Banco de Dados Ativos e Triggers

Herysson R. Figueiredo
herysson.figueiredo@ufn.edu.br

Conceitos de banco de dados ativos e triggers

São as regras que especificam ações que são disparadas automaticamente por certos eventos.

Conceito de trigger: uma técnica para especificar certos tipos de regras ativas

Modelo generalizado para bancos de dados ativos e triggers

O modelo usado para especificar regras de banco de dados ativo é conhecido como modelo **Evento-Condição-Ação (ECA).**

Modelo generalizado para bancos de dados ativos e triggers

O(s) **evento**(s) que dispara(m) a regra: esses eventos normalmente são operações de atualização do banco de dados que são aplicadas explicitamente ao banco de dados. No entanto, no modelo geral, eles também poderiam ser eventos temporais ou outros tipos de eventos externos.

Modelo generalizado para bancos de dados ativos e triggers

A **condição** que determina se a ação da regra deve ser executada: quando o evento que dispara a ação tiver ocorrido, uma condição **opcional** pode ser avaliada. Se **nenhuma** condição for especificada, a ação será executada quando ocorrer o evento. Se uma condição for especificada, ela é primeiro avaliada e, somente se for **avaliada** como verdadeira, a ação da regra será executada.

Triggers

Um **Trigger (Gatilho)** é um tipo especial de Stored Procedure que é executada automaticamente quando um usuário realiza uma operação de modificação de dados em uma tabela especificada.

Triggers

As operações que podem disparar um trigger são:

- INSERT
- UPDATE
- DELETE

Triggers DML

Os triggers não são executados diretamente, eles disparam apenas em resposta a eventos como INSERT, UPDATE ou DELETE em uma tabela.

No SQL Server, os triggers disparam uma vez para cada operação de modificaçõa - enão uma vez por linha afetada (no Oracle há as duas opções).

DML (Linguagem de Manipulação de Dados) – É um conjunto de instruções usada nas consultas e modificações dos dados armazenados nas tabelas do banco de dados.

Modos de disparo de um Trigger

Um Trigger no SQL Server pode ser disparado de dois modos diferentes:

- **After** - O código presente no trigger é executado após todas as ações terem sido completadas na tabela especificada;
- **Instead Of -** O código presente no trigger é executado no lugar da operação que causou seu disparo.

A tabela a seguir compara os dois tipos de Triggers

DML (Linguagem de Manipulação de Dados) – É um conjunto de instruções usada nas consultas e modificações dos dados armazenados nas tabelas do banco de dados.

Característica	INSTEAD OF	AFTER
Declaração DML	Simulada, mas não executada	Executada, mas pode ser revertida no trigger (Roll back)
Timing	Antes das constraints PK e FK	Após a transação completa, mas antes do commit
Nº de eventos por tabela	Um	Múltiplos
Aplicável em Views?	Sim	Não
Permite Aninhamento?	Depende das opções do servidor	Depende das opções do servidor
É Recursivo?	Não	Depende das opções do Banco de Dados.

Fluxo de Transações

Para desenvolver Triggers, é necessário conhecimento do fluxo da transação, para evitar conflitos entres triggers e constraints.

As transações se movem através de verificações e códigos na ordem mostrada a seguir:

DML (Linguagem de Manipulação de Dados) – É um conjunto de instruções usada nas consultas e modificações dos dados armazenados nas tabelas do banco de dados.

Fluxo de Transações

1. Verificação de IDENTITY INSERT
2. Restrição (Constraint) de Nulos (NULL)
3. Checagem de tipos de dados
4. Execução de trigger INSTEAD OF (a execução do DML pára aqui; esse trigger não é recursivo)
5. Restrição de Chave Primária
6. Restrição “Check”
7. Restrição Chave Estrangeira
8. Execução do DML e atualização do log de transações
9. Execução do trigger AFTER
10. Commit da transação (Confirmação)

Sintaxe do Trigger

CREATE TRIGGER nome_trigger

ON tabela

[WHIT ENCRYPTIO] – Opcional

AFTER | INSTEAF OF

[INSERT, UPDATE, DELETE]

AS

Código do Trigger

Argumentos do Trigger

nome_trigger = Nome do trigger

tabela = tabela (ou VIew) onde será executado o trigger.

WITH ENCRYPTION = Criptografia o código do trigger (opcional)

AFTER = O trigger é disparado quando todas as operações especificadas nas declarações SQL tiverem sido executadas com sucesso.

INSTEAD OF = o trigger é disparado no lugar das declarações SQL.

DELETE, UPDATE, INSERT = Definem quias tipos de modificações na tabela ativarão o trigger

Criar um Trigger AFTER - Exemplo

```
CREATE TRIGGER teste_trigger_after

ON Editora

AFTER INSERT

AS

PRINT ‘Olá Mundo’;
```

Criar um Trigger AFTER - Exemplo

```
CREATE TRIGGER trigger_after

ON Editora

AFTER INSERT

AS

INSERT INTO Autor VALUES ('Juca da SIlva','Brasileiro')

INSERT INTO Livro VALUES ('B2312','Mais um Juca no Brasil',1995,6,1)
```

Criar um Trigger INSTEAD OF - Exemplo

```
CREATE TRIGGER teste_trigger_insteafof

ON Autor

INSTEAD OF INSERT

AS

PRINT ‘Olá de novo! Não Inserir o registro desta vez!’;
```

Habilitar e Desabilitar Triggers

O Administrador do Sistema pode desabilitar temporariamente um trigger se houver necessidade. Para isso, use o comando DDL ALTER TABLE;

ALTER TABLE nome_tabela

ENABLE|DISABLE TRIGGER nome_trigger

Habilitar e Desabilitar Triggers

O Administrador do Sistema pode desabilitar temporariamente um trigger se houver necessidade. Para isso, use o comando DDL ALTER TABLE;

ALTER TABLE nome_tabela

ENABLE|DISABLE TRIGGER nome_trigger

Exemplo:

ALTER TABLE Editora

DISABLE TRIGGER trigger_after

Verificar a existência de triggers

Em um **tabela específica**:

EXEC sp_helptrigger @tabname=Nome_tabela

Exemplo:

EXEC sp_helptrigger @tabname=Editora

Verificar a existência de triggers

No banco de dados todo:

USE nome_banco_dados

SELECT *

FROM sys.triggers

WHERE is_disabled = 0 OR is_disabled = 1

0 = Triggers desabilitados; 1 - Triggers Habilitados

Determinando as colunas atualizadas

A função UPDATE() retorna True se uma coluna especificada for altera por uma transação DML

Podemos criar um gatilho que executa um código caso a coluna especificada seja alterada por comando DML usando essa função.

Determinando as colunas atualizadas

```
CREATE TRIGGER trigger_after_autores
ON Autor
AFTER INSERT,UPDATE
AS
IF UPDATE (nome_autor)
        BEGIN
                PRINT ‘O nome foi alterado’
        END
ELSE
        BEGIN
                PRINT ‘Nome não foi modificado’
        END
```

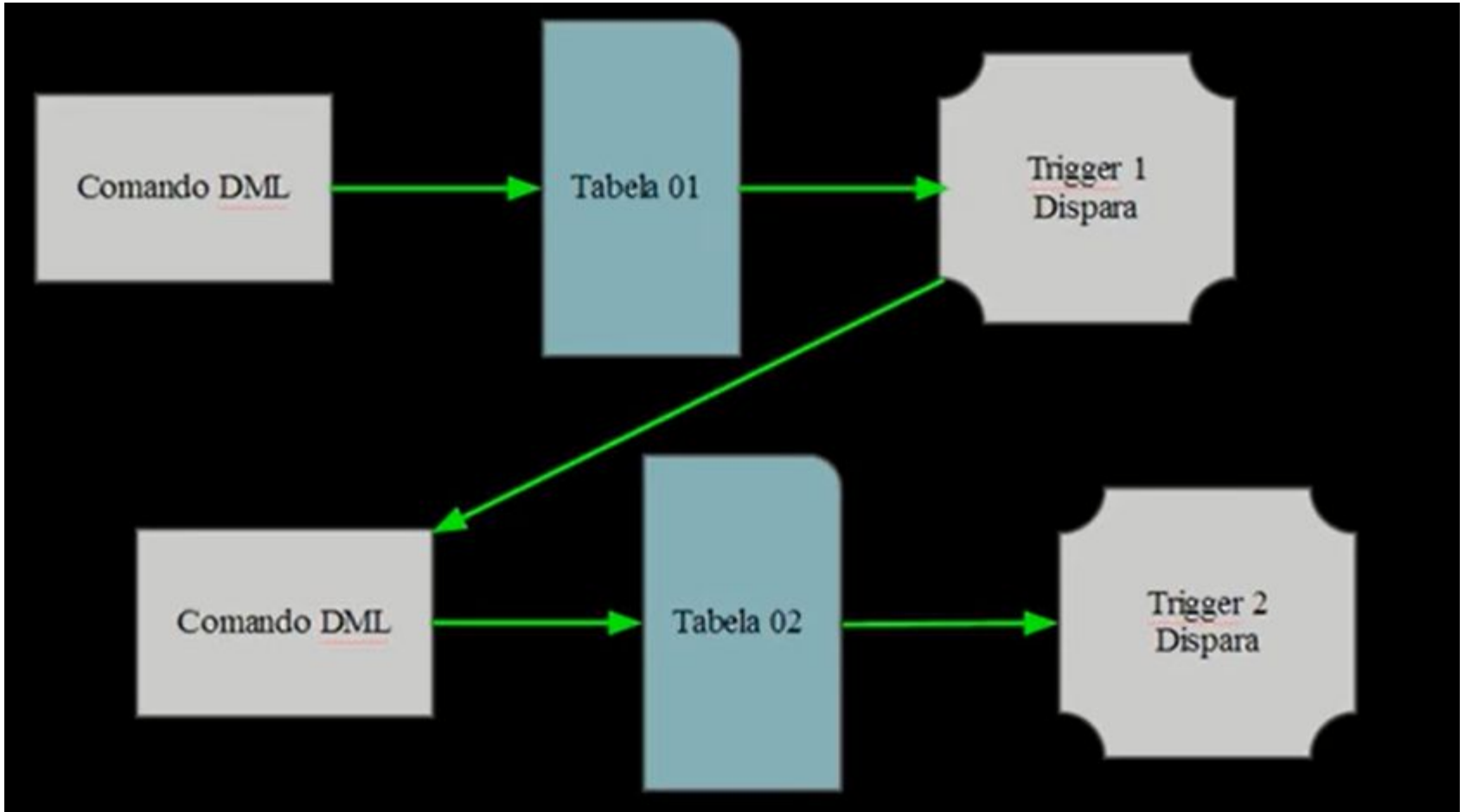
Aninhamento de Triggers DML

Um trigger, ao ser disparado, pode executar um declaração DML que leva ao disparo de outro Trigger

Para isso, a opção de servidor “Permitir que gatilhos disparem outros gatilhos”, em propriedades do Servidor -> Avançado, deve estar configurada como True (é o padrão)

Para desabilitar / habilitar a opção de aninhamento de trigger, use o comando:

```
EXE sp_configure ‘Nested Triggers’, 0|1
RECONFIGURE;
```


Comando DML
Tabela 01
Trigger 1
Dispara
Comando DML
Tabela 02
Trigger 2
Dispara

Triggers Recursivos

Um gatilho recursivo é um tipo especial de trigger AFTER aninhado.

O trigger recursivo ocorre quando um trigger executa uma declaração DML que o dispara novamente. Podemos habilitar ou desabilitar os triggers recursivos com o comando ALTER DATABASE:

```
ALTER DATABASE nome_banco_dados
SET RECURSIVE_TRIGGER ON|OFF
```

Triggers Recursivos - Exemplo

Vamos criar uma tablea chamada “trigger_recursivo” com um campo PK, tipo INT, chamado codigo, no banco Biblioteca:

```
CREATE TABLE Trigger_recursivo
(
    codigo INT PRIMARY KEY
)
```

Vamos inserir um registo que irá disparar o trigger a seguir, o qual por sua vez irá inserir outro registro, que o dispara novamente:

Criar Trigger Recursivo

```
CREATE TRIGGER trigger_rec
ON Trigger_recursivo
AFTER INSERT
AS
DECLARE @cod INT
SELECT
@cod = MAX(codigo)
FROM Trigger_recursivo
IF @cod < 10
        BEGIN
                INSERT INTO Trigger_recursivo
                SELECT MAX(codigo) + 1 FROM Trigger_recursivo
        END
ELSE
        BEGIN
                PRINT 'Trigger Recursivo Finalizado'
        END
```

Criação de exemplos

Abra o SQL Server e vamos a prática....

Bibliografia

- Christopher John Date. An introduction to database systems. Pearson Education India, 1981 (ver página 27).
- Ramez Elmasri e Sham Navathe. Fundamentals of Database Systems. 7ª edição. Pearson, 2016 (ver páginas 7, 8, 10, 16, 17, 19, 20).
- Nenad Jukic, Susan Vrbsky e Svetlozar Nestorov. Database systems: Introduction to databases and data warehouses. Pearson, 2014 (ver página 23).
- Michael McLaughlin. MySQL Workbench: Data Modeling Development. McGraw Hill Professional, 2013 (ver página 23).
- SQL Tutorial. w3schools, 2022. Disponível em: https://www.w3schools.com/sql/default.asp. Acesso em: 20, abril de 2022