Учреждение образования
«Белорусский государственный университет информатики и радиоэлектроники»
Кафедра инженерной психологии и эргономики

Отчет
По лабораторной работе №4

Проектирование базы данных

Выполнила:
Студент ФКП, гр.914302
Воробей Д.А.,
Проверила: Лукашевич А.Э.

Минск 2022

Цель работы: Изучить используемый в реляционных СУБД встроенный язык программирования Transact-SQL для написания программ в MS SQL Server. Изучить правила построения идентификаторов, правила объявления переменных и их типов. Изучить принципы работы с циклами и ветвлениями. Изучить работу с переменными типа Table. Изучить синтаксис и семантику функций и хранимых процедур Transact–SQL: способов их идентификации, методов задания и спецификации параметров и возвращаемых значений, и вызовов функций и хранимых процедур.Изучение синтаксиса и семантики функций и хранимых процедур Transact–SQL: способов их идентификации, методов задания и спецификации параметров и возвращаемых значений, кодирования тела и вызовов функций и хранимых процедур, применение команд для создания, изменения и удаления системных и пользовательских как скалярных, так и табличных (с одной Inline или несколькими Multi – statement командами в теле) функций, системных, пользовательских, временных (локальных или глобальных) и расширенных хранимых процедур, а также приобретение навыков программирования, отладки, тестирования и включения в группу или подключения библиотеки функций и хранимых процедур.

Ход работы:

```
--1 запрос для создания временной таблицы через переменную типа TABLE;

DECLARE @mytable TABLE (id INT, myname CHAR(20) DEFAULT 'Иванов Иван')
INSERT INTO @mytable(id) VALUES (1)
INSERT INTO @mytable(id, myname) VALUES (2,'Игорь Троцкий')
SELECT * FROM @mytable


--2 запроса с использованием условной конструкции IF;

DECLARE @a INT
DECLARE @str CHAR(50)
SET @a = (SELECT COUNT(*) FROM Customers)
IF @a > 5 BEGIN
SET @str = 'Суммарное количество клиентов больше 5' SELECT @str
END ELSE BEGIN
SET @str = 'Суммарное количество клиентов = ' + str(@a) SELECT @str
END


DECLARE @b INT
DECLARE @string CHAR(80)
SET @b = (SELECT COUNT(*) FROM Tours)
IF @b < 1 BEGIN
SET @string = 'В системе еще не зарегистрирован ни один тур' SELECT @string
END ELSE BEGIN
SET @string = 'В системе зарегистрировано  ='+str(@b)+' туров' SELECT @string
END
```

```sql
--1 запрос для создания скалярной функции;

    CREATE FUNCTION GetSumm
     (@name varchar(50))
    RETURNS numeric(10,2)
     BEGIN
      DECLARE @PricePerPerson numeric(10,2)
      SELECT @PricePerPerson = Price/PersonsQuantity
      FROM TravelPackages
      ---WHERE [PricePerPerson]=@name
      RETURN @PricePerPerson
     END
     GO
```

```sql
--1 запрос для создания функции, которая возвращает табличное значение;

   CREATE FUNCTION [dbo].[test_tabl]
   ( @id INT)
   RETURNS TABLE
   AS
   RETURN
   ( SELECT * FROM HouseType WHERE HouseCategory = 'Люкс')
   GO
   SELECT * FROM dbo.test_tabl (1)
   GO
```

```sql
--2 запроса с использованием цикла WHILE;

DECLARE @c INT SET @c = 1 WHILE @c <100
BEGIN
PRINT @c -- вывод на экран значения переменной I
IF (@c>40) AND (@c<50)
BREAK --выход и выполнение 1-й команды за циклом
ELSE
SET @c = @c+rand()*10
CONTINUE
END
PRINT @c


DECLARE @number INT, @factorial INT
SET @factorial = 1;
SET @number = (SELECT COUNT(*) FROM Customers);

WHILE @number > 0
    BEGIN
        SET @factorial = @factorial * @number
        SET @number = @number - 1
    END;
PRINT @factorial
```

```
--2 запроса для создания процедуры без параметров;

    CREATE PROCEDURE Count_Customers AS
    SELECT COUNT(*) FROM Customers
    GO
    EXEC Count_Customers
    GO

    CREATE PROCEDURE Count_Tours AS
    SELECT COUNT(*) FROM Tours
    GO
    EXEC Count_Tours
    GO
```

```
--2 запроса для создания процедуры с входным параметром;


    CREATE PROCEDURE Count_PricePerDay_Breakfast @Day_price INT
    AS
    BEGIN
    SELECT count(FoodType) FROM Tours
    WHERE FoodType='Завтраки' and PricePerDay>=@Day_price
    END
    GO
    EXEC Count_PricePerDay_Breakfast 1000
    GO

    CREATE PROCEDURE House_level @House_level AS INT
    AS
    SELECT COUNT(*) FROM BoardingHouses WHERE BoardingHouseLevel>=@House_level
    GO
    EXEC House_level 2
    GO


    --2 запроса для создания процедуры с входными параметрами и RETURN;

        CREATE PROCEDURE checkname @param INT
        AS
        IF (SELECT CName FROM Customers WHERE CName = @param) =
        'Петр'
        RETURN 1
        ELSE
        RETURN 2

        DECLARE @return_status INT
        EXEC @return_status = checkname 1
        SELECT 'Return Status' = @return_status
        GO


        CREATE PROCEDURE checknametour @param INT
        AS
        IF (SELECT * FROM Tours WHERE TourName = @param) =
        'Тибет'
        RETURN 1
        ELSE
        RETURN 2

        DECLARE @return_status INT
        EXEC @return_status = checkname 1
        SELECT 'Return Status' = @return_status
        GO
```

```
--2 запроса для создания процедуры обновления данных в таблице базы данных UPDATE;

    CREATE PROC update_proc AS
    UPDATE Tours SET PricePerDay = PricePerDay-50
    GO

    CREATE PROC update_house_category AS
    UPDATE HouseType SET HouseCategory = 'Люкс'
    GO


--2 запроса для создания процедуры извлечения данных из таблиц базы данных SELECT;

    CREATE PROCEDURE CityCustomers
    AS
    BEGIN
        SELECT * FROM Customers
        WHERE City ='Минск'
    END
    GO

    CREATE PROCEDURE SPAvailable
    AS
    BEGIN
        SELECT * FROM BoardingHouses
        WHERE SwimmingPoolAvailability ='Да'
    END
    GO
```

```
  --Часть 2

  -- task 1

  CREATE FUNCTION Calculator(@oprd_1 bigint,@oprd_2 bigint,@operator char(1))
  RETURNS bigint
  BEGIN

  DECLARE @result bigint
  SET @result =
  CASE @operator
      WHEN '+' THEN @oprd_1 + @oprd_2
      WHEN '-' THEN @oprd_1 - @oprd_2
      WHEN '*' THEN @oprd_1 * @oprd_2
      WHEN '/' THEN @oprd_1 / @oprd_2
      ELSE 0
  END
  Return @result
  END

  GO
```

```sql
--Часть 2

-- task 1

CREATE FUNCTION Calculator(@oprd_1 bigint,@oprd_2 bigint,@operator char(1))
RETURNS bigint
BEGIN

DECLARE @result bigint
SET @result =
CASE @operator
    WHEN '+' THEN @oprd_1 + @oprd_2
    WHEN '-' THEN @oprd_1 - @oprd_2
    WHEN '*' THEN @oprd_1 * @oprd_2
    WHEN '/' THEN @oprd_1 / @oprd_2
    ELSE 0
END
Return @result
END

GO

-- testing
SELECT dbo.Calculator(10,5, '/') as result


-- task 2

GO

CREATE FUNCTION DYNTAB (@tour_name char(50))
RETURNS TABLE
AS
RETURN (SELECT TourName FROM Tours WHERE TourName = @tour_name)

GO

-- testing

SELECT * FROM DYNTAB ('Тибет')

GO
```

```sql
-- task 3

CREATE FUNCTION parse_string (@input_string nvarchar(500))
RETURNS @tabl TABLE
    (Number int IDENTITY (1,1) NOT NULL,
    Substr nvarchar (30))
AS
BEGIN
    DECLARE @str_1 nvarchar(500), @pos int
    WHILE CHARINDEX(' ', @input_string) > 0
     BEGIN
        SET @pos  = CHARINDEX(' ', @input_string)
        SET @str_1 = SUBSTRING(@input_string, 1, @pos-1)

        INSERT INTO @tabl VALUES(@str_1)

        SET @input_string = SUBSTRING(@input_string, @pos+1, LEN(@input_string) - @pos)
     END
    INSERT INTO @tabl VALUES(@input_string)
    RETURN
END

GO

-- testing

SELECT * FROM dbo.parse_string('Ветер мчится в даль !')

GO
```

Вывод: в рамках выполнения лабораторной работы изучены механизмы реализации хранимых процедур и функций на базе языка Transact-SQL, изучить принцип работы с переменными типа TABLE и реализованы функции и хранимые процедуры в рамках процессов предметной области.