

DEFENSA HITO 3

TAREA FINAL

Estudiante: ANA CRISTINA CALDERÓN ORTEGA

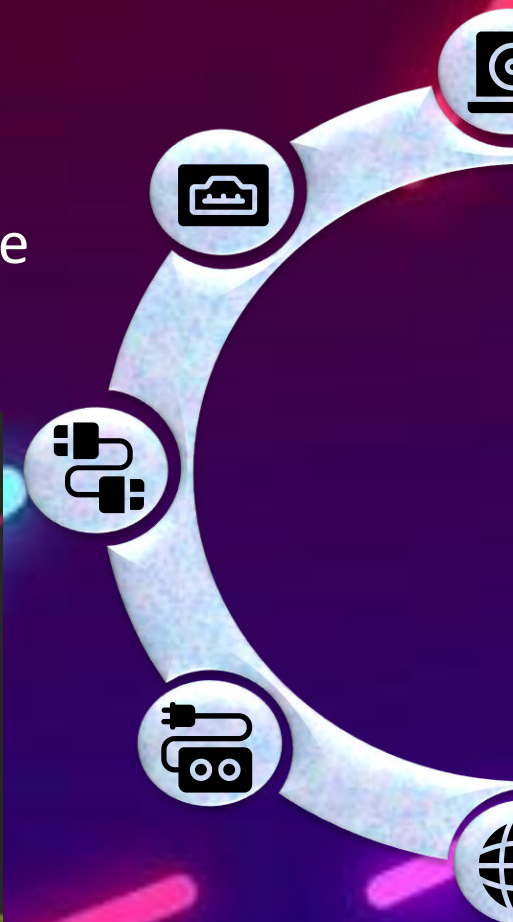
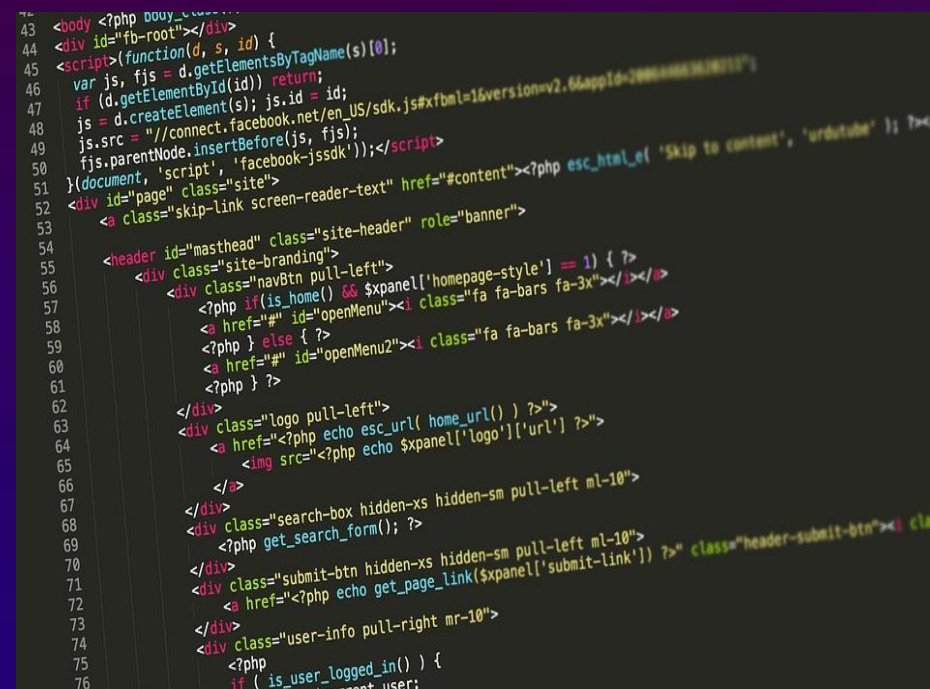
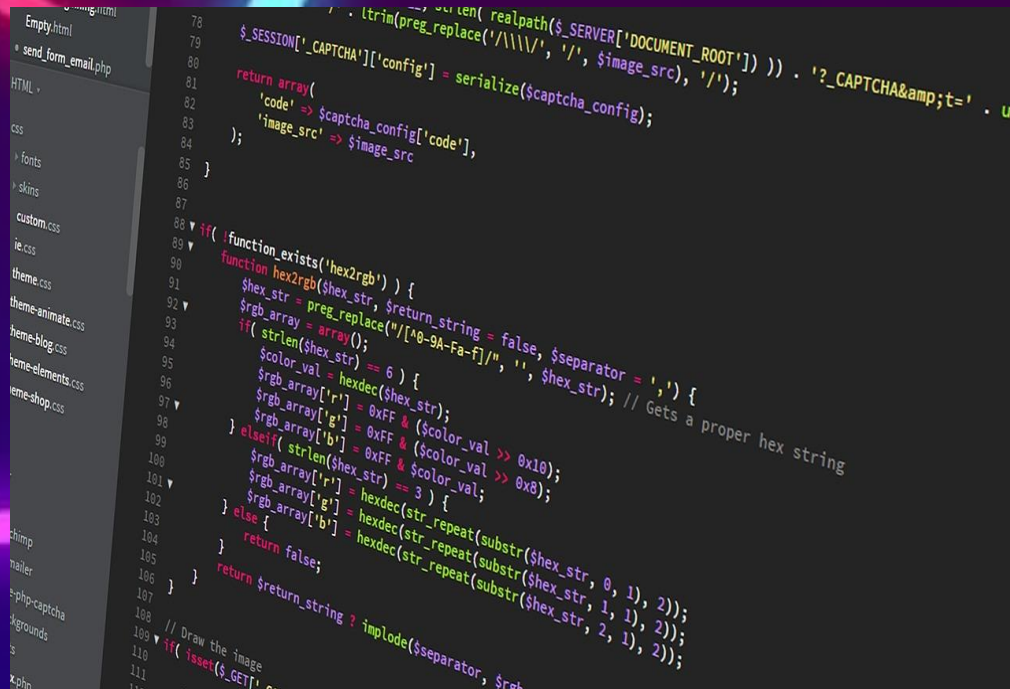
Asignatura: BASE DE DATOS II

Carrera: INGENIERÍA DE SISTEMAS

Manejo de Conceptos.

1. Defina que es lenguaje procedural en MySQL.

Conjunto de instrucciones SQL, más una serie de estructuras de control que pueden almacenarse en el servidor.



Manejo de Conceptos.

2. Defina que es una función en MySQL.

Las funciones de agregación se usan dentro de la cláusula SELECT en grupos de registros de devolver un único valor que se aplica a un grupo de registros

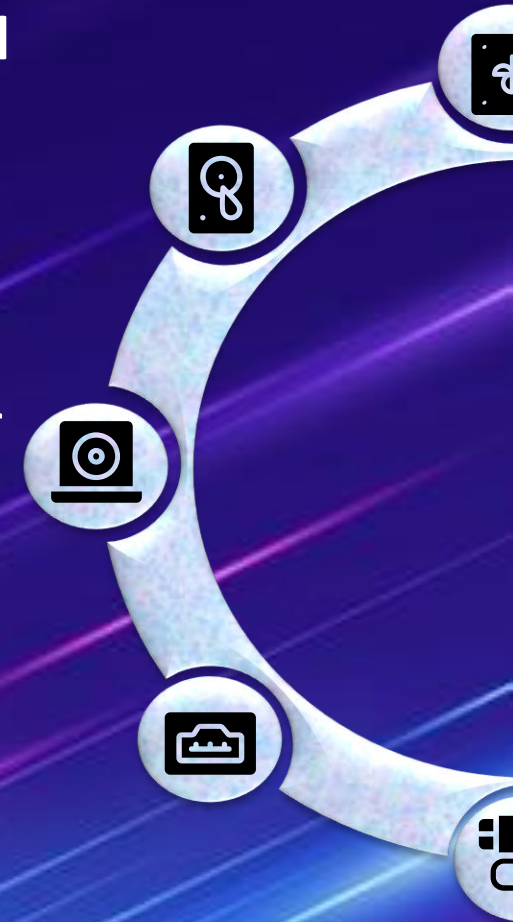


Manejo de Conceptos.

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

Una función debe estar en la clausula select cuando realizamos una consulta al crear una función propia debemos usar comando como créate funtion el nombre de la función, darle parámetros e indicar el tipo, utilizar el retorn, begin end y podemos utilizar el select, from y where también podemos crear una función y ponerla en where junto a los inner joins.

```
Para crear y alterar una función
CREATE OR REPLACE FUNCTION
max_edad_estudianes() RETURNS
intBEGIN
return
(
);
SELECT max(est.edad)
FROM estudiantes AS est
END;
```



Manejo de Conceptos.

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

Se deben usar los comando correspondientes como `create función`, `or alter` para modificar y para eliminar es `drop funtion` para eliminar la función.

```
create or alter function  
idestudiantes ()  
returns int  
begin  
return (  
    select max(est.id_est)  
    from estudiantes as est  
);  
END;
```

```
drop function idestudiantes;
```



Manejo de Conceptos.

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

La función CONCAT sirve para concatenar o unir una respuesta.

```
CREATE FUNCTION manejodeloop3(x INT, y INT, nom varchar(20))
RETURNS TEXT
BEGIN
    DECLARE str TEXT DEFAULT "";
    loop_label : LOOP
        IF x > y
        THEN
            LEAVE loop_label;
        end if;
        SET str = CONCAT(str,nom, x, ' , ');
        SET x = x + 1;
        ITERATE loop_label;
    end loop;
    RETURN str;
end;
SELECT manejodeloop3(1, 10, 'Serie de numeros: ');
```

```
`manejodeloop3(1, 10, 'Serie de numeros: ')`
1 Serie de numeros: 1 , Serie de numeros: 2 , Serie de numeros: 3 , Ser...
```



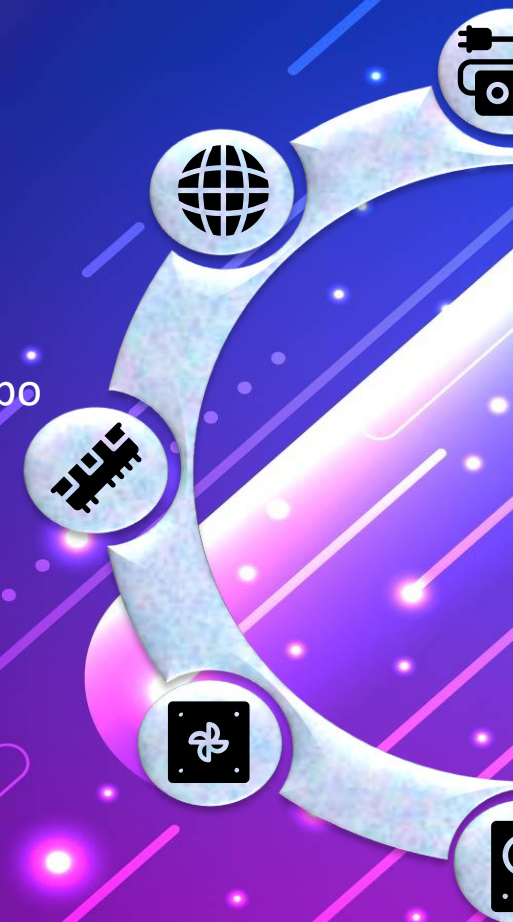
Manejo de Conceptos.

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: Ximen

SUBSTRING es una **función** de manipulación de series de caracteres **que** manipula todos los datos de tipo serie de caracteres

```
SELECT SUBSTR('Ximena Condori Mar', 1,5);
```

```
mysql> `SUBSTR('Ximena Condori Mar', 1,5)`  
1 Ximen
```



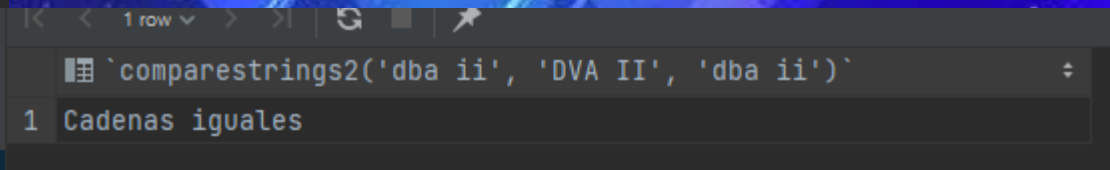
Manejo de Conceptos.

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

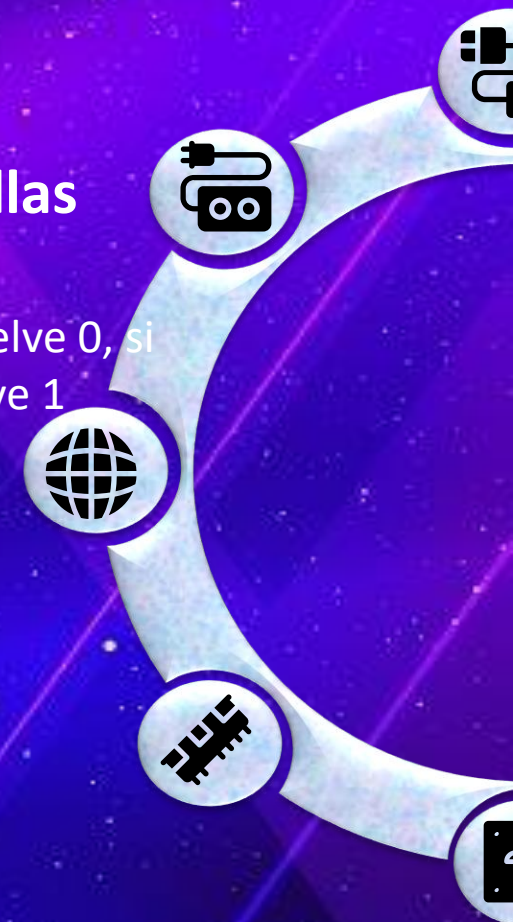
- ¿Crear una función que muestre el uso de la función STRCMP?
- La función debe comparar 3 cadenas. Y deberá determinar si dos de ellas son iguales.

La función STRCMP() en MySQL se usa para comparar dos strings. Si ambas strings son iguales, devuelve 0, si el primer argumento es más pequeño que el segundo según el orden definido, devuelve -1 y devuelve 1 cuando el segundo es más pequeño que el primero.

```
create function comparestrings2(cad1 text, cad2 text, cad3 text)
returns text
begin
    if strcmp(cad1, cad2)=0
    then
        return 'Cadenas iguales';
    else if strcmp(cad1, cad3)=0
    then
        return 'Cadenas iguales';
    else if strcmp(cad2, cad3)=0
    then
        return 'Cadenas iguales';
    else
        return 'Cadenas distintas';
    end if;
    end if;
    end if;
end;
select comparestrings2('dba ii', 'DVA II', 'dba ii');
```



The screenshot shows a MySQL query result. The query is: ``comparestrings2('dba ii', 'DVA II', 'dba ii')``. The result is a single row with the value `1 Cadenas iguales`. The interface includes navigation buttons at the top and a dropdown menu for the query.



Manejo de Conceptos.

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?

La función CHAR_LENGTH() en MySQL se usa para encontrar la longitud de una string dada (en caracteres).

La función LOCATE() en MySQL se usa para encontrar la ubicación de una substring en una string.

```
create function posicion(cadena varchar(50),  
subcadena varchar(50))  
returns INTEGER  
begin  
    declare position integer default 0;  
    set position = locate(cadena, subcadena);  
    return position;  
end;
```

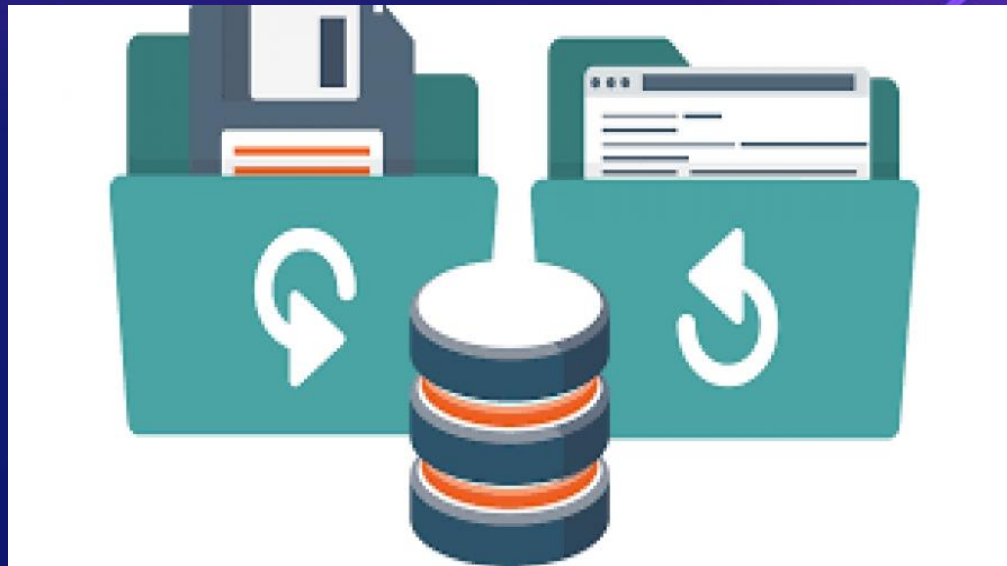
```
create function LONGITUD(cadena varchar(50))  
returns text  
begin  
    declare countString varchar(20) default ' ';  
    declare respuesta text default ' ';  
    set countString=char_length(cadena);  
  
    if (countString>7)  
    then  
        set respuesta= concat('Passed', ': ',countString);  
    else  
        set respuesta= concat('Failed', ': ',countString);  
    end if;  
    return respuesta;  
end;
```



Manejo de Conceptos.

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

En las funciones de agregación son creadas por el propio Software a comparación de las DBA que el usuario o administrador de la base puede crear para realizar tareas específicas.



Manejo de Conceptos.

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

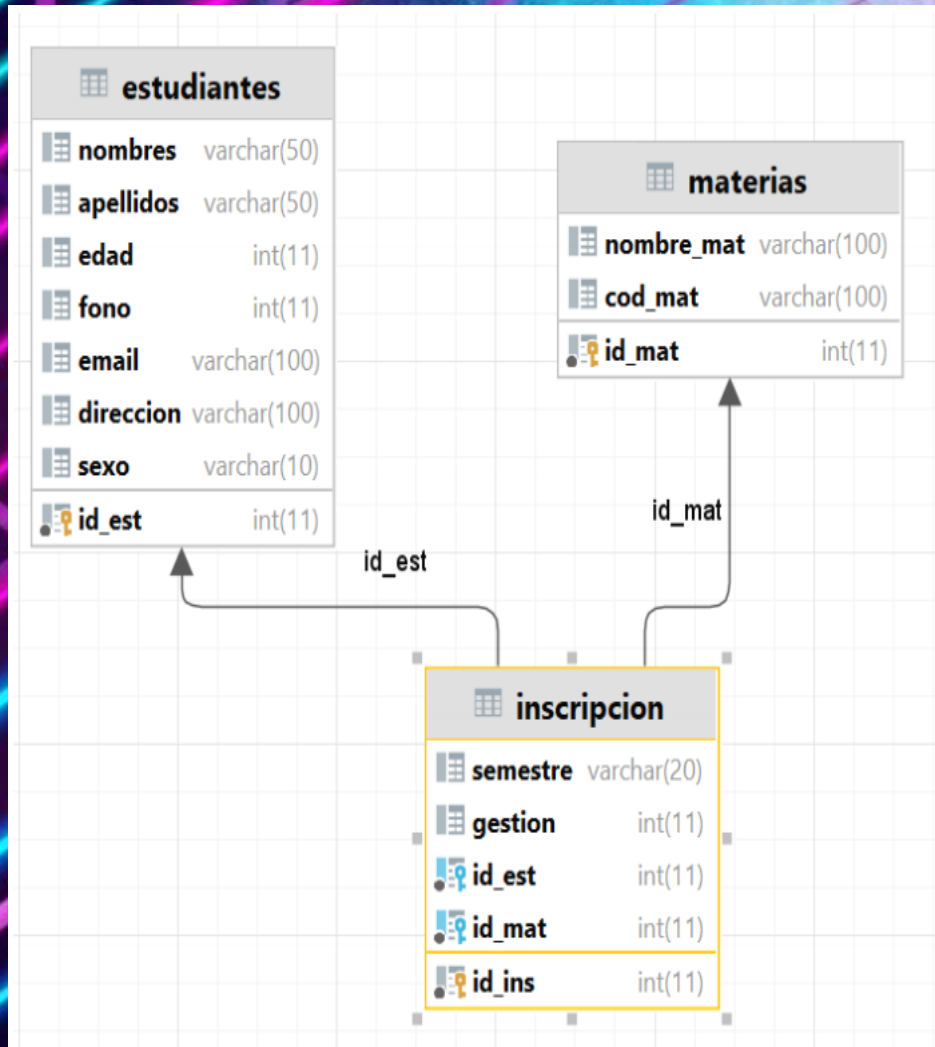
- Es decir IN INOUT, etc

Los parámetros de entrada permiten a quien realiza la llamada pasar un valor de datos a la función o al procedimiento almacenado. Los parámetros de salida permiten al procedimiento almacenado devolver un valor de datos o variable de cursor a quien realizó la llamada.



Parte Practica

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 Uria	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

Parte Practica

Crear la siguiente Base de datos y sus registros.

```
create database universidad3;
```

```
use universidad3;
```

```
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');
```

```
INSERT INTO estudiantes (nombres, apellidos, edad, fono, email, direccion, sexo)
VALUES ('Sandra', 'Mavir Uria', 25, 2832116, 'sandra@gmail.com', 'Av. 6 de Agosto', 'femenino');
```

```
INSERT INTO estudiantes (nombres, apellidos, edad, fono, email, direccion, sexo)
VALUES ('Joel', 'Adubiri Mondar', 30, 2832117, 'joel@gmail.com', 'Av. 6 de Agosto', 'masculino');
```

```
INSERT INTO estudiantes (nombres, apellidos, edad, fono, email, direccion, sexo)
VALUES ('Andrea', 'Arias Ballesteros', 21, 2832118, 'andrea@gmail.com', 'Av. 6 de Agosto', 'femenino');
```

```
INSERT INTO estudiantes (nombres, apellidos, edad, fono, email, direccion, sexo)
VALUES ('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');
```

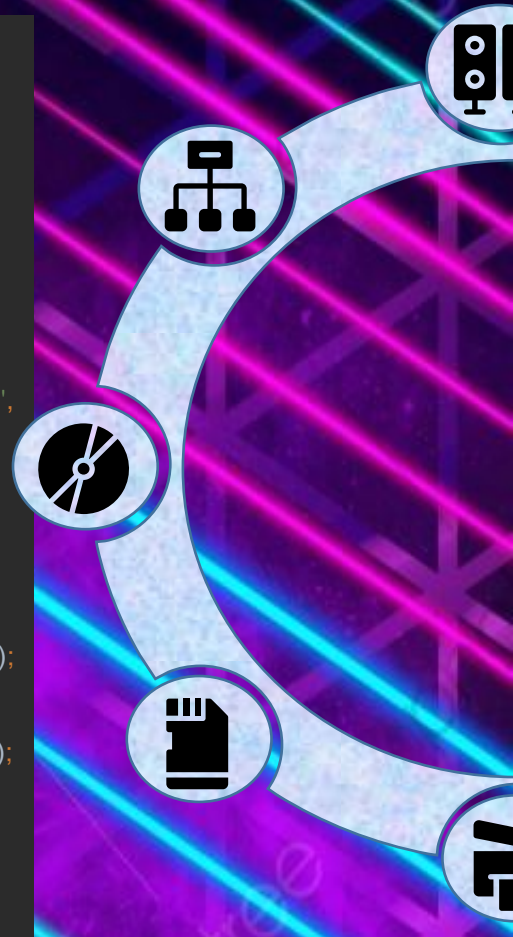
```
INSERT INTO materias (nombre_mat, cod_mat) VALUES ('Urbanismo y Diseno', 'ARQ-102');
```

```
INSERT INTO materias (nombre_mat, cod_mat) VALUES ('Dibujo y Pintura
Arquitectonica', 'ARQ-103');
```

```
INSERT INTO materias (nombre_mat, cod_mat) VALUES ('Matematica discreta', 'ARQ-104');
```

```
INSERT INTO materias (nombre_mat, cod_mat) VALUES ('Fisica Basica', 'ARQ-105');
```

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

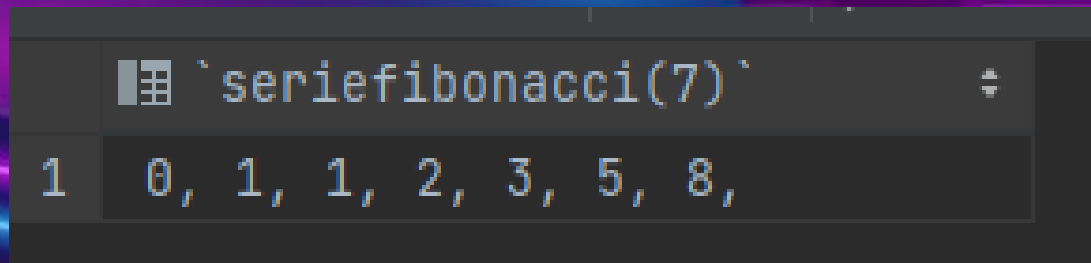


Parte Practica

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 function seriefibonacci(limite integer)
returns text
begin
  declare x integer default 0;
  declare y integer default 1;
  declare z integer default 0;
  declare cont integer default 0;
  declare resultado text default ' ';
  while(cont<limite) do
    set resultado=concat(resultado, z, ', ');
    set x=y;
    set y=z;
    set z=x+y;
    set cont=cont+1;
  end while;
  return resultado;
end;
select seriefibonacci(7);
```



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 two columns: an index column and a data column. The data column contains the string `0, 1, 1, 2, 3, 5, 8,`.

	<code>`seriefibonacci(7)`</code>
1	0, 1, 1, 2, 3, 5, 8,

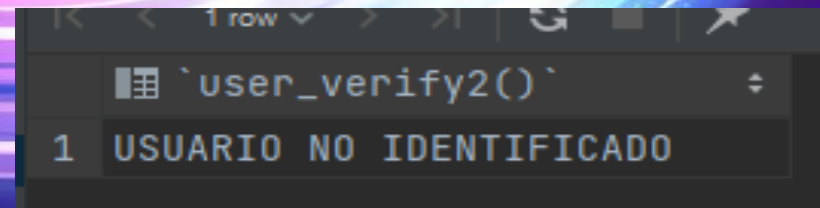


Parte Practica

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

```
set @usuario='HOLA';
create function user_verify2()
returns text
begin
    declare respuesta text default "";
    case @usuario
        when 'ADMIN' then SET respuesta='Usuario
ADMIN';
        when 'GUEST' then SET respuesta= 'Usuario
INVITADO';
        else SET respuesta='USUARIO NO
IDENTIFICADO';
    end CASE;
    return respuesta;
end;
SELECT user_verify2();
```



The screenshot shows a SQL query result window with a single row of data. The query is `user_verify2()`. The result is `1 USUARIO NO IDENTIFICADO`.

	user_verify2()
1	USUARIO NO IDENTIFICADO

Parte Practica

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

○ Previamente deberá de crear una función que obtenga la edad mínima de los estudiantes

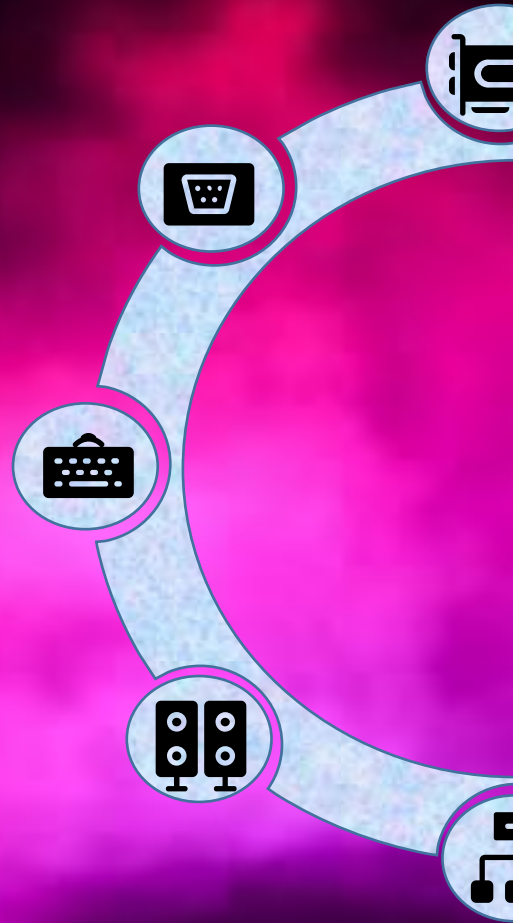
- La función no recibe ningún parámetro.
- La función debe de retornar un número. (LA EDAD MÍNIMA).

```
create function minedad()
returns int
begin
  declare respuesta int default 0;
  set respuesta=
    (select MIN(est.edad)
     from estudiantes as est);
  return respuesta;
end;
select minedad();
```

```
create function paredadmin()
returns text
begin
  declare minedad int default minedad();
  declare respuesta text default "";
  declare x int default 0;
  if (minedad%2=0)
  then
    while x<=minedad do
      set respuesta=concat(respuesta,x,',');
      set x=x+2;
    end while;
    return respuesta;
  else
    while minedad>=x do
      set respuesta=concat(respuesta,minedad,',');
      set minedad=minedad-2;
    end while;
    return respuesta;
  end if;
end;
select paredadmin();
```

```
`paredadmin()`
```

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



Parte Practica

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.

```
create function Vocales(cadena text)
returns text
begin
  declare x int default 1;
  declare y text default '';
  declare z int default char_length(cadena);
  declare l char default '';
  declare A int default 0;
  declare E int default 0;
  declare I int default 0;
  declare O int default 0;
  declare U int default 0;
  while x <= z
  do
    set l = substring(cadena,x, 1);
    if l = 'a'
    then
      set
        A = A + 1;
    else if l = 'e'
    then
      set
        E = E + 1;
```

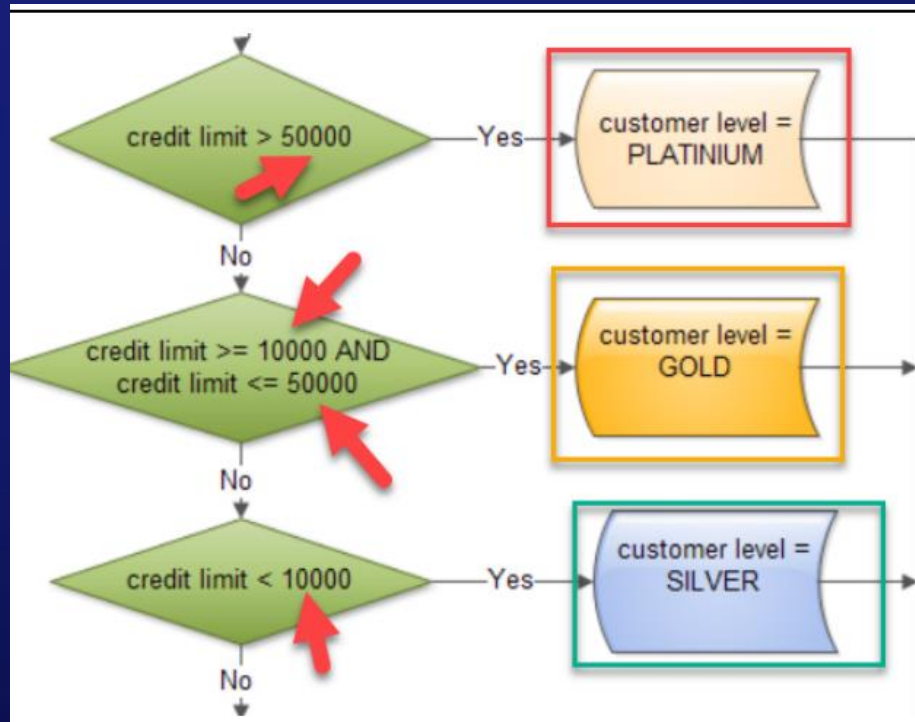
```
    else if l = 'i'
    then
      set
        I = I + 1;
    else if l = 'o'
    then
      set
        O = O + 1;
    else if l = 'u'
    then
      set
        U = U + 1;
    end if;
  end if;
  end if;
  end if;
  end if;
  set x = x + 1;
end while;
set y = concat('a:', A, ', ', 'e:', E, ', ', 'i:', I, ', ', 'o:', O, ', ', 'u:', U);
return (y);
end;
select Vocales('Taller de base de datos') as VOCALES_SEPARABLES;
```

```
VOCALES_SEPARABLES
1 a:3 e:4 i:0 o:1 u: 0
```

Parte Practica

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:

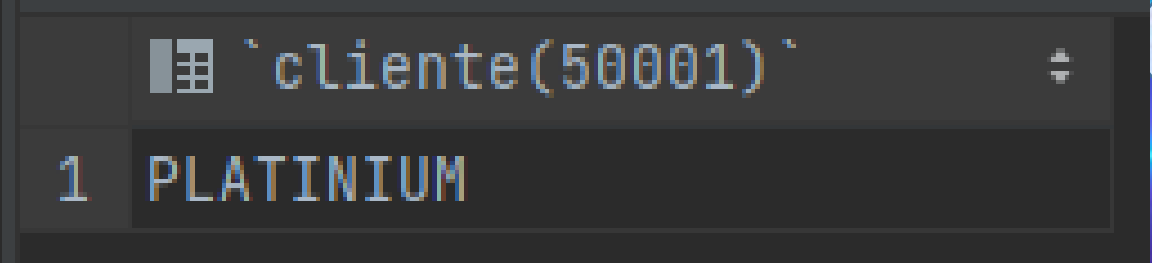


Parte Practica

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:

```
create function cliente (credit_number integer)
returns text
begin
  declare respuesta text default ' ';
  case
    when credit_number > 50000 then set respuesta = 'PLATINIUM';
    when credit_number >= 10000 and credit_number <= 50000 then set
respuesta = 'GOLD';
    when credit_number <= 10000 then set respuesta = 'SILVER';
  end case;
  return respuesta;
end;
select cliente(50001);
```

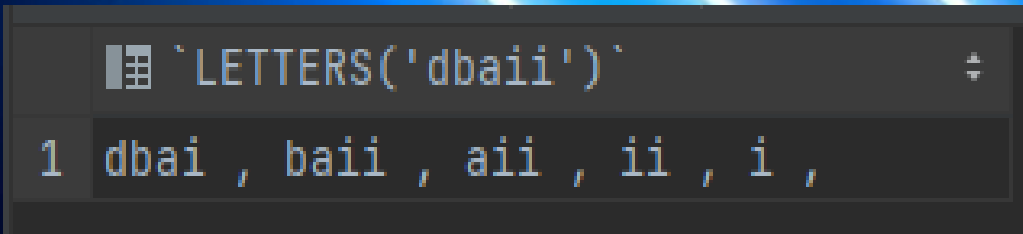
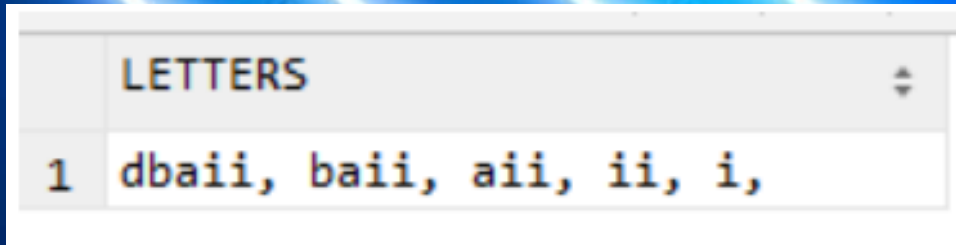


```
`cliente(50001)`
1 PLATINIUM
```

Parte Practica

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:



```
create function LETTERS(cadena text)
returns text
begin
    declare x text default "";
    declare y int default char_length(cadena);
    declare z int default 1;
    declare l int default y;
    repeat
        if (y >= z)
        then
            set x = concat (substr(cadena, y, l-1), ', ', x);
            set y = y- 1;
        end if;
    until y <= 0
    end repeat;
    return(x);
end;
select LETTERS('dbaii');
```

