# CICLO DE GRADO SUPERIOR: ABD - 2° curso

Unidad 06 - Procedimientos y funciones - Ejemplos (abdt6 ejemplos)

## **FUNCIONES**

Las **funciones** devuelven siempre 1 valor. Ese valor se devuelve con RETURN.

Se ejecutan con SELECT.

```
-- Llamada a la función
SELECT ejlf_saludo();
```

```
SELECT ej2f_saludoPersonal('Pepe');
```

```
Ejemplo 3

CREATE FUNCTION ej3f_unomas(numero INT) RETURNS INT
BEGIN

DECLARE resultado INT;

SET resultado = numero + 1;

RETURN resultado;
END;
```

```
SELECT ej3f_unomas('5')
```

```
Ejemplo 4

CREATE FUNCTION ej4f_doble(numero INT) RETURNS INT
BEGIN

DECLARE resultado INT;

SET resultado = numero * 2;

RETURN resultado;
END;
```

```
SELECT ej4f_doble('5')
```

Después de haber desarrollado la función anterior, podemos usarla de la siguiente manera:

```
SELECT ej5f_calcularEdad('1985-12-25', CURRENT_DATE()) AS Edad;
```

Para el ejemplo anterior, se quiere que se actualice la tabla **usuarios** con las edades correspondientes utilizando la función creada anteriormente:

codigo	nombre	fnac	edad
1	Pepe	1985-05-04	(NULL)
2	Juan	1946-11-02	(NULL)
3	Amparo	1981-12-04	(NULL)
4	Rosana	1995-08-03	(NULL)
5	Maite	1960-01-03	(NULL)

```
UPDATE usuarios
SET edad = ej5f_calcularEdad(fnac, CURRENT_DATE());
```

## Ejemplo 6

Para finalizar, algo un poco más complejo, vamos a crear una función que acepte un dividendo y un divisor y haga una división sin usar el operador división:

```
CREATE FUNCTION ej6f_divide(dividendo INT, divisor INT) RETURNS INT

DECLARE aux INT;
DECLARE contador INT;
DECLARE resto INT;

SET contador = 1;
SET aux = 0;

WHILE (aux + divisor) < dividendo DO

SET aux = aux + divisor;
SET contador = contador + 1;
END WHILE;

SET resto = dividendo - aux;

RETURN contador;

END;
```

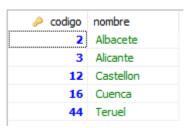
```
SELECT ej6f_divide(20,2);
```

## **PROCEDIMIENTOS**

Los **procedimientos** pueden devolver ninguno o muchos valores.

Se ejecutan con CALL si solo tiene parámetros de entrada y con CALL y SELECT si tiene parámetros de salida.

Para los siguientes ejemplos vamos a utilizar la tabla provincia:



## **Ejemplo 7**

Mostrar información calculada de una tabla:

```
CREATE PROCEDURE ej7p_cuentaProvincias(OUT cuenta INT)
BEGIN
SELECT COUNT(*) INTO cuenta FROM provincia;
END
```

```
CALL ej7p_cuentaProvincias(@numero);
SELECT @numero;
```

#### **Ejemplo 8**

Insertar en una tabla:

```
CREATE PROCEDURE ej8p_insertaProvincia(IN codpro INT, IN nompro VARCHAR(50))

BEGIN

INSERT INTO provincia VALUES(codpro, nompro);

SELECT concat("Provincia introducida: ", nompro);

END
```

```
CALL ej8p_insertaProvincia(36, 'Pontevedra')
```

## **Ejemplo 9**

#### IF...THEN...ELSE

Comprobar que el número de provincial es 46 para poder insertar una provincia nueva.

```
CALL ej9p_insertaValencia(46, 'Valencia');
```

#### **Ejemplo 10**

Comparación de cadenas.

```
CALL ej10p_compara('Pepe', 'Juan');
```

## **Ejemplo 11**

Supongamos que tenemos la siguiente tabla libro:



```
CREATE TABLE libro(
    codigo INT PRIMARY KEY,
    titulo VARCHAR(50),
    precio INT,
    editorial ENUM('Anaya', 'Santillana')
);

INSERT INTO libro VALUES (1, 'Los ojos amarillos', 51, 'Santillana');
```

Crear un procedimiento que inserte libros automáticamente:

```
CREATE PROCEDURE ejllp_insertaLibros()

BEGIN

DECLARE cont int;

SET cont = 5;

WHILE cont < 20 DO

IF ((cont mod 2) = 0) THEN

INSERT INTO libro VALUES (cont, null, null, 'Anaya');

ELSE

INSERT INTO libro VALUES (cont, null, null, 'Santillana');

END IF;

SET cont = cont + 1;

END;

END;
```

```
CALL ejllp_insertarLibros();
```

### **Ejemplo 12**

Modificar el ejemplo anterior para que introduzca también el título del libro con el siguiente formato *Libro5, Libro7...* y el precio que será siempre el precio que introducimos por parámetro de entrada.

#### **Ejemplo 13**

Modificar el ejemplo anterior para que introduzca el código del primer libro y el último que queremos introducir.