UNIVERSIDADE FEDERAL DE ITAJUBÁ

Banco de Dados I

Trigger
Baseado no material da profa Vanessa Cristina de Oliveira

Vanessa Cristina Oliveira de Souza

Triggers

- *Trigger* ou gatilho é um **comando executado automaticamente** pelo sistema como um efeito colateral de uma modificação no banco de dados.
- Resumidamente, é uma tarefa a ser executada no momento em que uma alteração é feita no banco de dados.

Triggers

- São usados para realizar tarefas relacionadas com validações, restrições de acesso, rotinas de segurança e consistência de dados
- Portanto, também são considerados um tipo de restrição de integridade do banco

Triggers

- Um *Trigger* é um **bloco de comandos** que é automaticamente executado quando um comando INSERT , DELETE ou UPDATE for executado em uma tabela do banco de dados
- Ação disparada automaticamente pelo banco

Triggers

- Todo e qualquer *trigger* trabalha sobre **transações**, ou seja, ele cria internamente um bloco de transação e tudo que estiver dentro dele será executado neste mesmo bloco
- É um bloco de comandos transacional
  - Faz parte de uma transação

Triggers

- Para projetar um *trigger*, precisamos
  - Especificar os eventos e os momentos sob as quais o gatilho deve ser acionado
    - Eventos: Insert, Update, Delete
    - Momentos: antes ou depois
  - Especificar as ações a serem tomadas quando o gatilho é executado

Trigger - Exemplo

- Dado o banco universidades
- Observe a tabela turmas:

```
mysql> describe turmas;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| disciplina | varchar(10) | NO   | PRI |         |       |
| codigo     | varchar(10) | NO   | PRI |         |       |
| vagas      | smallint(6) | YES  |     | NULL    |       |
| professor  | smallint(6) | YES  | MUL | NULL    |       |
+------------+-------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

Chave estrangeira
para Professores

Trigger - Exemplo

- Observe a tabela professores:

```
mysql> describe professores;
+----------------+-----------------------------------------------------+------+---
---+---------+-------+
| Field          | Type                                                | Null | K
ey | Default | Extra |
+----------------+-----------------------------------------------------+------+---
---+---------+-------+
| matricula      | smallint(6)                                         | NO   | P
RI | 0       |       |
| nome           | varchar(100)                                        | NO   |
   | NULL    |       |
| CPF            | varchar(11)                                         | NO   | U
NI | NULL    |       |
| sexo           | enum('M','F')                                       | NO   |
   | NULL    |       |
| data_nascimento | date                                               | NO   |
   | NULL    |       |
| titulacao      | enum('graduado','especialista','mestre','doutor')   | YES  |
   | NULL    |       |
| categoria      | enum('auxiliar','assistente','adjunto','titular')   | YES  |
   | NULL    |       |
| nro_Turmas     | smallint(6)                                         | YES  |
   | 0       |       |
+----------------+-----------------------------------------------------+------+---
```

Trigger - Exemplo

- Para manter a consistência do banco, ao criar uma turma e associá-la a um professor, o campo nro_turmas da tabela professores deve ser atualizado.
  - nro_turmas = nro_turmas + 1

Trigger - Exemplo

Relação Professores

Matricula	Nome	...	Nro_turmas
01	Maria		0
02	João		0

Trigger - Exemplo

Relação Turmas

Disciplina	Código	Vagas	Professor

Vamos inserir a a turma CCO_2018_1 com a disciplina COM230 e associá-la ao professor João

Trigger - Exemplo

Relação Turmas

Disciplina	Código	Vagas	Professor
COM230	CCO_2018_1	25	02

Seguindo os requisitos do sistema, ao **INSERIR** uma tupla na relação Turmas, o atributo nro_turmas na relação professor deverá ser atualizado.

Trigger - Exemplo

Relação Turmas

Disciplina	Código	Vagas	Professor
COM230	CCO_2018_1	25	02

Relação Professores

Matricula	Nome	...	Nro_turmas
01	Maria		0
02	João		1

A inserção na relação Turmas acarreta em uma ação no banco de dados:
• atualizar o número de turmas na relação professores.

Trigger - Exemplo

- Neste caso, a **condição para executar o gatilho** é uma inserção na tabela *Turmas*
- Qual outra condição poderia ser usada para disparar o *trigger*?
  - Vamos precisar alterar a tabela professor caso uma turma seja removida?
  - E se o professor de uma turma mudar?

Sintaxe Básica

CREATE TRIGGER nome_trigger

tempo_trigger<AFTER, BEFORE>

evento_trigger<INSERT, UPDATE, DELETE>

ON <tabela>

FOR EACH **tipo_trigger** <ROW, STATEMENT>‘

trigger_comandos‘;

Sintaxe Básica

- tempo_trigger
  - Quando o *trigger* será executado de acordo com o evento
  - As opções são:
    - **AFTER**
      - Os comandos do corpo do *trigger* serão executados DEPOIS dos dados da tabela serem alterados.
    - **BEFORE**
      - Os comandos do corpo do *trigger* serão executados ANTES dos dados da tabela serem alterados.

Trigger - Exemplo

Relação Turmas

Disciplina	Código	Vagas	Professor
COM230	CCO_2017_1	25	02

Relação Professores

Matricula	Nome	...	Nro_turmas
01	Maria		0
02	João		1

QUAL O TEMPO_TRIGGER NESSE CASO?

Sintaxe Básica

CREATE TRIGGER nome_trigger

tempo_trigger<AFTER. BEFORE>

evento_trigger<INSERT, UPDATE, DELETE>

ON <tabela>

FOR EACH **tipo_trigger** <ROW, STATEMENT>‘

trigger_comandos‘;

Sintaxe Básica

- evento_trigger
  - evento que **dispara** o *trigger*;
  - As opções são:
    - INSERT
    - UPDATE
    - DELETE

Sintaxe Básica

- Pode-se definir os *triggers* para serem disparados, por exemplo, antes (BEFORE) ou depois (AFTER) de um INSERT
- Percebe-se então que, para cada momento (BEFORE ou AFTER), pode-se ter um *trigger* a ser disparado para processar alguma regra

Trigger - Exemplo

Relação Turmas

Disciplina	Código	Vagas	Professor
COM230	CCO_2017_1	25	02

Relação Professores

Matricula	Nome	...	Nro_turmas
01	Maria		0
02	João		1

QUAL O(S) EVENTO(S)_TRIGGER NESSE CASO?

Operadores e comandos DML

- Cada tabela de uma banco de dados pode ter 6 *TRIGGERS*, já que temos dois momentos que disparam os *TRIGGERS* e 3 eventos:
  - BEFORE INSERT
  - BEFORE UPDATE
  - BEFORE DELETE
  - AFTER INSERT
  - AFTER UPDATE
  - AFTER DELETE

Sintaxe Básica

CREATE TRIGGER nome_trigger

tempo_trigger<AFTER, BEFORE>

evento_trigger<INSERT, UPDATE, DELETE>

ON <tabela>

FOR EACH **tipo_trigger** <ROW, STATEMENT>‘

trigger_comandos‘;

Sintaxe Básica

CREATE TRIGGER nome_trigger

tempo_trigger<AFTER, BEFORE>

evento_trigger<INSERT, UPDATE, DELETE>

ON <tabela>

FOR EACH **tipo_trigger** <ROW, STATEMENT>

‘trigger_comandos‘;

Sintaxe Básica

- Tipo_trigger
  - Define quantas vezes um *trigger* será executado.
  - O *trigger* pode ser executada uma vez para a instrução que a disparou ou ser disparada para cada linha afetada pela instrução que disparou o *trigger*.
- As opções são:
  - STATEMENT
    - Será disparado uma vez para cada evento de *trigger*, mesmo que nenhuma linha tenha sido afetada.
  - ROW
    - O *trigger* será executado toda vez que a tabela for afetada pelo evento do *trigger*.
    - Se nenhuma linha for afetada, o trigger não será executado.

Trigger - Exemplo

Relação Turmas

Disciplina	Código	Vagas	Professor
COM230	CCO_2017_1	25	02

Relação Professores

Matricula	Nome	...	Nro_turmas
01	Maria		0
02	João		1

QUAL O TIPO_TRIGGER NESSE CASO?

Sintaxe Básica

CREATE TRIGGER nome_trigger

tempo_trigger<AFTER, BEFORE>

evento_trigger<INSERT, UPDATE, DELETE>

ON <tabela>

FOR EACH **tipo_trigger** <ROW, STATEMENT>

‘trigger_comandos‘;

Sintaxe

- trigger_comandos
  - Comandos SQL executados pelo *trigger*

Trigger - Exemplo

Relação Turmas

Disciplina	Código	Vagas	Professor
COM230	CCO_2017_1	25	02

Relação Professores

Matricula	Nome	...	Nro_turmas
01	Maria		0
02	João		1

QUAL COMANDOS DEVEM SER EXECUTADOS NESSE CASO?

Operadores

- Existem dois operadores importantíssimos que possibilitam acessar os valores das colunas da tabela alvo do comando SQL antes (BEFORE) ou depois (AFTER) da alteração que dispara o *trigger*
  - NEW
  - OLD

Operadores

- NEW.nome_coluna
  - permite verificar o valor enviado para ser inserido em uma coluna de uma tabela.

- OLD.nome_coluna
  - permite verificar o valor excluído ou a ser excluído na coluna.

Prática

Criar um *trigger* para o exemplo

CREATE TRIGGER nome_trigger

tempo_trigger<AFTER, BEFORE>

evento_trigger<INSERT, UPDATE, DELETE>

ON <tabela>

FOR EACH **tipo_trigger** <ROW, STATEMENT>‘

trigger_comandos‘;

Trigger - Exemplo

Relação Turmas

Disciplina	Código	Vagas	Professor
COM230	CCO_2017_1	25	02

Relação Professores

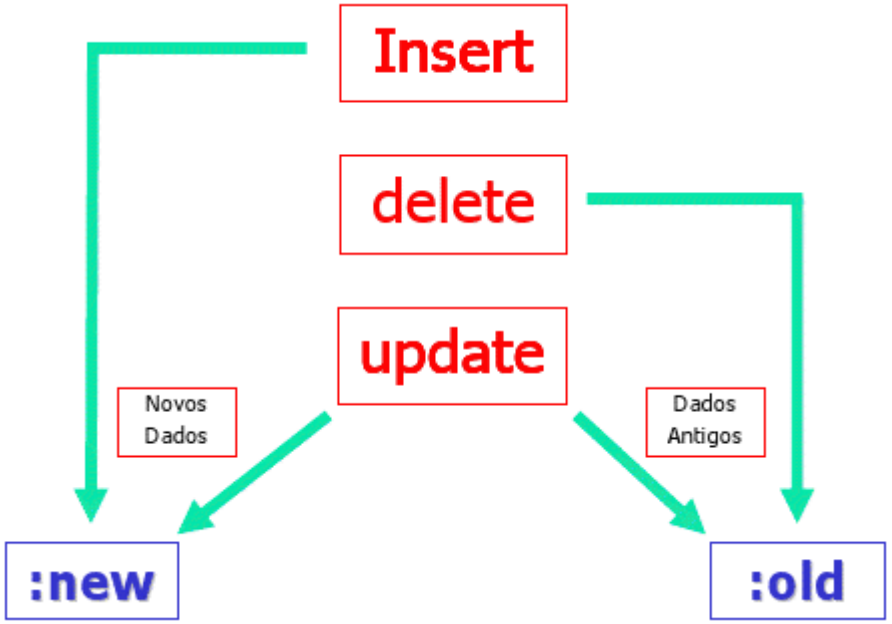
Matricula	Nome	...	Nro_turmas
01	Maria		0
02	João		1

```
CREATE TRIGGER atualiza_nro_turmas AFTER INSERT ON Turmas
FOR EACH ROW
BEGIN
UPDATE Professores SET nro_turmas = nro_turmas + 1 WHERE
matricula = NEW.professor;
END;
```

Entre a ação que dispara o *trigger* e a ação realizada pelo *trigger*, o banco fica temporariamente inconsistente

Operadores e comandos DML

- INSERT
  - NEW disponível
  - OLD não disponível
- UPDATE
  - NEW disponível
  - OLD disponível
- DELETE
  - NEW não disponível
  - OLD disponível



Prática

Criar um *trigger* para quando uma turma for apagada no banco

```
CREATE TRIGGER atualiza_nro_turmas AFTER
  DELETE ON Turmas FOR EACH ROW
BEGIN
UPDATE Professores SET nro_turmas = nro_turmas -1
  WHERE matricula = OLD.professor;
END;
```

Prática

- Criar um trigger para quando o professor de uma uma turma for alterado

```
CREATE TRIGGER atualiza_nro_turmas AFTER UPDATE
   ON Turmas OF professor FOR EACH ROW
BEGIN
UPDATE Professores SET nro_turmas = nro_turmas +1
   WHERE matricula = NEW.professor;
UPDATE Professores SET nro_turmas = nro_turmas -1
   WHERE matricula = OLD.professor;
END;
```