8.6.



Рисунок 8.6 – Выполненное задание

1. Вывести список стран мира, площадь которых больше, чем средняя площадь стран той части света, где они находятся.

Код выполнения задания представлен ниже:

WITH СредниеПоКонтиненту AS (

SELECT Континент, AVG(Площадь\*1.0) AS СредняяПлощадь

FROM Страны GROUP BY Континент

)

SELECT s.Название, s.Континент, s.Площадь, sp.СредняяПлощадь

FROM Страны s

JOIN СредниеПоКонтиненту sp ON s.Континент = sp.Континент

WHERE s.Площадь > sp.СредняяПлощадь;

Выполненное задание представлено на рисунке 8.7.



Рисунок 8.7 – Выполненное задание

1. Вывести список стран мира, которые находятся в тех частях света, средняя плотность населения которых превышает общемировую.

Код выполнения задания представлен ниже:

WITH Плотности AS (

SELECT Континент, AVG(Население\*1.0/NULLIF(Площадь,0)) AS СредняяПлотность

FROM Страны GROUP BY Континент

HAVING AVG(Население\*1.0/NULLIF(Площадь,0)) > (SELECT AVG(Население\*1.0/NULLIF(Площадь,0)) FROM Страны)

)

SELECT s.\* FROM Страны s

JOIN Плотности p ON s.Континент = p.Континент;

Выполненное задание представлено на рисунке 8.8.

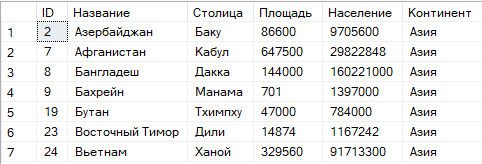


Рисунок 8.8 – Выполненное задание

1. Вывести список южноамериканских стран, в которых живет больше людей, чем в любой африканской стране.

Код выполнения задания представлен ниже:

SELECT Название, Население FROM Страны

WHERE Континент = N'Южная Америка'

AND Население > (SELECT MAX(Население) FROM Страны WHERE Континент = N'Африка');

Выполненное задание представлено на рисунке 8.9.



Рисунок 8.9 – Выполненное задание

1. Вывести список африканских стран, в которых живет больше людей, чем хотя бы в одной южноамериканской стране.

Код выполнения задания представлен ниже:

SELECT Название, Население FROM Страны

WHERE Континент = N'Африка'

AND Население > (SELECT MIN(Население) FROM Страны WHERE Континент = N'Южная

Америка');

Выполненное задание представлено на рисунке 8.10.



Рисунок 8.10 – Выполненное задание

1. Если в Африке есть хотя бы одна страна, площадь которой больше 2 млн. кв. км, вывести список всех африканских стран.

Код выполнения задания представлен ниже:

IF EXISTS(SELECT 1 FROM Страны WHERE Континент = N'Африка' AND Площадь > 2000000)

SELECT \* FROM Страны WHERE Континент = N'Африка';

Выполненное задание представлено на рисунке 8.11.

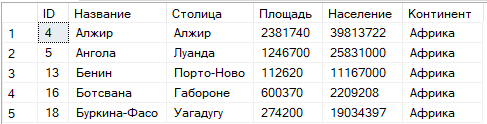


Рисунок 8.11 – Выполненное задание

1. Вывести список стран той части света, где находится страна «Фиджи».

Код выполнения задания представлен ниже:

SELECT \* FROM Страны

WHERE Континент = (SELECT Континент FROM Страны WHERE Название = N'Фиджи');

В результате выполнения запроса, данных не было найдено

1. Вывести список стран, население которых не превышает население страны «Фиджи».

Код выполнения задания представлен ниже:

SELECT \* FROM Страны

WHERE Население <= (SELECT Население FROM Страны WHERE Название = N'Фиджи');

В результате выполнения запроса, данных не было найдено

1. Вывести название страны с наибольшим населением среди стран с наименьшей площадью на каждом континенте.

Код выполнения задания представлен ниже:

WITH МинПлощади AS (

SELECT Континент, MIN(Площадь) AS МинПл FROM Страны GROUP BY Континент

),

Кандидаты AS (

SELECT s.\*,

ROW\_NUMBER() OVER(PARTITION BY s.Континент ORDER BY s.Население DESC) as rn

FROM Страны s

JOIN МинПлощади m ON s.Континент = m.Континент AND s.Площадь = m.МинПл

)

SELECT Название, Континент, Площадь, Население FROM Кандидаты WHERE rn = 1;

Выполненное задание представлено на рисунке 8.12.



Рисунок 8.12 – Выполненное задание

1. Вывести список академиков, которые входят в две самые распространённые специализации.

Код выполнения задания представлен ниже:

WITH Топ2Спец AS (

SELECT TOP 2 Специализация FROM Академики

GROUP BY Специализация ORDER BY COUNT(\*) DESC

)

SELECT \* FROM Академики WHERE Специализация IN (SELECT Специализация FROM

Топ2Спец);

Выполненное задание представлено на рисунке 8.13.

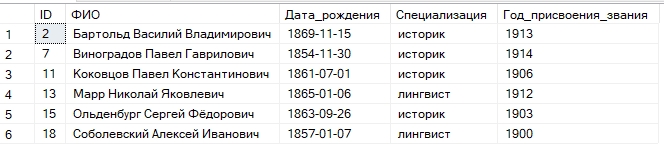


Рисунок 8.13 – Выполненное задание

1. Вывести список специализаций и процентное соотношение числа академиков в каждой специализации к общему количеству академиков. Исключить повторение наименований специализаций.

Код выполнения задания представлен ниже:

SELECT Специализация, COUNT(\*) AS Количество,

CAST(COUNT(\*)\*100.0/(SELECT COUNT(\*) FROM Академики) AS DECIMAL(5,2)) AS [%]

FROM Академики GROUP BY Специализация ORDER BY Количество DESC;

Выполненное задание представлено на рисунке 8.14.

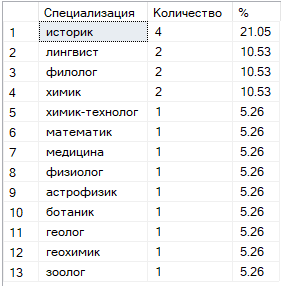


Рисунок 8.14 – Выполненное задание

1. Вывести список академиков, возраст которых на момент присвоения звания выше среднего возраста на момент присвоения звания.

Код выполнения задания представлен ниже:

SELECT \* FROM Академики

WHERE Год\_присвоения\_звания - DATEPART(YEAR, Дата\_рождения) >

(SELECT AVG(Год\_присвоения\_звания - DATEPART(YEAR, Дата\_рождения)) FROM

Академики);

Выполненное задание представлено на рисунке 8.15.

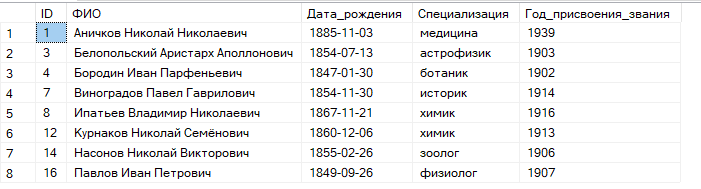


Рисунок 8.15 – Выполненное задание

1. С помощью подзапроса вывести список академиков, специализация которых содержит не более 4 академиков.

Код выполнения задания представлен ниже:

SELECT \* FROM Академики a

WHERE (SELECT COUNT(\*) FROM Академики WHERE Специализация = a.Специализация) <= 4;

Выполненное задание представлено на рисунке 8.16.

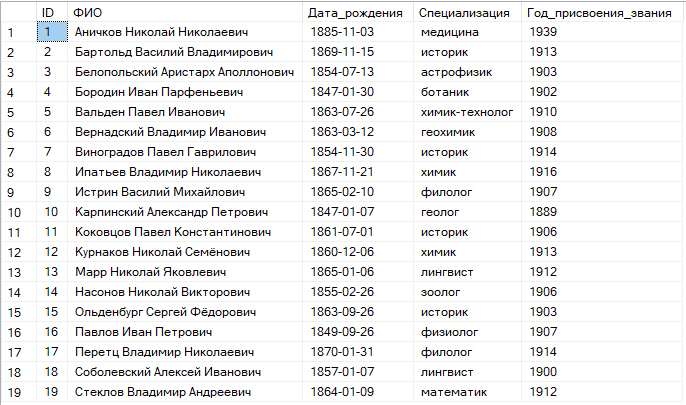


Рисунок 8.16 – Выполненное задание

1. Вывести список специализаций, количество академиков в которой больше, чем среднее количество академиков в специализации.

Код выполнения задания представлен ниже:

SELECT Специализация, COUNT(\*) AS Количество FROM Академики

GROUP BY Специализация

HAVING COUNT(\*) > (SELECT AVG(cnt\*1.0) FROM (SELECT COUNT(\*) cnt FROM Академики

GROUP BY Специализация) t);

Выполненное задание представлено на рисунке 8.17.



Рисунок 8.17 – Выполненное задание

**Рефлексия:** Я изучил: виды вложенных запросов; некоррелирующие подзапросы; коррелирующие подзапросы; применение конструкции IN к подзапросам; конструкцию ALL; конструкцию ANY/SOME; конструкцию EXISTS.

**Инструкционная карта №9**

**Тема: «Изучение команд создания и удаления БД и таблиц».**

**Цель работы:**

1. Изучить создание базы данных.
2. Изучить удаление базы данных.
3. Изучить создание таблиц.
4. Изучить удаление таблиц.

**Ход работы:**

1. Была создана база данных «ПЗ09\_Рыбалев», в ней были созданы таблицы «Страны» и «Управление». Таблицы были заполнена соответствующими данными.
2. Были выполнены следующие примеры:

Пример 1: Создать таблицу «Управление». Таблица «Управление» была создана.

Пример 2: удалить таблицу «Управление. Таблица «Управление» была удалена.

Пример 3: Создать таблицу «Управление», значения столбца «ID» сделать уникальными, а столбец «Вид» запретить оставлять незаполненным. Таблица «Управление» с ограничениями была создана.

Пример 4: Создать таблицу «Управление», в столбец «ID» разрешить вводить значения меньше 200, а для столбца «Вид» установить значение по умолчанию «Президентская республика. Таблица «Управления» была изменена с соответствующими изменениями.

Пример 5: Создать таблицу «Управление», столбец «ID» определить, как основной ключ, и настроить автоматический идентификатор с начальным значением 5 и с шагом 3. Таблица «Управления» была изменена с соответствующими изменениями.

**Самостоятельное задание**

1. В базе данных «TSQL2012» были созданы таблицы «Цветы», «Животные», «Управление», «Страны» с соответствующими ограничениями. Созданные таблицы были заполнены данными.
2. Были даны ответы на следующие вопросы:
3. Что выполняется по команде IF DB\_ID('TSQL2012') IS NOT NULL DROP DATABASE TSQL2012?

Эта команда проверяет, существует ли база данных с именем 'TSQL2012' в текущем экземпляре SQL Server. Если база данных существует, то команда DROP DATABASE TSQL2012 удаляет эту базу данных со всеми её таблицами, данными и другими объектами.

1. Перечислите типы полей, используемые в каждой создаваемой таблице в запросе TSQL2012

**Таблица Животные:**

* Животное - NVARCHAR(50) - строка переменной длины (до 50 символов)
* Потребляемая пища - NVARCHAR(200) - строка переменной длины (до 200 символов)
* Группа - NVARCHAR(50) - строка переменной длины (до 50 символов)
* ID (в альтернативной версии) - INT - целое число с автоинкрементном

**Таблица Управление:**

* ID - INT - целое число (первичный ключ)
* Вид - NVARCHAR(50) - строка переменной длины (до 50 символов)

**Таблица Страны:**

* Страна - NVARCHAR(100) - строка переменной длины (до 100 символов)
* Столица - NVARCHAR(100) - строка переменной длины (до 100 символов)
* Часть света - NVARCHAR(50) - строка переменной длины (до 50 символов)
* Население тыс. чел. - INT - целое число
* Площадь тыс. кв. км - DECIMAL(10,1) - десятичное число (10 цифр, 1 после запятой)
* Тип управления - INT - целое число (внешний ключ к таблице Управление)

**Таблица Цветы:**

* Название семейства - NVARCHAR(100) - строка (до 100 символов)
* число видов - NVARCHAR(50) - строка (до 50 символов)
* Соцветие - NVARCHAR(100) - строка (до 100 символов)
* Плод - NVARCHAR(100) - строка (до 100 символов)
* Особенности строения вегетативных органов - NVARCHAR(MAX) - строка неограниченной длины
* Представители и их практическое применение - NVARCHAR(MAX) - строка неограниченной длины
* Класс - NVARCHAR(50) - строка (до 50 символов)
* ID (в альтернативной версии) - INT - целое число с автоинкрементном

1. Есть ли в создаваемых таблицах DATABASE TSQL2012 таблицы без PRIMERY KEY, если есть, то какие

У всех таблиц есть первичный ключ.

1. При создании таблицы HR.Employees используется выражение CONSTRAINT CHK\_birthdate CHECK(birthdate <= CURRENT\_TIMESTAMP), что оно устанавливает?

Это ограничение устанавливает правило проверки, что дата рождения сотрудника (birthdate) должна быть меньше или равна текущей дате и времени (CURRENT\_TIMESTAMP).

**Что это означает**:

* Сотрудник не может иметь дату рождения в будущем
* Это предотвращает ввод некорректных данных (например, дата рождения 2050 года)
* CURRENT\_TIMESTAMP возвращает текущую системную дату и время сервера
* Ограничение будет автоматически проверяться при INSERT или UPDATE

1. При создании таблицы Production.Products используется выражение CONSTRAINT CHK\_Products\_unitprice CHECK(unitprice >= 0), что оно устанавливает?

Это ограничение устанавливает правило, что цена товара (unitprice) должна быть больше или равна нулю.

**Что это означает**:

* Цена товара не может быть отрицательной
* Допускается нулевая цена (возможно, для бесплатных товаров или акций)
* Это бизнес-правило, предотвращающее ввод некорректных цен
* Автоматически проверяется при каждой вставке или обновлении цены

**Рефлексия:** Я изучил: создание базы данных; удаление базы данных; создание таблиц; удаление таблиц.

**Инструкционная карта №10**

**Тема: «Изучение команд работы со строками».**

**Цель работы:**

1. Изучить команду INSERT.
2. Изучить команду UPDATE.
3. Изучить команду DELETE.
4. Изучить команду TRUNCATE TABLE.
5. Изучить конструкцию SELECT … INTO.

**Ход работы:**

1. Была создана база данных «ПЗ10\_Рыбалев», в ней была создана таблица «Ученики». Таблица была заполнена соответствующими данными.
2. Были выполнены следующие примеры:

Пример 1: В таблицу «Ученики» внести новую запись для ученика гимназии Маркина, который по физике набрал 96 баллов. Выполнение примера представлено на рисунке 10.1.

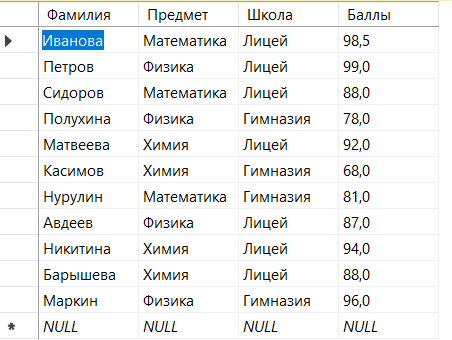


Рисунок 10.1 – Выполнение примера 1

Пример 2: В таблицу «Ученики» внести две строки, для ученицы лицея Никишиной, которая по химии набрала 77 баллов, и для ученика школы № 18 Андреева, оценка которого по математике неизвестна. Выполнение примера представлено на рисунке 10.2.

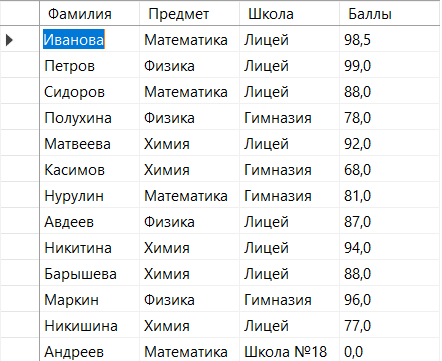


Рисунок 10.2 – Выполнение примера 2

Пример 3: В таблице «Ученики» изменить данные Андреева, оценку исправить на 87. Выполнение примера представлено на рисунке 10.3.

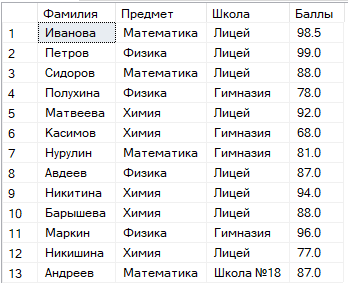


Рисунок 10.3 – Выполнение примера 3

Пример 4: В таблице «Ученики» изменить данные Никишиной, школу исправить на «Школа №31», а предмет на математику. Выполнение примера представлено на рисунке 10.4.

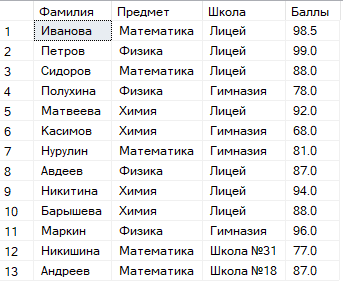


Рисунок 10.4 – Выполнение примера 4

Пример 5: В таблице «Ученики» изменить данные всех учеников по математике, оценку уменьшить на 5 баллов. Выполнение примера представлено на рисунке 10.5.



Рисунок 10.5 – Выполнение примера 5

Пример 6: В таблице «Ученики» удалить данные всех учеников из школы №18. Выполнение примера представлено на рисунке 10.6.

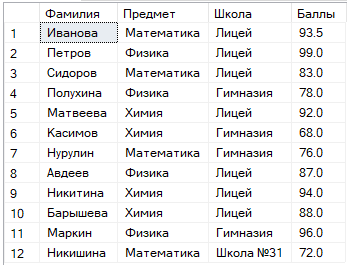


Рисунок 10.6 – Выполнение примера 6

Пример 7: Создать таблицу «Лицеисты» и скопировать туда всех лицеистов. Выполнение примера представлено на рисунке 10.7.

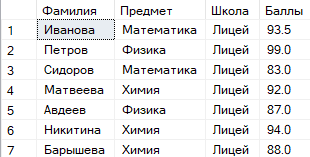


Рисунок 10.7 – Выполнение примера 7

Пример 8: Очистить таблицу «Лицеисты». В результате выполнения примера таблица «Лицеисты» была очищена.

**Самостоятельное задание**

Были выполнены следующие задания:

1. Вставьте в таблицу «Ученики» новые записи для каждого ученика содержащие сведения об отсутствующих предмете.

Код для выполнения задания представлен ниже:

INSERT INTO Ученики (Фамилия, Предмет, Школа, Баллы)

SELECT DISTINCT u1.Фамилия, u2.Предмет, u1.Школа, NULL

FROM Ученики u1

CROSS JOIN (SELECT DISTINCT Предмет FROM Ученики) u2

WHERE NOT EXISTS (

SELECT 1 FROM Ученики u3

WHERE u3.Фамилия = u1.Фамилия AND u3.Предмет = u2.Предмет

);

Выполнение задание представлено на рисунке 10.8.



Рисунок 10.8 – Выполнение задания

1. В таблице «Ученики» изменить данные следующим образом: все Баллы, имеющие значение NULL исправить у лицеистов на 50 баллов, у гимназистов на 51 балл.

Код для выполнения задания представлен ниже:

UPDATE Ученики

SET Баллы = CASE

WHEN Школа = N'Лицей' THEN 50.0

WHEN Школа = N'Гимназия' THEN 51.0

END

WHERE Баллы = 0;

Выполнение задание представлено на рисунке 10.9.



Рисунок 10.9 – Выполнение задания

1. В таблице «Ученики» изменить данные следующим образом: всем лицеистам, имеющим Баллы по химии <66 увеличить их на 15%, а гимназистам при таком же условии увеличить на 10%

Код для выполнения задания представлен ниже:

UPDATE Ученики

SET Баллы = CASE

WHEN Школа = N'Лицей' AND Предмет = N'Химия' AND Баллы < 66 THEN Баллы \* 1.15

WHEN Школа = N'Гимназия' AND Предмет = N'Химия' AND Баллы < 66 THEN Баллы \* 1.10

ELSE Баллы

END

WHERE Предмет = N'Химия' AND Баллы < 66;

Выполнение задание представлено на рисунке 10.10.



Рисунок 10.10 – Выполнение задания

1. Создать таблицу «Гимназисты» и скопировать туда данные всех гимназистов, кроме тех, которые набрали меньше 60 баллов по любому предмету.

Код для выполнения задания представлен ниже:

IF OBJECT\_ID('Гимназисты', 'U') IS NOT NULL

DROP TABLE Гимназисты;

CREATE TABLE Гимназисты (

ID INT IDENTITY(1,1) PRIMARY KEY,

Фамилия NVARCHAR(50) NOT NULL,

Предмет NVARCHAR(50) NOT NULL,

Школа NVARCHAR(50) NOT NULL DEFAULT N'Гимназия',

Баллы DECIMAL(4,1) NOT NULL,

CONSTRAINT CHK\_Гимназисты\_Баллы CHECK (Баллы >= 60),

CONSTRAINT CHK\_Гимназисты\_Школа CHECK (Школа = N'Гимназия')

);

INSERT INTO Гимназисты (Фамилия, Предмет, Школа, Баллы)

SELECT Фамилия, Предмет, Школа, Баллы

FROM Ученики

WHERE Школа = N'Гимназия' AND Баллы >= 60;

Выполнение задание представлено на рисунке 10.11.



Рисунок 10.11 – Выполнение задания

**Рефлексия:** Я изучил: команду INSERT; команду UPDATE; команду DELETE; команду TRUNCATE TABLE; конструкцию SELECT … INTO.

**Инструкционная карта №11**

**Тема: «Программирование на SQL. Переменные, условные выражения и циклы».**

**Цель работы:**

1. Изучение переменных в T-SQL.
2. Изучение условных выражений.
3. Изучение циклов.

**Ход работы:**

1. Была создана база данных «ПЗ11\_Рыбалев», в ней была создана таблица «Ученики». Таблица была заполнена соответствующими данными.
2. Были выполнены следующие примеры:

Пример 1: Даны числа a и b. Найти и вывести их сумму. Выполнение примера представлено на рисунке 11.1.

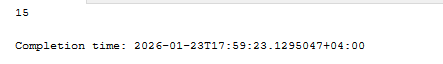


Рисунок 11.1 – Выполнение примера 1

Пример 2: В таблице «Ученики» найти разницу между наибольшими баллами среди лицеистов и гимназистов. Выполнение примера представлено на рисунке 11.2.

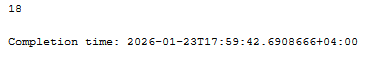


Рисунок 11.2 – Выполнение примера 2

Пример 3: В таблице «Ученики» найти разницу между наибольшими и наименьшими баллами. Выполнение примера представлено на рисунке 11.3.



Рисунок 11.3 – Выполнение примера 3

Пример 4: Дано случайное целое число меньше 1000. Вывести его квадрат. Выполнение примера представлено на рисунке 11.4.



Рисунок 11.4 – Выполнение примера 4

Пример 5: Даны случайные целые числа a и b. Найти наибольшие из них. Выполнение примера представлено на рисунке 11.5.

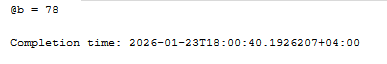


Рисунок 11.5 – Выполнение примера 5

Пример 6: Дано случайное целое число a. Проверить, делится ли данное число на 3. Выполнение примера представлено на рисунке 11.6.



Рисунок 11.6 – Выполнение примера 6

Пример 7: Дано случайное целое число N (N <1000). Если оно является степенью числа 5, то вывести «Да», если не является – вывести «Нет». Выполнение примера представлено на рисунке 11.7.



Рисунок 11.7 – Выполнение примера 7

Пример 8: Даны случайные целые числа a и b. Найти наибольший общий делитель. Выполнение примера представлено на рисунке 11.8.

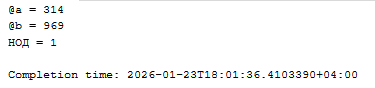


Рисунок 11.8 – Выполнение примера 8

Пример 9: Даны два целых числа A и B (A < B). Найти сумму всех целых чисел от A до B включительно. Выполнение примера представлено на рисунке 11.9.



Рисунок 11.9 – Выполнение примера 9

Пример 10: Дано случайное целое число N (N < 100). Найти квадрат данного числа, используя для его вычисления следующую формулу 𝑁2 = 1 + 3 + 5 + ⋯ + (2 ∙ 𝑁 − 1). После добавления к сумме каждого слагаемого выводить текущее значение суммы (в результате будут выведены квадраты всех целых чисел от 1 до N. Выполнение примера представлено на рисунке 11.10.



Рисунок 11.10 – Выполнение примера 10

Пример 11: Даны случайные целые числа A и B (A < B). Вывести все целые числа от A до B включительно; при этом число A должно выводиться 1 раз, число A + 1 должно выводиться 2 раза и т.д. Выполнение примера представлено на рисунке 11.11.

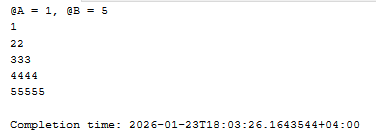


Рисунок 11.11 – Выполнение примера 11

Пример 12: напечатать те из двузначных чисел, которые делятся на 4, но не делятся на 6. Выполнение примера представлено на рисунке 11.12.



Рисунок 11.12 – Выполнение примера 12

Пример 13: Даны два целых числа D (день) и M (месяц), определяющие правильную дату не високосного года. Вывести значения D и M для даты, следующей за указанной. Выполнение примера представлено на рисунке 11.13.



Рисунок 11.13 – Выполнение примера 13

Пример 14: Вывести слово «Нижневартовск» на экран столько раз, сколько в нем букв. Выполнение примера представлено на рисунке 11.14.

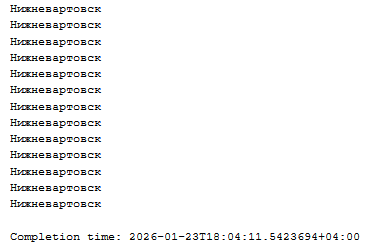


Рисунок 11.14 – Выполнение примера 14

Пример 15: Напишите код для вывода на экран с помощью цикла. Выполнение примера представлено на рисунке 11.15.

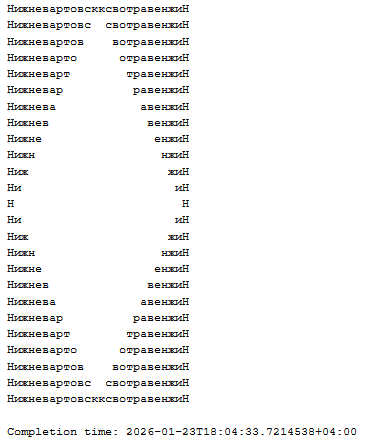


Рисунок 11.15 – Выполнение примера 15

**Самостоятельное задание**

Были выполнены следующие задания:

1. Даны числа A и B. Найти и вывести их произведение.

Код для выполнения задания представлен ниже:

DECLARE @A INT = 5, @B INT = 7;

SELECT @A AS A, @B AS B, @A \* @B AS [Произведение];

GO

Выполнение задание представлено на рисунке 11.16.



Рисунок 11.16 – Выполнение задания

1. В таблице «Ученики» найти разницу между средними баллами лицеистов и гимназистов.

Код для выполнения задания представлен ниже:

SELECT

(SELECT AVG(Баллы) FROM Ученики WHERE Школа = N'Лицей') -

(SELECT AVG(Баллы) FROM Ученики WHERE Школа = N'Гимназия') AS [Разница средних баллов];

Выполнение задание представлено на рисунке 11.17.



Рисунок 11.17 – Выполнение задания

1. В таблице «Ученики» проверить на четность количество строк.

Код для выполнения задания представлен ниже:

SELECT

COUNT(\*) AS [Количество строк],

CASE

WHEN COUNT(\*) % 2 = 0 THEN N'Четное'

ELSE N'Нечетное'

END AS [Четность]

FROM Ученики;

Выполнение задание представлено на рисунке 11.18.



Рисунок 11.18 – Выполнение задания

1. Дано четырехзначное число. Вывести это число и сумму его цифр.

Код для выполнения задания представлен ниже:

DECLARE @num INT = 1234;

DECLARE @sum INT = 0;

DECLARE @temp INT = @num;

WHILE @temp > 0

BEGIN

SET @sum = @sum + (@temp % 10);

SET @temp = @temp / 10;

END

SELECT @num AS [Число], @sum AS [Сумма цифр];

Выполнение задание представлено на рисунке 11.19.



Рисунок 11.19 – Выполнение задания

1. Даны случайные целые числа a, b и c. Вывести эти числа и наименьшее из них.

Код для выполнения задания представлен ниже:

DECLARE @a INT = 15, @b INT = 8, @c INT = 22;

SELECT

@a AS a,

@b AS b,

@c AS c,

CASE

WHEN @a <= @b AND @a <= @c THEN @a

WHEN @b <= @a AND @b <= @c THEN @b

ELSE @c

END AS [Наименьшее число];

Выполнение задание представлено на рисунке 11.20.



Рисунок 11.20 – Выполнение задания

1. Дано случайное целое число a. Проверить, делится ли данное число на 11. Вывести это число и результат проверки.

Код для выполнения задания представлен ниже:

DECLARE @number INT = 121;

SELECT

@number AS [Число],

CASE

WHEN @number % 11 = 0 THEN N'Делится на 11'

ELSE N'Не делится на 11'

END AS [Результат проверки];

Выполнение задание представлено на рисунке 11.22.



Рисунок 11.22 – Выполнение задания

1. Дано случайное целое число N (N <1000). Если оно является степенью числа 7, то вывести «Да», если не является – вывести «Нет», также вывести само число.

Код для выполнения задания представлен ниже:

DECLARE @N INT = 343;

DECLARE @result NVARCHAR(10) = N'Нет';

DECLARE @power INT = 1;

DECLARE @current INT = 7;

WHILE @current <= @N

BEGIN

IF @current = @N

BEGIN

SET @result = N'Да';

BREAK;

END

SET @power = @power + 1;

SET @current = POWER(7, @power);

END

SELECT @N AS N, @result AS [Является степенью 7?];

Выполнение задание представлено на рисунке 11.23.



Рисунок 11.23 – Выполнение задания

1. Даны случайные целые числа a и b. Найти наименьший общий кратный (НОК). Вывести числа а, b и НОК.

Код для выполнения задания представлен ниже:

DECLARE @A2 INT = 12, @B2 INT = 18;

DECLARE @max\_num INT = CASE WHEN @A2 > @B2 THEN @A2 ELSE @B2 END;

DECLARE @lcm INT = @max\_num;

WHILE 1 = 1

BEGIN

IF @lcm % @A2 = 0 AND @lcm % @B2 = 0

BREAK;

SET @lcm = @lcm + @max\_num;

END

SELECT @A2 AS a, @B2 AS b, @lcm AS [НОК];

Выполнение задание представлено на рисунке 11.24.



Рисунок 11.24 – Выполнение задания

1. Даны два целых числа A и B (A<B). Найти сумму квадратов всех целых чисел от A до B включительно.

Код для выполнения задания представлен ниже:

DECLARE @A3 INT = 1, @B3 INT = 5;

DECLARE @sum\_squares INT = 0;

DECLARE @i INT = @A3;

WHILE @i <= @B3

BEGIN

SET @sum\_squares = @sum\_squares + (@i \* @i);

SET @i = @i + 1;

END

SELECT @A3 AS A, @B3 AS B, @sum\_squares AS [Сумма квадратов];

Выполнение задание представлено на рисунке 11.25.



Рисунок 11.25 – Выполнение задания

1. Найти первое натуральное число, которое при делении на 2, 3, 4, 5, и 6 дает остаток 1, но делится на 7.

Код для выполнения задания представлен ниже:

DECLARE @num2 INT = 1;

DECLARE @found INT = 0;

WHILE @found = 0

BEGIN

IF (@num2 % 2 = 1) AND

(@num2 % 3 = 1) AND

(@num2 % 4 = 1) AND

(@num2 % 5 = 1) AND

(@num2 % 6 = 1) AND

(@num2 % 7 = 0)

BEGIN

SET @found = @num2;

BREAK;

END

SET @num2 = @num2 + 1;

END

SELECT @found AS [Найденное число];

Выполнение задание представлено на рисунке 11.26.



Рисунок 11.26 – Выполнение задания

1. Вывести свою фамилию на экран столько раз, сколько в нем букв.

Код для выполнения задания представлен ниже:

DECLARE @surname NVARCHAR(50) = N'Рыбалев';

DECLARE @length INT = LEN(@surname);

DECLARE @counter INT = 1;

WHILE @counter <= @length

BEGIN

PRINT @surname;

SET @counter = @counter + 1;

END

Выполнение задание представлено на рисунке 11.27.

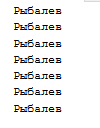


Рисунок 11.27 – Выполнение задания

1. Напишите код для вывода на экран с помощью цикла.

Код для выполнения задания представлен ниже:

DECLARE @word NVARCHAR(50) = N'Нижневартовск';

DECLARE @length INT = LEN(@word);

DECLARE @row INT = 1;

DECLARE @result\_string NVARCHAR(MAX);

DECLARE @spaces INT;

WHILE @row <= @length

BEGIN

SET @spaces = @length - @row;

SET @result\_string = REPLICATE(' ', @spaces) +

SUBSTRING(@word, 1, @row) +

SUBSTRING(REVERSE(@word), @length - @row + 2, @row - 1);

PRINT @result\_string;

SET @row = @row + 1;

END

Выполнение задание представлено на рисунке 11.28.

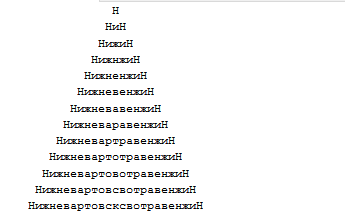


Рисунок 11.28 – Выполнение задания

**Рефлексия:** Я изучил: переменных в T-SQL; условных выражений; циклов.

**Инструкционная карта №12**

**Тема: «Скалярные функции, функции INLINE, MULTI-STATEMENT».**

**Цель работы:**

1. Изучение скалярных функций.
2. Изучение функции INLINE.
3. Изучение функции MULTI-STATEMENT.
4. Изучение удаления пользовательских функций.

**Ход работы:**

1. Была создана база данных «ПЗ12\_Рыбалев», в ней была создана таблица «Страны». Таблица была заполнена соответствующими данными.
2. Были выполнены следующие примеры:

Пример 1: Напишите функцию для вывода столицы данной страны, и вызовите ее. Выполнение примера представлено на рисунке 12.1.



Рисунок 12.1 – Выполнение примера 1

Пример 2: Напишите функцию для перевода площади в тыс. кв. км., и вызовите ее. Выполнение примера представлено на рисунке 12.2.

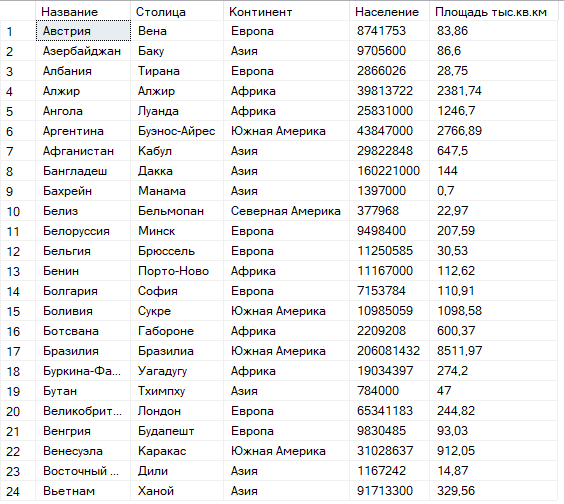


Рисунок 12.2 – Выполнение примера 2

Пример 3: Напишите функцию для вычисления плотности населения, и вызовите ее. Выполнение примера представлено на рисунке 12.3.

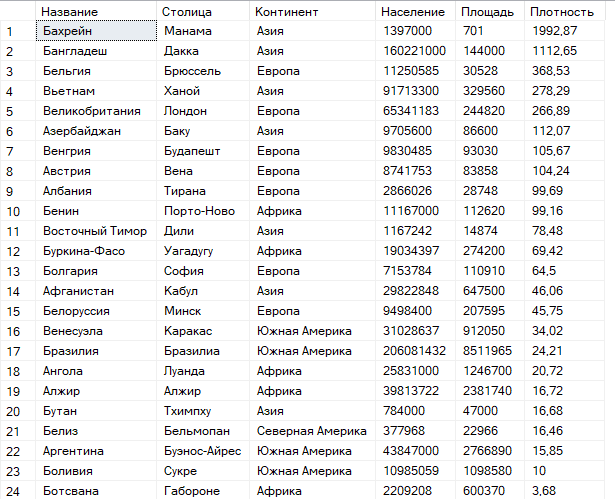


Рисунок 12.3 – Выполнение примера 3

Пример 4: Напишите функцию для поиска страны второй по площади, и вызовите ее. Выполнение примера представлено на рисунке 12.4.



Рисунок 12.4 – Выполнение примера 4

Пример 5: Напишите функцию для поиска страны с минимальной площадью в заданной части света, и вызовите ее. Если часть света не указана, выбрать Европу. Выполнение примера представлено на рисунке 12.5.

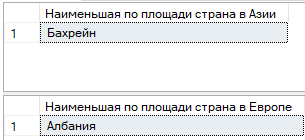


Рисунок 12.5 – Выполнение примера 5

Пример 6: Напишите функцию для замены букв в заданном слове от второй до предпоследней на точку, и примените ее для названия страны. Выполнение примера представлено на рисунке 12.6.



Рисунок 12.6 – Выполнение примера 6

Пример 7: Напишите функцию, которая возвращает количество стран, содержащих в названии заданную букву. Выполнение примера представлено на рисунке 12.7.

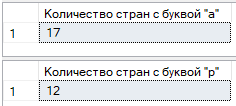


Рисунок 12.7 – Выполнение примера 7

Пример 8: Напишите функцию для вывода списка стран с населением больше заданного числа, и вызовите ее. Выполнение примера представлено на рисунке 12.8.



Рисунок 12.8 – Выполнение примера 8

Пример 9: Напишите функцию для вывода списка стран с площадью в интервале заданных значений, и вызовите ее. В результате выполнения примера запрос ни выдал искомых данных.

Пример 10: Напишите функцию для возврата таблицы с названием страны и плотностью населения, и вызовите ее. Выполнение примера представлено на рисунке 12.9.

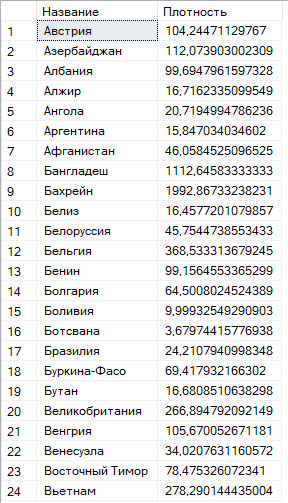


Рисунок 12.9 – Выполнение примера 10

Пример 11: удалите функцию из примера 10. В результате выполнения примера была удалена функция из примера 10.

**Рефлексия:** Я изучил: скалярные функций; функции INLINE; функции MULTI-STATEMENT; удаления пользовательских функций.

**Инструкционная карта №13**

**Тема: «Хранимые процедуры. Создание, передача входных и выходных параметров, вызов и удаление хранимых процедур».**

**Цель работы:**

1. Изучение создания хранимых процедур.
2. Изучение передачи входных параметров.
3. Изучение передачи выходных параметров.
4. Изучение вызовов хранимых процедур.
5. Изучение удаления хранимых процедур.

**Ход работы:**

1. Была создана база данных «ПЗ13\_Рыбалев», в ней была создана таблица «Страны». Таблица была заполнена соответствующими данными.
2. Были выполнены следующие примеры:

Пример 1: Напишите хранимую процедуру для вывода информации о сервере, о базе данных и о текущем пользователе, и вызовите ее. Выполнение примера представлено на рисунке 13.1.



Рисунок 13.1 – Выполнение примера 1

Пример 2: Напишите хранимую процедуру, которая выводит названия и столицы всех стран. Выполнение примера представлено на рисунке 13.2.

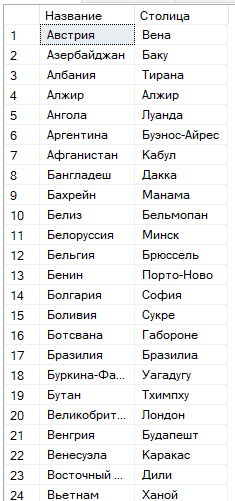


Рисунок 13.2 – Выполнение примера 2

Пример 3: Напишите хранимую процедуру, которая выводит список стран заданной части света, и вызовите ее. Выполнение примера представлено на рисунке 13.3.

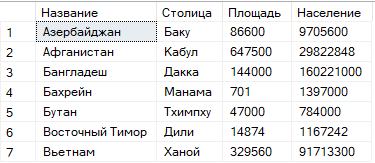


Рисунок 13.3 – Выполнение примера 3

Пример 4: Напишите хранимую процедуру, которая выводит список стран, площадь которых находится в заданном интервале, и вызовите ее. В результате выполнения примера запрос не выдал искомых данных.

Пример 5: Напишите хранимую процедуру, которая возвращает количество стран, содержащих в названии заданную букву, и вызовите ее. Выполнение примера представлено на рисунке 13.4.



Рисунок 13.4 – Выполнение примера 5

Пример 6: Напишите хранимую процедуру для вывода трех стран с наименьшей площадью в заданной части света, и вызовите ее. Если часть света не указана, выбрать Европу. Выполнение примера представлено на рисунке 13.5.

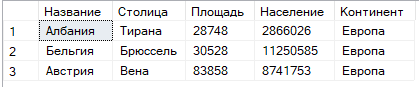


Рисунок 13.5 – Выполнение примера 6

Пример 7: Напишите хранимую процедуру, которая создает таблицу «Страны\_У», и заполняет ее странами, названия которых начинаются на букву «У». В результате выполнения примера была создана таблица «Страны\_У», но стран, начинающихся с буквы «У» нету в изначальной таблице.

Пример 8: Напишите хранимую процедуру, которая удаляет таблицу «Страны\_У» и возвращает количество строк. Выполнение примера представлено на рисунке 13.6.



Рисунок 13.8 – Выполнение примера 6

Пример 9: Напишите код, который удаляет хранимую процедуру «Пример8». В результате выполнения примера была удалена хранимая процедура из примера 8.

**Самостоятельное задание**

Были выполнены следующие задания:

1. Напишите хранимую процедуру для вывода информации о сервере, о базе данных, о текущем пользователе, о текущем времени, и вызовите ее.

Код выполнения задания представлен ниже:

CREATE PROC сам.раб1

AS

BEGIN

SELECT

@@SERVERNAME AS [Имя сервера],

@@VERSION AS [Версия СУБД],

DB\_NAME() AS [Текущая база данных],

CURRENT\_USER AS [Текущий пользователь],

GETDATE() AS [Текущее время];

END

EXEC сам.раб1;

Выполнение задания представлено на рисунке 13.7.



Рисунок 13.7 – Выполнение задания

1. Напишите хранимую процедуру, которая выводит данные всех стран.

Код выполнения задания представлен ниже:

CREATE PROC Proc2\_ВсеСтраны

AS

BEGIN

SELECT \* FROM Страны ORDER BY Название;

END

EXEC Proc2\_ВсеСтраны;

Выполнение задания представлено на рисунке 13.8.

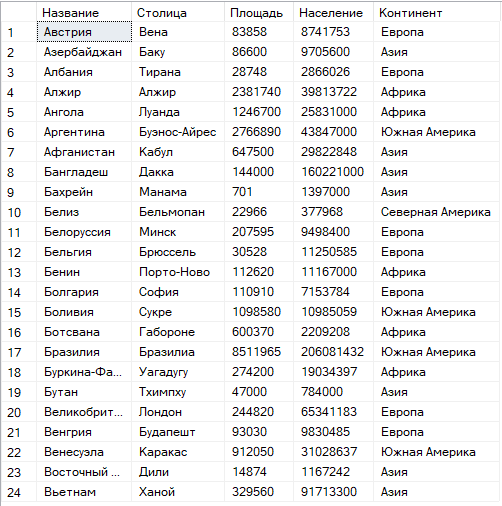


Рисунок 13.8 – Выполнение задания

1. Напишите хранимую процедуру, которая выводит список стран, кроме заданной части света, и вызовите ее.

Код выполнения задания представлен ниже:

CREATE PROC Proc3\_СтраныКроме

@континент NVARCHAR(50) = N'Европа'

AS

BEGIN

SELECT \* FROM Страны

WHERE Континент <> @континент

ORDER BY Континент, Название;

END

EXEC Proc3\_СтраныКроме N'Азия';

Выполнение задания представлено на рисунке 13.9.

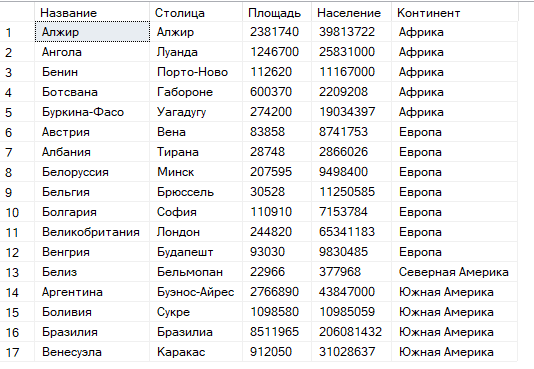


Рисунок 13.9 – Выполнение задания

1. Напишите хранимую процедуру, которая выводит список стран, население которых находится в заданном интервале, и вызовите ее.

Код выполнения задания представлен ниже:

CREATE PROC Proc4\_СтраныПоНаселению

@от INT = 1000000,

@до INT = 10000000

AS

BEGIN

SELECT \* FROM Страны

WHERE Население BETWEEN @от AND @до

ORDER BY Население DESC;

END

EXEC Proc4\_СтраныПоНаселению 5000000, 20000000;

Выполнение задания представлено на рисунке 13.10.



Рисунок 13.10 – Выполнение задания

1. Напишите хранимую процедуру, которая возвращает количество стран, у которых в названии отсутствует заданная буква, и вызовите ее.

Код выполнения задания представлен ниже:

CREATE PROC Proc5\_БезБуквы

@буква NCHAR(1) = N'а',

@количество INT OUTPUT

AS

BEGIN

SELECT @количество = COUNT(\*)

FROM Страны

WHERE CHARINDEX(@буква, Название) = 0;

END

DECLARE @к INT;

EXEC Proc5\_БезБуквы N'а', @к OUTPUT;

SELECT @к AS [Стран без буквы "а"];

Выполнение задания представлено на рисунке 13.11.



Рисунок 13.11 – Выполнение задания

1. Напишите хранимую процедуру для вывода пяти стран с наибольшим населением в заданной части света, и вызовите ее. Если часть света не указана, выбрать Африку.

Код выполнения задания представлен ниже:

CREATE PROC Proc6\_Топ5ПоНаселению

@континент NVARCHAR(50) = N'Африка'

AS

BEGIN

SELECT TOP 5 \*

FROM Страны

WHERE Континент = @континент

ORDER BY Население DESC;

END

EXEC Proc6\_Топ5ПоНаселению N'Европа';

EXEC Proc6\_Топ5ПоНаселению DEFAULT;

Выполнение задания представлено на рисунке 13.12.



Рисунок 13.12 – Выполнение задания

1. Напишите хранимую процедуру, которая создает таблицу «Страны\_<первая буква вашей фамилии>», и заполняет ее странами, названия которых начинаются с первой буквой вашей фамилии.

Код выполнения задания представлен ниже:

CREATE PROC Proc7\_СоздатьТаблицуНаБукву

@буква NCHAR(1) = N'И'

AS

BEGIN

DECLARE @имя\_таблицы NVARCHAR(50) = N'Страны\_' + @буква;

DECLARE @sql NVARCHAR(MAX) = '

IF OBJECT\_ID(''' + @имя\_таблицы + ''', ''U'') IS NOT NULL

DROP TABLE ' + @имя\_таблицы + ';

SELECT \*

INTO ' + @имя\_таблицы + '

FROM Страны

WHERE LEFT(Название, 1) = N''' + @буква + '''';

EXEC sp\_executesql @sql;

END

EXEC Proc7\_СоздатьТаблицуНаБукву N'И';

Выполнение задания представлено на рисунке 13.13.



Рисунок 13.13 – Выполнение задания

1. Напишите хранимую процедуру, которая удаляет таблицу, которую вы создали в предыдущем задании и возвращает количество удаленных строк.

Код выполнения задания представлен ниже:

CREATE PROC Proc8\_УдалитьТаблицуНаБукву

@буква NCHAR(1) = N'И',

@удалено INT OUTPUT

AS

BEGIN

DECLARE @имя\_таблицы NVARCHAR(50) = N'Страны\_' + @буква;

SET @удалено = 0;

IF OBJECT\_ID(@имя\_таблицы, 'U') IS NOT NULL

BEGIN

SELECT @удалено = COUNT(\*) FROM Страны\_И;

DECLARE @sql NVARCHAR(MAX) = 'DROP TABLE ' + @имя\_таблицы;

EXEC sp\_executesql @sql;

END

END

DECLARE @у INT;

EXEC Proc8\_УдалитьТаблицуНаБукву N'И', @у OUTPUT;

SELECT @у AS [Удалено строк];

Выполнение задания представлено на рисунке 13.14.



Рисунок 13.14 – Выполнение задания

1. Напишите хранимую процедуру, принимающую число и возвращающую количество цифр в нем через параметр OUTPUT.

Код выполнения задания представлен ниже:

CREATE PROC Proc9\_КоличествоЦифр

@число INT,

@количество INT OUTPUT

AS

BEGIN

DECLARE @строка NVARCHAR(20) = CAST(ABS(@число) AS NVARCHAR(20));

SET @количество = LEN(@строка);

END

DECLARE @к9 INT;

EXEC Proc9\_КоличествоЦифр 12345, @к9 OUTPUT;

SELECT @к9 AS [Количество цифр];

Выполнение задания представлено на рисунке 13.15.



Рисунок 13.15 – Выполнение задания

1. Напишите хранимую процедуру AddRightDigit, добавляющую к целому положительному числу K справа цифру D (D – входной параметр целого типа, лежащий в диапазоне [0..9], K – параметр целого типа, являющийся одновременно входным и выходным).

Код выполнения задания представлен ниже:

CREATE PROC Proc10\_AddRightDigit

@K INT OUTPUT,

@D INT

AS

BEGIN

IF @D BETWEEN 0 AND 9

SET @K = @K \* 10 + @D;

END

DECLARE @число10 INT = 123;

EXEC Proc10\_AddRightDigit @число10 OUTPUT, 4;

SELECT @число10 AS [Число после добавления цифры];

Выполнение задания представлено на рисунке 13.16.



Рисунок 13.16 – Выполнение задания

1. Напишите хранимую процедуру InvDigit, меняющую порядок следования цифр целого положительного числа K на обратный (K – параметр целого типа, являющийся одновременно входным и выходным).

Код выполнения задания представлен ниже:

CREATE PROC Proc11\_InvDigit

@K INT OUTPUT

AS

BEGIN

DECLARE @результат INT = 0;

DECLARE @остаток INT;

WHILE @K > 0

BEGIN

SET @остаток = @K % 10;

SET @результат = @результат \* 10 + @остаток;

SET @K = @K / 10;

END

SET @K = @результат;

END

DECLARE @число11 INT = 12345;

EXEC Proc11\_InvDigit @число11 OUTPUT;

SELECT @число11 AS [Число в обратном порядке];

Выполнение задания представлено на рисунке 13.17.



Рисунок 13.17 – Выполнение задания

1. Напишите хранимую процедуру Swap, меняющую содержимое переменных X и Y (X и Y – вещественные параметры, являющиеся одновременно входными и выходными).

Код выполнения задания представлен ниже:

CREATE PROC Proc12\_Swap

@X FLOAT OUTPUT,

@Y FLOAT OUTPUT

AS

BEGIN

DECLARE @temp FLOAT = @X;

SET @X = @Y;

SET @Y = @temp;

END

DECLARE @a FLOAT = 10.5, @b FLOAT = 20.3;

EXEC Proc12\_Swap @a OUTPUT, @b OUTPUT;

SELECT @a AS A, @b AS B;

Выполнение задания представлено на рисунке 13.18.



Рисунок 13.18 – Выполнение задания

1. Напишите хранимую процедуру SortInc, меняющую содержимое переменных A, B, C, таким образом, чтобы их значения оказались упорядоченными по возрастанию (A, B, C – вещественные параметры, являющиеся одновременно входными и выходными).

Код выполнения задания представлен ниже:

CREATE PROC Proc13\_SortInc

@A FLOAT OUTPUT,

@B FLOAT OUTPUT,

@C FLOAT OUTPUT

AS

BEGIN

DECLARE @min FLOAT, @mid FLOAT, @max FLOAT;

SET @min = CASE

WHEN @A <= @B AND @A <= @C THEN @A

WHEN @B <= @A AND @B <= @C THEN @B

ELSE @C

END;

SET @max = CASE

WHEN @A >= @B AND @A >= @C THEN @A

WHEN @B >= @A AND @B >= @C THEN @B

ELSE @C

END;

-- Находим среднее

SET @mid = CASE

WHEN (@A = @min AND @B = @max) OR (@A = @max AND @B = @min) THEN @C

WHEN (@A = @min AND @C = @max) OR (@A = @max AND @C = @min) THEN @B

ELSE @A

END;

SET @A = @min;

SET @B = @mid;

SET @C = @max;

END

DECLARE @x FLOAT = 30, @y FLOAT = 10, @z FLOAT = 20;

EXEC Proc13\_SortInc @x OUTPUT, @y OUTPUT, @z OUTPUT;

SELECT @x AS Min, @y AS Mid, @z AS Max;

Выполнение задания представлено на рисунке 13.19.



Рисунок 13.19 – Выполнение задания

1. Напишите хранимую процедуру DigitCountSum, находящую количество C цифр целого положительного числа K, а также их сумму S (K – входной, C, S – выходные параметры целого типа).

Код выполнения задания представлен ниже:

CREATE PROC Proc14\_DigitCountSum

@K INT,

@C INT OUTPUT,

@S INT OUTPUT

AS

BEGIN

SET @C = 0;

SET @S = 0;

DECLARE @temp INT = ABS(@K);

WHILE @temp > 0

BEGIN

SET @S = @S + (@temp % 10);

SET @C = @C + 1;

SET @temp = @temp / 10;

END

END

DECLARE @число14 INT = 12345, @количество INT, @сумма INT;

EXEC Proc14\_DigitCountSum @число14, @количество OUTPUT, @сумма OUTPUT;

SELECT @количество AS [Количество цифр], @сумма AS [Сумма цифр];

Выполнение задания представлено на рисунке 13.20.



Рисунок 13.20 – Выполнение задания

1. Напишите код, который удаляет все хранимые процедуры, вами созданные.

Код выполнения задания представлен ниже:

DECLARE @proc\_name NVARCHAR(100);

DECLARE cur CURSOR FOR

SELECT name FROM sys.objects

WHERE type = 'P' AND name LIKE 'Proc%';

OPEN cur;

FETCH NEXT FROM cur INTO @proc\_name;

WHILE @@FETCH\_STATUS = 0

BEGIN

DECLARE @drop\_sql NVARCHAR(200) = 'DROP PROC ' + @proc\_name;

EXEC sp\_executesql @drop\_sql;

FETCH NEXT FROM cur INTO @proc\_name;

END

CLOSE cur;

DEALLOCATE cur;

PRINT 'Все процедуры удалены';

В результате выполнения задания были удалены все ранее созданные процедуры.

**Рефлексия:** Я изучил: создание хранимых процедур; передачу входных параметров; передачу выходных параметров; вызовов хранимых процедур; удаление хранимых процедур.

**Инструкционная карта №14**

**Тема: «Триггеры. Создание, включение, виртуальные таблицы, приостановка, удаление триггеров».**

**Цель работы:**

1. Изучить создание триггеров.
2. Изучить триггеры после событий.
3. Изучить триггеры вместо событий.
4. Изучить виртуальные таблицы в триггерах.
5. Изучить приостановление триггеров.
6. Изучить удаление триггеров.

**Ход работы:**

1. Была создана база данных «ПЗ14\_Рыбалев», в ней была создана таблица «Ученики». Таблица была заполнена соответствующими данными.
2. Были выполнены следующие примеры:

Пример 1: Напишите триггер на добавление записи в таблицу «Ученики». Данный триггер, в случае успешного добавления данных, выводит «Запись добавлена». Выполнение примера представлено на рисунке 14.1.

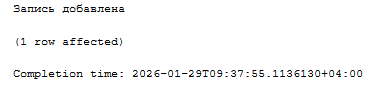


Рисунок 14.1 – Выполнение примера 1

Пример 2: Напишите триггер на удаление записи из таблицы «Ученики». Данный триггер, в случае успешного удаления данных, выводит «Запись удалена». Выполнение примера представлено на рисунке 14.2.

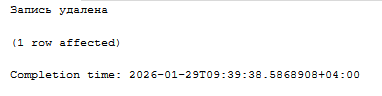


Рисунок 14.2 – Выполнение примера 2

Пример 3: Напишите триггер на добавление, изменение и удаление данных для таблицы «Ученики». Данный триггер выводит «Таблица изменена». Выполнение примера представлено на рисунке 14.3.

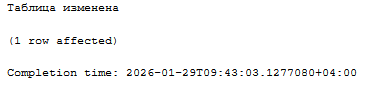


Рисунок 14.3 – Выполнение примера 3

Пример 4: Напишите триггер на удаление записи из таблицы «Ученики». Данный триггер, при попытке удаления данных, выводит «Нельзя удалить данные». Выполнение примера представлено на рисунке 14.4.

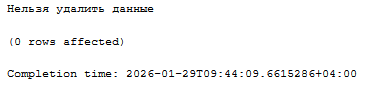


Рисунок 14.4 – Выполнение примера 4

Пример 5: создать таблицу «Ученики\_Архив», которая будет содержать все данные об удаленных учениках и даты их удаления. Написать триггер, который будет фиксировать в таблице «Ученики\_Архив» данные ученика, удаленного из таблицы «Ученики». Выполнение примера представлено на рисунке 14.5.

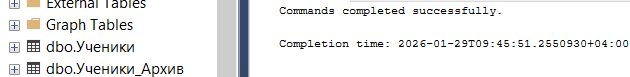


Рисунок 14.5 – Выполнение примера 5

Пример 6: Напишите команды для приостановления и запуска триггера из примера 5. Выполнение примера представлено на рисунке 14.6.

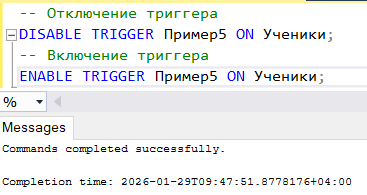


Рисунок 14.6 – Выполнение примера 6

Пример 7: Напишите команды для удаления триггера из примера 5. Выполнение примера представлено на рисунке 14.7.

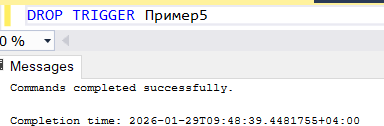


Рисунок 14.7 – Выполнение примера 7

**Самостоятельное задание**

Были выполнены следующие задания:

1. Напишите триггер на изменение записи в таблице «Ученики». Данный триггер, в случае изменения данных, должен вывести «Запись изменена».

Код выполнения задания представлен ниже:

CREATE TRIGGER Триггер\_Изменение ON Ученики

AFTER UPDATE

AS

BEGIN

PRINT 'Запись изменена'

END

Выполнение задания представлено на рисунке 14.8.

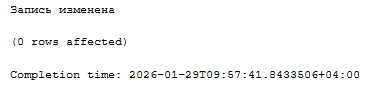


Рисунок 14.8 – Выполнение задания

1. Напишите триггер на добавление и удаление записи из таблицы «Ученики». Данный триггер, в случае успешного добавления или удаления данных, должен вывести «Количество строк изменено».

Код выполнения задания представлен ниже:

CREATE TRIGGER Триггер\_Добавление\_Удаление ON Ученики

FOR INSERT, DELETE

AS

BEGIN

DECLARE @count INT

IF EXISTS(SELECT \* FROM inserted)

SELECT @count = COUNT(\*) FROM inserted

ELSE

SELECT @count = COUNT(\*) FROM deleted

PRINT 'Количество строк изменено: ' + CAST(@count AS NVARCHAR(10))

END

Выполнение задания представлено на рисунке 14.9.



Рисунок 14.9 – Выполнение задания

1. Напишите триггер на добавление, изменение и удаление данных в таблице «Ученики». Данный триггер должен вывести «{Текущий пользователь} изменил таблицу. Время.

Код выполнения задания представлен ниже:

CREATE TRIGGER Триггер\_Все\_Операции ON Ученики

FOR INSERT, UPDATE, DELETE

AS

BEGIN

PRINT SUSER\_NAME() + ' изменил таблицу. Время: ' + CONVERT(VARCHAR, GETDATE(), 120)

END

Выполнение задания представлено на рисунке 14.10.

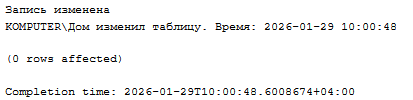


Рисунок 14.10 – Выполнение задания

1. Напишите триггер на изменение записи в таблице «Ученики». Данный триггер, при попытке изменения данных, должен вывести «Нельзя редактировать данные».

Код выполнения задания представлен ниже:

CREATE TRIGGER Триггер\_Запрет\_Изменения ON Ученики

INSTEAD OF UPDATE

AS

BEGIN

PRINT 'Нельзя редактировать данные'

END

Выполнение задания представлено на рисунке 14.11.



Рисунок 14.11 – Выполнение задания

1. Создать таблицу «Ученики\_{Ваша\_фамилия}», которая будет содержать фамилии удаленных учеников и даты их удаления. Написать триггер, который будет фиксировать в таблице «Ученики\_{Ваша\_фамилия}» данные учеников при удалении из таблицы «Ученики», в том случае, если у них остались однофамильцы в таблице «Ученики».

Код выполнения задания представлен ниже:

CREATE TABLE Ученики\_Рыбалев (

ID\_Удаленного INT NOT NULL,

Фамилия NVARCHAR(50) NOT NULL,

Дата\_Удаления DATETIME NOT NULL DEFAULT GETDATE()

)

CREATE TRIGGER Триггер\_Архив\_Однофамильцы ON Ученики

FOR DELETE

AS

BEGIN

INSERT INTO Ученики\_Рыбалев (ID\_Удаленного, Фамилия)

SELECT d.ID, d.Фамилия

FROM deleted d

WHERE EXISTS (

SELECT 1 FROM Ученики u

WHERE u.Фамилия = d.Фамилия

AND u.ID != d.ID

)

END

Выполнение задания представлено на рисунке 14.12.



Рисунок 14.12 – Выполнение задания

1. Напишите команды для приостановления и запуска триггера из предыдущей задачи.

Код выполнения задания представлен ниже:

-- Отключение

DISABLE TRIGGER Триггер\_Архив\_Однофамильцы ON Ученики

GO

ENABLE TRIGGER Триггер\_Архив\_Однофамильцы ON Ученики

В результате выполнения запросов триггер активировался и отключался.

1. Напишите команды для удаления всех созданных вами триггеров.

Код выполнения задания представлен ниже:

DROP TRIGGER IF EXISTS Триггер\_Изменение

DROP TRIGGER IF EXISTS Триггер\_Добавление\_Удаление

DROP TRIGGER IF EXISTS Триггер\_Все\_Операции

DROP TRIGGER IF EXISTS Триггер\_Запрет\_Изменения

DROP TRIGGER IF EXISTS Триггер\_Архив\_Однофамильцы

В результате выполнения запроса все ранее созданные триггеры были удалены.

**Рефлексия:** Я изучил: создание триггеров; триггеры после событий; триггеры вместо событий; виртуальные таблицы в триггерах; приостановление триггеров; удаление триггеров.

**Инструкционная карта №15**

**Тема: «Представления и табличные объекты. Создание и удаление представлений».**

**Цель работы:**

1. Изучить создание и удаление представлений.
2. Изучить табличные переменные.
3. Изучить временные таблицы.
4. Изучить производные таблицы.

**Ход работы:**

1. Были выполнены следующие примеры:

Пример 1: Создайте представление, содержащее список стран, население которых меньше 1 млн. чел., а площадь больше 100 тыс. кв. км, и используйте его. В результате выполнения примера было создано представление, но запрашиваемых данных не было.

Пример 2: Создайте представление, содержащее список континентов, суммарную площадь и суммарное население стран, которые находятся на каждом континенте и используйте его. Выполнение примера представлено на рисунке 15.1.

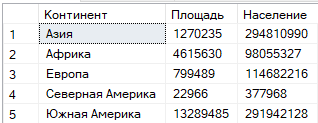


Рисунок 15.1 – Выполнение примера 2

Пример 3: Создайте представление, содержащее фамилии преподавателей, должность, каждого преподавателя, звание, степень, место работы, зарплату и используйте его. В результате выполнения примера таких данных не было найдено.

Пример 4: Создайте табличную переменную, содержащую три столбца («Номер недели», «Дата начала», «Дата конца»). Заполните ее для текущего года и используйте. Выполнение примера представлено на рисунке 15.2.

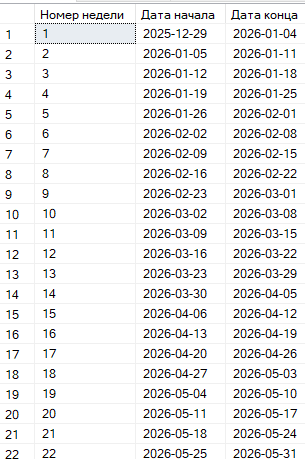


Рисунок 15.2 – Выполнение примера 4

Пример 5: Создайте табличную переменную, содержащую список стран, площадь которых в 1000 раз меньше, чем средняя площадь стран в мире и используйте. Выполнение примера представлено на рисунке 15.3.



Рисунок 15.3 – Выполнение примера 5

Пример 6: Создайте локальную временную таблицу, имеющую три столбца («Название месяца», «Количество экзаменов», «Количество студентов»), заполните и используйте ее. В результате выполнения примера таких данных не было найдено.

Пример 7: Создайте глобальную временную таблицу, содержащую название стран и плотность их населения, заполните и используйте ее. Выполнение примера представлено на рисунке 15.4.



Рисунок 15.4 – Выполнение примера 7

Пример 8: с помощью обобщенных табличных выражений, напишите запрос для вывода списка сотрудников, чьи зарплаты меньше, чем средняя зарплата по кафедре, их зарплаты и название кафедры. В результате выполнения примера таких данных не было найдено.

**Самостоятельное задание**

Были выполнены следующие задания:

1. Создайте представление, содержащее список африканских стран, население которых больше 10 млн. чел., а площадь больше 500 тыс. кв. км, и используйте его.

Код выполнения задания представлен ниже:

CREATE VIEW vw\_КрупныеАфриканскиеСтраны

AS

SELECT

Название,

Столица,

Площадь,

Население,

Континент

FROM Страны

WHERE Континент = N'Африка'

AND Население > 10000000

AND Площадь > 500000;

SELECT \* FROM vw\_КрупныеАфриканскиеСтраны;

Выполнение задания представлено на рисунке 15.5.

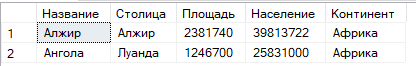


Рисунок 15.5 – Выполнение задания

1. Создайте представление, содержащее список континентов, среднюю площадь стран, которые находятся на нем, среднюю плотность населения, и используйте его.

Код выполнения задания представлен ниже:

CREATE VIEW vw\_СтатистикаПоКонтинентам

AS

SELECT

Континент,

COUNT(\*) AS [Количество стран],

AVG(Площадь \* 1.0) AS [Средняя площадь],

AVG(Население \* 1.0 / NULLIF(Площадь, 0)) AS [Средняя плотность населения]

FROM Страны

GROUP BY Континент;

SELECT \* FROM vw\_СтатистикаПоКонтинентам

ORDER BY [Средняя плотность населения] DESC;

Выполнение задания представлено на рисунке 15.6.

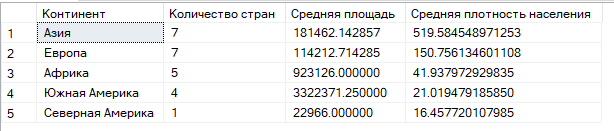


Рисунок 15.6 – Выполнение задания

1. Создайте представление, содержащее фамилии преподавателей, их должность, звание, степень, место работы, количество их экзаменов, и используйте его.

Код выполнения задания представлен ниже:

CREATE VIEW vw\_ПреподавателиСЭкзаменами

AS

SELECT

п.Фамилия,

п.Должность,

п.Звание,

п.Степень,

п.Кафедра AS [Место работы],

COUNT(э.ID) AS [Количество экзаменов]

FROM Преподаватели п

LEFT JOIN Экзамены э ON п.ID = э.ПреподавательID

GROUP BY п.ID, п.Фамилия, п.Должность, п.Звание, п.Степень, п.Кафедра;

SELECT \* FROM vw\_ПреподавателиСЭкзаменами;

Выполнение задания представлено на рисунке 15.7.

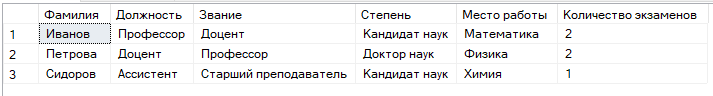


Рисунок 15.7 – Выполнение задания

1. Создайте табличную переменную, содержащую три столбца («Номер месяца», «Название месяца», «Количество дней»), заполните ее для текущего года, и используйте ее.

Код выполнения задания представлен ниже:

DECLARE @Месяцы TABLE

(

[Номер месяца] INT,

[Название месяца] NVARCHAR(20),

[Количество дней] INT

);

DECLARE @год INT = YEAR(GETDATE());

DECLARE @месяц INT = 1;

WHILE @месяц <= 12

BEGIN

INSERT INTO @Месяцы

VALUES (

@месяц,

DATENAME(MONTH, DATEFROMPARTS(@год, @месяц, 1)),

DAY(EOMONTH(DATEFROMPARTS(@год, @месяц, 1)))

);

SET @месяц = @месяц + 1;

END;

SELECT \* FROM @Месяцы

ORDER BY [Номер месяца];

Выполнение задания представлено на рисунке 15.8.



Рисунок 15.8 – Выполнение задания

1. Создайте табличную переменную, содержащую список стран, площадь которых в 100 раз меньше, чем средняя площадь стран на континенте, где они находятся, и используйте ее.

Код выполнения задания представлен ниже:

DECLARE @МаленькиеСтраны TABLE

(

Название NVARCHAR(100),

Континент NVARCHAR(50),

Площадь INT,

[Средняя площадь по континенту] INT

);

WITH СредниеПоКонтинентам AS (

SELECT

Континент,

AVG(Площадь \* 1.0) AS СредняяПлощадь

FROM Страны

GROUP BY Континент

)

INSERT INTO @МаленькиеСтраны

SELECT

с.Название,

с.Континент,

с.Площадь,

сп.СредняяПлощадь

FROM Страны с

INNER JOIN СредниеПоКонтинентам сп ON с.Континент = сп.Континент

WHERE с.Площадь < сп.СредняяПлощадь / 100;

SELECT \* FROM @МаленькиеСтраны;

Выполнение задания представлено на рисунке 15.9.



Рисунок 15.9 – Выполнение задания

1. Создайте локальную временную таблицу, имеющую три столбца («Номер недели», «Количество экзаменов», «Количество студентов»), заполните и используйте ее.

Код выполнения задания представлен ниже:

CREATE TABLE #ЭкзаменационнаяСтатистика

(

[Номер недели] INT,

[Количество экзаменов] INT,

[Количество студентов] INT

);

DECLARE @неделя INT = 1;

WHILE @неделя <= 52

BEGIN

INSERT INTO #ЭкзаменационнаяСтатистика

VALUES (

@неделя,

ABS(CHECKSUM(NEWID())) % 20 + 1, -- Случайное количество экзаменов 1-20

ABS(CHECKSUM(NEWID())) % 500 + 50 -- Случайное количество студентов 50-550

);

SET @неделя = @неделя + 1;

END;

SELECT \* FROM #ЭкзаменационнаяСтатистика

ORDER BY [Номер недели];

Выполнение задания представлено на рисунке 15.10.

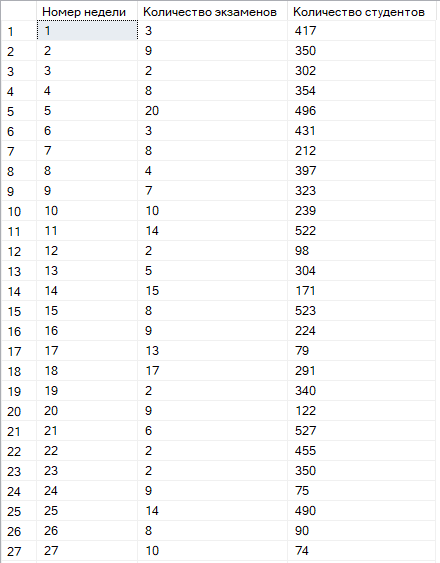


Рисунок 15.10 – Выполнение задания

1. Создайте глобальную временную таблицу, содержащую название континентов, наибольшую и наименьшую площадь стран на них, заполните и используйте ее.

Код выполнения задания представлен ниже:

CREATE TABLE ##СтатистикаКонтинентов

(

Континент NVARCHAR(50),

[Наибольшая площадь] INT,

[Наименьшая площадь] INT

);

INSERT INTO ##СтатистикаКонтинентов

SELECT

Континент,

MAX(Площадь) AS [Наибольшая площадь],

MIN(Площадь) AS [Наименьшая площадь]

FROM Страны

GROUP BY Континент;

SELECT \* FROM ##СтатистикаКонтинентов

ORDER BY [Наибольшая площадь] DESC;

Выполнение задания представлено на рисунке 15.11.

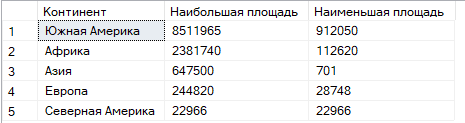


Рисунок 15.11 – Выполнение задания

1. С помощью обобщенных табличных выражений напишите запрос для вывода списка сотрудников, чьи зарплаты меньше, чем средняя зарплата по факультету, их зарплаты и название факультета.

Код выполнения задания представлен ниже:

WITH СредняяЗарплатаПоФакультету AS (

SELECT

ФакультетID,

AVG(Зарплата) AS СредняяЗарплата

FROM Сотрудники

GROUP BY ФакультетID

)

SELECT

с.Фамилия,

с.Зарплата,

ф.Название AS Факультет,

сз.СредняяЗарплата AS [Средняя зарплата по факультету]

FROM Сотрудники с

INNER JOIN Факультеты ф ON с.ФакультетID = ф.ID

INNER JOIN СредняяЗарплатаПоФакультету сз ON с.ФакультетID = сз.ФакультетID

WHERE с.Зарплата < сз.СредняяЗарплата

ORDER BY ф.Название, с.Фамилия;

Выполнение задания представлено на рисунке 15.12.

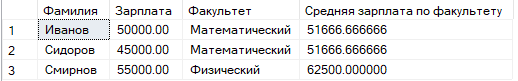


Рисунок 15.12 – Выполнение задания

1. Напишите команды для удаления всех созданных вами представлений.

Код выполнения задания представлен ниже:

DROP VIEW IF EXISTS vw\_КрупныеАфриканскиеСтраны;

DROP VIEW IF EXISTS vw\_СтатистикаПоКонтинентам;

DROP VIEW IF EXISTS vw\_ПреподавателиСЭкзаменами;

В результате выполнения задания были удалены все ранее созданные представлния.

**Рефлексия:** Я изучил: создание и удаление представлений; табличные переменные; временные таблицы; производные таблицы.

**Инструкционная карта №16**

**Тема: «Курсоры. Использование курсоров».**

**Цель работы:**

1. Изучить использование курсоров.
2. Изучить прокрутку курсоров.
3. Изучить использование циклов в курсорах.
4. Изучить использование переменных в курсорах.

**Ход работы:**

1. Была создана база данных «ПЗ16\_Рыбалев», в ней была создана таблица «Ученики» и заполнена соответствующими данными.
2. Были выполнены следующие примеры:

Пример 1: Создайте курсор, содержащий отсортированные по алфавиту фамилии учеников и названия их предметов, откройте его, выведите первую строку, закройте и освободите курсор. Выполнение примера представлено на рисунке 16.1.



Рисунок 16.1 – Выполнение примера 1

Пример 2: Создайте курсор с прокруткой, содержащий список учеников, откройте его, выведите пятую, предыдущую, с конца четвертую, шесть позиций назад находящуюся, четыре позиций вперед находящуюся, следующую, первую, последнюю строку, закройте и освободите курсор. Выполнение примера представлено на рисунке 16.2.

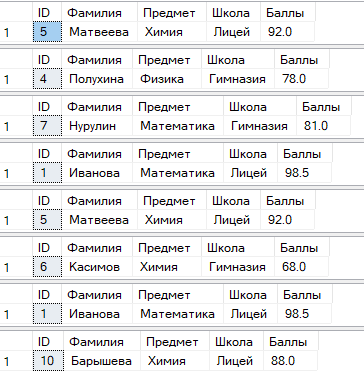


Рисунок 16.2 – Выполнение примера 2

Пример 3: с помощью курсора, вычислите среднее арифметическое значение балла у учеников с наибольшим и наименьшим баллом. Выполнение примера представлено на рисунке 16.3.



Рисунок 16.3 – Выполнение примера 3

Пример 4: с помощью курсора, сгенерируйте строку вида «Ученики <список фамилий и названий школ, разделенных запятыми>, участвовали в олимпиаде». Выполнение примера представлено на рисунке 16.4.

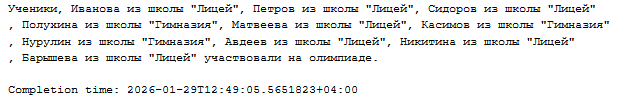


Рисунок 16.4 – Выполнение примера 4

Пример 5: Создайте курсор, содержащий список учеников, с его помощью выведите учеников с четной позицией. Выполнение примера представлено на рисунке 16.5.

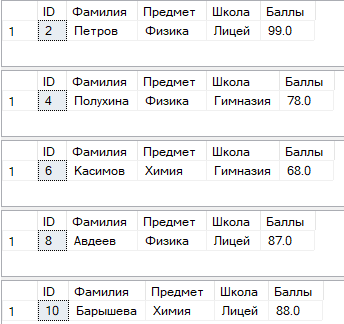


Рисунок 16.5 – Выполнение примера 5

Пример 6: Создайте курсор, содержащий отсортированный по убыванию баллов список учеников, откройте его, для каждого ученика выведите фамилию, предмет, школу, баллы и разницу баллов с предыдущим учеником. Выполнение примера представлено на рисунке 16.6.



Рисунок 16.6 – Выполнение примера 6

**Самостоятельное задание**

Были выполнены следующие задания:

1. Создайте курсор, содержащий отсортированные по баллам фамилии и баллы учеников, откройте его, выведите первую строку, закройте и освободите курсор.

Код выполнения задания представлен ниже:

DECLARE Курсор1 CURSOR FOR

SELECT Фамилия, Баллы

FROM Ученики

ORDER BY Баллы DESC;

DECLARE @Фамилия1 NVARCHAR(50), @Баллы1 DECIMAL(4,1);

OPEN Курсор1;

FETCH NEXT FROM Курсор1 INTO @Фамилия1, @Баллы1;

PRINT 'Первая строка: ' + @Фамилия1 + ' - ' + CAST(@Баллы1 AS NVARCHAR(10));

CLOSE Курсор1;

DEALLOCATE Курсор1;

Выполнение задания представлено на рисунке 16.7.

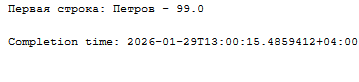


Рисунок 16.7 – Выполнение задания

1. Создайте курсор с прокруткой, содержащий список учеников, откройте его, выведите пятую, предыдущую, с конца четвертую, следующую, первую строку, закройте и освободите курсор.

Код выполнения задания представлен ниже:

DECLARE Курсор2 CURSOR SCROLL FOR

SELECT ID, Фамилия, Школа, Баллы

FROM Ученики

ORDER BY ID;

DECLARE @ID2 INT, @Фамилия2 NVARCHAR(50), @Школа2 NVARCHAR(50), @Баллы2 DECIMAL(4,1);

OPEN Курсор2;

-- Пятая строка

FETCH ABSOLUTE 5 FROM Курсор2 INTO @ID2, @Фамилия2, @Школа2, @Баллы2;

PRINT '5-я строка: ' + @Фамилия2 + ' (' + @Школа2 + ') - ' + CAST(@Баллы2 AS NVARCHAR(10));

-- Предыдущая (четвертая)

FETCH PRIOR FROM Курсор2 INTO @ID2, @Фамилия2, @Школа2, @Баллы2;

PRINT 'Предыдущая: ' + @Фамилия2 + ' (' + @Школа2 + ') - ' + CAST(@Баллы2 AS NVARCHAR(10));

-- С конца четвертая

FETCH ABSOLUTE -4 FROM Курсор2 INTO @ID2, @Фамилия2, @Школа2, @Баллы2;

PRINT '4-я с конца: ' + @Фамилия2 + ' (' + @Школа2 + ') - ' + CAST(@Баллы2 AS NVARCHAR(10));

-- Следующая

FETCH NEXT FROM Курсор2 INTO @ID2, @Фамилия2, @Школа2, @Баллы2;

PRINT 'Следующая: ' + @Фамилия2 + ' (' + @Школа2 + ') - ' + CAST(@Баллы2 AS NVARCHAR(10));

-- Первая

FETCH FIRST FROM Курсор2 INTO @ID2, @Фамилия2, @Школа2, @Баллы2;

PRINT 'Первая: ' + @Фамилия2 + ' (' + @Школа2 + ') - ' + CAST(@Баллы2 AS NVARCHAR(10));

CLOSE Курсор2;

DEALLOCATE Курсор2;

Выполнение задания представлено на рисунке 16.8.

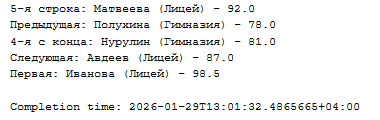


Рисунок 16.8 – Выполнение задания

1. Создайте курсор с прокруткой, содержащий список учеников, откройте его, выведите последнюю, шесть позиций назад находящуюся, четыре позиций вперед находящуюся строку, закройте и освободите курсор.

Код выполнения задания представлен ниже:

DECLARE Курсор3 CURSOR SCROLL FOR

SELECT ID, Фамилия, Предмет, Баллы

FROM Ученики

ORDER BY ID;

DECLARE @ID3 INT, @Фамилия3 NVARCHAR(50), @Предмет3 NVARCHAR(50), @Баллы3 DECIMAL(4,1);

OPEN Курсор3;

-- Последняя строка

FETCH LAST FROM Курсор3 INTO @ID3, @Фамилия3, @Предмет3, @Баллы3;

PRINT 'Последняя: ' + @Фамилия3 + ' (' + @Предмет3 + ') - ' + CAST(@Баллы3 AS NVARCHAR(10));

-- 6 позиций назад

FETCH RELATIVE -6 FROM Курсор3 INTO @ID3, @Фамилия3, @Предмет3, @Баллы3;

PRINT '6 позиций назад: ' + @Фамилия3 + ' (' + @Предмет3 + ') - ' + CAST(@Баллы3 AS NVARCHAR(10));

-- 4 позиции вперед

FETCH RELATIVE 4 FROM Курсор3 INTO @ID3, @Фамилия3, @Предмет3, @Баллы3;

PRINT '4 позиции вперед: ' + @Фамилия3 + ' (' + @Предмет3 + ') - ' + CAST(@Баллы3 AS NVARCHAR(10));

CLOSE Курсор3;

DEALLOCATE Курсор3;

Выполнение задания представлено на рисунке 16.9.

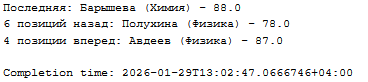


Рисунок 16.9 – Выполнение задания

1. С помощью курсора, вычислите сумму баллов у учеников с наибольшим и наименьшим.

Код выполнения задания представлен ниже:

DECLARE Курсор4 CURSOR FOR

SELECT Баллы

FROM Ученики

WHERE Баллы = (SELECT MAX(Баллы) FROM Ученики)

OR Баллы = (SELECT MIN(Баллы) FROM Ученики);

DECLARE @Баллы4 DECIMAL(4,1), @СуммаБаллов DECIMAL(5,1) = 0;

OPEN Курсор4;

FETCH NEXT FROM Курсор4 INTO @Баллы4;

WHILE @@FETCH\_STATUS = 0

BEGIN

SET @СуммаБаллов = @СуммаБаллов + @Баллы4;

FETCH NEXT FROM Курсор4 INTO @Баллы4;

END;

PRINT 'Сумма баллов у учеников с макс и мин баллом: ' + CAST(@СуммаБаллов AS NVARCHAR(10));

CLOSE Курсор4;

DEALLOCATE Курсор4;

Выполнение задания представлено на рисунке 16.10.

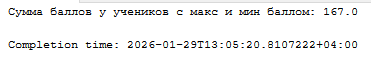


Рисунок 16.10 – Выполнение задания

1. С помощью курсора, сгенерируйте строку вида «Ученики <список фамилий и названий предметов, разделенных запятыми> участвовали в олимпиаде».

Код выполнения задания представлен ниже:

DECLARE Курсор5 CURSOR FOR

SELECT Фамилия, Предмет

FROM Ученики

ORDER BY Фамилия;

DECLARE @Фамилия5 NVARCHAR(50), @Предмет5 NVARCHAR(50);

DECLARE @Результат5 NVARCHAR(MAX) = 'Ученики ';

DECLARE @Счетчик5 INT = 0;

DECLARE @СтрокаСчетчик INT = 0;

OPEN Курсор5;

FETCH NEXT FROM Курсор5 INTO @Фамилия5, @Предмет5;

WHILE @@FETCH\_STATUS = 0

BEGIN

IF @Счетчик5 > 0

SET @Результат5 = @Результат5 + ', ';

SET @Результат5 = @Результат5 + @Фамилия5 + ' (' + @Предмет5 + ')';

SET @Счетчик5 = @Счетчик5 + 1;

SET @СтрокаСчетчик = @СтрокаСчетчик + 1;

-- Добавляем перенос строки после каждых 3-х учеников

IF @СтрокаСчетчик % 3 = 0 AND @@FETCH\_STATUS = 0

SET @Результат5 = @Результат5 + CHAR(13) + CHAR(10);

FETCH NEXT FROM Курсор5 INTO @Фамилия5, @Предмет5;

END;

SET @Результат5 = @Результат5 + ' участвовали в олимпиаде.';

PRINT @Результат5;

CLOSE Курсор5;

DEALLOCATE Курсор5;

Выполнение задания представлено на рисунке 16.11.

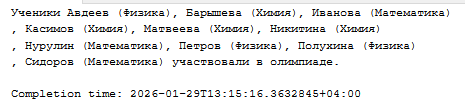


Рисунок 16.11 – Выполнение задания

1. Создайте курсор, содержащий список учеников, с его помощью выведите учеников с нечетной позицией.

Код выполнения задания представлен ниже:

DECLARE Курсор6 CURSOR FOR

SELECT ROW\_NUMBER() OVER (ORDER BY ID) AS Позиция, Фамилия, Школа, Баллы

FROM Ученики;

DECLARE @Позиция6 INT, @Фамилия6 NVARCHAR(50), @Школа6 NVARCHAR(50), @Баллы6 DECIMAL(4,1);

OPEN Курсор6;

FETCH NEXT FROM Курсор6 INTO @Позиция6, @Фамилия6, @Школа6, @Баллы6;

PRINT 'Ученики с нечетной позицией:';

WHILE @@FETCH\_STATUS = 0

BEGIN

IF @Позиция6 % 2 = 1

PRINT CAST(@Позиция6 AS NVARCHAR(10)) + '. ' + @Фамилия6 + ' (' + @Школа6 + ') - ' + CAST(@Баллы6 AS NVARCHAR(10));

FETCH NEXT FROM Курсор6 INTO @Позиция6, @Фамилия6, @Школа6, @Баллы6;

END;

CLOSE Курсор6;

DEALLOCATE Курсор6;

Выполнение задания представлено на рисунке 16.12.

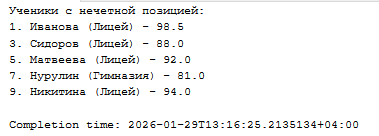


Рисунок 16.12 – Выполнение задания

1. Создайте курсор, содержащий отсортированный по убыванию баллов список учеников, откройте его, для каждого ученика выведите фамилию, предмет, школу, баллы и процентное соотношение баллов с предыдущим учеником.

Код выполнения задания представлен ниже:

DECLARE Курсор7 CURSOR SCROLL FOR

SELECT Фамилия, Предмет, Школа, Баллы

FROM Ученики

ORDER BY Баллы DESC;

DECLARE @Фамилия7 NVARCHAR(50), @Предмет7 NVARCHAR(50), @Школа7 NVARCHAR(50), @Баллы7 DECIMAL(4,1);

DECLARE @ПредыдущиеБаллы DECIMAL(4,1);

DECLARE @Процент DECIMAL(5,2);

DECLARE @ПерваяЗапись BIT = 1;

OPEN Курсор7;

FETCH FIRST FROM Курсор7 INTO @Фамилия7, @Предмет7, @Школа7, @Баллы7;

PRINT 'Процентное соотношение баллов с предыдущим учеником:';

PRINT '-----------------------------------------------------';

WHILE @@FETCH\_STATUS = 0

BEGIN

IF @ПерваяЗапись = 1

BEGIN

PRINT @Фамилия7 + ' | ' + @Предмет7 + ' | ' + @Школа7 + ' | ' +

CAST(@Баллы7 AS NVARCHAR(10)) + ' | 100.00% (первый)';

SET @ПерваяЗапись = 0;

END

ELSE

BEGIN

SET @Процент = (@Баллы7 / @ПредыдущиеБаллы) \* 100;

PRINT @Фамилия7 + ' | ' + @Предмет7 + ' | ' + @Школа7 + ' | ' +

CAST(@Баллы7 AS NVARCHAR(10)) + ' | ' + CAST(@Процент AS NVARCHAR(10)) + '%';

END

SET @ПредыдущиеБаллы = @Баллы7;

FETCH NEXT FROM Курсор7 INTO @Фамилия7, @Предмет7, @Школа7, @Баллы7;

END;

CLOSE Курсор7;

DEALLOCATE Курсор7;

Выполнение задания представлено на рисунке 16.13.

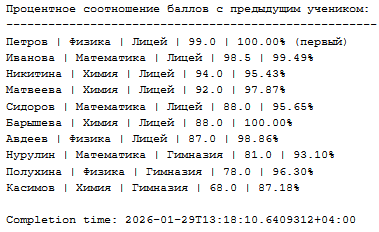


Рисунок 16.13 – Выполнение задания

**Рефлексия:** Я изучил: использование курсоров; прокрутку курсоров; использование циклов в курсорах; использование переменных в курсорах.

**Инструкционная карта №17**

**Тема: «Оконные функции. Аналитические, ранжирующие, смещения функции».**

**Цель работы:**

1. Изучить оконные функции.
2. Изучить аналитические функции.
3. Изучить ранжирующие функции.
4. Изучить функции смещения.

**Ход работы:**

1. Была создана база данных «ПЗ17\_Рыбалев», в ней была создана таблица «Ученики» и заполнена соответствующими данными.
2. Были выполнены следующие примеры:

Пример 1: вывести список учеников и максимальный балл в каждой строке. Выполнение примера представлено на рисунке 17.1.



Рисунок 17.1 – Выполнение примера 1

Пример 2: вывести список учеников и разницу между баллами ученика и минимальным баллом в каждой строке. Выполнение примера представлено на рисунке 17.2.



Рисунок 17.2 – Выполнение примера 2

Пример 3: вывести список учеников и процентное соотношение к суммарному баллу в каждой строке. Выполнение примера представлено на рисунке 17.3.

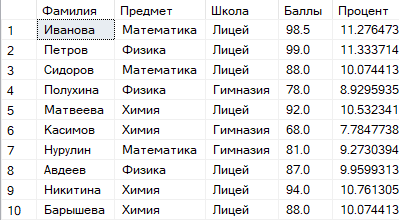


Рисунок 17.3 – Выполнение примера 3

Пример 4: вывести список учеников и средний балл в школе в каждой строке. Выполнение примера представлено на рисунке 17.4.



Рисунок 17.4 – Выполнение примера 4

Пример 5: вывести список учеников и количество учеников в школе в каждой строке, отсортировать по школам в оконной функции. Выполнение примера представлено на рисунке 17.5.

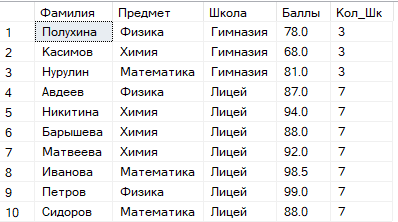


Рисунок 17.5 – Выполнение примера 5

Пример 6: вывести список учеников и номер строки при сортировке по баллам по убыванию. Выполнение примера представлено на рисунке 17.6.

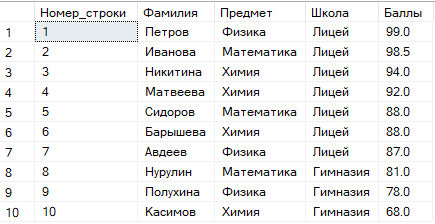


Рисунок 17.6 – Выполнение примера 6

Пример 7: вывести список учеников и номер строки внутри школы при сортировке по баллам по убыванию. Выполнение примера представлено на рисунке 17.7.

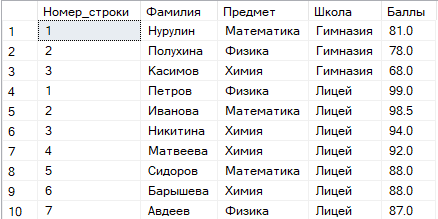


Рисунок 17.7 – Выполнение примера 7

Пример 8: вывести список учеников и ранг по баллам в каждой школе. Выполнение примера представлено на рисунке 17.8.



Рисунок 17.8 – Выполнение примера 8

Пример 9: вывести список учеников и сжатый ранг по баллам в каждой школе. Результат отсортировать по фамилии в алфавитном порядке. Выполнение примера представлено на рисунке 17.9.

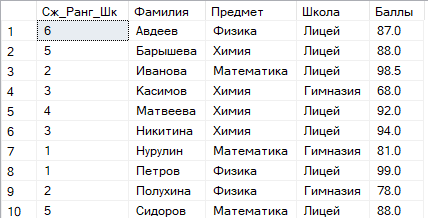


Рисунок 17.9 – Выполнение примера 9

Пример 10: вывести список учеников, распределенных по трем группам по фамилии. Выполнение примера представлено на рисунке 17.10.



Рисунок 17.10 – Выполнение примера 10

Пример 11: вывести список учеников, распределенных по двум группам по баллам внутри школы. Выполнение примера представлено на рисунке 17.11.



Рисунок 17.11 – Выполнение примера 11

Пример 12: вывести список учеников и разницу с баллами предыдущего ученика, при сортировке по возрастанию баллов. Выполнение примера представлено на рисунке 17.12.



Рисунок 17.12 – Выполнение примера 12

Пример 13: вывести список учеников и разницу с баллами ученика через две позиции при сортировке по убыванию баллов, значение по умолчанию использовать 0. Выполнение примера представлено на рисунке 17.13.



Рисунок 17.13 – Выполнение примера 13

Пример 14: вывести список учеников и разницу с баллами первого ученика при сортировке по убыванию баллов. Выполнение примера представлено на рисунке 17.14.

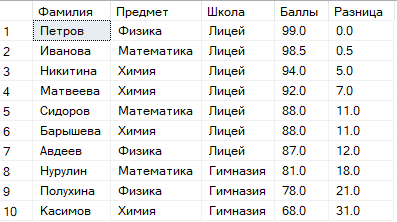


Рисунок 17.14 – Выполнение примера 14

Пример 15: вывести список учеников и разницу с баллами последнего ученика в школе при сортировке по убыванию баллов. Выполнение примера представлено на рисунке 17.15.



Рисунок 17.15 – Выполнение примера 15

**Самостоятельное задание**

Были выполнены следующие задания:

1. Вывести неповторяющийся список предметов и разницу между средними баллами по предмету и максимальным баллом в каждой строке.

Код выполнения задания представлен ниже:

SELECT DISTINCT

Предмет,

AVG(Баллы) OVER(PARTITION BY Предмет) AS [Средний балл по предмету],

MAX(Баллы) OVER() AS [Максимальный балл],

MAX(Баллы) OVER() - AVG(Баллы) OVER(PARTITION BY Предмет) AS Разница

FROM Ученики

ORDER BY Предмет;

Выполнение задания представлено на рисунке 17.16.

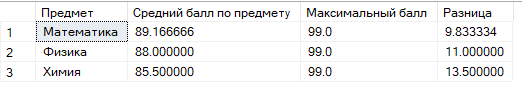


Рисунок 17.16 – Выполнение задания

1. Вывести список предметов и процентное соотношение к среднему баллу в каждой строке.

Код выполнения задания представлен ниже:

SELECT

Предмет,

Баллы,

AVG(Баллы) OVER(PARTITION BY Предмет) AS [Средний балл по предмету],

(Баллы \* 100.0) / AVG(Баллы) OVER(PARTITION BY Предмет) AS [Процент к среднему]

FROM Ученики

ORDER BY Предмет, Баллы DESC;

Выполнение задания представлено на рисунке 17.17.

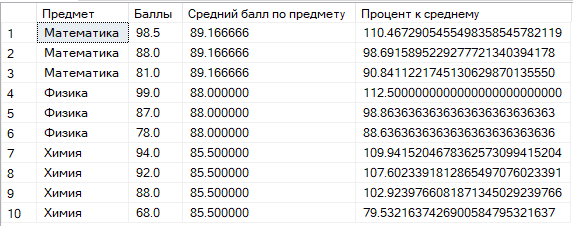


Рисунок 17.17 – Выполнение задания

1. Вывести список предметов и минимальный балл по предмету в каждой строке.

Код выполнения задания представлен ниже:

SELECT

Предмет,

Баллы,

MIN(Баллы) OVER(PARTITION BY Предмет) AS [Минимальный балл по предмету]

FROM Ученики

ORDER BY Предмет, Баллы;

Выполнение задания представлено на рисунке 17.18.



Рисунок 17.18 – Выполнение задания

1. Вывести неповторяющийся список школ и суммарный балл в школе в каждой строке, отсортировать по школам в оконной функции.

Код выполнения задания представлен ниже:

SELECT DISTINCT

Школа,

SUM(Баллы) OVER(PARTITION BY Школа) AS [Суммарный балл в школе]

FROM Ученики

ORDER BY Школа;

Выполнение задания представлено на рисунке 17.19.



Рисунок 17.19 – Выполнение задания

1. Вывести список учеников и номер строки при сортировке по фамилиям в обратном алфавитном порядке.

Код выполнения задания представлен ниже:

SELECT

ROW\_NUMBER() OVER(ORDER BY Фамилия DESC) AS [Номер строки],

Фамилия,

Предмет,

Школа,

Баллы

FROM Ученики

ORDER BY Фамилия DESC;

Выполнение задания представлено на рисунке 17.20.

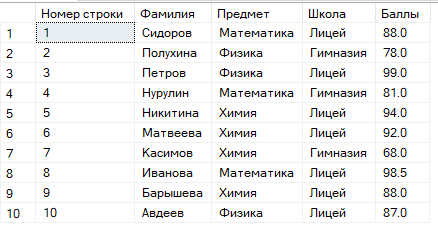


Рисунок 17.20 – Выполнение задания

1. Вывести список учеников, номер строки внутри школы и количество учеников в школе при сортировке по баллам по убыванию.

Код выполнения задания представлен ниже:

SELECT

Фамилия,

Предмет,

Школа,

Баллы,

ROW\_NUMBER() OVER(PARTITION BY Школа ORDER BY Баллы DESC) AS [Номер внутри школы],

COUNT(\*) OVER(PARTITION BY Школа) AS [Количество учеников в школе]

FROM Ученики

ORDER BY Школа, Баллы DESC;

Выполнение задания представлено на рисунке 17.21.



Рисунок 17.21 – Выполнение задания

1. Вывести список предметов и ранг по средним баллам.

Код выполнения задания представлен ниже:

WITH СредниеБаллыПоПредметам AS (

SELECT

Предмет,

AVG(Баллы) AS [Средний балл]

FROM Ученики

GROUP BY Предмет

)

SELECT

Предмет,

[Средний балл],

RANK() OVER(ORDER BY [Средний балл] DESC) AS [Ранг по среднему баллу]

FROM СредниеБаллыПоПредметам

ORDER BY [Ранг по среднему баллу];

Выполнение задания представлено на рисунке 17.22.



Рисунок 17.22 – Выполнение задания

1. Вывести список учеников и сжатый ранг по баллам. Результат отсортировать по фамилии в алфавитном порядке.

Код выполнения задания представлен ниже:

SELECT

DENSE\_RANK() OVER(ORDER BY Баллы DESC) AS [Сжатый ранг],

Фамилия,

Предмет,

Школа,

Баллы

FROM Ученики

ORDER BY Фамилия;

Выполнение задания представлено на рисунке 17.23.

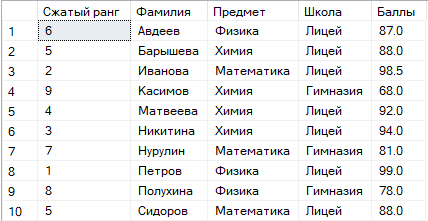


Рисунок 17.23 – Выполнение задания

1. Вывести список учеников, распределенных по пяти группам по фамилии.

Код выполнения задания представлен ниже:

SELECT

NTILE(5) OVER(ORDER BY Фамилия) AS [Группа],

Фамилия,

Предмет,

Школа,

Баллы

FROM Ученики

ORDER BY Фамилия;

Выполнение задания представлено на рисунке 17.24.

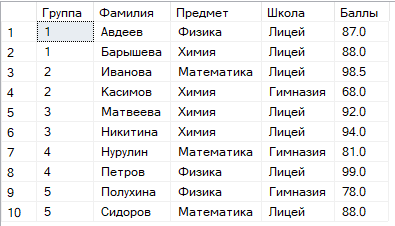


Рисунок 17.24 – Выполнение задания

1. Вывести список учеников, распределенных по трем группам по баллам внутри школы.

Код выполнения задания представлен ниже:

SELECT

NTILE(3) OVER(PARTITION BY Школа ORDER BY Баллы DESC) AS [Группа внутри школы],

Фамилия,

Школа,

Баллы

FROM Ученики

ORDER BY Школа, [Группа внутри школы], Баллы DESC;

Выполнение задания представлено на рисунке 17.25.

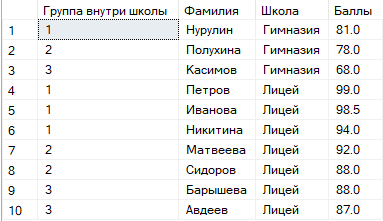


Рисунок 17.25 – Выполнение задания

1. Вывести список учеников и разницу с баллами ученика, находящегося выше на три позиции при сортировке по возрастанию баллов.

Код выполнения задания представлен ниже:

SELECT

Фамилия,

Предмет,

Школа,

Баллы,

LAG(Баллы, 3) OVER(ORDER BY Баллы) AS [Баллы ученика выше на 3 позиции],

Баллы - LAG(Баллы, 3) OVER(ORDER BY Баллы) AS [Разница с учеником выше на 3 позиции]

FROM Ученики

ORDER BY Баллы;

Выполнение задания представлено на рисунке 17.26.



Рисунок 17.26 – Выполнение задания

1. Вывести список учеников и разницу с баллами следующего ученика при сортировке по убыванию баллов, значение по умолчанию использовать 0.

Код выполнения задания представлен ниже:

SELECT

Фамилия,

Предмет,

Школа,

Баллы,

LEAD(Баллы, 1, 0) OVER(ORDER BY Баллы DESC) AS [Баллы следующего ученика],

Баллы - LEAD(Баллы, 1, 0) OVER(ORDER BY Баллы DESC) AS [Разница со следующим учеником]

FROM Ученики

ORDER BY Баллы DESC;

Выполнение задания представлено на рисунке 17.27.



Рисунок 17.27 – Выполнение задания

**Рефлексия:** Я изучил: оконные функции; аналитические функции; ранжирующие функции; функции смещения.

**Инструкционная карта №18**

**Тема: «Инструкции OLAP Использование ROLLUP, CUBE, GROUPING SETS, PIVOT, UNPIVOT».**

**Цель работы:**

1. Изучить использование ROLLUP.
2. Изучить использование CUBE.
3. Изучить использование GROUPING SETS.
4. Изучить использование PIVOT.
5. Изучить использование UNPIVOT.

**Ход работы:**

1. Была создана база данных «ПЗ18\_Рыбалев», в ней была создана таблица «Ученики» и заполнена соответствующими данными.
2. Были выполнены следующие примеры:

Пример 1: Напишите запрос, который выводит количество учеников по предметам по каждой школе, и промежуточные итоги. Выполнение примера представлено на рисунке 18.1.



Рисунок 18.1 – Выполнение примера 1

Пример 2: Напишите запрос, который выводит количество учеников по предметам и по школам, и промежуточные итоги. Выполнение примера представлено на рисунке 18.2.



Рисунок 18.2 – Выполнение примера 2

Пример 3: Напишите запрос, который выводит количество учеников по предметам и по школам. Выполнение примера представлено на рисунке 18.3.

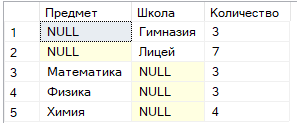


Рисунок 18.3 – Выполнение примера 3

Пример 4: Напишите запрос, который выводит количество учеников по предметам по каждой школе и промежуточные итоги. NULL значения заменить на соответствующий текст. Выполнение примера представлено на рисунке 18.4.



Рисунок 18.4 – Выполнение примера 4

Пример 5: Напишите запрос, который выводит количество учеников по предметам и по школам, и промежуточные итоги. В итоговых строках NULL значения заменить на соответствующий текст в зависимости от группировки. Выполнение примера представлено на рисунке 18.5.



Рисунок 18.5 – Выполнение примера 5

Пример 6: Напишите запрос, который выводит количество учеников по предметам и по школам. В итоговых строках NULL значения заменить на соответствующий текст в зависимости от уровней группировки. Выполнение примера представлено на рисунке 18.6.

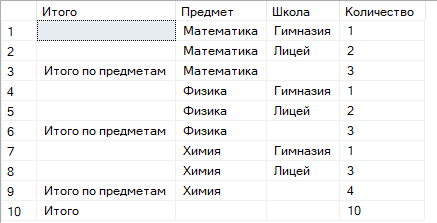


Рисунок 18.6 – Выполнение примера 6

Пример 7: Напишите запрос, который выводит количество учеников по предметам по столбцам. Выполнение примера представлено на рисунке 18.7.



Рисунок 18.7 – Выполнение примера 7

Пример 8: Напишите запрос для вывода количества учеников для каждой школы по каждому предмету (школы должны быть указаны в строках, предметы в столбцах). Выполнение примера представлено на рисунке 18.8.

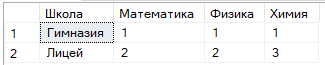


Рисунок 18.8 – Выполнение примера 8

Пример 9: Напишите запрос, который выводит фамилию учеников и предметы вместе со школами в один столбец. Выполнение примера представлено на рисунке 18.9.

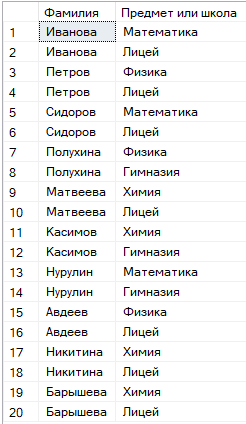


Рисунок 18.9 – Выполнение примера 9

**Самостоятельное задание**

Были выполнены следующие задания:

1. Напишите запрос, который выводит максимальный балл учеников по школам, по каждому предмету по каждой школе и промежуточные итоги.

Код выполнения задания представлен ниже:

SELECT

ISNULL(Школа, 'Все школы') AS Школа,

ISNULL(Предмет, 'Все предметы') AS Предмет,

MAX(Баллы) AS [Максимальный балл]

FROM Ученики

GROUP BY ROLLUP(Школа, Предмет)

ORDER BY Школа, Предмет;

Выполнение задания представлено на рисунке 18.10.



Рисунок 18.10 – Выполнение задания

1. Напишите запрос, который выводит минимальный балл учеников по школам и по предметам, и промежуточные итоги.

Код выполнения задания представлен ниже:

SELECT

ISNULL(Школа, 'Все школы') AS Школа,

ISNULL(Предмет, 'Все предметы') AS Предмет,

MIN(Баллы) AS [Минимальный балл]

FROM Ученики

GROUP BY ROLLUP(Школа, Предмет)

ORDER BY Школа, Предмет;

Выполнение задания представлено на рисунке 18.11.



Рисунок 18.11 – Выполнение задания

1. Напишите запрос, который выводит средний балл учеников по школам и по предметам.

Код выполнения задания представлен ниже:

SELECT

Школа,

Предмет,

AVG(Баллы) AS [Средний балл]

FROM Ученики

GROUP BY Школа, Предмет

ORDER BY Школа, Предмет;

Выполнение задания представлено на рисунке 18.12.

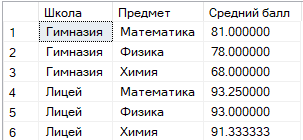


Рисунок 18.12 – Выполнение задания

1. Напишите запрос, который выводит количество учеников по каждой школе по предметам и промежуточные итоги. NULL значения заменить на соответствующий текст.

Код выполнения задания представлен ниже:

SELECT

CASE WHEN GROUPING(Школа) = 1 THEN 'Итого по всем школам' ELSE ISNULL(Школа, 'Не указана') END AS Школа,

CASE WHEN GROUPING(Предмет) = 1 THEN 'Итого по всем предметам' ELSE ISNULL(Предмет, 'Не указан') END AS Предмет,

COUNT(\*) AS [Количество учеников]

FROM Ученики

GROUP BY ROLLUP(Школа, Предмет)

ORDER BY Школа, Предмет;

Выполнение задания представлено на рисунке 18.13.

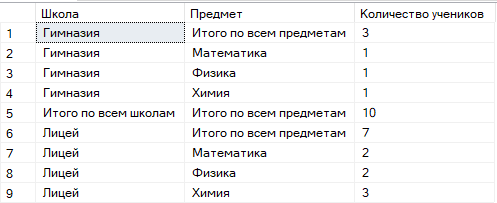


Рисунок 18.13 – Выполнение задания

1. Напишите запрос, который выводит суммарный балл учеников по школам и по предметам, и промежуточные итоги. В итоговых строках NULL значения заменить на соответствующий текст в зависимости от группировки.

Код выполнения задания представлен ниже:

SELECT

CASE

WHEN GROUPING(Школа) = 1 AND GROUPING(Предмет) = 1 THEN 'Общий итог'

WHEN GROUPING(Школа) = 1 THEN 'Итого по всем школам'

ELSE ISNULL(Школа, 'Не указана')

END AS Школа,

CASE

WHEN GROUPING(Школа) = 1 AND GROUPING(Предмет) = 1 THEN ''

WHEN GROUPING(Предмет) = 1 THEN 'Итого по всем предметам'

ELSE ISNULL(Предмет, 'Не указан')

END AS Предмет,

SUM(Баллы) AS [Суммарный балл]

FROM Ученики

GROUP BY ROLLUP(Школа, Предмет)

ORDER BY Школа, Предмет;

Выполнение задания представлено на рисунке 18.14.



Рисунок 18.14 – Выполнение задания

1. Напишите запрос, который выводит максимальный балл учеников по школам и по предметам. В итоговых строках NULL значения заменить на соответствующий текст в зависимости от уровней группировки.

Код выполнения задания представлен ниже:

SELECT

CASE

WHEN GROUPING\_ID(Школа, Предмет) = 3 THEN 'Общий итог'

WHEN GROUPING\_ID(Школа, Предмет) = 1 THEN 'Итого по школе: ' + ISNULL(Школа, 'Не указана')

ELSE ISNULL(Школа, 'Не указана')

END AS Школа,

CASE

WHEN GROUPING\_ID(Школа, Предмет) = 3 THEN ''

WHEN GROUPING\_ID(Школа, Предмет) = 2 THEN 'Итого по предмету: ' + ISNULL(Предмет, 'Не указан')

WHEN GROUPING\_ID(Школа, Предмет) = 1 THEN 'Все предметы'

ELSE ISNULL(Предмет, 'Не указан')

END AS Предмет,

MAX(Баллы) AS [Максимальный балл]

FROM Ученики

GROUP BY CUBE(Школа, Предмет)

ORDER BY GROUPING\_ID(Школа, Предмет), Школа, Предмет;

Выполнение задания представлено на рисунке 18.15.

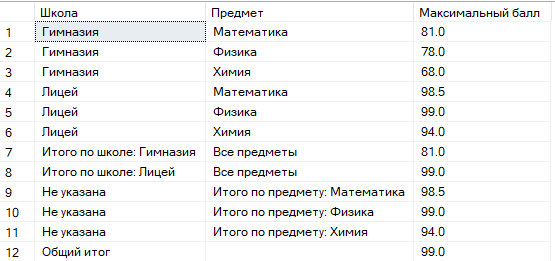


Рисунок 18.15 – Выполнение задания

1. Напишите запрос, который выводит средний балл учеников по школам в столбцы.

Код выполнения задания представлен ниже:

SELECT \*

FROM (

SELECT Школа, Баллы

FROM Ученики

) AS SourceTable

PIVOT (

AVG(Баллы)

FOR Школа IN ([Лицей], [Гимназия])

) AS PivotTable;

Выполнение задания представлено на рисунке 18.16.



Рисунок 18.16 – Выполнение задания

1. Напишите запрос, который выводит средний балл учеников по школам в столбцы и по предметам в строки.

Код выполнения задания представлен ниже:

SELECT

Предмет,

ISNULL([Лицей], 0) AS [Лицей],

ISNULL([Гимназия], 0) AS [Гимназия]

FROM (

SELECT Школа, Предмет, Баллы

FROM Ученики

) AS SourceTable

PIVOT (

AVG(Баллы)

FOR Школа IN ([Лицей], [Гимназия])

) AS PivotTable

ORDER BY Предмет;

Выполнение задания представлено на рисунке 18.17.

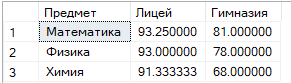


Рисунок 18.17 – Выполнение задания

1. Напишите запрос, который выводит названия предметов, фамилии учеников и школы в один столбец.

Код выполнения задания представлен ниже:

SELECT 'Предмет: ' + Предмет AS [Объединенные данные] FROM Ученики

UNION ALL

SELECT 'Ученик: ' + Фамилия FROM Ученики

UNION ALL

SELECT 'Школа: ' + Школа FROM Ученики

ORDER BY [Объединенные данные];

Выполнение задания представлено на рисунке 18.18.

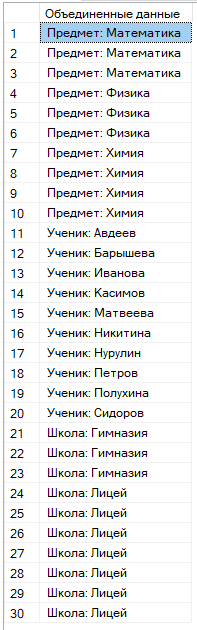


Рисунок 18.18 – Выполнение задания

**Рефлексия:** Я изучил: использование ROLLUP; использование CUBE; использование GROUPING SETS; использование PIVOT; использование UNPIVOT.

**Инструкционная карта №19**

**Тема: «Динамический SQL, XML и JSON. Использование динамического SQL, вывод данных в форматах XML, JSON».**

**Цель работы:**

1. Изучить использование динамического SQL.
2. Изучить вывод данных в формате XML.
3. Изучить вывод данных в формате JSON.

**Ход работы:**

1. Была создана база данных «ПЗ19\_Рыбалев», в ней была создана таблица «Ученики» и заполнена соответствующими данными.
2. Были выполнены следующие примеры:

Пример 1: Создайте и запустите динамический SQL-запрос, выбирающий все данные из заданной таблицы. Выполнение примера представлено на рисунке 19.1.

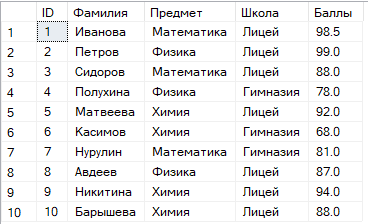


Рисунок 19.1 – Выполнение примера 1

Пример 2: Создайте и запустите динамический SQL-запрос, выбирающий всех учеников из таблицы «Ученики», вторая буква фамилии которых расположена в заданном диапазоне. Выполнение примера представлено на рисунке 19.2.



Рисунок 19.2 – Выполнение примера 2

Пример 3: Создайте временную таблицу #Temp и добавить в нее название пяти таблиц из базы данных «Учебная». Создайте курсор, который, построчно читая таблицу #Temp, выбирает из каждой таблицы все данные. Выполнение примера представлено на рисунке 19.3.

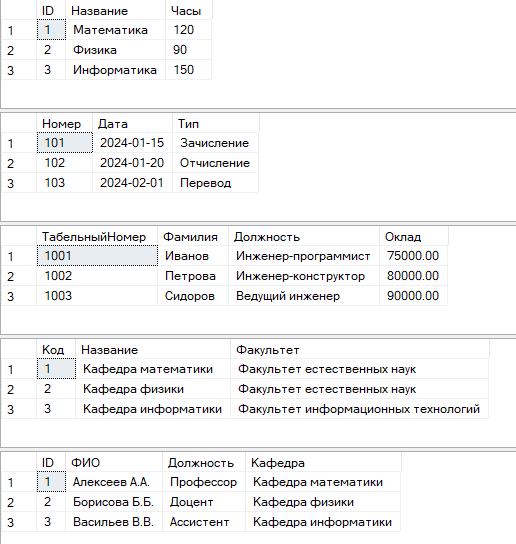


Рисунок 19.3 – Выполнение примера 3

Пример 4: Выберите данные из таблицы «Ученики» и экспортируйте в RAW формате XML. Выполнение примера представлено на рисунке 19.4.



Рисунок 19.4 – Выполнение примера 4

Пример 5: Выберите данные из таблицы «Страны» и экспортируйте в PATH формате XML. Выполнение примера представлено на рисунке 19.5.



Рисунок 19.5 – Выполнение примера 5

Пример 6: Выберите данные из таблицы «Ученики» и экспортируйте в PATH формате JSON. Выполнение примера представлено на рисунке 19.6.



Рисунок 19.6 – Выполнение примера 6

**Самостоятельное задание**

Были выполнены следующие задания:

1. Создайте и запустите динамический SQL-запрос, выбирающий первые N строк из заданной таблицы.

Код выполнения задания представлен ниже:

DECLARE @TableName NVARCHAR(100) = N'Страны';

DECLARE @N INT = 5;

DECLARE @SQL1 NVARCHAR(MAX);

SET @SQL1 = N'SELECT TOP(' + CAST(@N AS NVARCHAR(10)) + N') \* FROM ' + @TableName;

PRINT 'Запрос 1: ' + @SQL1;

EXEC sp\_executesql @SQL1;

Выполнение задания представлено на рисунке 19.7.



Рисунок 19.7 – Выполнение задания

1. Создайте и запустите динамический SQL-запрос, выбирающий все страны из таблицы «Страны», последняя буква названия которых расположена в заданном диапазоне.

Код выполнения задания представлен ниже:

DECLARE @StartLetter NCHAR(1) = N'А';

DECLARE @EndLetter NCHAR(1) = N'Г';

DECLARE @SQL2 NVARCHAR(MAX);

SET @SQL2 = N'

SELECT \* FROM Страны

WHERE RIGHT(Название, 1) BETWEEN @Start AND @End

ORDER BY Название';

PRINT 'Запрос 2: Выбор стран с последней буквой от ' + @StartLetter + ' до ' + @EndLetter;

EXEC sp\_executesql @SQL2, N'@Start NCHAR(1), @End NCHAR(1)', @StartLetter,

@EndLetter;

Выполнение задания представлено на рисунке 19.8.

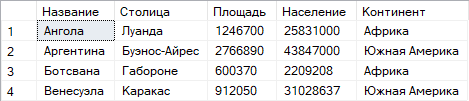


Рисунок 19.8 – Выполнение задания

1. Создайте временную таблицу #Temp и добавьте к ней название столбцов таблицы «Страны». Создайте курсор, который, построчно читая таблицу #Temp, выбирает каждый раз соответствующий столбец из таблицы «Страны».

Код выполнения задания представлен ниже:

CREATE TABLE #TempColumns (

ColumnName NVARCHAR(100)

);

-- Получаем названия столбцов таблицы Страны

INSERT INTO #TempColumns (ColumnName)

SELECT COLUMN\_NAME

FROM INFORMATION\_SCHEMA.COLUMNS

WHERE TABLE\_NAME = N'Страны'

ORDER BY ORDINAL\_POSITION;

DECLARE @ColumnName NVARCHAR(100);

DECLARE @SQL3 NVARCHAR(MAX);

DECLARE ColumnCursor CURSOR FOR

SELECT ColumnName FROM #TempColumns;

OPEN ColumnCursor;

FETCH NEXT FROM ColumnCursor INTO @ColumnName;

WHILE @@FETCH\_STATUS = 0

BEGIN

SET @SQL3 = N'SELECT DISTINCT [' + @ColumnName + N'] FROM Страны WHERE [' + @ColumnName + N'] IS NOT NULL ORDER BY 1';

PRINT 'Столбец: ' + @ColumnName;

EXEC sp\_executesql @SQL3;

PRINT '';

FETCH NEXT FROM ColumnCursor INTO @ColumnName;

END;

CLOSE ColumnCursor;

DEALLOCATE ColumnCursor;

DROP TABLE #TempColumns;

Выполнение задания представлено на рисунке 19.9.

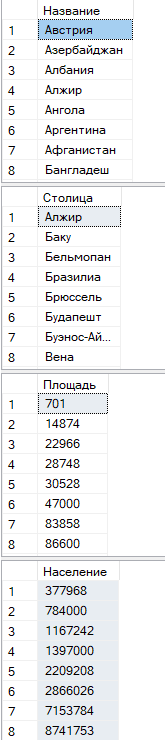


Рисунок 19.9 – Выполнение задания

1. Создайте процедуру, которая принимает как параметр список столбцов, название таблицы и выбирает заданные столбцы из заданной таблицы.

Код выполнения задания представлен ниже:

CREATE OR ALTER PROCEDURE ВыбратьСтолбцы

@Columns NVARCHAR(MAX),

@TableName NVARCHAR(100)

AS

BEGIN

DECLARE @SQL4 NVARCHAR(MAX);

IF @Columns LIKE '%[;--]%' OR @TableName LIKE '%[;--]%'

BEGIN

RAISERROR('Обнаружены недопустимые символы в параметрах', 16, 1);

RETURN;

END

SET @SQL4 = N'SELECT ' + @Columns + N' FROM ' + @TableName;

PRINT 'Выполняемый запрос: ' + @SQL4;

EXEC sp\_executesql @SQL4;

END

EXEC ВыбратьСтолбцы @Columns = N'Название, Столица, Континент', @TableName = N'Страны';

Выполнение задания представлено на рисунке 19.10.



Рисунок 19.10 – Выполнение задания

1. Создайте процедуру, принимающую как параметр список столбцов, название таблицы, название проверяемого столбца, знак сравнения, значение проверки и выбирающую заданные столбцы из заданной таблицы в заданных условиях.

Код выполнения задания представлен ниже:

CREATE OR ALTER PROCEDURE ВыбратьСФильтром

@Columns NVARCHAR(MAX),

@TableName NVARCHAR(100),

@FilterColumn NVARCHAR(100),

@Operator NVARCHAR(10),

@FilterValue NVARCHAR(100)

AS

BEGIN

DECLARE @SQL5 NVARCHAR(MAX);

IF @Operator NOT IN ('=', '<>', '>', '<', '>=', '<=', 'LIKE', 'IN')

BEGIN

RAISERROR('Недопустимый оператор сравнения', 16, 1);

RETURN;

END

SET @SQL5 = N'SELECT ' + @Columns + N' FROM ' + @TableName +

N' WHERE ' + @FilterColumn + N' ' + @Operator + N' ' +

CASE

WHEN @Operator IN ('LIKE') THEN 'N''' + @FilterValue + ''''

WHEN ISNUMERIC(@FilterValue) = 1 THEN @FilterValue

ELSE 'N''' + @FilterValue + ''''

END;

PRINT 'Запрос с фильтром: ' + @SQL5;

EXEC sp\_executesql @SQL5;

END

EXEC ВыбратьСФильтром

@Columns = N'Название, Столица, Население',

@TableName = N'Страны',

@FilterColumn = N'Континент',

@Operator = N'=',

@FilterValue = N'Европа';

Выполнение задания представлено на рисунке 19.11.

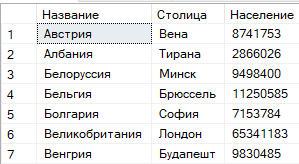


Рисунок 19.11 – Выполнение задания

1. Выберите список европейских стран из таблицы «Страны» и экспортируйте в RAW формате XML.

Код выполнения задания представлен ниже:

SELECT Название, Столица, Население, Площадь

FROM Страны

WHERE Континент = N'Европа'

FOR XML RAW('Страна'), ROOT('ЕвропейскиеСтраны'), ELEMENTS;

Выполнение задания представлено на рисунке 19.12.



Рисунок 19.12 – Выполнение задания

1. Выберите список стран с населением больше 100 млн. чел. из таблицы «Страны» и экспортируйте в PATH формате XML.

Код выполнения задания представлен ниже:

DECLARE @XML\_PATH XML;

SET @XML\_PATH = (

SELECT

Название AS [@Название],

Столица AS [Столица],

Население AS [Население],

Площадь AS [Площадь],

Континент AS [Континент]

FROM Страны

WHERE Население > 100000000

FOR XML PATH('Страна'), ROOT('КрупныеСтраны')

);

SELECT @XML\_PATH AS [PATH\_XML\_Результат];

Выполнение задания представлено на рисунке 19.13.



Рисунок 19.13 – Выполнение задания

1. Выберите список учеников из школы «Лицей» из таблицы «Ученики» и экспортируйте в PATH формате JSON.

Код выполнения задания представлен ниже:

DECLARE @JSON NVARCHAR(MAX);

SET @JSON = (

SELECT

Фамилия,

Предмет,

Баллы,

Школа

FROM Ученики

WHERE Школа = N'Лицей'

FOR JSON PATH, ROOT('УченикиЛицея')

);

SELECT @JSON AS [JSON\_Результат];

Выполнение задания представлено на рисунке 19.14.



Рисунок 19.14 – Выполнение задания

**Рефлексия:** Я изучил: использование динамического SQL; вывод данных в формате XML; вывод данных в формате JSON.

**Инструкционная карта №20**

**Тема: «Пространственные данные. Типы данных GEOMETRY, GEOGRAPHY».**

**Цель работы:**

1. Изучить пространственные данные.
2. Изучить тип данных GEOMETRY.
3. Изучить тип данных GEOGRAPHY.

**Ход работы:**

1. Были выполнены следующие примеры:

Пример 1: Создайте временную таблицу, которая имеет один столбец типа GEOMETRY. Добавьте в таблицу пространственный объект LINESTRING с координатами (1 1, 3 3, 2 4, 2 0, 1 1). Создайте SQL-запрос, выбирающий все данные из заданной таблицы. Выполнение примера представлено на рисунке 20.1.



Рисунок 20.1 – Выполнение примера 1

Пример 2: Создайте временную таблицу, которая имеет один столбец типа

GEOMETRY. Добавьте в таблицу пространственный объект POLYGON с координатами (-3 7,

10 24, 30 16, 13 18, 4 0, 14 -6, 6 -13, 0 -4, -14 -13, -22 -39, -20 -7, -7 3, -11 7, -12 9, -11 10, -9 10, -3 7). Создайте SQL-запрос, выбирающий все данные из заданной таблицы. Выполнение примера представлено на рисунке 20.2.



Рисунок 20.2 – Выполнение примера 2

**Самостоятельное задание**

Была создана временная таблица, состоящая из двух полей типа geometry и varchar (20). Таблица была заполнена заданными полигонами и их названиями.

Код выполнения задания представлен ниже:

CREATE TABLE #Участки (

Название NVARCHAR (50),

Полигон GEOMETRY

);

INSERT INTO #Участки (Название, Полигон) VALUES

('Участок №6', geometry: STPolyFromText ('POLYGON((21369.11 21603.40, 21584.64 21162.81, 21587.40 21027.71, 20727.78 21009.83, 20720.45 21369.76, 20898.96 21373.41, 21369.11 21603.40))', 0)),

-- Заполнение оставшихся участков

-- Выбор всех полей из таблицы

SELECT \* FROM #Участки;

DROP TABLE #Участки;

Выполнение задания представлено на рисунке 20.3.

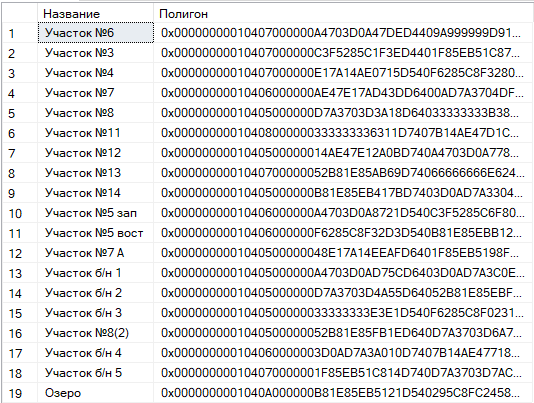


Рисунок 20.3 – Выполнение задания

**Рефлексия:** Я изучил: пространственные данные; тип данных GEOMETRY; тип данных GEOGRAPHY.

**Инструкционная карта №21**

**Тема: «Регистрация удаленных серверов. Запуск, остановка и приостановка служб сервера».**

**Цель работы:** «Научиться анализировать предметную область, проектировать базу данных и выполнять физическое проектирование базы данных»

**Ход работы:**

Задание 1. Анализ предметной области.

Был выбран вариант №16 для выполнения практической работы.

Артисты работают в одной из трупп цирка, возглавляемой художественным руководителем, в функции которого входит контроль за творческим процессом. В подготовке и проведении программы цирка участвуют артисты и постановщики, административно относящиеся к одной из трупп. Каждой программе могут принадлежать несколько трупп.

Таблица «Работники цирка»: характеризуются фамилией, именем и отчеством, должностью (артист, дрессировщик, режиссер), почетным званием, ставкой и стажем работы, адресом проживания, возрастом.

Таблица «Труппа цирка»: каждая труппа цирка готовит и исполняет определенный набор закрепленных за ней цирковых номеров.

Таблица «Программа цирка»: каждый номер характеризуется своим полным названием, указанием общей сложности и формы отчета (репетиционный показ, сдача номера, премьера).

Таблица «Расписание гастролей»: в рамках каждого гастрольного тура составляется расписание гастролей, в котором указываются программы цирка и для каких коллективов проводится выступление, фамилия ответственного режиссера, дата и место проведения.

Таблица «Представления»: для каждой точки в расписании формируется список представлений, где составляется список артистов и фиксируется их участие.

Задание 2. Описание основных сущностей ПО.

В результате проведенного анализа предметной области базы данных «Цирк» легко перечислить основные сущности этой БД. Так как на физическом уровне сущности соответствует таблица, то просто перечислим основные таблицы БД.

В реляционную модель проектированной БД будут входить следующие сущности: Работники цирка, Представления, Расписание гастролей, Труппа цирка, Программа цирка. Данные сущности соответствуют таблице 21.1.

Таблица 21.1 – Основные сущности предметной области.

|  |  |  |
| --- | --- | --- |
| № | Название | Назначение |
| 1 | Труппа цирка | Справочник творческих коллективов и их жанров, используется для классификации артистов и номеров. |
| 2 | Работники цирка | Учет всего персонала цирка (артистов, режиссеров, технического состава) с их характеристиками и принадлежностью к труппе |
| 3 | Программа цирка | Каталог цирковых шоу (представлений) с описанием, режиссером и статусом. |
| 4 | Представления | Фиксация конкретных сеансов (фактов показа): дата, время, место и связь «программа–площадка». |
| 5 | Расписание гастролей | Учет гастрольных туров, их маршрутов и временных периодов, связь с программами и представлениями. |

Затем, для каждой сущности было составлено описание ее атрибутов. Состав атрибутов таблицы «Труппа цирка» представлен в таблице 21.2.

Таблица 21.2 – Атрибуты таблицы «Труппа цирка».

|  |  |  |
| --- | --- | --- |
| Ключевое поле | Название | Назначение |
| Первичный ключ | Код труппы | Уникальный идентификатор творческого коллектива |
|  | Название труппы | Наименование труппы |
|  | Жанр | Основное направление выступлений |
|  | Главный режиссер | ФИО художественного руководителя или главного тренера труппы |
|  | Год\_образования | Дата основания или год создания коллектива. |

Состав атрибутов таблицы «Работники цирка» представлен в таблице 21.3.

Таблица 21.3 – Атрибуты таблицы «Работники цирка».

|  |  |  |
| --- | --- | --- |
| Ключевое поле | Название | Назначение |
| Первичный ключ | Код работника | Уникальный идентификатор сотрудника |
|  | Фамилия | Фамилия работника |
|  | Имя | Имя работника |
|  | Отчество | Отчество работника |
|  | Должность | Занимаемая должность |
| Вторичный ключ | Код труппы | Ссылка на труппу, в составе которой работает артист |
|  | Дата приёма | Дата трудоустройства в цирк |
|  | Стаж | Стаж работы в цирковой сфере |
|  | Телефон | Контактный телефон сотрудника |

Состав атрибутов таблицы «Программа цирка» представлен в таблице 21.4.

Таблица 21.4 – Атрибуты таблицы «Программа цирка».

|  |  |  |
| --- | --- | --- |
| Ключевое поле | Название | Назначение |
| Первичный ключ | Код программы | Уникальный идентификатор циркового шоу |
|  | Название программы | Наименование цирковой программы |
| Вторичный ключ | Код режиссера | Ссылка на таблицу Работники цирка |
|  | Дата премьеры | Дата первого публичного показа программы |
|  | Статус | Текущий статус |
|  | Продолжительность | Общая длительность программы в минутах |

Состав атрибутов таблицы «Представления» представлен в таблице 21.5.

Таблица 21.5 – Атрибуты таблицы «Представления».

|  |  |  |
| --- | --- | --- |
| Ключевое поле | Название | Назначение |
| Первичный ключ | Код представления | Уникальный идентификатор конкретного сеанса |
| Вторичный ключ | Код программы | Ссылка на программу, которая была показана |
| Вторичный ключ | Код тура | Ссылка на таблицу Расписание гастролей |
|  | Дата время | Дата и время начала данного сеанса |
|  | Город | Город, в котором проходило представление |
|  | Площадка | Наименование здания или места проведения |
|  | Статус | Фактический статус |

Состав атрибутов таблицы «Расписание гастролей» представлен в таблице 21.6.

Таблица 21.6 – Атрибуты таблицы «Расписание гастролей».

|  |  |  |
| --- | --- | --- |
| Ключевое поле | Название | Назначение |
| Первичный ключ | Код тура | Уникальный идентификатор гастрольного тура |
|  | Название тура | Наименование тура |
| Вторичный ключ | Код программы | Ссылка на таблицу **Программа цирка** |
|  | Дата начала | Дата старта тура |
|  | Дата окончания | Дата завершения тура |
|  | Маршрут | Описание маршрута |

Задание 3. Построение инфологической модели.

Для базы данных «Цирк» была построена инфологическая модель, включающая в себя 5 сущностей, связанных между собой. На рисунке 21.1 изображена схема инфологической модели.

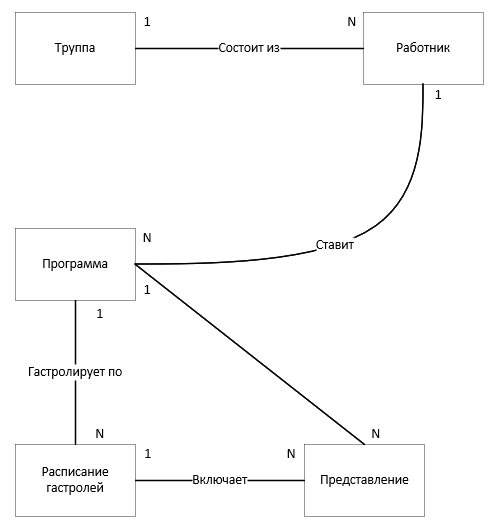


Рисунок 21.1 – Инфологическая модель базы данных «Цирк»

Для выявленных связей была создана таблица 21.7.

Таблица 21.7 – Связи между сущностями.

|  |  |  |  |
| --- | --- | --- | --- |
| № | Название связи | Сущности, участвующие в связи | Назначение |
| 1 | 1:M | Труппа цирка – Работники цирка | Связывает каждого работника-артиста с его творческим коллективом; одна труппа состоит из многих работников |
| 2 | 1:M | Работники цирка – Программа цирка | Фиксирует, какой режиссер-постановщик создал программу; один режиссер может быть автором многих программ |
| 3 | 1:M | Программа цирка – Расписание гастролей | Определяет, какая программа отправляется в тур; одну программу можно отправить в несколько гастрольных туров |
| 4 | 1:М | Расписание гастролей – Представления | Фиксирует состав тура; одно расписание (тур) включает в себя множество конкретных представлений в разных городах |
| 5 | 1:М | Программа цирка – Представления | Отражает факт проведения конкретного сеанса |

Задание 4. Построение даталогической модели БД.

Даталогическая модель базы данных «Цирк» была построена с набором отношений первичных/внешних ключей. Для типов данных использовался стандартные типы данных. Для бизнес-ограничений применялись CHECK‑ограничения. Схема даталогической модели представлена на рисунке 21.2.

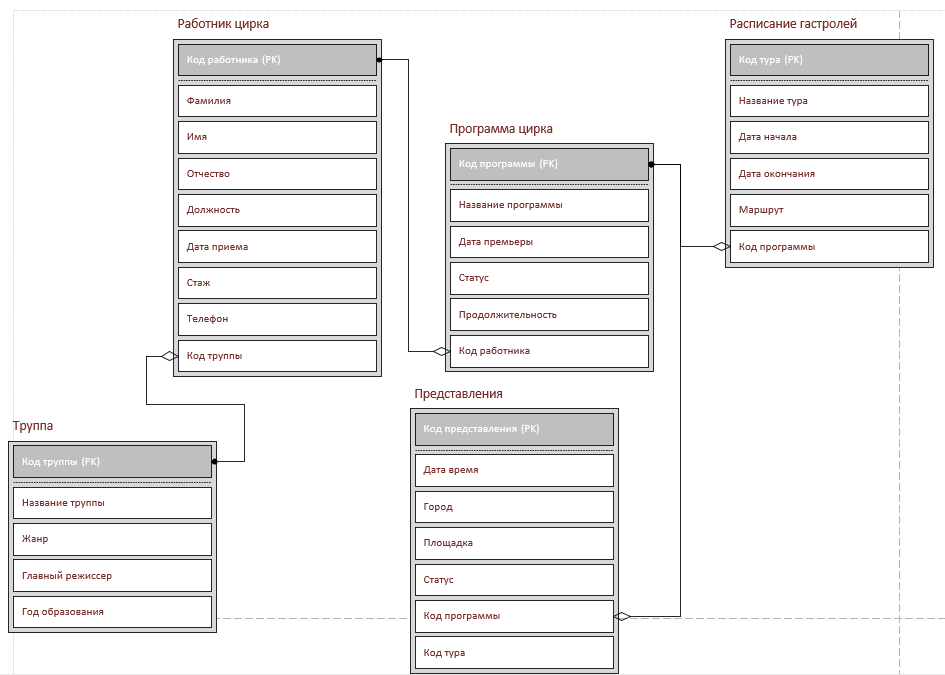


Рисунок 21.2 – Даталогическая модель базы данных «Цирк»

Для каждой сущности была составлена таблица атрибутов с полями: идентификатор, тип, допускает значения NULL, ограничения. Сущности представлены в таблицах 21.8 – 21.12.

Таблица 21.8 – Атрибуты таблицы «Труппа цирка».

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Название | Идентификатор | Тип | Допускает NULL | Ограничения |
| 1 | Идентификатор труппы | TroupeID | INT | Нет | PK, Identity(1,1) |
| 2 | Название труппы | TroupeName | NVARCHAR(100) | Нет | UNIQUE |
| 3 | Жанр | Genre | NVARCHAR(60) | Нет |  |
| 4 | Главный режиссёр | HeadDirector | NVARCHAR(150) | Да |  |
| 5 | Год образования | FormationYear | INT | Да | CHECK (FormationYear > 1800 AND FormationYear <= YEAR(GETDATE())) |

Таблица 21.9 – Атрибуты таблицы «Работники цирка».

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Название | Идентификатор | Тип | Допускает NULL | Ограничения |
| 1 | Табельный номер | WorkerID | INT | Нет | PK, Identity(1,1) |
| 2 | Фамилия | LastName | NVARCHAR(50) | Нет |  |
| 3 | Имя | FirstName | NVARCHAR(50) | Нет |  |
| 4 | Отчество | Patronymic | NVARCHAR(50) | Да |  |
| 5 | Должность | Position | NVARCHAR(50) | Нет |  |
| 6 | Идентификатор труппы | TroupeID | INT | Да | FK (Troupe.TroupeID) |
| 7 | Дата приема | HireDate | DATE | Нет | DEFAULT GETDATE() |
| 8 | Стаж | ExperienceYears | INT | Да | CHECK (ExperienceYears >= 0) |
| 9 | Контактный номер | Phone | NVARCHAR(20) | Да |  |

Таблица 21.10 – Атрибуты таблицы «Программа цирка».

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Название | Идентификатор | Тип | Допускает NULL | Ограничения |
| 1 | Идентификатор программы | ShowID | INT | Нет | PK, Identity(1,1) |
| 2 | Название программы | ShowTitle | NVARCHAR(150) | Нет |  |
| 3 | Код работника | DirectorID | INT | Нет | FK (Workers.WorkerID) |
| 4 | Дата премьеры | PremiereDate | DATE | Нет |  |
| 5 | Статус | Status | NVARCHAR(30) | Нет | DEFAULT 'В разработке' |
| 6 | Продолжительность | DurationMinutes | INT | Нет | CHECK (DurationMinutes > 0) |

Таблица 21.11 – Атрибуты таблицы «Расписание гастролей».

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Название | Идентификатор | Тип | Допускает NULL | Ограничения |
| 1 | Идентификатор тура | TourID | INT | Нет | PK, Identity(1,1) |
| 2 | Название тура | TourName | NVARCHAR(150) | Нет |  |
| 3 | Идентификатор программы | ShowID | INT | Нет | FK (Shows.ShowID) |
| 4 | Дата начала | StartDate | DATE | Да |  |
| 5 | Дата окончания | EndDate | DATE | Нет | CHECK (EndDate >= StartDate) |
| 6 | Маршрут | RouteDescription | NVARCHAR(150) | Да |  |

Таблица 21.12 – Атрибуты таблицы «Представления».

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Название | Идентификатор | Тип | Допускает NULL | Ограничения |
| 1 | Идентификатор представления | PerformanceID | INT | Нет | PK, Identity(1,1) |
| 2 | Идентификатор программы | ShowID | INT | Нет | FK (Shows.ShowID) |
| 3 | Идентификатор тура | TourID | INT | Да | FK (TourSchedule.TourID) |
| 4 | Дата и время | DateTime | DATETIME | Нет |  |
| 5 | Город | City | NVARCHAR(100) | Нет |  |
| 6 | Площадка | Venue | NVARCHAR(150) | Нет |  |
| 7 | Статус | Status | NVARCHAR(30) | Нет | DEFAULT 'Запланировано' |

Задание 5. Создание физической модели и заполнение.

Была создана база данных «CircusDB». В базе данных были созданы 5 таблиц (работники цирка, представления, расписание гастролей, труппа цирка, программа цирка).

Результат выполнения запроса представлен на рисунке 21.3.

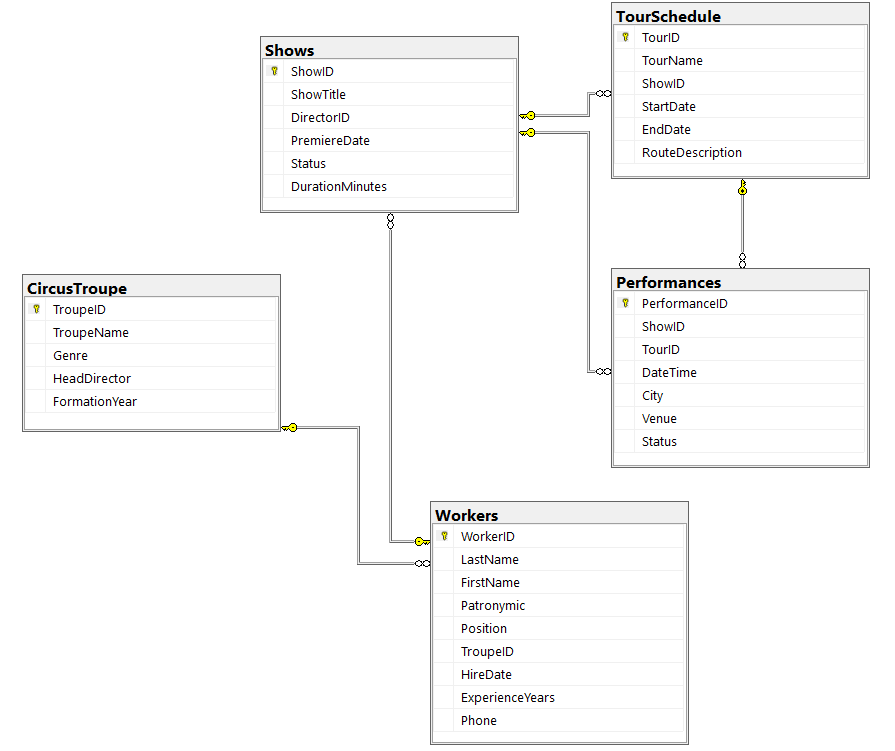


Рисунок 21.3 – Физическая модель базы данных

Таблицы были заполнены данными на основе предметной области.

Задание 6. Создание запросов на выборку.

Были выполнены следующие запросы:

1. Запрос на выбор всех данных по двум полям таблицы.

Код выполнения запроса представлен ниже:

-- Выбираем название и жанр всех трупп

SELECT TroupeName, Genre

FROM CircusTroupe;

Выполнение запроса представлено на рисунке 21.4.

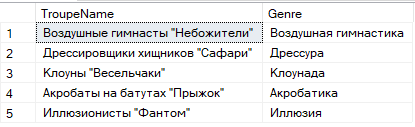


Рисунок 21.4 – Выполнение запроса

1. Запрос на выбор всех неповторяющихся данных по одному полю таблицы.

Код выполнения запроса представлен ниже:

-- Уникальные должности работников цирка

SELECT DISTINCT Position

FROM Workers;

Выполнение запроса представлено на рисунке 21.5.

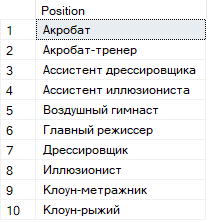


Рисунок 21.5 – Выполнение запроса

1. Запрос на выбор всех полей и записей таблицы, сгруппированных по значению одного поля, с использованием условия на группу (секции GROUP BY, HAVING) и с заголовками колонок, заданными в запросе.

Код выполнения запроса представлен ниже:

-- Группировка работников по труппам, показываем только труппы с 2 и более работниками

SELECT

w.TroupeID AS 'Код труппы',

ct.TroupeName AS 'Название труппы',

COUNT(\*) AS 'Количество работников',

AVG(w.ExperienceYears) AS 'Средний стаж в труппе (лет)',

STRING\_AGG(w.LastName + ' ' + LEFT(w.FirstName, 1) + '.', ', ') AS 'Список работников'

FROM Workers w

JOIN CircusTroupe ct ON w.TroupeID = ct.TroupeID

WHERE w.TroupeID IS NOT NULL

GROUP BY w.TroupeID, ct.TroupeName

HAVING COUNT(\*) >= 2

ORDER BY COUNT(\*) DESC;

Выполнение запроса представлено на рисунке 21.6.

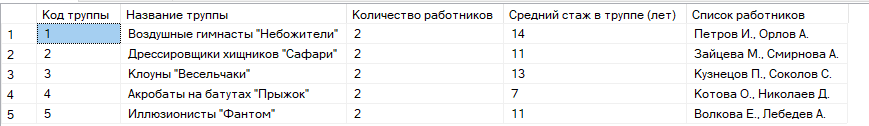


Рисунок 21.6 – Выполнение запроса

1. Запрос на выбор всех неповторяющихся записей по одному полю таблицы с колонкой, образованной агрегирующей функцией SUM и озаглавленной в соответствии со смыслом.

Код выполнения запроса представлен ниже:

-- Общая продолжительность программ по жанрам (через связь с труппами)

SELECT DISTINCT

ct.Genre AS 'Жанр',

SUM(s.DurationMinutes) AS 'Общая продолжительность программ (мин)'

FROM Shows s

JOIN Workers w ON s.DirectorID = w.WorkerID

JOIN CircusTroupe ct ON w.TroupeID = ct.TroupeID

GROUP BY ct.Genre;

Выполнение запроса представлено на рисунке 21.7.

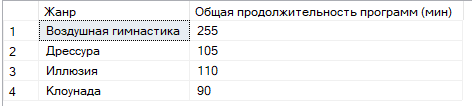


Рисунок 21.7 – Выполнение запроса

1. Выбор нескольких (не всех) полей таблицы, отсортированных по убыванию.

Код выполнения запроса представлен ниже:

-- Программы, отсортированные по длительности (самые долгие первыми)

SELECT

ShowTitle AS 'Название программы',

DurationMinutes AS 'Длительность (мин)',

PremiereDate AS 'Дата премьеры'

FROM Shows

ORDER BY DurationMinutes DESC;

Выполнение запроса представлено на рисунке 21.8.

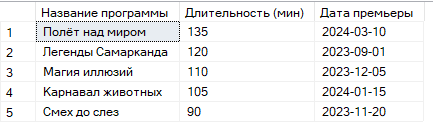


Рисунок 21.8 – Выполнение запроса

1. Выбор произвольного количества полей таблицы с добавлением поля, являющегося результатом арифметического выражения, в котором участвуют значения поля таблицы.

Код выполнения запроса представлен ниже:

-- Работники с расчетом года выхода на пенсию (предположим, пенсия в 60 лет)

SELECT

LastName AS 'Фамилия',

FirstName AS 'Имя',

ExperienceYears AS 'Стаж (лет)',

YEAR(HireDate) AS 'Год приема',

YEAR(HireDate) + 60 AS 'Предполагаемый год выхода на пенсию'

FROM Workers;

Выполнение запроса представлено на рисунке 21.9.



Рисунок 21.9 – Выполнение запроса

1. Запрос на выбор всех записей по одному полю таблицы с колонкой, образованной агрегирующей функцией SUM и озаглавленной в соответствии со смыслом; выбор записей с использованием условия диапазона (between).

Код выполнения запроса представлен ниже:

SELECT

s.ShowTitle AS 'Программа',

SUM(s.DurationMinutes) AS 'Общее время показов (мин)'

FROM Performances p

JOIN Shows s ON p.ShowID = s.ShowID

WHERE p.DateTime BETWEEN '2024-01-01T00:00:00' AND '2024-12-31T23:59:59'

GROUP BY s.ShowTitle;

Выполнение запроса представлено на рисунке 21.10.

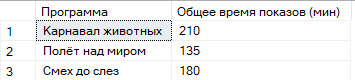


Рисунок 21.10 – Выполнение запроса

1. Запрос на выбор всех записей по произвольному количеству полей таблицы с использованием агрегирующей функции AVG и условием на отбор записей, заданном в секции WHERE.

Код выполнения запроса представлен ниже:

-- Средний стаж работников по должностям (только для артистических должностей)

SELECT

Position AS 'Должность',

AVG(ExperienceYears) AS 'Средний стаж (лет)'

FROM Workers

WHERE Position IN ('Воздушный гимнаст', 'Дрессировщик', 'Клоун', 'Акробат', 'Иллюзионист')

GROUP BY Position;

Выполнение запроса представлено на рисунке 21.11.

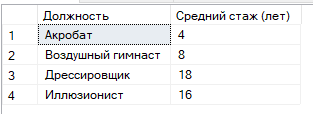


Рисунок 21.11 – Выполнение запроса

1. Запрос на выбор двух полей таблицы с вычислением третьего поля по данным таблицы и сортировкой по убыванию по первому полю, по возрастанию по второму и по убыванию по третьему.

Код выполнения запроса представлен ниже:

-- Гастрольные туры с расчетом длительности и сортировкой

SELECT

TourName AS 'Название тура',

DATEDIFF(DAY, StartDate, EndDate) AS 'Длительность тура (дней)',

ShowID AS 'Код программы'

FROM TourSchedule

ORDER BY

DATEDIFF(DAY, StartDate, EndDate) DESC,

ShowID ASC,

TourName DESC;

Выполнение запроса представлено на рисунке 21.12.

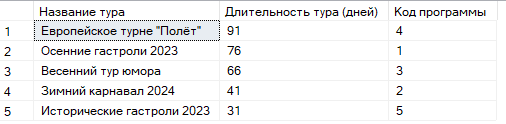


Рисунок 21.12 – Выполнение запроса

1. Запрос на выборку данных таблицы с условием сравнения по шаблону LIKE.

Код выполнения запроса представлен ниже:

-- Поиск работников с фамилией, начинающейся на "К"

SELECT

LastName AS 'Фамилия',

FirstName AS 'Имя',

Position AS 'Должность'

FROM Workers

WHERE LastName LIKE 'К%';

Выполнение запроса представлено на рисунке 21.13.



Рисунок 21.13 – Выполнение запроса

1. Запрос с отбором по условию и сортировкой по убыванию одного из полей, а также добавлением поля, содержащего для всех записей константу, определенную при конструировании запроса.

Код выполнения запроса представлен ниже:

-- Представления в Москве с добавлением константы-примечания

SELECT

City AS 'Город',

Venue AS 'Площадка',

DateTime AS 'Дата и время',

'Столичное представление' AS 'Тип события'

FROM Performances

WHERE City = 'Москва'

ORDER BY DateTime DESC;

Выполнение запроса представлено на рисунке 21.14.



Рисунок 21.14 – Выполнение запроса

1. Запрос с использованием агрегирующих функций MIN и MAX.

Код выполнения запроса представлен ниже:

-- Минимальная и максимальная продолжительность программ по статусам

SELECT

Status AS 'Статус программы',

MIN(DurationMinutes) AS 'Минимальная длительность (мин)',

MAX(DurationMinutes) AS 'Максимальная длительность (мин)',

COUNT(\*) AS 'Количество программ'

FROM Shows

GROUP BY Status;

Выполнение запроса представлено на рисунке 21.15.

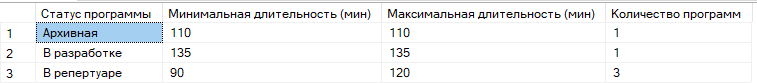


Рисунок 21.15 – Выполнение запроса

1. Запрос с использованием сложного условия с логическими операторами AND, OR и сортировкой.

Код выполнения запроса представлен ниже:

-- Поиск работников с определенным стажем ИЛИ определенной должности

SELECT

LastName AS 'Фамилия',

FirstName AS 'Имя',

Position AS 'Должность',

ExperienceYears AS 'Стаж (лет)',

TroupeID AS 'Код труппы'

FROM Workers

WHERE

(ExperienceYears >= 10 AND Position LIKE '%режиссер%')

OR

(Position IN ('Клоун', 'Акробат') AND ExperienceYears BETWEEN 5 AND 15)

ORDER BY ExperienceYears DESC, LastName ASC;

Выполнение запроса представлено на рисунке 21.16.



Рисунок 21.16 – Выполнение запроса

**Рефлексия**

В ходе практической работы закреплены навыки проектирования и реализации реляционной базы данных. Выделены сущности предметной области «Цирк» и определены связи между ними. Также была построена инфологическая и даталогическая модель.

**Инструкционная карта №22**

**Тема: «Создание приложения на C# для авторизации и регистрации пользователя к базе данных MS SQL Server».**

**Тема «Создание приложения на C# для авторизации и регистрации пользователя к базе данных MS SQL Server»**

**Ход работы:**

1. Для выполнения практической работы была создана база данных с именем «PZ\_22». В новой базе данных были созданы таблицы Users и Role. Созданные таблицы представлены на рисунке 22.1.

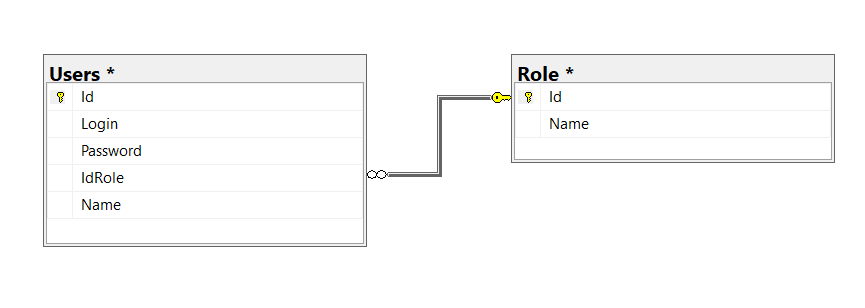


Рисунок 22.1 – Созданные таблицы

1. Затем таблицы были заполнены данными о двух пользователях. Заполненные таблицы представлены на рисунке 22.2.

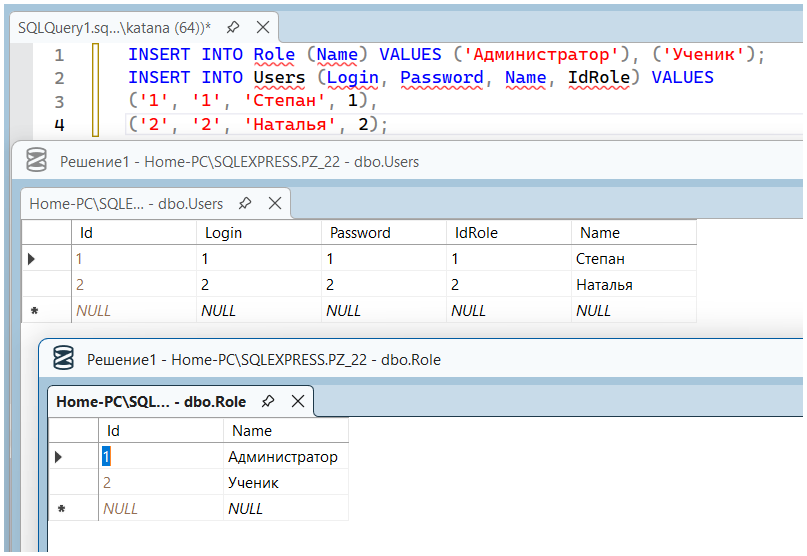


Рисунок 22.2 – Заполнение таблиц

1. Затем был создан проект WPF приложения в Microsoft Visual Studio. В проект добавлена папка ApplicationData. С использованием ADO.NET Entity Data Model создана модель данных, подключенная к ранее созданной базе данных. Созданная модель представлена на рисунке 22.3.

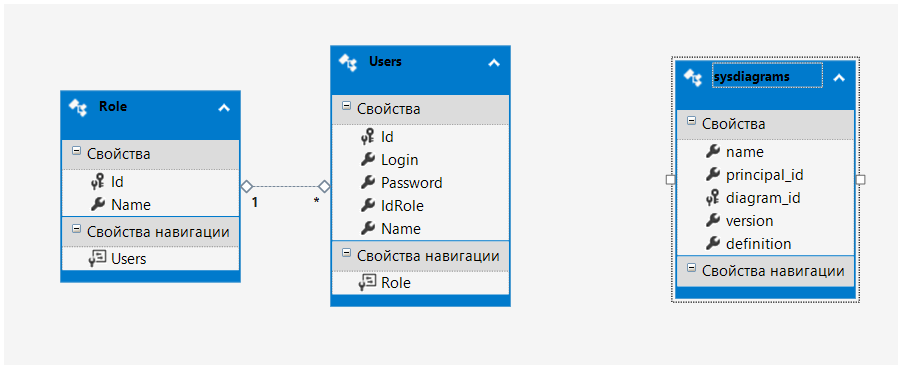


Рисунок 22.3 – Модель БД

1. Затем в папке ApplicationData был добавлен класс AppConnect, в котором было написано подключение к БД «PZ\_22». Созданный класс представлен ниже:

namespace PZ\_22\_St.ApplicationData {

class AppConnect {

public static PZ\_22Entities1 Model1;

}

}

Затем эта же самая модель была подключена к главному окну MainWindow.

1. После подключения БД к приложению, была организована разметка приложения с помощью контейнера Grid, состоящего из двух строк. Были установлены ширина и высота формы, а также нижняя строка содержит фрейм для переключения между страницами. Результат представлен на рисунке 22.4.

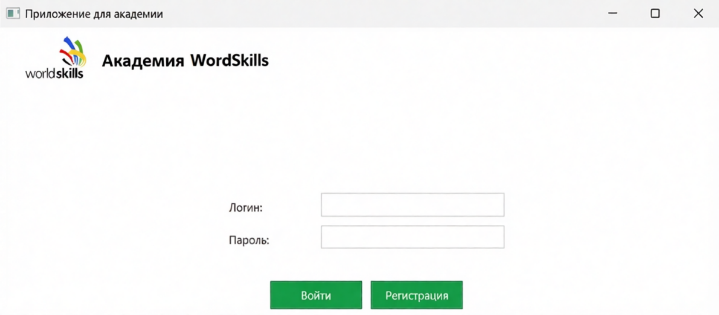


Рисунок 22.4 – Разметка окна приложения

1. Для работы с данными и навигацией были созданы вспомогательные классы AppConnect (подключение к БД) и AppFrame (навигация). Были реализованы страницы авторизации (PageLogin) и регистрации (PageCreateAcc) с проверкой логина, пароля и уникальности учетной записи.

Для разных ролей разработаны отдельные интерфейсы: PageMenuAdmin — панель управления с функциональными кнопками, PageAccountStudent — личный кабинет ученика с основной информацией. Обе страницы содержат кнопку выхода для возврата к авторизации.

1. После было проведено комплексное тестирование функционала приложения. Результат успешного входа в систему под учетной записью администратора представлен на рисунке 22.5.



Рисунок 22.5 – Авторизация администратора

Результат входа в систему под учетной записью ученика представлен на рисунке 22.6

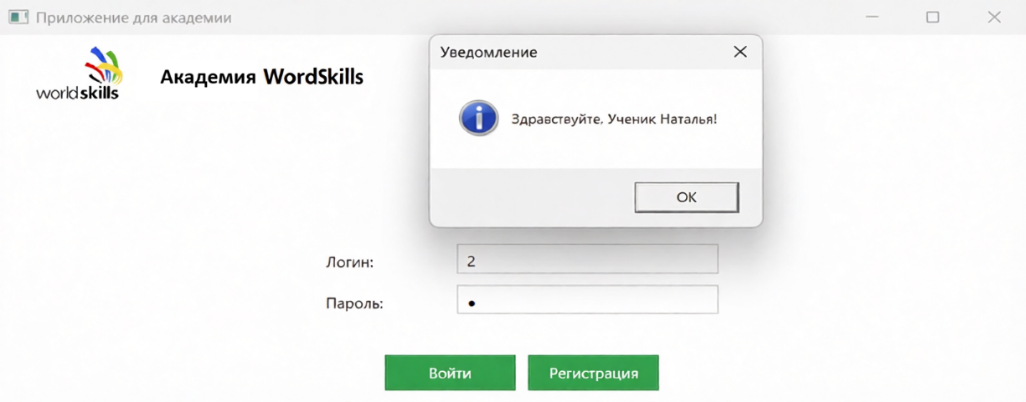


Рисунок 22.6 – Авторизация ученика

Попытка входа под несуществующей учетной записью представлена на рисунке 22.7.

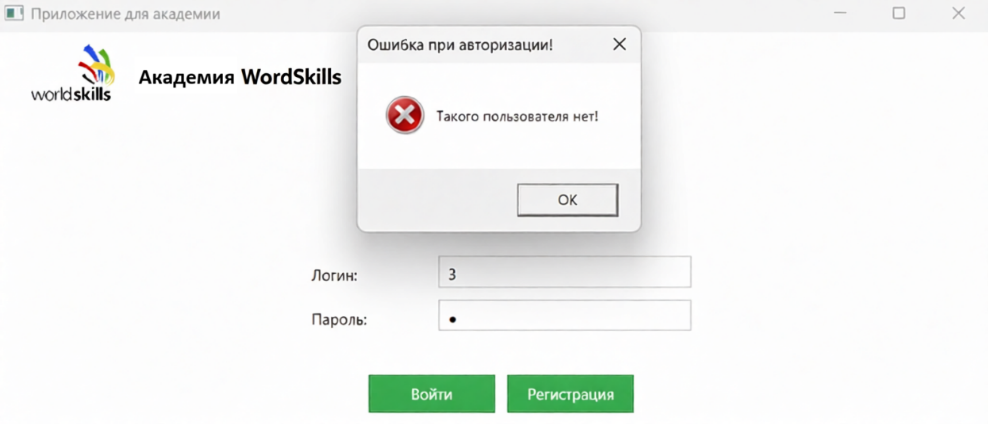


Рисунок 22.7 – Попытка входа с несуществующей учетной записью

1. Далее была создана страница регистрации нового пользователя и проведено тестирование. При попытке зарегистрировать нового пользователя с уже существующим логином система корректно реагирует. Попытка регистрации нового пользователя с существующим логином представлена на рисунке 22.8.

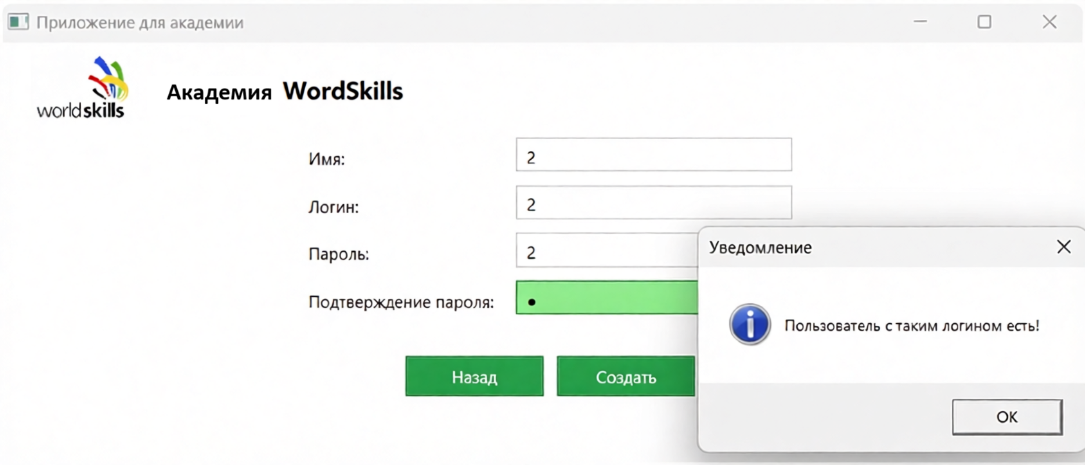


Рисунок 22.8 – Регистрация уже существующего пользователя

**Рефлексия:** в ходе выполнения практической работы мною были изучены пространственные данные; тип данных GEOMETRY; тип данных GEOGRAPHY.

**ЗАКЛЮЧЕНИЕ**

**о прохождении учебной практики**

Обучающийся Рыбалев Денис Александрович 4 курса группы 431 специальности 09.02.07 Информационные системы и программирование проходил учебную практику по УП.11.01 Учебная практика (по профилю специальности) ПМ.11. Разработка, администрирование и защита баз данных с 12.01.2026 по 07.02.2026 в техникуме ДИТИ НИЯУ МИФИ, г. Димитровград, ул. Куйбышева, 300.

По итогам учебной практики обучающийся представил в ДИТИ НИЯУ МИФИ следующие документы:

1. Дневник по учебной практике (включая аттестационный лист).

2. Отчёт по учебной практике.

Отчёт по учебной практике соответствует следующим характеристикам (нужное подчеркнуть):

* представлен своевременно/ несвоевременно;
* правильно и в полном объёме выполнены задания/ задания выполнены с ошибками/ задания не выполнены;
* приложения представлены в полном объеме/ приложения представлены в неполном объеме/ приложения не представлены.

Оформление дневника, отчета и приложений (нужное подчеркнуть):

* соответствует оформление дневника / не соответствует оформление дневника;
* соответствует оформление отчёта/ не соответствует оформление отчёта.

Результаты защиты отчёта по учебной практике, замечания и предложения преподавателя-руководителя практики от образовательной организации:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Заключение о прохождении учебной практики по** УП.11.01 Учебная практика (по профилю специальности) ПМ.11. Разработка, администрирование и защита баз данных**:** **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(прописью)

**Руководитель практики от**

**образовательной организации \_\_\_\_\_\_\_\_\_\_\_\_\_\_** В.В. Фролова

07.02.2026