Лабораторная работа №4

ОСВОЕНИЕ ПРОГРАММИРОВАНИЯ С ПОМОЩЬЮ ВСТРОЕННОГО ЯЗЫКА TRANSACT SQL В MICROSOFT SQL SERVER

Цель работы – знакомство с основными принципами программирования в MS SQL Server средствами встроенного языка Transact SQL.

Содержание работы:

1. Знакомство с правилами обозначения синтаксиса команд в справочной системе MS SQL Server.
2. Изучение правил написания программ на Transact SQL.
3. Изучение правил построения идентификаторов, правил объявления переменных и их типов.
4. Изучение работы с циклами и ветвлениями.
5. Изучение работы с переменными типа Table и Cursor.
6. Проработка всех примеров, анализ результатов их выполнения.
7. Выполнение индивидуальных заданий по вариантам.

Пояснения к выполнению работы

Для освоения программирования используем пример базы данных Б**иблиотека** и примеры из стандартных таблиц торговых БД. При выполнении примеров и заданий обращайте внимание на соответствие названий БД, таблиц и других объектов проекта.

Специальные знаки и простейшие операторы в Transact SQL

Знак	**Назначение**	**Знак**	**Назначение**
*	Знак умножения	" "	В них заключают строковые зна чения, если SET QUOTED_IDENTIFIER OFF
-	Знак вычитания	‘ ’	В них заключают строковые значения
%	Остаток от деления двух чисел	<>	Не равно
+	Знак сложения или конкатенации (объединение двух строк в одну)	[ ]	Аналог кавычек, в них можно заключать названия идентификаторов, если в их названиях встречаются пробелы
=	Знак равенства или сравнения	!<	Не менее чем
<=	Меньше или равно	!>	Не более чем
>=	Больше или равно	>	Больше
!=	Не равно	<	Меньше
@	Ставится перед именем переменной	.	Разделяет родительские и подчиненные объекты
@@	Указывает на системные функции	/	Знак деления
--	Однострочный комментарий или комментарий с текущей позиции и до конца строки	/* */	Многострочный комментарий

Идентификаторы – это имена объектов, на которые можно ссылаться в программе, написанной на языке Transact SQL. Первый символ может со стоять из букв английского алфавита или “_”, “@”, “#”. Остальные дополнительно из цифр и «$».

Имя идентификатора не должно совпадать с зарезервированным словом.

Для ограничителей идентификаторов при установленном параметре

SET QUOTED_IDENTIFIER ON

можно использовать как квадратные скобки, так и одинарные кавычки, а строковые значения только в одинарных кавычках (режим по умолчанию).

Если использовать установленный параметр в режиме

SET QUOTED_IDENTIFIER OFF,

то в качестве ограничителей идентификаторов можно использовать только квадратные скобки, а строковые значения указываются в одинарных или двойных кавычках.

Переменные используются для сохранения промежуточных данных в хранимых процедурах и функциях. Все переменные считаются локальными. Имя переменной должно начинаться с @.

Объявление переменных

Синтаксис в обозначениях MS SQL Server:

DECLARE @имя_переменной_1 тип_переменной, …, @имя_переменной_N тип_переменной

Если тип переменной предполагает указание размера, то используется следующий синтаксис для объявления переменных:

DECLARE @имя_переменной_1 тип_переменной (размер), …, @имя_переменной_N тип_переменной(размер)

Пример:

```
DECLARE @a INT, @b NUMERIC(10,2)
DECLARE @str CHAR(20)
```

Присвоение значений переменным и вывод значений на экран

Присвоение с помощью **SET** – обычное присвоение, синтаксис: SET @имя_переменной = значение.

Пример:

```
DECLARE @a INT, @b NUMERIC(10,2) SET @a = 20
SET @b = (@a+@a)/15
SELECT @b --вывод на экран результата
```

Присвоение с помощью **SELECT** – помещение результата запроса в переменную. Если в результате выполнения запроса не будет возвращено ни одной строки, то значение переменной не меняется, т.е. остается старым.

Пример:

```
DECLARE @a INT
SELECT @a = COUNT(*) FROM Authors
```

Пример:

```
DECLARE @str CHAR(30)
```

```
SELECT @str = name FROM Authors
```

В данном примере в переменную поместится последнее значение из результата запроса.

Сочетание ключевых слов SET и SELECT

Пример:

```
DECLARE @a INT
SET @a = (SELECT COUNT(*) FROM Authors)
```

Работа с датой и временем

Оператор SET DATEFORMAT dmy | ymd | mdy задает порядок следо вания компонентов даты.

Пример:

```
SET DATEFORMAT dmy
DECLARE @d DateTime
SET @d = '31.01.2005 13:23:15'
SET @d = @d+1
SELECT @d
```

Создание временной таблицы через переменную типа TABLE

Объявляется через DECLARE с указанием в скобках столбцов табли цы, их типов, размеров, значений по умолчанию, а также индексов типа PRIMARY KEY или UNIQUE.

Пример:

```
DECLARE @mytable TABLE(id INT, myname CHAR(20) DEFAULT 'Введите имя')
INSERT INTO @mytable(id) VALUES (1)
SELECT * FROM @mytable
```

Пример:

```
DECLARE @mytable TABLE(id INT, myname CHAR(20) DEFAULT 'Введите имя')
INSERT @mytable SELECT Code_publish, City FROM Publishing_house
SELECT * FROM @mytable
```

Преобразование типов переменных

Функция **CAST** возвращает значение, преобразованное к указанному типу:
CAST(@переменная или значение AS требуемый_тип_данных)

Пример:

```
DECLARE @d DateTime, @str CHAR(20)
SET @d = '31.01.2005 13:23:15'
SET @str = CAST(@d AS CHAR(20))
SELECT 2str
```

Функция **CONVERT** возвращает значение, преобразованное к указан ному типу по заданному формату. Изучить дополнительно, по желанию.

Операторские скобки

```
BEGIN
/* в них нельзя помещать команды, изменяющие структуры объектов БД.
Операторские скобки должны содержать хотя бы один оператор. Требуются
для конструкций поливариантных ветвлений, условных и циклических
конструкций
*/
END
```

Условная конструкция IF

Синтаксис:

IF условие

Набор операторов1

ELSE

Набор операторов2

Пример:

```
DECLARE @a INT DECLARE @str CHAR(30)
SET @a = (SELECT COUNT(*) FROM Authors)
IF @a >10
BEGIN
 SET @str = 'Количество авторов больше 10'
 SELECT @str
END
ELSE
BEGIN
 SET @str = 'Количество авторов = ' + str(@a)
 SELECT @str
END
```

Цикл WHILE

Синтаксис: WHILE *Условие*

Набор операторов1

BREAK

Набор опреторов2

CONTINUE

Конструкции BREAK и CONTINUE являются необязательными.

Цикл можно принудительно остановить, если в его теле выполнить команду BREAK. Если же нужно начать цикл заново, не дожидаясь выпол нения всех команд в теле, необходимо выполнить команду CONTINUE.

Пример:

```
DECLARE @a INT
SET @a = 1
WHILE @a <100
 BEGIN
  PRINT @a - вывод на экран значения переменной
  IF (@a>40) AND (@a<50)
   BREAK --выход и выполнение 1-й команды за циклом
  ELSE
```

```
   SET @a = @a+rand()*10
    CONTINUE
 END
PRINT @a
```

Объявление курсора

CURSOR – это набор строк, являющийся результатом выполнения за проса. В один момент времени доступна лишь одна строка (текущая), по курсору можно передвигаться и получать доступ к элементарным данным. При объявлении курсора создается временная копия данных, которая сохраняется в БД tempdb.
Динамический курсор – данные в курсоре могут быть изменены.
Статический курсор – данные в курсоре не меняются.

Стандартный способ объявления курсора, синтаксис в обозначениях
DECLARE cursor_name [INSENSITIVE] [SCROLL] CURSOR
FOR select_statement
[FOR { READ ONLY | UPDATE [OF column_name [,...n]] }]

Примеры объявления курсоров:

```
DECLARE MyCursor1 CURSOR FOR (SELECT * FROM Authors)
/*объявили курсор с названием MyCursor1, который содержит всю
информацию об авторах, двигаться по нему можно только от первой записи
вниз до последней. Курсор является динамическим.*/

DECLARE MyCursor1 INSENSITIVE CURSOR FOR (SELECT * FROM Authors)
/*объявили курсор с названием MyCursor1, который содержит всю
информацию об авторах, двигаться по нему можно только от первой записи
вниз до последней. Курсор является статическим.*/

DECLARE MyCursor1 SCROLL CURSOR FOR (SELECT * FROM Authors)
/*объявили курсор с названием MyCursor1, который содержит всю
информацию об авторах, двигаться по нему можно в любом направлении.
Курсор является динамическим.*/

DECLARE MyCursor1 INSENSITIVE SCROLL CURSOR FOR (SELECT * FROM
Authors)
/*объявили курсор с названием MyCursor1, который содержит всю
информацию об авторах, двигаться по нему можно в любом направлении.
Курсор является статическим.*/

DECLARE MyCursor1 CURSOR FOR (SELECT * FROM Authors) FOR READ ONLY
/*объявили курсор с названием MyCursor1, который содержит всю
информацию об авторах, двигаться по нему можно только от первой записи
вниз до последней. Курсор является динамическим. Данные доступны
только для чтения.*/

DECLARE MyCursor1 CURSOR FOR (SELECT * FROM Authors) FOR UPDATE
/*объявили курсор с названием MyCursor1, который содержит всю
информацию об авторах, двигаться по нему можно только от первой записи
вниз до последней. Курсор является динамическим. Данные курсора можно
менять.*/
```

Операторы для работы с курсором

Прежде чем обратиться к данным курсора, его нужно после объявления открыть.

Синтаксис оператора OPEN в обозначениях MS SQL Server:

OPEN { { [GLOBAL] *cursor_name* } | *cursor_variable_name* }

Пример:

```
DECLARE MyCursor1 CURSOR FOR (SELECT * FROM Authors)
OPEN MyCursor1
```

После прекращения работы с курсором, его нужно закрыть. Курсор остается доступным для последующего использования в рамках процедуры или триггера, в котором он создан.

Синтаксис оператора CLOSE в обозначениях MS SQL Server:

CLOSE { { [GLOBAL] *cursor_name* } | *cursor_variable_name* }

Пример:

```
DECLARE MyCursor1 CURSOR FOR (SELECT * FROM Authors)
OPEN MyCursor1
--здесь операторы работы с курсором
CLOSE MyCursor1
```

Если курсором больше не будут пользоваться, то его необходимо уничтожить и освободить переменную.

Синтаксис оператора DEALLOCATE в обозначениях MS SQL Server: DEALLOCATE { { [GLOBAL] *cursor_name* } | *@cursor_variable_name* }

Пример:

```
DECLARE MyCursor1 CURSOR FOR (SELECT * FROM Authors)
OPEN MyCursor1
--здесь операторы работы с курсором
CLOSE MyCursor1
DEALLOCATE MyCursor1
```

FETCH – оператор движения по записям курсора и извлечения данных те кущей записи в указанные переменные.

Синтаксис оператора FETCH в обозначениях MS SQL Server: FETCH

[[NEXT | PRIOR | FIRST | LAST

| ABSOLUTE { *n* | **@***nvar* }

| RELATIVE { *n* | **@***nvar* }

] FROM

]

{ { [GLOBAL] *cursor_name* } | *@cursor_variable_name* } [INTO *variable_name* [**,**...*n*]]

Пример:

```
DECLARE MyCursor1 SCROLL CURSOR FOR (SELECT * FROM Authors)
DECLARE @i BIGINT, @s CHAR(20), @d smalldatetime
OPEN MyCursor1
FETCH FIRST FROM MyCursor1 INTO @i, @s, @d
PRINT @i
PRINT @s
PRINT @d
CLOSE MyCursor1
DEALLOCATE MyCursor1
```

@@FETCH_STATUS – данная функция определяет признак конца или начала текущего курсора. Функция принимает одно из следующих значений: 0 – находимся в пределах курсора, не в конце; 1 – попытка выйти за пределы первой записи вверх (в никуда); 2 – попытка выйти за пределы последней записи вниз (в никуда).

Пример:

```
DECLARE MyCursor1 SCROLL CURSOR FOR (SELECT * FROM Authors)
DECLARE @i BIGINT, @s CHAR(20), @d smalldatetime
OPEN MyCursor1
FETCH FIRST FROM MyCursor1 INTO @i, @s, @d
WHILE @@FETCH_STATUS = 0
  BEGIN
   FETCH NEXT FROM MyCursor1 INTO @i, @s, @d
   PRINT @i
   PRINT @s
   PRINT @d
  END
CLOSE MyCursor1
DEALLOCATE MyCursor1
```

Встроенные функции

Встроенные функции, имеющиеся в распоряжении пользователей при работе с SQL, можно условно разделить на следующие группы:

- математические функции;
- строковые функции;
- функции для работы с датой и временем;
- функции конфигурирования;
- функции системы безопасности;
- функции управления метаданными;
- статистические функции.

Использование функций для работы со строковыми переменными

Краткий обзор строковых функций

Название функции	**Действие, выполняемое функцией**
ASCII	Возвращает код ASCII левого символа строки
CHAR	По коду ASCII возвращает символ
CHARINDEX	Определяет порядковый номер символа, с которого начинается вхождение подстроки в строку
DIFFERENCE	Возвращает показатель совпадения строк
LEFT	Возвращает указанное число символов с начала строки
LEN	Возвращает длину строки
LOWER	Переводит все символы строки в нижний регистр
LTRIM	Удаляет пробелы в начале строки
NCHAR	Возвращает по коду символ Unicode
PATINDEX	Выполняет поиск подстроки в строке по указанному шаблону
REPLACE	Заменяет вхождения подстроки на указанное значение
QUOTENAME	Конвертирует строку в формат Unicode
REPLICATE	Выполняет тиражирование строки определенное число раз

REVERSE	Возвращает строку, символы которой записаны в обратном поряд ке
RIGHT	Возвращает указанное число символов с конца строки
RTRIM	Удаляет пробелы в конце строки
SOUNDEX	Возвращает код звучания строки
SPACE	Возвращает указанное число пробелов
STR	Выполняет конвертирование значения числового типа в символь ный формат
STUFF	Удаляет указанное число символов, заменяя новой подстрокой
SUBSTRING	Возвращает для строки подстроку указанной длины с заданного символа
UNICODE	Возвращает Unicode-код левого символа строки
UPPER	Переводит все символы строки в верхний регистр

Использование функций для работы с числами

Краткий обзор математических функций

Название функции	**Действие, выполняемое функцией**
ABS	Вычисляет абсолютное значение числа
ACOS	Вычисляет арккосинус
ASIN	Вычисляет арксинус
ATAN	Вычисляет арктангенс
ATN2	Вычисляет арктангенс с учетом квадратов
CEILING	Выполняет округление вверх
COS	Вычисляет косинус угла
COT	Возвращает котангенс угла
DEGREES	Преобразует значение угла из радиан в градусы
EXP	Возвращает экспоненту
FLOOR	Выполняет округление вниз
LOG	Вычисляет натуральный логарифм
LOG10	Вычисляет десятичный логарифм
PI	Возвращает значение "пи"
POWER	Возводит число в степень
RADIANS	Преобразует значение угла из градуса в радианы
RAND	Возвращает случайное число
ROUND	Выполняет округление с заданной точностью
SIGN	Определяет знак числа
SIN	Вычисляет синус угла
SQUARE	Выполняет возведение числа в квадрат
SQRT	Извлекает квадратный корень
TAN	Возвращает тангенс угла

Использование функций для работы с типом дата/время

Краткий обзор основных функций для работы с датой и временем

Название функции	Действие, выполняемое функцией
DATEADD	Добавляет к дате указанное значение дней, месяцев, часов и т.д.
DATEDIFF	Возвращает разницу между указанными частями двух дат

DATENAME	Выделяет из даты указанную часть и возвращает ее в символьном формате
DATEPART	Выделяет из даты указанную часть и возвращает ее в числовом формате
DAY	Возвращает число из указанной даты
GETDATE	Возвращает текущее системное время
ISDATE	Проверяет правильность выражения на соответствие одному из возможных форматов ввода даты
MONTH	Возвращает значение месяца из указанной даты
YEAR	Возвращает значение года из указанной даты
MINUTE	Возвращает значение минут из указанной даты/времени
HOUR	Возвращает значение часов из указанной даты/времени
SECOND	Возвращает значение секунд из указанной даты/времени

Варианты заданий к лабораторной работе

Общие сведения

Для выполнения заданий ориентироваться на вариант и список номеров заданий.

Список вариантов заданий

Вариант	Список номеров упражнений												
1	1	6	11	16	21	26	31	36	41	46	51	56	61
2	2	7	12	17	22	27	32	37	42	47	52	57	62
3	3	8	13	18	23	28	33	38	43	48	53	58	63
4	4	9	14	19	24	29	34	39	44	49	54	59	64
5	5	10	15	20	25	30	35	40	45	50	55	60	64
6	6	11	16	21	26	31	36	41	46	51	56	61	1
7	7	12	17	22	27	32	37	42	47	52	57	62	2
8	8	13	18	23	28	33	38	43	48	53	58	63	3
9	9	14	19	24	29	34	39	44	49	54	59	64	4
10	10	15	20	25	30	35	40	45	50	55	60	65	5
11	2	6	12	16	22	26	32	36	42	46	52	56	62
12	1	5	11	15	21	25	31	35	41	45	51	55	61
13	3	7	13	17	23	27	33	37	43	47	53	57	63

Специальные знаки и простейшие операторы в Transact SQL

1. Проверить работу описанной установки SET QUOTED_IDENTIFIER.
2. Проверить работу описанной установки SET DATEFIRST.

Объявление переменных

3. Объявить переменную Perem1 типа денежный, а переменную Perem2 типа число с целой частью равной 8 и дробной частью равной 2.
4. Объявить переменную Perem1 типа строка длиной 100, а переменную Perem2

типа длинное целое.

5. Объявить переменную Perem1 типа динамическая строка с максимальной длиной 1000, а переменную Perem2 типа целое число.

6. Объявить переменную Perem1 типа строка длиной 30, а переменную Perem2 типа число с целой частью равной 10 и дробной частью равной 3.

7. Объявить переменную Perem1 типа дата/ время, а переменную Perem2 типа число в диапазоне от 0 до 255.

Присвоение значений переменным и вывод значений на экран

8. Подсчитать среднее значение атрибута по своему выбору из своей базы данных (например, среднюю цену) с помощью запроса SELECT и умножить ее на значение 123,34, которое необходимо сохранить в отдельной переменной, вывести значение переменной на экран.

9. Подсчитать сумму значений поля по своему выбору из своей базы данных (например, сумму закупок), результат поместить в переменную, вывести значение переменной на экран.

10. Подсчитать количество значений поля по своему выбору из своей базы данных (например, количество книг), результат поместить в переменную, вывести значение переменной на экран.

11. Определить минимальное значение поля по своему выбору из своей базы данных (например, минимальную дату рождения автора в справочнике авторов), результат поместить в переменную, вывести значение переменной на экран.

Сочетание ключевых слов SET и SELECT

12. Выполнить для таблицы из своей БД задание, аналогичное следующему: Подсчитать количество поставщиков, результат поместить в переменную.

13. Выполнить для таблицы из своей БД задание, аналогичное следующему: Подсчитать сумму закупок, результат поместить в переменную.

14. Выполнить для таблицы из своей БД задание, аналогичное следующему: Подсчитать среднюю цену в таблице покупок, результат поместить в переменную.

15. Выполнить для таблицы из своей БД задание, аналогичное следующему: Подсчитать максимальную стоимость товаров в закупке, результат поместить в переменную.

Работа с датой и временем

16. Определить переменную Date1 типа дата/время. Присвоить ей значение даты 31.12.2016 в формате dd.mm.yyyy.

17. Определить переменную Date1 типа дата/время. Присвоить ей значение даты 31.12.2016 в формате mm.dd.yyyy.

18. Определить переменную Date1 типа дата/время. Присвоить ей значение даты 31.12.2016 в формате yyyy.mm.dd.

Создание временной таблицы через переменную типа TABLE

19. Создать локальную таблицу с названием TEMP и полями типа, дата/время, длинное целое, строка. Добавить в нее две записи с данными и вывести результат на экран.

20. Создать локальную таблицу с названием TEMP и полями типа длинное целое, строка и значением по умолчанию «введите что-нибудь», денежный. Добавить в нее две записи с данными и вывести результат на экран.

21. Создать локальную таблицу с названием TEMP и полями типа целое, динамическая строка, бит со значением по умолчанию «1». Добавить в нее две записи с данными и вывести результат на экран.

22. Создать локальную таблицу с названием TEMP и полями типа, дата/время, длинное целое, строка. Добавить в нее две записи с данными и вывести результат на экран.

23. Создать локальную таблицу с названием TEMP и полями типа, дата/время, длинное целое с автонаращиванием, динамическая строка. Добавить в нее две записи с данными и вывести результат на экран.

Преобразование типов переменных

24. Объявить переменные типа FLOAT, CHAR, TINYINT. Присвоить значения, соответствующие типам. Выполнить преобразование переменных типа FLOAT, CHAR, TINYINT в INT, DATETIME, BIT соответственно и вывести результат на экран.

25. Объявить переменные типа INT, DATETIME, BIT. Присвоить значения, соответствующие типам. Выполнить преобразование переменных типа INT, DATETIME, BIT в FLOAT, CHAR, TINYINT соответственно и вывести результат на экран.

26. Объявить переменные типа NUMERIC, VARCHAR, DATETIME. Присвоить значения, соответствующие типам. Выполнить преобразование переменных типа NUMERIC, VARCHAR, DATETIME в FLOAT, CHAR, BIGINT соответственно и вывести результат на экран.

27. Объявить переменные типа BIT, NVARCHAR, DATETIME. Присвоить значения, соответствующие типам. Выполнить преобразование переменных типа BIT, NVARCHAR, DATETIME в FLOAT, INT, BIGINT со ответственно и вывести результат на экран.

Условная конструкция IF

28. Выполнить для таблицы из своей БД задание, аналогичное следующему: Подсчитать количество поставщиков в таблице Поставщики. Если их в таблице от 2 до 5, то ничего не сообщать, в противном случае вывести сообщение вида "В таблице ... поставщиков" (вместо многоточия поставить точное количество поставщиков).

29. Выполнить для таблицы из своей БД задание, аналогичное следующему: Подсчитать сумму стоимости книг. Если полученная сумма в диапазоне от 1000 до 5000, то ничего не сообщать, в против ном случае вывести сообщение вида "Сумма стоимости книг = …" (вместо многоточия поставить точную сумму).

30. Выполнить для таблицы из своей БД задание, аналогичное следующему: Подсчитать среднюю стоимость книги. Если полученная стоимость в диапазоне от 50 до 150, то ничего не сообщать, в противном случае вывести сообщение вида "Средняя стоимость книги = …" (вместо многоточия поставить точную среднюю стоимость).

31. Выполнить для таблицы из своей БД задание, аналогичное следующему: Определить минимальную стоимость закупки в таблице поставок. Если полученная стоимость в диапазоне от 200 до 300, то ничего не сообщать, в противном случае вывести сообщение вида "Минимальная стоимость закупки = …" (вместо многоточия поставить точную стоимость).

Цикл WHILE

32. Определить количество записей в любой таблице по своему выбору. Пока записей меньше 15, делать в цикле добавление записи в таблицу с автоматическим наращиванием значения ключевого поля, а вместо имени автора ставить значение 'Значение не известно'.

33. Определить количество записей в таблице по своему выбору. Пока записей

меньше 20, делать в цикле добавление записи в таблицу с автоматическим наращиванием значения ключевого поля, а вместо названия издательства ставить значение 'не известно'.

34. Определить количество записей в таблице по своему выбору. Пока записей меньше 17, делать в цикле добавление записи в таблицу с автоматическим наращиванием значения ключевого поля, а вместо названия поставщика ставить значение 'не известен'.

Объявление курсора

35. Создать статический курсор по данным таблицы по своему выбору из своей БД (например, Books с полями Code_book, Title_book).
36. Создать динамический курсор по данным таблицы по своему выбору из своей БД (например, таблица поставщиков с полями Name_delivery, Name_company).
37. Создать статический курсор по данным объединения 2-х таблиц по своему выбору из своей БД (пример - Books и Authors с полями Code_book, Title_book, Name_author).
38. Создать статический курсор по данным объединения 2-х таблиц по своему выбору из своей БД (пример - Books и Authors с полями Code_book, Title_book, Name_author).

Операторы для работы с курсором

39. Выполнить для таблицы из своей БД задание, аналогичное следующему: Создать динамический курсор для чтения по данным таблицы Books с полями Code_book, Name_book. Вывести данные 3-й записи.

40. Выполнить для таблицы из своей БД задание, аналогичное следующему: Поместить в курсор данные таблицы Поставки. Перебрать все записи таблицы Поставки. Просуммировать значения произведений полей Цена и Количество и результат сохранить в переменной Sum_table, которую после суммирования вывести на экран. Закрыть и удалить из памяти курсор.

41. Выполнить для таблицы из своей БД задание, аналогичное следующему: Объявить статический курсор по данным таблиц Authors и Books. Вывести данные 5-й записи.

Использование функций для работы со строковыми переменными

Для выполнения этого блока заданий в начале программы, которую вы создаете, объявите переменную типа varchar и присвойте ей в качестве значения строку с любым базовым текстом, который будет анализироваться и/или исправляться в заданиях.

42. Удалить в тексте лишние пробелы. Лишними считаются те, которые идут непосредственно за пробелом. Подсчитать количество исправлений.

43. Подсчитать количество встреч каждой из следующих букв: "а", "в", "и", "п" в базовом тексте.

44. Подсчитать доли процентов встречи следующих букв: "е", "о", если суммарный процент встречаемости всех этих букв равен 100% или процент встречаемости е% + о% равен 100%.

45. По правилам оформления машинописных текстов перед знаками .,!?:; пробелы не ставятся, но обязательно ставятся после этих знаков. Удалите лишние пробелы. Подсчитать количество исправлений.

46. По правилам оформления машинописных текстов перед знаками .,!?:; пробелы не ставятся, но обязательно ставятся после этих знаков. Расставьте недостающие пробелы. Подсчитать количество исправлений.

47. Найти из исходного текста второе предложение и вернуть его в переменную Perem, а также вывести на экран весь исходный текст и найденное предложение.

48. Удалить из базового текста 2, 4, 6, 8 слова.

49. Удалить из базового текста 3, 5, 7, 10 слова.

50. Вставить в базовый текст вместо букв «а» «АА».

51. Вставить в базовый текст вместо букв «е» и «о» «ББ».

52. Поменять местами первое и последнее слова в базовом тексте.

Использование функций для работы с числами

53. Вывести значение формулы, переменные которой нужно описать и присвоить произвольные значения.

$$v = v_0 \cdot e^{\sqrt{\frac{R \cdot T}{45}}}$$

.

54. Подсчитать значение формулы, переменные которой нужно описать и присвоить произвольные значения.

$$y = 2^x \cdot \exp(\ln(x^2))$$

55. Подсчитать значение формулы, переменные которой нужно описать и присвоить произвольные значения.

$$y = \frac{sin(a)}{x^2 - b^3} \cdot a.$$

56. Подсчитать значение формулы, переменные которой нужно описать и присвоить произвольные значения.

$$y = \sum_{n=1}^{10} I_n \cdot a$$

.

57. Подсчитать значение формулы, переменные которой нужно описать и присвоить произвольные значения.

$$y = \frac{tg(a)}{a + b - c} \cdot \sqrt{a \cdot b \cdot c}.$$

58. Подсчитать значение формулы, переменные которой нужно описать и присвоить произвольные значения.

$$y = \sqrt{\sin(a) \cdot \exp(b \cdot c)}$$

59. Подсчитать значение формулы, переменные которой нужно описать и присвоить произвольные значения.

$$y = x^4 \cdot ln(a) - b \cdot c$$

60. Подсчитать значение формулы, переменные которой нужно описать и присвоить произвольные значения.

$$y = \frac{\sqrt{x - a}}{b^3}$$

61. Подсчитать значение формулы, переменные которой нужно описать и присвоить произвольные значения.

$$y = \frac{a \cdot cos(x)}{b^2 - a^2} \cdot sin(x)$$

Использование функций для работы с типом дата/время

62. Вывести на экран название текущего месяца и текущее время. За писать в таблицу Purchases в поле Date_order одинаковую дату поступления, которая равна 12.03.2000.

63. Разобрать на отдельные составляющие текущую дату и время и вывести значения на экран в следующем порядке (вместо многоточий): "Сегодня: День = …, Месяц = …, Год = …, Часов = …, Минут = …,Секунд= …"

64. В исходный текст, сохраненный в переменной Perem, после слова "время" вставить текущее время. Результат сохранить в той же переменной Perem и вывести на экран.

Контрольные вопросы

1. Как записываются комментарии в языке Transact SQL?
2. Какие типы встроенных функций существуют в TRANSACT SQL? К какому типу относятся функции RTRIM, STR и UPPER? К какому типу относятся функции GETDATE, MONTH и DATEADD?
3. В чем отличие присвоения значений переменным с помощью SET и SELECT?
4. Чем отличаются структуры Table и Cursor?
5. Опишите основные приемы работы с набором строк Cursor.
6. Какие вы знаете операторы условных конструкций и циклов в Transact SQL?