

Universidade Tecnológica Federal do Paraná UTFPR - Campus Cornélio Procópio



AS33C Banco de Dados 2

Professor: Eduardo Cotrin Teixeira



cotrin@utfpr.edu.br

SQL - Triggers

- Regras ativas são implementadas por meio de gatilhos (triggers).
- Triggers seguem o paradigma E-C-A:

ECA = Evento-Condição-Ação

quando o evento ocorre,

se a condição é satisfeita,

Então a ação é executada



CREATE TRIGGER <nome>{BEFORE|AFTER|INSTEAD OF}{evento [OR...]}
ON tabela [FOR EACH {ROW|STATEMENT}] **EXECUTE PROCEDURE** função();

before | after | instead of determina se a função será chamada antes, depois ou em substituição ao evento.

evento indica a que evento o disparo da trigger está vinculado, e pode ser DELETE, UPDATE ou INSERT.

tabela indica a qual tabela a trigger estará associada.

row | statement especifica se a trigger deve ser disparada uma vez para cada linha afetada (row) ou uma vez por comando SQL (statement - padrão).

SQL - Triggers INSTEAD OF

Sintaxe:

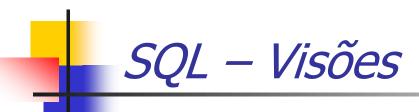
CREATE TRIGGER<nome>{BEFORE|AFTER|INSTEAD OF}{evento [OR...]}
ON tabela [FOR EACH { ROW | STATEMENT }]
EXECUTE PROCEDURE nome_da_função ()

before | after | instead of determina se a função será chamada antes, depois ou em **substituição** ao evento.

evento indica a que evento o disparo da trigger está vinculado, e pode ser DELETE, UPDATE ou INSERT.

tabela indica a qual tabela a trigger estará associada.

row | statement especifica se a trigger deve ser disparada uma vez para cada linha afetada pelo evento (row) ou uma vez por comando SQL (statement - padrão).



- **Tabela** = relação armazenada
 - São definidas com o comando CREATE TABLE e existem fisicamente no BD.
- Visão = relação (tabela) virtual
 - Visões são relações que não são armazenadas fisicamente.
 - São definidas como consultas.
 - Podem ser consultadas como se existissem fisicamente.



Visão dos projetos "curtos": Número, Nome e Duração dos projetos com duração abaixo de 3 meses.

CREATE VIEW ProjetosCurtos AS SELECT PNro, PNome, PDuracao FROM Projeto WHERE PDuracao < 3;

PNro	PNome	PDuracao
P3	Alfa	2
P5	Paraíso	1



Operações de dados diretamente em visões:

- Só é possível em visões atualizáveis que são aquelas definidas por meio de seleção simples (sem DISTINCT) de alguns atributos de uma única tabela ou de outra visão atualizável.
- * A modificação nos dados da visão é possível se for possível traduzir a modificação da visão em uma modificação equivalente em tabelas-base.



CREATE VIEW ProjetosCurtos AS SELECT PNro, PNome, PDuracao FROM Projeto WHERE PDuracao < 3;

Visão ProjetosCurtos

PNro	PNome	PDuracao
P3	Alfa	2
P5	Paraíso	1

INSERT INTO ProjetosCurtos
VALUES ('P6','Beta',2);

PNro	PNome	PDuracao
P3	Alfa	2
P5	Paraíso	1
P6	Beta	2

Tabela Projeto

PNro	PNome	PDuracao	PCusto	
P1	Detroit	5	43.000	
P2	Pegasus	3	37.000	
P3	Alfa	2	26.700	
P4	Sea	3	21.200	
P5	Paraíso	1	17.000	

PNro	PNome	PDuracao	PCusto
P1	Detroit	5	43.000
P2	Pegasus	3	37.000
P3	Alfa	2	26.700
P4	Sea	3	21.200
P5	Paraíso	1	17.000
P6	Beta	2	null



CREATE VIEW ProjetosCurtos AS SELECT PNro, PNome, PDuracao FROM Projeto WHERE PDuracao < 3;

Visão ProjetosCurtos

PNro	PNome	PDuracao
P3	Alfa	2
P5	Paraíso	1
P6	Beta	2

INSERT INTO ProjetosCurtos VALUES ('P7','Gama',5);

PNro	PNome	PDuracao
P3	Alfa	2
P5	Paraíso	1
P6	B <u>e</u> ta	2

Não altera a visão, mas altera a tabela-base!

Ta	bel	a	Pr	oj	eto

PNro	PNome	PDuracao	PCusto
P1	Detroit	5	43.000
P2	Pegasus	3	37.000
P3	Alfa	2	26.700
P4	Sea	3	21.200
P5	Paraíso	1	17.000
P6	Beta	2	null

PNro	PNome	PDuracao	PCusto	
P1	Detroit	5	43.000	

P6 Beta 2 null			null	
P7	Gama	5	null	

Operações de dados diretamente em visões, mesmo que possíveis, não são desejáveis.

SQL – Visões

Visão do fornecimento de peças: Nome da Peça, Número do Projeto, do Fornecedor e Quantidade das peças fornecidas.

CREATE VIEW FornecePecas AS

SELECT PeNome, PNro, FNro, Quant

FROM Peca JOIN Fornece_para

USING (PeNro);

PeNome	PNro	FNro	Quant
Cinto	P4	F5	5
Volante	P2	F2	1
Lanterna	P4	F3	2
Limpador	P5	F4	3
Painel	P1	F1	1
Volante	P3	F2	1
Limpador	P5	F3	2



Visão FornecePecas

PeNome	PNro	FNro	Quant	
Cinto	P4	F5	5	
•••				
Volante	P3	F2	1	
Limpador	P5	F3	2	

INSERT INTO FornecePecas VALUES ('Cinto','P5','F1',4);

****** Error *******

ERROR: cannot insert into view "testevisao"

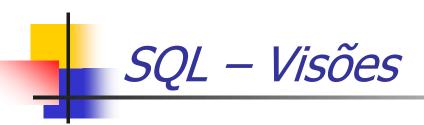
SQL state: 55000

Detail: Views that do not select from a single table or view are

not automatically updatable.

Hint: To enable inserting into the view, provide an INSTEAD OF

INSERT trigger or an unconditional ON INSERT DO INSTEAD rule.



- Para que a inserção (ou outra operação) seja possível, deve ser usada uma trigger com evento instead of (ao invés de).
- A função da trigger deve, se for possível, traduzir a modificação da visão em modificações equivalentes nas tabelas-base.
- Nem sempre é possível fazer a tradução (a visão pode não incluir campos de chave primária, por exemplo), portanto a função da trigger pode tomar qualquer providência necessária.



CREATE TRIGGER InsereFornecePecas INSTEAD OF INSERT ON FornecePecas FOR EACH ROW EXECUTE PROCEDURE InsereFornecePecas();

INSERT INTO FornecePecas VALUES ('Cinto','P5','F1',4);

NroPeca := SELECT PeNro FROM Peca WHERE PeNome='Cinto';
INSERT INTO Fornece_para VALUES (NroPeca, 'F1', 'P5', 4);

Visão FornecePecas

PeNome	PNro	FNro	Quant		
Cinto	P4	F5	5		
•••					
Volante	P3	F2	1		
Limpador	P5	F3	2		

Tabela Peça

<u>rabçia i cça</u>				
PeNro	PeNome			
PE1	Cinto			
PE2	Volante			
PE3	Lanterna			
PE4	Limpador			
PE5	Painel			

Tabela Fornece para

PeNro	FNro	PNro	Quant		
PE1	F5	P4	5		
•••					
PE2	F2	P3	1		
PE4	F3	P5	2		



CREATE OR REPLACE FUNCTION InsereFornecePecas() RETURNS TRIGGER AS \$\$ DECLARE NroPeca CHAR(5);

BEGIN

```
NroPeca := (SELECT PeNro FROM Peca WHERE PeNome = NEW.PeNome);
INSERT INTO Fornece para VALUES
```

(NroPeca, NEW.FNro, NEW.PNro, NEW.Quant);

RETURN NEW;

END;\$\$ LANGUAGE plpgsql;

INSERT INTO FornecePecas VALUES ('Cinto', 'P5', 'F1', 4);

Tabela Fornece para

PeNro	FNro	PNro	Quant		
PE1	F5	P4	5		
•••					
PE4	F3	P5	2		
PE1	F1	P5	4		

Visão FornecePecas
PeNome PNro FNro Quant
Cinto P4 F5 5

Limpador P5 F3 2
Cinto P5 F1 4



Como ficaria a exclusão?

```
DELETE FROM FornecePecas WHERE PeNome='Cinto' AND PNro='P5' AND FNro='F1';
 CREATE OR REPLACE FUNCTION ExcluifornecePecas() RETURNS TRIGGER
AS $$ DECLARE NroPeca CHAR(5);
BEGIN
   NroPeca := (SELECT PeNro FROM Peca WHERE PeNome = OLD.PeNome);
   DELETE FROM Fornece para WHERE Penro=NroPeca AND Fnro=OLD.Fnro
   AND PNro=OLD.PNro;
RETURN OLD;
END;$$ LANGUAGE plpgsql;
 CREATE TRIGGER ExcluiFornecePecas INSTEAD OF DELETE ON
 FornecePecas FOR EACH ROW EXECUTE PROCEDURE
ExcluiFornecePecas();
```