Практическая работа № 12

Пользовательские функции

1. Цель работы

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

2. Теоретическая часть

На языке Transact-SQL очень много встроенных функций. Несмотря на этот факт, иногда требуются функции, которых нет в стандартной библиотеке. Transact-SQL предоставляет возможность создать собственную функцию для решения задач. Пользовательская скалярная функция возвращает в качестве ответа единственное значение.

Пользовательскую функцию можно использовать следующими способами:

– В инструкциях Transact-SQL, например, SELECT.

– В приложениях, вызывающих функцию.

– В определении другой пользовательской функции.

– Для определения столбца таблицы.

– Для определения ограничения CHECK на столбец.

Пользовательские функции не могут выполнять действия, изменяющие состояние базы данных.

Пользовательские функции не могут возвращать несколько результирующих наборов.

Пользовательские функции не могут использовать динамический SQL и временные таблицы. Табличные переменные разрешены к использованию.

Пользовательские функции могут быть вложенными, то есть из одной функции может быть вызвана другая. Вложенность функций не может превышать 32 уровней.

Упрощенный синтаксис создания пользовательской скалярной функции имеет следующий вид:

```
CREATE FUNCTION <название>
(
[{<@параметр> [AS] <тип> [= <значение по умолчанию>]}]
)
RETURNS <тип возврата>
[AS]
BEGIN
<команды>
```

RETURN <значение>
END

Значение, переменная или выражение после ключевого слова RETURN имеет такой же тип, который указан после ключевого слова RETURNS.

Созданная функция может быть вызвана, как и обычная встроенная функция, но при этом должны вызываться с помощью имени владельца. Простой вызов имеет следующий вид:

SELECT <владелец>.<функция>(<параметры>)

Для пользовательских функций допускается не более 2100 параметров. При выполнении функции значение каждого из объявленных параметров должно быть указано пользователем, если для них не определены значения по умолчанию.

Имя параметра, как и имя переменных, использует знак @ как первый символ.

Для INLINE функций ключевого слова RETURNS указывается тип TABLE без указания списка столбцов. Тело такой функции представляет собой единственный оператор SELECT, который начинается сразу после ключевого слово RETURN. Упрощенный синтаксис создания пользовательской функции INLINE имеет следующий вид:

```
CREATE FUNCTION <название>
(
[{<@параметр> [AS] <тип> [= <значение по умолчанию>]}]
)
RETURNS TABLE
AS
RETURN
(
SELECT
<список столбцов>
FROM
<таблица>
WHERE
<условие>
)
```

В MULTI-STATEMENT функциях после ключевого слова RETURNS указывается тип TABLE с определением столбцов и их типов данных. Упрощенный синтаксис создания пользовательской MULTI-STATEMENT функции имеет следующий вид:

```
CREATE FUNCTION <название>
(
[{<@параметр> [AS] <тип> [= <значение по умолчанию>]}]
)
RETURNS <@таблица> TABLE (<определение таблицы>)
AS
```

BEGIN

<команды>

RETURN

END

Для MULTI-STATEMENT функций оператор RETURN не имеет аргумента. Значение возвращаемой переменной функции возвращается как значение функции.

Для удаления пользовательских функций используется команда DROP FUNCTION. Упрощенный синтаксис имеет следующий вид:

DROP FUNCTION [IF EXISTS] <название функции>

Ключевые слова IF EXISTS удаляют функцию только в том случае, если она уже существует.

3. Практическая часть

Дана таблица ***Страны:***

Название	Столица	Площадь	Население	Континент
Австрия	Вена	83858	8741753	Европа
Азербайджан	Баку	86600	9705600	Азия
Албания	Тирана	28748	2866026	Европа
Алжир	Алжир	2381740	39813722	Африка
Ангола	Луанда	1246700	25831000	Африка
Аргентина	Буэнос-Айрес	2766890	43847000	Южная Америка
Афганистан	Кабул	647500	29822848	Азия
Бангладеш	Дакка	144000	160221000	Азия
Бахрейн	Манама	701	1397000	Азия
Белиз	Бельмопан	22966	377968	Северная Америка
Белоруссия	Минск	207595	9498400	Европа
Бельгия	Брюссель	30528	11250585	Европа
Бенин	Порто-Ново	112620	11167000	Африка
Болгария	София	110910	7153784	Европа
Боливия	Сукре	1098580	10985059	Южная Америка
Ботсвана	Габороне	600370	2209208	Африка
Бразилия	Бразилиа	8511965	206081432	Южная Америка
Буркина-Фасо	Уагадугу	274200	19034397	Африка
Бутан	Тхимпху	47000	784000	Азия
Великобритания	Лондон	244820	65341183	Европа
Венгрия	Будапешт	93030	9830485	Европа
Венесуэла	Каракас	912050	31028637	Южная Америка
Восточный Тимор	Дили	14874	1167242	Азия
Вьетнам	Ханой	329560	91713300	Азия

Пример 1: Напишите функцию для вывода столицы данной страны, и вызовите ее:

```
CREATE FUNCTION Пример1
(
	@Страна AS VARCHAR(50)
)
RETURNS VARCHAR(50)
AS
BEGIN
	DECLARE @S AS VARCHAR(50)
	SELECT
		@S = Столица
	FROM
		Страны
	WHERE
		Название = @Страна
	RETURN @S
END


SELECT dbo.Пример1('Австрия')
```

Пример 2: Напишите функцию для перевода площади в тыс. кв. км., и вызовите ее:

```
CREATE FUNCTION Пример2
(
	@Площадь AS FLOAT
)
RETURNS FLOAT
AS
BEGIN
	DECLARE @P AS FLOAT
	SET @P = ROUND(@Площадь / 1000, 2)
	RETURN @P
END

SELECT
	Название,
	Столица,
	Континент,
	Население,
	dbo.Пример2(Площадь) AS [Площадь тыс.кв.км]
FROM
	Страны
```

Пример 3: Напишите функцию для вычисления плотности населения, и вызовите ее:

```
CREATE FUNCTION Пример3
(
	@Население AS INT,
	@Площадь AS FLOAT
)
RETURNS FLOAT
AS
BEGIN
	DECLARE @P AS FLOAT
	SET @P = ROUND(CAST(@Население AS FLOAT) / @Площадь, 2)
	RETURN @P
END

SELECT
	Название,
	Столица,
	Континент,
	Население,
	Площадь,
	dbo.Пример3(Население, Площадь) AS Плотность
FROM
	Страны
ORDER BY
	Плотность DESC
```

Пример 4: Напишите функцию для поиска страны второй по площади, и вызовите ее:

```
CREATE FUNCTION Пример4()
RETURNS VARCHAR(50)
AS
BEGIN
	DECLARE @P AS VARCHAR(50)
	DECLARE @M1 AS FLOAT
	DECLARE @M2 AS FLOAT

	SELECT
		@M1 = MAX(Площадь)
	FROM
		Страны

	SELECT
		@M2 = MAX(Площадь)
```

```
        FROM
            Страны
        WHERE
            Площадь < @M1

        SELECT
            @P = Название
        FROM
            Страны
        WHERE
            Площадь = @M2

        RETURN @P
END

SELECT
    dbo.Пример4() AS [Второй по площади страна]
```

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

```
CREATE FUNCTION Пример5
(
    @Конт AS VARCHAR(50) = 'Европа'
)
RETURNS VARCHAR(50)
AS
BEGIN
    DECLARE @P AS VARCHAR(50)
    DECLARE @M AS FLOAT

    SELECT
        @M = MIN(Площадь)
    FROM
        Страны
    WHERE
        Континент = @Конт

    SELECT
        @P = Название
    FROM
        Страны
    WHERE
```

```
            Континент = @Конт
            AND
            Площадь = @M

    RETURN @P
END

SELECT
    dbo.Пример5('Азия') AS [Наименьшая по площади страна в Азии]

SELECT
    dbo.Пример5(DEFAULT) AS [Наименьшая по площади страна в Европе]
```

Пример 6: Напишите функцию для замены букв в заданном слове от второй до предпоследней на точку, и примените ее для названия страны:

```
CREATE FUNCTION Пример6
(
    @A AS VARCHAR(50)
)
RETURNS VARCHAR(50)
AS
BEGIN
    RETURN LEFT(@A, 1) + REPLICATE('.', LEN(@A) - 2) + RIGHT(@A, 1)
END

SELECT
    dbo.Пример6(Название) AS [Скрытое название]
    ,Столица
    ,Континент
    ,Площадь
    ,Население
FROM
    Страны
```

Пример 7: Напишите функцию, которая возвращает количество стран, содержащих в названии заданную букву:

```
CREATE FUNCTION Пример7
(
    @C AS CHAR(1)
)
RETURNS INT
AS
```

```
BEGIN
	DECLARE @K AS INT

	SELECT
		@K = COUNT(*)
	FROM
		Страны
	WHERE
		CHARINDEX(@C, Название) > 0

	RETURN @K
END
```

Пример 8: Напишите функцию для вывода списка стран с населением больше заданного числа, и вызовите ее:

```
CREATE FUNCTION Пример8
(
	@N AS INT
)
RETURNS TABLE
AS
RETURN (
		SELECT
			Название
			,Столица
			,Площадь
			,Население
			,Континент
		FROM
			Страны
		WHERE
			Население > @N
			)
SELECT
	*

FROM
	dbo.Пример8(100000000)
```

Пример 9: Напишите функцию для вывода списка стран с площадью в интервале заданных значений, и вызовите ее:

```
CREATE FUNCTION Пример9
```

```
(
	@A AS FLOAT,
	@B AS FLOAT
)
RETURNS TABLE
AS
RETURN (
		SELECT
			Название
			,Столица
			,Площадь
			,Население
			,Континент
		FROM
			Страны
		WHERE
			Площадь BETWEEN @A AND @B
			)
SELECT
	*

FROM
	dbo.Пример9(1000, 10000)
```

Пример 10: Напишите функцию для возврата таблицы с названием страны и плотностью населения, и вызовите ее:

```
CREATE FUNCTION Пример10()
RETURNS @Ст_Плот TABLE
			(
				Название VARCHAR(50),
				Плотность FLOAT
			)
AS
BEGIN
	INSERT
		@Ст_Плот
	SELECT
		Название
		, CAST(Население AS FLOAT) / Площадь AS Плотность
	FROM
		Страны
	RETURN
END
```

```
SELECT
        Название
        ,Плотность
FROM
        dbo.Пример10()
```

Пример 11: Удалите функцию из примера 10:

```
DROP FUNCTION Пример10
```

4. Задание

1. Напишите функцию для вывода названия страны с заданной столицей, и вызовите ее.
2. Напишите функцию для перевода населения в млн. чел. и вызовите ее.
3. Напишите функцию для вычисления плотности населения заданной части света и вызовите ее.
4. Напишите функцию для поиска страны, третьей по населению и вызовите ее.
5. Напишите функцию для поиска страны с максимальным населением в заданной части света и вызовите ее. Если часть света не указана, выбрать Азию.
6. Напишите функцию для замены букв в заданном слове от третьей до предпоследней на “тест” и примените ее для столицы страны.
7. Напишите функцию, которая возвращает количество стран, не содержащих в названии заданную букву.
8. Напишите функцию для возврата списка стран с площадью меньше заданного числа и вызовите ее.
9. Напишите функцию для возврата списка стран с населением в интервале заданных значений и вызовите ее.
10. Напишите функцию для возврата таблицы с названием континента и суммарным населением и вызовите ее.
11. Напишите функцию IsPalindrom(P) целого типа, возвращающую 1, если целый параметр P (P > 0) является палиндромом, и 0 в противном случае.
12. Напишите функцию Quarter(x, y) целого типа, определяющую номер координатной четверти, содержащей точку с ненулевыми вещественными координатами (x, y).
13. Напишите функцию IsPrime(N) целого типа, возвращающую 1, если целый параметр N (N > 1) является простым числом, и 0 в противном случае.
14. Напишите код для удаления созданных вами функций.