

# Banco de Dados : Triggers

Prof. Márcio Funes



## Plano de aula

Triggers

Variáveis

Estruturas de decisão

Exercícios

# Introdução

## O contexto

Uma **Trigger** é um procedimento associado a uma tabela no banco de dados. Elas são utilizadas para automatizar chamadas a rotinas.

O código é executado quando ocorre certa *ação* no banco de dados.

**Triggers** podem ser executadas quando uma das seguintes ações ocorre:

- Inserção
- Atualização
- Deleção

Podem ser disparadas **antes** ou **depois** da *ação*.

## O contexto

Para exemplificar o uso de gatilhos, tomaremos como cenário uma certa aplicação financeira que contém um controle de caixa e efetua vendas.

Sempre que forem registradas ou excluídas vendas, essas operações devem ser automaticamente refletidas na tabela de caixa, aumentando ou reduzindo o saldo.

```
create database lojaLegal;  
use lojaLegal;
```

```
create table caixa(  
    dia date,  
    saldo_inicial decimal(10,2),  
    saldo_final decimal(10,2)  
);
```

```
create table vendas(  
    id int primary key,  
    dia date,  
    valor decimal(10,2)  
);
```

```
insert into caixa values ('2023-05-30', 100, 100);  
select * from caixa;
```

100 %

 Resultados  Mensagens

	dia	saldo_inicial	saldo_final
1	2023-05-30	100.00	100.00

# Triggers

## Triggers

```
CREATE TRIGGER [NOME DO TRIGGER]
ON [NOME DA TABELA]
[FOR/AFTER/INSTEAD] [INSERT/UPDATE/DELETE]
AS
BEGIN
END
```



## Triggers – SQL SERVER

```
CREATE TRIGGER atualizaCaixa
ON vendas
FOR INSERT
AS
BEGIN
    DECLARE @valor decimal(10,2), @dia date

    SELECT @dia = dia, @valor = valor FROM INSERTED

    UPDATE caixa SET saldo_final = saldo_final + @valor
    WHERE dia = @dia;
END
```

## Triggers – MY SQL


```
CREATE TRIGGER atualizaCaixa
AFTER INSERT ON vendas
FOR EACH ROW
BEGIN
    DECLARE valor DECIMAL(10, 2);
    DECLARE dia DATE;

    SELECT NEW.dia, NEW.valor INTO dia, valor;

    UPDATE caixa SET saldo_final = saldo_final + valor
    WHERE dia = dia;
END;
```

```
insert into vendas values (1, '2023-05-30', 10);
```

00 % ▾

 Mensagens

(1 linha afetada)

(1 linha afetada)

```
select * from vendas;  
select * from caixa;
```

100 %


 Resultados  Mensagens

	id	dia	valor
1	1	2023-05-30	10.00

	dia	saldo_inicial	saldo_final
1	2023-05-30	100.00	110.00

```
insert into vendas values (2, '2023-05-30', 20);
```

100 %

 Mensagens


(1 linha afetada)

(1 linha afetada)

```
select * from vendas;  
select * from caixa;
```

100 %

 Resultados

 Mensagens

	id	dia	valor
1	1	2023-05-30	10.00
2	2	2023-05-30	20.00

	dia	saldo_inicial	saldo_final
1	2023-05-30	100.00	130.00

## Prevenir Delete ou Update sem WHERE – SQL SERVER

```
CREATE TRIGGER deleteSemWhere ON vendas
FOR UPDATE, DELETE AS
BEGIN
    DECLARE
        @Linhas_Alteradas INT = @@ROWCOUNT,
        @Linhas_Tabela INT = (
            SELECT SUM(row_count)
            FROM sys.dm_db_partition_stats
            WHERE [object_id] = OBJECT_ID('vendas') AND (index_id <= 1)
        )

    IF (@Linhas_Alteradas >= @Linhas_Tabela)
    BEGIN
        ROLLBACK TRANSACTION;
        RAISERROR ('Operações de DELETE e/ou UPDATE
sem cláusula WHERE não são permitidas
na tabela "vendas"', 15, 1);
        RETURN;
    END
END;
```

## Para saber mais...

<https://www.mysqltutorial.org/create-the-first-trigger-in-mysql.aspx/>

<https://dev.mysql.com/doc/refman/8.0/en/trigger-syntax.html>

<https://dev.mysql.com/doc/refman/8.0/en/triggers.html>

<https://www.mysqltutorial.org/mysql-triggers.aspx/>



## Exercícios

# Exercícios

1. Crie uma tabela para armazenar dados de pessoas. Como restrição, ninguém pode ter idade negativa, nem idade maior que 200 anos. Gere mensagens de erro adequadas para cada situação.
2. Crie um campo `ultima_atualizacao` que armazena a data e hora da última alteração de cada registro.
3. Crie uma tabela para log do banco de dados. Nessa tabela devem ser registrados metadados referentes a atividades no banco de dados como inserções, atualizações e deleções. Cada registro desta tabela deve conter o nome da tabela alterada, o nome do usuário logado, data e hora da atividade.
4. Crie uma tabela para armazenar dados de funcionários. Como restrição, ninguém pode ter salário inferior a R\$ 12.000,00 e a jornada semanal deve estar entre 20 e 40 horas. Gere mensagens de erro adequadas para cada situação.

