# DML Trigger

# (Лаборатори №4)

У. Төрболд

ХШУИС, Мэдээллийн технологийн IV-р түвшний оюутан, turbold1125@gmail.com

# ОРШИЛ

Энэхүү лабораторын ажил нь SQL хэлийн чухал шинж чанар болох триггерийг судлах болно. Энэ функц нь өгөгдлийн сан дахь тодорхой үйл явдлууд дээр үндэслэн үйлдлүүдийг автоматжуулах боломжийг олгодог ба үүнийг ашиглан практик дээр туршина.

# 2. ЗОРИЛГО

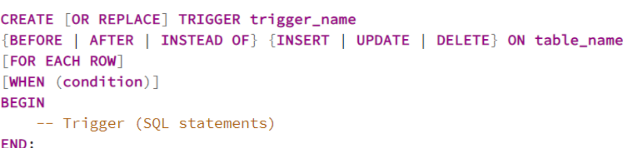
Энэхүү лабораторын зорилго нь өгөгдлийн сан дээрх хязгаарлалт буюу constraint, триггер буюу автоматжуулалтыг хэрэгжүүлэх юм.

# 3. ОНОЛЫН СУДАЛГАА

## 3.1 Trigger

SQL – д триггер нь датабааз дахь тодорхой хүснхэгт дээрх үйл явдлыг автоматаар ажилладаг хадгалагдсан процедур юм.

Syntax:



CREATE TRIGGER нь шинэ триггер үүсгэх эсвэл байгаа нэгийг солиход ашиглана.

BEFORE, AFTER нь хэзээ асах ёстойх заана.

INSERT, UPDATE, DELETE нь триггерийн үйлдлийн төрлийг заана.

WHEN нь нөхцөл

# 4. ХЭРЭГЖҮҮЛЭЛТ

1. A Хүснэгтэд өгөгдөл оруулахад Б хүснэгтийн тодорхой нэг багана дахь өгөдлийн утга нь X утгаас их байвал Insert, Update хийж болдог, X утгаас бага байвал Insert, Update хийлгэдэггүй байх trigger бич.

A болон Б хүснэгт

CREATE TABLE TableA (

ID INT PRIMARY KEY IDENTITY,

DataValue VARCHAR(50)

);

CREATE TABLE TableB (

ID INT PRIMARY KEY IDENTITY,

ValueColumn INT

);

Триггер

-- TableA дээр INSERT болон UPDATE үйлдэлд зориулсан триггер үүсгэх

CREATE TRIGGER CheckValueTrigger

ON TableA

FOR INSERT, UPDATE

AS

BEGIN

-- X утгыг хадгалахын тулд локал хувьсагчийг зарлана

DECLARE @Threshold INT;

-- X босго утгыг тохируулах (үүнийг өөрийн хүссэн босго болгон өөрчлөх)

SET @Threshold = 100;

-- B Хүснэгтийн ValueColumn дахь утга босго хэмжээнээс бага эсвэл тэнцүү эсэхийг шалгах

IF EXISTS (

SELECT 1

FROM TableB

WHERE ValueColumn <= @Threshold

)

BEGIN

-- ХүснэгтВ дэх ямар нэг утга босго хэмжээнээс бага эсвэл тэнцүү байвал алдаа гаргаж, алдааг буцаах.

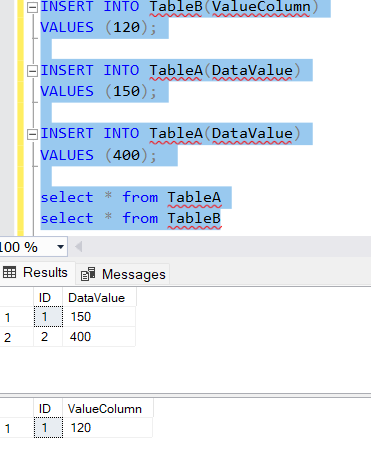
RAISERROR('Insert and Update operations are not allowed because at least one value in TableB is less than or equal to X.', 16, 1);

ROLLBACK TRANSACTION;

END

END;

Үр дүн:

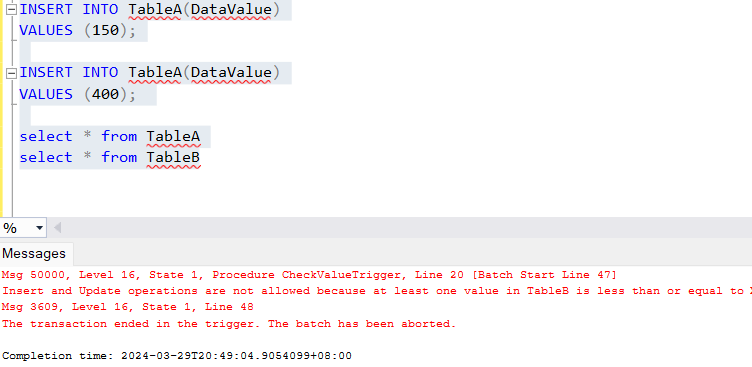


-- ХүснэгтВ дэх утгыг X утгаас бага буюу 80 болгоод insert хийх

UPDATE TableB

SET ValueColumn = 80

WHERE ID = 1;



Энэхүү тохиолдолд X утга нь 100 байгаа тул ValueColumn дахь утгыг 80 болгож өөрчлөхөд 100 > 80 тул А хүснэгтэнд INSERT, UPDATE үйлдэл зөвшөөрөгдөхгүй болох нь батлагдлаа.

1. X хэмжээний хугацаанд Y хүүтэйгээр бүтээж байна. Мэдээллийг нь А хүснэгтэд хадгална. Хадгалуулсан мөнгөөрөө хүүг авах үед нь B хүснэгтэд бичлэг хийгдэх бөгөөд хүү бодох огноо нь хүү бодох тоогоор олж авсан хүснэгтээс хүүг нь олж хоногоор үржүүлж B-д бичнэ. Ж: А=2005.01.01 5'000'000 0.1 B=2005.02.01 150'000 B=2005.05.01 450'000.

А болон Б хүснэгт

CREATE TABLE TableA (

DepositDate DATE,

Amount MONEY,

InterestRate DECIMAL(5, 4)

);

CREATE TABLE TableB (

InterestDate DATE,

InterestAmount MONEY

);

Триггер

-- TableA дээр INSERT үйлдэлд зориулсан триггер үүсгэх

CREATE TRIGGER CalculateInterestTrigger

ON TableA

AFTER INSERT

AS

BEGIN

-- Хадгаламж, хүү, хоног, хүүгийн мэдээллийг хадгалах хувьсагч зарлах

DECLARE @DepositDate DATE; -- огноог зарлах

DECLARE @Amount MONEY; -- мөнгөн дүнг зарлах

DECLARE @InterestRate DECIMAL(5, 4); -- хүүгийн хувь зарлах

DECLARE @InterestDays INT; -- хадгалах өдрийг зарлах

DECLARE @InterestAmount MONEY; -- тооцоолсон дүнг хадгалах

-- ТableА-д оруулсан мэдээллийг хувьсагчдад хадгалах

SELECT @DepositDate = DepositDate, @Amount = Amount, @InterestRate = InterestRate \* 0.01

FROM inserted;

-- Хадгаламж хийснээс хойших өдрийг тооцоолох

SET @InterestDays = DATEDIFF(DAY, @DepositDate, GETDATE()) - 1;

-- Хүүгийн хэмжээг дараах томъёогоор тооцоолох

SET @InterestAmount = @Amount \* @InterestRate \* @InterestDays;

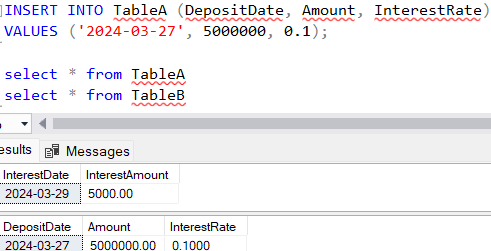
-- Тооцоолсон хүүг TableB-д оруулах

INSERT INTO TableB (InterestDate, InterestAmount)

VALUES (GETDATE(), @InterestAmount);

END;

Үр дүн:



1. Оны шилдэг захиалагчийг өөрйин сонгон шалгаруулалтаар олох stored procedure бич.

Хүснэгт болон анхны утга оруулах

CREATE TABLE student (

ID INT,

Name1 VARCHAR(50),

Age INT

);

INSERT INTO student (ID, Name1, Age)

VALUES

(1, 'Dorj', 20),

(2, 'Bat', 21),

(4, 'Dulmaa', 20);

Триггер

CREATE TRIGGER InsertMissingID

ON student

INSTEAD OF INSERT

AS

BEGIN

-- Хувьсагч зарлах

DECLARE @NextID INT;

-- Хоосон ID хайх

SELECT @NextID = COALESCE(MIN(s.ID + 1), 1)

FROM student s

LEFT JOIN inserted i ON s.ID = i.ID

WHERE i.ID IS NULL;

-- ID-г олоод зөв байрлалд нь оруулах

INSERT INTO student (ID, Name1, Age)

SELECT

COALESCE(i.ID, @NextID + ROW\_NUMBER() OVER (ORDER BY (SELECT NULL))) AS ID,

i.Name1,

i.Age

FROM (

SELECT

ID,

Name1,

Age,

ROW\_NUMBER() OVER (ORDER BY (SELECT NULL)) AS RowNum

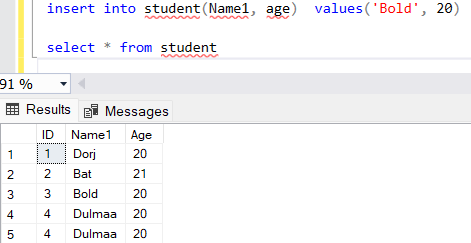
FROM inserted

) i

ORDER BY i.RowNum ASC;

END;

Үр дүн:



# 5. ДҮГНЭЛТ

Энэхүү лабораторын ажлаар SQL триггерүүдийн практик хэрэгжилтийг судалсан болно. Өгөгдлийн сан дахь тодорхой үйл явдлуудад тулгуурлан үйлдлүүдийг автоматжуулахын тулд триггерийг ашигласан.