INSTITUTO FEDERAL
Pernambuco

Linguagem de 3ª Geração

Banco de Dados 2

Professora Carolina Torres

INSTITUTO FEDERAL
Pernambuco

Estruturas de Programação

Conceitos apresentados utilizando a sintaxe do Postgres

Estruturas de Programação

INSTITUTO FEDERAL
Pernambuco

- Declaração de

Variável:

```
DECLARE nome tipo;
```

Bloco:

```
BEGIN
    Código_a_ser_executado
END;
```

- Controle de Fluxo:

```
IF condição_IF THEN
    código_IF
ELSIF condição_ELSIF THEN
    código_ELSIF
ELSE
    código_ELSE
END IF;
```

Exemplo:

INSTITUTO FEDERAL
Pernambuco

```
CREATE FUNCTION checar_igual (a1 TEXT,  a2 TEXT)
RETURNS BOOLEAN
LANGUAGE plpgsql AS $$
DECLARE igual BOOLEAN;
BEGIN
        SELECT  ($1 = $2) INTO igual ;
        RETURN igual ;
END;
$$
```

```
select checar_igual (‘teste',‘Teste');
select checar_igual (‘Maria',‘Maria');
```

INSTITUTO FEDERAL
Pernambuco

Subprogramas
Funções e procedimentos
...

Subprogramas

INSTITUTO FEDERAL
Pernambuco

- Procedimentos, funções e pacotes que são armazenados na base de dados
  - Logo ocupam espaço
- Em geral não são alterados depois de construídos e são executados muitas vezes
- São executados explicitamente, através de uma chamada a eles

Esquema Adotado

INSTITUTO FEDERAL
Pernambuco

```
create table setor(
  cod_setor integer primary key,
  nome_setor varchar (30) );

create table produto(
  cod_produto integer primary key,
  nome_produto varchar(30),
  preço decimal (8,2),
  categoria varchar(30),
  cd_setor integer,
  foreign key (cd_setor) references
setor (cod_setor) );
```

```
create table fornecedor (
  cod_fornecedor integer primary key,
  nome_fornecedor varchar(30),
  cidade varchar(30) );

create table forn_Prod(
  cd_fornecedor integer,
  cd_produto integer,
 primary key (cd_fornecedor,cd_produto),
 foreign key (cd_fornecedor) references
fornecedor (cod_fornecedor),
 foreign key (cd_produto) references
produto (cod_produto) );
```

INSTITUTO FEDERAL
Pernambuco

Procedures

...

Procedure
- Sintaxe de definição -

INSTITUTO FEDERAL
Pernambuco

```
CREATE [OR REPLACE] PROCEDURE nome_do_proced (argumentos)
LANGUAGE <SQL|plpgsql> AS $$
DECLARE
    -- variable declaration
BEGIN
    -- logic
END; $$
```

Procedure
- Acesso -

INSTITUTO FEDERAL
Pernambuco

CALL *nome_do_procedimento* (<*lista_de_parâmetros*>);

Procedure
- Operações -

INSTITUTO FEDERAL
Pernambuco

- Alterar procedimentos:
  - ALTER PROCEDURE ***nome_da_procedure*** (tipos dos params) RENAME TO **novo_nome**;
- Deletar procedimentos :
  - DROP PROCEDURE ***nome_da_procedure()***

Procedure
- Exemplos -

INSTITUTO FEDERAL
Pernambuco

- Criar um procedimento ***inserir_relacao*** que insira os dados na tabela de produto e fornecedor;
- Executar o procedimento criado com os código 5 para produto e 3 para fornecedor;

```
CREATE PROCEDURE inserir_relacao
(cod_f integer, cod_p integer)
LANGUAGE SQL
AS $$
insert into forn_prod values (cod_f,cod_p);
$$;
```

```
CALL inserir_relacao (3, 5);
```

Procedure
- Exemplos -

INSTITUTO FEDERAL
Pernambuco

- Evoluir procedimento ***inserir_relacao*** verificando se os valores de FK existem nas tabelas de produto e fornecedor e se o par já foi utilizado como PK;

```
CREATE OR REPLACE PROCEDURE inserir_relacao (cod_f integer, cod_p integer)
LANGUAGE PLPGSQL AS $$
DECLARE
existe TEXT;
BEGIN
  select 'true' from forn_prod
            where cd_fornecedor=cod_f and cd_produto=cod_p Into existe;
   if existe is null then
      insert into forn_prod values (cod_f,cod_p);
   end if;
END;$$;
```

Funções

...

Function
- Sintaxe de definição -

INSTITUTO FEDERAL
Pernambuco

```
CREATE [OR REPLACE] FUNCTION nome_da_função (argumentos)
RETURNS retorno
LANGUAGE <SQL | plpgsql> AS $$
DECLARE
    -- variable declaration
BEGIN
    -- logic
END; $$
```

- Lista de argumentos:
  - “nome tipo” separados por vírgula
- Retorno:
  - Tipos comuns (INTEGER, VARCHAR(40), ...)
  - Tipos criados (CREATE TYPE...)
  - Estruturas (especificar colunas)
  - TABLE (especificar colunas)

Function
- Sintaxe de Acesso -

INSTITUTO FEDERAL
Pernambuco

- Se retornar um tipo comum

```
SELECT <nome_função>(argumentos);
```

- Se retornar outra opção

```
SELECT * FROM
<nome_função>(argumentos);
```

Function
- Operações -

INSTITUTO FEDERAL
Pernambuco

- Alterar funções:
  - ALTER FUNCTION <nome da função> RENAME <novo_nome>
- Deletar funções:
  - DROP FUNCTION <nome da função>

Function
- Exemplos -

INSTITUTO FEDERAL
Pernambuco

```
CREATE FUNCTION add(x integer, y integer)
RETURNS integer
LANGUAGE SQL AS
    'select x + y;'
;
```

```
CREATE FUNCTION increment(i integer)
RETURNS integer
LANGUAGE plpgsql AS $$
BEGIN
    RETURN i + 1;
END;
$$;
```

Function
- Exemplos -

INSTITUTO FEDERAL
Pernambuco

```
CREATE FUNCTION dup(in int, out f1 int, out f2 text)
LANGUAGE SQL AS $$
    SELECT $1, CAST($1 AS text) || ' is text'
$$;

SELECT * FROM dup(42);
```

```
CREATE TYPE dup_result AS (f1 int, f2 text);

CREATE FUNCTION dup(f1 int)
RETURNS dup_result
LANGUAGE SQL AS $$
   SELECT f1, CAST(f1 AS text) || ' is text'
$$;

SELECT * FROM dup(42);
```

```
CREATE FUNCTION dup(int)
RETURNS TABLE(f1 int, f2 text)
LANGUAGE SQL AS $$
   SELECT $1, CAST($1 AS text) || ' is text'
$$;

SELECT * FROM dup(42);
```

Exercícios

INSTITUTO FEDERAL
Pernambuco

1. Criar a estrutura **altera_nome_produto** para atualizar o nome de produto com um determinado código.
   - Deve receber como parâmetros o novo nome e o código do produto a ter seu nome alterado;
2. Altere o nome do produto de código 2 para ‘Creta’;
3. Delete a estrutura **altera_nome_produto**
4. Criar a estrutura **insere_relação** que recebe dois inteiros, referentes aos códigos de um produto e de um fornecedor. Deve-se inserir essa dupla na tabela forn_prod e retornar uma string de sucesso, mas antes deve verificar a existência dos código passados realmente existem ou se a dupla já tem relação; nesses últimos casos, não deve realizar a inserção e deve retornar uma string contendo a descrição do erro.
5. Criar a estrutura **cadastro_completo** que receba informações de um produto (nome, preço e categoria) além do nome do setor ao qual pertence e do fornecedor que oferece esse produto. O procedimento deve cadastrar todas essas informações.
6. Execute **cadastro_completo** para inserir um novo produto.
7. Crie uma estrutura que receba 1 argumento inteiro e que verifica (retorna True ou False) se é maior ou igual a 18;
8. Crie uma estrutura que receba 2 argumentos (um nome e uma idade) e identifica se a pessoa é de maior ou de menor.
   - Deve usar/chamar a estrutura criada no item 7.

INSTITUTO FEDERAL
Pernambuco

Linguagem de 3ª Geração

Banco de Dados 2

Professora Carolina Torres