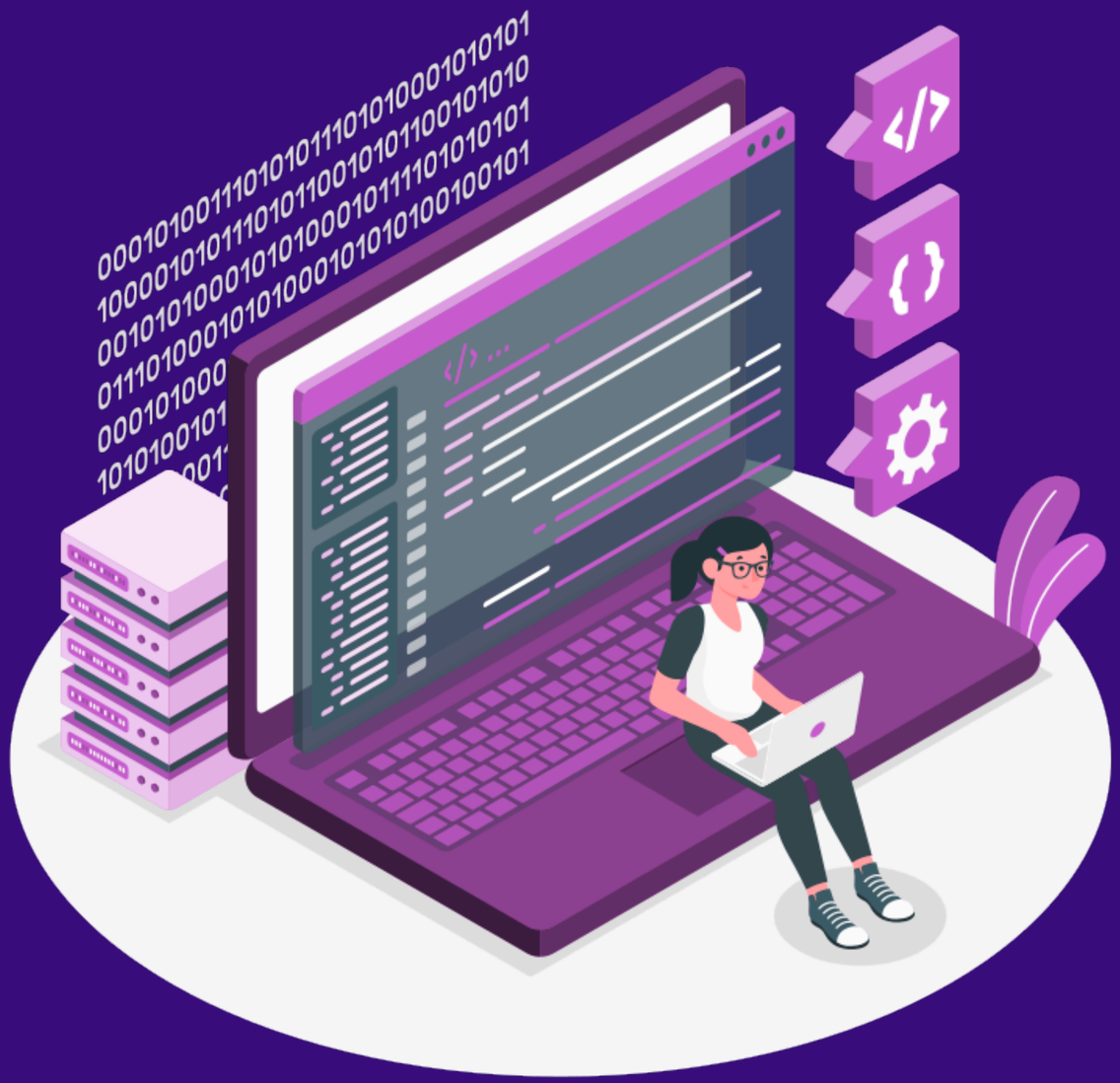
PROCEDIMENTOS ARMAZENADOS

ESTRUTURAS DE REPETIÇÃO



LOOP

Define um *loop* incondicional que executa um bloco de código repetidamente até ser encerrado por uma declaração de saída ou retorno

```
loop
    -- lógica
end loop;
```

LOOP (CONT.)

Normalmente, você usa uma instrução ***if*** dentro do ***loop*** para finalizá-lo, com base em uma condição

```
loop
    -- lógica;

    if condição then
        exit;
    end if;
end loop;
```

LOOP - EXEMPLO

```
create or replace procedure imprime_ate_cinco( ) as
$$
declare
    n integer;
begin
    n := 1;
    loop
        if n <= 5 then
            raise notice 'Numero: %', n;
            n := n + 1;
        else
            exit;
        end if;
    end loop;
end;
$$
language plpgsql;
```

LOOP - COM WHEN

Quando *WHEN* está presente, a saída do laço ocorre somente se a condição especificada for verdadeira

```
loop
    -- lógica;

    exit when <<condição>>
end loop;
```

LOOP - EXEMPLO COM WHEN

```
create or replace procedure imprime_ate_cinco( ) as
$$
declare
    n integer;
begin
    n := 1;
    loop
        exit when n > 5;
        raise notice 'Numero: %', n;
        n := n + 1;
    end loop;
end;
$$
language plpgsql;
```

WHILE LOOP

- A instrução ***WHILE*** repete uma sequência de instruções desde que a expressão *booleana* seja avaliada como verdade
- A expressão é verificada imediatamente antes de cada entrada para o corpo do laço

```
while <<condição>> loop
    -- lógica;
end loop;
```

WHILE LOOP - EXEMPLO

```
create or replace procedure imprime_ate_cinco_com_while()
as
$$
    declare
        n integer;
    begin
        n := 1;
        while n <= 5 loop
            raise notice 'numero: %', n;
            n := n + 1;
        end loop;

    end;
$$
language plpgsql;
```

FOR LOOP

```
for contador in [ reverse ] início .. fim  loop
    -- lógica;
end loop;
```

FOR LOOP - EXEMPLO

```
create or replace procedure imprime_ate_cinco_com_for() as
$$
    begin
        for contador in 1..5 loop
            raise notice 'numero: %', contador;
        end loop;
    end;

$$
language plpgsql;
```

FOR LOOP - EXEMPLO COM BY

```
create or replace procedure imprime_impares_entre_1_e_5() as
$$
    begin
        for contador in 1..5 by 2 loop
            raise notice 'numero: %', contador;
        end loop;
    end;

$$
language plpgsql;
```

FOR LOOP - EXEMPLO COM REVERSE

```
create or replace function inverte_url(url varchar) returns varchar as
$$
    declare
        nova_url varchar := '';
    begin
        for caractere in reverse length(url) .. 1 loop
            nova_url := nova_url || substr(url,caractere,1);
        end loop;
        return nova_url;
    end;
$$
language plpgsql;
```

USANDO FOR PARA ITERAR UM CONJUNTO DE RESULTADOS

```
for linha in consulta loop
    -- lógica;
end loop;
```

FOR - EXEMPLO QUERY

```
create or replace procedure consulta_alunos() as
$$
    declare
        registro record;
    begin
        for registro in select * from alunos loop
            raise notice 'id: % - nome: %', registro.id_aluno, registro.nome;
        end loop;
    end;
$$
language plpgsql;
```

FOR - EXEMPLO QUERY SETOF ALUNOS

```
create or replace function fn_consulta_alunos( ) returns setof alunos as
$$
    declare
        registro record;
    begin
        for registro in select * from alunos loop
            raise notice 'id: % - nome: %', registro.id_aluno, registro.nome;
            return next registro;
        end loop;
        return;
    end;
$$
language plpgsql;
```

FOR - EXEMPLO QUERY SETOF RECORD

```
create or replace function fn_consulta_alunos_record() returns setof record as
$$
    declare
        registro record;
    begin
        for registro in select * from alunos loop
            raise notice 'id: % - nome: %', registro.id_aluno, registro.nome;
            return next registro;
        end loop;
        return;
    end;
$$
language plpgsql;
```

HORA DE PRATICAR!

Implemente uma função ou procedimento que some as horas trabalhadas de cada projeto e atualize a duração deles(em dias).

https://drive.google.com/file/d/1khOpxcGYu2FyfefScYWe0wZX_wLzshOS/view?usp=sharing

INSTITUTO FEDERAL
Piauí

PAPO DE PROGRAMADOR