

DEFENSA

BDA II-H3

Evaluacion

Procesual

DATOS ESTUDIANTILES

Nombres y Apellidos:	Mijail Oliver Choque Amaro
Carrera:	Ingenieria en Sistemas
Docente:	William Roddy Barra Paredes
Código Estudiantil:	SIS12955851
Universidad:	UNIFRANZ
Fecha de Entrega:	29/05/2023



MariaDB

Tabla de Contenidos

01 MANEJO DE CONCEPTOS

Se responderan
Preguntas
propuestas en la
defensa

02 PARTE PRACTICA

Resolución de
los ejercicios
propuestos

01 Manejo de Conceptos

1. Defina que es lenguaje procedural en MySQL.

- En MySQL, el lenguaje procedural se refiere a la capacidad de escribir código procedimental dentro del motor de base de datos utilizando sentencias SQL y lógica de programación. Esto se logra mediante la creación de procedimientos almacenados, funciones y desencadenadores en MySQL.

2. Defina que es una función en MySQL.

- En MySQL, una función es un objeto que realiza un cálculo o procesamiento específico y devuelve un valor como resultado. Las funciones se utilizan para encapsular lógica y operaciones repetitivas que se pueden reutilizar en consultas SQL. Una función puede aceptar parámetros como entrada, realizar cálculos o manipulaciones de datos y devolver un valor como resultado.

3. ¿Qué cosas características debe de tener una función? Explique sobre el nombre, el return, parametros, etc.

- Nombre: Una función debe tener un nombre único y descriptivo que refleje su propósito y comportamiento.
- Parámetros: Una función puede tener cero o más parámetros. Los parámetros son valores que se pasan a la función como entrada y se utilizan dentro de la función para realizar operaciones.
- Return: Una función debe tener una declaración de retorno, que especifica el tipo de dato que devolverá la función como resultado. Puede ser un tipo de dato numérico, cadena, fecha, etc.
- Lógica interna: Una función contiene la lógica de programación necesaria para realizar las operaciones y cálculos requeridos.
- Portabilidad: Una función debe ser portátil, lo que significa que se pueda utilizar en diferentes consultas y scripts de MySQL.

4. ¿Cómo crear, modificar y cómo eliminar una función? Adjunte un ejemplo de su uso?

```
#Crear una funcion
CREATE FUNCTION nombre_funcion(param1 INT, param2 VARCHAR(50))
  RETURNS tipo_dato
  BEGIN
    -- Lógica de la función
    -- Utilizar los parámetros y realizar cálculos o manipulaciones de datos
    -- Retornar el resultado utilizando la sentencia RETURN
    RETURN resultado;
  END;
```


#Modificar una funcion

```
ALTER FUNCTION nombre_funcion(param1 INT, param2 VARCHAR(50))  
  RETURNS tipo_dato  
  BEGIN  
    -- Nueva lógica de la función  
    -- Utilizar los parámetros y realizar cálculos o manipulaciones de datos  
    -- Retornar el resultado utilizando la sentencia RETURN  
    RETURN nuevo_resultado;  
  END;
```

#Eliminar una funcion

```
DROP FUNCTION nombre_funcion;
```

#Ejemplo de su Uso

```
CREATE FUNCTION calcular_area_circulo(radio FLOAT)  
  RETURNS FLOAT  
  BEGIN  
    DECLARE area FLOAT;  
    SET area = 3.14159 * radio * radio;  
    RETURN area;  
  END;
```

5. Para qué sirve la función CONCAT y como funciona en MYSQL

- ¿Crear una función que muestre el uso de la función CONCAT?
- La función debe concatenar 3 cadenas.

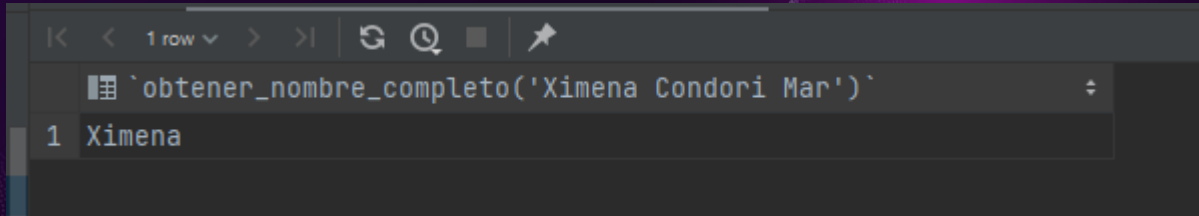
```
CREATE FUNCTION concatenar_cadenas  
(cadena1 VARCHAR(50), cadena2 VARCHAR(50), cadena3 VARCHAR(50))  
  RETURNS VARCHAR(150)  
  BEGIN  
    DECLARE R VARCHAR(150);  
    SET R = CONCAT(cadena1, ' ', cadena2, ' ', cadena3);  
    RETURN R;  
  END;
```

6. Para qué sirve la función SUBSTRING y como funciona en MYSQL

- ¿Crear una función que muestre el uso de la función SUBSTRING?
- La función recibe un nombre completo.
 - INPUT: Ximena Condori Mar
- La función solo retorna el nombre.
 - OUTPUT: Ximena

```
CREATE FUNCTION obtener_nombre_completo(nombre_completo VARCHAR(100))  
RETURNS VARCHAR(50)  
BEGIN  
    DECLARE nombre VARCHAR(50);  
    SET nombre = SUBSTRING(nombre_completo, 1, LOCATE(' ', nombre_completo) - 1);  
    RETURN nombre;  
END;
```

```
SELECT obtener_nombre_completo('Ximena Condori Mar');
```

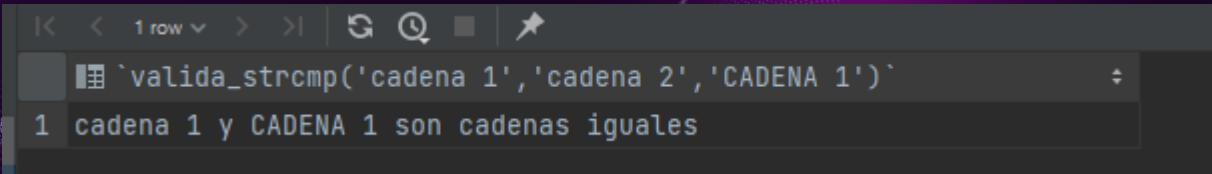


7. Para qué sirve la función STRCMP y como funciona en MYSQL

- ¿Crear una función que muestre el uso de la función STRCMP?
- La función debe comparar 3 cadenas. Y deberá determinar si dos de ellas son iguales.

```
CREATE OR REPLACE FUNCTION valida_strcmp(cad TEXT, cad1 TEXT, cad2 TEXT)
RETURNS TEXT
BEGIN
  DECLARE R TEXT DEFAULT "";
  CASE
    WHEN strcmp(cad,cad1) = 0 THEN SET R = CONCAT(cad,' y ',cad1,' son cadenas iguales');
    WHEN strcmp(cad,cad2) = 0 THEN SET R = CONCAT(cad,' y ',cad2,' son cadenas iguales');
    WHEN strcmp(cad1,cad2) = 0 THEN SET R = CONCAT(cad1,' y ',cad2,' son cadenas iguales');
    ELSE SET R = 'las 3 cadenas son distintas';
  END CASE;
  RETURN R;
END;
```

```
SELECT valida_strcmp('cadena 1','cadena 2','CADENA 1');
```



The screenshot shows a MySQL query result. At the top, there is a toolbar with navigation icons. Below it, a dropdown menu shows the query: `valida_strcmp('cadena 1','cadena 2','CADENA 1')`. The result is displayed in a table with one row and one column. The value in the column is 'cadena 1 y CADENA 1 son cadenas iguales'.

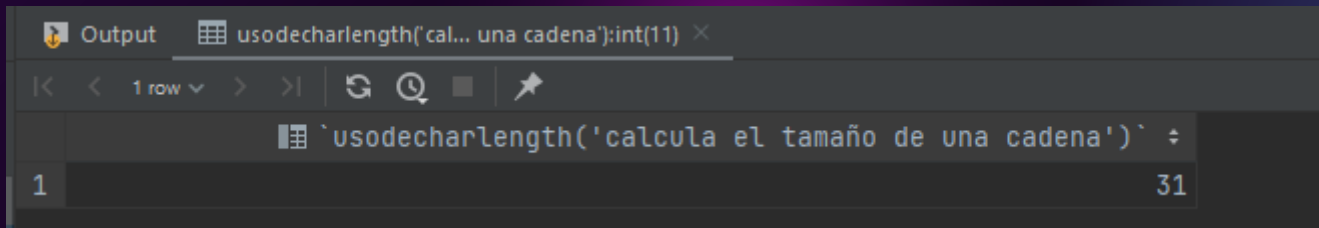
1 cadena 1 y CADENA 1 son cadenas iguales

8. Para qué sirve la función CHAR_LENGTH y LOCATE y como funciona en MYSQL

- ¿Crear una función que muestre el uso de ambas funciones?

```
CREATE OR REPLACE FUNCTION usodecharlength(cadena TEXT)
  RETURNS INT
  BEGIN
    DECLARE Tamaño INT;
    SET Tamaño = char_length(cadena);
    RETURN Tamaño;
  END;
```

```
SELECT usodecharlength('calcula el tamaño de una cadena');
```



The screenshot shows a MySQL database client window titled "Output" with a tab for the query `usodecharlength('calcula el tamaño de una cadena');int(11)`. The query results are displayed in a table with one row and one column. The row number is 1, and the result value is 31.

	<code>`usodecharlength('calcula el tamaño de una cadena')`</code>
1	31


```
CREATE OR REPLACE FUNCTION usodeLocate(cadena TEXT, encuentra TEXT)
RETURNS INT
BEGIN
  DECLARE posicion INT;
  SET posicion = LOCATE(encuentra, cadena,1);
  RETURN posicion;
END;
```

```
SELECT usodeLocate('Encuentra posicion de una cadena en especifico','posicion');
```

Output	
usodeLocate('Encuentra...', 'posicion'):int(11) X	
1 row	
`usodeLocate('Encuentra posicion de una cadena en especifico','posicion')`	
1	11

9. ¿Cual es la diferencia entre las funciones de agregación y funciones creados por el DBA? Es decir funciones creadas por el usuario.

- ❑ Las funciones de agregación son funciones predefinidas en MySQL que realizan cálculos sobre conjuntos de valores, mientras que las funciones creadas por el DBA o por el usuario son funciones personalizadas definidas por el DBA o los usuarios para realizar cálculos o lógica de programación específica adaptada a las necesidades de la aplicación o del sistema.

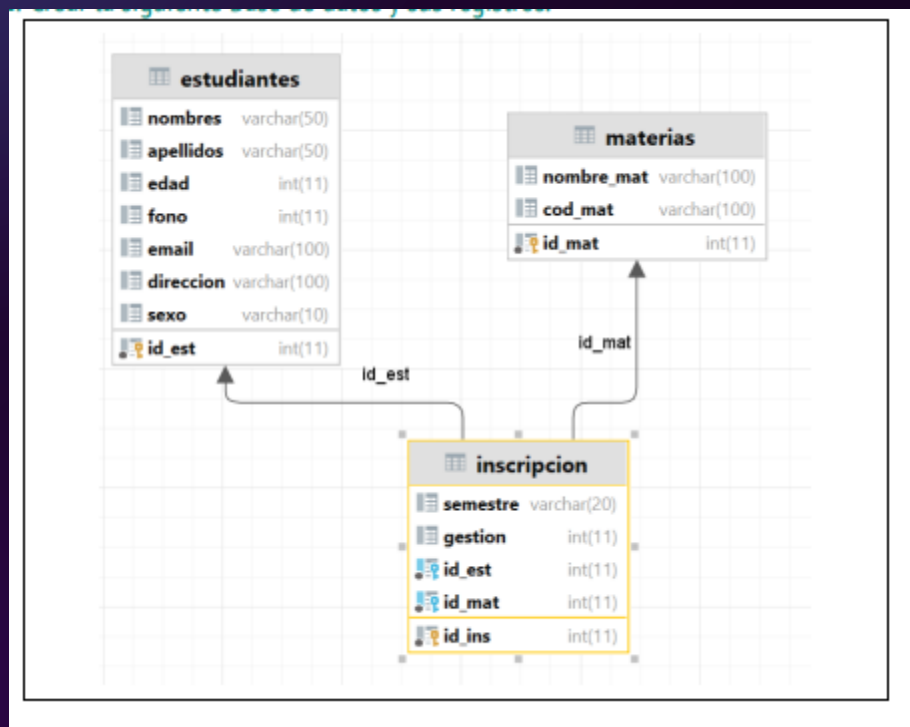
10.¿Busque y defina a qué se referirá cuando se habla de parámetros de entrada y salida en MySQL?

- Es decir IN INOUT, etc

- los parámetros de entrada (IN) se utilizan para pasar valores de datos a una función o procedimiento, los parámetros de salida (OUT) se utilizan para devolver valores de una función o procedimiento, y los parámetros de entrada y salida (INOUT) se utilizan para pasar valores de entrada y recibir valores actualizados. Estos modos de paso de parámetros permiten controlar cómo se manejan los valores dentro de una función o procedimiento, brindando flexibilidad y comunicación entre el código que llama y la función o procedimiento.

02 Parte Practica

11. Crear la siguiente Base de datos y sus registros



DATOS TABLA ESTUDIANTES

id_est	nombres	apellidos	edad	fono	email	direccion	sexo
1	Miguel	Gonzales Veliz	20	2832115	miguel@gmail.com	Av. 6 de Agosto	masculino
2	Sandra	Mavir Uriá	25	2832116	sandra@gmail.com	Av. 6 de Agosto	femenino
3	Joel	Adubiri Mondar	30	2832117	joel@gmail.com	Av. 6 de Agosto	masculino
4	Andrea	Arias Ballesteros	21	2832118	andrea@gmail.com	Av. 6 de Agosto	femenino
5	Santos	Montes Valenzuela	24	2832119	santos@gmail.com	Av. 6 de Agosto	masculino

DATOS TABLA MATERIAS

id_mat	nombre_mat	cod_mat
1	Introduccion a la Arquitectura	ARQ-101
2	Urbanismo y Diseno	ARQ-102
3	Dibujo y Pintura Arquitectonico	ARQ-103
4	Matematica discreta	ARQ-104
5	Fisica Basica	ARQ-105

DATOS TABLA INSCRIPCION

id_ins	semestre	gestion	id_est	id_mat
1	1er Semestre	2018	1	1
2	2do Semestre	2018	1	2
3	1er Semestre	2019	2	4
4	2do Semestre	2019	2	3
5	2do Semestre	2020	3	3
6	3er Semestre	2020	3	1
7	4to Semestre	2021	4	4
8	5to Semestre	2021	5	5

```
CREATE DATABASE COLEGIO;  
USE COLEGIO;
```

```
CREATE TABLE estudiantes
```

```
(  
  id_est INT PRIMARY KEY AUTO_INCREMENT,  
  nombres VARCHAR(50),  
  apellidos VARCHAR(50),  
  edad INT,  
  fono INT,  
  email VARCHAR(100),  
  direccion VARCHAR(100),  
  sexo VARCHAR(10)  
);
```

```
INSERT INTO estudiantes(nombres, apellidos, edad, fono, email, direccion, sexo)  
VALUES ('Miguel','Gonzales Veliz',20,2832115,'miguel@gmail.com','Av. 6 de Agosto','masculino'),  
       ('Sandra','Mavir Uria',25,2832116,'sandra@gmail.com','Av. 6 de Agosto','femenino'),  
       ('Joel','Aduviri Mondar',30,2832117,'joel@gmail.com','Av. 6 de Agosto','masculino'),  
       ('Andrea','Arias Ballesteros',21,2832118,'andrea@gmail.com','Av. 6 de Agosto','femenino'),  
       ('Santos','Montes Valenzuela',24,2832119,'santos@gmail.com','Av. 6 de Agosto','masculino');
```

```
CREATE TABLE materias
```

```
(  
  id_mat INT PRIMARY KEY AUTO_INCREMENT,  
  nombres_mat VARCHAR(100),  
  cod_mat VARCHAR(100)  
);
```

```
INSERT INTO materias(nombres_mat, cod_mat)  
VALUES('Introduccion a la Arquitectura','ARQ-101'),  
      ('Urbanismo y Diseo','ARQ-102'),  
      ('Dibujo y Pintura Arquitectonico','ARQ-103'),  
      ('Matematica Discreta','ARQ-104'),  
      ('Fisica Basica','ARQ-105');
```

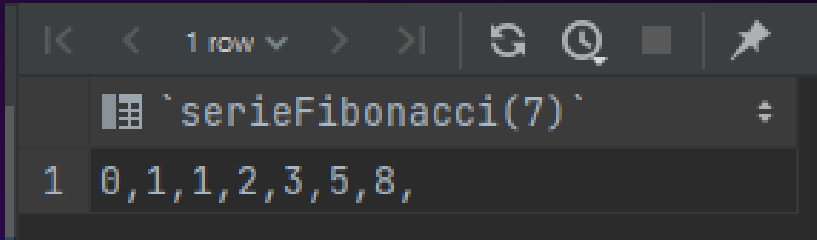
```
CREATE TABLE inscripcion
```

```
(  
  id_ins INT PRIMARY KEY AUTO_INCREMENT,  
  semestre VARCHAR(20),  
  gestion INT,  
  id_est INT,  
  id_mat INT,  
  FOREIGN KEY (id_est) REFERENCES estudiantes(id_est),  
  FOREIGN KEY (id_mat) REFERENCES materias(id_mat)  
);
```

```
INSERT INTO inscripcion(semestre, gestion, id_est, id_mat)  
VALUES ('1er Semestre',2018,1,1),  
      ('2do Semestre',2018,1,2),  
      ('1er Semestre',2018,2,4),  
      ('2do Semestre',2018,2,3),  
      ('2do Semestre',2018,3,3),  
      ('3er Semestre',2018,3,1),  
      ('4to Semestre',2018,4,4),  
      ('5to Semestre',2018,5,5);
```

12. Crear una función que genere la serie Fibonacci.

- La función recibe un límite(number)
- La función debe de retornar una cadena.
- Ejemplo para n=7. OUTPUT: 0, 1, 1, 2, 3, 5, 8,
- Adjuntar el código SQL generado y una imagen de su correcto funcionamiento.



The screenshot shows a SQL query result in a dark-themed interface. The query is `serieFibonacci(7)`. The result is displayed in a table with one row and one column, showing the output `0,1,1,2,3,5,8,`.

	serieFibonacci(7)
1	0,1,1,2,3,5,8,

#EJERCICIO 12

```
CREATE OR REPLACE FUNCTION serieFibonacci(n INT)
RETURNS TEXT
BEGIN
    DECLARE a INT DEFAULT 1;
    DECLARE b INT DEFAULT 0;
    DECLARE c INT DEFAULT 0;
    DECLARE R TEXT DEFAULT "";
    DECLARE cont INT DEFAULT 0;

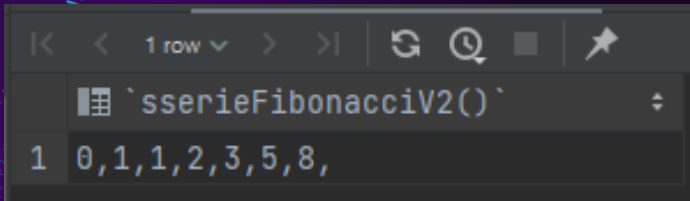
    WHILE (n > cont) DO
        SET R = CONCAT(R,c,',');
        SET c = a + b;
        SET a = b;
        SET b = c;
        SET cont = cont + 1;
    END WHILE;
    RETURN R;
END;

SELECT serieFibonacci(15);
```

13. Crear una variable global a nivel BASE DE DATOS.

- Crear una función cualquiera.
- La función debe retornar la variable global.
- Adjuntar el código SQL generado y una imagen de su correcto funcionamiento.

- Crear una variable global de nombre LIMIT.
- Este valor debe almacenar un valor entero.
 - Ejemplo, **LIMIT = 7**
 - OUTPUT: **0,1,1,2,3,5,8**
- Crear una función que genere la serie **fibonacci** hasta ese valor LIMIT.
 - Note que el valor LIMIT debe ser usado en la función
 - La función no recibe ningún parámetro.



The screenshot shows a database interface with a query bar containing the function call ``sserieFibonacciV2()'`. Below the query bar, a table displays the result of the function. The table has one column and one row, showing the Fibonacci sequence up to the 7th term: 0, 1, 1, 2, 3, 5, 8.

1 0,1,1,2,3,5,8,

#EJERCICIO 13

SET @limit = 7;

SELECT @limit;

CREATE OR REPLACE FUNCTION *sserieFibonacciV2()*
RETURNS TEXT

BEGIN

DECLARE a INT DEFAULT 1;
DECLARE b INT DEFAULT 0;
DECLARE c INT DEFAULT 0;
DECLARE R TEXT DEFAULT " "
DECLARE cont INT DEFAULT 0;

WHILE (@limit > cont) DO
SET R = CONCAT(R,c,',');
SET c = a + b;
SET a = b;
SET b = c;
SET cont = cont + 1;
END WHILE;
RETURN R;

END;

SELECT *sserieFibonacciV2()*;

14. Crear una función no recibe parámetros (Utilizar WHILE, REPEAT o LOOP).

- Previamente deberá de crear una función que obtenga la edad mínima de los estudiantes
 - La función no recibe ningún parámetro.
 - La función debe de retornar un número. (LA EDAD MÍNIMA).
- Si la edad mínima es PAR mostrar todos los pares empezando desde 0 a este ese valor de la edad mínima.

```
`paresImpares()`  
1 0,2,4,6,8,10,12,14,16,18,20,22,24,
```

- Si la edad mínima es IMPAR mostrar descendentemente todos los impares hasta el valor 0.

```
`paresImpares()`  
1 25,23,21,19,17,15,13,11,9,7,5,3,1,
```

- Retornar la nueva cadena concatenada.
- Adjuntar el código SQL generado y una imagen de su correcto funcionamiento.
- Nota: Esta función está llamando a otra función, considere eso.

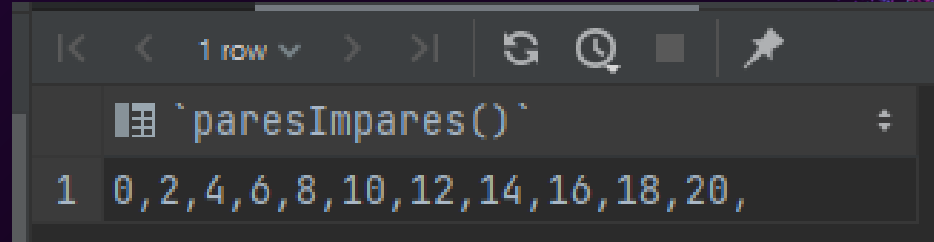
```
#EJERCICIO 14  
CREATE OR REPLACE FUNCTION minEdad()  
RETURNS TEXT  
BEGIN  
    DECLARE R INT DEFAULT 0;  
  
    SELECT MIN(E.edad)  
    FROM estudiantes E  
    INTO R;  
    RETURN R;  
END;  
SELECT minEdad();
```

```
1 row  
`minEdad()`  
1 20
```

```
CREATE OR REPLACE FUNCTION paresImpares()
RETURNS TEXT
BEGIN
    DECLARE R TEXT DEFAULT "";
    DECLARE num INT DEFAULT minEdad();
    DECLARE cont INT DEFAULT 0;
    DECLARE numipar INT DEFAULT minEdad();

    REPEAT
        IF num % 2 = 0 THEN
            SET R = CONCAT(R,cont,',');
            SET cont = cont + 2;
        ELSE
            SET R = CONCAT(R,numipar,',');
            SET numipar = numipar - 2;
            SET cont = cont + 2;
        END IF;
    UNTIL cont > num END REPEAT;
    RETURN R;
END;

SELECT paresImpares();
```

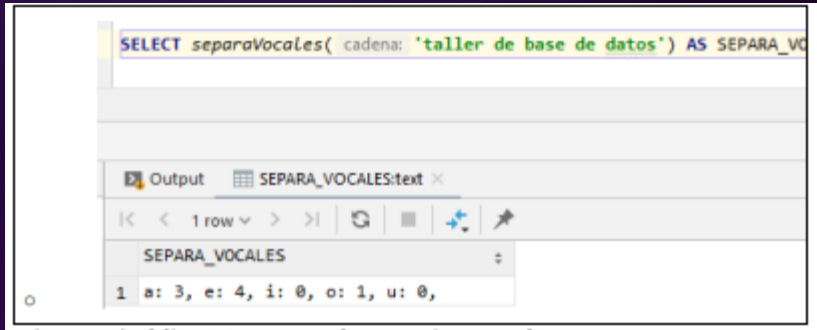


The screenshot shows a database interface with a query result. The query is ``paresImpares()``. The result is a single row containing the string `0,2,4,6,8,10,12,14,16,18,20,`. The interface includes navigation buttons at the top and a table icon next to the query name.

1 0,2,4,6,8,10,12,14,16,18,20,

15. Crear una función que determina cuantas veces se repite las vocales.

- La función recibe una cadena y retorna un TEXT.
- Retornar todas las vocales ordenadas e indicando la cantidad de veces que se repite en la cadena.
- Resultado esperado.



The screenshot shows a SQL query in a text editor and its output in a database client. The query is: `SELECT separaVocales(cadena: 'taller de base de datos') AS SEPARA_VO`. The output window, titled "Output" and "SEPARA_VOCALES:text", shows a single row with the result: `1 a: 3, e: 4, i: 0, o: 1, u: 0,`.

```
SELECT separaVocales( cadena: 'taller de base de datos') AS SEPARA_VO
```

SEPARA_VOCALES
1 a: 3, e: 4, i: 0, o: 1, u: 0,

- Adjuntar el código SQL generado y una imagen de su correcto funcionamiento.

```

#EJERCICIO 15
CREATE OR REPLACE FUNCTION separaVocales(CAD VARCHAR(100))
RETURNS TEXT
BEGIN
    DECLARE R TEXT DEFAULT "";
    DECLARE cont INT DEFAULT 1;
    DECLARE Nveces TEXT DEFAULT 0;
    DECLARE PUNTERO CHAR;
    DECLARE acont INT DEFAULT 0;
    DECLARE econt INT DEFAULT 0;
    DECLARE icont INT DEFAULT 0;
    DECLARE ocont INT DEFAULT 0;
    DECLARE ucont INT DEFAULT 0;

    WHILE cont <= char_length(CAD) DO
        SET PUNTERO = SUBSTR(CAD,cont,1);
        CASE PUNTERO
            WHEN 'a' THEN SET acont = acont + 1;
            WHEN 'e' THEN SET econt = econt + 1;
            WHEN 'i' THEN SET icont = icont + 1;
            WHEN 'o' THEN SET ocont = ocont + 1;
            WHEN 'u' THEN SET ucont = ucont + 1;
            ELSE SET Nveces = 'no hay vocales';
        END CASE;
        SET cont = cont + 1;
    END WHILE;

    IF(acont != 0 || econt != 0 || icont != 0 || ocont != 0 || ucont != 0) THEN
        SET R = CONCAT('a:',acont,', e:',econt,', i:',icont,', o:',ocont,', u:',ucont);
    ELSE
        SET R = Nveces;
    END IF;

    RETURN R;
END;

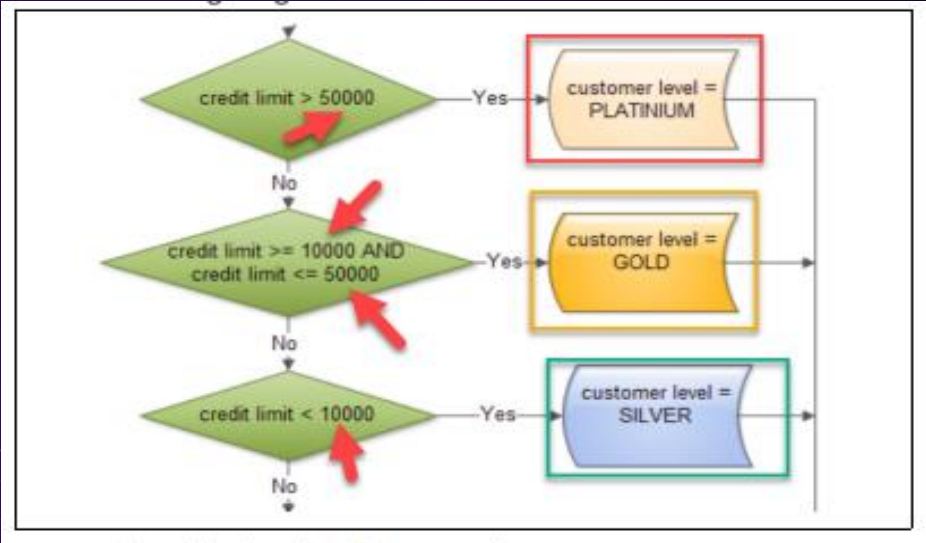
SELECT separaVocales('TALLER');

```

1 row	
	<code>`separaVocales('TALLER')`</code>
1	a:1, e:1, i:0, o:0, u:0

16. Crear una función que recibe un parámetro INTEGER.

- La función debe de retornar un texto(TEXT) como respuesta.
- El parámetro es un valor numérico credit_number.
- Si es mayor a 50000 es PLATINIUM.
- Si es mayor igual a 10000 y menor igual a 50000 es GOLD.
- Si es menor a 10000 es SILVER
- La función debe retornar indicando si ese cliente es PLATINUM, GOLD o SILVER en base al valor del credit_number.
- Considere la imagen siguiente:



- Para resolver debe de utilizar la instrucción CASE - WHEN.
- Adjuntar el código SQL generado y una imagen de su correcto funcionamiento.

#EJERCICIO 16

```
CREATE FUNCTION detTipoCliente(creditLimit INT)
```

```
RETURNS TEXT
```

```
BEGIN
```

```
    DECLARE R TEXT DEFAULT 'Customer Level: ';
```

```
    CASE
```

```
        WHEN creditLimit > 50000 THEN SET R = CONCAT(R,'PLATINIUM');
```

```
        WHEN creditLimit BETWEEN 10000 AND 50000 THEN SET R = CONCAT(R,'GOLD');
```

```
        WHEN creditLimit < 10000 THEN SET R = CONCAT(R,'SILVER');
```

```
        ELSE SET R = 'No esta dentro del rango de dinero requerido para determinar el tipo de cliente';
```

```
    END CASE;
```

```
    RETURN R;
```

```
END;
```

```
SELECT detTipoCliente(9999);
```

```
SELECT detTipoCliente(25000);
```

```
SELECT detTipoCliente(50001);
```

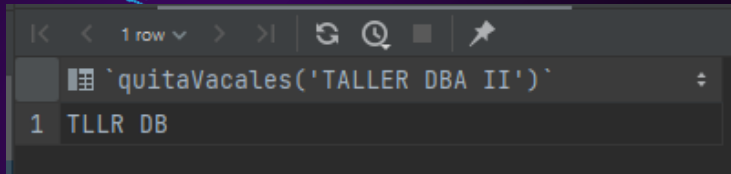
1	Customer Level: SILVER
---	------------------------

1	Customer Level: GOLD
---	----------------------

1	Customer Level: PLATINIUM
---	---------------------------

17. Crear una función que recibe 2 parámetros VARCHAR(20), VARCHAR(20).

- La función debe de retornar un texto TEXT como respuesta.
- Si las cadenas fueran “TALLER DBA II” y la segunda cadena fuese “GESTION 2023”.
- La nueva cadena debería ser “TLLR DB -GSTN 2023”.
- La nueva cadena es resultado de la concatenación de todos los valores distintos a las vocales.
- Retornar la nueva cadena concatenada.



The screenshot shows a database interface with a query bar containing the function call: ``quitaVocales('TALLER DBA II')``. Below the query bar, a table displays the result of the function. The table has one row with the value 'TLLR DB'.

1	TLLR DB
---	---------

#EJERCICIO 17

```
CREATE OR REPLACE FUNCTION quitaVocales(cadena VARCHAR(20))  
RETURNS TEXT
```

```
BEGIN
```

```
    DECLARE R TEXT DEFAULT '';
```

```
    DECLARE cont INT DEFAULT 1;
```

```
    DECLARE PUNTERO CHAR;
```

```
    WHILE cont <= char_length(cadena) DO
```

```
        SET PUNTERO = SUBSTR(cadena,cont,1);
```

```
        CASE PUNTERO
```

```
            WHEN 'a' THEN SET R = R;
```

```
            WHEN 'e' THEN SET R = R;
```

```
            WHEN 'i' THEN SET R = R;
```

```
            WHEN 'o' THEN SET R = R;
```

```
            WHEN 'u' THEN SET R = R;
```

```
            WHEN '' THEN SET R = CONCAT(R, '');
```

```
            ELSE SET R = CONCAT(R,PUNTERO);
```

```
        END CASE;
```

```
        SET cont = cont + 1;
```

```
    END WHILE;
```

```
    RETURN R;
```

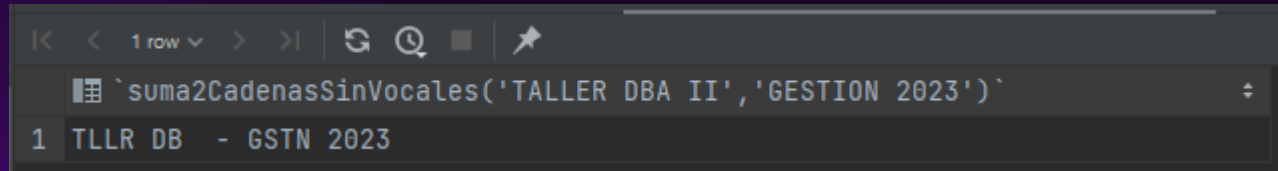
```
END;
```

```
SELECT quitaVocales('TALLER DBA II');
```

```
CREATE OR REPLACE FUNCTION suma2CadenasSinVocales(CAD1 VARCHAR(20), CAD2 VARCHAR(20))
RETURNS TEXT
BEGIN
    DECLARE R TEXT DEFAULT "";
    DECLARE R1 TEXT DEFAULT quitaVocales(CAD1);
    DECLARE R2 TEXT DEFAULT quitaVocales(CAD2);

    SET R = CONCAT(R1,' - ',R2);
    RETURN R;
END;

SELECT suma2CadenasSinVocales("TALLER DBA II",'GESTION 2023');
```

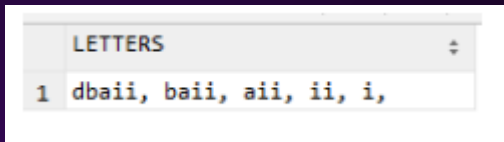


The screenshot shows a database interface with a query bar containing the function call: ``suma2CadenasSinVocales('TALLER DBA II','GESTION 2023')``. Below the query bar, a table displays the result of the query.

1	TLLR DB - GSTN 2023

18. Crear una función que reciba un parámetro TEXT

- En donde este parámetro deberá de recibir una cadena cualquiera y retorna un TEXT de respuesta.
- Concatenar N veces la misma cadena reduciendo en uno en cada iteración hasta llegar a una sola letra.
- Utilizar REPEAT y retornar la nueva cadena concatenada.
- Considerar la siguiente imagen:



	LETTERS
1	dbaii, baii, aii, ii, i,

- Adjuntar el código SQL generado y una imagen de su correcto funcionamiento.

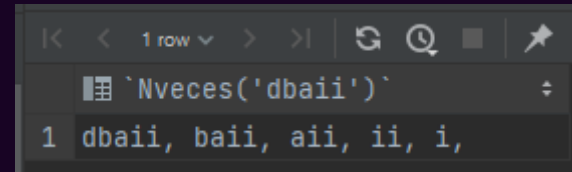
```
CREATE OR REPLACE FUNCTION Nvecas(cadena VARCHAR(20))
RETURNS TEXT
BEGIN

    DECLARE R TEXT DEFAULT "";
    DECLARE cont INT DEFAULT 1;
    DECLARE PUNTERO VARCHAR(100);

    REPEAT
        SET PUNTERO = SUBSTR(cadena,cont);
        SET R = CONCAT(R,PUNTERO,', ');
        SET cont = cont + 1;
    UNTIL cont > char_length(cadena) END REPEAT;

    RETURN R;
END;

SELECT Nvecas('dbaii');
```



	`Nvecas('dbaii')`
1	dbaii, baii, aii, ii, i,



**Muchas
Gracias**