**Icono

Descripción generada automáticamente**

APUNTES TEMA-VI

Bases de Datos (BBDD)

22-4-2021

OLIVER LINARES COLINAS

Contenido

[PROCEDIMIENTOS 2](#_Toc69978758)

[CREACIÓN 2](#_Toc69978759)

[BORRAR 2](#_Toc69978760)

[EJEMPLO 2](#_Toc69978761)

[EJEMPLO DE ENTRADA-SALIDA 3](#_Toc69978762)

[FUNCIONES 4](#_Toc69978763)

[CREACIÓN 4](#_Toc69978764)

[BORRAR 4](#_Toc69978765)

[VISUALIZACIÓN 4](#_Toc69978766)

[SENTENCIAS 5](#_Toc69978767)

[VARIABLES 5](#_Toc69978768)

[Locales 5](#_Toc69978769)

[Globales 5](#_Toc69978770)

[SELECT … INTO 6](#_Toc69978771)

[IF 7](#_Toc69978772)

[CASE 7](#_Toc69978773)

[BUCLES 8](#_Toc69978774)

[WHILE 8](#_Toc69978775)

[REPEAT 8](#_Toc69978776)

# PROCEDIMIENTOS

Son módulos que reciben unas entradas y tienen un procesamiento, almacenaje y salida. Existen funciones predefinidas del sistema, como LOCATE, ABS, etc…

La diferencia principal entre los procedimientos y las funciones es que los procedimientos pueden tener varias salidas, mientras que las funciones solo pueden devolver un valor.

## CREACIÓN

CREATE PROCEDURE *NOMBRE* ([*PARAMETROS*])

BEGIN

[CUERPO]

END;

Los parámetros pueden ser de entrada, salida o ambas (IN | OUT | INOUT)

A la hora de crear uno de estos módulos tenemos que cambiar el delimitador temporalmente para evitar que al poner un “;” se ejecute solo hasta esa sentencia. Esto se hace con DELIMITER.

DELIMITER $

CREATE PROCEDURE PROSUMA(IN N1 INT, IN N2 INT, OUT SUMA INT)

    BEGIN

        SET SUMA = N1 + N2;     -- TODAS LAS SENTENCIAS EMPIEZAN CON SET

    END$

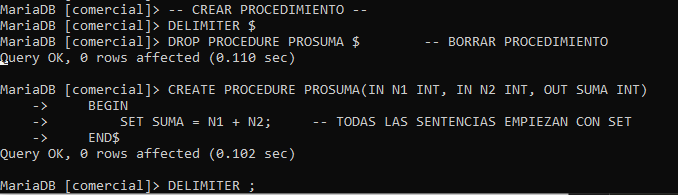
DELIMITER ;

## BORRAR

DROP PROCEDURE PROSUMA;

## EJEMPLO

DROP PROCEDURE PROSUMA $        -- BORRAR PROCEDIMIENTO

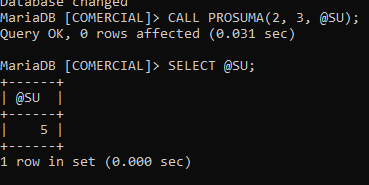


Para probar el funcionamiento del procedimiento se usa CALL *NOMBRE\_PROCEDIMIENTO(VALORES\_ENTRADA)*.

A la hora de llamar al módulo, tenemos que indicar a las variables de entrada unos valores; si son de salida, una variable (@s); si son de entrada-salida, usamos SET *VARIABLE* = *VALOR*;

CALL PROSUMA(2, 3, @SU);

SELECT @SU;



## EJEMPLO DE ENTRADA-SALIDA

-- SUMAR 3 NÚMEROS, PERO QUE UNO DE ELLOS NOS SIRVA COMO SALIDA

DELIMITER $

DROP PROCEDURE PROSUMA $

CREATE PROCEDURE PROSUMA(IN N1 INT, IN N2 INT, INOUT SUMA INT)

    BEGIN

        SET SUMA = N1 + N2 + SUMA;

    END$

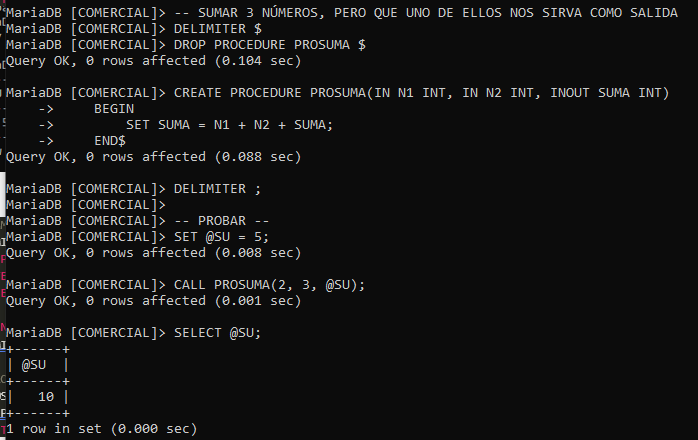
DELIMITER ;

-- PROBAR --

SET @SU = 5;

CALL PROSUMA(2, 3, @SU);

SELECT @SU;

****

# FUNCIONES

## CREACIÓN

CREATE FUNCTION *NOMBRE*(*VALORES­\_ENTRADA*) RETURNS *TIPO*

BEGIN

[CUERPO]

END;

Solo se le pasan parámetros de entrada ya que la propia función será la que devuelva el valor, indicando previamente de qué tipo va a ser dicho valor.

DELIMITER $

DROP FUNCTION FUNSUMA $        -- BORRAR PROCEDIMIENTO

CREATE FUNCTION FUNSUMA(N1 INT, N2 INT) RETURNS INT

    BEGIN

        RETURN N1 + N2;     -- TODAS LAS SENTENCIAS EMPIEZAN CON SET

    END$

DELIMITER ;

## BORRAR

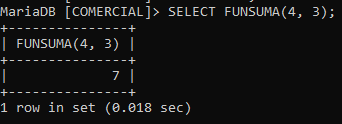
Similar al procedimiento:

DROP FUNCTION FUNSUMA $

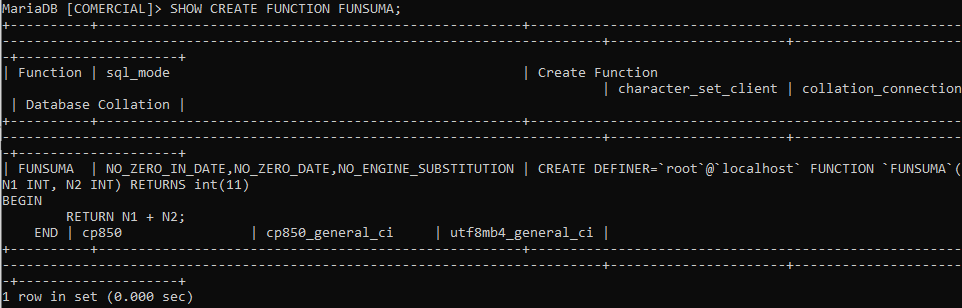
## VISUALIZACIÓN

Para visualizar el resultado de la función, realizamos una consulta:

SELECT FUNSUMA(4, 3);



Se puede visualizar el código de una función/procedimiento con el comando SHOW CREATE FUNCTION/PROCEDURE *NOMBRE*;



# SENTENCIAS

Ya hemos visto BEGIN … END y CALL y para qué sirven.

## VARIABLES

### Locales

Todas las declaraciones tienen que ir al principio de la función y seguidas una de otras. Además, se pueden inicializar las variables en su asignación con DEFAULT *VALOR*.

DELIMITER $

DROP FUNCTION FUNSUMA $

CREATE FUNCTION FUNSUMA(N1 INT, N2 INT) RETURNS INT

    BEGIN

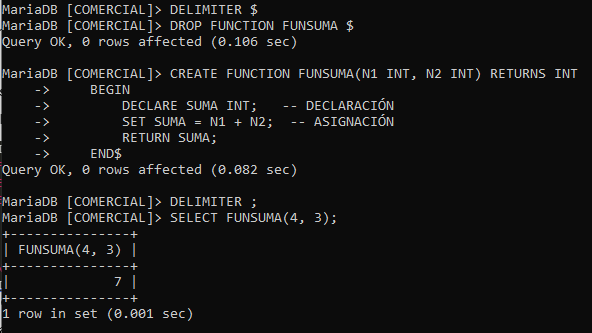
DECLARE SUMA INT DEFAULT 0;   -- DECLARACIÓN

        SET SUMA = N1 + N2;  -- ASIGNACIÓN

        RETURN SUMA;

    END$

DELIMITER ;



### Globales

Aquella que es conocida que se puede usar tanto dentro de un procedimiento como fuera.

DELIMITER $

DROP PROCEDURE PROSUMA $

CREATE PROCEDURE PROSUMA(IN N1 INT, IN N2 INT)

    BEGIN

        SET @SUMA = N1 + N2;

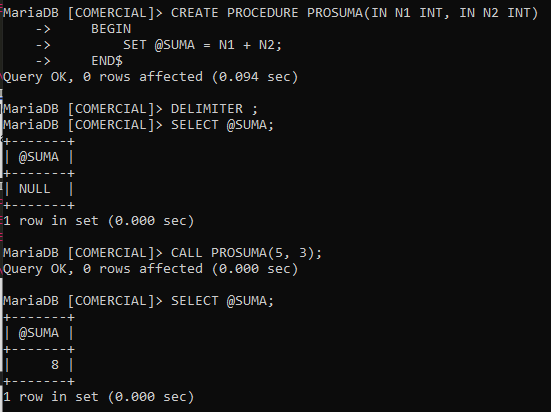
    END$

DELIMITER ;

SELECT @SUMA;

CALL PROSUMA(5, 3);

SELECT @SUMA;



## SELECT … INTO

Nos permite guardar el valor devuelto de una sentencia en una variable, siempre y cuando solo se devuelva un valor.

DELIMITER $

DROP FUNCTION FUNTOTAR $

CREATE FUNCTION FUNTOTAR() RETURNS INT

    BEGIN

        DECLARE TNA INT;

        SELECT COUNT(\*) INTO TNA

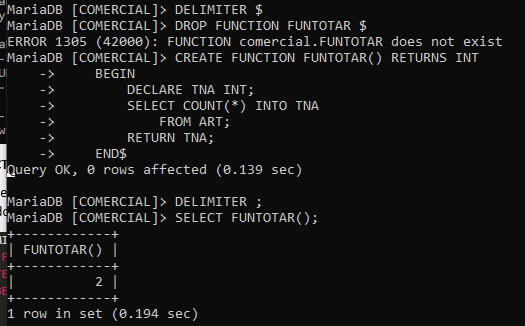
            FROM ART;

        RETURN TNA;

    END$

DELIMITER ;

SELECT FUNTOTAR();



## IF

IF [CONDICION] THEN [ACCIONES]

ELSEIF [CONDICION] THEN [ACCIONES]

ELSE [ACCIONES]

END IF;

Ej.: Comprobar el mayor número de entre 3.

DROP FUNCTION MAYOR2;

DELIMITER $

CREATE FUNCTION MAYOR2(N1 INT, N2 INT, N3 INT) RETURNS INT

    BEGIN

        DECLARE AUX INT DEFAULT 0;

        IF(N1 > N2 AND N1 > N3) THEN

            SET AUX = N1;

        ELSEIF(N2 > N1 AND N2 > N3) THEN

            SET AUX = N2;

        ELSEIF(N3 > N1 AND N3 > N2) THEN

            SET AUX = N3;

        END IF;

        RETURN AUX;

    END$

DELIMITER ;

SELECT MAYOR2(2, 1, 3);

## CASE

CASE *NUM*

WHEN VALOR THEN

SENTENCIAS

WHEN VALOR THEN

SENTENCIAS

ELSE

SENTENCIAS

END CASE;

CASE *NUM*

WHEN *CONDICION* THEN

SENTENCIAS;

WHEN *CONDICION* THEN

SENTENCIAS;

ELSE

SENTENCIAS;

END CASE;

# BUCLES

## WHILE

WHILE *CONDICIÓN* DO

*SENTENCIAS*;

END WHILE;

DROP FUNCTION WHILEFUNC;

DELIMITER $

CREATE FUNCTION WHILEFUNC(P1 INT) RETURNS INT

    BEGIN

        DECLARE V\_LOCAL INT DEFAULT 0;

        WHILE V\_LOCAL <= P1 DO

            SET V\_LOCAL = V\_LOCAL + 1;

        END WHILE;

        RETURN V\_LOCAL;

    END$

DELIMITER ;

SELECT WHILEFUNC(100);

## REPEAT

REPEAT

*ACCION;*

UNTIL *CONDICION:*

END REPEAT;

DROP PROCEDURE DOREPEAT;

DELIMITER $

CREATE PROCEDURE DOREPEAT(IN P1 INT, OUT SALIDA INT)

    BEGIN

        DECLARE V\_LOCAL INT DEFAULT 0;

        REPEAT

            SET V\_LOCAL = V\_LOCAL + 1;

        UNTIL V\_LOCAL > P1 END REPEAT;

        SET SALIDA = V\_LOCAL;

    END$

DELIMITER ;

CALL DOREPEAT(100, @X);

SELECT @X;