Ministerul Educației al Republicii Moldova
Universitatea Tehnică a Moldovei
Facultatea Calculatoare, Informatică și Microelectronică
Ingineria Software

REPORT

Laboratory work Nr.5

Transact-SQL: Procedural instructions

Realized by: Popa Eugeniu
FAF-202

Chișinău, 2021

The theoretical part

1. Types of blocks in Transact-SQL.
   Anonymous blocks: these are Transact-SQL blocks, which are defined within an application and have no name.
   Procedures: these are Transact-SQL blocks, which have a name, have input parameters, but no explicit output parameters.
   Functions: these are Transact-SQL blocks, which have a name, have input parameters and always return a value.

2. Presentation of variables.
   A local variable is declared by DECLARE + @
   A global statement with @@

3. Explain the operation of the WHILE repetitive loop.
   The WHILE operator tests a boolean condition and repeatedly executes an instruction or instruction block. Execution of these instructions will be repeated as long as the specified condition remains true.

4. Explain the operation of the CASE expression.
   Parts of SQL statements can be executed conditionally.

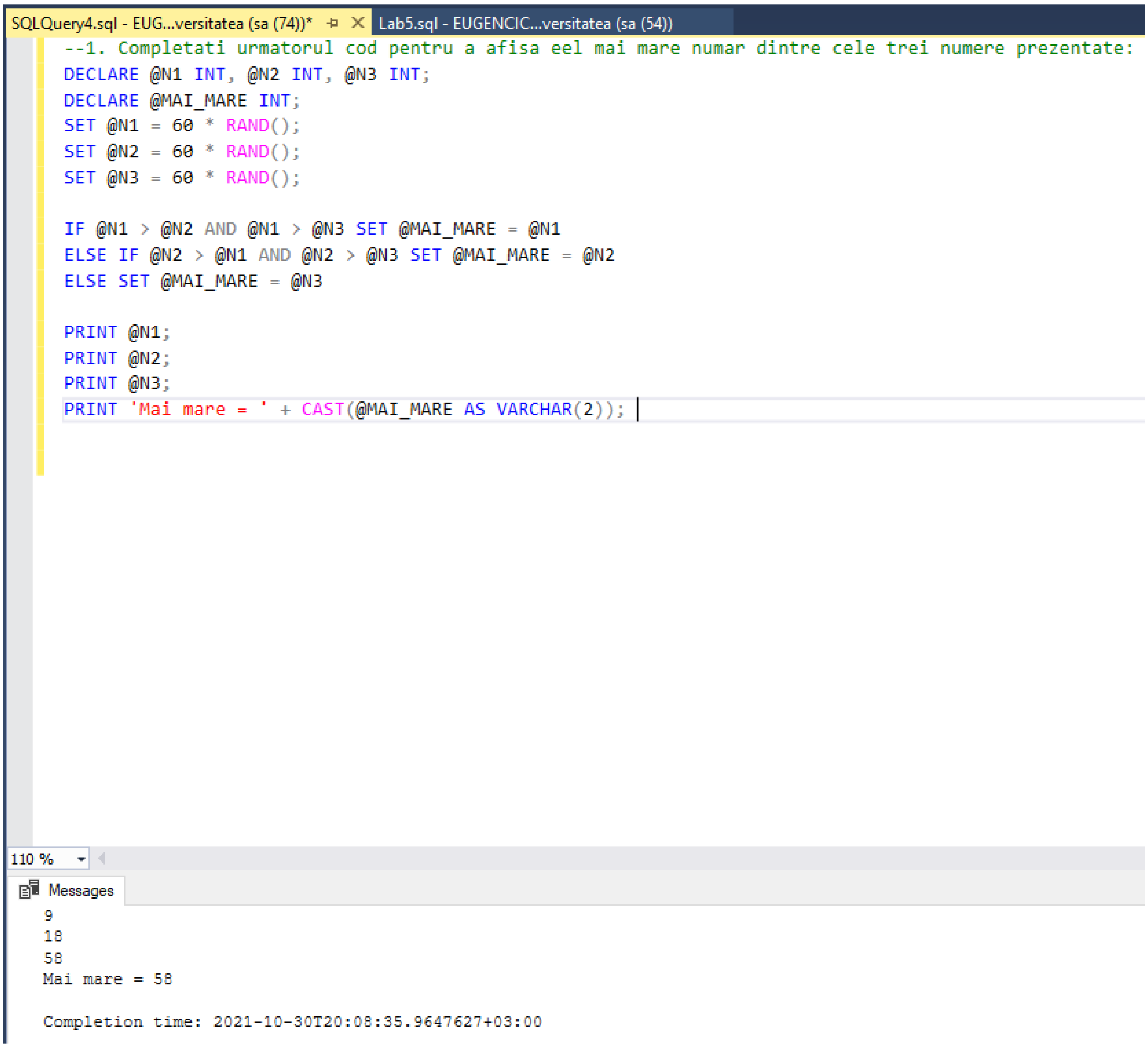
5. Use and actions of the alternative structure IF... ELSE.
   The simplest form of IF statement associates a condition with a sequence of statements enclosed by the keywords THEN and END IF. The sequence of statements is executed only if the condition is TRUE. If the condition is FALSE or NULL, the IF statement does nothing. In either case, control passes to the next statement.
   The second form of IF statement adds the keyword ELSE followed by an alternative sequence of statements. The statements in the ELSE clause are executed only if the condition is FALSE or NULL. The IF-THEN-ELSE statement ensures that one or the other sequence of statements is executed. The first UPDATE statement is executed when the condition is TRUE, and the second UPDATE statement is executed when the condition is FALSE or NULL.

6. What is an exception? Capturing and handling exceptions.
   An exception is a PL/SQL error that is raised during program execution, either implicitly by TimesTen or explicitly by your program. Handle an exception by trapping it with a handler or propagating it to the calling environment.
   Trap a predefined TimesTen error by referencing its predefined name in your exception-handling routine. PL/SQL declares predefined exceptions in the STANDARD package.

You can define your own exceptions in PL/SQL in TimesTen, and you can raise user-defined exceptions explicitly with either the PL/SQL RAISE statement or the RAISE_APPLICATION_ERROR procedure.
The RAISE statement stops normal execution of a PL/SQL block or subprogram and transfers control to an exception handler. RAISE statements can raise predefined exceptions, or user-defined exceptions whose names you decide.

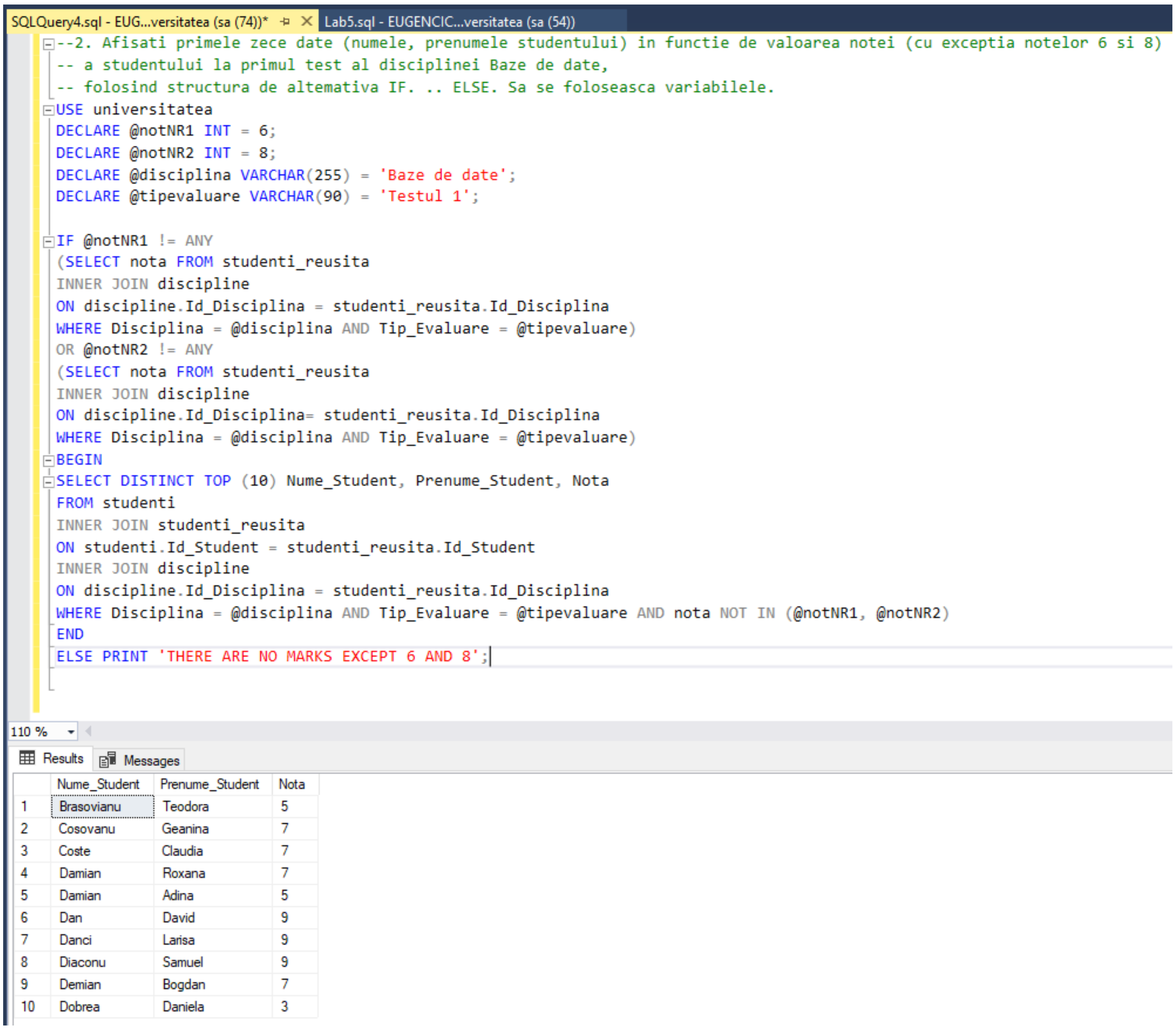
7. RAISERROR instruction function.
RAISERROR is a SQL Server error handling statement that generates an error message and initiates error processing. RAISERROR can either reference a user-defined message that is stored in the sys.messages catalog view or it can build a message dynamically. The message is returned as a server error message to the calling application or to an associated CATCH block of a TRY...CATCH construct.

The practical part


SQLQuery4.sql - EUG...versitatea (sa (74))*
Lab5.sql - EUGENCIC...versitatea (sa (54))
110 %
Messages

```
--1. Completati urmatorul cod pentru a afisa eel mai mare numar dintre cele trei numere prezentate:
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));
```

```
9
18
58
Mai mare = 58

Completion time: 2021-10-30T20:08:35.9647627+03:00
```

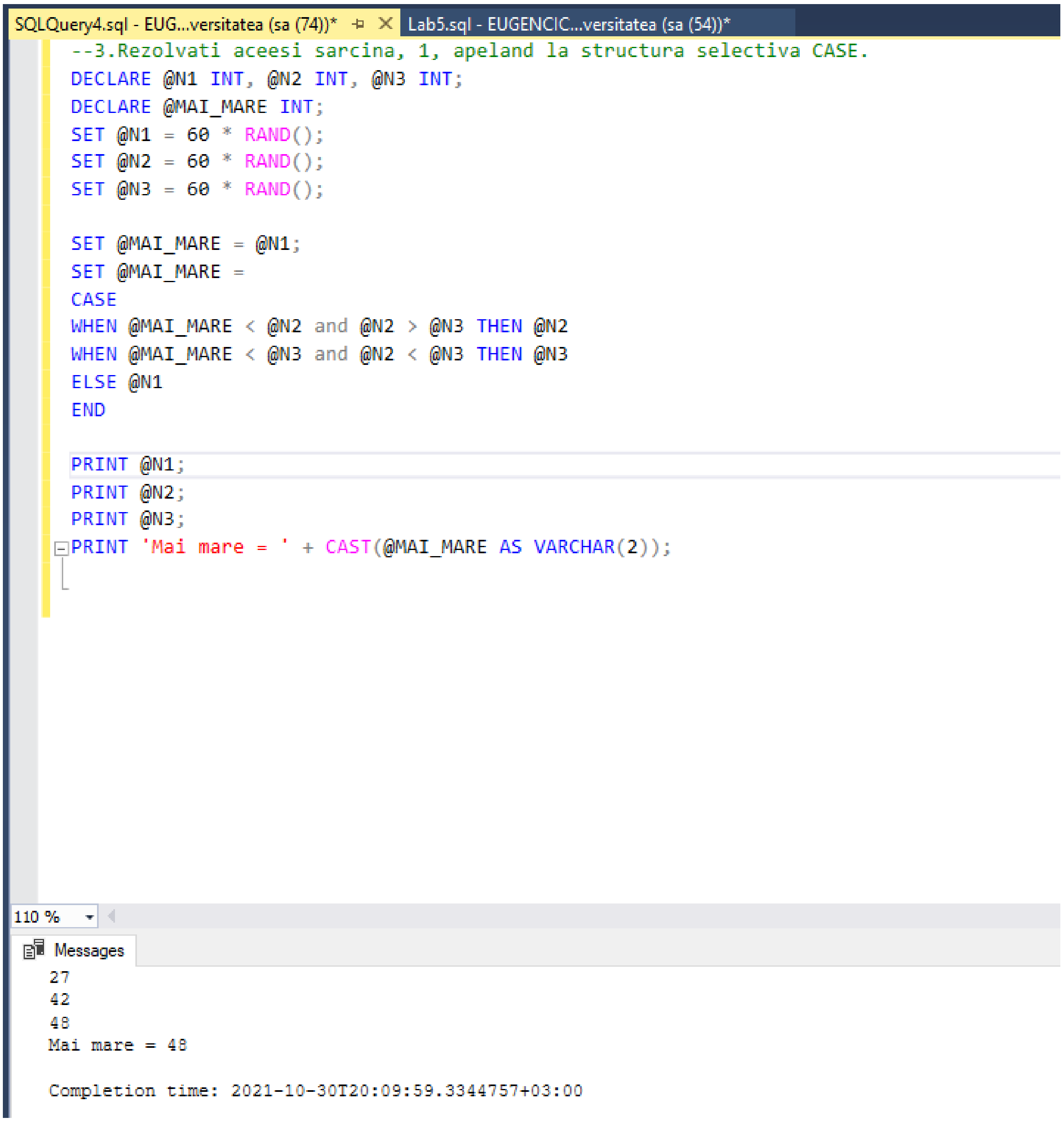

SQLQuery4.sql - EUG...versitatea (sa (74))* Lab5.sql - EUGENCIC...versitatea (sa (54))

```
--2. Afisati primele zece date (numele, prenumele studentului) in functie de valoarea notei (cu exceptia notelor 6 si 8)
-- a studentului la primul test al disciplinei Baze de date,
-- folosind structura de altemativa IF. .. ELSE. Sa se foloseasca variabilele.
USE universitatea
DECLARE @notNR1 INT = 6;
DECLARE @notNR2 INT = 8;
DECLARE @disciplina VARCHAR(255) = 'Baze de date';
DECLARE @tipevaluare VARCHAR(90) = 'Testul 1';

IF @notNR1 != ANY
(SELECT nota FROM studenti_reusita
INNER JOIN discipline
ON discipline.Id_Disciplina = studenti_reusita.Id_Disciplina
WHERE Disciplina = @disciplina AND Tip_Evaluare = @tipevaluare)
OR @notNR2 != ANY
(SELECT nota FROM studenti_reusita
INNER JOIN discipline
ON discipline.Id_Disciplina= studenti_reusita.Id_Disciplina
WHERE Disciplina = @disciplina AND Tip_Evaluare = @tipevaluare)
BEGIN
SELECT DISTINCT TOP (10) Nume_Student, Prenume_Student, Nota
FROM studenti
INNER JOIN studenti_reusita
ON studenti.Id_Student = studenti_reusita.Id_Student
INNER JOIN discipline
ON discipline.Id_Disciplina = studenti_reusita.Id_Disciplina
WHERE Disciplina = @disciplina AND Tip_Evaluare = @tipevaluare AND nota NOT IN (@notNR1, @notNR2)
END
ELSE PRINT 'THERE ARE NO MARKS EXCEPT 6 AND 8';
```

110 %

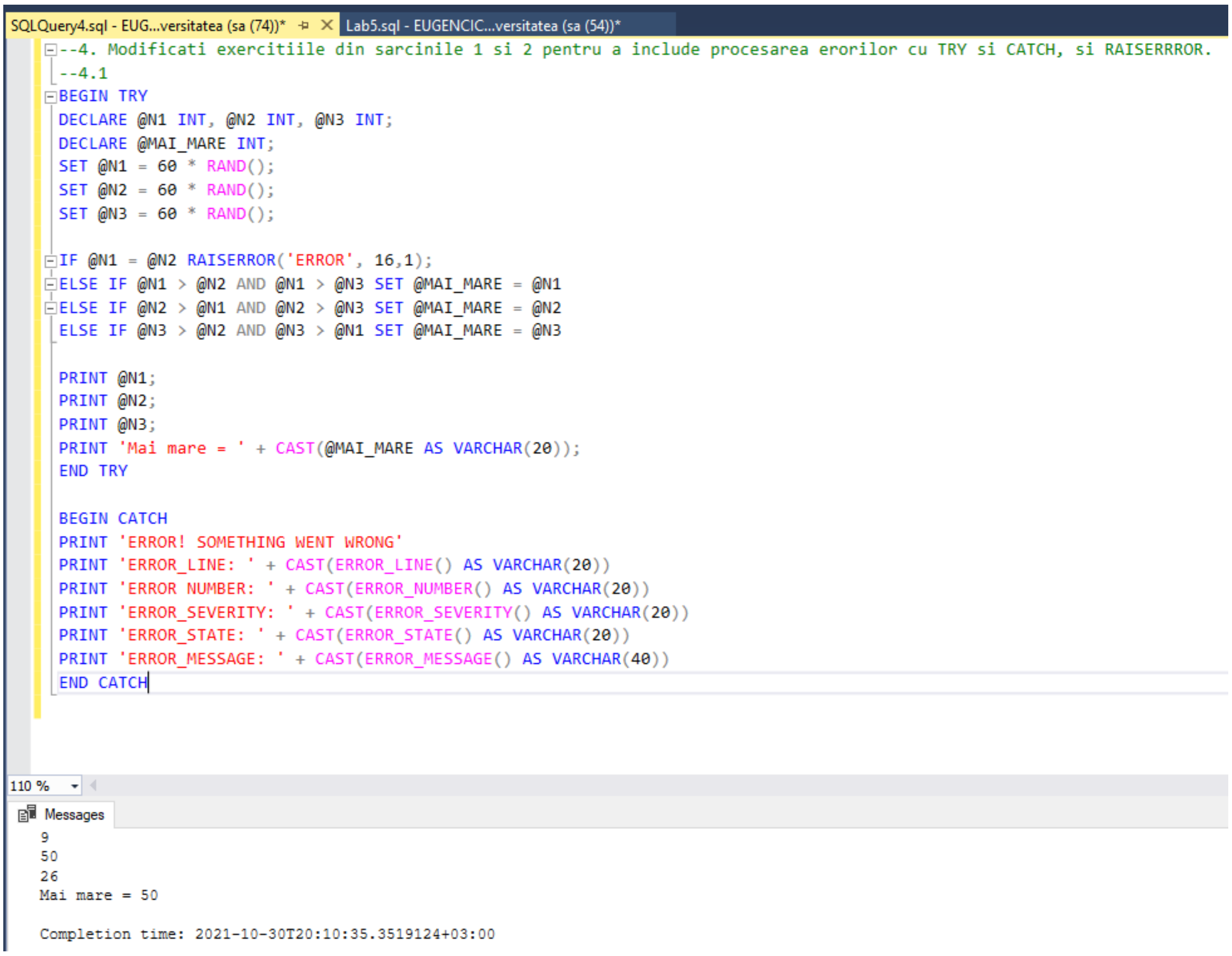
Results | Messages

	Nume_Student	Prenume_Student	Nota
1	Brasovianu	Teodora	5
2	Cosovanu	Geanina	7
3	Coste	Claudia	7
4	Damian	Roxana	7
5	Damian	Adina	5
6	Dan	David	9
7	Danci	Larisa	9
8	Diaconu	Samuel	9
9	Demian	Bogdan	7
10	Dobrea	Daniela	3


SQLQuery4.sql - EUG...versitatea (sa (74))*
Lab5.sql - EUGENCIC...versitatea (sa (54))*
110 %
Messages

```sql
--3.Rezolvati aceesi sarcina, 1, apeland la structura selectiva CASE.
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 = @N1;
SET @MAI_MARE =
CASE
WHEN @MAI_MARE < @N2 and @N2 > @N3 THEN @N2
WHEN @MAI_MARE < @N3 and @N2 < @N3 THEN @N3
ELSE @N1
END

PRINT @N1;
PRINT @N2;
PRINT @N3;
PRINT 'Mai mare = ' + CAST(@MAI_MARE AS VARCHAR(2));
```

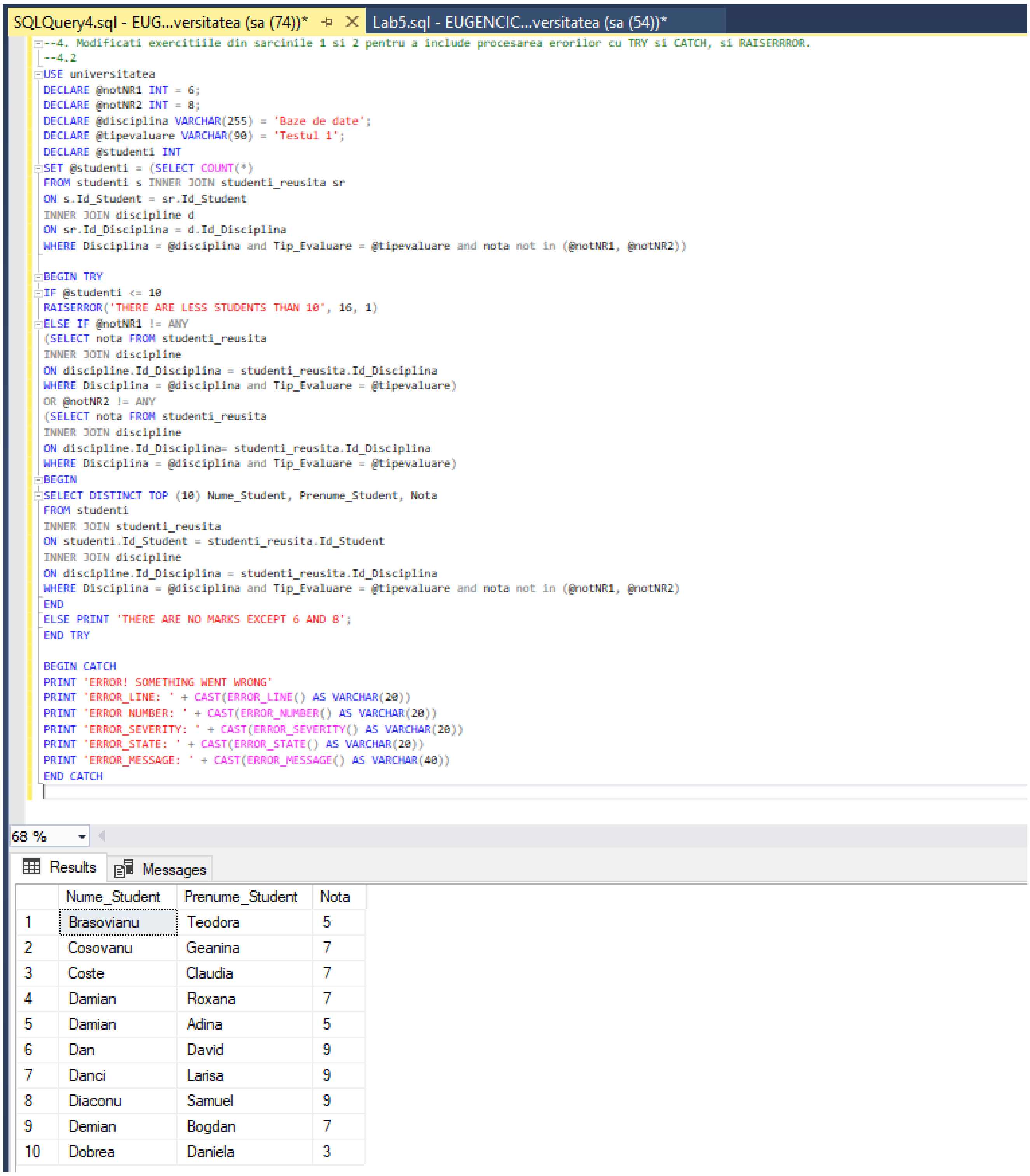
```
27
42
48
Mai mare = 48

Completion time: 2021-10-30T20:09:59.3344757+03:00
```



SQLQuery4.sql - EUG...versitatea (sa (74))*
Lab5.sql - EUGENCIC...versitatea (sa (54))*
110 %
Messages

```sql
--4. Modificati exercitiile din sarcinile 1 si 2 pentru a include procesarea erorilor cu TRY si CATCH, si RAISERRROR.
--4.1
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 RAISERROR('ERROR', 16,1);
ELSE IF @N1 > @N2 AND @N1 > @N3 SET @MAI_MARE = @N1
ELSE IF @N2 > @N1 AND @N2 > @N3 SET @MAI_MARE = @N2
ELSE IF @N3 > @N2 AND @N3 > @N1 SET @MAI_MARE = @N3

PRINT @N1;
PRINT @N2;
PRINT @N3;
PRINT 'Mai mare = ' + CAST(@MAI_MARE AS VARCHAR(20));
END TRY

BEGIN CATCH
PRINT 'ERROR! SOMETHING WENT WRONG'
PRINT 'ERROR_LINE: ' + CAST(ERROR_LINE() AS VARCHAR(20))
PRINT 'ERROR NUMBER: ' + CAST(ERROR_NUMBER() AS VARCHAR(20))
PRINT 'ERROR_SEVERITY: ' + CAST(ERROR_SEVERITY() AS VARCHAR(20))
PRINT 'ERROR_STATE: ' + CAST(ERROR_STATE() AS VARCHAR(20))
PRINT 'ERROR_MESSAGE: ' + CAST(ERROR_MESSAGE() AS VARCHAR(40))
END CATCH
```

```
9
50
26
Mai mare = 50

Completion time: 2021-10-30T20:10:35.3519124+03:00
```



SQLQuery4.sql - EUG...versitatea (sa (74))* Lab5.sql - EUGENCIC...versitatea (sa (54))*

```sql
--4. Modificati exercitiile din sarcinile 1 si 2 pentru a include procesarea erorilor cu TRY si CATCH, si RAISERRROR.
--4.2
USE universitatea
DECLARE @notNR1 INT = 6;
DECLARE @notNR2 INT = 8;
DECLARE @disciplina VARCHAR(255) = 'Baze de date';
DECLARE @tipevaluare VARCHAR(90) = 'Testul 1';
DECLARE @studenti INT
SET @studenti = (SELECT COUNT(*)
FROM studenti s INNER JOIN studenti_reusita sr
ON s.Id_Student = sr.Id_Student
INNER JOIN discipline d
ON sr.Id_Disciplina = d.Id_Disciplina
WHERE Disciplina = @disciplina and Tip_Evaluare = @tipevaluare and nota not in (@notNR1, @notNR2))

BEGIN TRY
IF @studenti <= 10
RAISERROR('THERE ARE LESS STUDENTS THAN 10', 16, 1)
ELSE IF @notNR1 != ANY
(SELECT nota FROM studenti_reusita
INNER JOIN discipline
ON discipline.Id_Disciplina = studenti_reusita.Id_Disciplina
WHERE Disciplina = @disciplina and Tip_Evaluare = @tipevaluare)
OR @notNR2 != ANY
(SELECT nota FROM studenti_reusita
INNER JOIN discipline
ON discipline.Id_Disciplina= studenti_reusita.Id_Disciplina
WHERE Disciplina = @disciplina and Tip_Evaluare = @tipevaluare)
BEGIN
SELECT DISTINCT TOP (10) Nume_Student, Prenume_Student, Nota
FROM studenti
INNER JOIN studenti_reusita
ON studenti.Id_Student = studenti_reusita.Id_Student
INNER JOIN discipline
ON discipline.Id_Disciplina = studenti_reusita.Id_Disciplina
WHERE Disciplina = @disciplina and Tip_Evaluare = @tipevaluare and nota not in (@notNR1, @notNR2)
END
ELSE PRINT 'THERE ARE NO MARKS EXCEPT 6 AND 8';
END TRY

BEGIN CATCH
PRINT 'ERROR! SOMETHING WENT WRONG'
PRINT 'ERROR_LINE: ' + CAST(ERROR_LINE() AS VARCHAR(20))
PRINT 'ERROR NUMBER: ' + CAST(ERROR_NUMBER() AS VARCHAR(20))
PRINT 'ERROR_SEVERITY: ' + CAST(ERROR_SEVERITY() AS VARCHAR(20))
PRINT 'ERROR_STATE: ' + CAST(ERROR_STATE() AS VARCHAR(20))
PRINT 'ERROR_MESSAGE: ' + CAST(ERROR_MESSAGE() AS VARCHAR(40))
END CATCH
```

68 %

Results Messages

	Nume_Student	Prenume_Student	Nota
1	Brasovianu	Teodora	5
2	Cosovanu	Geanina	7
3	Coste	Claudia	7
4	Damian	Roxana	7
5	Damian	Adina	5
6	Dan	David	9
7	Danci	Larisa	9
8	Diaconu	Samuel	9
9	Demian	Bogdan	7
10	Dobrea	Daniela	3