Государственное автономное профессиональное образовательное учреждение    
Чувашской Республики «Межрегиональный центр компетенций –  
 Чебоксарский электромеханический колледж» Министерства образования    
Чувашской Республики

Учебный предмет / Дисциплина / МДК

ОТЧЕТ

по лабораторным работам

ЛР.Ип5-21.18.МДК.11.01.00.ОТ

Выполнил студент 3 курса, группы Ип5-21

Софронов Евгений Владимирович

(Фамилия И. О.)

(подпись) (чч.мм.гггг)

Преподаватель

(Фамилия И. О.)

Зачтено

(чч.мм.гггг)

Подпись

(подпись) (расшифровка подписи)

2023

# Лабораторная работа № 3

Хранимые процедуры

Цель: научиться создавать хранимые процедуры в среде SQL Server Management Studio Express.

Программное обеспечение: SQL Server Management Studio Express 2008 R2.

Ход работы:

Задание 1: Создать хранимую процедуру "Специальность\_Insert" добавления в таблицу "Специальности" информации о специальности.

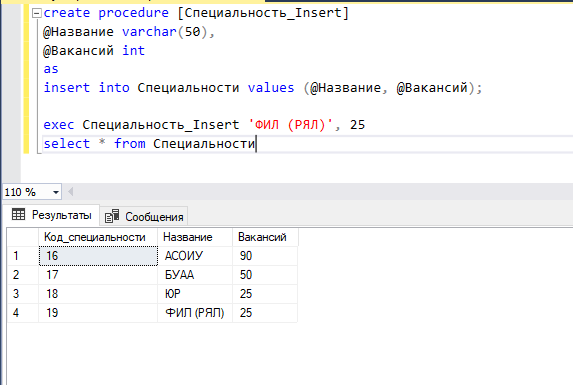


Рисунок 3.1 – Задание 1

Задание 2: Создать хранимую процедуру "Специальность\_Delete" удаления из таблицы "Специальности" информации о специальности.

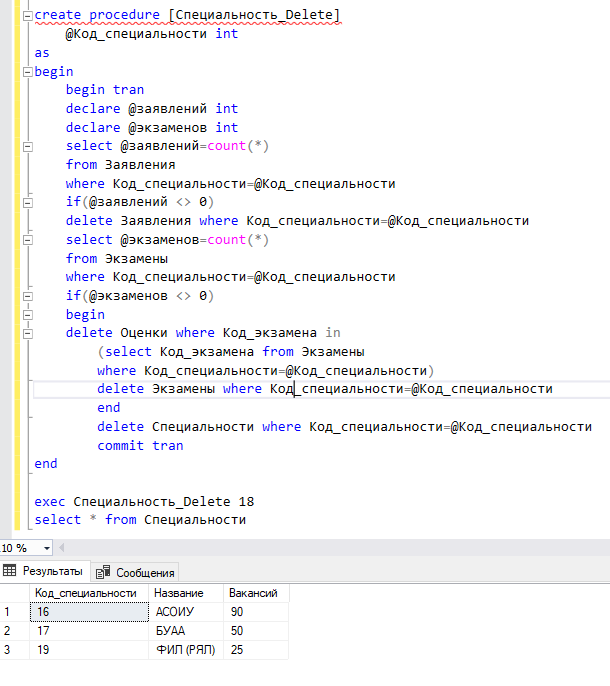


Рисунок 3.2 – Задание 2

Задание 3: Самостоятельно создать хранимые процедуры:

1) Создайте хранимую процедуру "Дисциплина\_Insert" добавления в таблицу "Дисциплины" информации о дисциплине экзамена.

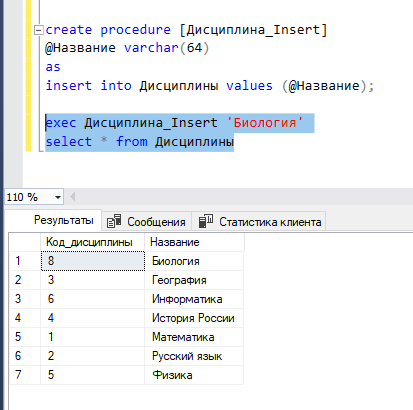


Рисунок 3.3 – Задание 3.1

2) Создайте хранимую процедуру "Абитуриент\_Insert" добавления в таблицу "Абитуриенты" информации об абитуриенте.

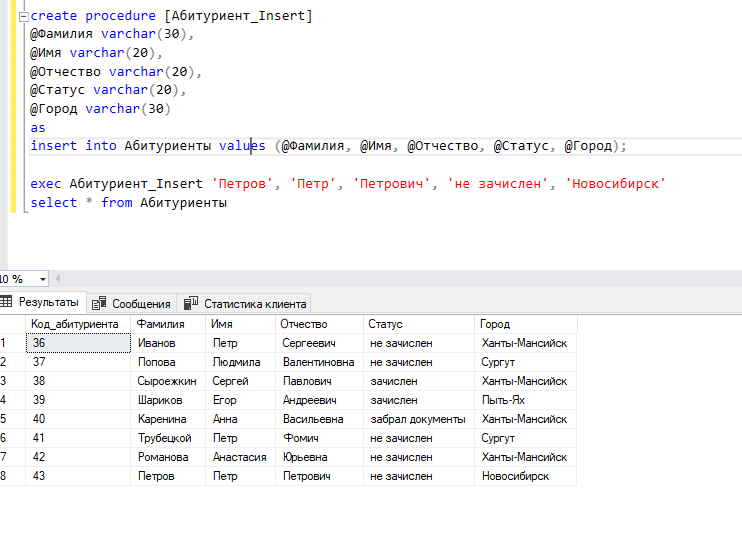


Рисунок 3.4 – Задание 3.2

3) Создайте хранимую процедуру "Заявление\_Insert" добавления в таблицу "Заявления" информации о заявлении абитуриента на специальность. Статус заявления - "не зачислен".

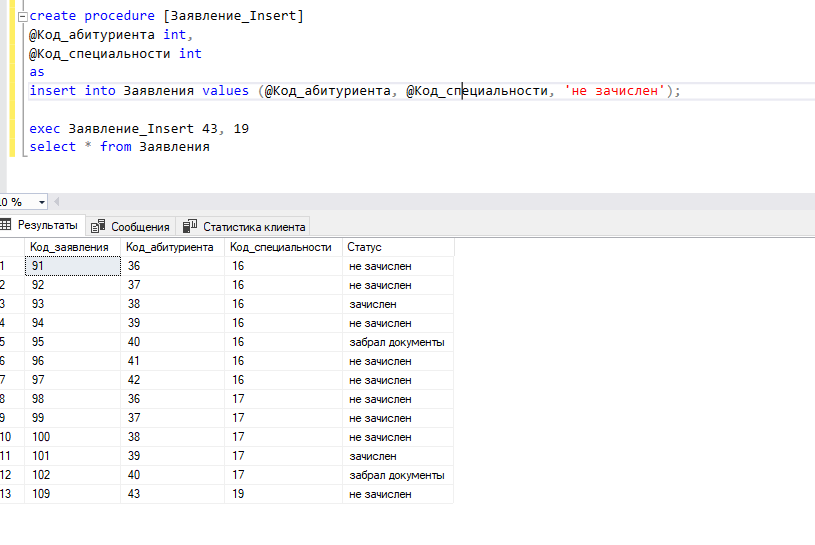


Рисунок 3.5 – Задание 3.3

4) Создайте хранимую процедуру "Дисциплина\_Delete" удаления из таблицы "Дисциплины" информации о дисциплине.

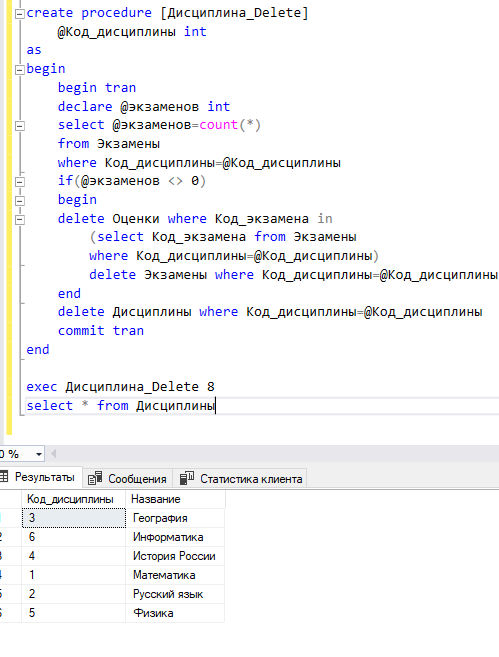


Рисунок 3.6 – Задание 3.4

5) Создайте хранимую процедуру "Абитуриент\_Delete" удаления из таблицы "Абитуриенты" информации об абитуриенте. Удаление возможно только в том случае, если статус абитуриента не равен "зачислен".

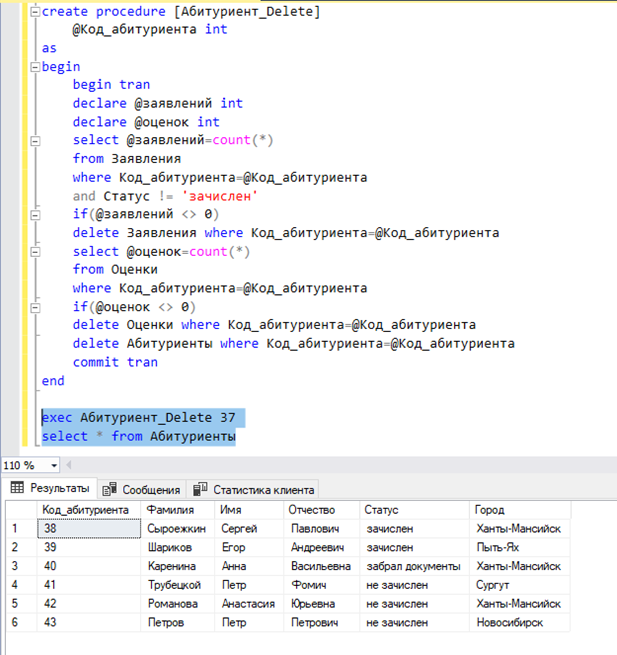


Рисунок 3.7 – Задание 3.5

6) Создайте хранимую процедуру "Экзамен\_Delete" удаления из таблицы "Экзамены" информации об экзамене. Если по экзамену есть оценки, то они должны быть удалены из таблицы "Оценки".

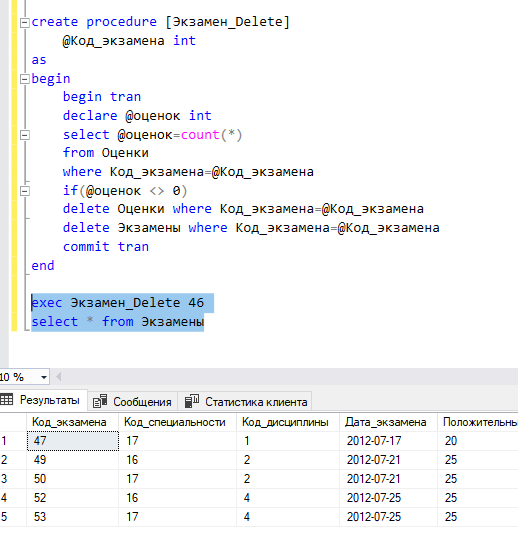


Рисунок 3.8 – Задание 3.6

7) Создайте хранимую процедуру "Специальность\_Update" увеличения количества вакансий на всех специальностях на 10%.

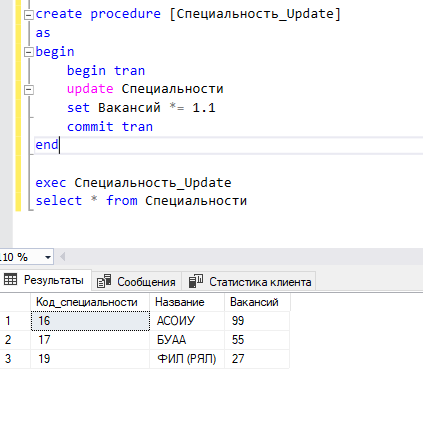


Рисунок 3.9 – Задание 3.7

8) Создайте хранимую процедуру "Экзамен\_Update" изменения даты экзамена.

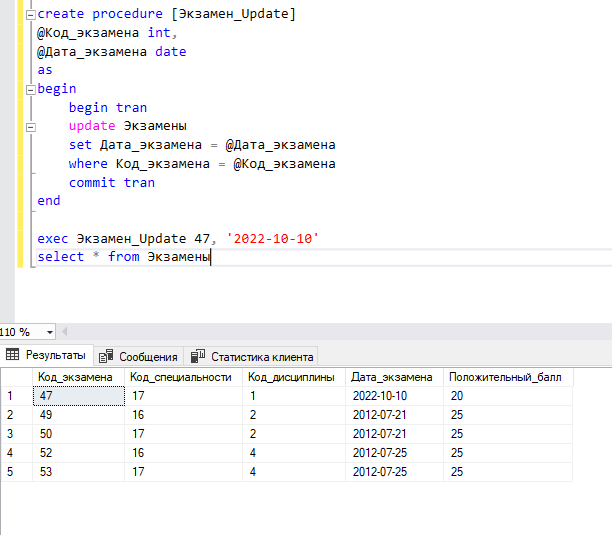


Рисунок 3.10 – Задание 3.8

9) Создайте хранимую процедуру "Абитуриент\_Update" обновления статуса абитуриента в таблицах "Абитуриент" и "Заявления" на "зачислен".

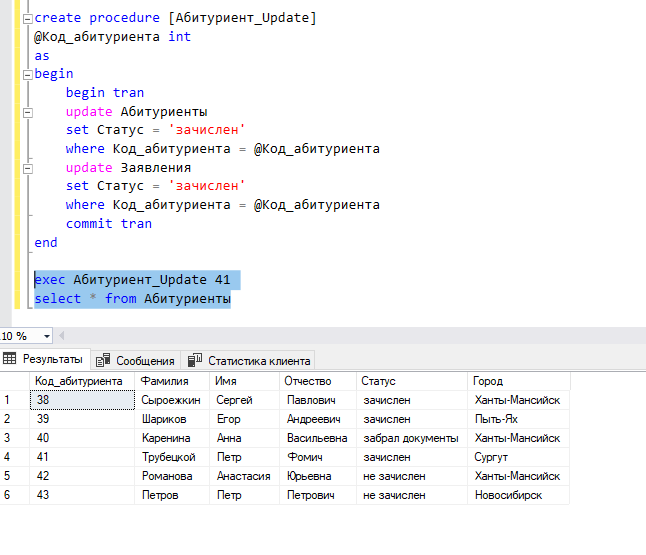


Рисунок 3.11 – Задание 3.9

Вывод: научились создавать хранимые процедуры в среде SQL Server Management Studio Express.

# Лабораторная работа № 4

Триггеры

Цель: научиться создавать триггеры в среде SQL Server Management Studio Express.

Программное обеспечение: SQL Server Management Studio Express 2008 R2.

Ход работы:

Задание 1: Создать триггер “Заявление\_OnInsert” на событие INSERT для таблицы “Заявления”. Триггер должен проверять, что абитуриент не может подать более трех заявлений на разные специальности. Если абитуриент уже подал три заявления, то вызывается откат транзакции по добавлению заявления в таблицу “Заявления”.

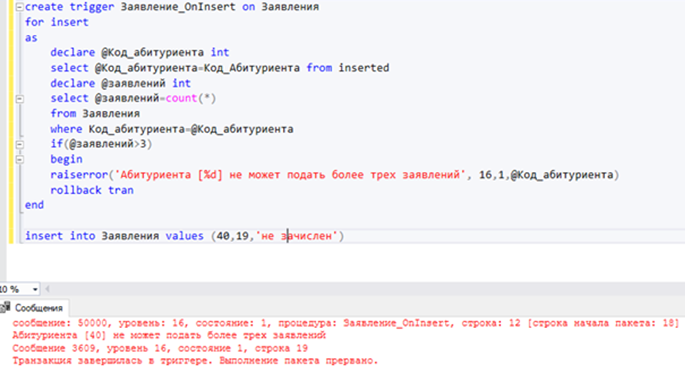


Рисунок 4.1 – Задание 1

Задание 2: Создать триггер “Абитуриент\_OnUpdate”, который вызывается при обновлении таблицы “Абитуриенты”. Триггер используется для записи в таблицу “Журнал” сообщения об изменения статуса абитуриента.

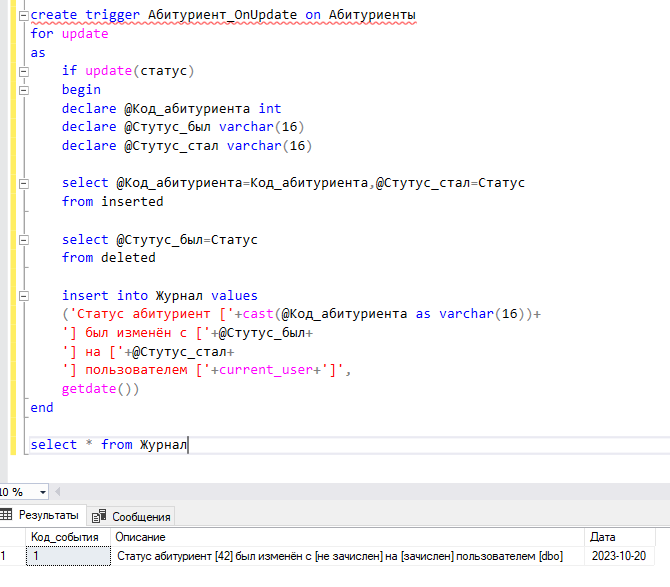


Рисунок 4.2 – Задание 2

Задание 3: Создайте триггер “Экзамен\_OnInsert” на событие INSERT для таблицы “Экзамены”. Триггер должен проверять, что для специальности в один день проводится лишь один экзамен. Если на дату, на которую добавляется экзамен, уже назначен для этой специальности экзамен, то должен вызываться откат транзакции по добавлению экзамена в таблицу “Экзамены”.

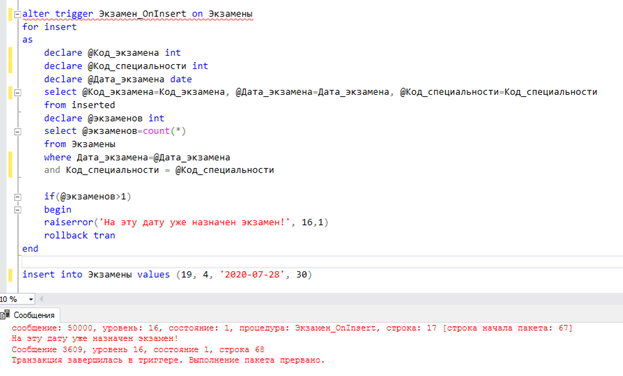


Рисунок 4.3 – Задание 3

Задание 4: Создайте триггер “Специальность\_OnUpdate” на событие UPDATE для таблицы “Специальности”. Триггер должен записывать в таблицу “Журнал” сообщение об изменении количества вакансий по специальности.

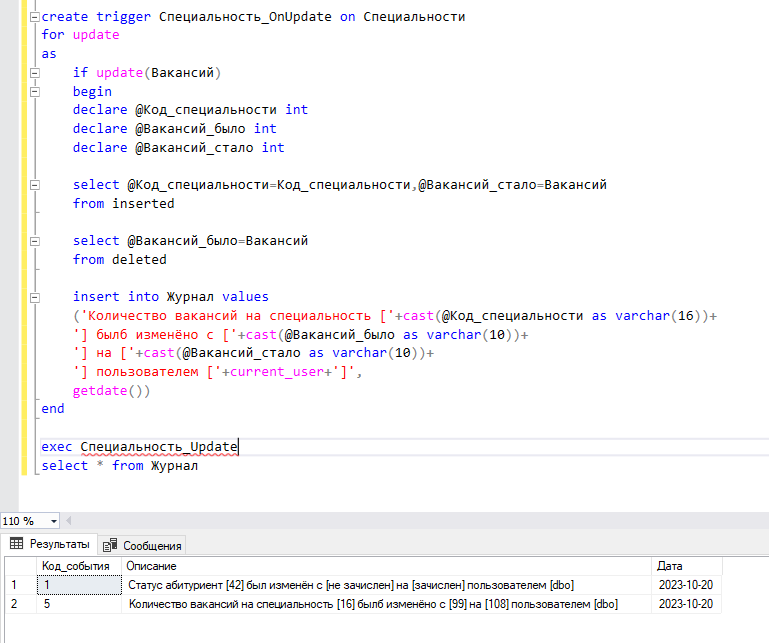


Рисунок 4.4 – Задание 4

Задание 5: Создайте триггер “Оценка\_OnInsert” на событие INSERT для таблицы “Оценки”. Триггер должен проверять, что у абитуриента, оценка которого добавляется, есть заявление на специальность, для которой проводился экзамен. Если добавляемая оценка абитуриента связана с экзаменом по специальности, на которую абитуриент не подавал заявления, то должен вызываться откат транзакции по добавлению оценки в таблицу “Оценки”

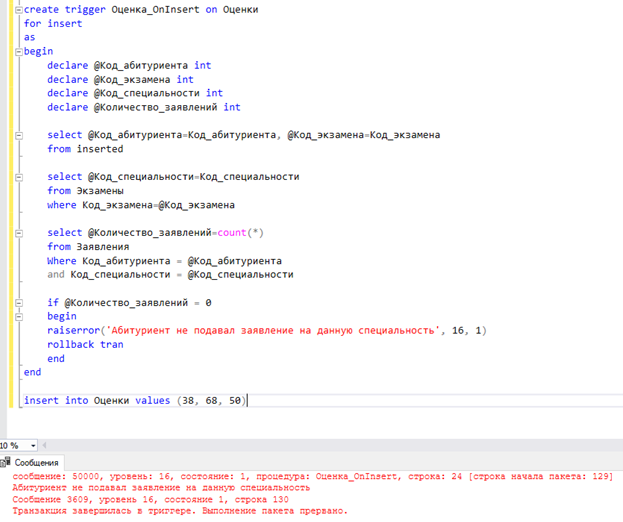


Рисунок 4.5 – Задание 5

Вывод: научились создавать триггеры в среде SQL Server Management Studio Express.

Лабораторная работа № 5

Создание и использование пользовательских функций.

Цель: приобрести навыки создания и вызова пользовательских функций.

Программное обеспечение: SQL Server Management Studio

Ход работы:

Задание 1. Создайте пользовательскую функцию, возвращающую ФИО абитуриента. В качестве параметра используйте столбец Код\_абитуриента.

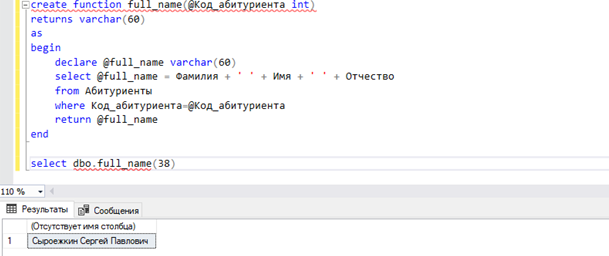


Рисунок 5.1 – Задание 1

Задание 2. Создайте пользовательскую функцию, возвращающую средний балл абитуриентов по заданной специальности. В качестве параметра используйте код специальности.

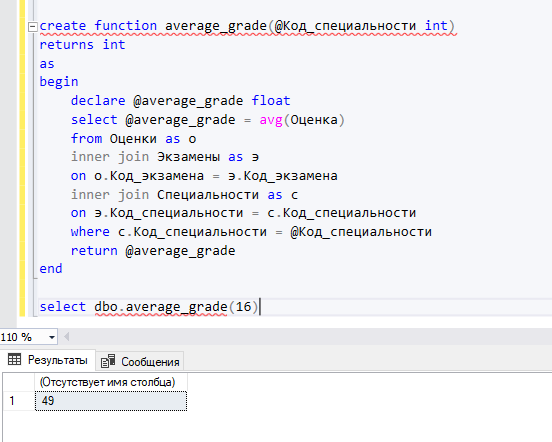


Рисунок 5.2 – Задание 2

Задание 3. Создайте пользовательскую функцию, возвращающую количество заявлений заданного абитуриента. В качестве параметра используйте код абитуриента.

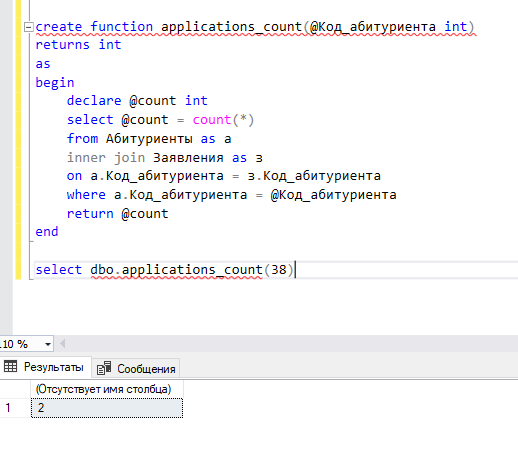


Рисунок 5.3 – Задание 3

Задание 4. Создайте пользовательскую функцию, возвращающую набор данных: ФИО абитуриента, Название специальности, Количество сданных экзаменов. (Используйте запрос из представления в лабораторной работе № 3)

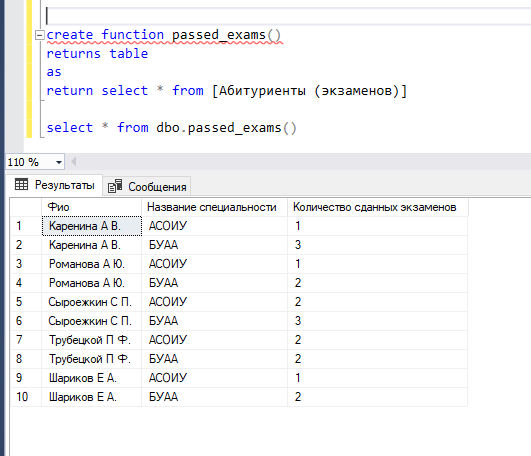


Рисунок 5.4 – Задание 4

Задание 5. Создайте пользовательскую функцию, возвращающую набор данных: Название специальности, Дисциплина, Средний балл абитуриента.

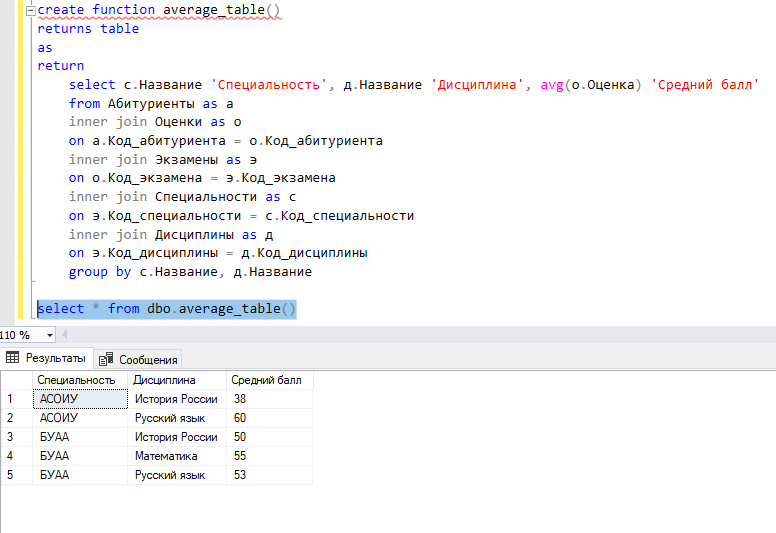


Рисунок 5.5 – Задание 5

Вывод: приобрели навыки создания и вызова пользовательских функций.

Лабораторная работа № 6

Transact-SQL

Цель: научиться экспортировать базы данных MS SQL Server в формат базы данных MS Access.

Программное обеспечение: SQL Server Management Studio, MS Office

Задача 1.1: Требуется подсчитать, сколько раз символ или подстрока встречаются в заданной строке.

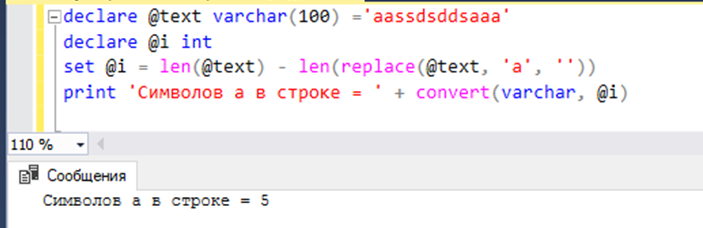


Рисунок 6.1 – Задача 1.1

Задача 1.2: Требуется преобразовать ФИО в инициалы. Например, если возьмем ФИО «Иванов Иван Иванович», то необходимо получить: Иванов И.И.

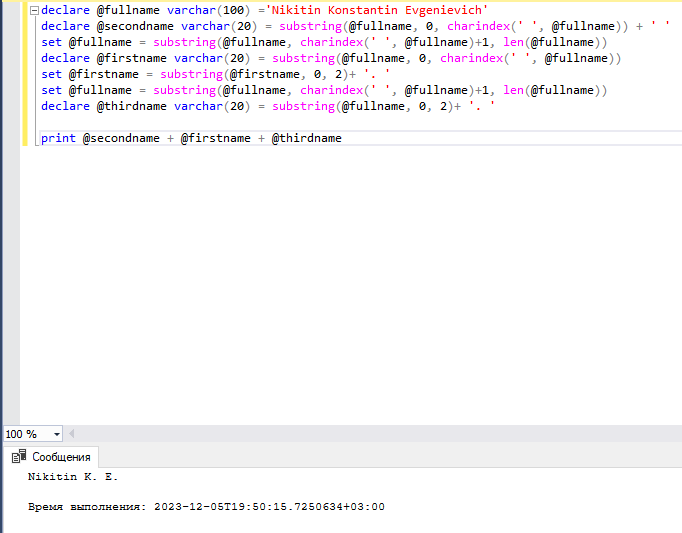


Рисунок 6.2 – Задача 1.2

Задача 2.1: Требуется подсчитать количество дней в текущем году.

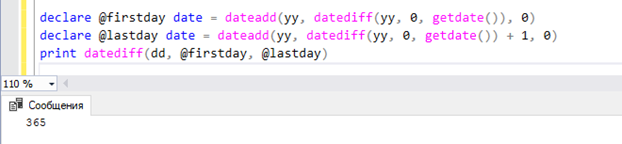


Рисунок 6.3 – Задача 2.1

Задача 2.2: Требуется разложить текущую дату на шесть частей: день, месяц, год, секунды, минуты и часы. Результаты должны быть возвращены в численном виде.

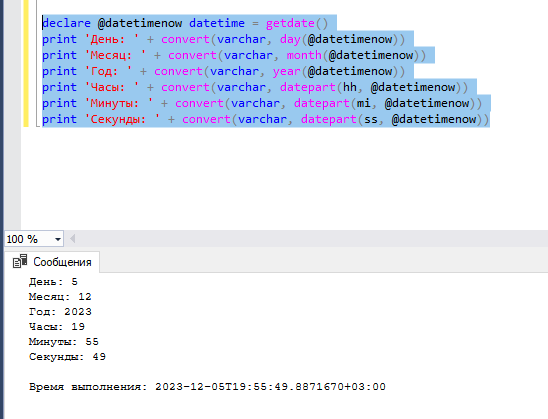


Рисунок 6.4 – Задача 2.2

Задача 2.3: Требуется получить первый и последний дни текущего месяца.

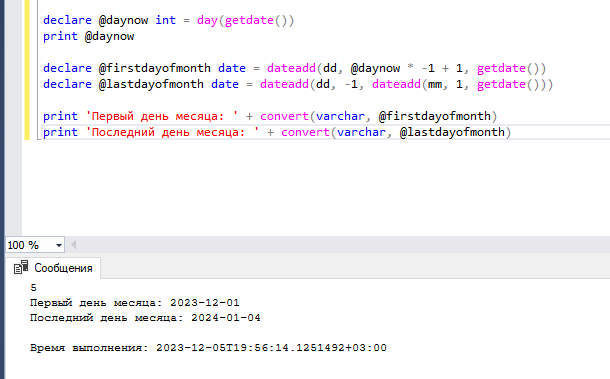


Рисунок 6.5 – Задача 2.3

Вывод: научились работать с Transact-SQL.

Лабораторная работа №7

Тема: Transact-SQL.

Цель работы: изучить основные операторы и алгоритмические конструкции Transact-SQL.

Задание. Выполнить SQL-запросы.

Последовательность выполнения заданий:

1. Выполнение заданий:

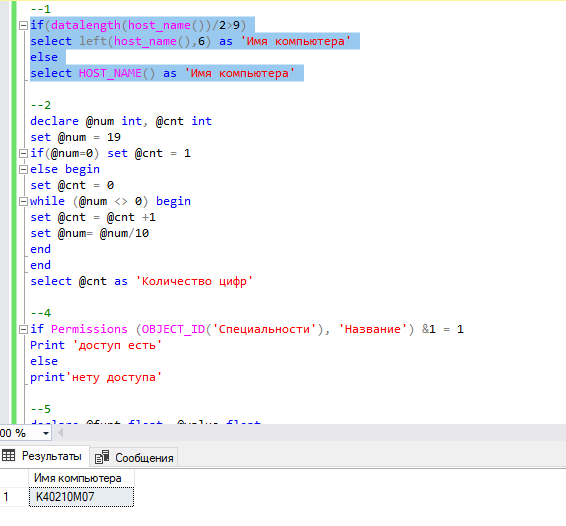


Рисунок 7.1 – Запрос 1

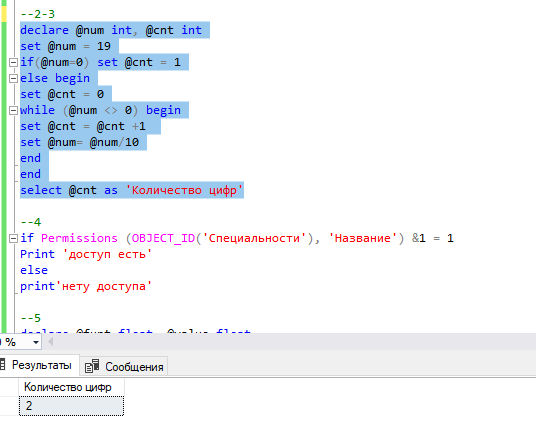


Рисунок 7.2 – Запрос 2

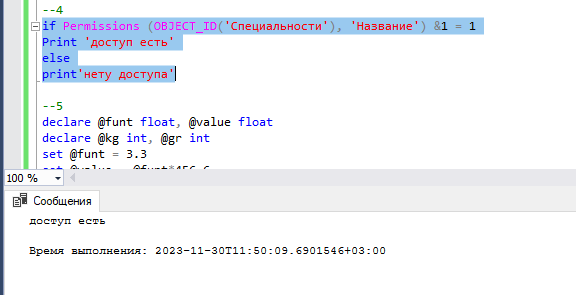


Рисунок 7.3 – Запрос 3

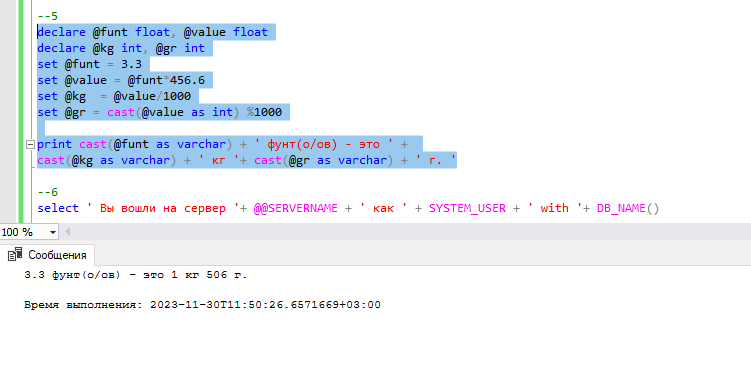


Рисунок 7.4 – Запрос 4

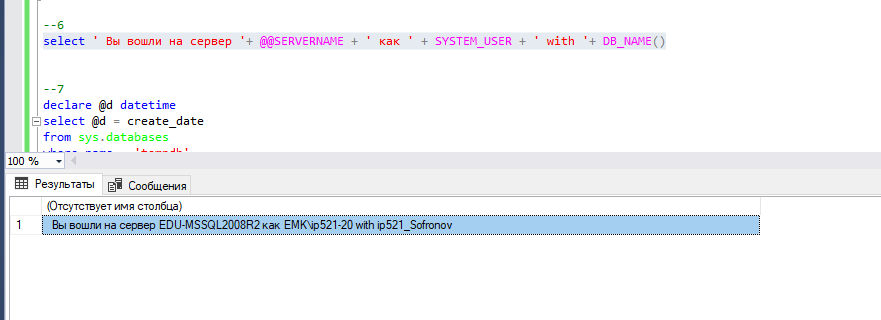


Рисунок 7.5 – Запрос 5

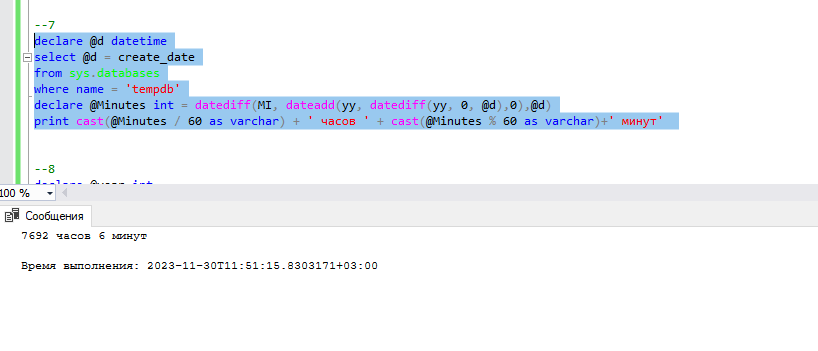


Рисунок 7.6 – Запрос 6

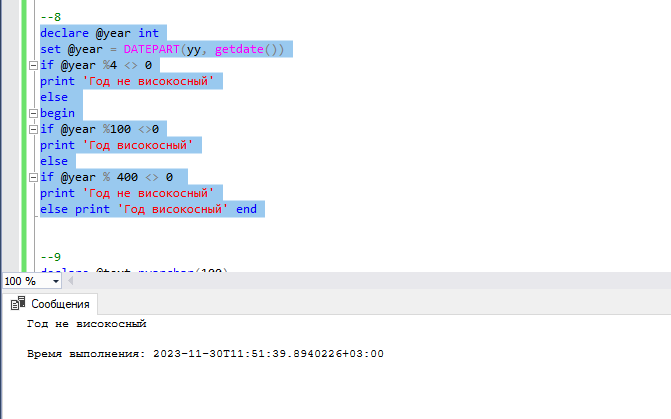


Рисунок 7.7 – Запрос 7

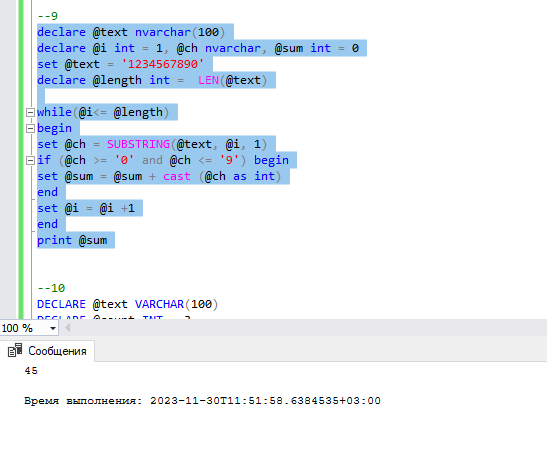


Рисунок 7.8 – Запрос 8

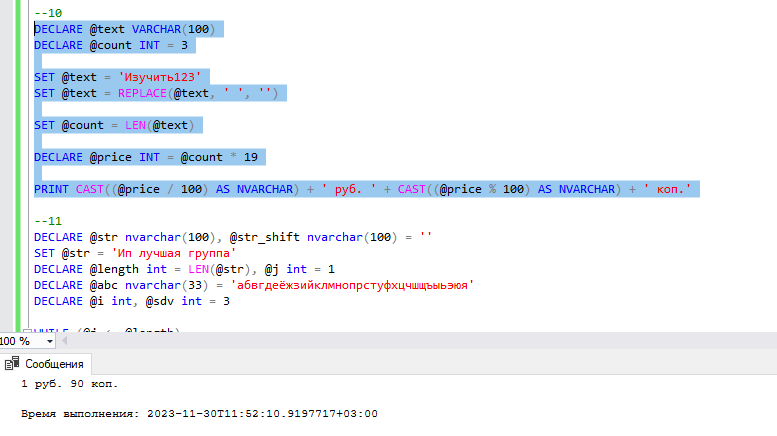


Рисунок 7.9 – Запрос 9

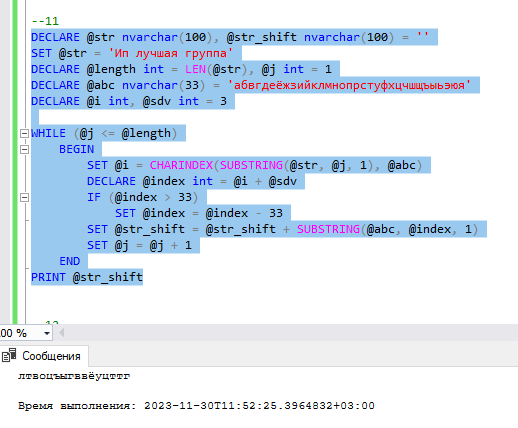


Рисунок 7.10 – Запрос 10

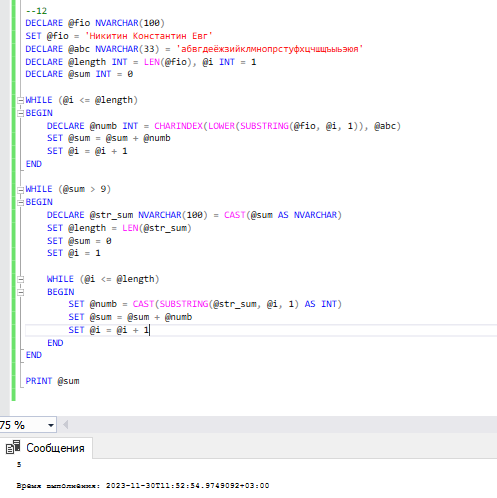


Рисунок 7.11 – Запрос 11

Вывод: изучили основные операторы и алгоритмические конструкции Transact-SQL.

ЛАБОРАТОРНАЯ РАБОТА № 8.

Система безопасности SQL Server.

Цель: научиться разграничивать доступ к данным в среде SQL Server Management Studio Express (использовать системные хранимые процедуры для управления именами входа MS SQL Server и пользователями баз данных, а также разрешать и запрещать выполнение определенных действий некоторому пользователю).

Программное обеспечение: SQL Server Management Studio Express

Задание 1: создать пользователя базы данных.

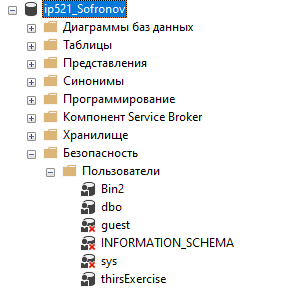


Рисунок 8.1 –Задание 1

Задание 2: Создать роль “Оператор” в SQL Server Management Studio

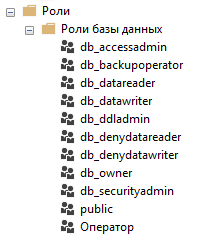


Рисунок 8.2 – Задание 2

Задание 3: Настроить разрешения для роли “Оператор” с использованием

оператора GRANT.

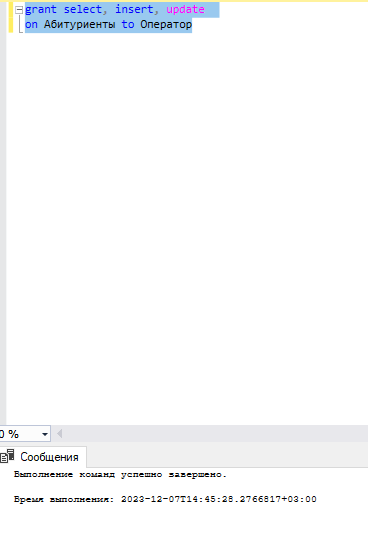


Рисунок 8.3 – Задание 3

Задание 4: Настроить разрешения для роли “Оператор” с помощью

графического интерфейса в SQL Server Management Studio

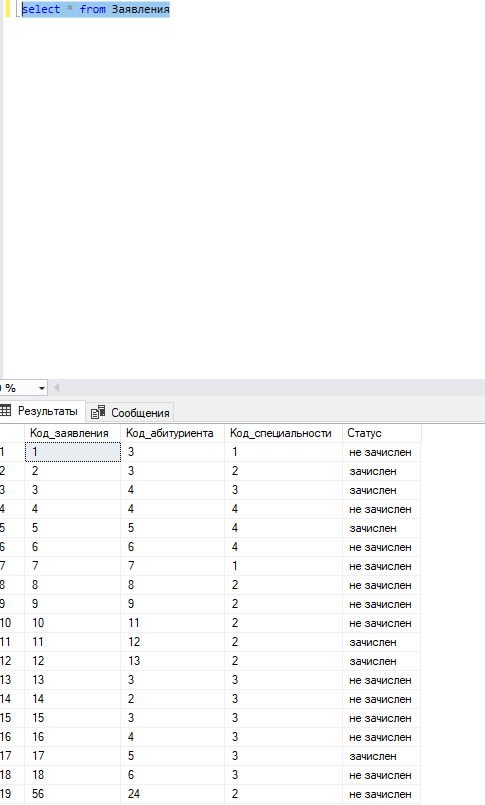


Рисунок 8.4 – Задание 4

Задание 5: Создайте роль “Методист” и предоставьте данной роли разрешение на выполнение всех действий с таблицами “Абитуриенты”, “Заявления”, “Оценки” и разрешение на выполнение оператора select на всех остальных таблицах.

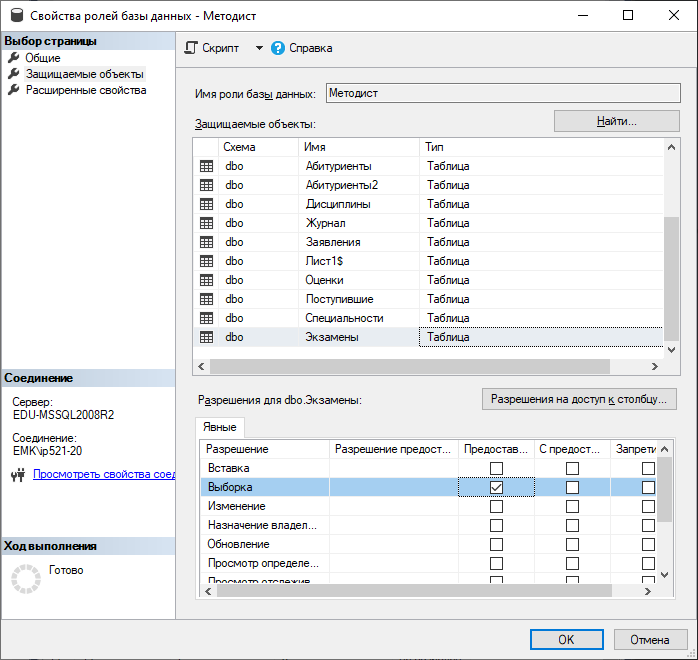


Рисунок 8.5 – Задание 5

Задание 6: Создайте и настройте учетную запись пользователя с помощью кода на языке SQL

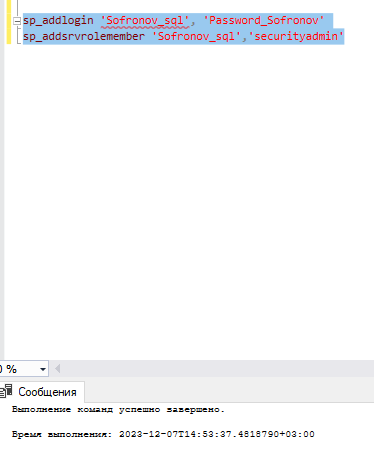


Рисунок 8.6 – Задание 6

Задание 7: Определите список ролей своей базы данных и членов роли

db\_owner.

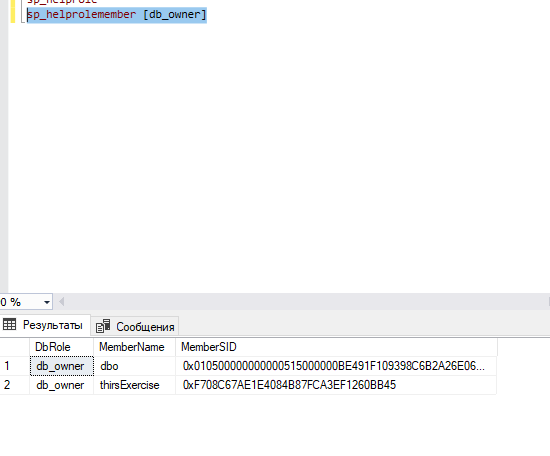


Рисунок 8.7 – Задание 7

Задание 8: Создайте нового пользователя базы данных для логина

Damir\_SQL.

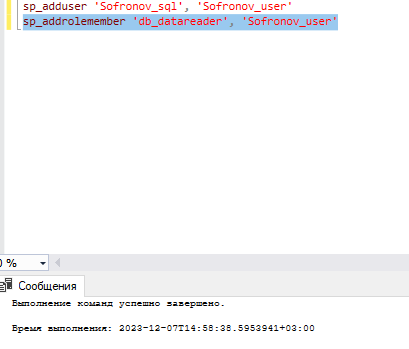


Рисунок 8.8 – Задание 8

Задание 9: Настройте права доступа пользователю Damir\_User: предоставьте явным образом право только для выборки из таблицы Абитуриенты и обновления только полей Фамилия, Имя и Отчество этой таблицы.

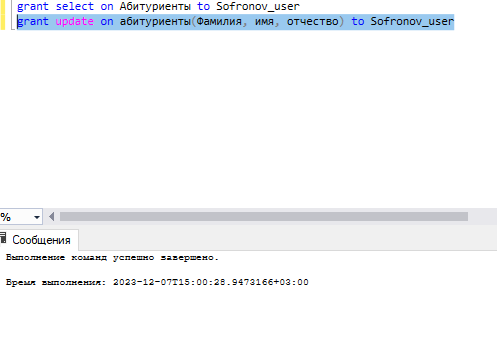


Рисунок 8.9 – Задание 9

Задание 10: Определите список всех ролей сервера и действия, разрешенные

пользователям роли dbcreator.

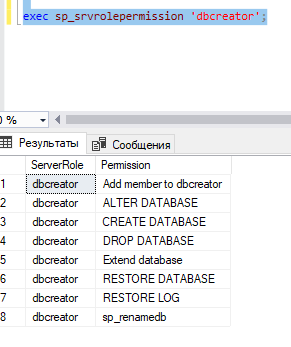


Рисунок 8.10 – Задание 10

Задание 11: Определите, пользователь какой роли имеет возможность

создания и удаления учетных записей для входа.

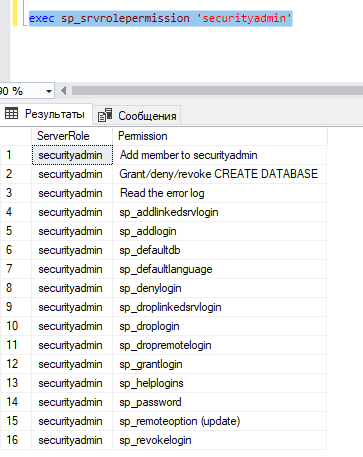


Рисунок 8.11 – Задание 11

Задание 12: Измените пароль учетной записи пользователя для входа с

помощью процедуры sp\_password.

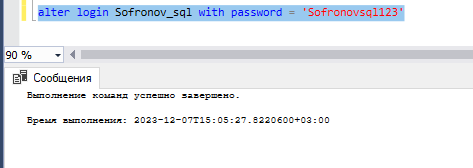


Рисунок 8.12 – Задание 12

Задание 13: Создайте собственную учетную запись для входа с подключением к своей базе данных, докажите правильность выполненных действий. Для созданной учетной записи присвойте права на создание и изменение баз данных, докажите правильность выполненных действий. Подключитесь к MS SQL Server, используя созданную учетную запись, и создайте еще одну учетную запись пользователя для входа, результат объясните.

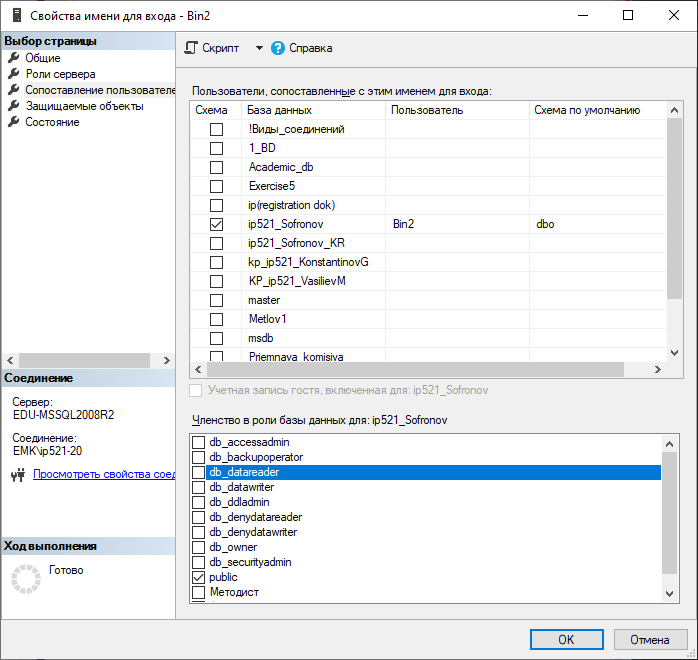


Рисунок 8.13 – Задание 13

Задание 14: Создайте пользователя manager своей базы данных на

основе созданной ранее учетной записи для входа. Докажите правильность

выполненных действий.

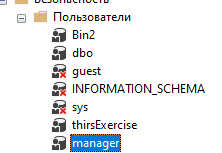


Рисунок 8.14 – Задание 14

Задание 15: Пользователю manager присвойте роль, обладающую только

возможностью просмотра содержимого своей базы данных

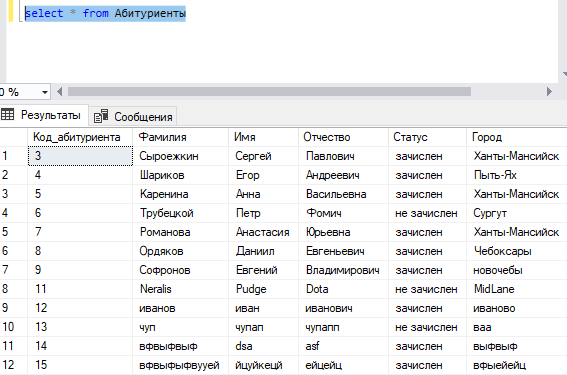


Рисунок 8.15 – Задание 15

Задание 16: Пользователю manager запретите просмотр данных своей

БД, присвоив необходимую роль. Как доказать правильность внесенных

изменений?

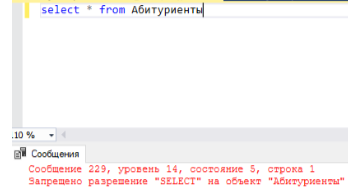


Рисунок 8.16 – Задание 16

Вывод: научились разграничивать доступ к данным в среде SQL Server Management Studio Express (использовать системные хранимые процедуры для управления именами входа MS SQL Server и пользователями баз данных, а также разрешать и запрещать выполнение определенных действий некоторому пользователю).

**Лабораторная работа №9**

**Тема:** Экспорт и импорт базы данных

**Цель работы:** научиться экспортировать базы данных MS SQL Server в формат базы данных MS Access.

**Задание.** Импортировать данные из файлов Excel в SQL Server можно несколькими способами.

**Последовательность выполнения заданий:**

**1. Выполнение заданий:**

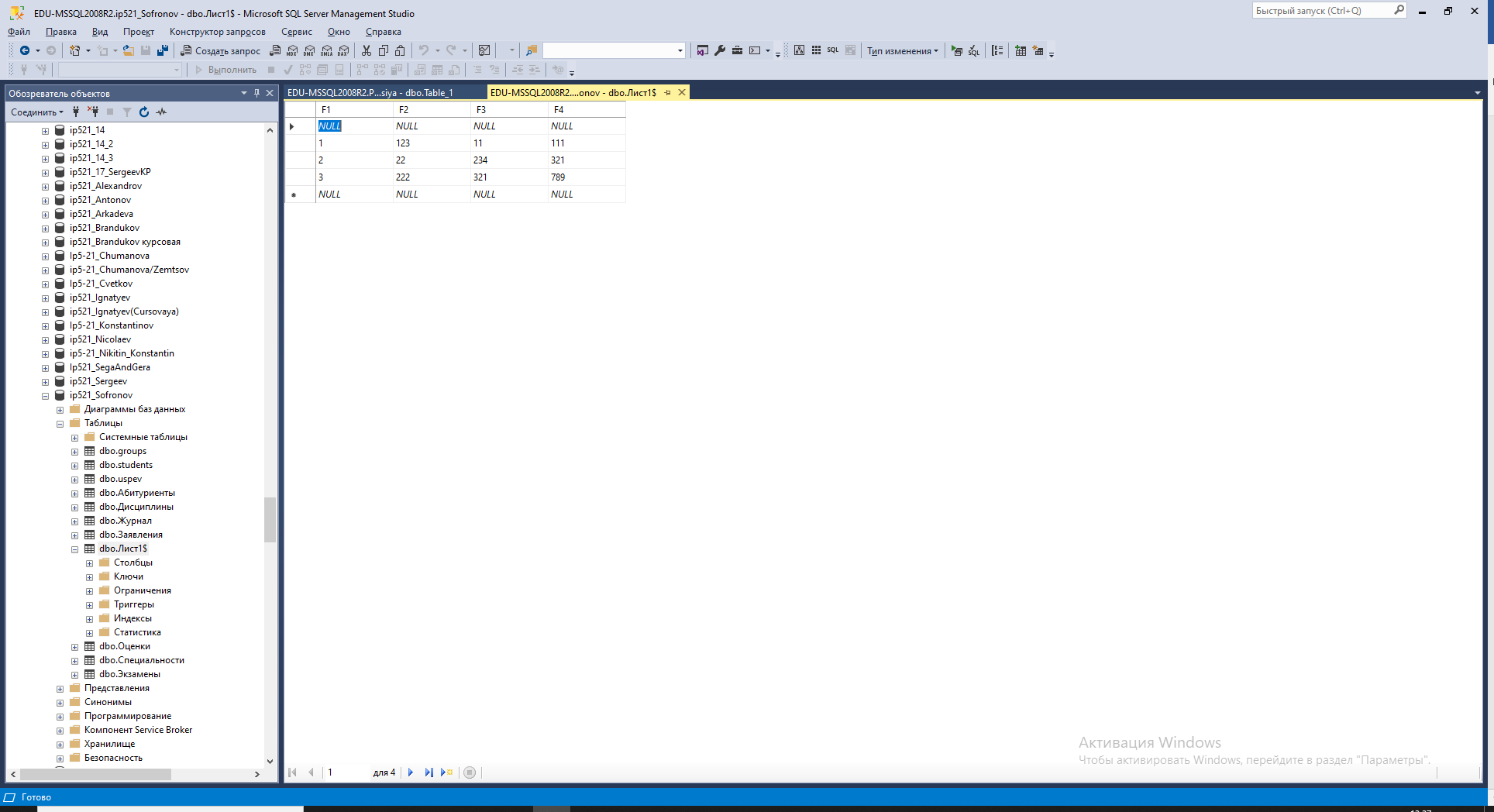


Рисунок 9.1 – Запрос 1

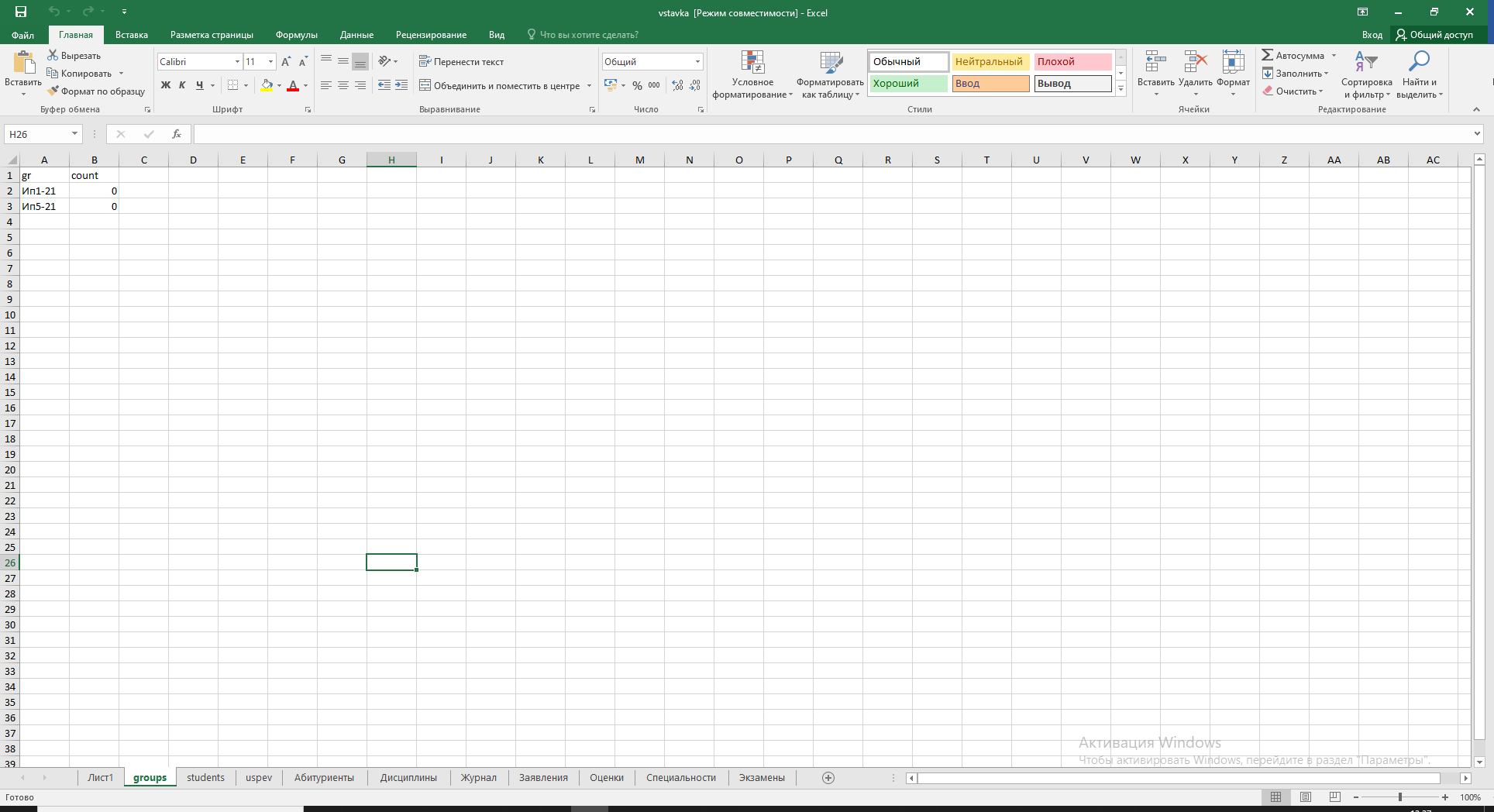


Рисунок 9.2 – Запрос 2

Ответьте на контрольные вопросы:

1. Из каких этапов состоит процесс экспортирования базы данных?

Процесс экспортирования базы данных обычно состоит из следующих этапов:

1. Выбор данных для экспорта.
2. Выбор формата экспорта.
3. Запуск процесса экспорта.
4. Настройка параметров экспорта.
5. Мониторинг процесса.
6. Проверка результатов.
7. Развертывание экспортированных данных.
8. Сопоставьте типы данных баз данных Access и SQL Server, заполнив таблицу 1. Для этого выполните импорт базы данных Access, содержащей данные типов данных, перечисленных в таблице 1, в базу данных MS SQL Server. В среде SQL Server посмотрите полученные типы данных в результате экспортирования.

Таблица 1 — Сопоставление типов данных

|  |  |
| --- | --- |
| Тип данных Access2003 | Тип данных SQL Server |
| Текстовый | NVARCHAR(MAX) |
| Поле MEMO | NVARCHAR(MAX) |
| Числовой | INT |
| Дата/время | DATETIME |
| Денежный | MONEY |
| Счетчик | INT IDENTITY(1,1) |
| Логический | BIT |
| Поле объекта OLE | VARBINARY(MAX) |

Вывод: научились экспортировать базы данных MS SQL Server в формат базы данных MS Access.

**Лабораторная работа №10**

**Тема:** Получение XML-данных с помощью операторов Transact-SQL

**Цель работы:** получить практические навыки использования различных режимов команды FOR XML для извлечения данных в формате XML.

**Задание.** C помощью команды FOR XML можно извлечь из БД XML-данные.

**Последовательность выполнения заданий:**

**1. Выполнение заданий:**

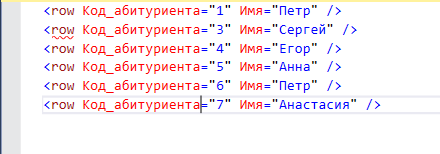


Рисунок 10.1 – Запрос 1

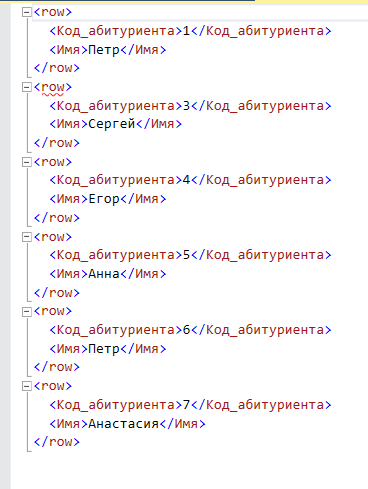


Рисунок 10.2 – Запрос 2

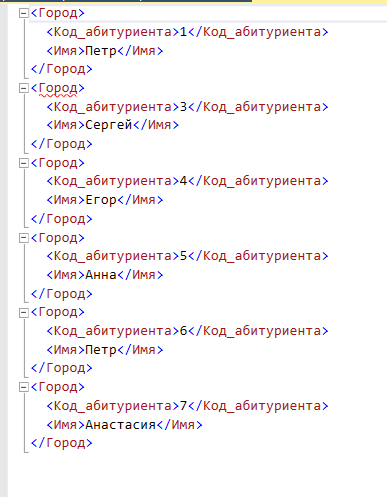


Рисунок 10.3 – Запрос 3

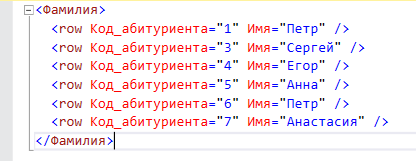


Рисунок 10.4 – Запрос 4

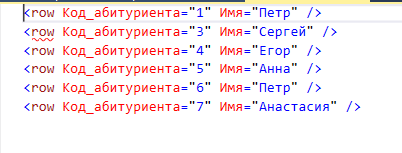


Рисунок 10.5 – Запрос 5



Рисунок 10.6 – Запрос 6



Рисунок 10.7 – Запрос 7



Рисунок 10.8 – Запрос 8



Рисунок 10.9 – Запрос 9

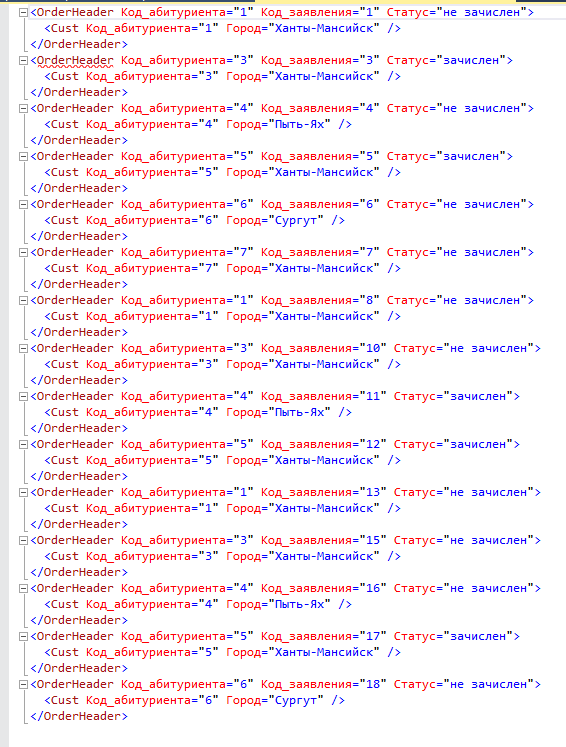


Рисунок 10.9 – Запрос 9



Рисунок 10.11 – Запрос 11

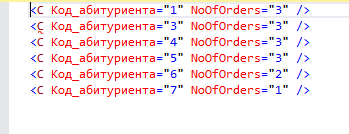


Рисунок 10.12 – Запрос 12

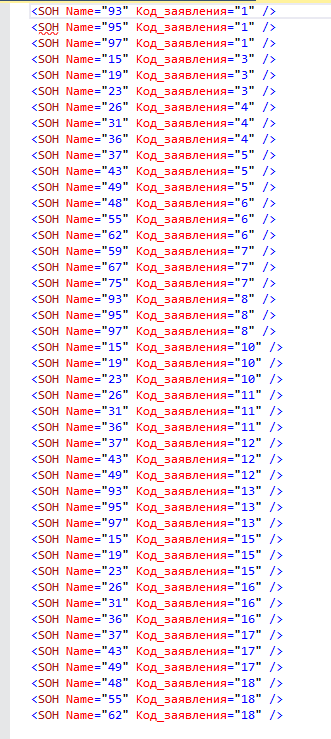


Рисунок 10.13 – Запрос 13

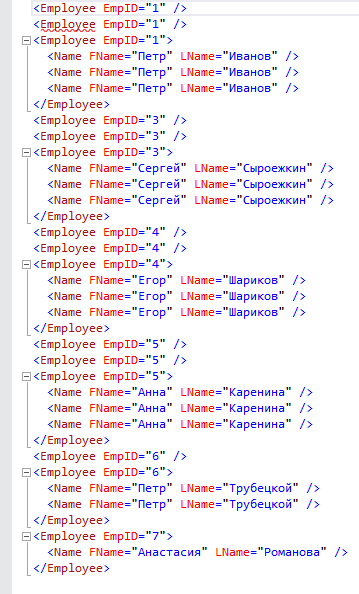


Рисунок 10.14 – Запрос 14

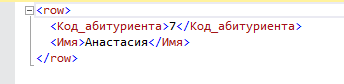


Рисунок 10.15 – Запрос 15

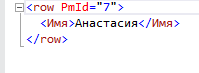


Рисунок 10.16 – Запрос 16

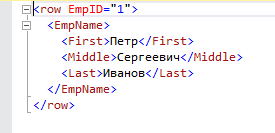


Рисунок 10.17 – Запрос 17



Рисунок 10.18 – Запрос 18

Код программы:

--1

SELECT Код\_абитуриента, Имя

FROM [dbo].[Абитуриенты]

FOR XML RAW

--2

SELECT Код\_абитуриента, Имя

FROM [dbo].[Абитуриенты]

FOR XML RAW, ELEMENTS

--3

SELECT Код\_абитуриента, Имя

FROM [dbo].[Абитуриенты]

FOR XML RAW ('Город'), ELEMENTS

--4

SELECT Код\_абитуриента, Имя

FROM [dbo].[Абитуриенты]

FOR XML RAW, ROOT('Фамилия')

--5

SELECT Код\_абитуриента, Имя

FROM [dbo].[Абитуриенты]

FOR XML RAW, TYPE

--6

SELECT Код\_абитуриента, Имя

FROM [dbo].[Абитуриенты]

FOR XML RAW, ELEMENTS XSINIL

--7

SELECT Код\_абитуриента, Имя

FROM [dbo].[Абитуриенты]

FOR XML RAW, XMLSCHEMA

--8

SELECT Код\_абитуриента, Имя

FROM [dbo].[Абитуриенты]

FOR XML RAW, XMLSCHEMA ('urn:example.com')

--9

SELECT Cust.Код\_абитуриента, OrderHeader.Код\_абитуриента,

OrderHeader.Код\_заявления,OrderHeader.Статус, Cust.Город

FROM Абитуриенты Cust, Заявления OrderHeader

WHERE Cust.Код\_абитуриента = OrderHeader.Код\_абитуриента

ORDER BY Cust.Код\_абитуриента

FOR XML AUTO

--10

SELECT OrderHeader.Код\_абитуриента,OrderHeader.Код\_заявления,

OrderHeader.Статус, Cust.Код\_абитуриента, Cust.Город

FROM Абитуриенты Cust, Заявления OrderHeader

WHERE Cust.Код\_абитуриента = OrderHeader.Код\_абитуриента

FOR XML AUTO

--11

SELECT

OrderHeader.Код\_абитуриента AS 'Cust.Код\_абитуриента',

OrderHeader.Код\_заявления AS 'Cust.Код\_заявления',

OrderHeader.Статус,

Cust.Код\_абитуриента,

Cust.Город

FROM

Абитуриенты AS Cust

JOIN

Заявления AS OrderHeader ON Cust.Код\_абитуриента = OrderHeader.Код\_абитуриента

FOR XML AUTO, ELEMENTS

--12

SELECT C.Код\_абитуриента, COUNT(\*) AS NoOfOrders

FROM Абитуриенты AS C INNER JOIN Заявления AS SOKF

on C.Код\_абитуриента = SOKF.Код\_абитуриента

GROUP BY C.Код\_абитуриента

for xml auto

--13

SELECT P.Код\_экзамена+' '+P.Оценка as Name,SOH.Код\_заявления

FROM Абитуриенты AS C INNER JOIN Заявления AS SOH ON

C.Код\_абитуриента = SOH.Код\_абитуриента JOIN Оценки AS P ON P.Код\_абитуриента= C.Код\_абитуриента

FOR XML AUTO

--14

SELECT 1 as Tag,

NULL as Parent,

E.Код\_абитуриента as [Employee!1!EmpID],

NULL as [Name!2!FName],

NULL as [Name!2!LName]

FROM Абитуриенты AS E

INNER JOIN Заявления AS P

ON E.Код\_абитуриента = P.Код\_абитуриента

UNION ALL

SELECT 2 as Tag,

1 as Parent,

E.Код\_абитуриента,

Имя,

Фамилия

FROM Абитуриенты AS E

INNER JOIN Заявления AS P

ON E.Код\_абитуриента = P.Код\_абитуриента

ORDER BY [Employee!1!EmpID],[Name!2!FName]

FOR XML EXPLICIT

--15

SELECT Код\_абитуриента, Имя

FROM Абитуриенты

WHERE Код\_абитуриента=7

FOR XML PATH

--16

SELECT Код\_абитуриента as "@PmId", Имя

FROM Абитуриенты

WHERE Код\_абитуриента=7

FOR XML PATH

--17

SELECT e.Код\_абитуриента "@EmpID",

Имя "EmpName/First",

Отчество "EmpName/Middle",

Фамилия "EmpName/Last"

FROM Абитуриенты E, Заявления C

WHERE E.Код\_абитуриента = C.Код\_заявления

AND E.Код\_абитуриента=1

FOR XML PATH

--18

Select Код\_абитуриента as 'data()'

from Абитуриенты for xml path('')

**Контрольный вопросы**

1. Какой режим команды FOR XML возвращает результат в виде вложенных XML-элементов?

Ответ: Режим RAW команды FOR XML возвращает результат в виде вложенных XML-элементов.

1. Какой режим команды FOR XML представляет результирующую строку выборки как один элемент документа, а все колонки – как его атрибуты?

Ответ: Режим AUTO команды FOR XML представляет результирующую строку выборки как один элемент документа, а все колонки – как его атрибуты.

1. Какой режим команды FOR XML представляет возможность генерировать иерархию узлов XML?

Ответ: Режим ELEMENTS команды FOR XML представляет возможность генерировать иерархию узлов XML.

1. Для какого режима команды FOR XML набор строк (универсальная таблица) должен иметь определенный формат?

Ответ: Режим PATH команды FOR XML предполагает, что набор строк имеет определенный формат, такой как "родительский/дочерний", чтобы корректно построить иерархию XML-элементов.

Вывод: получили практические навыки использования различных режимов команды FOR XML для извлечения данных в формате XML.

**Лабораторная работа №11**

**Тема:** Извлечение данные из XML документа

**Цель работы:** получить практические навыки использования команды OPENXML для извлечения данных из XML документа.

**Последовательность выполнения заданий:**

* 1. **Выполнение заданий:**

**Задание 1:**

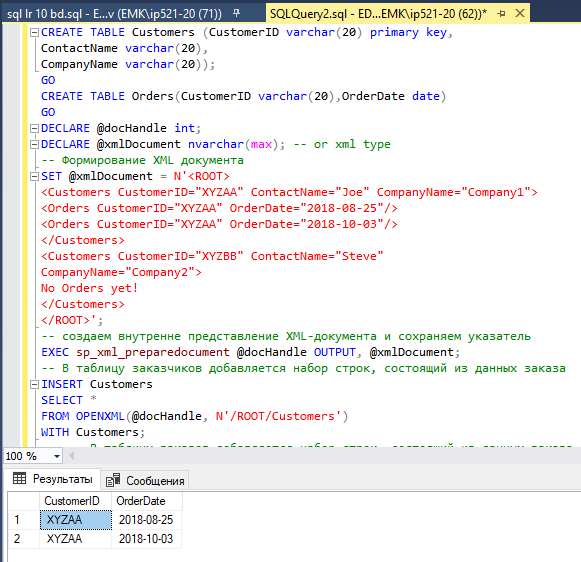
****

Рисунок 11.1 – Запрос 1

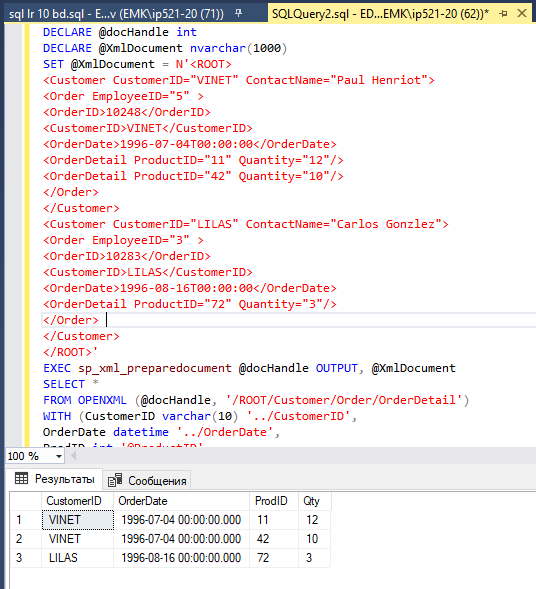


Рисунок 11.2 – Запрос 2

**Задание 2:**

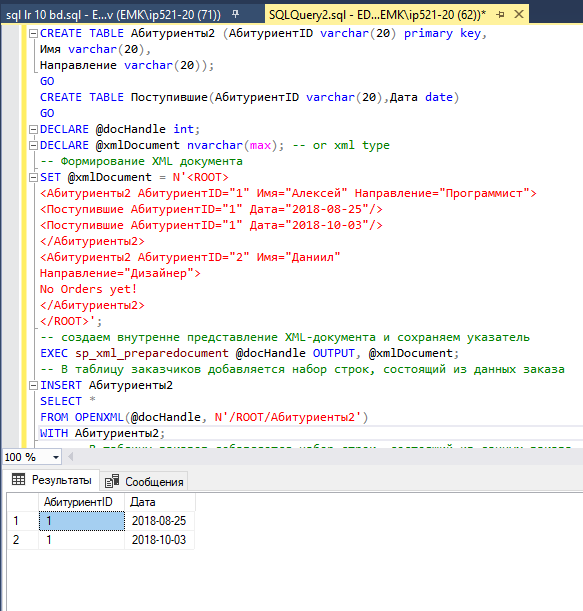


Рисунок 11.3 – Запрос 3

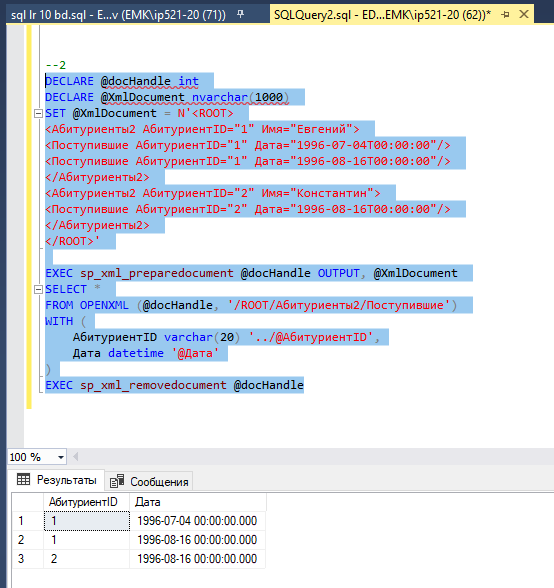


Рисунок 11.4 – Запрос 4

Код программы:

Запрос 3:

--1

CREATE TABLE Абитуриенты2 (АбитуриентID varchar(20) primary key,

Имя varchar(20),

Направление varchar(20));

GO

CREATE TABLE Поступившие(АбитуриентID varchar(20),Дата date)

GO

DECLARE @docHandle int;

DECLARE @xmlDocument nvarchar(max); -- or xml type

-- Формирование XML документа

SET @xmlDocument = N'<ROOT>

<Абитуриенты2 АбитуриентID="1" Имя="Алексей" Направление="Программист">

<Поступившие АбитуриентID="1" Дата="2018-08-25"/>

<Поступившие АбитуриентID="1" Дата="2018-10-03"/>

</Абитуриенты2>

<Абитуриенты2 АбитуриентID="2" Имя="Даниил"

Направление="Дизайнер">

No Orders yet!

</Абитуриенты2>

</ROOT>';

-- создаем внутренне представление XML-документа и сохраняем указатель

EXEC sp\_xml\_preparedocument @docHandle OUTPUT, @xmlDocument;

-- В таблицу заказчиков добавляется набор строк, состоящий из данных заказа

INSERT Абитуриенты2

SELECT \*

FROM OPENXML(@docHandle, N'/ROOT/Абитуриенты2')

WITH Абитуриенты2;

-- В таблицу заказов добавляется набор строк, состоящий из данных заказа

INSERT Поступившие

SELECT \*

FROM OPENXML(@docHandle, N'//Поступившие')

WITH Поступившие;

-- Извлечение данных из XML документа

SELECT \* FROM OPENXML(@docHandle, N'/ROOT/Абитуриенты2/Поступившие') WITH (

АбитуриентID nchar(5) '../@АбитуриентID',

Дата date

);

-- Удаление дескриптора XML документа

EXEC sp\_xml\_removedocument @docHandle;

Запрос 4:

--2

DECLARE @docHandle int

DECLARE @XmlDocument nvarchar(1000)

SET @XmlDocument = N'<ROOT>

<Абитуриенты2 АбитуриентID="1" Имя="Евгений">

<Поступившие АбитуриентID="1" Дата="1996-07-04T00:00:00"/>

<Поступившие АбитуриентID="1" Дата="1996-08-16T00:00:00"/>

</Абитуриенты2>

<Абитуриенты2 АбитуриентID="2" Имя="Константин">

<Поступившие АбитуриентID="2" Дата="1996-08-16T00:00:00"/>

</Абитуриенты2>

</ROOT>'

EXEC sp\_xml\_preparedocument @docHandle OUTPUT, @XmlDocument

SELECT \*

FROM OPENXML (@docHandle, '/ROOT/Абитуриенты2/Поступившие')

WITH (

АбитуриентID varchar(20) '../@АбитуриентID',

Дата datetime '@Дата'

)

EXEC sp\_xml\_removedocument @docHandle

Вывод: получили практические навыки использования команды OPENXML для извлечения данных из XML документа.

**Лабораторная работа №12**

**Тема:** разработка приложения базы данных.

**Цель работы:** научиться создавать объекты баз данных, научиться создавать приложение для просмотра данных из базы данных.

**Последовательность выполнения заданий:**

**Выполнение заданий:**

* + 1. **Программный код:**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.SqlClient;

namespace Kollege

{

public partial class Form1 : Form

{

private DataSet dataSet;

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

LoadData();

tabControl1.SelectedIndexChanged += tabControl1\_SelectedIndexChanged;

tabControl1.SelectedTab = tabPage1;

UpdateRecordCount();

lbRecordCount1.Text = $"Количество записей: {dataGridView1.Rows.Count - 1}";

}

private void statusStrip\_ItemClicked(object sender, ToolStripItemClickedEventArgs e)

{

}

private void tabControl1\_SelectedIndexChanged(object sender, EventArgs e)

{

UpdateRecordCount();

TabPage selectedTab = tabControl1.SelectedTab;

if (selectedTab == tabPage1)

lbRecordCount1.Text = $"Количество записей: {dataGridView1.Rows.Count-1}";

else if (selectedTab == tabPage2)

lbRecordCount1.Text = $"Количество записей: {dataGridView2.Rows.Count-1}";

else if (selectedTab == tabPage3)

lbRecordCount1.Text = $"Количество записей: {dataGridView3.Rows.Count-1}";

else if (selectedTab == tabPage4)

lbRecordCount1.Text = $"Количество записей: {dataGridView4.Rows.Count-1}";

else if (selectedTab == tabPage5)

lbRecordCount1.Text = $"Количество записей: {dataGridView5.Rows.Count-1}";

}

private void UpdateRecordCount()

{

// Определяем, какая вкладка выбрана

string selectedTable = tabControl1.SelectedTab?.Name;

// Проверяем, что dataSet существует и содержит выбранную таблицу

if (dataSet?.Tables[selectedTable] is DataTable dataTable)

{

// Обновляем счетчик записей в StatusStrip

lbRecordCount1.Text = $"Количество записей: {dataTable.Rows.Count}";

}

else

{

// Обработка случая, если dataSet или таблица отсутствуют

lbRecordCount1.Text = "Количество записей: 0";

}

}

private void LoadData()

{

dataSet = new DataSet();

using (SqlConnection conn = new SqlConnection(Properties.Settings.Default.ConString))

{

conn.Open();

SqlDataAdapter adapter1 = new SqlDataAdapter("SELECT \* FROM Абитуриенты", conn);

adapter1.Fill(dataSet, "Абитуриенты");

SqlDataAdapter adapter2 = new SqlDataAdapter("SELECT \* FROM Заявления", conn);

adapter2.Fill(dataSet, "Заявления");

SqlDataAdapter adapter3 = new SqlDataAdapter("SELECT \* FROM Экзамены", conn);

adapter2.Fill(dataSet, "Экзамены");

SqlDataAdapter adapter4 = new SqlDataAdapter("SELECT \* FROM Специальности", conn);

adapter2.Fill(dataSet, "Специальности");

SqlDataAdapter adapter5 = new SqlDataAdapter("SELECT \* FROM Оценки", conn);

adapter2.Fill(dataSet, "Оценки");

// Добавьте дополнительные адаптеры для других таблиц, если необходимо

conn.Close();

}

// Используйте dataSet.Tables["TableName"] для доступа к каждой таблице

dataGridView1.DataSource = dataSet.Tables["Абитуриенты"];

dataGridView2.DataSource = dataSet.Tables["Заявления"];

dataGridView1.DataSource = dataSet.Tables["Экзамены"];

dataGridView2.DataSource = dataSet.Tables["Специальности"];

dataGridView1.DataSource = dataSet.Tables["Оценки"];

// ... добавьте другие DataGridView для других таблиц, если необходимо

}

}

}

Задание 1:

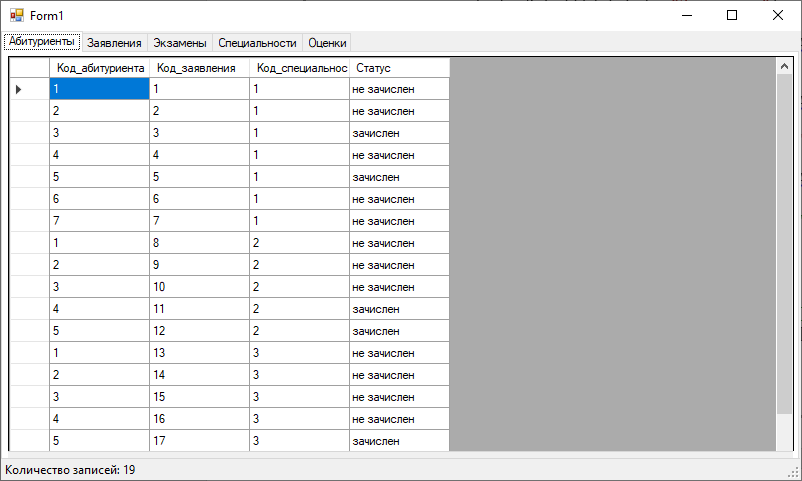


Рисунок 12.1 – Форма 1

**Вывод:** научились создавать объекты баз данных, научиться создавать приложение для просмотра данных из базы данных.

Лабораторная работа № 13

Тема: Разработка приложения для модификации данных.

Цель: научиться создавать объекты баз данных, научиться создавать приложение для просмотра данных из базы данных

Программное обеспечение: SQL Server Management Studio, MS Office

Ход работы:

Задание 1: Разработать приложение для добавления данных в таблицу «Абитуриенты» базы данных «Kollege»

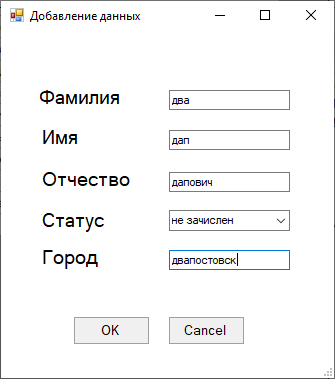


Рисунок 13.1 – Форма добавления

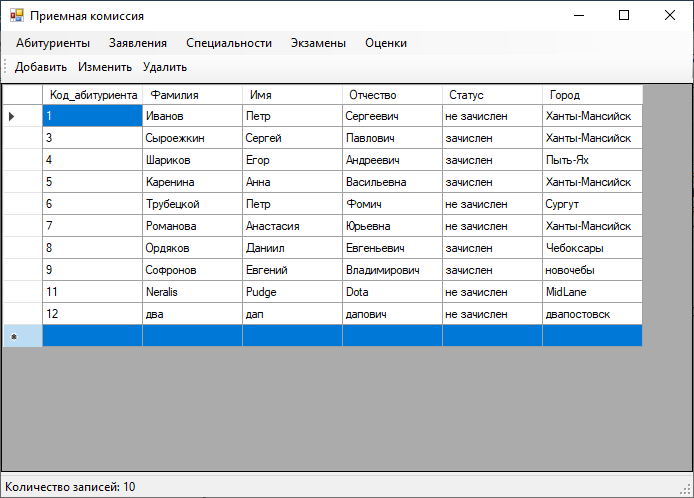


Рисунок 13.2 – После добавления

Задание 2: Разработать приложение для обновления данных в таблице «Абитуриенты» базы данных «Kollege».

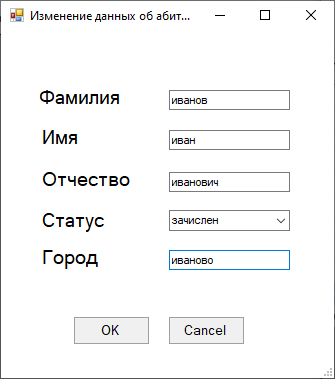


Рисунок 13.3 – Изменение записи

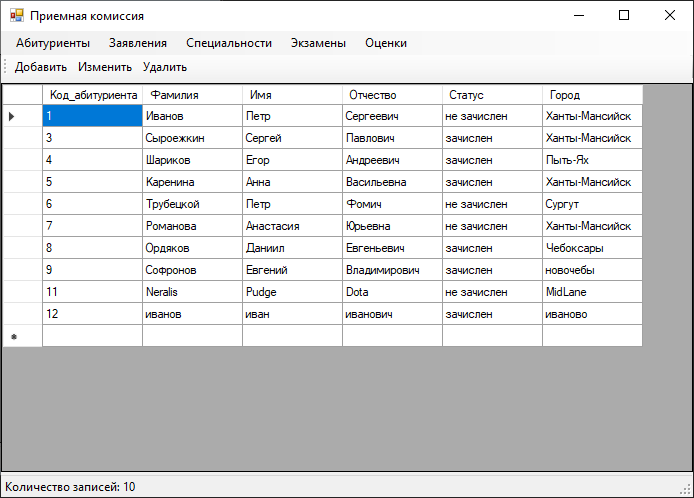


Рисунок 13.4 – Запись изменена

Задание 3: Разработать приложение для удаления данных в таблице «Абитуриенты» базы данных «Kollege».

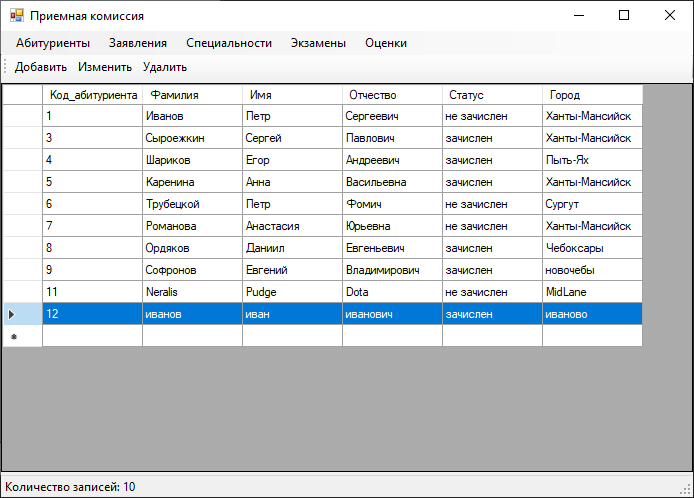


Рисунок 13.5 – Перед удалением записи

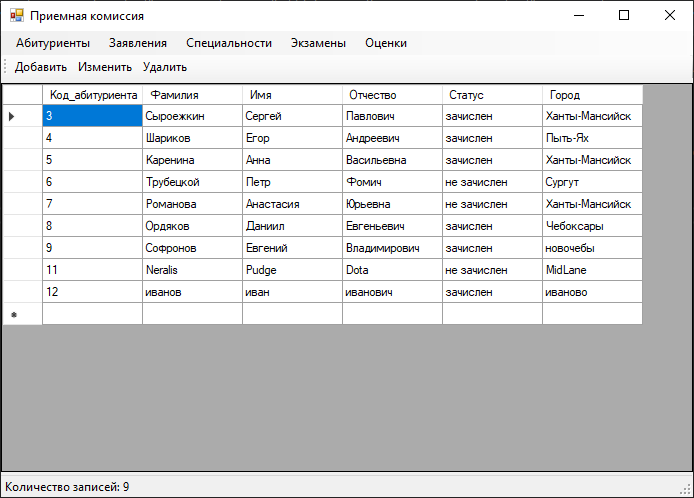


Рисунок 13.6 – После удаления записи

Вывод: научились создавать объекты баз данных, научились создавать приложение для просмотра данных из базы данных.

Лабораторная работа № 14

Тема: Разработка приложения с фильтрацией данных с помощью хранимых процедур.

Программное обеспечение: MS Visual Studio 2010

Цель: научиться создавать приложение, позволяющее с помощью хранимых процедур выполнять фильтрацию данных.

Ход работы:

Задание 1: Создайте хранимые процедуры: − Student\_Select\_Faculty – для выборки данных из таблицы Student с параметром для фильтрации по специальности; − Course\_Select – для получения списка дисциплин; − Grade\_Select\_Course – для выборки данных из таблицы Grade с параметром для фильтрации по дисциплине.

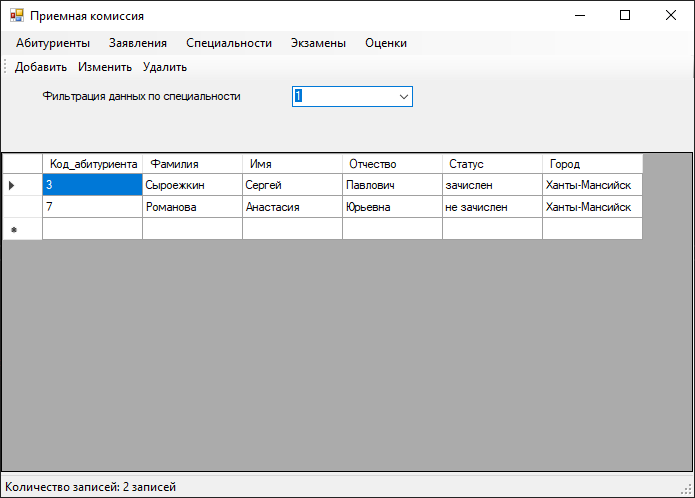


Рисунок 14.1 – Фильтрация данных по специальности

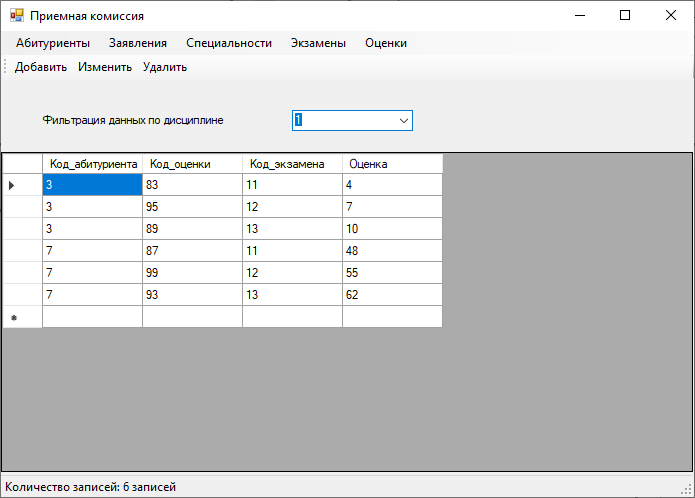


Рисунок 14.2 – Фильтрация данных по дисциплине

* 1. Программный код:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace lb\_12\_bd

{

public partial class Form1 : Form

{

private SqlConnection conn;

private string currentTableName;

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

conn = new SqlConnection(Properties.Settings.Default.Ip5\_21\_SofronovConnectionString);

currentTableName = "Абитуриенты";

LoadData(currentTableName);

}

private void абитуриентыToolStripMenuItem\_Click(object sender, EventArgs e)

{

currentTableName = "Абитуриенты";

LoadData(currentTableName);

label2.Visible = false;

comboBox2.Visible = false;

label1.Visible = true;

comboBox1.Visible = true;

}

private void LoadData(string tableName)

{

SqlConnection conn = new SqlConnection();

conn.ConnectionString = (Properties.Settings.Default.Ip5\_21\_SofronovConnectionString);

SqlCommand cmd = new SqlCommand();

cmd.Connection = conn;

cmd.CommandText = $"Select \* from {tableName}";

conn.Open();

SqlDataReader rdr = cmd.ExecuteReader();

DataTable dt = new DataTable();

for (int i = 0; i < rdr.FieldCount; i++)

{

dt.Columns.Add(new DataColumn(rdr.GetName(i), rdr.GetFieldType(i)));

}

while (rdr.Read())

{

DataRow row = dt.NewRow();

for (int i = 0; i < rdr.FieldCount; i++)

{

row[i] = rdr.GetValue(i);

}

dt.Rows.Add(row);

}

conn.Close();

dataGridView1.DataSource = dt;

lbRecordCount.Text = "Количество записей: " + String.Format("{0}", dt.Rows.Count);

}

private void заявленияToolStripMenuItem\_Click(object sender, EventArgs e)

{

currentTableName = "Заявления";

LoadData(currentTableName);

}

private void специальностиToolStripMenuItem\_Click(object sender, EventArgs e)

{

currentTableName = "Специальности";

LoadData(currentTableName);

}

private void экзаменыToolStripMenuItem\_Click(object sender, EventArgs e)

{

currentTableName = "Экзамены";

LoadData(currentTableName);

}

private void оценкиToolStripMenuItem\_Click(object sender, EventArgs e)

{

currentTableName = "Оценки";

LoadData(currentTableName);

label2.Visible = true;

comboBox2.Visible = true;

label1.Visible = false;

comboBox1.Visible = false;

}

private void toolStripButton1\_Click(object sender, EventArgs e)

{

Form2 form2 = new Form2(((DataTable)dataGridView1.DataSource).NewRow(), false);

if (form2.ShowDialog() == DialogResult.OK)

{

DataRow newRow = form2.Row;

using (SqlConnection conn = new SqlConnection(Properties.Settings.Default.Ip5\_21\_SofronovConnectionString))

using (SqlCommand cmd = new SqlCommand("Абитуриент\_Insert", conn))

{

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.Add("@Фамилия", SqlDbType.NVarChar).Value = newRow["Фамилия"];

cmd.Parameters.Add("@Имя", SqlDbType.NVarChar).Value = newRow["Имя"];

cmd.Parameters.Add("@Отчество", SqlDbType.NVarChar).Value = newRow["Отчество"];

cmd.Parameters.Add("@Статус", SqlDbType.NVarChar).Value = newRow["Статус"];

cmd.Parameters.Add("@Город", SqlDbType.NVarChar).Value = newRow["Город"];

conn.Open();

if (cmd.ExecuteNonQuery() > 0)

{

MessageBox.Show("Студент успешно добавлен");

}

}

LoadData(currentTableName);

dataGridView1.Rows[dataGridView1.Rows.Count - 1].Selected = true;

}

}

private void toolStripButton2\_Click(object sender, EventArgs e)

{

if (dataGridView1.SelectedRows.Count == 0)

{

MessageBox.Show("Выберите запись для изменения.");

return;

}

DataGridViewRow selectedRow = dataGridView1.SelectedRows[0];

DataRow selectedDataRow = ((DataRowView)selectedRow.DataBoundItem).Row;

Form2 form2 = new Form2(selectedDataRow, true);

if (form2.ShowDialog() == DialogResult.OK)

{

using (SqlConnection conn = new SqlConnection(Properties.Settings.Default.Ip5\_21\_SofronovConnectionString))

using (SqlCommand cmd = new SqlCommand("Абитуриент\_Update", conn))

{

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.Add("@Фамилия", SqlDbType.NVarChar).Value = form2.Row["Фамилия"];

cmd.Parameters.Add("@Имя", SqlDbType.NVarChar).Value = form2.Row["Имя"];

cmd.Parameters.Add("@Отчество", SqlDbType.NVarChar).Value = form2.Row["Отчество"];

cmd.Parameters.Add("@Статус", SqlDbType.NVarChar).Value = form2.Row["Статус"];

cmd.Parameters.Add("@Город", SqlDbType.NVarChar).Value = form2.Row["Город"];

cmd.Parameters.Add("@Код\_абитуриента", SqlDbType.Int).Value = selectedDataRow["Код\_абитуриента"];

conn.Open();

if (cmd.ExecuteNonQuery() > 0)

{

MessageBox.Show("Запись успешно изменена");

}

else

{

MessageBox.Show("Операция отменена или не удалась.");

}

LoadData(currentTableName);

}

}

}

private void toolStripButton3\_Click(object sender, EventArgs e)

{

if (dataGridView1.SelectedRows.Count == 0)

{

MessageBox.Show("Выберите запись для удаления.");

return;

}

DataGridViewRow selectedRow = dataGridView1.SelectedRows[0];

DataRow selectedDataRow = ((DataRowView)selectedRow.DataBoundItem).Row;

DialogResult result = MessageBox.Show("Вы уверены, что хотите удалить выбранную запись?", "Подтверждение удаления", MessageBoxButtons.YesNo, MessageBoxIcon.Question);

if (result == DialogResult.Yes)

{

using (SqlConnection conn = new SqlConnection(Properties.Settings.Default.Ip5\_21\_SofronovConnectionString))

using (SqlCommand cmd = new SqlCommand("Абитуриент\_Delete1", conn))

{

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.Add("@Абитуриент", SqlDbType.NVarChar).Value = selectedDataRow["Код\_абитуриента"];

conn.Open();

if (cmd.ExecuteNonQuery() > 0)

{

MessageBox.Show("Запись успешно удалена");

}

else

{

MessageBox.Show("Операция отменена или не удалась.");

}

LoadData(currentTableName);

}

}

}

private void label1\_Click(object sender, EventArgs e)

{

}

private void comboBox1\_SelectedIndexChanged(object sender, EventArgs e)

{

SqlConnection conn = new SqlConnection(Properties.Settings.Default.Ip5\_21\_SofronovConnectionString);

SqlCommand cmd = new SqlCommand();

cmd.Connection = conn;

cmd.CommandType = CommandType.StoredProcedure;

cmd.CommandText = "Student\_Select\_Faculty";

cmd.Parameters.Add(new SqlParameter("@Код\_специальности", SqlDbType.Int));

cmd.Parameters["@Код\_специальности"].Value = comboBox1.Text;

conn.Open();

SqlDataReader rdr = cmd.ExecuteReader();

DataTable dt = new DataTable();

for (int i = 0; i < rdr.FieldCount; i++)

{

dt.Columns.Add(new DataColumn(rdr.GetName(i), rdr.GetFieldType(i)));

}

while (rdr.Read())

{

DataRow row = dt.NewRow();

for (int i = 0; i < rdr.FieldCount; i++)

{

row[i] = rdr.GetValue(i);

}

dt.Rows.Add(row);

}

conn.Close();

dataGridView1.DataSource = dt;

lbRecordCount.Text = $"Количество записей: {dt.Rows.Count} записей";

}

private void comboBox2\_SelectedIndexChanged(object sender, EventArgs e)

{

SqlConnection conn = new SqlConnection(Properties.Settings.Default.Ip5\_21\_SofronovConnectionString);

SqlCommand cmd = new SqlCommand();

cmd.Connection = conn;

cmd.CommandType = CommandType.StoredProcedure;

cmd.CommandText = "Grade\_Select\_Faculty";

cmd.Parameters.Add(new SqlParameter("@Код\_специальности", SqlDbType.Int));

cmd.Parameters["@Код\_специальности"].Value = comboBox2.Text;

conn.Open();

SqlDataReader rdr = cmd.ExecuteReader();

DataTable dt = new DataTable();

for (int i = 0; i < rdr.FieldCount; i++)

{

dt.Columns.Add(new DataColumn(rdr.GetName(i), rdr.GetFieldType(i)));

}

while (rdr.Read())

{

DataRow row = dt.NewRow();

for (int i = 0; i < rdr.FieldCount; i++)

{

row[i] = rdr.GetValue(i);

}

dt.Rows.Add(row);

}

conn.Close();

dataGridView1.DataSource = dt;

lbRecordCount.Text = $"Количество записей: {dt.Rows.Count} записей";

}

private void comboBox1\_SelectedValueChanged(object sender, EventArgs e)

{

}

private void panel1\_Paint(object sender, PaintEventArgs e)

{

}

private void comboBox2\_SelectedValueChanged(object sender, EventArgs e)

{

}

}

}

Вывод: научились создавать приложение, позволяющее с помощью хранимых процедур выполнять фильтрацию данных.

Лабораторная работа № 15

Тема: Разработка приложения по технологии Entity Framework.

Программное обеспечение: MS Visual Studio 2015

Цель: научиться создавать приложение Database First Entity Framework.

Ход работы:

Задание 1:

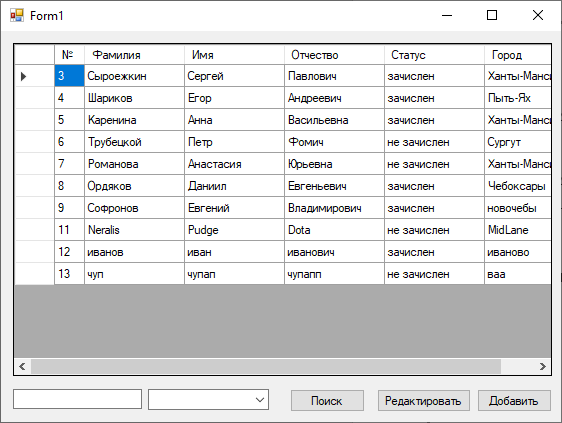


Рисунок 15.1 – Отображение данных таблицы

Программный код:

Form1:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace lr\_fifteen

{

public partial class Form1 : Form

{

public List<Абитуриенты> sheetAbiturients;

public ip521\_SofronovEntities db;

public Form1()

{

InitializeComponent();

db = new ip521\_SofronovEntities();

sheetAbiturients = db.Абитуриенты.OrderBy(o => o.Код\_абитуриента).ToList();

dataGridView1.DataSource = sheetAbiturients;

dataGridView1.ReadOnly = true;

if (dataGridView1.RowCount == 0) label1.Visible = true;

else label1.Visible = false;

}

private void Form1\_Load(object sender, EventArgs e)

{

dataGridView1.Columns[0].HeaderText = "№";

dataGridView1.Columns[0].Width = 30;

dataGridView1.Columns[6].Visible = false;

dataGridView1.Columns[7].Visible = false;

}

private void comboBox1\_SelectedIndexChanged(object sender, EventArgs e)

{

}

private void dataGridView1\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

}

}

}

Абитуриенты.cs:

namespace lr\_fifteen

{

using System;

using System.Collections.Generic;

public partial class Абитуриенты

{

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]

public Абитуриенты()

{

this.Заявления = new HashSet<Заявления>();

this.Оценки = new HashSet<Оценки>();

}

public int Код\_абитуриента { get; set; }

public string Фамилия { get; set; }

public string Имя { get; set; }

public string Отчество { get; set; }

public string Статус { get; set; }

public string Город { get; set; }

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]

public virtual ICollection<Заявления> Заявления { get; set; }

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]

public virtual ICollection<Оценки> Оценки { get; set; }

}

}

Вывод: научились создавать приложение Database First Entity Framework.

Лабораторная работа № 16

Тема: Разработка приложения по технологии Entity Framework: расширение функционала: поиск, добавление и редактирование данных.

Программное обеспечение: MS Visual Studio 2015

Цель: научиться создавать приложение Database First Entity Framework.

Ход работы:

Задание 1:

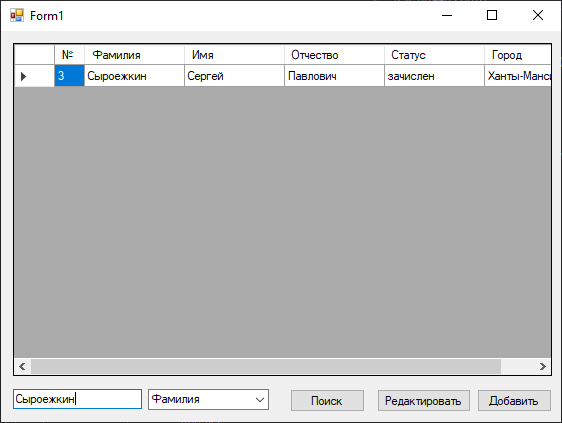


Рисунок 16.1 – Функция поиска

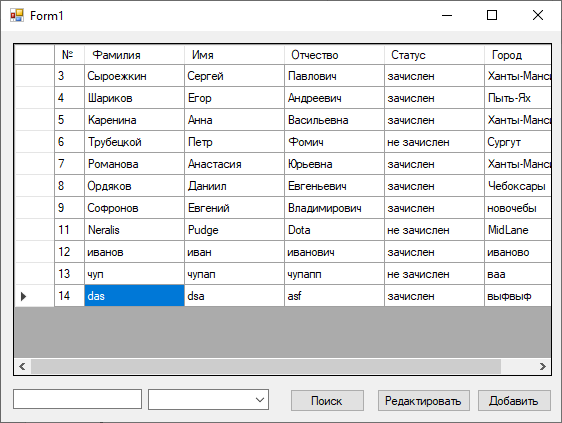


Рисунок 16.2 – До редактирования

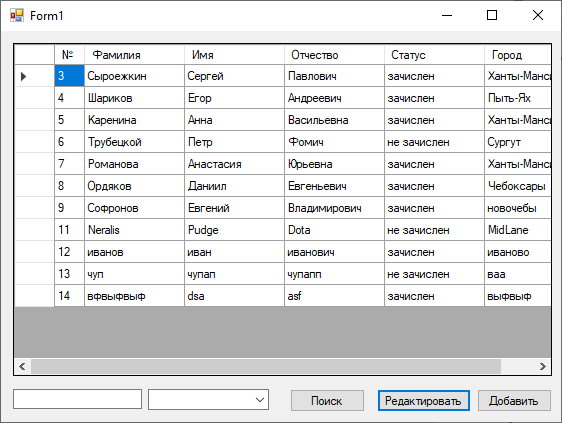


Рисунок 16.3 – После редактирования

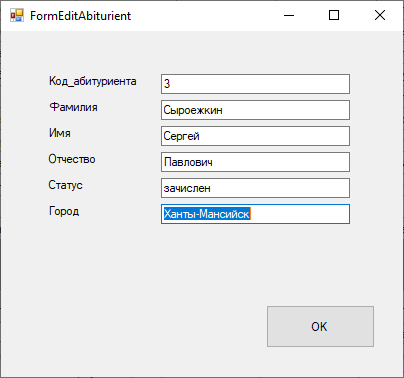


Рисунок 16.4 – Форма редактирования

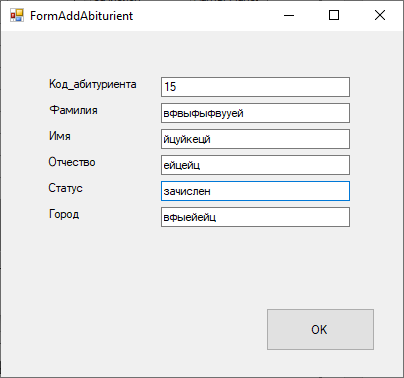


Рисунок 16.5 – Форма добавления

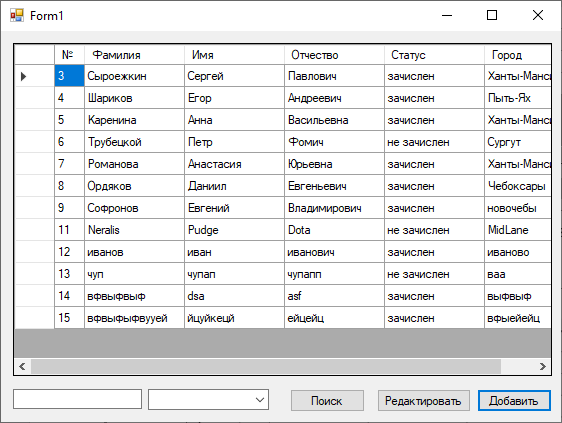


Рисунок 16.5 – После добавления

Код программы:

Form1:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace lr\_fifteen

{

public partial class Form1 : Form

{

public List<Абитуриенты> sheetAbiturients;

public ip521\_SofronovEntities db;

public Form1()

{

InitializeComponent();

db = new ip521\_SofronovEntities();

sheetAbiturients = db.Абитуриенты.OrderBy(o => o.Код\_абитуриента).ToList();

dataGridView1.DataSource = sheetAbiturients;

dataGridView1.ReadOnly = true;

if (dataGridView1.RowCount == 0) label1.Visible = true;

else label1.Visible = false;

}

private void Form1\_Load(object sender, EventArgs e)

{

dataGridView1.Columns[0].HeaderText = "№";

dataGridView1.Columns[0].Width = 30;

dataGridView1.Columns[6].Visible = false;

dataGridView1.Columns[7].Visible = false;

}

private void comboBox1\_SelectedIndexChanged(object sender, EventArgs e)

{

}

private void dataGridView1\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

}

private void button1\_Click(object sender, EventArgs e)

{

if (textBox1.Text != "")

{

switch (comboBox1.SelectedIndex)

{

case 0:

dataGridView1.DataSource = sheetAbiturients.Where(p => p.Код\_абитуриента.ToString() == textBox1.Text.ToString());

break;

case 1:

dataGridView1.DataSource = sheetAbiturients.Where(p => p.Фамилия.ToString().Contains(textBox1.Text.ToString())).ToList();

break;

}

}

else

{

dataGridView1.DataSource = sheetAbiturients;

}

dataGridView1.Update();

if (dataGridView1.RowCount == 0) label1.Visible = true;

else label1.Visible = false;

}

private void button3\_Click(object sender, EventArgs e)

{

if (Application.OpenForms.Count == 1)

{

FormAddAbiturient addAbit = new FormAddAbiturient();

addAbit.Owner = this;

addAbit.Show();

}

else

{

Application.OpenForms[1].Focus();

}

}

private void button2\_Click(object sender, EventArgs e)

{

if (dataGridView1.SelectedCells.Count == 1)

{

Абитуриенты item = sheetAbiturients.First(w => w.Код\_абитуриента.ToString() == dataGridView1

.SelectedCells[0]

.OwningRow

.Cells[0]

.Value

.ToString());

FormEditAbiturient edAbit = new FormEditAbiturient(item);

edAbit.Owner = this;

edAbit.Show();

}

else

{

Application.OpenForms[1].Focus();

}

}

}

}

FormEditAbiturient:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace lr\_fifteen

{

public partial class FormEditAbiturient : Form

{

Абитуриенты item;

public FormEditAbiturient(Абитуриенты itm)

{

item = itm;

InitializeComponent();

label7.Visible = false;

textBox1.Text = item.Код\_абитуриента.ToString();

textBox2.Text = item.Фамилия.ToString();

textBox3.Text = item.Имя.ToString();

textBox4.Text = item.Отчество.ToString();

textBox5.Text = item.Статус.ToString();

textBox6.Text = item.Город.ToString();

}

private void FormEditAbiturient\_Load(object sender, EventArgs e)

{

}

private void button1\_Click(object sender, EventArgs e)

{

if (textBox2.Text.Trim() == "" || textBox3.Text.Trim() == "" || textBox4.Text.Trim() == "")

{

label7.Visible = true;

label7.Text = "Заполните поля \'Фамилия\', \'Имя\', \'Отчество\'";

}

else if (textBox2.Text.ToCharArray().All(w => ((w >= 'a' && w <= 'я') || (w >= 'А' && w <= 'Я') || w == ' ')) == false)

{

label7.Visible = true;

label7.Text = "Поле ФИО может содержать только кириллиуц и пробелы";

}

else

{

var result = ((Form1)Owner).db.Абитуриенты.SingleOrDefault(w => w.Код\_абитуриента == item.Код\_абитуриента);

result.Фамилия = textBox2.Text.ToString();

result.Имя = textBox3.Text.ToString();

result.Отчество = textBox4.Text.ToString();

result.Статус = textBox5.Text.ToString();

result.Город = textBox6.Text.ToString();

((Form1)Owner).db.SaveChanges();

((Form1)Owner).sheetAbiturients = ((Form1)Owner).db.Абитуриенты.OrderBy(o => o.Код\_абитуриента).ToList();

((Form1)Owner).dataGridView1.DataSource = ((Form1)Owner).sheetAbiturients;

this.Close();

}

}

}

}

FormAddAbiturient:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace lr\_fifteen

{

public partial class FormAddAbiturient : Form

{

Абитуриенты result = new Абитуриенты();

public FormAddAbiturient()

{

InitializeComponent();

label7.Visible = false;

}

private void FormAddAbiturient\_Load(object sender, EventArgs e)

{

}

private void button1\_Click(object sender, EventArgs e)

{

if (textBox2.Text.Trim() == "" || textBox3.Text.Trim() == "" || textBox4.Text.Trim() == "" || textBox5.Text.Trim() == "")

{

label7.Visible = true;

label7.Text = "Заполните поля \'Фамилия\', \'Имя\', \'Отчество\' и \'Статус\'";

}

else if (textBox2.Text.ToCharArray().All(w => ((w >= 'a' && w <= 'я') || (w >= 'А' && w <= 'Я') || w == ' ')) == false)

{

label7.Visible = true;

label7.Text = "Поле ФИО может содержать только кириллиуц и пробелы";

}

else

{

result.Фамилия = textBox2.Text;

result.Имя = textBox3.Text;

result.Отчество = textBox4.Text;

result.Статус = textBox5.Text;

result.Город = textBox6.Text;

((Form1)Owner).db.Абитуриенты.Add(result);

((Form1)Owner).db.SaveChanges();

((Form1)Owner).sheetAbiturients = ((Form1)Owner).db.Абитуриенты.OrderBy(o => o.Код\_абитуриента).ToList();

((Form1)Owner).dataGridView1.DataSource = ((Form1)Owner).sheetAbiturients;

}

}

}

}

Вывод: научились создавать приложение Database First Entity Framework.

Лабораторная работа № 17

Тема: Добавление и редактирование данных в связанных таблицах.

Программное обеспечение: MS Visual Studio 2015

Цель: научиться реализовывать добавление и редактирование данных в связанных таблицах, использую технологию Database First Entity Framework.

Ход работы:

Задание 1: Реализовать добавление и редактирование данных в таблице

«Заявления».

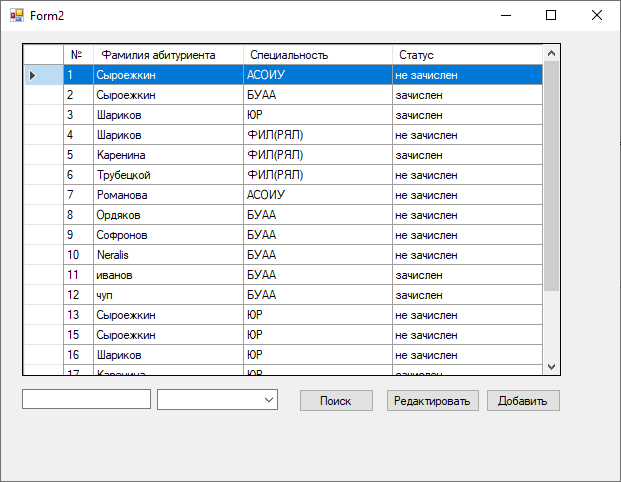


Рисунок 17.1 – Главная форма

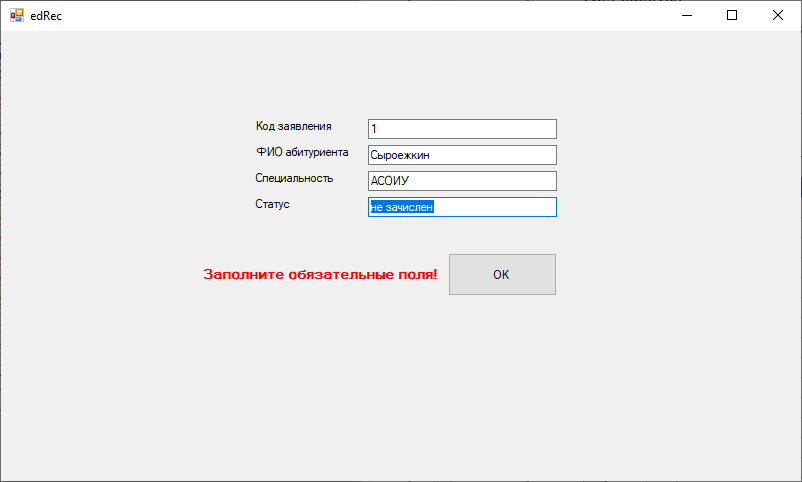


Рисунок 17.2 – Форма редактирования

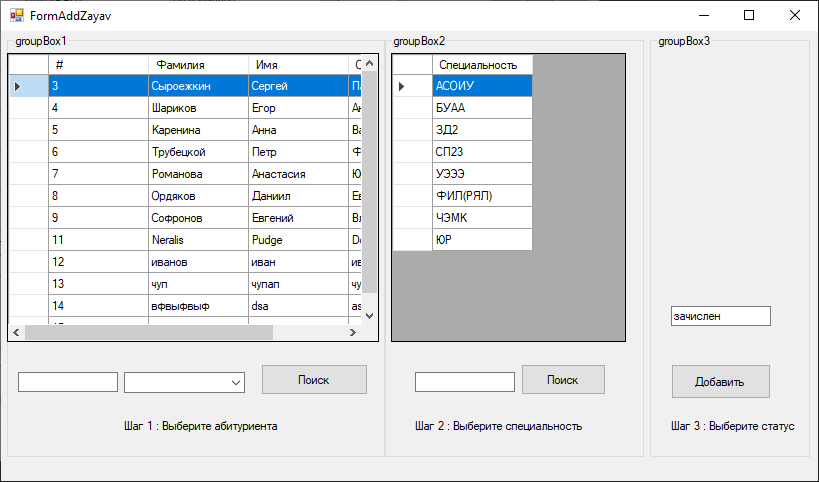


Рисунок 17.3 –Форма добавления

Код программы:

Form2:  
using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace lr\_fifteen

{

public partial class Form2 : Form

{

public List<view> viewzayav = new List<view>();

public ip521\_SofronovEntities db;

public Form2()

{

InitializeComponent();

db = new ip521\_SofronovEntities();

viewzayav = db.Абитуриенты.Join(db.Заявления, а => а.Код\_абитуриента, з => з.Код\_абитуриента, (а, з) => new { а, з }).Join(db.Специальности, зс => зс.з.Код\_специальности, с => с.Код\_специальности, (зс, с) => new { зс, с }).AsEnumerable().Select(х => new view(х.зс.з.Код\_заявления, х.зс.а.Фамилия, х.с.Название, х.зс.з.Статус)).OrderBy(о => о.Код\_заявления).ToList();

dataGridView1.DataSource = viewzayav.ToList();

}

public partial class view

{

public view (int id, string ab, string sp, string s)

{

this.Код\_заявления = id;

this.Фамилия = ab;

this.Название = sp;

this.Статус = s;

}

public int Код\_заявления { get; set; }

public string Фамилия { get; set; }

public string Название { get; set; }

public string Статус { get; set; }

}

private void button3\_Click(object sender, EventArgs e)

{

if (Application.OpenForms.Count == 2)

{

FormAddZayav addZayav = new FormAddZayav();

addZayav.Owner = this;

addZayav.Show();

}

else

{

Application.OpenForms[1].Focus();

}

}

private void Form2\_Load(object sender, EventArgs e)

{

dataGridView1.DataSource = viewzayav.ToList();

dataGridView1.ReadOnly = true;

if (dataGridView1.RowCount == 0) label1.Visible = true;

else label1.Visible = false;

dataGridView1.Columns[0].HeaderText = "№";

dataGridView1.Columns[0].Width = 30;

dataGridView1.Columns[1].HeaderText = "Фамилия абитуриента";

dataGridView1.Columns[2].HeaderText = "Специальность";

}

private void Form2\_FormClosing(object sender, FormClosingEventArgs e)

{

Application.Exit();

}

private void button2\_Click(object sender, EventArgs e)

{

if(dataGridView1.SelectedCells.Count == 1)

{

if (Application.OpenForms.Count == 2)

{

view item = viewzayav.First(w => w.Код\_заявления.ToString() == dataGridView1

.SelectedCells[0]

.OwningRow

.Cells[0]

.Value

.ToString());

FormEditZayav edZayav = new FormEditZayav(item);

edZayav.Owner = this;

edZayav.Show();

}

else Application.OpenForms[1].Focus();

}

}

}

}

FormEditZayav:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using static lr\_fifteen.Form2;

namespace lr\_fifteen

{

public partial class FormEditZayav : Form

{

view item;

public FormEditZayav(view itm)

{

InitializeComponent();

item = itm;

textBox1.Text = item.Код\_заявления.ToString();

textBox2.Text = item.Фамилия.ToString();

textBox3.Text = item.Название.ToString();

textBox4.Text = item.Статус.ToString();

}

private void edRec\_Load(object sender, EventArgs e)

{

}

private void label2\_Click(object sender, EventArgs e)

{

}

private void button1\_Click(object sender, EventArgs e)

{

if (textBox1.Text.Trim() == "")

{

label7.Visible = true;

label7.Text = "Заполните поле \'Диагноз\'";

}

else

{

var result = ((Form2)Owner).db.Заявления.SingleOrDefault(w => w.Код\_заявления == item.Код\_заявления);

result.Статус = textBox4.Text.ToString();

((Form2)Owner).db.SaveChanges();

((Form2)Owner).viewzayav =

((Form2)Owner).db.Абитуриенты

.Join(((Form2)Owner).db.Заявления,

а => а.Код\_абитуриента,

з => з.Код\_абитуриента,

(а, з) => new { а, з }).

Join(((Form2)Owner).db.Специальности,

зс => зс.з.Код\_специальности,

с => с.Код\_специальности,

(зс, с) => new { зс, с }).

AsEnumerable()

.Select(х => new view(

х.зс.з.Код\_заявления,

х.зс.а.Фамилия,

х.с.Название,

х.зс.з.Статус))

.OrderBy(о => о.Код\_заявления).ToList();

((Form2)Owner).dataGridView1.DataSource =

((Form2)Owner).viewzayav;

((Form2)Owner).dataGridView1.Refresh();

this.Close();

}

}

private void FormEditZayav\_FormClosing(object sender, FormClosingEventArgs e)

{

Application.Exit();

}

}

}

FormAddZayav:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using static lr\_fifteen.Form2;

namespace lr\_fifteen

{

public partial class FormAddZayav : Form

{

public ip521\_SofronovEntities db;

public List<Абитуриенты> sheetabiturients;

public List<Специальности> sheetspecs;

public FormAddZayav()

{

InitializeComponent();

db = new ip521\_SofronovEntities();

sheetabiturients = db.Абитуриенты.OrderBy(о => о.Код\_абитуриента).ToList();

dataGridView1.DataSource = sheetabiturients;

dataGridView1.ReadOnly = true;

if (dataGridView1.RowCount == 0) label1.Visible = true;

else label1.Visible = false;

dataGridView1.Columns[0].HeaderText = "#";

dataGridView1.Columns[4].Visible = false;

dataGridView1.Columns[5].Visible = false;

dataGridView1.Columns[6].Visible = false;

dataGridView1.Columns[7].Visible = false;

sheetspecs = db.Специальности.OrderBy(о => о.Название).ToList();

dataGridView2.DataSource = sheetspecs;

dataGridView2.ReadOnly = true;

if (dataGridView2.RowCount == 0) label1.Visible = true;

else label1.Visible = false;

dataGridView2.Columns[0].Visible = false;

dataGridView2.Columns[1].HeaderText = "Специальность";

dataGridView2.Columns[2].Visible = false;

dataGridView2.Columns[3].Visible = false;

dataGridView2.Columns[4].Visible = false;

}

private void FormAddZayav\_Load(object sender, EventArgs e)

{

}

private void FormAddZayav\_FormClosing(object sender, FormClosingEventArgs e)

{

Application.Exit();

}

private void button3\_Click(object sender, EventArgs e)

{

if (dataGridView1.SelectedCells.Count == 1 && dataGridView2.SelectedCells.Count == 1)

{

if (!string.IsNullOrWhiteSpace(textBox3.Text))

{

try

{

Заявления result = new Заявления();

result.Код\_абитуриента = Int32.Parse(dataGridView1.SelectedCells[0].OwningRow.Cells[0].Value.ToString());

result.Код\_специальности = Int32.Parse(dataGridView2.SelectedCells[0].OwningRow.Cells[0].Value.ToString());

result.Статус = textBox3.Text;

((Form2)Owner).db.Заявления.Add(result);

((Form2)Owner).db.SaveChanges();

// Блок кода для обновления таблицы

((Form2)Owner).viewzayav = ((Form2)Owner).db.Абитуриенты

.Join(((Form2)Owner).db.Заявления,

a => a.Код\_абитуриента,

з => з.Код\_абитуриента,

(a, з) => new { a, з })

.Join(((Form2)Owner).db.Специальности,

зс => зс.з.Код\_специальности,

с => с.Код\_специальности,

(зс, с) => new { зс, с })

.AsEnumerable()

.Select(x => new view(

x.зс.з.Код\_заявления,

x.зс.a.Фамилия,

x.с.Название,

x.зс.з.Статус))

.OrderBy(o => o.Код\_заявления)

.ToList();

((Form2)Owner).label1.Visible = false;

((Form2)Owner).dataGridView1.DataSource = ((Form2)Owner).viewzayav;

((Form2)Owner).dataGridView1.Refresh();

this.Close();

}

catch (Exception err)

{

label1.Visible = true;

label1.Text = "Ошибка ввода! " + err.Message;

}

}

else

{

label1.Visible = true;

label1.Text = "Укажите статус!";

}

}

else

{

label1.Visible = true;

label1.Text = "Выберите ровно одного абитуриента и ровно одну специальность!";

}

}

}

}

Вывод: научились реализовывать добавление и редактирование данных в связанных таблицах, использую технологию Database First Entity Framework.

Лабораторная работа № 18

Тема: Формирование отчетов.

Программное обеспечение: MS Visual Studio 2015, библиотека NPOI.

Цель: научиться выполнять автоматическую генерацию отчетов: экспорт данных в XLS формат.

Ход работы:

Задание 1. Реализовать формирование отчета на примере списка абитуриентов:

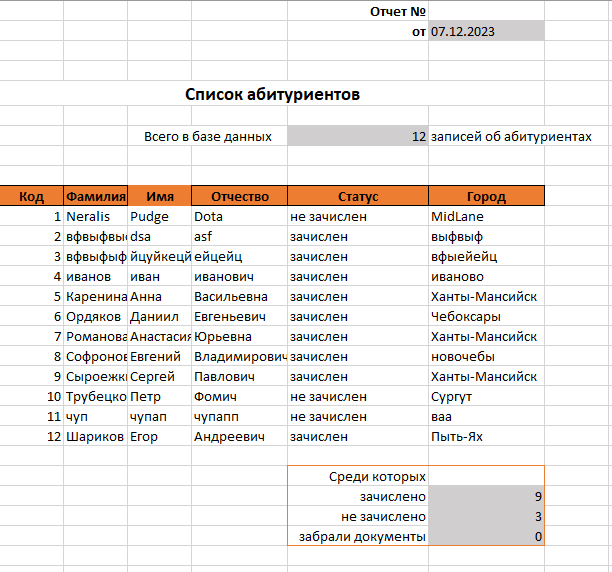


Рисунок 18.1 – Сгенерированный отчет

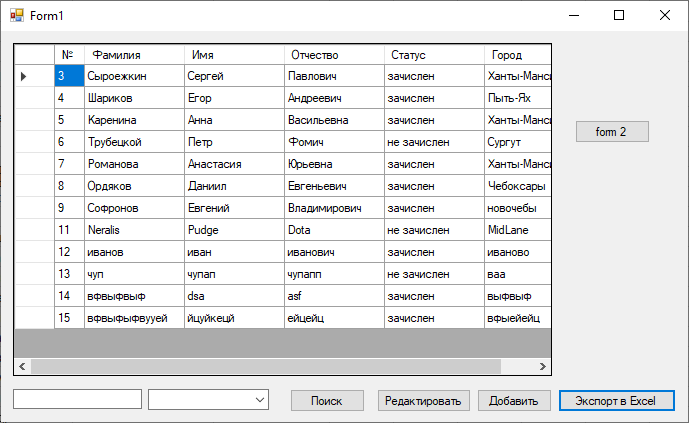


Рисунок 18.2 – Главная форма

Код программы:

Form1:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.IO;

using NPOI.XSSF.UserModel;

namespace lr\_fifteen

{

public partial class Form1 : Form

{

public List<Абитуриенты> sheetAbiturients;

public ip521\_SofronovEntities db;

public Form1()

{

InitializeComponent();

db = new ip521\_SofronovEntities();

sheetAbiturients = db.Абитуриенты.OrderBy(o => o.Код\_абитуриента).ToList();

dataGridView1.DataSource = sheetAbiturients;

dataGridView1.ReadOnly = true;

if (dataGridView1.RowCount == 0) label1.Visible = true;

else label1.Visible = false;

}

private void Form1\_Load(object sender, EventArgs e)

{

dataGridView1.Columns[0].HeaderText = "№";

dataGridView1.Columns[0].Width = 30;

dataGridView1.Columns[6].Visible = false;

dataGridView1.Columns[7].Visible = false;

}

private void comboBox1\_SelectedIndexChanged(object sender, EventArgs e)

{

}

private void dataGridView1\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

}

private void button1\_Click(object sender, EventArgs e)

{

if (textBox1.Text != "")

{

switch (comboBox1.SelectedIndex)

{

case 0:

dataGridView1.DataSource = sheetAbiturients.Where(p => p.Код\_абитуриента.ToString() == textBox1.Text.ToString());

break;

case 1:

dataGridView1.DataSource = sheetAbiturients.Where(p => p.Фамилия.ToString().Contains(textBox1.Text.ToString())).ToList();

break;

}

}

else

{

dataGridView1.DataSource = sheetAbiturients;

}

dataGridView1.Update();

if (dataGridView1.RowCount == 0) label1.Visible = true;

else label1.Visible = false;

}

private void button3\_Click(object sender, EventArgs e)

{

if (Application.OpenForms.Count == 1)

{

FormAddAbiturient addAbit = new FormAddAbiturient();

addAbit.Owner = this;

addAbit.Show();

}

else

{

Application.OpenForms[1].Focus();

}

}

private void button2\_Click(object sender, EventArgs e)

{

if (dataGridView1.SelectedCells.Count == 1)

{

Абитуриенты item = sheetAbiturients.First(w => w.Код\_абитуриента.ToString() == dataGridView1

.SelectedCells[0]

.OwningRow

.Cells[0]

.Value

.ToString());

FormEditAbiturient edAbit = new FormEditAbiturient(item);

edAbit.Owner = this;

edAbit.Show();

}

else

{

Application.OpenForms[1].Focus();

}

}

private void button4\_Click(object sender, EventArgs e)

{

Form2 f = new Form2();

f.Show();

this.Hide();

}

private void Form1\_FormClosing(object sender, FormClosingEventArgs e)

{

Application.Exit();

}

private void button5\_Click(object sender, EventArgs e)

{

SaveFileDialog dialog = new SaveFileDialog();

dialog.InitialDirectory = Environment.GetFolderPath(Environment.SpecialFolder.Desktop);

dialog.DefaultExt = ".xlsx";

dialog.Filter = "Таблицы Excel (\*.xlsx)|\*.xlsx|Все файлы (\*.\*)|\*.\*";

dialog.FilterIndex = 1;

dialog.FileName = "Отчет";

if(dialog.ShowDialog()== DialogResult.OK)

{

var file = new FileStream(dialog.FileName, FileMode.Create, FileAccess.ReadWrite);

var template = new MemoryStream(Properties.Resources.template, true);

var workbook = new XSSFWorkbook(template);

var sheet = workbook.GetSheetAt(0);

//header

sheet.GetRow(1).GetCell(5).SetCellValue(DateTime.Today.ToShortDateString());

sheet.GetRow(6).GetCell(4).SetCellValue(sheetAbiturients.Count());

//footer

sheet.GetRow(12).GetCell(5).SetCellValue(sheetAbiturients.Count(w => w.Статус == "зачислен"));

sheet.GetRow(13).GetCell(5).SetCellValue(sheetAbiturients.Count(w => w.Статус == "не зачислен"));

sheet.GetRow(14).GetCell(5).SetCellValue(sheetAbiturients.Count(w => w.Статус == "забрал документы"));

sheet.ShiftRows(10, sheet.LastRowNum, sheetAbiturients.Count(), true, true);

int row = 10;

foreach(var item in sheetAbiturients.OrderBy(o => o.Фамилия))

{

var rowInsert = sheet.CreateRow(row);

rowInsert.CreateCell(0).SetCellValue(row - 9);

rowInsert.CreateCell(1).SetCellValue(item.Фамилия);

rowInsert.CreateCell(2).SetCellValue(item.Имя);

rowInsert.CreateCell(3).SetCellValue(item.Отчество);

rowInsert.CreateCell(4).SetCellValue(item.Статус);

rowInsert.CreateCell(5).SetCellValue(item.Город);

row++;

}

workbook.Write(file);

}

}

}

}

Вывод: научились выполнять автоматическую генерацию отчетов: экспорт данных в XLS формат.

ЛАБОРАТОРНАЯ РАБОТА № 19.

Управление базами данных

Цель: научиться использовать SQL-команды для создания баз данных и

изменения их структуры, а также выполнять резервное копирование и

восстанавливать базы данных из резервных копий.

Программное обеспечение: SQL Server Management Studio Express.

Требования к отчету: по результатам работы представить отчёт со

скриншотами, содержащими SQL-команды и результаты их выполнения для каждой задачи.

Задание 1: Создайте базу данных с журналом с помощью команд языка SQL.

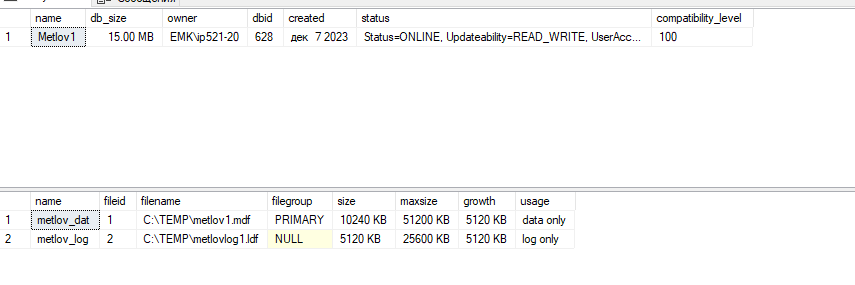


Рисунок 19.1 – Задание 1

Задание 2: Создайте резервную копию своей базы данных.

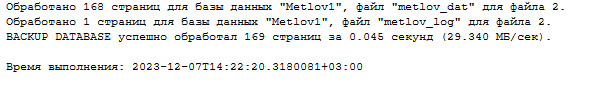


Рисунок 19.2 – Задание 2

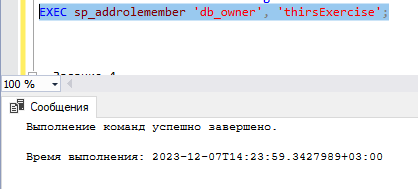


Рисунок 19.3 – Задание 3

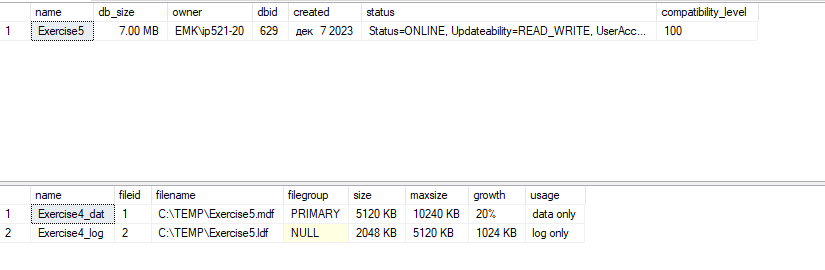


Рисунок 19.4 – Задание 4

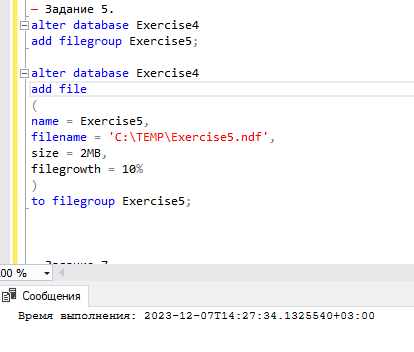


Рисунок 19.5 – Задание 5

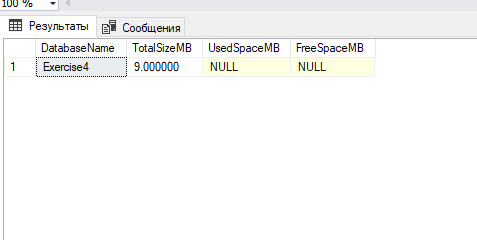


Рисунок 19.7 – Задание 7

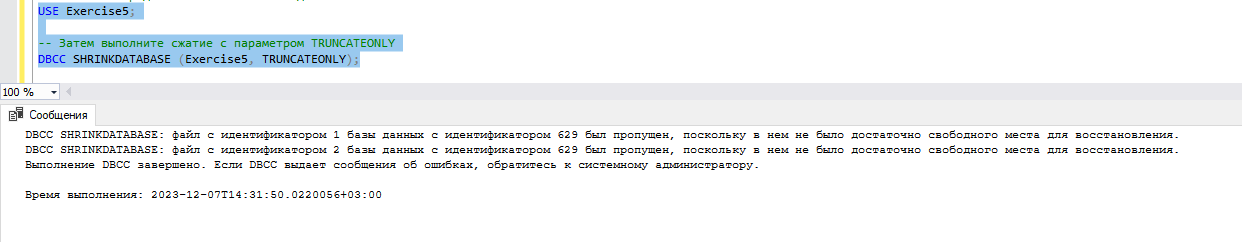


Рисунок 19.8 – Задание 8

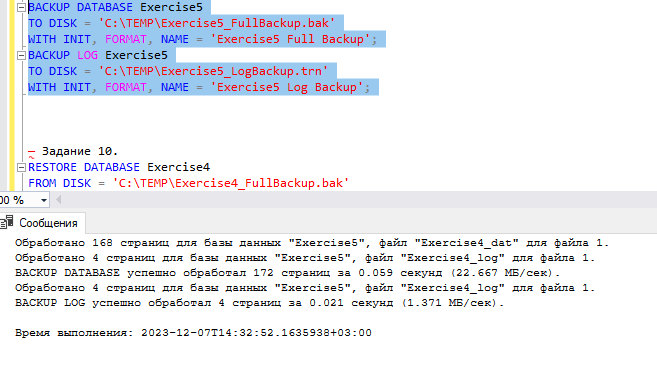


Рисунок 19.9 – Задание 9

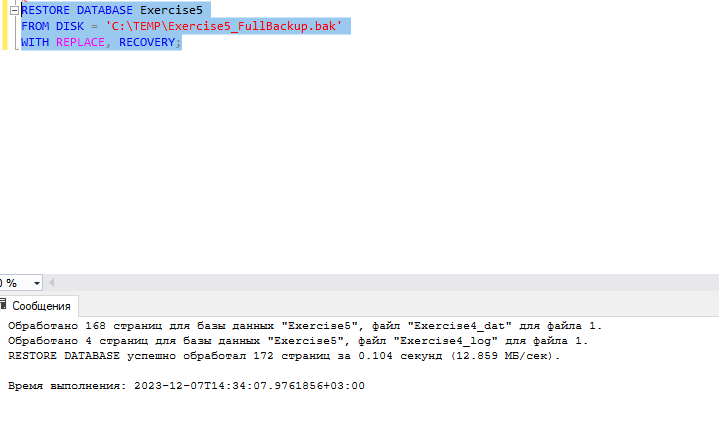


Рисунок 19.10 – Задание 10

Код программы:

— Задание 1.

create database Metlov1

on

(name=metlov\_dat,

filename='C:\TEMP\metlov1.mdf',

size=10,

maxsize=50,

filegrowth=5)

log on

(name=metlov\_log,

filename='C:\TEMP\metlovlog1.ldf',

size=5mb,

maxsize=25mb,

filegrowth=5mb)

sp\_helpdb Metlov1

— Задание 2

sp\_addumpdevice 'disk', 'backupdisk\_metlov1', 'c:\temp\metlob1.bak'

backup database Metlov1 TO backupdisk\_metlov1

— Задание 3

create user thirsExercise for login thirsExercise

EXEC sp\_addrolemember 'db\_owner', 'thirsExercise';

— Задание 4

— Создаем базу данных

create DATABASE Exercise5

ON

(

NAME = Exercise4\_dat,

FILENAME = 'C:\TEMP\Exercise5.mdf',

SIZE = 5MB,

MAXSIZE = 10MB,

FILEGROWTH = 20%

)

LOG ON

(

NAME = Exercise4\_log,

FILENAME = 'C:\TEMP\Exercise5.ldf',

SIZE = 2MB,

MAXSIZE = 5MB,

FILEGROWTH = 1MB

);

sp\_helpdb Exercise5

— Задание 5.

alter database Exercise4

add filegroup Exercise5;

alter database Exercise4

add file

(

name = Exercise5,

filename = 'C:\TEMP\Exercise5.ndf',

size = 2MB,

filegrowth = 10%

)

to filegroup Exercise5;

— Задание 7.

select

db\_name(database\_id) as DatabaseName,

sum(size/128.0) as TotalSizeMB,

sum(cast(fileproperty(name, 'SpaceUsed') as float)/128.0) as UsedSpaceMB,

sum((size - cast(fileproperty(name, 'SpaceUsed') as float))/128.0) as FreeSpaceMB

from

sys.master\_files

where

database\_id = db\_id('Exercise4')

group by

database\_id;

— Задание 8.

-- Сначала определите имя вашей базы данных

USE Exercise5;

-- Затем выполните сжатие с параметром TRUNCATEONLY

DBCC SHRINKDATABASE (Exercise5, TRUNCATEONLY);

— Задание 9.

BACKUP DATABASE Exercise5

TO DISK = 'C:\TEMP\Exercise5\_FullBackup.bak'

WITH INIT, FORMAT, NAME = 'Exercise5 Full Backup';

BACKUP LOG Exercise5

TO DISK = 'C:\TEMP\Exercise5\_LogBackup.trn'

WITH INIT, FORMAT, NAME = 'Exercise5 Log Backup';

— Задание 10.

RESTORE DATABASE Exercise5

FROM DISK = 'C:\TEMP\Exercise5\_FullBackup.bak'

WITH REPLACE, RECOVERY;

Вывод: научиться использовать SQL-команды для создания баз данных и

изменения их структуры, а также выполнять резервное копирование и

восстанавливать базы данных из резервных копий.