



MANIPULAÇÃO DE DADOS

Prof. Rangel Nunes

DATA MANIPULATION LANGUAGE

Comandos DML



INSERINDO DADOS NAS TABELAS DO BANCO DE DADOS

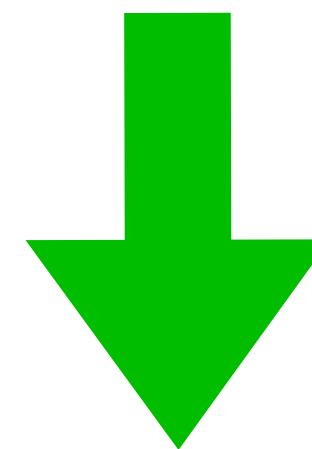
INSERT

Esse comando insere uma linha especificada por expressões de valor ou várias linhas resultantes de uma consulta

</ CODING SQL>

INSERT INTO **TABELA**
[(ATRIBUTOS)] VALUES (VALOR1, VALOR2 ...) |
[RESULTADO DE UMA CONSULTA]

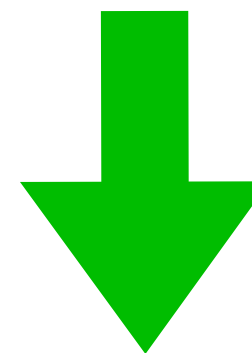
cliente	
PK	<u>cpf</u>
	rg
	nome
	data_nascimento
	ativo



INSERT INTO **CLIENTE**
VALUES (999999999999, 111111, 'RANGEL', '01/01/2000', TRUE)

</ CODING SQL >

cliente	
PK	<u>cpf</u>
	rg nome data_nascimento ativo



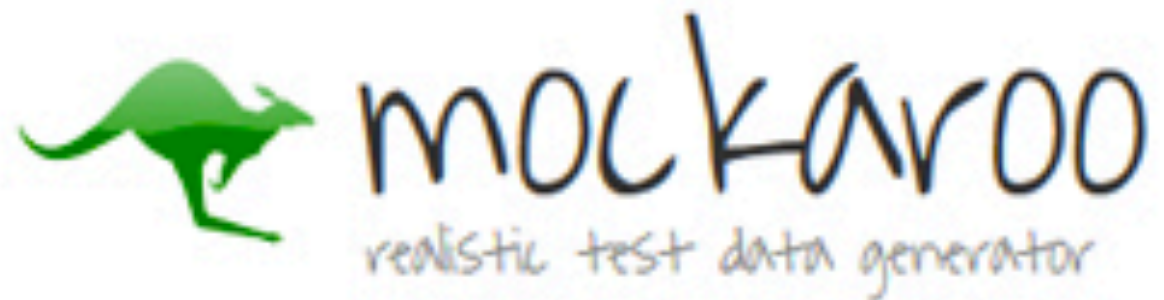
```
INSERT INTO CLIENTE
(CPF, RG, NOME, DATA_NASCIMENTO, ATIVO)
VALUES (888888888888, 222222, 'MARIA', '23/12/2000', FALSE)
```

```
INSERT INTO CLIENTE
VALUES (777777777777, 333333, 'JOAO', '23/12/2000', DEFAULT)
```

CLÁUSULA RETURNING

```
INSERT INTO TABELA(COLUNA1, COLUNA2, ...)  
VALUES (VALOR1, VALOR2, ...)  
RETURNING *;
```

```
INSERT INTO CLIENTE  
VALUES (777777777777, 333333, 'JOAO', '23/12/2000', DEFAULT)  
RETURNING *
```



Field Name	Type	Options
<div><div></div>id</div>	Row Number <div></div>	blank: <input type="text" value="0"/> % <input type="checkbox"/>
<div><div></div>first_name</div>	First Name <div></div>	blank: <input type="text" value="0"/> % <input type="checkbox"/>
<div><div></div>last_name</div>	Last Name <div></div>	blank: <input type="text" value="0"/> % <input type="checkbox"/>
<div><div></div>email</div>	Email Address <div></div>	blank: <input type="text" value="0"/> % <input type="checkbox"/>
<div><div></div></div>	Country <div></div>	blank: <input type="text" value="0"/> % <input type="checkbox"/>

MODIFICANDO DADOS INSERIDOS NAS TABELAS DO BANCO DE DADOS

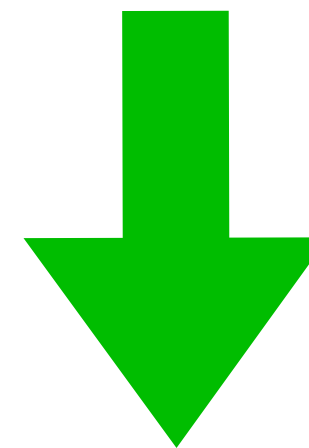
UPDATE

O comando UPDATE muda os valores das colunas especificadas em todas as linhas que satisfazem uma determinada condição

</ CODING SQL>

```
UPDATE TABELA  
SET COLUNA = NOVO_VALOR  
[WHERE CONDIÇÃO]
```

cliente	
PK	<u>cpf</u>
	rg
	nome
	data_nascimento
	ativo



```
UPDATE CLIENTE  
SET DATA_NASCIMENTO = '30/06/2013'  
WHERE CPF = 888888888888
```

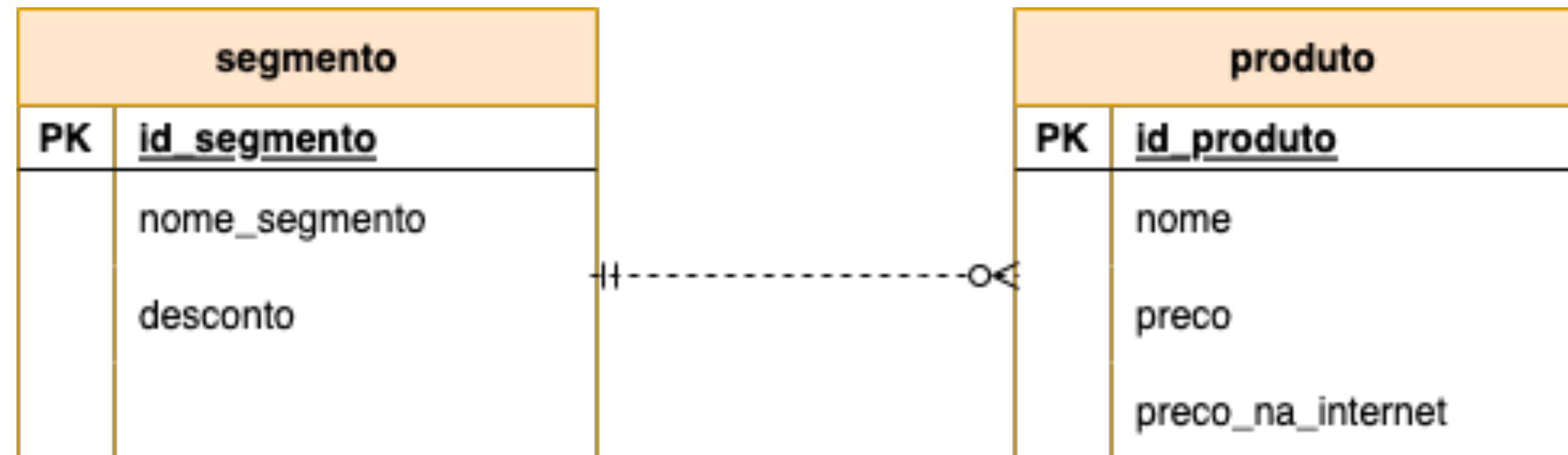
UPDATE JOIN

Atualize os dados de uma tabela, com base nos dados de outra tabela

</ CODING SQL>

```
UPDATE TABELA1  
SET TABELA1.COLUNA1 = NOVO_VALOR  
FROM TABELA2  
WHERE TABELA1.COLUNA2 = TABELA2.COLUNA2;
```

</ CODING SQL >



</ CODING SQL>

```
CREATE TABLE SEGMENTO(  
  ID_SEGMENTO SERIAL PRIMARY KEY,  
  NOME_SEGMENTO VARCHAR(50) NOT NULL,  
  DESCONTO NUMERIC(3,2) NOT NULL  
);
```

```
CREATE TABLE PRODUTO(  
  ID_PRODUTO SERIAL PRIMARY KEY,  
  NOME VARCHAR(50) NOT NULL,  
  PRECO NUMERIC(9,2) NOT NULL,  
  PRECO_NA_INTERNET NUMERIC(9,2) NOT NULL,  
  SEGMENTO_ID INT REFERENCES SEGMENTO(ID_SEGMENTO)  
);
```

</ CODING SQL>

```
UPDATE PRODUTO
SET PRECO_NA_INTERNET = PRODUTO.PRECO -
(PRODUTO.PRECO * SEGMENTO.DESCONTO)
FROM SEGMENTO
WHERE PRODUTO.SEGMENTO_ID = SEGMENTO.ID_SEGMENTO;
```

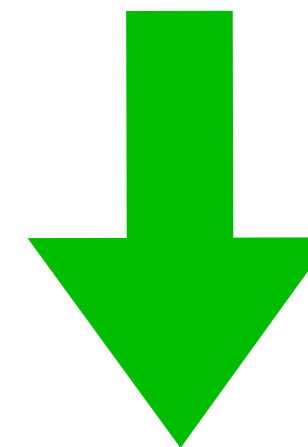
REMOVENDO DADOS DAS TABELAS DO BANCO DE DADOS

DELETE

O comando DELETE exclui da tabela especificada as linhas que satisfazem uma determinada condição

</ CODING SQL>

DELETE FROM TABELA
[WHERE CONDIÇÃO]



cliente	
PK	<u>cpf</u>
	rg
	nome
	data_nascimento
	ativo

DELETE FROM CLIENTE
WHERE CPF = 777777777777

DELETE JOIN

Remova os dados de uma tabela, com base nos dados de outra tabela

</ CODING SQL>

```
DELETE FROM TABELA1  
USING TABELA2  
WHERE CONDIÇÃO
```

</ CODING SQL>

```
CREATE TABLE USUARIO(  
    ID SERIAL PRIMARY KEY,  
    NOME_USUARIO VARCHAR(30) NOT NULL  
);
```

```
CREATE TABLE BLOQUEADOS(  
    NOME_USUARIO VARCHAR(30) PRIMARY KEY  
);
```

</ CODING SQL>

```
DELETE FROM USUARIO  
USING BLOQUEADOS  
WHERE USUARIO.NOME_USUARIO =  
BLOQUEADOS.NOME_USUARIO
```

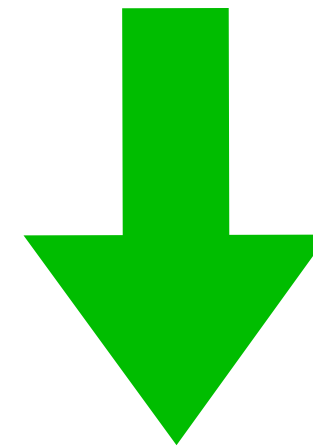
ALÉM DO DELETE...

TRUNCATE

Prove um mecanismo rápido de remover todas as linhas de uma tabela

</ CODING SQL>

TRUNCATE TABLE TABELA



TRUNCATE TABLE CLIENTE

INSERT ... ON CONFLICT

Sintaxe do PostgreSQL

</ CODING SQL>

```
INSERT INTO NOME_TABELA  
(COLUNA1,COLUNA2, ...)  
VALUES (VALUE1, VALUE2, ...)  
ON CONFLICT (COLUNA_DE_CONFLITO)  
DO NOTHING | DO UPDATE SET COLUMN1 = VALUE1;
```

CENÁRIO

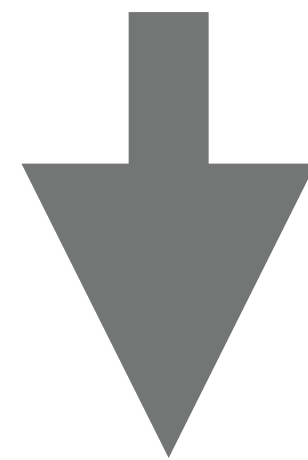
```
CREATE TABLE PRODUTOS(  
  ID_PRODUTO INT PRIMARY KEY,  
  NOME VARCHAR(30) UNIQUE,  
  PRECO_UNITARIO NUMERIC(9,2) NOT NULL,  
  QUANTIDADE INT NOT NULL  
);
```

Ao inserir um produto, tem-se duas opções:

- Se o produto não existir, ele será inserido;
- Se ele já existir, serão atualizados o seu preço e a quantidade em estoque

CENÁRIO

```
INSERT INTO PRODUTOS  
VALUES (1, 'NOTEBOOK', 3000, 10)  
ON CONFLICT(NOME)  
DO UPDATE SET  
    PRECO_UNITARIO = EXCLUDED.PRECO_UNITARIO,  
    QUANTIDADE = EXCLUDED.QUANTIDADE;
```



Você pode usar novos valores ou fazer referência aos valores que tentou inserir usando a palavra-chave **EXCLUDED**