**Базы данных**

**Отчет по лабораторной работе №8, Лешук Дмитрий ПОИБМС 7-2**

**Цель**: Ознакомиться с основами программирования на T-SQL, а именно с использованием локальных и глобальных переменных, способами вывода данных, использованием встроенных функций, конструкцией IF…ELSE, оператором WHILE, временной таблицей, оператором RETURN и блоками для обработки ошибок TRY,CATCH.

В Transact-SQL определение переменных выполняется в специальном блоке, под названием DECLARE, причем имя переменной всегда начинается с символа «@», а после имени следует тип данных – рисунок 1.1.

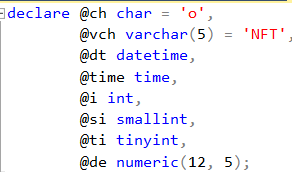


Рисунок 1.1 – Определение переменных в секции DECLARE

Инициализировать переменную можно как и в блоке DECLARE, так и с помощью SET и SELECT, оба варианта представлены на рисунке 1.2. Стоит отметить, что если не инициализировать переменную, ей присваивается значение NULL.



Рисунок 1.2 – Инициализация переменных

Вывод данных в T-SQL возможен двумя способами: оператором SELECT можно сформировать выходной результирующий набор и с помощью оператора PRINT можно вывести строку в стандартный выходной поток. Если одновременно выводятся данные, сформированные оператором SELECT и данные, сформированные оператором PRINT, то посмотреть последние можно на вкладке Messages. Выглядит это следующим образом – рисунок 1.3.

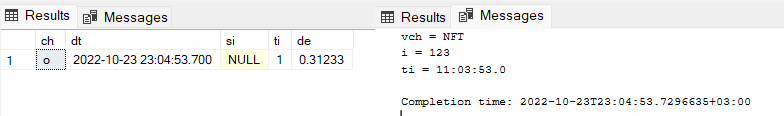
****

Рисунок 1.3 – SELECT и PRINT

Если же мы хотим вывести какую-либо строку а после неё переменную с помощью PRINT, нам понадобится выполнить явное приведение типов для переменной с помощью функции CAST() к типу varchar.

IF…ELSE – уже знакомая конструкция, которая есть и в других языках программирования. В IF мы записываем какое-либо логическое условие, и если оно истинно, выполняется последующая инструкция, если ложно – инструкции, следующие после ELSE. Вместо фигурных скобок для определения блока операций в T-SQL используется конструкция BEGIN…END. Пример запроса с использованием ранее перечисленного, а также с использованием переменных представлен на рисунке 1.4.

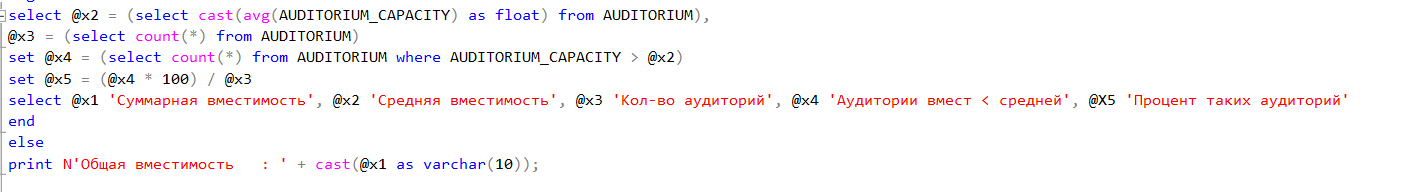


Рисунок 1.4 – Задание №3

В этом задании необходимо было разработать скрипт, в котором определяется общая вместимость аудиторий. Когда общая вместимость превышает 200, то вывести количество аудиторий, среднюю вместимость аудиторий, количество аудиторий, вместимость которых меньше средней, и процент таких аудиторий. Когда общая вместимость аудиторий меньше 200, то вывести сообщение о размере общей вместимости.

Глобальные переменные – такие переменные, которые хранятся и меняют свое значение в самой системе. Например, @@ROWCOUNT содержит число обработанных в последнем запросе строк, @@VERSION – текущую версию SQL-Сервера, @@ERROR – код последней ошибки, @@SERVERNAME – имя сервера. Использовать данные переменные необходимо было в задании №4, а именно, вывести их с помощью PRINT – рисунок 1.5.

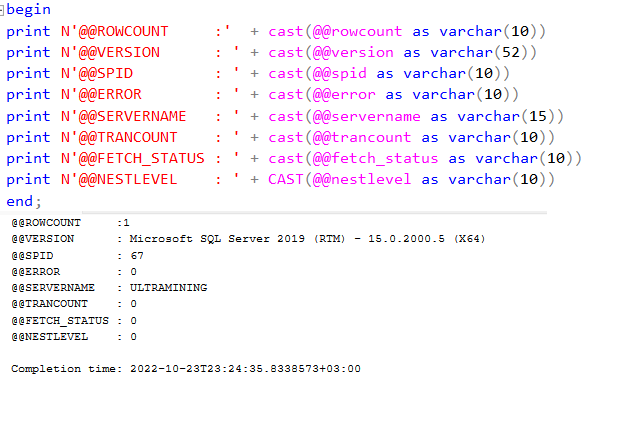


Рисунок 1.5 – Глобальные переменные

Довольно интересным было задание, которое заключалось в том, чтобы из полного ФИО получить фамилию и инициалы имени и отчества – рисунок 1.6.

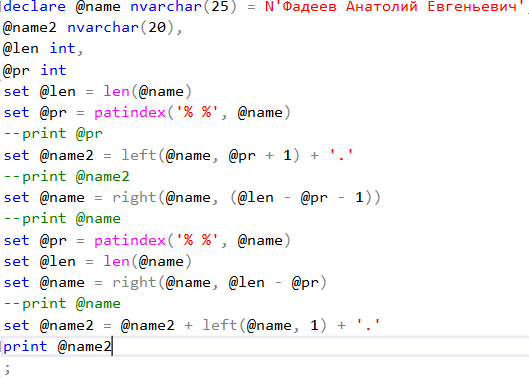


Рисунок 1.6 – Работа со строкой

Для этого нам понадобились функции выделение символов из строки RIGHT и LEFT, а также функция поиска по шаблону PATINDEX. Сначала мы находим положение первого пробела в строке и извлекаем из этой строки подстроку так, чтобы она захватывала первую букву имени, а затем добавляем знак «.». После этого «вырезаем» фамилию из исходной строки, ищем следующий пробел, извлекаем строку, которая содержит только отчество, а затем к нашей результирующей переменной добавляем первую букву отчества и символ «.».

Следующая задание – создание временной таблицы. Основное отличие временных таблиц от постоянных в том, что они хранятся в системной базе данных TEMPDB и не могут иметь внешние ключи.

Локальные временные таблицы имеют имена, начинающиеся с символа # и доступны только пользователю, ее создавшему.

Для её заполнения необходимо было использовать цикл WHILE, который также не нуждается в представлении – рисунок 1.7. Здесь же была использована функция rand(), которая возвращает случайное значение, floor(), выполняющая округление в меньшую сторону, а также replicate(), дублирующая в одной строке значение определенное количество раз.

Для обработки ошибок выполнения в сценарии T-SQL предусмотрена конструкция, состоящая из двух блоков: TRY и CATCH. Блок TRY содержит код T-SQL, в котором могут возникнуть ошибки, а блок CATCH – код, предназначенный для обработки ошибок.

Ошибка, возникающая в охраняемом коде, приводит к передаче управления в блок обработки ошибок. Пример представлен на рисунке 1.8.

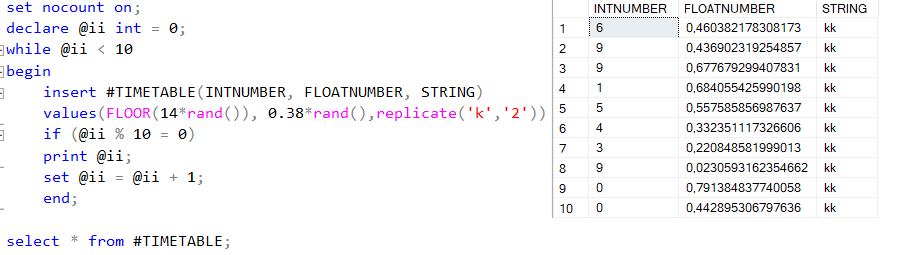


Рисунок 1.7 – заполнение таблицы с помощью цикла WHILE



Рисунок 1.8 – Использование конструкции TRY…CATCH

В одном из заданий необходимо было опробовать оператор RETURN. В тот момент, когда выполнение запроса достигает этого оператора, запрос автоматически завершается, даже если после него указан какой-либо код. Блок кода, расположенный после RETURN не выполняется.

**Вывод:**

В результате лабораторной работы было рассмотрено применение и реализация переменных в, а также их использование в запросах, проанализированы глобальные переменные, получены навыки в программировании на T-SQL, изучена обработка ошибок.