Funciones y procedimientos

Indice

- PL/pgSQL
- Funciones
- Procedimientos

PL/pgSQL documentación

- **Visión general PL/pgSQL:** https://www.postgresql.org/docs/9.4/plpgsql-overview.html#PLPGSQL-ADVANTAGES
- **Tutorial:** https://www.postgresqltutorial.com/postgresql-create-function/
- **Declaraciones:** https://www.postgresql.org/docs/9.4/plpgsql-declarations.html
- **Sentencias básicas:** https://www.postgresql.org/docs/current/plpgsql-statements.html
- **Estructuras de control:** https://www.postgresql.org/docs/current/plpgsql-control-structures.html

PL/pgSQL

- Lenguaje SQL procedimental proporcionado por PostgreSQL.
- Permite crear funciones, procedimientos y triggers.
- Nombres de funciones/procedimientos son case-insensitive.
- Incluye sentencias de control tales como if, while, repeat, for.
- Cada sentencia SQL es realizada de manera individual por el servidor de la base de datos.
- La lógica de la base de datos reside en la parte del servidor y no en la parte del cliente.
- La aplicación cliente se puede escribir en diferentes lenguajes de programación.

Definir una Función

```
create [or replace] function nombre_función (lista_parámetros)
      returns tipo_retorno
      language plpgsql
    as
$$
declare
   <declaración de variables>
begin
    <sentencias>
end;
$$;
```

Ejecutar: **select** nombre_función (lista_parámetros);

Nota: cuando creamos nuestras propias funciones/procedimientos es mejor escribir **"create or replace" en lugar de "create".**

Definir una función

```
create [or replace] function nombre_función (lista_parámetros)
      returns tipo_retorno
as
$$
declare
   <declaración de variables>
begin
    <sentencias>
end;
$$  language plpgsql;
```

Ejecutar: **select** nombre_función(lista_parámetros);

Función - Ejemplo (I)

- **Dado el nombre de un estudiante, se pide retornar su id.**
- **'return' [argumento] se usa para retornar de una función/procedimiento.**

```
create function getEstudianteId(nombreEstudiante text) returns int as
$$
declare
        estudianteId int;
begin
        select estudiante_id into estudianteId from estudiante where
                estudiante.estudiante_nombre = nombreEstudiante;
        return estudianteId;
end;
$$ language plpgsql;
```

- **Presenta el id de un estudiante dado su nombre:**

```
select getEstudianteId('Juan');
```

```
 getestudianteid
-----------------
               5
(1 fila)
```

Función - Ejemplo (II)

- **Dado el nombre de un estudiante, se pide retornar su id.**

```
create function getEstudianteId(nombreEstudiante text) returns int as
$$
declare
        estudianteId int;
begin
        select estudiante_id into estudianteId from estudiante where
                estudiante.estudiante_nombre = nombreEstudiante;
        return estudianteId;
end;
$$ language plpgsql;
```

- **Presenta el id de todos los estudiantes:**

```
select estudiante_id from estudiante est
where getEstudianteId(est.estudiante_nombre) > 0;
```

```
 estudiante_id
---------------
             1
             2
             3
             4
             5
             6
             7
(7 filas)
```

Función – Funciones de tabla

- **Devolver una tabla como resultado.**

- **Presentar el id y el nombre de todos los profesores que pertenecen a un departamento específico.**

```
create or replace function getProfesoresDepartamento(id int)
returns table (profesor_id int, profesor_nombre varchar(30)) as $$

begin
  return query select pro.profesor_id, pro.profesor_nombre from profesor pro
                                                  where pro.departamento_id = id;
end;
$$ language plpgsql;
```

```
select getProfesoresDepartamento(2);
```

```
 getprofesoresdepartamento
---------------------------
 (7,"Maria Zulima")
 (8,Virginia)
 (13,Matias)
(3 filas)
```

Parámetros y excepciones

- **Un parámetro es referenciado como $<posición en la lista de parámetros>**

```
create or replace function getProfesoresDepartamentov2(id int)
returns table (profesor_id int, profesor_nombre varchar(30)) as $$

begin
  return query select pro.profesor_id, pro.profesor_nombre from profesor pro
                                                  where pro.departamento_id = id;
   if not found
      then raise exception 'No hay ningun profesor en el departamento %', $1;
   end if;
end;
$$ language plpgsql;
```

- **Presenta el id y el nombre de todos los profesores que pertenezcan a un departamento específico.**

```
select getProfesoresDepartamentov2(4);
```

```
ERROR:  No hay ningun profesor en el departamento 4
CONTEXTO:  función PL/pgSQL getprofesoresdepartamentov2(integer) en la línea 7 en RAISE
```

Alias

- **Usar un nombre distinto para referirse a una variable. Dentro del bloque Declare.**

```
create or replace function getProfesoresDepartamentov3(id int)
returns table (profesor_id int, profesor_nombre varchar(30)) as $$
declare id_aux alias for $1;
begin
  return query select pro.profesor_id, pro.profesor_nombre from profesor pro
                                                    where pro.departamento_id = id;
   if not found
     then raise exception 'No hay ningún profesor en el departamento %',  id_aux;
   end if;
end;
$$ language plpgsql;
```

- **Presenta el id y el nombre de todos los profesores que pertenezcan a un departamento específico.**

```
select getProfesoresDepartamentov3(4);
```

```
ERROR:  No hay ning£n profesor en el departamento 4
CONTEXTO:  función PL/pgSQL getprofesoresdepartamentov3(integer) en la línea 7 en RAISE
```

Función – Ejemplo de consultas con for

[etiqueta]
for target **in** consulta **loop**
sentencias
end loop [etiqueta];

- **Presenta el número de profesores de un departamento específico.**

```
create function getNumeroProfesoresDepartamento(id int) returns integer as
$$
declare  r record; n integer default 0;
begin
  for r in
      select * from profesor pro where pro.departamento_id = id
  loop
          n = n+1;
  end loop;
  return n;
end;
$$ language plpgsql;
```

```
select getNumeroProfesoresDepartamento(1);
 getnumeroprofesoresdepartamento
---------------------------------
                               9
(1 fila)
```

Función – Ejemplo de resultados de consultas con for e if

[etiqueta]
for target **in** consulta **loop**
sentencias
end loop [etiqueta];

if *booleana-expresión* **then** *sentencias*
[elseif *booleana-expr* **then** *sentencias*
[elseif *booleana-expr* **then** *sentencias …]]*
[else *sentencias]* **end if;**

```
create function getNumeroProfesoresDepartamentov2(id int) returns integer as $$
declare  r record; n integer default 0;
begin
  for r in
      select * from profesor pro
  loop
        if r.departamento_id = id
        then n = n+1;
        end if;
  end loop;
  return n;
end;
$$ language plpgsql;
```

- **Presenta el número de profesores de un departamento específico**

select getNumeroProfesoresDepartamentov2(1);

```
 getnumeroprofesoresdepartamentov2
-----------------------------------
                                 9
(1 fila)
```

Procedimiento - Ejemplo (I)

- **create [or replace] procedure …**
- **call** <nombre procedimiento>(<argumentos>)
- **inout** <nombre variable> -> para retornar un valor

- **Presenta el número de profesores de un departamento específico.**

```
create procedure getNumeroProfesoresDepartamentov3(id int,
                                                   inout total_profesores int) as $$
begin
  select count(*) into total_profesores from profesor pro
                                          where pro.departamento_id = id;
end;
$$ language plpgsql;
```

```
call getNumeroProfesoresDepartamentov3(1, 0);
```

```
 total_profesores
------------------
                9
(1 fila)
```

Procedimiento - Ejemplo (II)

- **create [or replace] procedure …**
- **call** <nombre procedimiento>(<argumentos>)
- **inout** <nombre variable> -> para retornar un valor

- **Presenta el número de profesores de un departamento específico.**

```
create procedure getNumeroProfesoresDepartamentov4(id int,
                                                   inout total_profesores int) as $$
begin
  select count(*) into total_profesores t from profesor pro
                                          where pro.departamento_id = id;
end;
$$ language plpgsql;
```

```
do $$
declare result int;
begin
   call getNumeroProfesoresDepartamentov4(1, result);
   raise notice 'result = %', result;
end;
$$;
```

```
NOTICE:  result = 9
DO
```

Procedimiento – Ejemplo while

[etiqueta]
while condicion **loop**
sentencias
end loop;

- **Presenta los 'n' primeros profesores que pertenezcan a un departamento específico.**

```
create procedure getProfesoresDepartamento(id int, n int) as $$
declare  r record; i integer default 0;
begin
  while i <> n loop
      i = i + 1;
      select * into r from profesor pro where pro.departamento_id = id and
                                              pro.profesor_id = i;
      raise notice '(%, %, %)', r.profesor_id, r.profesor_nombre, r.profesor_apellidos;
  end loop;
end;
$$ language plpgsql;
```

```
call getProfesoresDepartamento(1, 2);
```

```
NOTICE:  (1, Jorge, Diez Pelaez)
NOTICE:  (2, Pedro, Hernandez Arauzo)
CALL
```

Procedimiento – Ejemplo for

[etiqueta]
for nombre **in** [**reverse**] expr **..** expr **[by expr] loop**
sentencias
end loop;

- **Presenta los 'n' primeros profesores que pertenezcan a un departamento específico.**

```
create procedure getProfesoresDepartamentov2(id int, n int) as $$
declare  r record; i integer default 0;
begin
  for i in 1..n loop
      select * into r from profesor pro where pro.departamento_id = id and
                                                pro.profesor_id = i;
      raise notice '(%, %, %)', r.profesor_id, r.profesor_nombre, r.profesor_apellidos;
  end loop;
end;
$$ language plpgsql;
```

```
call getProfesoresDepartamentov2(1, 2);
```

```
NOTICE:  (1, Jorge, Diez Pelaez)
NOTICE:  (2, Pedro, Hernandez Arauzo)
CALL
```

Procedimiento – Ejemplo array

```
[etiqueta]
foreach target in array expr loop
        sentencias
end loop [etiqueta];
```

- **Presenta el nombre de un conjunto de departamentos**

```
create procedure getNombreDepartamentos(int []) as $$
declare  r record; i integer default 0;
begin
  foreach i in array $1 loop
      select * into r from departamento dep where dep.departamento_id = i;
      raise notice '(%, %)', r.departamento_id, r.departamento_nombre;
  end loop;
end;
$$ language plpgsql;
```

```
call getNombreDepartamentos(array[1, 2]);
NOTICE:  (1, Ciencias de la Computacion)
NOTICE:  (2, Matematicas)
CALL
```

Funciones y procedimientos