Учреждение образования

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

# Отчет

По лабораторной работе №3

# ЯЗЫК SQL. СОЗДАНИЕ ТАБЛИЦ И ОГРАНИЧЕНИЙ НА SQL. СОЗДАНИЕ ЗАПРОСОВ НА ВЫБОРКУ. ОТБОР СТРОК ПО УСЛОВИЮ. СОЗДАНИЕ МНОГОТАБЛИЧНЫХ ЗАПРОСОВ. ЗАПРОСЫ НА СОЕДИНЕНИЕ. СОЗДАНИЕ ЗАПРОСОВ НА ГРУППИРОВКУ И СОРТИРОВКУ ДАННЫХ. ЗАПРОСЫ НА ИЗМЕНЕНИЕ. ИСПОЛЬЗОВАНИЕ ВСТРОЕННЫХ ФУНКЦИЙ

Выполнила:

Студент ФКП, гр.914302

Тумащик А.В.

Проверила: Лукашевич А.Э.

# Минск 2022

## Цель работы

Изучение структурированного языка запросов Transact - SQL, являющегося основой системы программирования SQL Server, и приобретение навыков применение инструментальных средств разработки и программирования объектов создаваемых баз данных. Изучить SQL- операторы для работы с таблицами и индексами. Изучить sql- команды для создания, изменения и удаления таблиц. Изучить используемые в SQL Server типы ограничений. Изучить SQL-операторы для работы с ограничениями.

## Исходные данные

Исходными данными является индивидуальное задание и результат предыдущих лабораторных работ.

## Используемые программы

Программа Visual Studio и установленный сервер Microsoft SQL Server.

USE supermarket

GO

-- part 1

-- 1

DECLARE @mytable TABLE (

id INT,

name NVARCHAR(MAX) DEFAULT 'Temporary table name'

)

INSERT INTO @mytable(id)

VALUES (1)

INSERT INTO @mytable(id, name)

VALUES (2, 'name')

SELECT \*

FROM @mytable

GO

-- 2

DECLARE @a INT

DECLARE @str NVARCHAR(MAX)

SET @a = (SELECT COUNT(\*)

FROM departments)

IF @a > 5

BEGIN

SET @str = 'Count of lines less then 5'

PRINT @str

END

ELSE

BEGIN

SET @str = 'Count of lines = ' + str(@a)

PRINT @str

END

GO

DECLARE @b INT

DECLARE @string NVARCHAR(MAX)

SET @b = (SELECT COUNT(\*)

FROM employees)

IF @b < 1 BEGIN

SET @string = 'Zero emloyees in database'

PRINT @string

END ELSE BEGIN

SET @string = 'Count of emloyees =' + str(@b)

PRINT @string

END

GO

-- 3

DECLARE @n INT

SET @n = 1

WHILE @n <100

BEGIN

PRINT @n

IF (@n > 40) AND (@n < 50)

BREAK

ELSE

SET @n = @n + rand() \* 10

CONTINUE

END

PRINT @n

GO

DECLARE @number INT

DECLARE @factorial INT

SET @factorial = 1;

SET @number = 10;

WHILE @number > 0

BEGIN

SET @factorial = @factorial \* @number

SET @number = @number - 1

END

PRINT @factorial

GO

-- 4

CREATE FUNCTION GetSumm (@productId INT)

RETURNS INT

BEGIN

DECLARE @Summ INT

SELECT @Summ = count \* price

FROM supplyProducts

RETURN @Summ

END

GO

-- 5

CREATE FUNCTION [dbo].[test\_tabl] ( @id INT)

RETURNS TABLE

AS

RETURN ( SELECT \* FROM products WHERE name = 'Milk')

GO

SELECT \* FROM dbo.test\_tabl (1)

GO

-- 6

CREATE PROCEDURE count\_employees

AS

SELECT COUNT(\*) FROM employees

GO

EXEC count\_employees

GO

CREATE PROCEDURE count\_positions AS

SELECT COUNT(\*) FROM positions

GO

EXEC count\_positions

GO

-- 7

CREATE PROCEDURE count\_products\_by\_price

@price INT

AS

BEGIN

SELECT count(\*)

FROM supplyProducts

WHERE price <= @price

END

GO

EXEC count\_products\_by\_price 80

GO

CREATE PROCEDURE count\_by\_sallary

@sallary INT

AS

SELECT COUNT(\*)

FROM positions

WHERE sum <= @sallary

GO

EXEC count\_by\_sallary 100

GO

-- 8

CREATE PROCEDURE checkname

@param NVARCHAR(MAX)

AS

IF (SELECT sex

FROM employees

WHERE name = @param) = 'male'

RETURN 1

ELSE

RETURN -1

DECLARE @return INT

EXEC @return = checkname 'Artem'

SELECT 'Return Status' = @return

GO

CREATE PROCEDURE checknametour

@param INT

AS

IF (SELECT name

FROM employees

WHERE yearOfBirth = @param) = 'Artem'

RETURN 1

ELSE

RETURN 2

DECLARE @return\_status INT

EXEC @return\_status = checkname 1

SELECT 'Return Status' = @return\_status

GO

-- 9

CREATE PROC update\_proc

AS

UPDATE positions

SET sum = sum + 50

GO

CREATE PROC update\_products\_name\_plus

AS

UPDATE products

SET name = name + '+'

GO

-- 10

CREATE PROCEDURE departments\_select

AS

SELECT \*

FROM departments

GO

CREATE PROCEDURE positions\_select

AS

SELECT \*

FROM positions

GO

-- part 2

-- 1

CREATE FUNCTION Calculator

(@oprd\_1 BIGINT,

@oprd\_2 BIGINT,

@operator NVARCHAR(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

SELECT [dbo].[Calculator](1, 2, '+')

GO

-- 2

CREATE FUNCTION DYNTAB

(@state NVARCHAR(MAX))

RETURNS TABLE

AS

RETURN

SELECT \*

FROM employees

WHERE sex = @state

GO

SELECT \* FROM DYNTAB('male')

GO

-- 3

CREATE FUNCTION parse\_string

(@input\_string nvarchar(500))

RETURNS @tabl TABLE (

Number INT IDENTITY(1,1) NOT NULL,

Substr NVARCHAR(MAX)

)

AS

BEGIN

DECLARE @string NVARCHAR(MAX)

DECLARE @pos INT

WHILE CHARINDEX(' ', @input\_string) > 0

BEGIN

SET @pos = CHARINDEX(' ', @input\_string)

SET @string = SUBSTRING(@input\_string, 1, @pos - 1)

INSERT INTO @tabl

VALUES(@string)

SET @input\_string = SUBSTRING(@input\_string, @pos+1, LEN(@input\_string) - @pos)

END

INSERT INTO @tabl VALUES(@input\_string)

RETURN

END

GO

SELECT \*

FROM dbo.parse\_string('Hello my name is Artem')

GO

-- 8

CREATE PROCEDURE Calculator\_out

(@oprd\_1 BIGINT,

@oprd\_2 BIGINT,

@operator NVARCHAR(1),

@result BIGINT OUTPUT)

AS

BEGIN

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

END

GO

Вывод: в результате выполнения лабораторной работы изучены основные команды и операторы языка SQL. Набор команд включает операторы для создания таблиц и ограничений для них, многотабличных запросов, запросов на группировку и сортировку данных, изменение и использование встроенных функций.