**MINISTERUL EDUCAŢIEI ȘI CERCETĂRII AL REPUBLICII MOLDOVA**

**Universitatea Tehnică a Moldovei**

**Facultatea Calculatoare, Informatică şi Microelectronică**

**Departamentul Ingineria Software și Automatică**

**Programul de studii: Tehnologia informației**

RAPORT

# LUCRARE DE LABORATOR NR. 2

# la Baze de Date

A efectuat:

st. gr. TI-211 Popa Cătălin

A verificat: Olga Grosu

UTM, Chișinău 2023

**Transact SQL: INSTRUCȚIUNI PROCEDURALE**

În Transact-SQL, instrucțiunile procedurale se referă la blocurile de cod care sunt executate secvențial pentru a realiza anumite operațiuni. Acestea sunt utilizate în principal în cadrul procedurilor stocate, funcțiilor definite de utilizator și declarații de blocuri BEGIN...END în T-SQL.

Mai întâi am realizat 4 interogări care folosesc variabile globale ale serverului. Funcțiile @@ sunt funcții de sistem. Mai jos sunt câteva exemple și rezultatul este în figura 1.

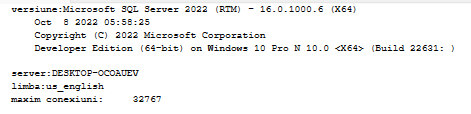


Figura 1 – Informație despre sistem

print 'versiune:' + @@version

print 'server:' + @@servername

print 'limba:' + @@language

print 'maxim conexiuni:' + str(@@max\_connections)

Respectiv avem afișat în final versiunea sistemului, denumirea serverului, limba setată și maxim conectări posibile.

**Case**

--Cu ajutorul expresiei CASE evaluam suma totala achitata din tabelul bon

SELECT BonID,CumparatorID,

case when SumaTotala > 100

then 'Suma mare'

else 'Suma mica'

end as cantitate

from SistemGestionalBon.dbo.Bon

order by BonID

În rezultat vom avea afișat dacă suma este mare sau mică, în dependență de filtru dacă este mai mare ca 100 sau mai mic.Rezultatul este prezentat în figura 2.

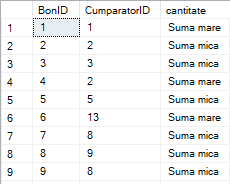


Figura 2 – Afișare caracteristică suma totală

**IF, ELSE și WHILE**

---Mai jos am realizat o procedură care foloseste instrcutiunile IF,ELSE si WHILE

--am declarat caateva variabile iar apoi am folosite datele din table Furnizori

--pentru a afisa adresele care contin cuvantul strada in adresa sa.

select \* from SistemGestionalBon.dbo.Furnizori

DECLARE @i INT = 1;

DECLARE @rowCount INT;

DECLARE @id INT;

DECLARE @adresa NVARCHAR(MAX);

SELECT @rowCount = COUNT(\*) FROM SistemGestionalBon.dbo.Furnizori;

WHILE @i <= @rowCount

BEGIN

SELECT @id = FurnizorID, @adresa = Adresa FROM SistemGestionalBon.dbo.Furnizori WHERE FurnizorID = @i;

IF @adresa LIKE '%Strada%'

BEGIN

PRINT 'Adresa cu id-ul ' + CAST(@id AS NVARCHAR(MAX)) + ' este: ' + @adresa;

END

SET @i = @i + 1;

END

În rezultat vom avea afișat adresele care conțin cuvântul stradă în adresa sa și textul corespunzător. Putem observa rezultatul în figura 3.



Figura 3 – Afișare adrese

IF, ELSE și WHILE

---In aceasta procedură am folost la fel IF, ELSE si While

--In final afisez numele care incep cu litear 'A' din tabela Cumparatori

select \* from SistemGestionalBon.dbo.Cumparatori;

DECLARE @l INT = 1;

DECLARE @rowCounts INT;

DECLARE @id1 INT;

DECLARE @numee NVARCHAR(MAX);

SELECT @rowCounts = COUNT(\*) FROM SistemGestionalBon.dbo.Cumparatori;

WHILE @l <= @rowCounts

BEGIN

SELECT @id1 = CumparatorID, @numee = Nume FROM SistemGestionalBon.dbo.Cumparatori WHERE CumparatorID = @l;

IF LEFT(@numee, 1) = 'A'

BEGIN

PRINT 'Numele cu id-ul ' + CAST(@id1 AS NVARCHAR(MAX)) + ' este: ' + @numee;

END

SET @l= @l + 1;

END

În această procedură am realizat o verificare dacă numele începe cu caracterul A și apoi afișez id-ul lui și numel. Rezultatul este în figura 4.



Figura 4 – Afișare nume și id

**CONTINUE**

--In procedura de mai jost folosesc instructiunile BREAK,CONTINUE si RETURN

-- In rezultat voi obtine afisarea dunimirii sefului care are functia de manager

--select \* from SistemGestionalBon.dbo.Functii;

DECLARE @num INT = 1;

DECLARE @rowCount2 INT;

DECLARE @FunctieID INT;

DECLARE @Denumire NVARCHAR(MAX);

DECLARE @SefFunctie NVARCHAR(MAX);

SELECT @rowCount2 = COUNT(\*) FROM SistemGestionalBon.dbo.Functii;

WHILE @num <= @rowCount2

BEGIN

SELECT @FunctieID = FunctieID, @Denumire = Denumire, @SefFunctie = SefFunctie FROM SistemGestionalBon.dbo.Functii WHERE FunctieID = @num;

IF @Denumire LIKE '%Manager%'

BEGIN

PRINT 'Functia cu FunctieID ' + CAST(@FunctieID AS NVARCHAR(MAX)) + ' are seful cu numele: ' + @SefFunctie;

END

ELSE

BEGIN

SET @num = @num + 1;

CONTINUE;

END

SET @num = @num + 1;

END

În această procedură am realizat afișsarea denumirii șefului care are funcția de manager. Rezultatul este în figura 5.



Figura 5 – Afișare denumire șef

**GOTO**

--In continuarea am folosit GOTO

--Aici se va executa loop pana cand @i2 va deveni mai mare decat numarul total de randuri si va trece deodata la LoopEnd

DECLARE @i2 INT = 1;

DECLARE @rowCount3 INT;

DECLARE @FunctieID2 INT;

DECLARE @Denumire2 NVARCHAR(MAX);

DECLARE @SefFunctie2 NVARCHAR(MAX);

SELECT @rowCount3 = COUNT(\*) FROM SistemGestionalBon.dbo.Functii;

LoopStart:

IF @i2 > @rowCount3

GOTO LoopEnd;

SELECT @FunctieID2 = FunctieID, @Denumire2 = Denumire, @SefFunctie2 = SefFunctie FROM SistemGestionalBon.dbo.Functii WHERE FunctieID = @i2;

IF @Denumire2 LIKE '%Manager%'

BEGIN

PRINT 'Functia cu FunctieID ' + CAST(@FunctieID2 AS NVARCHAR(MAX)) + ' are seful cu numele: ' + @SefFunctie2;

END

SET @i2 = @i2 + 1;

GOTO LoopStart;

LoopEnd:

Procedura dată va executa o buclă până când variabila @i2 va deveni mai mare decât numărul total de rânduri și va trece deodată spre final dacă valoare deja depășeste numărul de rânduri. Rezultatul este afișat în figura 6.



Figura 6 – Afișare cine are funcția cu id 1

**WAITFOR**

--Aceasta interogare va astepta 30 secunde iar apoi va afisa toate randurile din tabela Valabilitate

WAITFOR DELAY '00:00:30';

SELECT ProdusID, DataProducere, DataExpirare

FROM SistemGestionalBon.dbo.Valabilitate;

Interogarea dată va aștepta 30 de secunde iar apoi va afișa toate rândurile din tabela Valabilitate. Rezultatul este în figura 7.

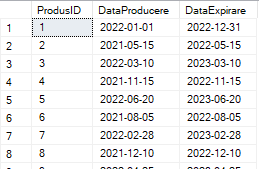


Figura 7 – WAITFOR

**COALESCE**

--In aceasta interogare afisez produsid si dataepirare, iar daca data este nula atunci se afiseaza Data nedisponibila

SELECT

ProdusID,

COALESCE(DataExpirare,DataProducere, 'Data nedisponibilă') AS DataExpirare

FROM

SistemGestionalBon.dbo.Valabilitate;

Interogarea dată afișează ProdusID și prima valoare nenulă dintre DataExpirare și DataProducere. Rezultatul este în figura 8.

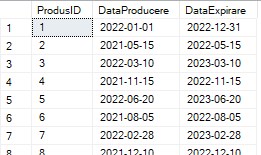


Figura 8 – Coalesce

**NULLIF**

--Daca valoare AngajatID este 0, atunci se va afisa null, altfel se va afișa AngajatID

SELECT

BonID,

NULLIF(AngajatID, 1) AS Disponibilitate\_angajat

FROM

SistemGestionalBon.dbo.Bon;

Interogarea dată va afișa null, dacă AngajatID va fi 0.

**TRY CATCH**

--Mai jos este o interogare care va returna mesajul de eraore deoarece imaprtirea la 0 nu are sens

--pentru acceasta am folost try catch

BEGIN TRY

DECLARE @num1 int,

@num2 int,

@res int

SET @num1 = 200

SET @num2 = 0

SET @res = @num1/@num2

END TRY

BEGIN CATCH

PRINT 'A aparut eroare'

END CATCH

În procedura dată are loc reprezentarea unei erori, atunci când programul nu a executat codul. Respectiv împărțirea la 0 nu are ses și se va executa catch care returneaza un print cu un text informativ despre eroare. Observăm figura 9.



Figura 9 – TRY CATCH.

**Concluzie**

Trasact SQL ne ajută la creare blocurilor folisind diverse instrucțiuni precum IF,ELSE,WHILE, GOTO și altele. Datorită acestor instrucțiuni, putea crea proceduri mai complexe și ușor de manipulat cu datele necesare. Ele ușurează lucrul cu baza de date și face datele mult mai accesbilie.