

Частное учреждение образования  
«Колледж бизнеса и права»

УТВЕРЖДАЮ  
Ведущий методист  
колледжа  
\_\_\_\_\_ Е.В. Паскал  
«\_\_\_» \_\_\_\_\_ 2022

Специальность: «Программное обеспечение информационных технологий»	Учебная дисциплина: «Базы данных и системы управления базами данных»
--	--

ЛАБОРАТОРНАЯ РАБОТА № 22

Инструкционно-технологическая карта

Тема: Создание хранимых процедур и определенных пользователем функций.

Цель работы: научиться разрабатывать хранимые процедуры и пользовательские функции.

Время выполнения: 2 часа

**Содержание работы**

1. Теоретические сведения для выполнения работы
2. Порядок выполнения работы
3. Пример выполнения работы
4. Контрольные вопросы
5. Литература

**1. ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ ДЛЯ ВЫПОЛНЕНИЯ РАБОТЫ**

**Хранимые процедуры** представляют собой набор команд, состоящий из одного или нескольких операторов SQL или функций и сохраняемых в базе данных в откомпилированном виде.

Основное различие между пакетом и хранимой процедурой состоит в том, что последняя сохраняется в виде объекта базы данных, т.е. сохраняется на стороне сервера.

**Типы хранимых процедур**

SQL Server поддерживает *пользовательские хранимые процедуры* и *системные процедуры*. Хранимые процедуры создаются таким же образом, как и все другие объекты базы данных, т.е. при помощи языка DDL. Системные процедуры представляются SQL Server.

*Системные хранимые процедуры* предназначены для выполнения различных административных действий. Практически все действия по администриро-

ванию сервера выполняются с их помощью. Можно сказать, что системные хранимые процедуры являются интерфейсом, обеспечивающим работу с системными таблицами. Системные хранимые процедуры имеют префикс **sp\_**, в системной базе данных и могут быть вызваны в контексте любой другой базы данных.

*Пользовательские хранимые процедуры* реализуют те или иные действия. Хранимые процедуры – полноценный объект базы данных. Хранимая процедура предварительно компилируется перед тем, как она сохраняется в виде объекта базы данных. Предварительно скомпилированная форма процедуры сохраняется в базе данных и используется при каждом ее вызове. Хранимые процедуры создаются таким же образом, как и все другие объекты базы данных, т.е. при помощи языка DDL.

При создании хранимой процедуры можно определить необязательный список параметров. Таким образом, процедура будет принимать соответствующие аргументы при каждом ее вызове. Хранимые процедуры могут возвращать значение, содержащее определенную пользователем информацию или, в случае ошибки, соответствующее сообщение об ошибке.

*Временные хранимые процедуры* существуют лишь некоторое время, после чего автоматически уничтожаются сервером. Они делятся на *локальные* и *глобальные*. Локальные временные хранимые процедуры могут быть вызваны только из одного соединения, в котором созданы. При создании такой процедуры ей необходимо дать имя, начинающееся из одного символа #. Как и все временные объекты, хранимые процедуры этого типа удаляются при отключении пользователя, перезапуске или остановке сервера. Глобальные временные хранимые процедуры доступны для любых соединений сервера, на котором имеется такая же хранимая процедура. Для ее определения достаточно ей дать имя, начинающееся с символа ##. Удаляются эти процедуры при перезапуске или остановке сервера, а также при закрытии соединения, в контексте которого они были созданы.

Хранимые процедуры можно также использовать для следующих целей:

- управления авторизацией доступа;
- создания аудиторского следа действий с таблицами баз данных.

Синтаксис оператора создания новой или изменения имеющейся хранимой процедуры в обозначениях MS SQL Server:

```
CREATE PROC[EDURE] [schema_name.]proc_name
[({ @param1 } type1[VARYING] [= default1] [OUTPUT])] {, ...}
[WITH {RECOMPILE | ENCRYPTION | EXECUTE AS 'user_name'}]
[FOR REPLICATION]
AS batch| EXTERNAL NAME method_name
```

Параметр	Значение параметра
schema_name	имя схемы, которая назначается владельцем созданной хранимой процедуры
proc_name	имя хранимой процедуры

@param1	параметр процедуры (формальный аргумент) – значения, которые передаются вызывающим объектом процедуры для использования в ней. Параметра процедуры являются локальными в пределах процедуры
type1	тип данных параметра процедуры @param1. Для определения типа данных параметров хранимой процедуры подходят любые типы данных SQL, включая определенные пользователем
default1	определяет факультативное значение по умолчанию для соответствующего параметра процедуры
OUTPUT	указывает, что параметр процедуры является возвращаемым, и с его помощью можно вернуть значение из хранимой процедуры вызывающей процедуре или системе
VARYING	применяется совместно с параметром OUTPUT, имеющим тип CURSOR, и определяет, что выходным параметром будет результирующее множество
RECOMPILE	предписывает системе создавать план выполнения хранимой процедуры при каждом ее вызове в случае необходимости. См. замечание 1.
FOR REPLICATION	необходим при репликации данных
ENCRYPTIO N	выполнение шифрования кода хранимой процедуры для обеспечения защиты от использования авторских алгоритмов

**Замечание 1.** Использование опции WITH RECOMPILE сводит на нет одно из наиболее важных преимуществ хранимых процедур: улучшение производительности благодаря одной компиляции.

Ключевое слово AS размещается в начале собственно тела хранимой процедуры. В тело процедуры могут применяться практически все команды SQL, объявляться транзакции, устанавливаться блокировки и вызываться другие хранимые процедуры. Выход из хранимой процедуры можно осуществить посредством команды **RETURN**.

**Замечание 2.** Для разделения двух пакетов используется инструкция GO. Инструкцию CREATE PROCEDURE нельзя объединять с другими инструкциями Transact-SQL в одном пакете.

### Удаление и изменение хранимой процедуры

SQL Server поддерживает инструкцию **ALTER PROCEDURE** для модификации структуры хранимых процедур. Инструкция ALTER PROCEDURE обычно применяется для изменения инструкций Transact-SQL внутри процедуры. Все параметры инструкции ALTER PROCEDURE имеют такое же значение, как и одноименные параметры инструкции CREATE PROCEDURE. Основной целью

использования этой инструкции является избежание переопределения существующих прав хранимой процедуры.

Для удаления одной или группы хранимых процедур используется инструкция **DROP PROCEDURE**. Удалить хранимую процедуру может только ее владелец или члены предопределенных ролей db\_owner и sysadmin.

### Выполнение хранимой процедуры

Жизненный цикл хранимой процедуры состоит из двух этапов: ее создания и ее выполнения. Каждая процедура создается один раз, а выполняется многократно. Хранимая процедура выполняется посредством инструкции **EXECUTE** пользователем. Инструкция имеет следующий:

```
[[EXEC[UTE]] [ @return_status=] {proc_name|@proc_name_var}
  {[[@parameter1 =] value|[@parameter1=] @variable[OUTPUT]]|DEFAULT}..
[WITH RECOMPILE]
```

Параметр	Значение параметра
return_status	определяет факультативную целочисленную переменную, в которой сохраняется состояние возврата процедуры
DEFAULT	предоставляет значения по умолчанию для параметра процедуры, которое было указано в определении процедуры
@param1	параметр процедуры (формальный аргумент) – значения, которые передаются вызывающим объектом процедуре для использования в ней. Параметра процедуры являются локальными в пределах процедуры

Использование ключевого слова OUTPUT при вызове процедуры разрешается только для параметров, которые были объявлены при создании процедуры с ключевым словом OUTPUT.

**Замечание 3.** Из синтаксиса команды EXECUTE видно, что имена параметров могут быть опущены при вызове процедуры. Однако в этом случае пользователь должен указывать значения для параметров в том же порядке, в каком они перечислялись при создании процедуры. Присвоить параметру значение по умолчанию, просто пропустив его при перечислении, нельзя. Если же требуется опустить параметры, для которых определено значение по умолчанию, достаточно явного указания имен параметров при вызове хранимой процедуры. Более того, таким способом можно перечислять параметры и их значения в произвольном порядке.

Важно, что при вызове процедуры указываются либо имена параметров со значениями, либо только значения без имени параметра. Их комбинирование не допускается.

**Использование предложения WITH RESULTS SETS инструкции EXECUTE** В SQL Server 2012 для инструкции EXECUTE вводится предложение **WITH RESULT SETS**, посредством которого при выполнении определенных условий можно изменять форму результирующего набора хранимой процедуры.

**Функции, определенные пользователем** – подпрограммы, которые принимают параметры, выполняют действие, например, сложные вычисления, и

возвращают результат этого действия в виде значения. Возвращаемое значение может быть либо единичным скалярным значением, либо результирующим набором (таблицей).

**В языках программирования обычно существуют два типа подпрограмм:**

- 1) хранимые процедуры;
- 2) функции, определенные пользователем (**User Defined Functions, UDF**).

Хранимые процедуры содержат несколько операторов, могут иметь ноль или несколько входных параметров, но обычно не возвращают выходных параметров. В противоположность этому функции всегда имеют одно возвращаемое значение. Однако в функциях нельзя использовать временные таблицы и операции изменения данных в физических таблицах. Вместо временных таблиц, в функциях используются табличные переменные.

Функции, определенные пользователем, могут быть скалярными или табличными. Скалярная функция возвращает скалярное значение (число). Это означает, что в предложении RETURNS скалярной функции вы задаете один из стандартных типов данных. Функции являются табличными, если предложение RETURNS возвращает набор строк.

Структура табличной функции	Структура Inline табличной функции	Структура скалярной функции
<pre>CREATE FUNCTION &lt;Имя&gt; (&lt;Параметры&gt;) RETURNS @tableVariable TABLE (     &lt;Список полей с типом данных&gt; ) AS BEGIN     Блок объявлений     (Declare statements)     Код процедуры     RETURN; END; GO</pre>	<pre>CREATE FUNCTION &lt;Имя&gt; (&lt;Параметры&gt;) RETURNS TABLE AS RETURN (     WITH         tbl1 as (             SELECT ...             FROM ...             WHERE ...             GROUP BY ...             HAVING ...         ),         tbl2 as (             SELECT ...             FROM tbl1             WHERE ...         ),         ...         tblN as (             SELECT ...             FROM ...             WHERE ...             GROUP BY ...             HAVING ...         )     SELECT ...     FROM tbl1,         tbl2, ... , tblN ); GO</pre>	<pre>CREATE FUNCTION &lt;Имя&gt; (&lt;Параметры&gt;) RETURNS INT AS BEGIN     Блок объявлений     (Declare statements)     Код процедуры     RETURN &lt;VALUE&gt;; END; GO</pre>

Определяемые пользователем функции предоставляют следующие возможности:

1. Делают возможным использование модульного программирования;
2. Позволяют ускорить выполнение;
3. Позволяют уменьшить сетевой трафик.

Виды табличных функций:

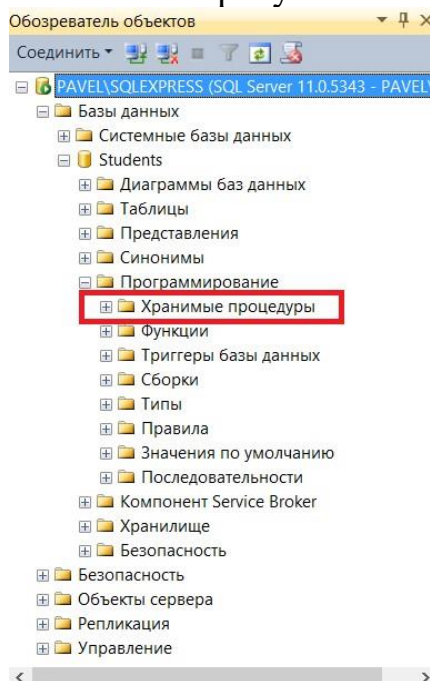
1. Табличная функция – функция, в которой задается структура результирующей таблицы. В теле функции выполняется запрос, который наполняет эту результирующую таблицу данными. После чего, возвращается эта таблица.
2. Inline-табличная функция – данный вид функции содержит комбинированный запрос (SELECT), который возвращает результирующий набор данных (таблицу). Обычно строится в формате WITH ... SELECT ...
3. Скалярная функция – Функция, которая возвращает 1 значение. Значение может иметь различный формат данных.

## 2. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

1. Изучить теоретическую часть настоящей инструкционно-технологической карты.
2. Рассмотреть работу процедур и функций, описанную в разделе «Примеры выполнения работы» настоящей инструкционно-технологической карты.
3. Получить у преподавателя индивидуальное задание и выполнить лабораторную работу в соответствии с вариантом задания согласно описанной в разделе «Пример выполнения работы» методике настоящей инструкционно-технологической карты.
4. Ответить на контрольные вопросы.

## 3. ПРИМЕР ВЫПОЛНЕНИЯ РАБОТЫ

В среде SQL Server Management Studio все хранимые процедуры находятся в папке «Хранимые процедуры», расположенной в папке «Программирование» в обозревателе объектов, представленной на рисунке ниже.



Для создания новой хранимой процедуры в обозревателе объектов необходимо щёлкнуть правой кнопкой мыши по папке «Хранимые процедуры» и в

появившемся меню выбрать пункт «Создать хранимую процедуру...». Появится окно новой хранимой процедуры, представленное на рисунке ниже.

```
-- =====
-- Template generated from Template Explorer using:
-- Create Procedure (New Menu).SQL
--
-- Use the Specify Values for Template Parameters
-- command (Ctrl-Shift-M) to fill in the parameter
-- values below.
--
-- This block of comments will not be included in
-- the definition of the procedure.
-- =====

SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO

-- =====
-- Author:      <Author,,Name>
-- Create date: <Create Date,,>
-- Description: <Description,,>
-- =====

CREATE PROCEDURE <Procedure_Name, sysname, ProcedureName>
-- Add the parameters for the stored procedure here
    <@Param1, sysname, @p1> <Datatype_For_Param1, , int> = <Default_Value_For_Param1, , 0>,
    <@Param2, sysname, @p2> <Datatype_For_Param2, , int> = <Default_Value_For_Param2, , 0>
AS
BEGIN
    -- SET NOCOUNT ON added to prevent extra result sets from
    -- interfering with SELECT statements.
    SET NOCOUNT ON;

    -- Insert statements for procedure here
    SELECT <@Param1, sysname, @p1>, <@Param2, sysname, @p2>
END
GO
```

Хранимая процедура состоит из следующих разделов:

1. область определения имени процедуры;
2. параметры, передаваемые в процедуру (@param1, @param2);
3. тело самой хранимой процедуры состоит из команды SELECT языка

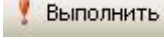
TSQL.

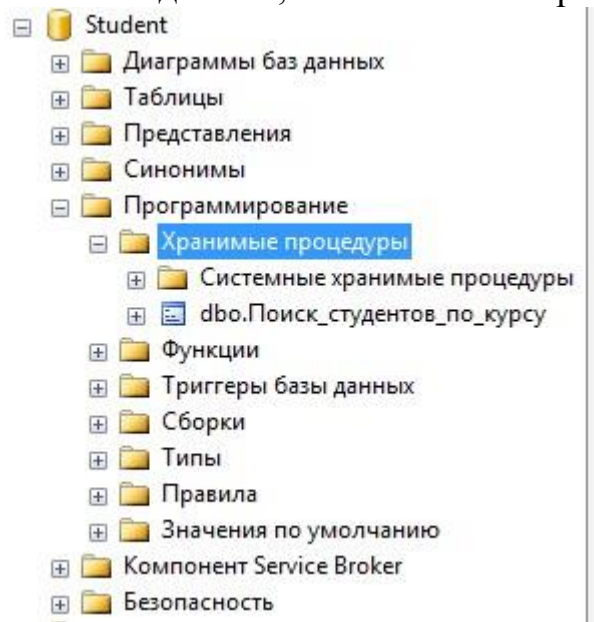
Для того, чтобы создать хранимую процедуру с входными параметрами на поиск студентов заданного курса обучения, в окне новой хранимой процедуры необходимо набрать следующий код, представленный на рисунке ниже

```
CREATE PROCEDURE Поиск_студентов_по_курсу
    @Введите_курс bigint
AS
BEGIN
    SET NOCOUNT ON;
    SELECT ФИО, Курс
    from Студенты
    where Курс=@Введите_курс
END
```

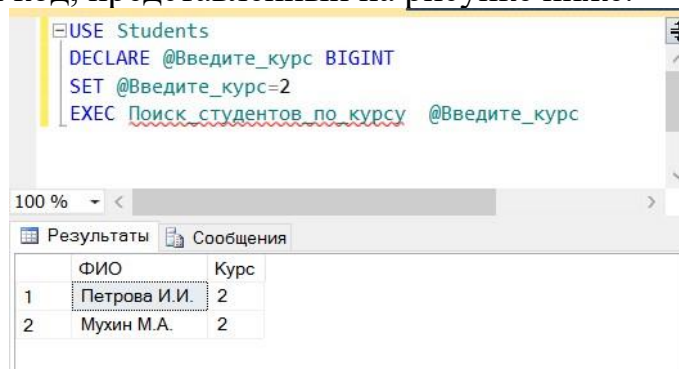
Сообщения  
Выполнение команд успешно завершено.



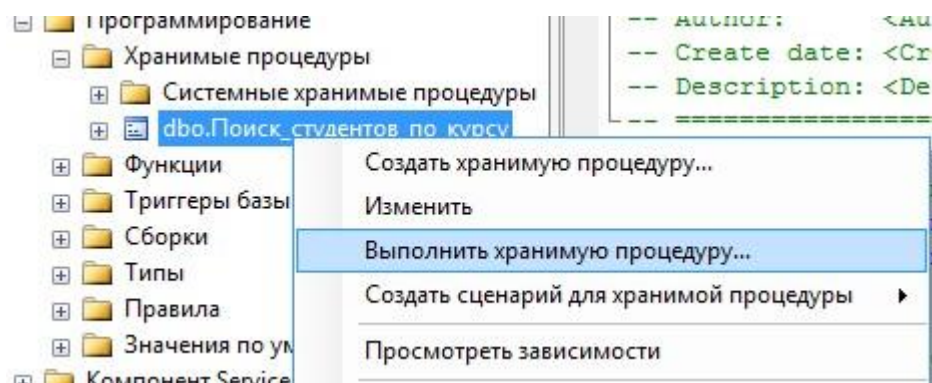
После этого необходимо нажать на кнопку , при этом созданная хранимая процедура появится в базе данных, как показано на рисунке ниже:



Для вызова хранимой процедуры Поиск студентов\_по\_курсу необходимо создать новый запрос с помощью кнопки  и в открывшемся окне нового запроса ввести код, представленный на рисунке ниже.

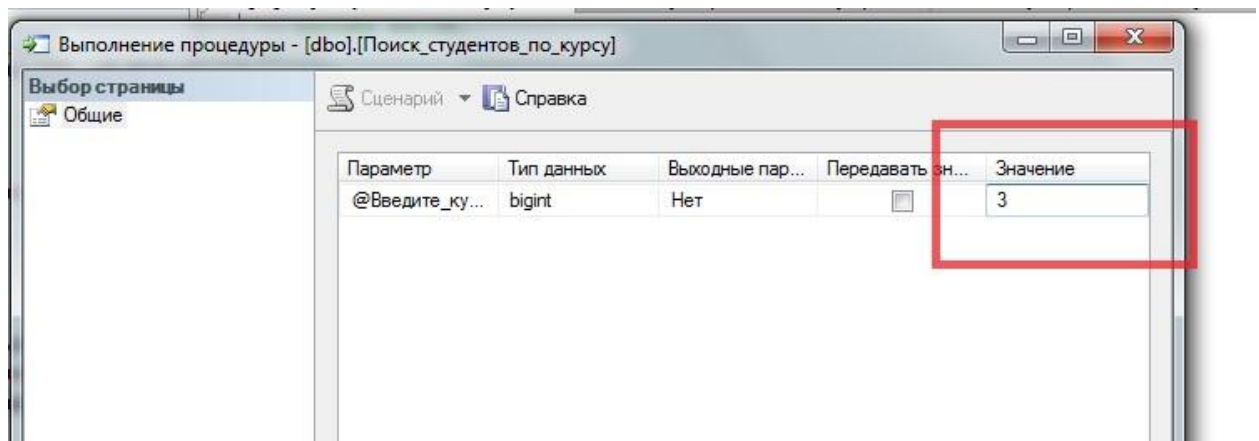


Выполнить хранимую процедуру можно и визуальными средствами среды SQL Server Management Studio. Для выполнения хранимой процедуры необходимо нажать правой кнопкой мыши по названию хранимой процедуры и выбрать **«Выполнить хранимую процедуру»**, как показано на рисунке ниже.

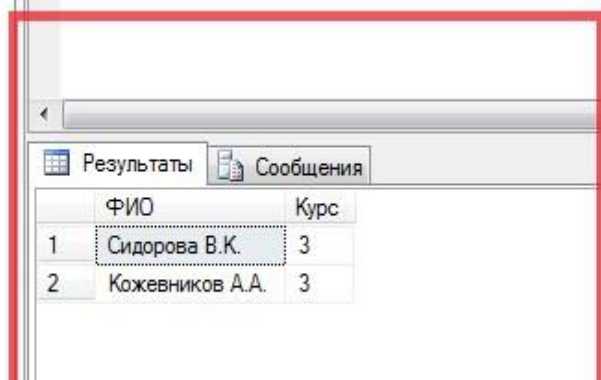


В появившемся окне необходимо ввести значение параметра для поиска и нажать кнопку **«ОК»**, как показано на рисунке ниже.

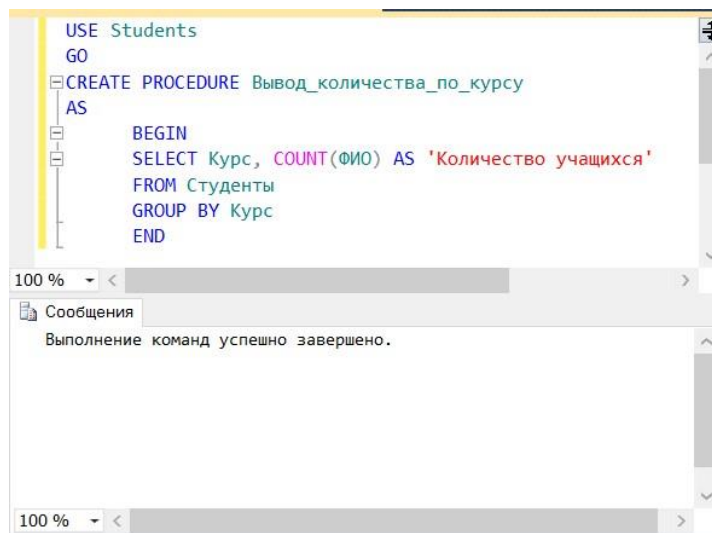





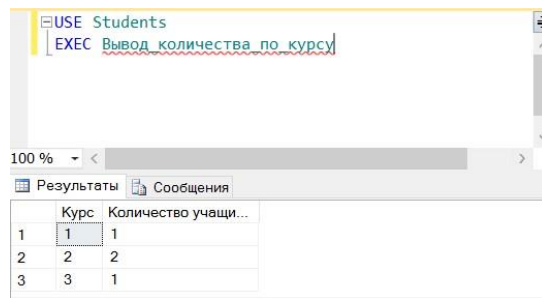
Результат выполнения хранимой процедуры показан ниже.



Для того, чтобы создать хранимую процедуру без входных параметров для вывода количества учащихся каждого курса обучения, в окне новой хранимой процедуры необходимо набрать следующий код, представленный на рисунке ниже.

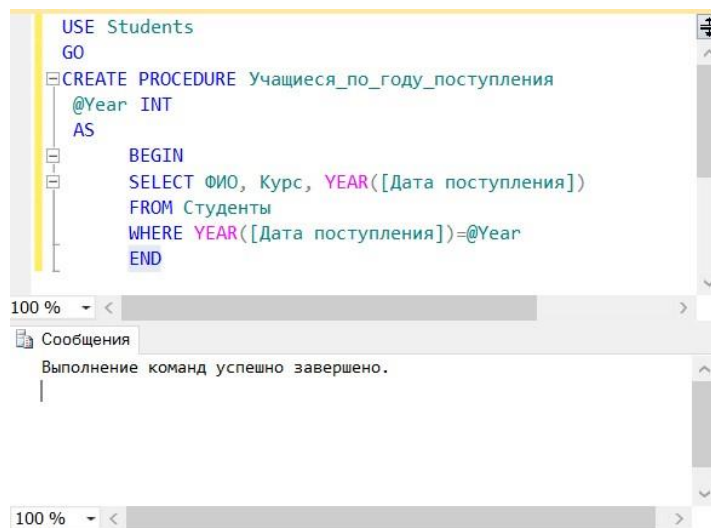


Для вызова хранимой процедуры Вывод\_количества\_по\_курсу необходимо создать новый запрос с помощью кнопки  Создать запрос и в открывшемся окне нового запроса ввести код, представленный на рисунке ниже.



	Курс	Количество учащихся
1	1	1
2	2	2
3	3	1

Для того, чтобы создать хранимую процедуру с входными параметрами для вывода информации об учащихся, поступивших в заданном году, в окне новой хранимой процедуры необходимо набрать следующий код, представленный на рисунке ниже.



```

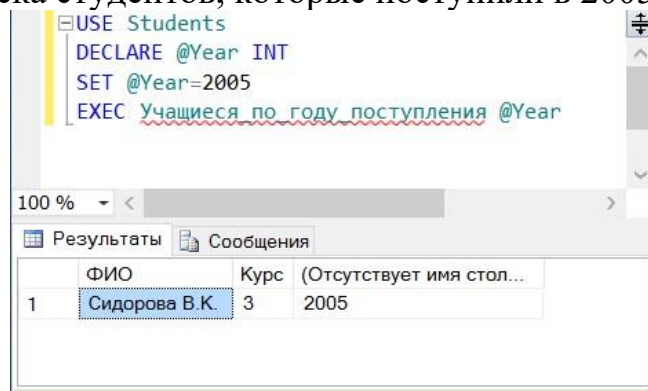
USE Students
GO
CREATE PROCEDURE Учащиеся_по_году_поступления
@Year INT
AS
BEGIN
SELECT ФИО, Курс, YEAR([Дата поступления])
FROM Студенты
WHERE YEAR([Дата поступления])=@Year
END

```

Сообщения

Выполнение команд успешно завершено.

Результатом поиска студентов, которые поступили в 2005 году, будет



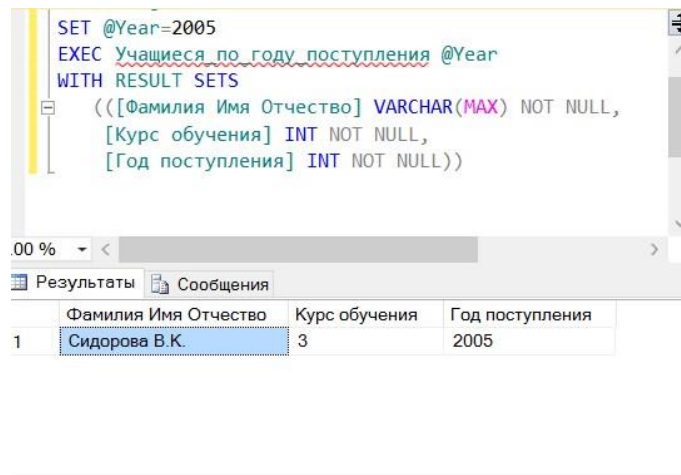
```

USE Students
DECLARE @Year INT
SET @Year=2005
EXEC Учащиеся_по_году_поступления @Year

```

	ФИО	Курс	(Отсутствует имя столбца)
1	Сидорова В.К.	3	2005

Выполнение этой процедуры выводит таблицу с двумя столбцами, заглавия которых совпадают с наименованиями соответствующих столбцов таблицы базы данных, и один столбец без имени: ФИО, Курс, Отсутствует имя столбца. Чтобы изменить заглавия столбцов (а также их тип данных), в SQL Server 2012 применяется новое предложение WITH RESULT SETS. Предыдущий пример с применением этого предложения представлен на рисунке ниже.



Пример создания пользовательской функции:

```
CREATE FUNCTION udf_Product(@num1 INT, @num2 INT)
RETURNS INT AS
BEGIN
    DECLARE @Product INT;
    SET @Product = ISNULL(@num1,0) * ISNULL(@num2,0);
    RETURN @Product;
END;
```

Пример ызова пользовательской функции:

```
SELECT udf_Product(2,10)
```

#### 4. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Дайте определение понятию хранимая процедура.
2. Назовите основное отличие пакета от хранимых процедур.
3. Какой префикс имеют системные хранимые процедуры?
4. Для чего предназначены системные хранимые процедуры?
5. Назовите типы временных хранимых процедур?
6. С какого символа должно начинаться имя временной хранимой процедуры?
7. Назовите виды хранимые процедуры?
8. Какую команду необходимо выполнить для того, чтобы разместить создаваемую хранимую процедуру в конкретной базе данных?
9. С какого символа должны начинаться имена параметров для передачи входных и выходных данных в создаваемой хранимой процедуре?
10. Сколько параметров можно задавать в хранимой процедуре?
11. Какое ключевое слово представляет собой значение, которое будет принимать соответствующий параметр по умолчанию?
12. Какое ключевое слово определяет начало тела хранимой процедуры?
13. Перечислите цели создания хранимых процедур.

14. Дайте определение понятию параметр хранимой процедуры.
15. Назовите этапы жизненного цикла хранимой процедуры.
16. Назовите команды создания, выполнения, удаления и изменения хранимой процедуры.
17. Дайте определение понятию пользовательская функция.
18. Перечислите виды табличных функций.

## 5. ЛИТЕРАТУРА

1. Петкович, Д. Microsoft SQL Server 2012. Руководство для начинающих: пер. с английского / Д. Петкович. – СПб.: БХВ-Петербург, 2013. – 816 с.: ил.
2. Сеть разработчиков Microsoft [Электронный ресурс]. – Режим доступа: <https://msdn.microsoft.com/ru-ru/library>

Преподаватель

В.Ю.Купцова

Рассмотрено на заседании цикловой  
комиссии программного обеспечения  
информационных технологий №10  
Протокол № \_\_ от «\_\_» \_\_\_\_\_ 2022  
Председатель ЦК В.Ю.Михалевич