Практическая работа №8

Создание и управление представлениями

Цель работы: изучение назначения представлений БД, запросов программирования

Задание I. Изучить теоретическую часть, приметь практическую часть, оформить отчет. Самостоятельно, после выполнения все ПР, сформировать 5 запросов на процедуры и функции относительно своей БД

Представление – это виртуальные таблицы, определяемые запросом на языке Transact-SQL. Часто применяются для ограничения доступа к конфиденциальным данным в таблицах БД. Представление может выбирать данные из других представлений, вложенность представлений не должна превышать 32. Максимальное количество столбцов в представлении равно 1024. Для представления нельзя определить ограничения целостности, триггеры, правила.

Права доступа:

Select -

Insert –

Update –

Delete -

Чтобы создавать представления необходимо обладать правами владельца БД и иметь разрешение

With chec option – не позволяет изменять строки таким образом, чтобы они исчезли при отборе командой select.

Задание I. Выполните все запросы и зафиксируйте в отчете

1. Напишите запрос см. рис.1, напишите в отчете, какой список получился

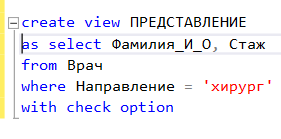


Рис.1

1. Самостоятельно создайте представление с помощью графического интерфейса, см. рис. 2
2. Напишите запрос на представление, которое ссылается на другое представление

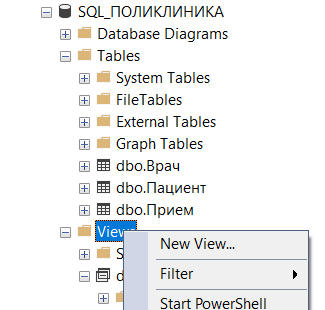


Рис.2

Программирование

Идентификатор – имя переменной, первый символ @. Объявление переменной:

Declare @имя переменной тип переменной(размер)

Присвоение значений переменным:

Set @имя переменной=значение

1. Выполните запрос, см. рис.3

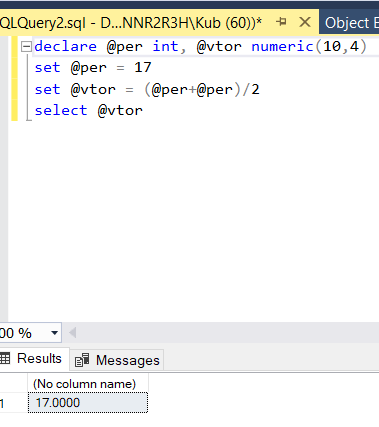
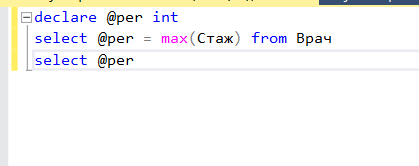
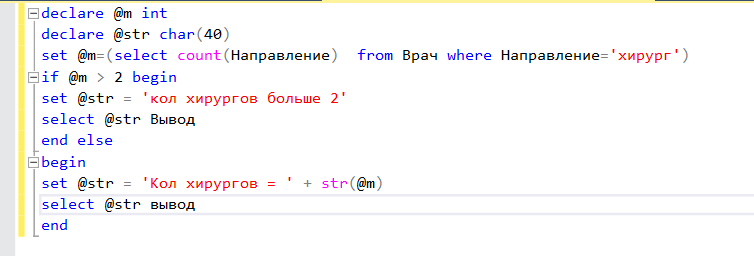


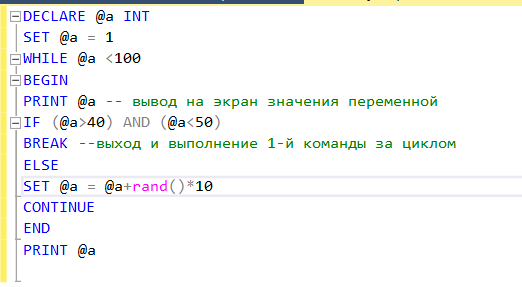
Рис.3

1. Выполните запрос, опишите в отчете, какая функция данного запроса



1. Выполните запрос, где применяется условный оператор



1. Рассмотрите запрос, протестируйте его, поэкспериментируйте и зафиксируйте в отчет
2. Самостоятельно на основе своей БД создайте минимум 10 запросов по программированию

Функции и хранимые процедуры

Используются для реализации сложных запросов обработки данных или различный административных действий создания учетных записей, получение информации об объекте и т.д. Хранятся они в виде исходного текста и существуют независимо от таблиц или каких либо других объектов БД (исключение – расширенные хранимые процедуры)

Плюсы такого хранения:

1. Уменьшает размер запроса;
2. Уменьшает нагрузку на сеть;
3. Упрощает сопровождение программных комплексов;

**Системные (built-in встроенные)** функции и процедуры – поставляются в составе SQL Server.

**Пользовательские (user-defined)** функции и процедуры – разрабатываются самим пользователем

**Хранимые процедуры** могут вызываться командой execute или exec

Процедура может возвращать результаты только через параметры с ключевым словом OUTPUT

**Команды:**

Create function

Create procedure

Alter function

Alter procedure

Drop function

Drop procedure

При создании функции указывается тип возвращаемого значения и в теле функции задается команда Return

Пример 1: Применение скалярной определяемой пользователем функции, вычисляющей месяц:

| use SQL\_ПОЛИКЛИНИКА  go  if object\_id(N'dbo.iso',N'FN') is not null | проверка isо пользовательская функция есть под таким именем, если да то |
| --- | --- |
| drop function dbo.iso | она будет удалена |
| go  create function dbo.iso(@DATE date)  returns char(15)  with execute as caller  AS  BEGIN  DECLARE @man int;  DECLARE @ISOweek char(15);  SET @man= MONTH(@DATE)  IF (@man=1) SET @ISOweek='Январь';  IF (@man=2) SET @ISOweek='Февраль';  IF (@man=3) SET @ISOweek='Март';  IF (@man=4) SET @ISOweek='Апрель';  IF (@man=5) SET @ISOweek='Май';  IF (@man=6) SET @ISOweek='Июнь';  IF (@man=7) SET @ISOweek='Июль';  IF (@man=8) SET @ISOweek='Август';  IF (@man=9) SET @ISOweek='Сентябрь';  IF (@man=10) SET @ISOweek='Октябрь';  IF (@man=11) SET @ISOweek='Ноябрь';  IF (@man=12) SET @ISOweek='Декабрь';    RETURN(@ISOweek);  END; | Returns(возвращает список) |

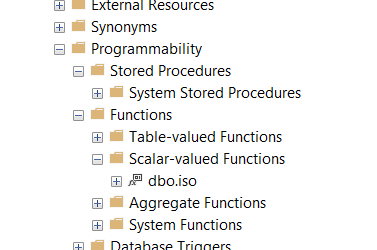
Посмотреть на эту функцию можно в разделе см. рис.4

Рис.4

Обратиться к функции можно через запрос см. рис.5

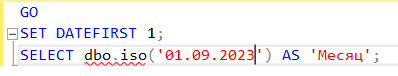


Рис.5

Пример 2: Создание пользовательской функции с табличным значением.

use SQL\_ПОЛИКЛИНИКА

GO

SELECT \* from DEKAN.iso\_g('глаукома')

Не получилось!!!!

Пример 3: Процедура без параметров, см. рис.6

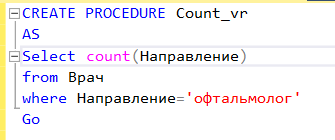
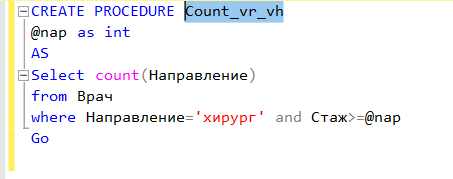


Рис.6

Пример 4: Процедура с входным параметром



Для вызова процедуры напишите запрос:

exec Count\_vr\_vh 5