



INSTITUTO FEDERAL
Rio Grande do Sul

Banco de Dados II

Prof. Bruna Flor da Rosa



LINGUAGEM SQL STRUCTURED QUERY LANGUAGE

Triggers

Triggers

Triggers (gatilhos) são trechos de códigos desenvolvidos para serem executados automaticamente quando alguma tabela do banco de dados sofre alteração.

Os gatilhos que disparam as triggers são operações de INSERT, UPDATE ou DELETE executadas sobre uma tabela definida como alvo.



Triggers

Triggers são semelhantes às procedures pois pode-se incluir blocos com estruturas e comandos típicos das linguagens de programação em seu corpo.

Vantagens:

- Uma forma simples de realizar validação.
- Execução se dá imediatamente após o evento, ao invés de esperar a chamada do CALL.

Desvantagens:

- A aplicação não tem conhecimento da existência do trigger, e isto pode gerar confusão na hora de atualizar o sistema ou na busca por erros.



Triggers

Código:

```
DELIMITER //  
CREATE TRIGGER nome_da_trigger  
instante evento ON tabela  
FOR EACH ROW BEGIN  
/*Corpo da trigger*/  
END //  
DELIMITER ;
```

- **CREATE TRIGGER** é o comando utilizado para criar a nossa trigger, da mesma maneira que utilizamos create table para tabelas e create procedure para procedimentos.
- **nome_da_trigger** será o nome que esta trigger receberá. Note que esta trigger deverá ter um nome único no banco de dados inteiro.

Triggers

Código:

```
DELIMITER //  
CREATE TRIGGER nome_da_trigger  
instante evento ON tabela  
FOR EACH ROW BEGIN  
/*Corpo da trigger*/  
END //  
DELIMITER ;
```

- **instante** receberá as palavras BEFORE ou AFTER.

BEFORE significa que sua trigger será executada antes que alteração que engatilhou a trigger entre em vigor.

AFTER é utilizado quando queremos que a trigger seja executada depois da alteração que a engatilhou.

Triggers

Código:

```
DELIMITER //  
CREATE TRIGGER nome_da_trigger  
instante evento ON tabela  
FOR EACH ROW BEGIN  
/*Corpo da trigger*/  
END //  
DELIMITER ;
```


- **evento** indica qual evento que irá disparar a trigger. As três opções disponíveis aqui são INSERT, UPDATE ou DELETE.
- **tabela** será o nome da tabela que o trigger ficará “vigiando”. Quando o evento escolhido acontecer (antes ou depois) na tabela indicada, o trigger iniciará.



Triggers

Código:

```
DELIMITER //  
CREATE TRIGGER nome_da_trigger  
instante evento ON tabela  
FOR EACH ROW BEGIN  
/*Corpo da trigger*/  
END //  
DELIMITER ;
```

- **FOR EACH ROW BEGIN** indica que o trigger será executado sempre e para todas as linhas da tabela escolhida que acontecer o evento.
 - **END //** indica onde o trigger irá terminar. O delimitador // é usado pelo mesmo motivo que o visto nas Stored Procedures.
- 

Triggers

Para acessar e modificar os valores dos campos da tabela alvo do trigger, existem as palavras-chave NEW e OLD.

- NEW é utilizado para acessar o novo valor de um campo, após ele ser modificado.
- OLD é utilizado para acessar o antigo valor do campo, antes da modificação.

O evento INSERT permite somente acesso aos campos através da palavra-chave NEW (pois não há valor antigo, antes da inserção).

O evento DELETE só permite o acesso com a palavra-chave OLD (pois após o delete, não haverá valor novo).

O evento UPDATE permite ambos.



Evento INSERT

```
DELIMITER //  
DROP TRIGGER IF EXISTS calcula_total//  
CREATE TRIGGER calcula_total  
BEFORE INSERT ON quiz  
FOR EACH ROW BEGIN  
    SET NEW.total = NEW.pontosA + NEW.pontosB + NEW.pontosC +  
END//  
DELIMITER ;
```

Note que o valor de um campo só pode ser alterado se o instante escolhido for BEFORE.



Evento DELETE

```
DELIMITER //  
DROP TRIGGER IF EXISTS apaga_mensagens//  
CREATE TRIGGER apaga_mensagens  
BEFORE DELETE ON usuarios  
FOR EACH ROW BEGIN  
    DELETE FROM mensagens WHERE usuario = OLD.codigo;  
END//  
DELIMITER ;
```

Note que caso o campo que queremos alterar seja de uma tabela que não seja a tabela alvo do trigger, devemos usar os comandos SQL INSERT, UPDATE ou DELETE. Para os campos da tabela alvo usamos direto o campo com as palavras-chave OLD ou NEW normalmente.

Evento UPDATE

```
DELIMITER //  
DROP TRIGGER IF EXISTS atualiza_historico//  
CREATE TRIGGER atualiza_historico  
AFTER UPDATE ON contas  
FOR EACH ROW BEGIN  
    DECLARE diferenca DECIMAL(10,2);  
    SET diferenca = NEW.saldo - OLD.saldo;  
    INSERT INTO historico(conta,modificacao,hora)  
VALUES(NEW.numero,diferenca,now());  
END//  
DELIMITER ;
```

Como pode ser visto, no evento UPDATE podemos usar tanto o OLD como NEW, pois existe dados na linha tanto antes como após a execução do UPDATE na tabela.

