**Databases Laboratory Work Nr 4**

**Title:** Transact-SQL: Procedure Instructions

**Prerequisites:** SQL Server 2019 and SSMS

**Objectives:** Understand how to work with procedure instructions in T-SQL, such as IF/ELSE, WHILE, CASE, CONTINUE, BREAK, etc.

**Tasks:**

1. Write code to print the biggest of the given numbers.
2. Select TOP 10 students which don’t have mark 6 or 8 for the first test for ‘Baze de date’ subject. Use IF…ELSE and variables.
3. Implement first task using CASE.
4. Modify code from task 1 and 2 with use of TRY CATCH.

**Implementation**

1.

--- Task 1

DECLARE @N1 INT, @N2 INT, @N3 INT;

DECLARE @MAI\_MARE INT;

SET @N1 = 60 \* RAND();

SET @N2 = 60 \* RAND();

SET @N3 = 60 \* RAND();

IF @N1 > @N2 AND @N1 > @N3

SET @MAI\_MARE = @N1

ELSE IF @N2 > @N1 AND @N2 > @N3

SET @MAI\_MARE = @N2

ELSE

SET @MAI\_MARE = @N3

PRINT @N1;

PRINT @N2;

PRINT @N3;

PRINT 'Mai mare = ' + CAST(@MAI\_MARE AS VARCHAR(2));

2.

BEGIN

USE universitatea;

DECLARE @DATE TABLE(Nume\_Student NVARCHAR(100), Prenume\_Student NVARCHAR(100), Nota INT);

INSERT INTO @DATE SELECT DISTINCT Nume\_Student, Prenume\_Student, Nota FROM dbo.studenti

INNER JOIN dbo.studenti\_reusita ON studenti\_reusita.Id\_Student = studenti.Id\_Student

INNER JOIN dbo.discipline ON discipline.Id\_Disciplina = studenti\_reusita.Id\_Disciplina

WHERE Disciplina = 'Baze de date' AND Tip\_Evaluare = 'Testul 1'

DECLARE @COUNTER INT = 0;

DECLARE @LEN INT;

DECLARE @CURRENT\_NOTA INT;

DECLARE @CURRENT\_ROW TABLE(Nume\_Student NVARCHAR(100), Prenume\_Student NVARCHAR(100), Nota INT);

DECLARE @RESULT TABLE(Nume\_Student NVARCHAR(100), Prenume\_Student NVARCHAR(100), Nota INT);

SELECT @LEN = COUNT(\*) FROM @DATE;

WHILE @COUNTER < @LEN

BEGIN

INSERT INTO @CURRENT\_ROW SELECT \* FROM @DATE ORDER BY Nume\_Student OFFSET @COUNTER ROWS FETCH NEXT 1 ROW ONLY;

SELECT @CURRENT\_NOTA = Nota FROM @CURRENT\_ROW

IF @CURRENT\_NOTA != 6 AND @CURRENT\_NOTA != 8

INSERT INTO @RESULT SELECT \* FROM @CURRENT\_ROW

SET @COUNTER = @COUNTER + 1;

DELETE @CURRENT\_ROW

END

SELECT TOP(10) Nume\_Student, Prenume\_Student FROM @RESULT ORDER BY Nota DESC

END

3.

--- Task 3

BEGIN

DECLARE @N1 INT, @N2 INT, @N3 INT;

DECLARE @MAI\_MARE INT;

SET @N1 = 60 \* RAND();

SET @N2 = 60 \* RAND();

SET @N3 = 60 \* RAND();

SET @MAI\_MARE = CASE

WHEN @N1 > @N2 AND @N1 > @N3 THEN @N1

WHEN @N2 > @N1 AND @N2 > @N3 THEN @N2

ELSE @N3

END

PRINT @N1;

PRINT @N2;

PRINT @N3;

PRINT 'Mai mare = ' + CAST(@MAI\_MARE AS VARCHAR(2));

END

4.

--- Task 4

BEGIN TRY

DECLARE @N1 INT, @N2 INT, @N3 INT;

DECLARE @MAI\_MARE INT;

SET @N1 = 60 \* RAND();

SET @N2 = 60 \* RAND();

SET @N3 = 60 \* RAND();

IF @N1 = @N2 OR @N1 = @N3 OR @N2 = @N3

RAISERROR ('2 or 3 numbers has the same value', 12, 1)

SET @MAI\_MARE = CASE

WHEN @N1 > @N2 AND @N1 > @N3 THEN @N1

WHEN @N2 > @N1 AND @N2 > @N3 THEN @N2

ELSE @N3

END

PRINT @N1;

PRINT @N2;

PRINT @N3;

PRINT 'Mai mare = ' + CAST(@MAI\_MARE AS VARCHAR(2));

END TRY

BEGIN CATCH

IF @@ERROR != 0

PRINT(ERROR\_MESSAGE())

END CATCH

BEGIN TRY

USE universitatea;

DECLARE @DATE TABLE(Nume\_Student NVARCHAR(100), Prenume\_Student NVARCHAR(100), Nota INT);

INSERT INTO @DATE SELECT DISTINCT Nume\_Student, Prenume\_Student, Nota FROM dbo.studenti

INNER JOIN dbo.studenti\_reusita ON studenti\_reusita.Id\_Student = studenti.Id\_Student

INNER JOIN dbo.discipline ON discipline.Id\_Disciplina = studenti\_reusita.Id\_Disciplina

WHERE Disciplina = 'Baze de date' AND Tip\_Evaluare = 'Testul 1'

DECLARE @COUNTER INT = 0;

DECLARE @LEN INT;

DECLARE @CURRENT\_NOTA INT;

DECLARE @CURRENT\_ROW TABLE(Nume\_Student NVARCHAR(100), Prenume\_Student NVARCHAR(100), Nota INT);

DECLARE @RESULT TABLE(Nume\_Student NVARCHAR(100), Prenume\_Student NVARCHAR(100), Nota INT);

SELECT @LEN = COUNT(\*) FROM @DATE;

WHILE @COUNTER < @LEN

BEGIN

INSERT INTO @CURRENT\_ROW SELECT \* FROM @DATE ORDER BY Nume\_Student OFFSET @COUNTER ROWS FETCH NEXT 1 ROW ONLY;

SELECT @CURRENT\_NOTA = Nota FROM @CURRENT\_ROW

IF @CURRENT\_NOTA != 6 AND @CURRENT\_NOTA != 8

INSERT INTO @RESULT SELECT \* FROM @CURRENT\_ROW

SET @COUNTER = @COUNTER + 1;

DELETE @CURRENT\_ROW

END

SELECT @LEN = COUNT(\*) FROM @RESULT

IF @LEN < 10

RAISERROR ('There are less then 10 rows in result', 12, 1)

SELECT TOP(10) Nume\_Student, Prenume\_Student FROM @RESULT ORDER BY Nota DESC

END TRY

BEGIN CATCH

IF @@ERROR != 0

PRINT(ERROR\_MESSAGE())

END CATCH

**Conclusion**

In this work I gathered a better understanding of what types of blocks exist in T-SQL. I had an opportunity to work with a lot of new syntax for creating variables, settings conditions, repeating actions in a loop, raising and catching exceptions. Now I know how to use these features of T-SQL in order to operate with data in case of more complex tasks.