



Defensa Hito 3

Tarea final

MARTIN JOSUE MAMANI PILCO



Parte teórica

MANEJO DE CONCEPTOS

1. Defina que es lenguaje procedural en MySQL

Es un tipo de programación que consiste en estructurar el código en diferentes componentes, que en el caso de MySQL serian las funciones que podemos utilizar y crear.

2. Defina que es una función en MySQL

Una función es un procedimiento que o bien podemos utilizar los predeterminados o crear uno que realice lo que codifiquemos nosotros.

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

Una función posee varios elementos necesarios que vienen desde el nombre, que se coloca después de “create function”, además debe tener ciertos parámetros como el return que establece el tipo de variable que va a retornar la función, para posteriormente incluir el proceso que indica lo que va a realizar la función. Finalmente se debe cerrar con un return para determinar la variable que va a devolver la función.

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

Para crear una función se debe seguir la siguiente estructura donde se debe poner el nombre, el tipo de variable a retornar y el desarrollo junto a la variable que se va a mostrar.

```
create function ejemplo ()  
  returns text  
begin  
  declare variable text default ' '  
  set variable = 'Creacion de una funcion';  
  return variable;  
end;
```

Para modificar una función se debe utilizar la siguiente estructura, donde al lado de la palabra “create” debemos adicionar la palabra “replace” para realizar modificaciones a una función ya existente añadiendo o cambiando lo que deseamos.

```
create or replace function ejemplo ()  
  returns text  
begin  
  declare variable text default ' '  
  set variable = 'Modificacion de una funcion';  
  return variable;  
end;
```

Y finalmente para eliminar una función debemos colocar “drop function” junto con el nombre de la función que queremos eliminar

```
drop function ejemplo;
```

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

La función CONCAT sirve para concatenar caracteres como letras o números, funciona escribiendo SELECT CONCAT seguido de un paréntesis donde van los elementos a concatenar separados por comas.

```
select concat('Ejemplo',' de',' concatenacion');
```

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

La función substring sirve para seleccionar una determinada cantidad de caracteres de un texto, colocando entre paréntesis primero el texto, luego el numero de la posición desde donde se va a seleccionar y finalmente el numero de cuantas posiciones va a recorrer.

```
select substr('Hola mundo', 2, 2);
```


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

La función STRCMP funciona para comparar 2 cadenas, si son iguales retorna un numero 1 y si no son iguales retorna 0.

```
select strcmp('i','dba ii');
```


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

La función CHAR_LENGTH sirve para contar los caracteres de una cadena, devolviendo un numero con la cantidad de caracteres que hay.

```
select char_length('hola');
```

La función LOCATE funciona para buscar una cadena en otra devolviendo su posición desde donde se encuentran los caracteres en la segunda cadena.

```
select locate('bar', 'fobarbar');
```



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 realizan acciones predeterminadas sin posibilidad a cambiarlas, además de que solo pueden ser utilizadas en la clausula "SELECT".

Las funciones creadas por el DBA pueden realizar varias tareas, no recibir o recibir varios parámetros además de que pueden llamar a otras funciones para añadirlas a sus procesos. Estas pueden ser utilizadas tanto en la clausula "SELECT" como en la clausula "WHERE"



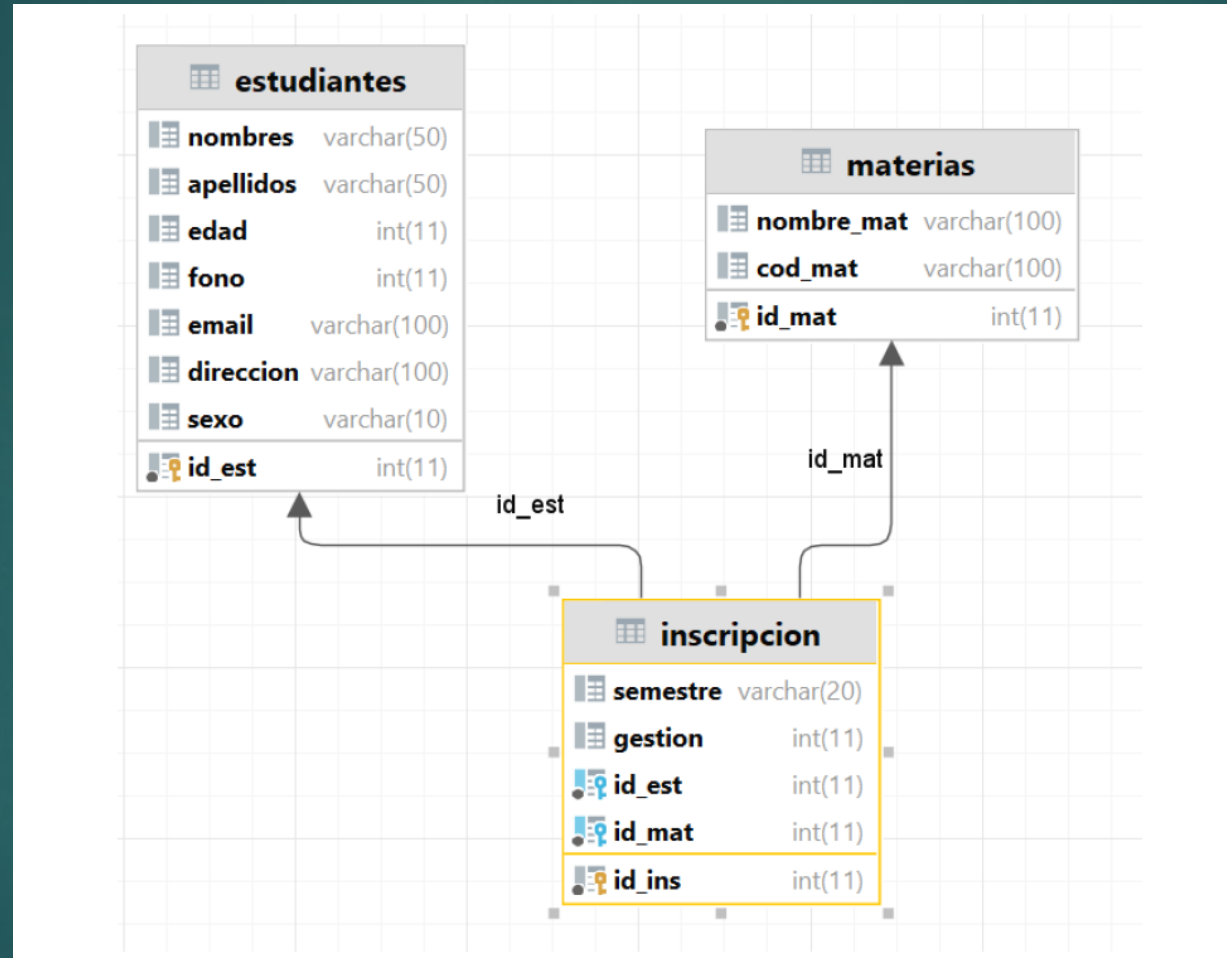
10.¿Busque y defina a qué se referirá cuando se habla de parámetros de entrada y salida en MySQL?

Los parámetros de entrada son las variables que se implementan cuando se crea una función y los parámetros de salida es el return que es la variable que devuelve la función.



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 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

estudiantes	
 id_est	int(11)
nombres	varchar(30)
apellidos	varchar(50)
edad	int(11)
fono	int(11)
email	varchar(100)
direccion	varchar(100)
sexo	varchar(10)

materias	
 id_mat	int(11)
nombre_mat	varchar(100)
cod_mat	varchar(100)

inscripcion	
semestre	varchar(20)
gestion	int(11)
 id_est	int(11)
 id_mat	int(11)

id_est

id_mat

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

	seme...	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

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

12. Crear una función que genere la serie Fibonacci.

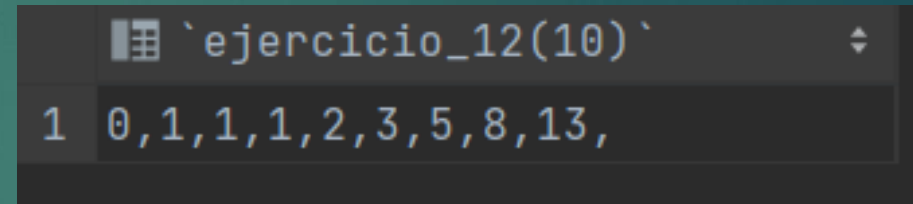
```
create or replace function ejercicio_12(lim int)
returns text
begin
  declare x int default 0;
  declare y int default 1;
  declare z int default 0;
  declare aux int default 1;
  declare respuesta text;

  if lim >= 1
  then
    set respuesta = concat(x, ',');
  end if;

  if lim >= 2
  then
    set respuesta = concat(respuesta, y, ',');
  end if;

  if lim >= 3
  then
    while x <= (lim - 2) do
      set z = x + y;
      set respuesta = concat(respuesta, y, ',');
      set x = y;
      set y = z;
      set aux = aux + 1;
    end while;
  end if;
  return respuesta;
end;

select ejercicio_12(10);
```



The screenshot shows a SQL query result in a dark-themed interface. The query is ``ejercicio_12(10)``. The result is displayed in a table with two columns: an identifier '1' and the output string '0,1,1,1,2,3,5,8,13,'. The output string represents the first 10 terms of the Fibonacci sequence, with a trailing comma.

1	0,1,1,1,2,3,5,8,13,

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

```
create function variableglobal()
```

```
returns int
```

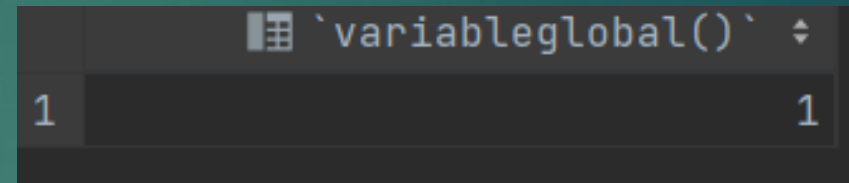
```
begin
```

```
    set @variable = 1;
```

```
    return @variable;
```

```
end;
```

```
select variableglobal();
```



The screenshot shows a SQL query result in a dark-themed interface. The query is ``variableglobal()``. The result is a single row with the value `1`.

	<code>`variableglobal()`</code>
1	1

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

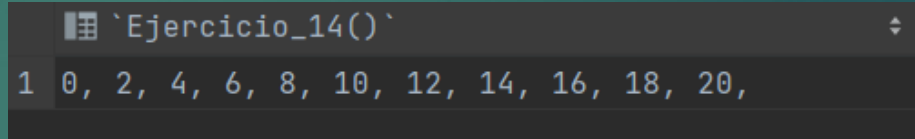
```
create function EdadMinima ()
returns int
begin
    declare est int default 0;
    set est = (select min(est.edad)
    from estudiantes as est);
    return est;
end;

select EdadMinima();

create function funtion1()
returns text
begin
    declare edad int default 0;
    declare cont int default 0;
    declare res text default '';

    set edad = EdadMinima();
    if (edad % 2 = 0)
    then
        while (cont <= edad) do
            # PAR
            if(cont % 2 = 0)
            then
                set res = concat(res, cont, ', ');
            end if;
            set cont = cont + 1;
        end while;
    else
        set cont = edad;
        while (cont >= 0) do
            if(cont % 2 = 1)
            then
                set res = concat(res, cont, ', ');
            end if;
            set cont = cont - 1;
        end while;
    end if;
    return res;
end;

select funtion1();
```



The screenshot shows a SQL query result for the function 'Ejercicio_14()'. The result is a single row with the value '0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20,'.

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

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

```
create or replace function cuentavocales(cadena text)
returns text
begin
  declare x int default 1;
  declare res text default '';
  declare limite int default char_length(cadena);
  declare letra char default '';
  declare numA int default 0;
  declare numE int default 0;
  declare numI int default 0;
  declare numO int default 0;
  declare numU int default 0;
  while x <= limite
  do
    set letra = substring(cadena,x, 1);
    if letra = 'a'
    then
      set
        numA = numA + 1;
    else if letra = 'e'
    then
      set
        numE = numE + 1;
    else if letra = 'i'
    then
      set
        numI = numI + 1;
    else if letra = 'o'
    then
      set
        numO = numO + 1;
    else if letra = 'u'
    then
      set
        numU = numU + 1;
    end if;
    end if;
    end if;
    end if;
    end if;
    set x= x + 1;
  end while;
  set res = concat( 'a:', numA, ', ', 'e:', numE, ', ', 'i:', numI, ', ', 'o:', numO, ', ', 'u:', numU);
  return (res);
end;
```

```
select cuentavocales( 'Practica de base de datos');
```

```
`cuentavocales('Practica de base de datos')`  
1 a:4 e:3 i:1 o:1 u: 0
```

16. Crear una función que recibe un parámetro INTEGER.

```
create or replace function usuario(credit_number INT)
```

```
returns text
```

```
begin
```

```
  declare respuesta text default '';
```

```
  set respuesta = (
```

```
    select case
```

```
      when credit_number > 50000 then 'PLATINIUM'
```

```
      when credit_number >= 10000 && credit_number <= 50000 then 'GOLD'
```



```
      when credit_number < 10000 then 'SILVER'
```

```
    end);
```

```
  return respuesta;
```

```
end;
```

```
select usuario(20000);
```

	 `usuario(20000)`	
1	GOLD	

17. Crear una función que reciba un parámetro TEXT

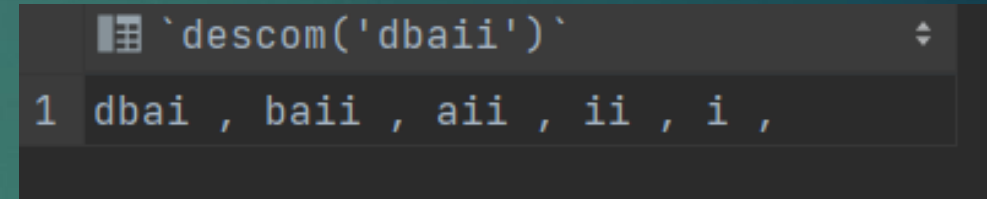
```
create or replace function descom(num text)
returns text
begin

  declare res text default '';
  declare concatenar int default char_length(num);
  declare lim int default 1;
  declare y int default concatenar;

  repeat
    if concatenar >= lim

      then
        set res = concat(substr(num, concatenar, y-1), ', ', res);
        set concatenar = concatenar - 1;
      end if;
    until concatenar <= 0
    end repeat;
  return(res);
end;

select descom('dbaii');
```



The screenshot shows a SQL query result in a dark-themed interface. The query is ``descom('dbaii')``. The result is a single row with the value `1 dbai , baii , aii , ii , i ,`.

	`descom('dbaii')`
1	dbai , baii , aii , ii , i ,