Facilitador(a): Ing. Carmen Ortega Asignatura: Base de Datos II

Fecha: 28/10/2020 Grupo: \_\_\_1IL131

Estudiantes: Jacky He, 3-740-114

Yui Lo, 8-929-854

Maria José Martínez PE-15-1146

Dhiraj Lakhwani 3-743-1798

Richie Ieong 8-970-668

1. **TÍTULO** **DE LA EXPERIENCIA**: Crear procedimientos y funciones invocados MySQL
2. **TEMAS:**

En este ejercicio se utilizarán todas las instrucciones para crear procedimientos y funciones invocados por SQL

1. **OBJETIVO(S):**

Aplicar conceptos para crear procedimientos y funciones invocados por SQL y aplicarlo en un ejemplo práctico.

La actividad le permite al estudiante enriquecer su experiencia en el desarrollo y construcción de conocimientos del curso de Base de Datos II.

1. **METODOLOGÍA:**
2. Divida el salón en grupos.
3. Distribuya el taller a cada grupo.
4. Cada equipo contará con un período de laboratorio (2 períodos de 45 min) para dar respuesta a las preguntas.
5. Primero realice el taller de forma individual y una vez desarrollado haga una reunión video llamada en equipo para que colaborativamente den respuesta en equipo las preguntas del ejercicio.
6. Se deben recibir en sección tareas de la plataforma de clases virtuales la respectiva asignación resuelta o realizar el test online del taller.
7. **PROCEDIMIENTO O ENUNCIADO DE LA EXPERIENCIA:**

**PARTE I. REVISE, SELECCIONE LA RESPUESTA CORRECTA Y EVIDENCIA SU RESPUESTA. USAR MOTOR MYSQL PARA RESOLVER LOS PROBLEMAS (20 PUNTOS)**

1.Teniendo en cuenta el significado de los siguientes códigos de error: (3 puntos)

Error: 1036 (ER\_OPEN\_AS\_READONLY). Table ‘%s’ is read only

Error: 1062 (ER\_DUP\_ENTRY). Duplicate entry ‘%s’ for key %d

**-- Paso 1**

CREATE TABLE t (s1 INT, PRIMARY KEY (s1));

**-- Paso 2**

DELIMITER $$

CREATE PROCEDURE handlerexam(IN a INT, IN b INT, IN c INT, OUT x INT)

BEGIN

DECLARE EXIT HANDLER FOR 1036 SET x = 10;

DECLARE EXIT HANDLER FOR 1062 SET x = 30;

SET x = 1;

INSERT INTO t VALUES (a);

SET x = 2;

INSERT INTO t VALUES (b);

SET x = 3;

INSERT INTO t VALUES (c);

SET x = 4;

END

$$

**¿Qué devolvería la última sentencia SELECT @x en cada caso (a y b)? Justifique su respuesta. Sin una justificación válida la respuesta será considerada incorrecta.**

-- a)

CALL handlerexam(1, 2, 3, @x);

SELECT @x;

-- b)

CALL handlerexam(1, 2, 1, @x);

SELECT @x;

2. Dado el siguiente procedimiento:

**-- Paso 1**

CREATE TABLE t (s1 INT, PRIMARY KEY (s1));

**-- Paso 2**

DELIMITER $$

CREATE PROCEDURE test(IN a INT, OUT b INT)

BEGIN

SET b = 0;

WHILE a > b DO

SET b = b + 1;

IF b != 2 THEN

INSERT INTO t VALUES (b);

END IF

END WHILE;

END;

**¿Qué valores tendría la tabla t y qué valor devuelve la sentencia SELECT value en cada caso (a y b)? Justifique la respuesta. Sin una justificación válida la respuesta será considerada incorrecta.**

-- a)

CALL test(-10, @value);

SELECT @value;

-- b)

CALL test(10, @value);

SELECT @value;

3. ¿Pueden aparecer las siguientes sentencias en el mismo bloque de código? (3 puntos)

DECLARE a INT;

DECLARE a FLOAT;

4. ¿Pueden aparecer las siguientes sentencias en el mismo bloque de código? (3 puntos)

DECLARE b VARCHAR(20);

DECLARE b HANDLER FOR SQLSTATE '02000';

5.¿Qué valor devuelve la sentencia SELECT value? (4 puntos)

* 0
* 9
* 10
* NULL
* El código entra en un bucle infinito y nunca alcanza la sentencia SELECT value

DELIMITER $$

CREATE PROCEDURE incrementor (OUT i INT)

BEGIN

REPEAT

SET i = i + 1;

UNTIL i > 9

END REPEAT;

END;

DELIMITER $$

CREATE PROCEDURE test ()

BEGIN

DECLARE value INT default 0;

CALL incrementor(value);

-- ¿Qué valor se muestra en esta sentencia?

SELECT value;

END;

DELIMITER ;

CALL test();

6.¿Qué valor devuelve la sentencia SELECT value? (4 puntos)

* 0
* 9
* 10
* NULL
* El código entra en un bucle infinito y nunca alcanza la sentencia SELECT value

DELIMITER $$

CREATE PROCEDURE incrementor (IN i INT)

BEGIN

REPEAT

SET i = i + 1;

UNTIL i > 9

END REPEAT;

END;

DELIMITER $$

CREATE PROCEDURE test ()

BEGIN

DECLARE value INT default 0;

CALL incrementor(value);

-- ¿Qué valor se muestra en esta sentencia?

SELECT value;

END;

DELIMITER ;

CALL test();

**PARTE II. CONSTRUIR EN EQUIPO DE TRABAJO - PROCEDIMIENTOS (40 PUNTOS)**

1. Escribe un procedimiento que reciba un número real de entrada y muestre un mensaje indicando si el número es positivo, negativo o cero. Diseñe el procedimiento para que tenga un parámetro de entrada, con el valor un número real, y un parámetro de salida, con una cadena de caracteres indicando si el número es positivo, negativo o cero.
2. Escribe un procedimiento que reciba un número real de entrada, que representa el valor de la nota de un alumno, y muestre un mensaje indicando qué nota ha obtenido teniendo en cuenta las siguientes condiciones:
   * [0,5) = Insuficiente
   * [5,6) = Aprobado
   * [6, 7) = Bien
   * [7, 9) = Notable
   * [9, 10] = Sobresaliente
   * En cualquier otro caso la nota no será válida.

Diseñe el procedimiento para que tenga un parámetro de entrada, con el valor de la nota en formato numérico y un parámetro de salida, con una cadena de texto indicándola nota correspondiente.

1. Escribe un procedimiento que reciba como parámetro de entrada un valor numérico que represente un día de la semana y que devuelva una cadena de caracteres con el nombre del día de la semana correspondiente. Por ejemplo, para el valor de entrada 1 debería devolver la cadena lunes.
2. Crea una base de datos llamada procedimientos que contenga una tabla llamada cuadrados. La tabla cuadrados debe tener dos columnas de tipo INT UNSIGNED, una columna llamada número y otra columna llamada cuadrado.

Una vez creada la base de datos y la tabla deberá crear un procedimiento llamado calcular\_cuadrados con las siguientes características. El procedimiento recibe un parámetro de entrada llamado tope de tipo INT UNSIGNED y calculará el valor de los cuadrados de los primeros números naturales hasta el valor introducido como parámetro. El valor del número y de sus cuadrados deberán ser almacenados en la tabla cuadrados que hemos creado previamente.

Tenga en cuenta que el procedimiento deberá eliminar el contenido actual de la tabla antes de insertar los nuevos valores de los cuadrados que va a calcular.

Utilice un bucle WHILE para resolver el procedimiento.

Utilice un bucle REPEAT para resolver el procedimiento del ejercicio anterior.

Utilice un bucle LOOP para resolver el procedimiento del ejercicio anterior.

**PARTE III. CONSTRUIR EN EQUIPO DE TRABAJO - FUNCIONES (30 PUNTOS)**

1. Escribe una función que devuelva el valor de la hipotenusa de un triángulo a partir de los valores de sus lados.
2. Escribe una función que reciba como parámetro de entrada un valor numérico que represente un día de la semana y que devuelva una cadena de caracteres con el nombre del día de la semana correspondiente. Por ejemplo, para el valor de entrada 1 debería devolver la cadena lunes.
3. Escribe una función que devuelva como salida el número de años que han transcurrido entre dos fechas que se reciben como parámetros de entrada. Por ejemplo, si pasamos como parámetros de entrada las fechas 2018-01-01 y 2008-01-01 la función tiene que devolver que han pasado 10 años.

Para realizar esta función puede hacer uso de las siguientes funciones que nos proporciona MySQL:

• DATEDIFF

• TRUNCATE

**PARTE IV CONSTRUIR EN EQUIPO DE TRABAJO - TRIGGERS O DISPARADORES (10 PUNTOS)**

Crea una base de datos llamada test que contenga una tabla llamada alumnos con las siguientes columnas.

**Tabla alumnos:**

* id (entero sin signo)
* nombre (cadena de caracteres)
* apellido1 (cadena de caracteres)
* apellido2 (cadena de caracteres)
* email (cadena de caracteres)

Escriba un procedimiento llamado crear\_email que dados los parámetros de entrada: nombre, apellido1, apellido2 y dominio, cree una dirección de email y la devuelva como salida.

**Procedimiento: crear\_email**

**Entrada:**

* nombre (cadena de caracteres)
* apellido1 (cadena de caracteres)
* apellido2 (cadena de caracteres)
* dominio (cadena de caracteres)

**Salida:**

email (cadena de caracteres)

Devuelva una dirección de correo electrónico con el siguiente formato:

* El primer carácter del parámetro nombre.
* Los tres primeros caracteres del parámetro apellido1.
* Los tres primeros caracteres del parámetro apellido2.
* El carácter @.
* El dominio pasado como parámetro.

Una vez creada la tabla escriba un **trigger** con las siguientes características:

**Trigger: trigger\_crear\_email\_before\_insert**

* Se ejecuta sobre la tabla alumnos.
* Se ejecuta antes de una operación de inserción.
* Si el nuevo valor del email que se quiere insertar es NULL, entonces se le creará automáticamente una dirección de email y se insertará en la tabla.
* Si el nuevo valor del email no es NULL se guardará en la tabla el valor del email.
* Nota: Para crear la nueva dirección de email se deberá hacer uso del procedimiento crear\_email.

**NOTA: DESARROLLE SUS RESPUESTAS EN LA SECCION G. RESULTADOS DE ESTA PLANTILLA.**

**RESULTADOS:**

**PARTE I**

**PARTE II**

1. **Dhiraj Lakhwani**

Procedimiento que reciba un número real de entrada y muestre un mensaje indicando si el número es positivo, negativo o cero.

DELIMITER //

CREATE Procedure Signo (num float)

BEGIN

DECLARE resp VARCHAR(10);

IF num= 0 THEN

SET resp = 'CERO';

ELSEIF num>0 THEN

SET resp = 'POSITIVO';

ELSE

SET resp = 'NEGATIVO';

End If;

Select Concat('El numero es ', resp) as 'Respuesta';

END; //

DELIMITER ;



1. **Dhiraj Lakhwani**  
     
   Procedimiento que reciba la nota de un alumno y devuelva el tipo de nota

DELIMITER //

CREATE Procedure Nota (calif int)

BEGIN

DECLARE resp VARCHAR(30);

IF calif< 5 THEN

SET resp = 'INSUFICIENTE';

ELSEIF calif<6 THEN

SET resp = 'APROBADO';

ELSEIF calif<7 THEN

SET resp = 'BIEN';

ELSEIF calif<9 THEN

SET resp = 'NOTABLE';

ELSEIF calif<=10 THEN

SET resp = 'SOBRESALIENTE';

ELSE

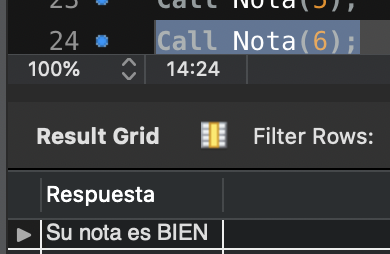
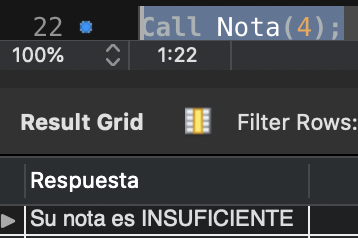
SET resp = 'LA NOTA INGRESADA NO ES VÁLIDA';

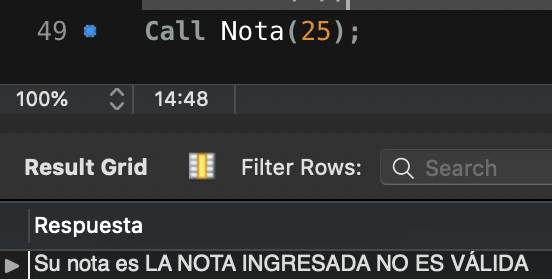
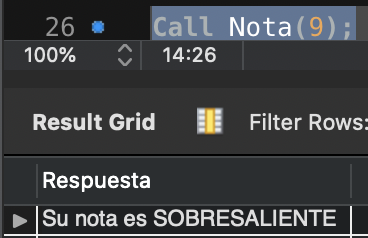
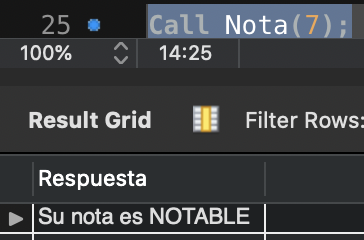
End If;

Select Concat('Su nota es ', resp) as 'Respuesta';

END; //

DELIMITER ;





1. **María José Martínez**

**Procedimiento para ingresar un numero y devolver un día de la semana**

DELIMITER //

create procedure diasem(in d int, out res varchar (50) )

BEGIN

SET res = (CASE d

when 1 then 'Lunes'

when 2 then 'Martes'

when 3 then 'Miercoles'

when 4 then 'Jueves'

when 5 then 'Viernes'

when 6 then 'Sábado'

when 7 then 'Domingo'

else ' no coincide con un dia de la semana'

END);

end

//



call diasem(2, @res);

Select @res as 'dia de la semana';

Captura de pantalla de un celular

Descripción generada automáticamente

**\*En caso de ingresar un número mayor a 7\***

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

1. **María José Martínez**

**Creación de la base de datos y la tabla**

create Database procedimientos; -- BASE DE DATOS

use procedimientos;

create table cuadrados( -- TABLA

numero int unsigned,

cuadrado int unsigned

)

**Procedimiento que calcula el valor al cuadrado de un número**

* **Utilizando While**

DELIMITER //

drop procedure if exists calcular\_cuadrados//

create procedure calcular\_cuadrados( in tope int unsigned)

begin

declare x , y int unsigned;

set x=0;

truncate cuadrados;

while x<=tope do

set y = pow(x,2);

insert into cuadrados values(x, y);

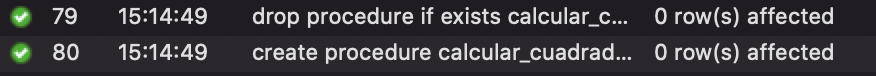
set x= x+1;

end while;

select\* from cuadrados;

end

//



call calcular\_cuadrados(5);

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente

* **Utilizando Repeat**

DELIMITER //

drop procedure if exists calcular\_cuadrados//

create procedure calcular\_cuadrados( in tope int unsigned)

begin

declare x , y int unsigned;

set x=0;

truncate cuadrados;

repeat

set y = pow(x,2);

insert into cuadrados values(x, y);

set x= x+1;

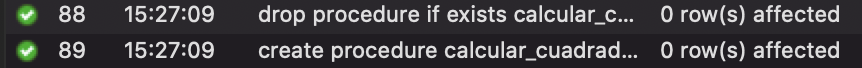
until x>tope

end repeat;

select\* from cuadrados;

end

//



call calcular\_cuadrados(5);

Imagen que contiene Tabla

Descripción generada automáticamente

* **Utilizando Loop**

DELIMITER //

drop procedure if exists calcular\_cuadrados//

create procedure calcular\_cuadrados( in tope int unsigned)

begin

declare x , y int unsigned;

set x=0;

truncate cuadrados;

empieza: LOOP

set y = pow(x,2);

insert into cuadrados values(x, y);

set x= x+1;

if x>tope then

leave empieza;

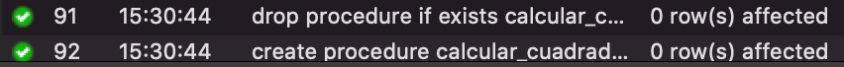
end if;

end loop;

select\* from cuadrados;

end

//



call calcular\_cuadrados(4);

Tabla

Descripción generada automáticamente

**PARTE III**

1. **Jacky He**

DELIMITER //

CREATE FUNCTION HIPO(ladoA float, ladoB float)

RETURNS FLOAT

DETERMINISTIC BEGIN

DECLARE A, B, C float;

SET A = ladoA;

SET B = ladoB;

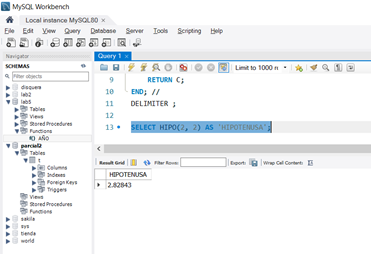
SET C = SQRT((A\*A) + (B\*B));

RETURN C;

END; //

DELIMITER ;

SELECT HIPO(2, 2) AS 'HIPOTENUSA';



1. **Jacky He**

DELIMITER //

CREATE FUNCTION DIAS(num INT)

RETURNS VARCHAR(34)

DETERMINISTIC BEGIN

DECLARE N INT;

DECLARE dia VARCHAR(34);

SET N = num;

IF N = 1 THEN

SET dia = 'LUNES';

ELSEIF N = 2 THEN

SET dia = 'MARTES';

ELSEIF N = 3 THEN

SET dia = 'MIERCOLES';

ELSEIF N = 4 THEN

SET dia = 'JUEVES';

ELSEIF N = 5 THEN

SET dia = 'VIERNES';

ELSEIF N = 6 THEN

SET dia = 'SABADO';

ELSEIF N = 7 THEN

SET dia = 'DOMINGO';

ELSE

SET dia = 'ERROR, ELIJA UN NUMERO ENTRE 1 - 7.';

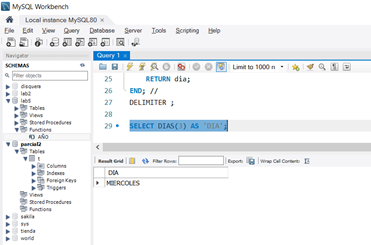
END IF;

RETURN dia;

END; //

DELIMITER ;

SELECT DIAS(3) AS 'DIA';



1. **Richie Ieong**

DELIMITER //

CREATE FUNCTION AÑO (fecha1 DATE, fecha2 DATE)

returns int deterministic

begin

declare dias int;

declare años int;

set dias= datediff(fecha1,fecha2);

set años=dias/365;

return años;

END//

DELIMITER ;

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

**PARTE IV**

**Yui Lo / Richie Ieong**

DROP DATABASE IF EXISTS test;

CREATE DATABASE test;

USE test;

CREATE TABLE alumnos (

id INT UNSIGNED AUTO\_INCREMENT PRIMARY KEY,

nombre VARCHAR(50) NOT NULL,

apellido1 VARCHAR(50) NOT NULL,

apellido2 VARCHAR(50),

email VARCHAR(50)

);

DELIMITER $$

DROP PROCEDURE IF EXISTS crear\_email$$

CREATE PROCEDURE crear\_email (

in pnombre varchar(50),

in papellido1 varchar(50),

in papellido2 varchar(50),

in pdominio varchar(20),

out pemail varchar(50))

BEGIN

set pemail= CONCAT(LEFT(pnombre, 1),LEFT(papellido1, 3),LEFT(papellido2, 3),'@',pdominio);

END$$

DELIMITER ;

DELIMITER $$

DROP TRIGGER IF EXISTS trigger\_crear\_email\_before\_insert$$

CREATE TRIGGER trigger\_crear\_email\_before\_insert

BEFORE INSERT

ON alumnos FOR EACH ROW

BEGIN

DECLARE pnombre varchar(50);

DECLARE papellido1 varchar(50);

DECLARE papellido2 varchar(50);

DECLARE pdominio varchar(20);

DECLARE pemail varchar(50);

set pnombre=NEW.nombre;

set papellido1=NEW.apellido1;

set papellido2=NEW.apellido2;

set pdominio='hotmail.com';

IF NEW.email is null THEN

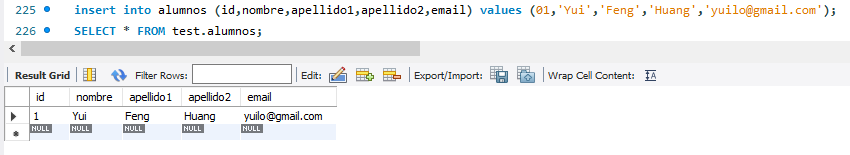
call crear\_email(pnombre,papellido1,papellido2,pdominio,pemail);

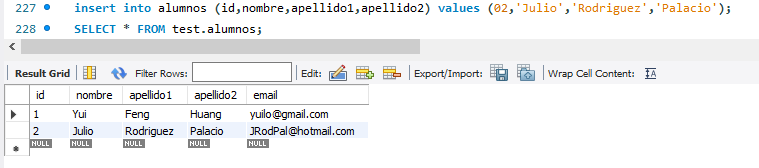
set NEW.email = pemail;

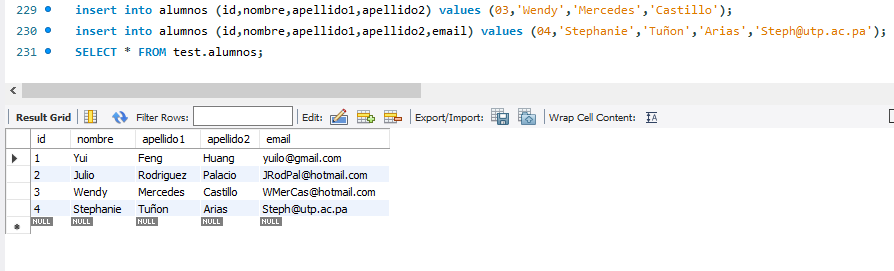
END IF;

END $$

DELIMITER ;







**CONSIDERACIONES FINALES – CONCLUSIONES:**

* Opinión del estudiante(s) sobre el logro del objetivo y el desarrollo de la Asignación.
* Opinión del estudiante(s) sobre el valor de esta herramienta para apoyar los procesos de aprendizaje.

Esta actividad sirvió para poner a prueba lo aprendido en la clase y saber manejar los distintos tipos de estructuras que se presentan en un gestor de Base de Datos. Además de que se fomentó el trabajo en equipo para poder apoyarnos en las partes que se nos dificultaba un poco.

**BIBLIOGRAFIA:**

**RÚBRICAS:**

|  |  |
| --- | --- |
| ***Detalle - Evaluar*** | ***Valor de la respuesta*** |
| ***Sección G – (100 PUNTOS)*** | ***100 pts*** |