

Defensa Hito 3

Base de Datos II



01

PARTE TEÓRICA

Manejo de Conceptos





1. Defina que es lenguaje procedural en MySQL.

```
1 • use employees;  
2  
3 • select count(*)  
4   into @numEmpleados  
5   from dept_emp  
6   where dept_no = 'd005';  
7  
8 • select @numEmpleados;  
9
```

Result Grid Filter Rows:

#	@numEmpleados
1	85707

Los lenguajes procedimentales son programación a nivel de base de datos, para poder programar a nivel de base de datos debemos entender conceptos como procedimientos integrados y funciones en MySQL

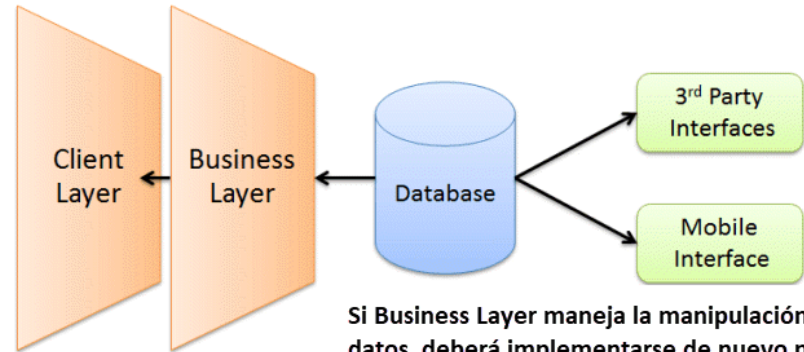


2. Defina que es una función en MySQL.

Una función en MySQL es una pieza de código de lenguaje procesal que devuelve un valor.

¿Por qué usar las funciones?

El uso de la capa empresarial para la manipulación de datos aumentará la carga en el tráfico de red



Si Business Layer maneja la manipulación de datos, deberá implementarse de nuevo para otras interfaces, lo que aumentará el reproceso y el riesgo de inconsistencia de los datos.

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



Al crear una función en MySQL, debe ingresar un nombre único para identificar la función, en el retorno decimos qué valor devolveremos, los parámetros son los datos de la función que necesitamos, pueden o no ser importantes, dependiendo de, en el retorno donde devolvemos un valor.

```
CREATE FUNCTION function_name [ (parameter datatype [, parameter datatype]) ]  
RETURNS return_datatype  
  
BEGIN  
  
    declaration_section  
  
    executable_section  
  
END;
```



Para crear una función, debemos ingresar "CRÉAR FUNCIÓN" {nombre de la función}, un RETURN, BEGIN y END. Para modificar una función, debemos colocarla al lado de "CREATE OR REPLACE FUNCTION". Eliminar es "DROP FUNCTION"

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

```
#como crear una funcion
```

```
CREATE FUNCTION nombre();
```

```
#como modificar una funcion
```

```
CREATE OR REPLACE FUNCTION nombre();
```

```
#como eliminar una funcion
```

```
DROP FUNCTION nombre;
```

```
DROP FUNCTION IF EXISTS nombre;
```

Ejemplo de Funciones:

```
#un ejemplo de funciones
CREATE OR REPLACE FUNCTION num_mayor(n INT)
RETURNS TEXT
BEGIN
    DECLARE respuesta TEXT DEFAULT '';
    DECLARE limite INT DEFAULT 30;

    IF (n > limite)
    THEN
        SET
            respuesta = CONCAT('El numero: ', n , ' es mayor al limite');
    ELSE
        SET
            respuesta = CONCAT('El numero: ', n , ' no es mayor al limite');
    end if;

    RETURN (respuesta);
end;

SELECT num_mayor( n: 100) AS mayor_que;
```

Output mayor_que_text

1 row

mayor_que
1 El numero: 100 es mayor al limite

mayor_que

1 El numero: 10 no es mayor al limite

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.



```
create or replace function getNombreCompleto(nombre varchar(100),apellidos varchar(100), edad INT)
returns varchar(200)
begin
  declare nombreCompleto varchar(200);
  set nombreCompleto = CONCAT(nombre, ' ', apellidos, ' ; edad: ', edad, ' años');
  return nombreCompleto;
end;

select getNombreCompleto( nombre: est.nombres, apellidos: est.apellidos, edad: est.edad) as persona
from estudiantes as est;
```

La función "Concat" puede concatenar múltiples variables y cadenas

persona	
1	Miguel Gonzales Veliz ; edad: 20 años
2	Sandra Mavir Uria ; edad: 25 años
3	Joel Adubiri Mondar ; edad: 30 años
4	Andrea Arias Ballesteros ; edad: 21 años
5	Santos Montes Valenzuela ; edad: 24 años

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



La función "SUBSTRING" sirve para establecer un rango en las palabras de una cadena

```
37
38 ✓ SELECT SUBSTR('Ximena Condori Mar', 1,6);
39
```

concatenar_text

Output × SUBSTR('Ximena Condori Mar', 1,6):varchar ×

1 row

5 ms

1	Ximena
---	--------



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.

```
SELECT STRCMP('HOLA', 'HOLA') AS compara_cadenas;
```

Output x compara_cadenas:tinyint x

1 row


	compara_cadenas
1	0

Compara cadenas y
retorna un valor
numerico

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

si retorna 0 = Significa que son iguales
si retorna -1 = Significa que son distintos
si retorna 1 = Significa que son distintos

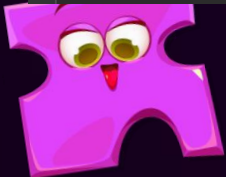





```
CREATE OR REPLACE FUNCTION comparaCadenas (cadena1 TEXT, cadena2 TEXT, cadena3 TEXT)
RETURNS TEXT
BEGIN
    IF (strcmp(cadena1, cadena2)=0) and (strcmp(cadena1, cadena3)=0)
        THEN
            RETURN ('DOS DE LAS CADENAS SON IGUALES');
        ELSE
            RETURN ('DOS DE LAS CADENAS SON DISTINTAS');
        END IF;

    IF (strcmp(cadena1, cadena3)=0)
        THEN
            RETURN ('DOS DE LAS CADENAS SON IGUALES');
        ELSE
            RETURN ('DOS DE LAS CADENAS SON DISTINTAS');
        END IF;
    END;

SELECT comparaCadenas( cadena1: 'dba II', cadena2: 'dba II', 'dba III') AS comparacadenas;
```



 comparacadenas

1 DOS DE LAS CADENAS SON DISTINTAS

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 mide el tamaño de una cadena
- La función Locate encuentra en que posición se encuentra una letra



```
SELECT CHAR_LENGTH('HOLA MUNDO') AS cadenas;
```

cadenas	
1	10



```
SELECT LOCATE('M', 'HOLA MUNDO') AS locate;
```

locate	
1	6

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



Las funciones de agregación ya se encuentran predeterminada y se la ejecuta con la cláusula SELECT.

Las funciones creadas por el DBA se utilizan para realizar tareas o ejecutas datos específicos; y se ejecuta con la cláusula SELECT y WHERE.



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



INPUT: Los parámetros de entrada son los parámetros definidos al principio de la función, son las variables requeridas por la función.

OUTPUT: Los parámetros de salida son aquellas variables utilizadas en la función y devuelve un valor por el "return"





PARTE PRACTICA

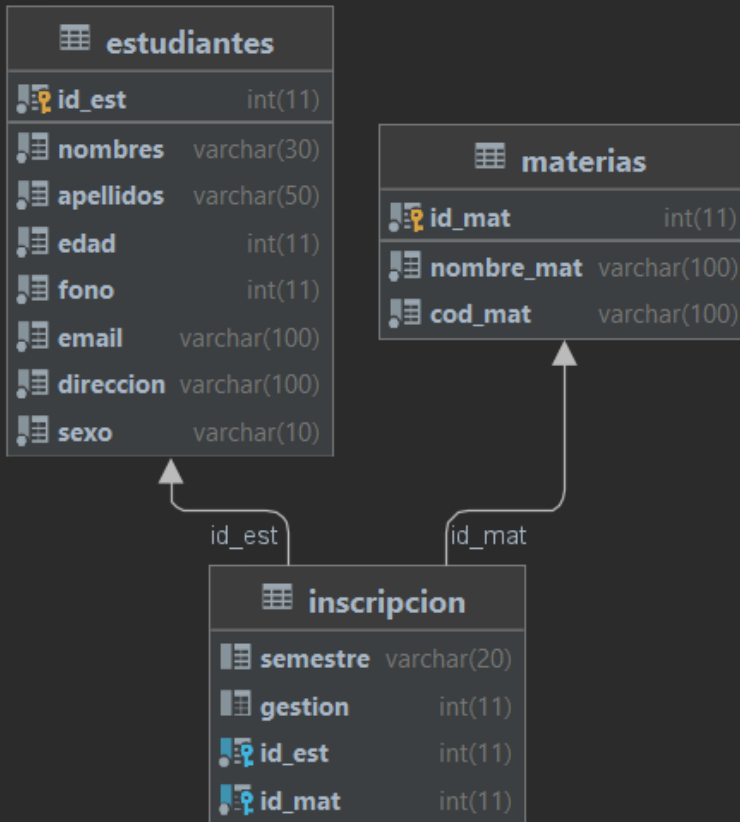
Resolución de Ejercicios



02

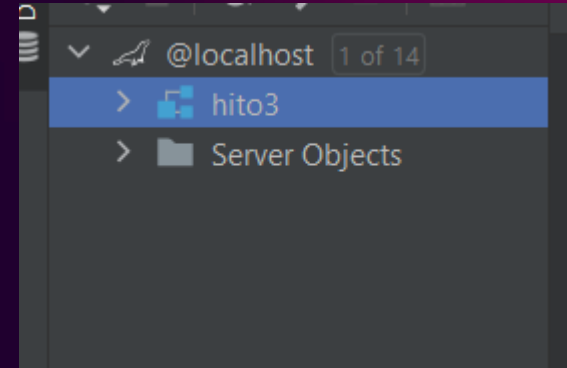


n. Crear la siguiente Base de datos y sus registros.



Creando la base de datos:

```
CREATE DATABASE Hito3;
```




Posicionándonos en la base de datos:

```
1 CREATE DATABASE
2 USE Hito3;
3
```



Tabla estudiantes



	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

```
CREATE TABLE estudiantes
(
    nombres VARCHAR (30) NOT NULL,
    apellidos VARCHAR (50) NOT NULL,
    edad INT NOT NULL,
    fono INT NOT NULL,
    email VARCHAR (100) NOT NULL,
    direccion VARCHAR (100) NOT NULL,
    sexo VARCHAR (10) NOT NULL,
    id_est INT AUTO_INCREMENT PRIMARY KEY NOT NULL
);
```





Tabla inscripción

```
CREATE TABLE inscripcion (  
    semestre VARCHAR (20),  
    gestion INT,  
  
    id_est INT NOT NULL,  
    FOREIGN KEY (id_est) REFERENCES estudiantes(id_est),  
  
    id_mat INT NOT NULL,  
    FOREIGN KEY (id_mat) REFERENCES materias(id_mat)  
);
```





	 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





Tabla materias

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

WHERE		ORDER BY	
	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.

- ❖ 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 fibonnaci(limite INT)
RETURNS TEXT
BEGIN
    DECLARE x INT DEFAULT 0;
    DECLARE y INT DEFAULT 1;
    DECLARE z INT DEFAULT 0;
    DECLARE w INT DEFAULT 1;

    DECLARE respuesta TEXT;

    IF limite >= 1
    THEN
        SET
            respuesta = CONCAT(x, ' ', y);
    end if;

    IF limite >= 2
    THEN
        SET
            respuesta = CONCAT(respuesta, ' ', z, ' ', w);
    end if;
```

```
IF limite >= 3
  THEN
    WHILE x < (limite - 2 )
      DO
        SET
          z = x + y;
        SET
          respuesta = CONCAT(respuesta, ' ', z, ' ', '');
        SET
          x = y;
        SET
          y = z;
        SET
          w = w + 1;
      end while;
    end if;

    RETURN (respuesta);
end;

SELECT fibonacci( limite: 7) AS serie_fibonnaci;
```

1 row	
serie_fibonnaci	
1	0 , 1 , 1 , 2 , 3 , 5 , 8 ,





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

Crear una función cualquiera.

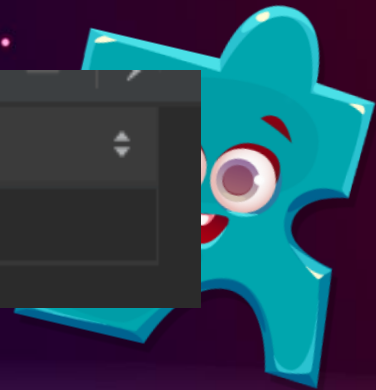
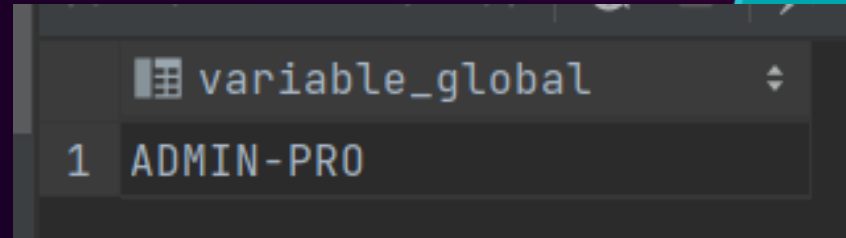
La función debe retornar la variable global.

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

```
SET @admin = 'ADMIN-PRO';

CREATE FUNCTION variable_global ()
RETURNS TEXT
BEGIN
    RETURN @admin;
end;

SELECT variable_global() AS variable_global;
```





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).

```
CREATE FUNCTION edad_min ()  
RETURNS INT  
BEGIN  
    DECLARE res INT DEFAULT 0;  
  
    SET  
        res = (  
            SELECT MIN(est.edad)  
            FROM estudiantes AS est  
        );  
    RETURN res;  
end;  
  
✓ SELECT edad_min() AS minima_edad;
```

minima_edad	
1	20



Si la edad mínima es PAR mostrar todos los pares empezando desde 0 a este ese valor de la edad mínima.

- Si la edad mínima es IMPAR mostrar descendentemente todos los impares hasta el valor 0.
 - 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 FUNCTION serie_edad_min()
RETURNS TEXT
BEGIN
    DECLARE x INT DEFAULT 0;
    DECLARE cont INT DEFAULT 0;
    DECLARE respuesta TEXT DEFAULT '';

    SET x = edad_min();

    IF (x % 2 = 0)
    THEN
        WHILE (cont <= x)
        DO
            IF (cont % 2 = 0)
            THEN
                SET
                    respuesta = CONCAT(respuesta, ' ', cont, ' , ');
            end if;
            SET
                cont = cont + 1;
            end while;
```



```
ELSE
    SET
        cont = x;
    WHILE (cont >= x)
        DO
            IF (cont % 2 = 1)
                THEN
                    SET
                        respuesta = CONCAT(respuesta, ' ', cont, ' ', ');
                end if;
            SET
                cont = cont - 1;
            end while;
        end if;

    RETURN respuesta;
end;
```

```
SELECT serie_edad_min() AS series;
```

series	
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.

```
CREATE OR REPLACE FUNCTION cuenta_vocales(cadena TEXT)  
RETURNS TEXT
```

```
BEGIN
```

```
    DECLARE res TEXT DEFAULT '';
```

```
    DECLARE lim INT DEFAULT CHAR_LENGTH(cadena);
```

```
    DECLARE vocal TEXT DEFAULT '';
```

```
    DECLARE x INT DEFAULT 1;
```

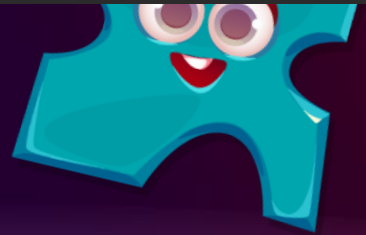
```
    DECLARE contA INT DEFAULT 0;
```

```
    DECLARE contE INT DEFAULT 0;
```

```
    DECLARE contI INT DEFAULT 0;
```

```
    DECLARE contO INT DEFAULT 0;
```

```
    DECLARE contU INT DEFAULT 0;
```





```
WHILE x <= lim
DO
    SET
        vocal = SUBSTR( cadena , x, 1);

        IF vocal = 'a'
        THEN
            SET
                contA = contA + 1;

        ELSE IF vocal = 'e'
        THEN
            SET
                contE = contE + 1;

        ELSE IF vocal = 'i'
        THEN
            SET
                contI = contI + 1;

        ELSE IF vocal = 'o'
        THEN
            SET
                contO = contO + 1;
```





```
ELSE IF vocal = 'u'
  THEN
    SET
      contU = contU + 1;

end if;
end if;
end if;
end if;
end if;

SET x = x + 1;
end while;
SET
  res = CONCAT(' a: ', contA, ', ', ' e: ', contE, ', ', ' i: ', contI, ', ', ' o: ', contO, ', ', ' u: ', contU);

RETURN (res);

end;

SELECT cuenta_vocales( cadena: 'taller de base de datos') AS SEPARA_VOCALES;
```

1 row	
SEPARA_VOCALES	
1	a: 3 , e: 4 , i: 0 , o: 1 , 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.

```
CREATE FUNCTION clase_usuario( num INT)
RETURNS TEXT
BEGIN
    DECLARE credit_number INT DEFAULT 0;
    DECLARE respuesta TEXT DEFAULT '';

    SET credit_number = num;

    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';

    ELSE SET respuesta = 'NO EXISTE EL USUARIO';
    END CASE;

    RETURN (respuesta);
end;

SELECT clase_usuario( num: 50001) AS Tipos_de_usuario;
```

Tipos_de_usuario	
1	PLATINIUM

Tipos_de_usuario	
1	GOLD

Tipos_de_usuario	
1	SILVER



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


```
CREATE OR REPLACE FUNCTION descomponiendo_cadenas (cadena TEXT)
RETURNS TEXT
BEGIN
    DECLARE respuesta TEXT DEFAULT '';
    DECLARE cad INT DEFAULT CHAR_LENGTH(cadena);
    DECLARE num INT DEFAULT cad;
    DECLARE limite INT DEFAULT 1;

    REPEAT
        IF
            cad >= limite

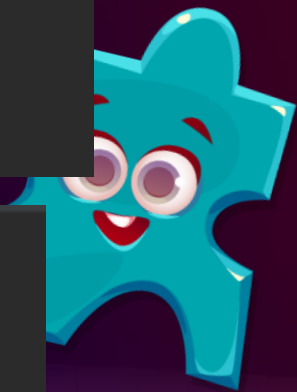
        THEN
            SET
                respuesta = CONCAT(SUBSTR(cadena, cad, num -1), ' ', respuesta );
            SET
                cad = cad - 1;
        end if;
    until cad <= 0
    end repeat;

    RETURN (respuesta);
end;

SELECT descomponiendo_cadenas( cadena: 'dbaii') AS cadenas;
```

 cadenas

1	dbai	,	baii	,	aii	,	ii	,	i	,
---	------	---	------	---	-----	---	----	---	---	---



GRACIAS POR VER

