HITO 1 DEL 3º TRIMESTRE DE BASE DE DATOS

Alejandro Cortés Díaz

CampusFP

Diseño de aplicaciones multiplataforma – BASE DE DATOS  14/02/2025

Índice

[Programación con MySQL (5 puntos en total) 2](#_Toc190082688)

[1. Crea una base de datos llamada triángulos. 2](#_Toc190082689)

[2. Crea un procedimiento almacenado dentro de la base de datos triángulos que realice las siguientes acciones: 3](#_Toc190082690)

[a. Crea una tabla llamada triángulo con los campos lado1, lado2 y 3](#_Toc190082691)

[lado3. Si la tabla ya existe, se borrará y se volverá a crear de nuevo. 3](#_Toc190082692)

[b. Añade 20 filas a la tabla triangulo con valores al azar entre 1 y 5 para cada uno de los lados. 3](#_Toc190082693)

[3. Crea una función PL/SQL que reciba tres números enteros (los tres lados del triángulo) y retorna una cadena indicando si el triángulo es Equilátero, Isósceles o Escaleno. 5](#_Toc190082694)

[4. Crea una función PL/SQL que reciba tres números enteros (los tres lados del triángulo) y retorne el perímetro de dicho triángulo (suma de los lados). 7](#_Toc190082695)

[5. Utiliza las dos funciones anteriores en una sentencia SELECT para obtener una relación de triángulos con el perímetro y el tipo. 8](#_Toc190082696)

[Programación con PostgreSQL (5 puntos total) 9](#_Toc190082697)

[6. Crea una base de datos llamada triángulos. 9](#_Toc190082698)

[7. Crea un procedimiento almacenado dentro de la base de datos triángulos que realice las siguientes acciones: 10](#_Toc190082699)

[a. Crea una tabla llamada triángulo con un único campo de tipo array 10](#_Toc190082700)

[donde se almacenarán los tres lados del triángulo. Si la tabla ya 10](#_Toc190082701)

[existe, se borrará y se volverá a crear de nuevo. 10](#_Toc190082702)

[b. Añade 20 filas a la tabla triangulo con valores al azar entre 1 y 5 para 10](#_Toc190082703)

[cada uno de los lados. 10](#_Toc190082704)

[8. Crea una función PL/PGSQL que reciba un array (con los tres lados del triángulo) y retorna una cadena indicando si el triángulo es Equilátero, Isósceles o Escaleno. 13](#_Toc190082705)

[9. Crea una función PL/PGSQL que reciba un array (con los tres lados del triángulo) y retorne el perímetro de dicho triángulo (suma de los lados). 15](#_Toc190082706)

[10. Utiliza las dos funciones anteriores en una sentencia SELECT para obtener una relación de triángulos (cada lado en una columna independiente) con el perímetro y el tipo. 17](#_Toc190082707)

[**Enlace a GitHub** 19](#_Toc190082708)

[**Bibliografía** 20](#_Toc190082709)

# Programación con MySQL (5 puntos en total)

## Crea una base de datos llamada triángulos.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

## Crea un procedimiento almacenado dentro de la base de datos triángulos que realice las siguientes acciones:

### a. Crea una tabla llamada triángulo con los campos lado1, lado2 y

### lado3. Si la tabla ya existe, se borrará y se volverá a crear de nuevo.

### b. Añade 20 filas a la tabla triangulo con valores al azar entre 1 y 5 para cada uno de los lados.

CREATE DEFINER=`root`@`localhost` PROCEDURE `Crear\_Triangulos`()

BEGIN

-- Primero declaro el i utilizaré en el while

declare i int default 0;

-- Dropeo la tabla si existía ya la tabla

drop table if exists triangulo;

-- Crear tabla triangulo con los atributos pertinentes

create table triangulo(

id\_triangulo int auto\_increment primary key,

lado1 int,

lado2 int,

lado3 int

);

-- Bucle while para sacar las 20 filas

while i< 20 do

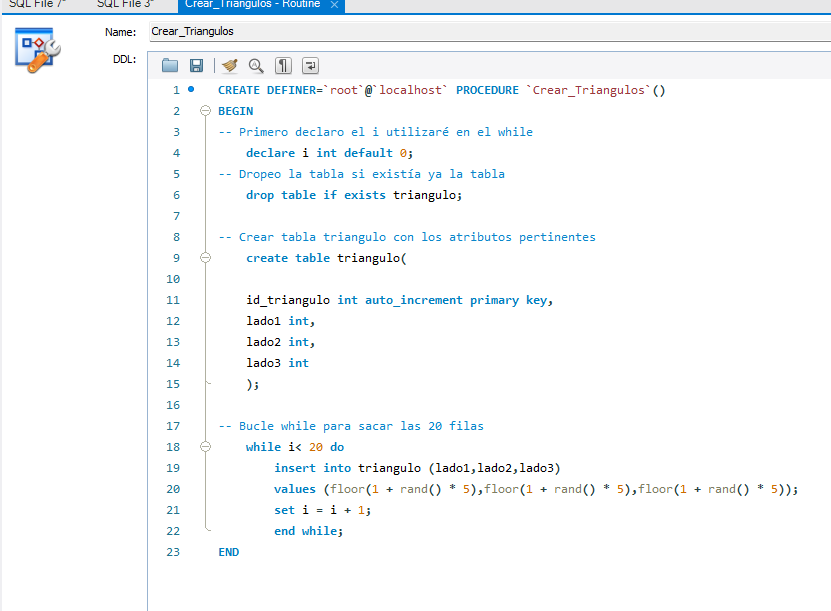
insert into triangulo (lado1,lado2,lado3)

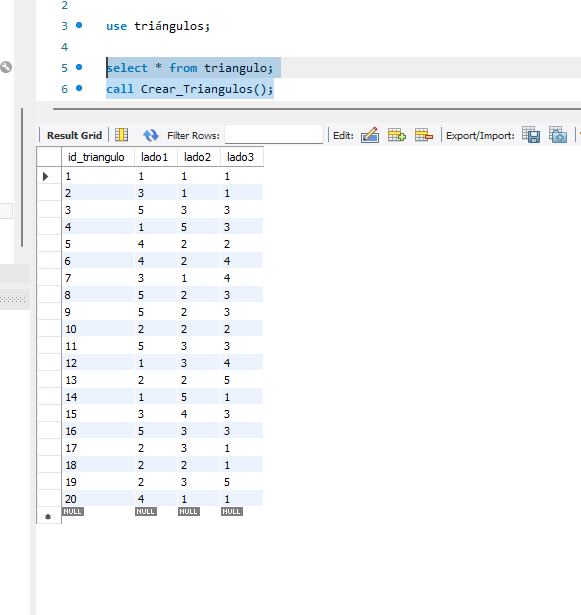
values (floor(1 + rand() \* 5),floor(1 + rand() \* 5),floor(1 + rand() \* 5));

set i = i + 1;

end while;

END





## Crea una función PL/SQL que reciba tres números enteros (los tres lados del triángulo) y retorna una cadena indicando si el triángulo es Equilátero, Isósceles o Escaleno.

* La función que he utilizado;

CREATE DEFINER=`root`@`localhost` FUNCTION `Tipo\_Triangulo`(lado1 int, lado2 int,lado3 int) RETURNS varchar(20) CHARSET utf8mb4

DETERMINISTIC

BEGIN

-- Declaro lo que quiero printear

declare Tipo\_Triangulo varchar(20);

-- Establezco las condiciones para que sea un tipo de triángulo u otro.

if lado1 = lado2 and lado2 = lado3 then

set tipo\_triangulo = "Es equilátero";

elseif lado1 = lado2 or lado1 = lado3 or lado2 = lado3 then

set Tipo\_Triangulo = "Es isósceles";

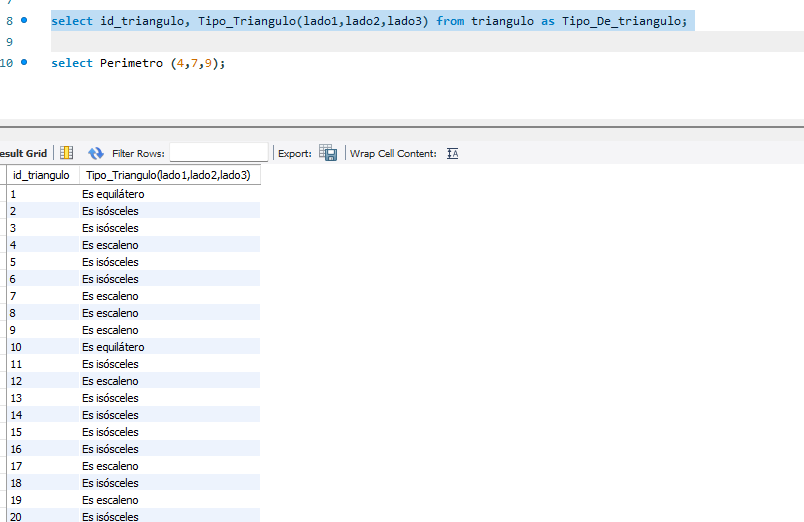
else

set Tipo\_Triangulo = "Es escaleno";

end if;

return Tipo\_Triangulo;

END



## Crea una función PL/SQL que reciba tres números enteros (los tres lados del triángulo) y retorne el perímetro de dicho triángulo (suma de los lados).

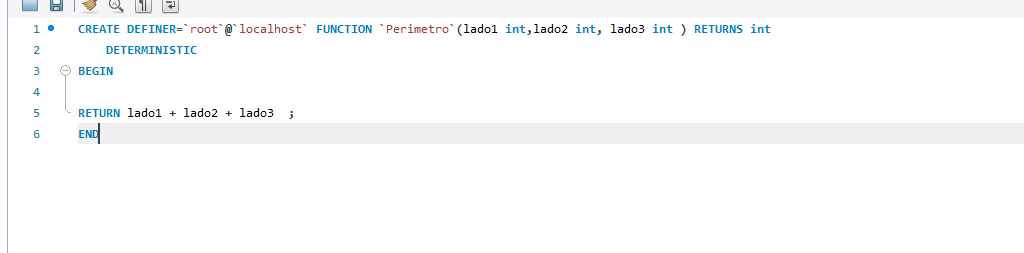
CREATE DEFINER=`root`@`localhost` FUNCTION `Perimetro`(lado1 int,lado2 int, lado3 int ) RETURNS int

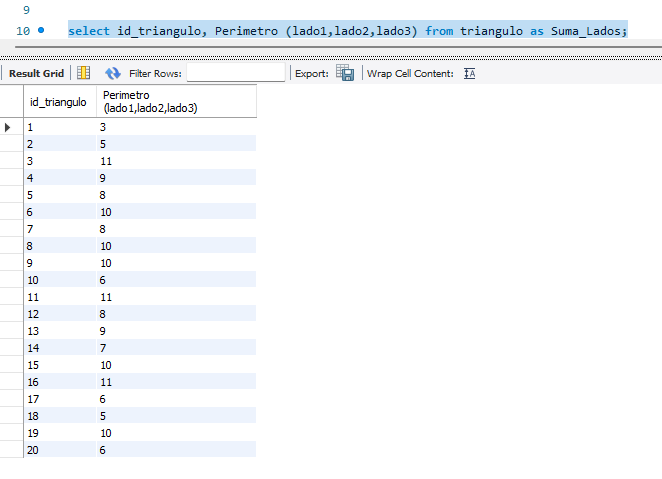
DETERMINISTIC

BEGIN

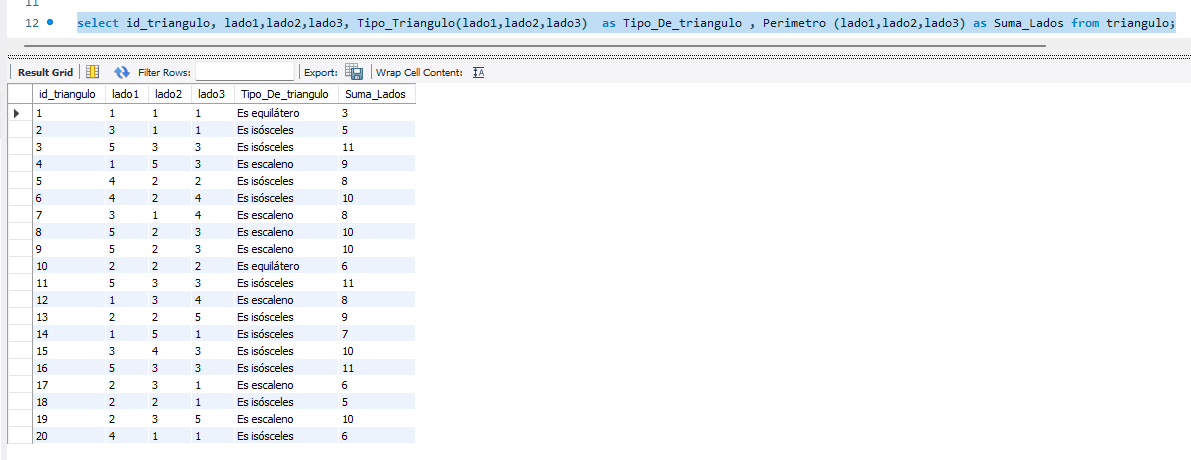
RETURN lado1 + lado2 + lado3 ;

END





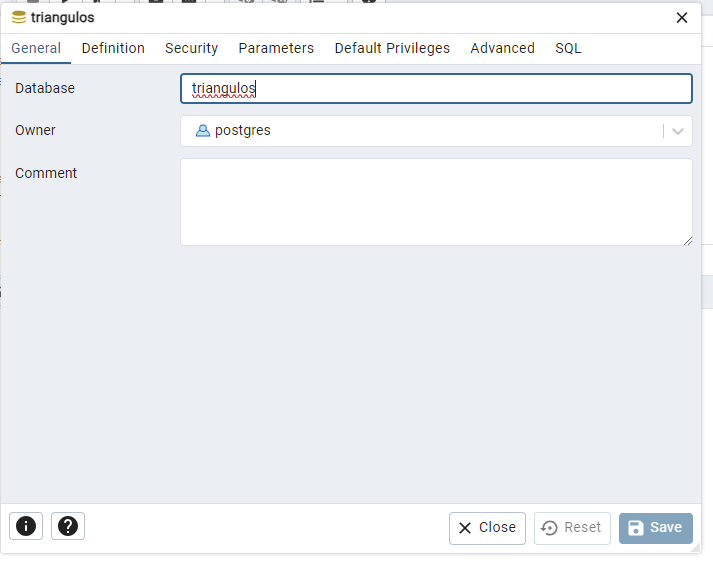
## 5. Utiliza las dos funciones anteriores en una sentencia SELECT para obtener una relación de triángulos con el perímetro y el tipo.



# 

# Programación con PostgreSQL (5 puntos total)

## Crea una base de datos llamada triángulos.



## Crea un procedimiento almacenado dentro de la base de datos triángulos que realice las siguientes acciones:

### a. Crea una tabla llamada triángulo con un único campo de tipo array

### donde se almacenarán los tres lados del triángulo. Si la tabla ya

### existe, se borrará y se volverá a crear de nuevo.

### b. Añade 20 filas a la tabla triangulo con valores al azar entre 1 y 5 para

### cada uno de los lados.

-- Procedimiento base para crear la tabla y generar las 20 filas

CREATE OR REPLACE FUNCTION Crear\_Triangulos()

RETURNS void AS $$

BEGIN

-- Eliminar la tabla si ya existe

DROP TABLE IF EXISTS triangulo;

-- Crear la tabla triangulo con un campo de tipo array

CREATE TABLE triangulo (

id\_triangulo SERIAL PRIMARY KEY,

lados INT[]

);

-- Insertar 20 filas con valores aleatorios entre 1 y 5 para cada lado

FOR i IN 1..20 LOOP

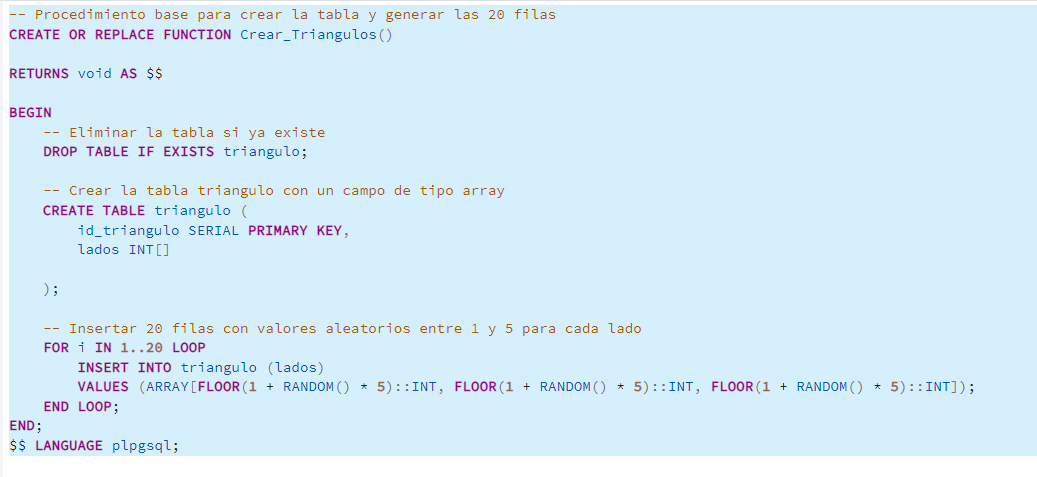
INSERT INTO triangulo (lados)

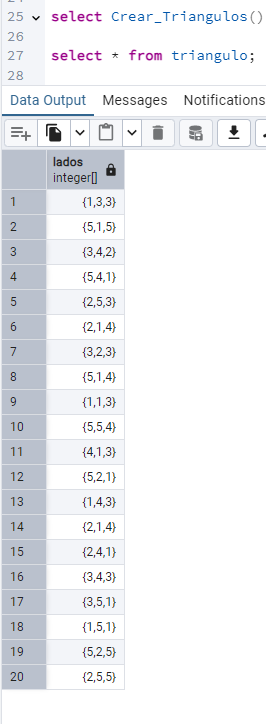
VALUES (ARRAY[FLOOR(1 + RANDOM() \* 5)::INT, FLOOR(1 + RANDOM() \* 5)::INT, FLOOR(1 + RANDOM() \* 5)::INT]);

END LOOP;

END;

$$ LANGUAGE plpgsql;





## Crea una función PL/PGSQL que reciba un array (con los tres lados del triángulo) y retorna una cadena indicando si el triángulo es Equilátero, Isósceles o Escaleno.

-- Función para determinar el tipo de triángulo

CREATE OR REPLACE FUNCTION Tipo\_Triangulo(lados INT[])

-- Que devuelva un varchar de 20

RETURNS VARCHAR(20) AS $$

-- Declaro el nombre de lo que deseo que devuelva

DECLARE

Tipo\_Triangulo VARCHAR(20);

-- Establezco las condiciones pertinentes para que funcione correctamente

BEGIN

IF lados[1] = lados[2] AND lados[2] = lados[3] THEN

Tipo\_Triangulo := 'Es equilátero';

ELSIF lados[1] = lados[2] OR lados[1] = lados[3] OR lados[2] = lados[3] THEN

Tipo\_Triangulo := 'Es isósceles';

ELSE

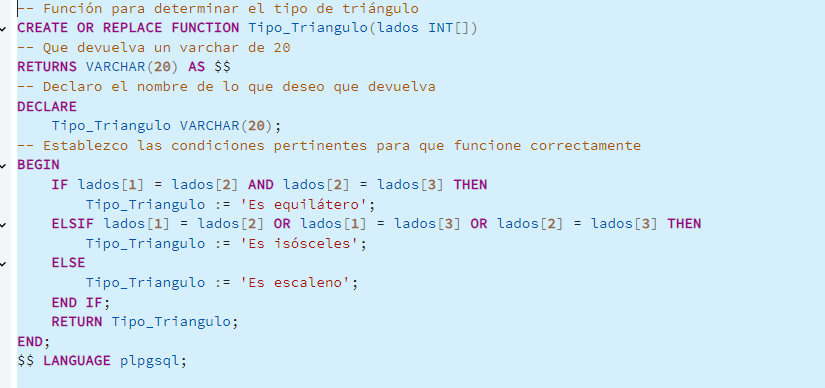
Tipo\_Triangulo := 'Es escaleno';

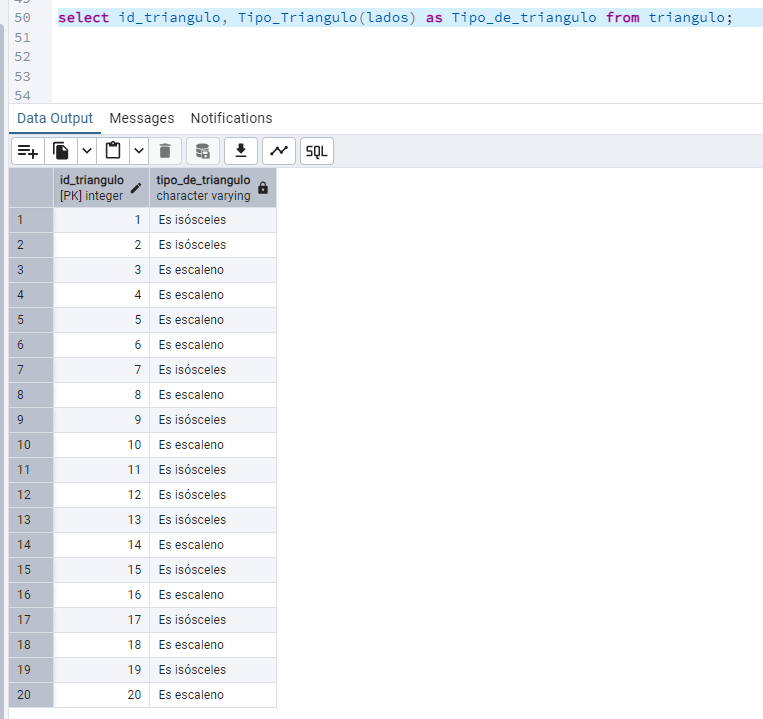
END IF;

RETURN Tipo\_Triangulo;

END;

$$ LANGUAGE plpgsql;





## Crea una función PL/PGSQL que reciba un array (con los tres lados del triángulo) y retorne el perímetro de dicho triángulo (suma de los lados).

CREATE OR REPLACE FUNCTION Perimetro(lados INT[])

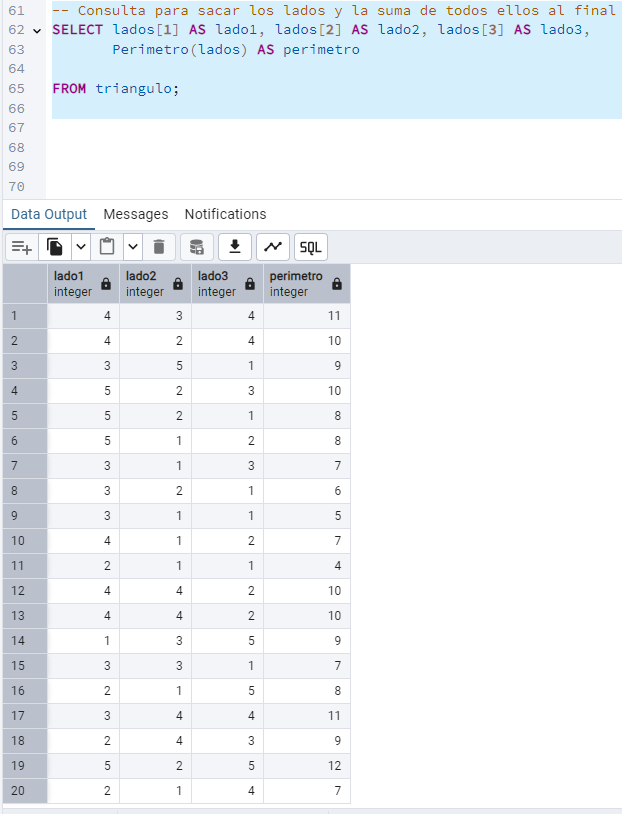
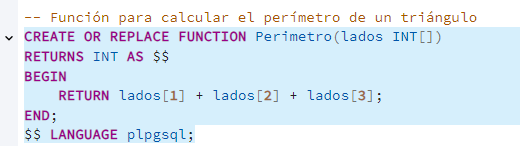
RETURNS INT AS $$

BEGIN

RETURN lados[1] + lados[2] + lados[3];

END;

$$ LANGUAGE plpgsql;



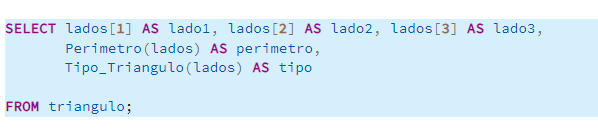
## Utiliza las dos funciones anteriores en una sentencia SELECT para obtener una relación de triángulos (cada lado en una columna independiente) con el perímetro y el tipo.

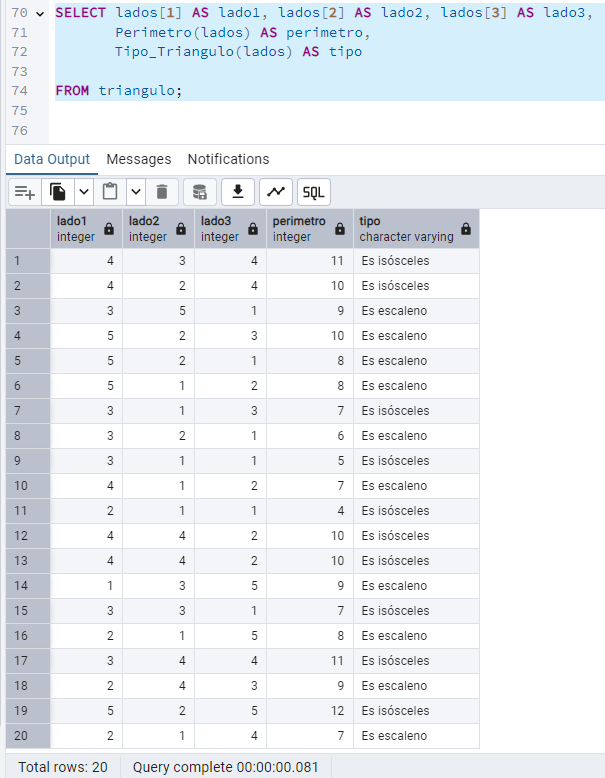
SELECT lados[1] AS lado1, lados[2] AS lado2, lados[3] AS lado3,

Perimetro(lados) AS perimetro,

Tipo\_Triangulo(lados) AS tipo

FROM triangulo;





# **Enlace a GitHub**

<https://github.com/Cortes-cmd/BBDD.git>

# **Bibliografía**

Arrays en Postgres · Etaoin Shrldu. (s/f). Netlify.app. Recuperado el 10 de febrero de 2025, de <https://etaoinshrldu.netlify.app/arrays-en-postgres/>

ChatGPT. (s/f). Chatgpt.com. Recuperado el 10 de febrero de 2025, de <https://chatgpt.com/c/67a9bab2-ff40-8001-b47e-af98967ac466>

Como utilizar un array dinámico en función plpgsql. (s/f). Narkive.com. Recuperado el 10 de febrero de 2025, de <https://pgsql-es-ayuda.postgresql.narkive.com/m3RoLpkh/como-utilizar-un-array-dinamico-en-funcion-plpgsql>

Grupo de sierras perforadoras de Beijing. (2023, agosto 31). Uso de matrices en PostgreSQL: una guía. es.xtshengguo.com. <https://es.xtshengguo.com/news/using-arrays-in-postgresql-a-guide.html>

Hosting, S. W. (2020, enero 7). Tutorial de MySQL Workbench. SiteGround. <https://www.siteground.es/tutoriales/php-mysql/mysql-workbench/>

Manejo de arrays en PostgreSQL - Foros Club Delphi. (s/f). Clubdelphi.com. Recuperado el 10 de febrero de 2025, de <https://www.clubdelphi.com/foros/showthread.php?t=91918>

pasar un array a una funcion en postgresql. (s/f). Stack Overflow en español. Recuperado el 10 de febrero de 2025, de <https://es.stackoverflow.com/questions/260042/pasar-un-array-a-una-funcion-en-postgresql>