

HITO 3, EVALUACIÓN PROCESUAL BASE DE DATOS 2

UNIVERSIDAD PRIVADA FRANZ TAMAYO
DEFENSA HITO 3 – EVALUACION PROCESUAL

Estudiante: Univ. Jhomar Huaycho Quispe

Asignatura: BASE DE DATOS II

Carrera: INGENIERÍA DE SISTEMAS

Paralelo: BDA (1)

Docente: Lic. William Barra Paredes

fecha: 21/10/2022

GITHUB: <https://github.com/JHOMARHUAYCHO/BASE-DE-DATOS-2/tree/main/HITO%203>

VIDEO EXPOSICION :

https://drive.google.com/drive/folders/1RJR5ewBdjST_a6ELJI3QC14LwqUv4hRO?usp=sharing

MANEJO DE CONCEPTOS

- **1- Defina que es lenguaje procedural en Mysql.**

R. El lenguaje procedural es el manejo de instrucciones o estructuras de control (programación)dentro de una base de datos.

- **2. Defina que es una función en MySQL.**

como su nombre indican son funciones almacenadas que pueden modificar o realizar operaciones con los registros de la base de datos, estos pueden o no recibir parámetros .

3. ¿QUÉ COSAS CARACTERÍSTICAS DEBE DE TENER UNA FUNCIÓN? EXPLIQUE SOBRE EL: NOMBRE, EL RETURN, PARAMETROS, ETC

NOMBRE DE LA FUNCION
el nombre debe ser único,
y no debe tener espacios.

CREATE FUNCTION SUMAR (a INTEGER, b INTEGER)

Returns INTEGER

Begin

OPERACIONES
En esta parte se pueden
declara variables extras
y operaciones con estas

Declare numero INTEGER default 0;
Set numero=a+b;

Return numero;
End;

PARAMETROS

- Son los datos que se envían , para hacer operaciones.
- Para declarar un paramtero primero debe ser el nombre del parametro seguido del tipo de dato.
- No es obligatorio enviar parámetros.
- Se puede enviar una cantidad n de parámetros

RETURNS

se refiere al tipo de dato que va ha devolver la funcion

RETURN

Este return retorna el valor final, de la funcion,
El tipo de dato debe coincidir con el tipo de dato del 'returns'.

4. ¿CÓMO CREAR, MODIFICAR Y CÓMO ELIMINAR UNA FUNCIÓN? ADJUNTE UN EJEMPLO DE SU USO.

- Para crear la función, se debe usar la sentencia **CREATE FUNCTION**.

Ejemplo:

```
CREATE FUNCTION SUMAR (a INTEGER, b INTEGER)
Returns INTEGER
Begin
Declare numero INTEGER default 0;
Set numero=a+b;
Return numero;
End;
```

- Para modificar una función se debe hacer uso de la sentencia **REPLACE**

Ejemplo:

```
CREATE OR REPLACE FUNCTION SUMAR (a INTEGER, b INTEGER)
Returns INTEGER
Begin
Declare numero INTEGER default 0;
Set numero=a+b;
Return numero;
End;
```

- Para eliminar la función se debe hacer uso de la sentencia **DROP**

Ejemplo

```
DROP FUNCTION SUMAR;
```

5. PARA QUÉ SIRVE LA FUNCIÓN CONCAT Y COMO FUNCIONA EN MYSQL

- ¿CREAR UNA FUNCIÓN QUE MUESTRE EL USO DE LAS FUNCIÓN CONCAT?
- LA FUNCIÓN DEBE CONCATENAR 3 CADENAS.

- La función CONCAT se encuentra ya almacenada en mysql y sirve para unir 3 o mas cadenas en una sola ejemplo:

```
CREATE OR REPLACE FUNCTION CONCATENAR (A VARCHAR(20),B VARCHAR(20),C VARCHAR(20))  
RETURNS VARCHAR(60)  
BEGIN  
    DECLARE CADENA VARCHAR(60) DEFAULT "";  
    SET CADENA= CONCAT(A,' ',B,' ',C);  
    RETURN CADENA;  
END;  
  
SELECT CONCATENAR('HOLA','COMO','ESTAS');
```

```
1 CONCATENAR('HOLA','COMO','ESTAS')  
1 HOLA COMO ESTAS
```

6. PARA QUÉ SIRVE LA FUNCIÓN SUBSTRING Y COMO FUNCIONA EN MYSQL

- ¿CREAR UNA FUNCIÓN QUE MUESTRE EL USO DE LAS FUNCIÓN SUBSTRING?
- LA FUNCIÓN RECIBE UN NOMBRE COMPLETO.
 - INPUT: XIMENA CONDORI MAR
- LA FUNCIÓN SOLO RETORNA EL NOMBRE.
 - OUTPUT: XIMENA

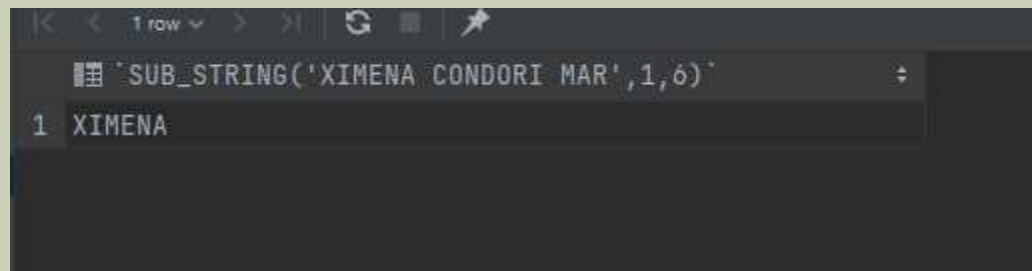
- LA FUNCION SUBSTRING PERMITE TOMAR PARTE DE UNA CADENA, LA SENTENCIA UTILIZADA ES
SUBSTRING(CADENA,POSICION,LONGITUD)

Donde :

CADENA es la palabra(s), **POSICION** debe ser un numero mayor a 1, se refiere a la letra desde donde se contara, **LONGITUD** debe ser un nmero, se refiere a la cantidad de letras que se tomaran.

```
CREATE OR REPLACE FUNCTION SUB_STRING (NOMBRE VARCHAR(50),POSICION INTEGER,LONGITUD INTEGER)
RETURNS VARCHAR(50)
BEGIN
  DECLARE CADENA VARCHAR(50) DEFAULT "";
  SET CADENA= SUBSTRING(NOMBRE,POSICION,LONGITUD);
  RETURN CADENA;
END;

SELECT SUB_STRING('XIMENA CONDORI MAR',1,6);
```



7. PARA QUÉ SIRVE LA FUNCION STRCMP Y COMO FUNCIONA EN MYSQL

- ¿CREAR UNA FUNCIÓN QUE MUESTRE EL USO DE LAS FUNCIÓN STRCMP?
- LA FUNCIÓN DEBE COMPARAR 3 CADENAS. Y DEBERÁ DETERMINAR SI DOS DE ELLAS SON IGUALES.

- La función **strcmp** nos permite comparar 2 cadenas

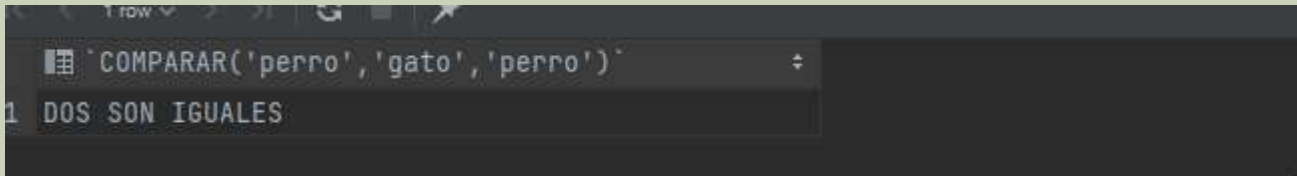
Ejemplo strcmp('perro','gato')

```
CREATE OR REPLACE FUNCTION COMPARAR (A VARCHAR(20),B VARCHAR(20),C VARCHAR(20))
RETURNS VARCHAR(60)
BEGIN
    DECLARE RESPUESTA VARCHAR(60) DEFAULT "";

    IF STRCMP(A,B)=0 THEN SET RESPUESTA='DOS SON IGUALES';

        ELSEIF STRCMP(A,C)=0 THEN SET RESPUESTA='DOS SON IGUALES';
        ELSEIF STRCMP(B,C)=0 THEN SET RESPUESTA='DOS SON IGUALES';
    ELSE
        SET RESPUESTA='NINGUNO SON IGUALES';
    END IF;
    RETURN RESPUESTA;
END;

SELECT COMPARAR('perro','gato','perro');
```



The screenshot shows a MySQL query window with the following content:

```
COMPARAR('perro','gato','perro')
```

The result of the query is displayed in a table with one row and one column:

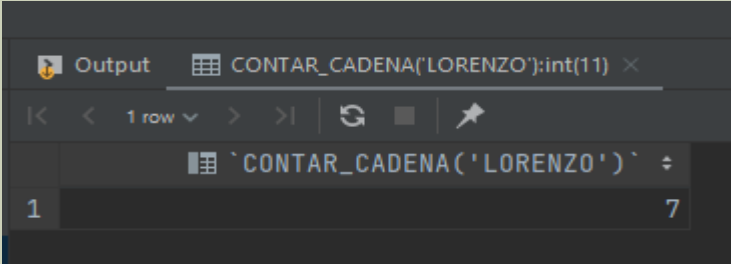
1	DOS SON IGUALES
---	-----------------

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?

- **CHAR_LENGTH:** NOS PERMITE CONTAR LA CANTIDAD DE LETRAS Y ESPACIOS DE UNA CADENA.

```
CREATE OR REPLACE FUNCTION CONTAR_CADENA(A VARCHAR(20))
RETURNS INTEGER
BEGIN
    DECLARE RESPUESTA VARCHAR(20) DEFAULT "";
    SET RESPUESTA=CHAR_LENGTH(A);
    RETURN RESPUESTA;
END;
SELECT CONTAR_CADENA('LORENZO');
```

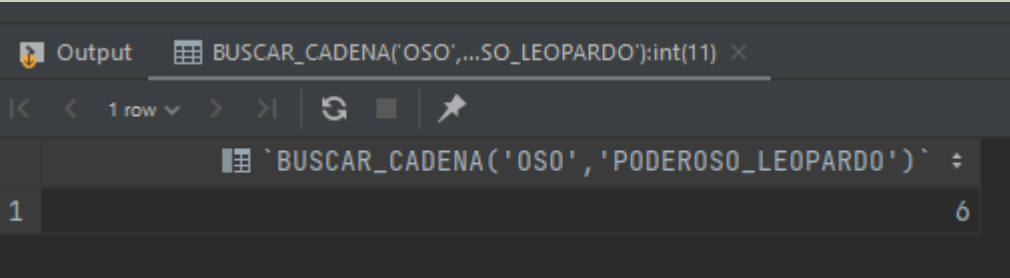


Output: CONTAR_CADENA('LORENZO');int(11) ×

1 row	
`CONTAR_CADENA('LORENZO')`	
1	7

- **LOCATE :** NOS PERMITE ENCONTRAR UNA CADENA DENTRO OTRA CADENA

```
CREATE OR REPLACE FUNCTION BUSCAR_CADENA(SUBCADENA VARCHAR(20),CADENA VARCHAR(20))
RETURNS INTEGER
BEGIN
    DECLARE RESP INTEGER DEFAULT 0;
    SET RESP=LOCATE(SUBCADENA,CADENA);
    RETURN RESP;
END;
SELECT BUSCAR_CADENA('OSO','PODEROSO_LEOPARDO');
```



Output: BUSCAR_CADENA('OSO','PODEROSO_LEOPARDO');int(11) ×

1 row	
`BUSCAR_CADENA('OSO','PODEROSO_LEOPARDO')`	
1	6

9. ¿CUAL ES LA DIFERENCIA ENTRE LAS FUNCIONES DE AGREGACIÓN Y FUNCIONES CREADOS POR EL DBA? ES DECIR FUNCIONES CREADAS POR EL USUARIO.

- **FUNCIONES DE AGREGACION:** Son funciones ya almacenadas no necesitan ser creadas, las cuales ya podemos usar o llamar.
- **FUNCIONES CREADAS POR EL USUARIO:** son funciones que necesitan ser creadas, en la mayoría estas funciones se crean haciendo uso de las funciones de agregación.

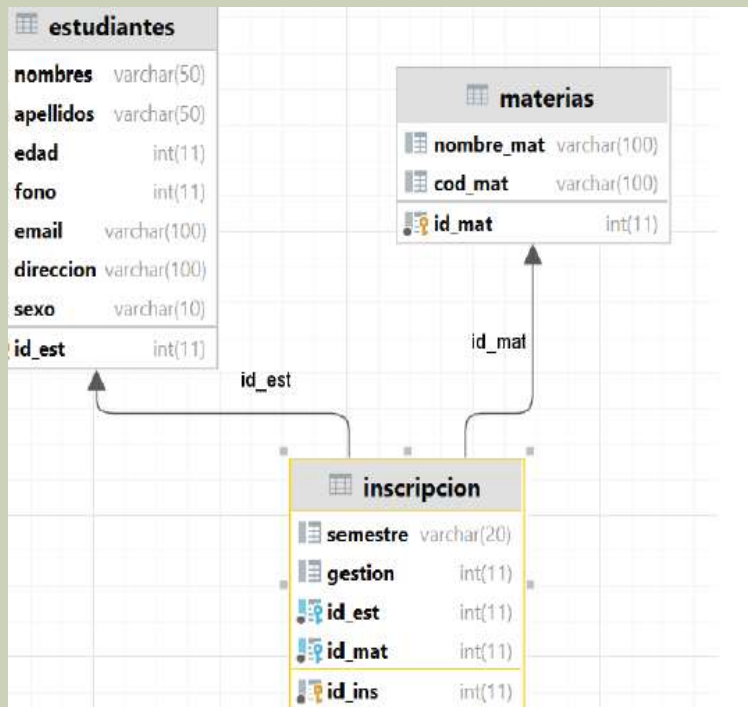
10.¿BUSQUE Y DEFINA A QUÉ SE REFERIRÁ CUANDO SE HABLA DE PARÁMETROS DE ENTRADA Y SALIDA EN MYSQL? ES DECIR IN INOUT, ETC

- **PARAMETROS DE ENTRADA «IN»:** es un parámetro que entra dentro del procedimiento almacenado con el cual se harán operaciones.
- **PARAMETROS DE SALIDA»OUT»:** es un parámetro el cual devolverá un dato de salida al usuario.
- **PARRAMETRO DE ENTRADA Y SALIDA «INOUT»:** es un parámetro que entrara dentro del procedimiento y devolverá un dato al usuario.

PARTE PRACTICA

PREGUNTA 11

- 11. Crear la siguiente Base de datos y sus registros.



DATOS TABLA ESTUDIANTES						
id_est	nombres	apellidos	edad	fono	email	direccion
1	Nigel	Narcales Vella	20	3032133	nigel@gmail.com	Av. 8 de Agosto
2	Sandra	Navio Wila	25	3032130	sandrad@gmail.com	Av. 8 de Agosto
3	Joel	Abdini Mendez	30	3032137	joel@gmail.com	Av. 8 de Agosto
4	Andrea	Arias Salazar	25	3032139	andrea@gmail.com	Av. 8 de Agosto
5	Santon	Martinez Valenzuela	24	3032130	santon@gmail.com	Av. 8 de Agosto

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	1er Semestre	2020	3	1
7	4to Semestre	2021	4	4
8	5to Semestre	2021	5	5

```
CREATE DATABASE H3_UNIVERSIDAD;
USE H3_UNIVERSIDAD;
CREATE TABLE estudiantes
(
  id_est INTEGER AUTO_INCREMENT PRIMARY KEY NOT NULL,
  nombres VARCHAR(50),
  apellidos VARCHAR(50),
  edad INTEGER,
  fono INTEGER,
  email VARCHAR(100),
  direccion VARCHAR(100),
  sexo VARCHAR(10)
);

CREATE TABLE materias
(
  id_mat INTEGER AUTO_INCREMENT PRIMARY KEY NOT NULL,
  nombre_mat VARCHAR(100),
  cod_mat VARCHAR(100)
);

CREATE TABLE inscripcion
(
  id_ins INTEGER AUTO_INCREMENT PRIMARY KEY NOT NULL,
  semestre VARCHAR(20),
  gestion INTEGER,

  id_est INT NOT NULL,
  id_mat INT NOT NULL,
  FOREIGN KEY (id_est) REFERENCES estudiantes (id_est),
  FOREIGN KEY (id_mat) REFERENCES materias (id_mat)
);
```

```
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', 'Aduhiri 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');
```

```
INSERT INTO materias (nombre_mat, cod_mat)
VALUES ('Introduccion a la Arquitectura', 'ARQ-101'),
       ('Urbanismo y Diseno', 'ARQ-102'),
       ('Dibujo y Pintura Arquitectonico', 'ARQ-103'),
       ('Matematica discreta', 'ARQ-104'),
       ('Fisica Basica', 'ARQ-105');
```

```
INSERT INTO inscripcion (id_est, id_mat, semestre, gestion)
VALUES (1, 1, '1er Semestre', 2018),
       (1, 2, '2do Semestre', 2018),
       (2, 4, '1er Semestre', 2019),
       (2, 3, '2do Semestre', 2019),
       (3, 3, '2do Semestre', 2020),
       (3, 1, '3er Semestre', 2020),
       (4, 4, '4to Semestre', 2021),
       (5, 5, '5to Semestre', 2021);
```



	id_est	nombres	apellidos	edad	fono	email	direccion	sexo
1	1	Miguel	Gonzales Veliz	20	2832115	miguel@gmail.com	Av. 8 de Agosto	masculino
2	2	Sandra	Mavir Uria	25	2832116	sandra@gmail.com	Av. 8 de Agosto	femenino
3	3	Joel	Adubiri Mondar	30	2832117	joel@gmail.com	Av. 8 de Agosto	masculino
4	4	Andrea	Arias Ballesteros	21	2832118	andrea@gmail.com	Av. 8 de Agosto	femenino
5	5	Santos	Montes Valenzuela	24	2832119	santos@gmail.com	Av. 8 de Agosto	masculino

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

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

PREGUNTA 12

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.**

```
CREATE OR REPLACE FUNCTION FIBONANCI(NUMBER INTEGER)
```

```
RETURNS TEXT
```

```
BEGIN
```

```
    DECLARE A INTEGER DEFAULT 0;
```

```
    DECLARE B INTEGER DEFAULT 1;
```

```
    DECLARE AUX INTEGER DEFAULT 0;
```

```
    DECLARE CONTADOR INTEGER DEFAULT 0;
```

```
    DECLARE CADENA TEXT DEFAULT '';
```

```
    SET CADENA = CONCAT(A, ',', B);
```

```
    IF NUMBER=1 THEN SET CADENA='0';
```

```
    ELSEIF NUMBER=2 THEN SET CADENA='0,1';
```

```
    ELSEIF NUMBER<=0 THEN SET CADENA='DEBE SER MAYOR A CERO';
```

```
    ELSE
```

```
        REPEAT
```

```
            SET AUX=A+B;
```

```
            SET CADENA = CONCAT(CADENA, ',', AUX);
```

```
            SET A=B;
```

```
            SET B=AUX;
```

```
            SET CONTADOR=CONTADOR+1;
```

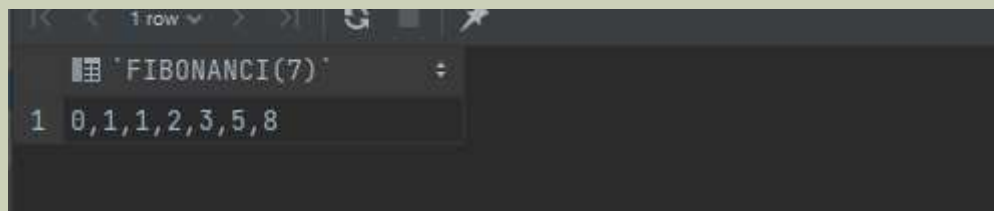
```
            UNTIL CONTADOR = NUMBER-2 END REPEAT;
```

```
    END IF;
```

```
    RETURN CADENA;
```

```
END;
```

```
SELECT FIBONANCI(8);
```



The screenshot shows a database query result window. At the top, it says "1 row". Below that, there is a table with one column and one row. The column header is "FIBONANCI(7)". The row contains the value "0,1,1,2,3,5,8".

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

PREGUNTA 13

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.

```

SET @LIMIT = 10;

CREATE OR REPLACE FUNCTION FIBONANCI_V2()

RETURNS TEXT
BEGIN
    DECLARE A INTEGER DEFAULT 0;
    DECLARE B INTEGER DEFAULT 1;
    DECLARE AUX INTEGER DEFAULT 0;
    DECLARE CONTADOR INTEGER DEFAULT 0;
    DECLARE CADENA TEXT DEFAULT '';
    SET CADENA =CONCAT(A,',',B);
    IF @LIMIT=1 THEN SET CADENA='0';
    ELSEIF @LIMIT=2 THEN SET CADENA='0,1';
    ELSEIF @LIMIT<=0 THEN SET CADENA='DEBE SER MAYOR A CERO';
    ELSE
        REPEAT

            SET AUX=A+B;
            SET CADENA = CONCAT(CADENA,',',AUX);
            SET A=B;
            SET B=AUX;
            SET CONTADOR=CONTADOR+1;
            UNTIL CONTADOR = @LIMIT-2 END REPEAT;
    END IF;
    RETURN CADENA;
END;

SELECT FIBONANCI_V2();

```

SELECT FIBONANCI_V2()	
1	0,1,1,2,3,5,8,13,21,34

PREGUNTA 14

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,
```

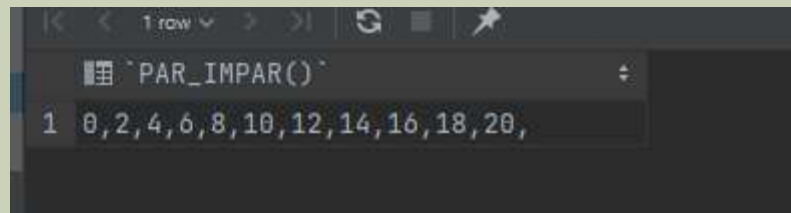
- Retornar la nueva cadena concatenada.
- Adjuntar el código SQL generado y una imagen de su correcto funcionamiento.
- **Nota:** Esta función está llamando a otra función, considere eso.

```
CREATE OR REPLACE FUNCTION EDAD_MINIMA ()  
RETURNS INTEGER  
BEGIN  
    DECLARE NUMERO INTEGER DEFAULT 0;  
    SELECT MIN(estudiantes.edad) INTO NUMERO  
    FROM estudiantes;  
    RETURN NUMERO;  
end;
```

```
SELECT EDAD_MINIMA();
```

```
CREATE OR REPLACE FUNCTION PAR_IMPAR()  
RETURNS TEXT  
BEGIN  
    DECLARE CADENA TEXT DEFAULT '';  
    DECLARE AUX INTEGER DEFAULT 0;  
  
    IF EDAD_MINIMA()%2=0 THEN  
        WHILE AUX<=EDAD_MINIMA() DO  
            SET CADENA=CONCAT(CADENA,AUX,',');  
            SET AUX=AUX+2;  
        end while;  
    ELSE  
        SET AUX=EDAD_MINIMA();  
        WHILE AUX>=0 DO  
            SET CADENA=CONCAT(CADENA,AUX,',');  
            SET AUX=AUX-2;  
        end while;  
  
    end if;  
    RETURN CADENA;  
END;
```

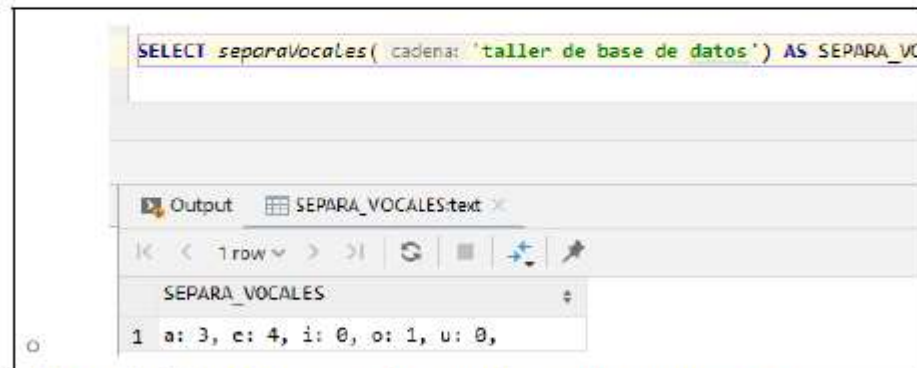
```
SELECT PAR_IMPAR();
```



PREGUNTA 15

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

- o La función recibe una cadena y retorna un TEXT.
- o Retornar todas las vocales ordenadas e indicando la cantidad de veces que se repite en la cadena.
- o 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 value: `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,

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

```

CREATE OR REPLACE FUNCTION CONTAR_VOCALES(PALABRA TEXT)
RETURNS TEXT
BEGIN
    DECLARE RESPUESTA TEXT DEFAULT '';
    DECLARE CONTADOR INTEGER DEFAULT CHAR_LENGTH(PALABRA);
    DECLARE AUX INTEGER DEFAULT 1;
    DECLARE CA INTEGER DEFAULT 0;
    DECLARE CE INTEGER DEFAULT 0;
    DECLARE CI INTEGER DEFAULT 0;
    DECLARE CO INTEGER DEFAULT 0;
    DECLARE CU INTEGER DEFAULT 0;
    DECLARE LETRA TEXT DEFAULT '';

    REPEAT
    SET LETRA=SUBSTRING(PALABRA,AUX,1);
    IF
        STRCMP('A',LETRA)=0 THEN
            SET CA=CA+1;
        ELSEIF STRCMP('E',LETRA)=0 THEN
            SET CE=CE+1;
        ELSEIF STRCMP('I',LETRA)=0 THEN
            SET CI=CI+1;
        ELSEIF STRCMP('O',LETRA)=0 THEN
            SET CO=CO+1;
        ELSEIF STRCMP('U',LETRA)=0 THEN
            SET CU=CU+1;
    end if;

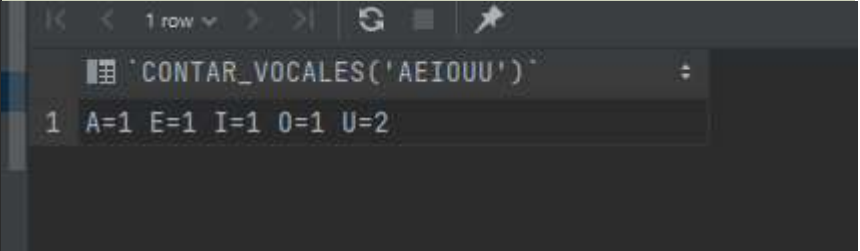
    SET AUX=AUX+1;
    SET CONTADOR=CONTADOR-1;

    until CONTADOR <=0 end repeat;

    SET RESPUESTA=CONCAT('A=',CA,'E=',CE,'I=',CI,'O=',CO,'U=',CU);
    RETURN RESPUESTA;
end;

SELECT CONTAR_VOCALES('AEIOUU');

```



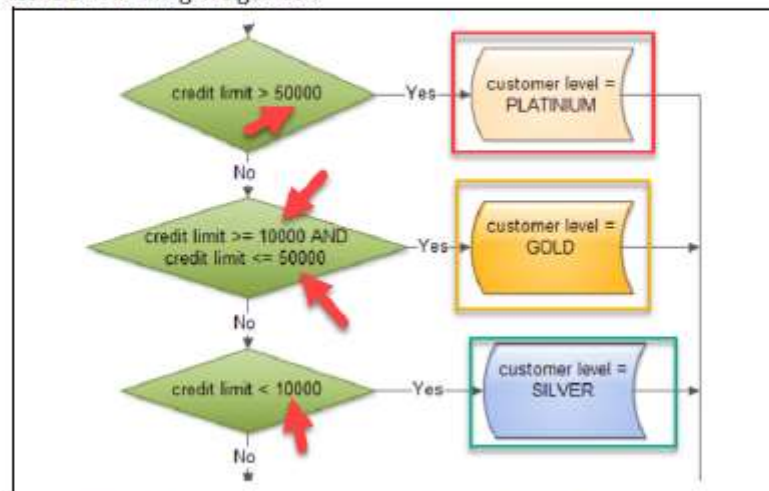
The screenshot shows a database interface with a query result table. The query is 'CONTAR_VOCALES('AEIOUU')'. The result is a single row with the following values: A=1, E=1, I=1, O=1, U=2.

	A=1	E=1	I=1	O=1	U=2
1	A=1	E=1	I=1	O=1	U=2

PREGUNTA 16

6. 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.

```
CREATE OR REPLACE FUNCTION CLASE_CLIENTE( CREDIT_NUMBER INTEGER)

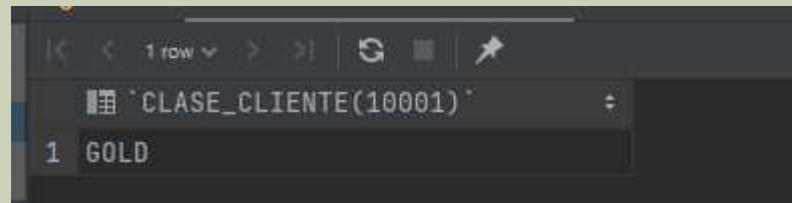
RETURNS TEXT
BEGIN
    DECLARE RESPUESTA TEXT DEFAULT "";

    CASE
        WHEN CREDIT_NUMBER >= 50000 THEN SET RESPUESTA = 'PLATINUM';
        WHEN CREDIT_NUMBER >= 10000 AND CREDIT_NUMBER < 50000 THEN SET RESPUESTA = 'GOLD';
        WHEN CREDIT_NUMBER < 10000 THEN SET RESPUESTA = 'SILVER';
        ELSE SET RESPUESTA = 'NO PERTENCE A NINGUNA CLASE';
    END CASE ;

    RETURN RESPUESTA;

END;

SELECT CLASE_CLIENTE(10001);
```



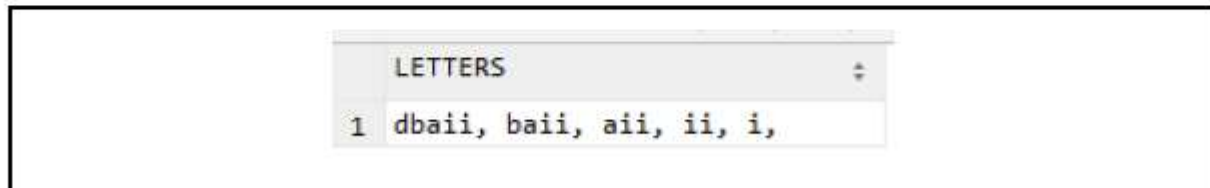
The screenshot shows a database query result in a dark-themed interface. At the top, there is a toolbar with navigation icons and a dropdown menu showing '1 row'. Below the toolbar, the query 'CLASE_CLIENTE(10001)' is displayed. The result is a single row with the value 'GOLD'.

1	GOLD
---	------

PREGUNTA 17

17. 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:



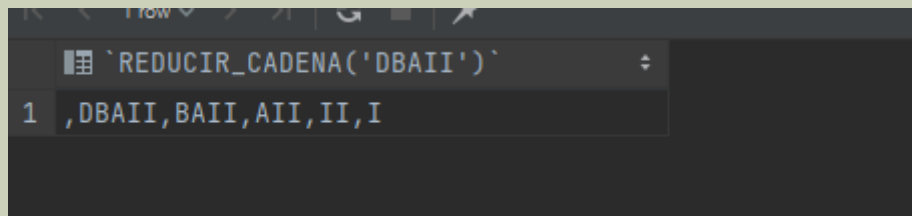
- Adjuntar el **código SQL** generado y una imagen de su correcto **funcionamiento**.

```
CREATE OR REPLACE FUNCTION REDUCIR_CADENA (PALABRA TEXT)
RETURNS TEXT
BEGIN
    DECLARE RESPUESTA TEXT DEFAULT '';
    DECLARE CONTADOR INTEGER DEFAULT CHAR_LENGTH(PALABRA);
    DECLARE AUX INTEGER DEFAULT 1;
    DECLARE AUX2 INTEGER DEFAULT CHAR_LENGTH(PALABRA);

    REPEAT
        SET RESPUESTA= CONCAT(RESPUESTA,',',SUBSTRING(PALABRA,AUX,AUX2));
        SET CONTADOR=CONTADOR-1;
        SET AUX=AUX+1;

    until CONTADOR <=0 end repeat;
    RETURN RESPUESTA;
end;

SELECT REDUCIR_CADENA('DBAII');
```



The screenshot shows a database query result window. The query is `REDUCIR_CADENA('DBAII')`. The result is a single row with the value `,DBAII,BAII,AII,II,I`. The window has a title bar with standard icons and a search bar.

	REDUCIR_CADENA('DBAII')
1	,DBAII,BAII,AII,II,I