Gatilhos (*Triggers*)

Prof. Márcio Bueno
{bd2tarde,bd2noited}@marciobueno.com

O que é um gatilho *(trigger)* ?

- Um gatilho é um tipo especial de procedimento armazenado (*stored procedure*) que executa quando uma tentativa para modificar dados é feita em uma tabela protegida pelo gatilho.
- Um gatilho é uma regra do tipo E_C_A:
  - E: Evento
  - C: Condição a ser satisfeita na presença do evento E
  - A: Ação a ser tomada caso a condição C seja satisfeita

Gatilhos (i)

- Associados a uma tabela
  - São definidos para tabelas específicas
- Invocados automaticamente
  - Quando uma tentativa de modificação na tabela é feita, e um gatilho foi definido para uma determinada ação (inserir, excluir ou alterar), o gatilho executa automaticamente

Gatilhos (ii)

- É uma transação
  - O gatilho e o comando que o dispara são tratados como uma única transação que pode ser desfeita de qualquer ponto dentro do gatilho.
  - Um gatilho pode incluir um ROLLBACK TRANSACTION mesmo sem ter um BEGIN TRANSACTION.
    - O comando que invoca o gatilho é considerado o início de uma transação implícita, a menos que um BEGIN TRANSACTION seja incluído.
    - Se um gatilho que inclui um ROLLBACK TRANSACTION é disparado de uma transação com BEGIN TRANSACTION, a transação toda é desfeita.

Considerações para usar gatilhos

- São usados para operações de inserção, exclusão e alteração
- São reativos
  - Os gatilhos são executados depois de uma operação (INSERT, DELETE ou UPDATE) ser executada na tabela para a qual o gatilho foi definido
- Tabelas podem ter vários gatilhos
  - Cada gatilho pode ser definido para uma ou várias ações
- Gatilhos não retornam resultados

REVISÃO DE JUNÇÃO

```
USE joindb
SELECT nome_cliente, vendas.cliente_id, qtd
FROM clientes JOIN vendas
ON clientes.cliente_id = vendas.cliente_id
```

clientes

Nome_cliente	Cliente_id
Adão	1
Silva	2
Eva	3
Elias	4

resultado

Nome_cliente	cliente_id	qtd
Adão	1	15
Adão	1	5
Elias	4	34
Eva	3	11
Elias	4	22

vendas

Cliente_id	Prod_id	qtd
1	2	15
1	3	5
4	1	34
3	5	11
4	2	22

Criando gatilhos (SQL Server)

- Gatilhos são criados com o comando CREATE TRIGGER
- O comando especifica a tabela para a qual o gatilho é definido, os eventos para os quais o gatilho executa, e as instruções do gatilho.

```
CREATE TRIGGER [owner.] trigger_name
ON [owner.] table_name  {FOR {INSERT I UPDATE I DELETE }
AS
[IF UPDATE (columnn_name)....]
[{AND I OR} UPDATE (columnn_name)....]
```

Como um gatilho *Insert* funciona

- Quando um gatilho INSERT é disparado, novas linhas são adicionadas para a tabela do gatilho e a tabela *inserted*.
- A tabela *inserted* é uma tabela que mantém uma cópia das linhas que foram inseridas.
- As linhas na tabela *inserted* são duplicatas de uma ou mais linhas da tabela do gatilho

Exemplo 1

- O exemplo abaixo cria um gatilho que gera o número de identificação do cliente **cliente_id** para cada linha inserida na tabela.
- O campo **cliente_id** consiste da concatenação do campo **id** (gerado pelo sistema) concatenado com as **três primeiras letras do sobrenome** e a **primeira letra do nome** do cliente.

```
CREATE TABLE clientes
( id              int  IDENTITY (1,1),
cliente_id     char(5),
nome           char(10),
sobrenome  char(10) )
```

Exemplo 1 (continuação)

- Criando o gatilho

```
CREATE TRIGGER gera_cliente_id ON clientes FOR
INSERT AS
UPDATE clientes SET cliente_id =  (i.id +
SUBSTRING(i.sobrenome,1,3) +
SUBSTRING(i.nome,1,1) ))
FROM clientes c JOIN inserted i ON i.id = c.id
```

- O seguinte INSERT dispara o gatilho
  INSERT clientes (sobrenome, nome) VALUES (‘Damasco’, ‘José’)
- O seguinte SELECT verifica o efeito do gatilho
  SELECT * FROM clientes

Resultado:

id	id_cliente	sobrenome	nome
1	1DamJ	Damasco	José

Exemplo 2

- Considere as tabelas **emprestimo** e **copia** abaixo:

emprestimo

isbn	Copia_num	Membro_num	data
1	1	1	03/08/01
4	1	1	03/08/01
1	2	2	03/08/01
3	1	3	03/08/01

copia

isbn	Copia_num	Titulo_num	Em_emprestimo
1	1	1	Y
1	2	1	Y
2	1	2	N
3	1	3	Y

Exemplo 2 (continuação)

- O gatilho abaixo foi criado para atualizar uma coluna derivada (**em_emprestimo**) na tabela **copia** sempre que um livro é emprestado (ou seja, quando um registro é incluído na tabela **emprestimo**).

```
USE biblioteca
CREATE TRIGGER insere_emprestimo
    ON emprestimo  FOR INSERT
    AS
      UPDATE c SET em_emprestimo = 'Y'
            FROM copia c JOIN  inserted i
            ON c.isbn = i.isbn AND c.copia_num =
                  i.copia_num
```

Como um gatilho *Delete* funciona

- Quando um gatilho DELETE é disparado, as linhas são removidas da tabela do gatilho são inseridas na tabela *deleted*
- A tabela *deleted* é uma tabela que mantém uma cópia das linhas que foram removidas.

Exemplo 3

- O gatilho abaixo foi criado para atualizar uma coluna derivada (**em_emprestimo**) na tabela **copia** sempre que um livro é devolvido (ou seja, quando um registro é excluído da tabela **emprestimo**).

```
USE biblioteca
CREATE TRIGGER remove_emprestimo
ON emprestimo  FOR DELETE
    AS
     UPDATE c SET em_emprestimo = 'N'
     FROM copia c JOIN deleted d
     ON c.isbn = d.isbn AND c.copia_num =
                d.copia_num
```

Como um gatilho *update* funciona

- UM gatilho UPDATE pode ser visto como dois passos – o passo DELETE e o passo INSERT.
- O passo DELETE captura a imagem dos dados antes da modificação e passo INSERT captura a imagem do dado após a modificação.
- Quando um UPDATE é executado na tabela as linhas originais são movidas para a tabela *deleted* e as linhas atualizadas são movidas para a tabela *inserted*.

Exemplo 4

- O gatilho abaixo evita a modificação do número do usuário.

```
USE biblioteca
CREATE TRIGGER atualiza_membro
    ON membro  FOR UPDATE
    AS
    IF UPDATE (membro_num)
      BEGIN
        PRINT (‘Número do cliente não pode ser modificado’)
        ROLLBACK TRANS
      END
```

Exemplo 5

```
CREATE TRIGGER VerifiqueEstoque
ON Produtos
FOR UPDATE
AS
IF (SELECT emEstoque from INSERTED) < 0
BEGIN
  PRINT ‘Não pode vender mais produtos do que existem’
  PRINT ‘Operação cancelada’
  ROLLBACK TRANS
END
```

Exemplo 6

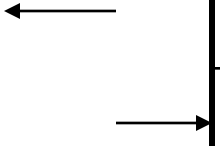
```
CREATE TRIGGER VerificaFone ON clientes
FOR UPDATE
AS
IF UPDATE (telefone)
BEGIN
    PRINT ‘Não pode atualizar telefone’
    PRINT ‘Operação cancelada’
    ROLLBACK TRANS
END
```

Exemplo 7

- O exemplo seguinte mostra como o gatilho garante a integridade da tabela reservas por remover a entrada de reserva quando um membro efetiva o empréstimo de um livro.

empréstimo

isbn	Copia_num	Mem_num
1	1	1
4	1	7
4	2	4



reserva

isbn	Mem_num	Data_res
1	1	10/01/02
1	2	11/01/02
4	7	12/01/02

Exemplo 7 (continuação)

```
USE biblioteca
CREATE TRIGGER remove_reserva
    ON emprestimo FOR INSERT
    AS
    IF (SELECT r.membro_num FROM reserva r
                JOIN inserted i
        ON r.membro_num = i.membro_num
            AND r.isbn = i.isbn) > 0
    BEGIN
      DELETE r FROM reserva r JOIN inserted i
       ON r.membro_num = i.membro_num
           AND r.isbn = i.isbn)
    END
```

Gatilho INSTEAD OF (i)

- Quando você utiliza um gatilho INSTEAD OF, o código original que fez uma modificação na tabela não é executado
- Em vez disso, é executado o código do gatilho
- Por exemplo, você poderia criar um gatilho em uma tabela AUTORES para notificar aos usuários que os autores não podem ser excluídos
- O mesmo poderia ser feito com gatilho padrão, mas exigiria que os dados fossem realmente excluídos

Gatilho INSTEAD OF (ii)

- Exemplo
  CREATE TRIGGER trIO_Delautores
  ON autores INSTEAD OF DELETE
  AS
  PRINT ‘Você não pode remover um autor’

- Para executar o gatilho
  DELETE autores WHERE au_lnome = ‘Loiola’