

SISTEMAS DE BANCO DE DADOS 2

AULA 13

TRIGGER - Programação Procedural

ORACLE

Vandor Roberto Vilardi Rissoli



APRESENTAÇÃO

- Fundamentos de *Trigger* (gatilho)
- Identificadores de Correlação
- Impossibilidade de uso de *Trigger*
- Gatilhos (*triggers*) de Sistema
- Referências



Trigger (gatilho)

Uma **trigger** é um subprograma associado a uma tabela, *view* ou evento. Ela pode ser acionada uma vez quando um determinado evento ocorrer ou várias vezes para cada tupla afetada por uma instrução DML

- inserção
- exclusão
- alteração



Trigger (gatilho)

- A trigger pode ser acionada após um determinado **evento** para registrá-lo ou efetuar alguma atividade anterior e/ou posterior a este evento.
- Ela pode ser acionada antes do evento para prevenir operações indevidas ou ajustar os novos dados para que estes estejam de acordo, por exemplo, com a regra de negócio envolvida.



Triggers em Oracle

- Tipos

- Para tabelas

- Triggers de DML (INSERT, UPDATE, DELETE)
 - Triggers de Sistema (DDL, e logs)

- Para visões

- Triggers de DML Instead-OF (INSERT, UPDATE, DELETE)



Triggers de DML

- **Tabela desencadeadora**
- **Instrução de disparo**
 - INSERT
 - UPDATE
 - DELETE
- **Tempo**
 - BEFORE
 - AFTER
- **Nível**
 - linha
 - instrução



Triggers de DML

- **Identificadores de correlação** - variáveis de vínculo PL/SQL (para triggers com nível de linha)
 - sempre vinculados à tabela desencadeadora do trigger
 - pseudoregistros do tipo *tabela_desencadeadora*%ROWTYPE

<div>instrução</div> <div>identificador</div>	:old	:new
INSERT	NULL	valores que serão inseridos
UPDATE	valores antes da atualização	novos valores para a atualização
DELETE	valores antes da remoção	NULL



Trigger (gatilho)

Turma = {Sigla, Numero, NAlunos}

Matrícula = {Sigla, Numero, Aluno, Ano, Nota}

```
CREATE OR REPLACE TRIGGER NroDeAlunos
AFTER DELETE ON Matricula
FOR EACH ROW /* nível de linha */
BEGIN
    UPDATE Turma
        SET NAlunos = NAlunos - 1
        WHERE Sigla = :old.Sigla AND
            Numero = :old.Numero;
EXCEPTION
    ....
END NroDeAlunos;
```

identificador :old
refere-se à tabela
Matricula

```
/*... em um bloco anônimo, por exemplo... */
DELETE FROM matricula WHERE aluno = 222;
```

O que acontece na base de dados???

Quais dados são disponíveis?

```
CREATE OR REPLACE TRIGGER Teste  
AFTER INSERT OR UPDATE OR DELETE ON LBD01_VINCULO_USP  
FOR EACH ROW
```

```
BEGIN
```

```
  insert into output values (:old.nrousp || '-'  
    ' || :old.tipovinc || '-' || :old.nome || '-'  
    ' || :old.dataingresso || '-' || :old.datanascimento || '-'  
    ' || :old.ativo);
```

```
  insert into output values (:new.nrousp || '-'  
    ' || :new.tipovinc || '-' || :new.nome || '-'  
    ' || :new.dataingresso || '-' || :new.datanascimento || '-'  
    ' || :new.ativo);
```

```
END;
```

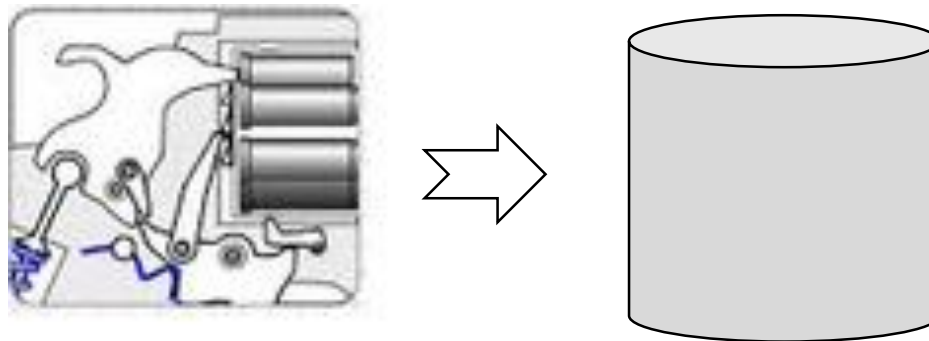
**Output*

```
CREATE TABLE output(msg varchar2(200));
```



Trigger e Stored Procedure

As *triggers*, assim como as *stored procedures*, são armazenadas no banco de dados e podem ser compostas de instruções SQL e PL/SQL no **ORACLE**.



As outras instruções DDL que podem ser executadas sobre as Triggers são: ALTER e DROP trigger.



Trigger e Stored Procedure

Entretanto, *stored procedures* e *triggers* diferem na forma como são acionadas:

Stored procedure é explicitamente acionada por um usuário, aplicação ou trigger.

Trigger é implicitamente disparada pelo servidor de banco de dados quando um determinado evento ocorre. O disparo da trigger independe do usuário ou aplicação que gerou o evento.



O que um trigger executa?

- Suponha os seguintes comandos SQL
 - INSERT
 - INSERT INTO LBD01_VINCULO_USP(NROUSP, TIPOVINC, NOME, DATAINGRESSO) VALUES(21, 2, 'João', '05/09/2002')
 - INSERT INTO LBD01_VINCULO_USP(NROUSP, TIPOVINC, NOME, DATAINGRESSO) VALUES(22, 1, 'Gilberto', '09/05/2000', '05/02/1980', 'y')
 - INSERT INTO LBD01_VINCULO_USP(NROUSP, TIPOVINC, NOME, DATAINGRESSO) VALUES(23, 3, 'Alfredo', '03/04/2002', '05/04/1982', 'y')
 - UPDATE
 - UPDATE LBD01_VINCULO_USP SET NOME = UPPER(NOME) WHERE NROUSP > 20;
 - DELETE
 - DELETE LBD01_VINCULO_USP WHERE NROUSP > 20;
- Quais são os valores de :old e :new para cada uma destas operações?



O que um trigger executa?

- Primeiramente, o corpo do trigger será executado 3 vezes para cada tipo de operação
- Os valores para cada execução considerando cada operação serão:

INSERT	:old						:new					
	NroUSP	TipoVinc	Nome	DataIngresso	DataNascimento	Ativo	NroUSP	TipoVinc	Nome	DataIngresso	DataNascimento	Ativo
	NULL	NULL	NULL	NULL	NULL	NULL	21	2	João	05/09/2002	NULL	y
	NULL	NULL	NULL	NULL	NULL	NULL	22	1	Gilberto	09/05/2000	NULL	y
	NULL	NULL	NULL	NULL	NULL	NULL	23	3	Alfredo	03/04/2002	NULL	y

- DataNascimento não foi inserido, e Ativo tem valor DEFAULT.



O que um trigger executa?

- Primeiramente, o corpo do trigger será executado 3 vezes para cada tipo de operação
- Os valores para cada execução considerando cada operação serão:

INSERT	:old						:new					
	NroUSP	TipoVinc	Nome	DataIngresso	DataNascimento	Ativo	NroUSP	TipoVinc	Nome	DataIngresso	DataNascimento	Ativo
	NULL	NULL	NULL	NULL	NULL	NULL	21	2	João	05/09/2002	NULL	y
	NULL	NULL	NULL	NULL	NULL	NULL	22	1	Gilberto	09/05/2000	NULL	y
UPDATE	NULL	NULL	NULL	NULL	NULL	NULL	23	3	Alfredo	03/04/2002	NULL	y
	NroUSP	TipoVinc	Nome	DataIngresso	DataNascimento	Ativo	NroUSP	TipoVinc	Nome	DataIngresso	DataNascimento	Ativo
	21	2	João	05/09/2002	NULL	y	21	2	JOÃO	05/09/2002	NULL	y
	22	1	Gilberto	09/05/2000	NULL	y	22	1	GILBERTO	09/05/2000	NULL	y
	23	3	Alfredo	03/04/2002	NULL	y	23	3	ALFREDO	03/04/2002	NULL	y



O que um trigger executa?

- A trigger será executada 3 vezes para cada tipo de operação
- Os valores para cada execução considerando cada operação serão:

	:old						:new					
INSERT	NroUSP	TipoVinc	Nome	DataIngresso	DataNascimento	Ativo	NroUSP	TipoVinc	Nome	DataIngresso	DataNascimento	Ativo
	NULL	NULL	NULL	NULL	NULL	NULL	21	2	João	05/09/2002	NULL	y
	NULL	NULL	NULL	NULL	NULL	NULL	22	1	Gilberto	09/05/2000	NULL	y
	NULL	NULL	NULL	NULL	NULL	NULL	23	3	Alfredo	03/04/2002	NULL	y
UPDATE	NroUSP	TipoVinc	Nome	DataIngresso	DataNascimento	Ativo	NroUSP	TipoVinc	Nome	DataIngresso	DataNascimento	Ativo
	21	2	João	05/09/2002	NULL	y	21	2	JOÃO	05/09/2002	NULL	y
	22	1	Gilberto	09/05/2000	NULL	y	22	1	GILBERTO	09/05/2000	NULL	y
	23	3	Alfredo	03/04/2002	NULL	y	23	3	ALFREDO	03/04/2002	NULL	y
DELETE	NroUSP	TipoVinc	Nome	DataIngresso	DataNascimento	Ativo	NroUSP	TipoVinc	Nome	DataIngresso	DataNascimento	Ativo
	21	2	JOÃO	05/09/2002	NULL	y	NULL	NULL	NULL	NULL	NULL	NULL
	22	1	GILBERTO	09/05/2000	NULL	y	NULL	NULL	NULL	NULL	NULL	NULL
	23	3	ALFREDO	03/04/2002	NULL	y	NULL	NULL	NULL	NULL	NULL	NULL



O que um trigger executa?

Esta análise é válida tanto para BEFORE quanto para AFTER.

- Primeiramente, o corpo do trigger será executado 3 vezes para cada tipo de operação
- Os valores para cada execução considerando cada operação serão:

	:old						:new					
INSERT	NroUSP	TipoVinc	Nome	DataIngresso	DataNascimento	Ativo	NroUSP	TipoVinc	Nome	DataIngresso	DataNascimento	Ativo
	NULL	NULL	NULL	NULL	NULL	NULL	21	2	João	05/09/2002	NULL	y
	NULL	NULL	NULL	NULL	NULL	NULL	22	1	Gilberto	09/05/2000	NULL	y
	NULL	NULL	NULL	NULL	NULL	NULL	23	3	Alfredo	03/04/2002	NULL	y
UPDATE	NroUSP	TipoVinc	Nome	DataIngresso	DataNascimento	Ativo	NroUSP	TipoVinc	Nome	DataIngresso	DataNascimento	Ativo
	21	2	João	05/09/2002	NULL	y	21	2	JOÃO	05/09/2002	NULL	y
	22	1	Gilberto	09/05/2000	NULL	y	22	1	GILBERTO	09/05/2000	NULL	y
	23	3	Alfredo	03/04/2002	NULL	y	23	3	ALFREDO	03/04/2002	NULL	y
DELETE	NroUSP	TipoVinc	Nome	DataIngresso	DataNascimento	Ativo	NroUSP	TipoVinc	Nome	DataIngresso	DataNascimento	Ativo
	21	2	JOÃO	05/09/2002	NULL	y	NULL	NULL	NULL	NULL	NULL	NULL
	22	1	GILBERTO	09/05/2000	NULL	y	NULL	NULL	NULL	NULL	NULL	NULL
	23	3	ALFREDO	03/04/2002	NULL	y	NULL	NULL	NULL	NULL	NULL	NULL

O que um Trigger NÃO executa?

- Não são permitidos comandos transacionais (SET TRANSACTION, COMMIT, SAVEPOINT, e ROLLBACK) dentro de uma trigger
- Por consequência, não são permitidos comandos DDL (CREATE, ALTER, e DROP), pois eles disparam COMMIT automaticamente
- Também não são permitidos procedimentos/funções que impliquem em controle transacional
- Assim, uma trigger fica sujeito à transação definida na sessão em que o trigger foi disparado



O que um Trigger NÃO executa?

- DML sobre tabelas mutantes
- Uma tabela mutante é um tabela que está sendo alterada por INSERT, UPDATE ou DELETE
- Erro **ORA-04091**: table is **mutating**, trigger/function may not see it



O que um Trigger NÃO executa?

- Exemplo - tabela mutante

```
CREATE OR REPLACE TRIGGER Emp_count AFTER DELETE ON Emp_tab
FOR EACH ROW
DECLARE
    nro INTEGER;
BEGIN
    SELECT COUNT(*) INTO nro FROM Emp_tab; DBMS_OUTPUT.PUT_LINE('
    There are now ' || nro || ' employees.');
```

END;

```
DELETE FROM Emp_tab WHERE Empno = 7499;
```

➔ ORA-04091: table SCOTT.Emp_tab is mutating, trigger/function may not see it



Exemplo

Um recurso que permite contornar parte das limitações de tabelas mutantes para operações de update/insert é a edição do conteúdo da variável :new, em timing BEFORE.

```
CREATE OR REPLACE TRIGGER AcertaNota
BEFORE INSERT OR UPDATE ON Matricula
FOR EACH ROW
BEGIN
    --UPDATE Matricula set nota = 5
    /*erro de tabela mutante*/
    --WHERE Sigla = :new.sigla;
    :new.nota := 5.0; /*Ok*/
EXCEPTION
    . . . . .
END AcertaNota;
```

Exemplo

```
CREATE OR REPLACE TRIGGER AcertaNota
BEFORE INSERT OR UPDATE ON Matricula
/*especificando nomes para NEW e OLD ...*/
REFERENCING new AS nova_matricula
FOR EACH ROW
WHEN (nova_matricula.nota < 0)

BEGIN
    :nova_matricula.nota := 0;
END AcertaNota;
```

Exemplo - criando log

```
CREATE OR REPLACE TRIGGER LogDisciplina
AFTER INSERT OR UPDATE OR DELETE ON Disciplina
FOR EACH ROW

DECLARE
    v_operacao CHAR;

BEGIN
    /*usando predicados booleanos...*/
    IF INSERTING THEN v_operacao := 'I';
        ELIF UPDATING THEN v_operacao := 'U';
        ELIF DELETING THEN v_operacao := 'D';
    END IF;

    INSERT INTO logTabelaDisciplina
        VALUES (USER, SYSDATE, v_operacao);

END LogDisciplina;
```

Triggers de Sistema

- *Triggers* disparados por:
 - instruções DDL
 - CREATE – before/after
 - ALTER - before/after
 - DROP - before/after
 - DDL - before/after
 - ...
 - eventos do banco de dados
 - STARTUP - after
 - SHUTDOWN - before
 - LOGON - after
 - LOGOFF - before
 - SERVERERROR – after
 -:
- Níveis
 - **DATABASE**
 - **SCHEMA**
 - do usuário que criou o *trigger* ou de outro usuário



Triggers de Sistema

- *Triggers* disparados por:
 - instruções DDL
 - CREATE – before/after
 - ALTER - before/after
 - DROP - before/after
 - DDL - before/after
 - ...
 - eventos do banco de dados
 - STARTUP - after
 - SHUTDOWN - before
 - LOGON - after
 - LOGOFF - before
 - SERVERERROR – after
 - ...
- Níveis
 - **DATABASE**
 - **SCHEMA**
 - do usuário que criou o *trigger* ou de outro usuário

Dados disponíveis:

- sysdate
- sys_context('USERENV','OS_USER')
- sys_context('USERENV','CURRENT_USER')
- sys_context('USERENV','HOST')
- sys_context('USERENV','TERMINAL')
- ora_dict_obj_owner
- ora_dict_obj_type
- ora_dict_obj_name
- ora_sysevent



Triggers de Sistema

EXEMPLOS

```
-- conectado com role labbd
```

```
CREATE OR REPLACE TRIGGER TodosUsuarios
```

```
AFTER LOGON ON DATABASE
```

```
BEGIN
```

```
    INSERT INTO logUser VALUES (USER, 'Trigger  
    TodosUsuarios');
```

```
END;
```

```
-- conectado com role labbd
```

```
CREATE OR REPLACE TRIGGER UsuarioLogado
```

```
AFTER LOGON ON SCHEMA
```

```
BEGIN
```

```
    INSERT INTO logUser VALUES (USER, 'Trigger  
    UsuarioLogado');
```

```
END;
```



Triggers Instead-of

- Usados para alterar (DML) visões não atualizáveis e visões de junção atualizáveis
 - permitem fazer as atualizações de maneira adequada para a **semântica da aplicação**
- Executa o corpo do trigger **AO INVÉS** (*instead-of*) da instrução que o acionou



Exemplo

Professor = {Nome, NFunc, Idade, Titulação}

Disciplina = {Sigla, Nome, NCred, Professor, Livro}

```
CREATE VIEW prof_disciplina AS
  SELECT d.sigla, d.nome, p.nfunc, p.nome as professor
 FROM disciplina d, professor p
 WHERE d.professor = p.nfunc;
```

-- qual o efeito do comando abaixo na base de dados???

```
DELETE FROM prof_disciplina WHERE nfunc = 111;
```



Exemplo

Professor = {Nome, NFunc, Idade, Titulação}

Disciplina = {Sigla, Nome, NCred, Professor, Livro}

```
CREATE VIEW prof_disciplina AS
  SELECT d.sigla, d.nome, p.nfunc, p.nome as professor
  FROM disciplina d, professor p
  WHERE d.professor = p.nfunc;
```

Nesta view, apenas a tabela disciplina terá preservação de chave, portanto a operação de delete sobre um atributo da tabela Professor causará um erro: tentativa de atualização de tabela sem preservação de chave.

-- qual o efeito do comando abaixo na base de dados???

```
DELETE FROM prof_disciplina WHERE nfunc = 111;
```



Exemplo

Professor = {Nome, NFunc, Idade, Titulação}

Disciplina = {Sigla, Nome, NCred, Professor, Livro}

```
CREATE OR REPLACE TRIGGER RemoveProfDisciplina
INSTEAD OF DELETE ON prof_disciplina
FOR EACH ROW /* opcional - sempre é nível de linha */
BEGIN

    UPDATE disciplina SET professor = null
        WHERE professor = :old.nfunc;

    DELETE FROM professor WHERE nfunc = :old.nfunc;

END RemoveProfDisciplina;
```



Procedures X Triggers

Procedure/Function	Trigger
bloco identificado PL/SQL	bloco identificado PL/SQL
pode ser usado em pacotes ou mesmo em triggers	objeto independente
recebe parâmetros	não recebe parâmetros, usa apenas new e old
executado explicitamente	executado (disparado) implicitamente – execução orientada a eventos



Triggers

- Para que usar?
 - Restrições de consistência e validade que não possam ser implementadas com *constraints* - por exemplo, envolvendo múltiplas tabelas
 - Criar conteúdo de uma coluna derivado de outras
 - Atualizar tabelas em função da atualização de uma determinada tabela
 - Criar logs - segurança → auditoria
 - entre outras situações interessantes e possíveis...



Referência de Criação e Apoio ao Estudo

Material para Consulta e Apoio ao Conteúdo

- Programando em Oracle 9i – PL/SQL
 - <http://pt.slideshare.net/79anderson/apostila-completaoracleprogramandooracle>
- Oracle Database 11G SQL: Domine SQL e PL/SQL no banco de dados Oracle
 - Livro
- Oracle Database – Developer's Guide 11g
 - http://docs.oracle.com/cd/E11882_01/appdev.112/e10766.pdf
- Application Developer's Guide – Fundamentals
 - SQL Reference - Database Concepts (usando triggers: informações, exemplos, eventos,...)