# PRACTICA 17

1. Crea la base de datos **P17** y en ella las tablas;

```
CREATE TABLE ninos(edad int, nombre varchar(50));
CREATE TABLE adultos(edad int, nombre varchar(50));
```

```
MariaDB [(none)]> CREATE DATABASE P17;
Query OK, 1 row affected (0.00 sec)

MariaDB [(none)]> use P17;
Database changed
MariaDB [P17]> CREATE TABLE ninos(edad int, nombre varchar(50));
Query OK, 0 rows affected (0.03 sec)

MariaDB [P17]> CREATE TABLE adultos(edad int, nombre varchar(50));
Query OK, 0 rows affected (0.03 sec)
```

2. Creamos el procedimiento almacenado corrigiendo errores si existen:

```
CREATE procedure introPersona(IN ed int,IN nom varchar(50))
begin
IF ed < 18 then
INSERT INTO ninos VALUES(ed,nom);
else
INSERT INTO adultos VALUES(ed,nom);
end IF;
end;
```

```
MariaDB [p17]> DELIMITER //
MariaDB [p17]> CREATE procedure introPersona(ed int, nom varchar(50))
    -> begin
    ->     IF ed < 18 then
    ->         INSERT INTO ninos VALUES(ed,nom);
    ->     else
    ->         INSERT INTO adultos VALUES(ed,nom);
    ->     end IF;
    -> end//
Query OK, 0 rows affected (0.00 sec)

MariaDB [p17]> DELIMITER ;
```

# PRACTICA 17

3. ¿Qué sucede al ejecutar las siguientes instrucciones?

```
CALL introPersona(25,”Pedro Manuel”);
CALL introPersona(45,”Carlota”);
CALL introPersona(17,”Susana”);
CALL introPersona(25,”Antonio”);
CALL introPersona(15,”Vanesa Francisca”);
```

```
MariaDB [p17]> CALL introPersona(25,"Pedro Manuel");
Query OK, 1 row affected (0.00 sec)

MariaDB [p17]> CALL introPersona(45,"Carlota");
Query OK, 1 row affected (0.00 sec)

MariaDB [p17]> CALL introPersona(17,”Susana”);
Query OK, 1 row affected (0.01 sec)

MariaDB [p17]>
MariaDB [p17]>
MariaDB [p17]> CALL introPersona(25,”Antonio”);
Query OK, 1 row affected (0.00 sec)

MariaDB [p17]>
MariaDB [p17]>
MariaDB [p17]> CALL introPersona(15,”Vanesa Francisca”);
Query OK, 1 row affected (0.00 sec)
```

Introduce en las diferentes tablas los valores introducidos.

4. Haz un procedimiento llamado **INTROADULTO** con dos parámetros de entrada que permita introducir registros en la tabla adultos y muestre un mensaje de error si la edad es menor de 18 años y no introduzca dicho registro en ninguna tabla.

```
MariaDB [p17]> DELIMITER //
MariaDB [p17]> CREATE PROCEDURE INTROADULTO(edad INT, nombre VARCHAR(50))
    -> BEGIN
    ->     IF edad<18 THEN
    ->         SELECT "ERROR - Edad inferior a 18 años";
    ->     ELSE
    ->         INSERT INTO adultos VALUES(edad,nombre);
    ->     END IF;
    -> END//
Query OK, 0 rows affected (0.01 sec)

MariaDB [p17]> DELIMITER ;
```

# PRACTICA 17

5. ¿Qué sucede al ejecutar las siguientes instrucciones?

```
CALL introAdulto(25,”Pedro Manuel”);
CALL introAdulto(45,”Carlota”);
CALL introAdulto(17,”Susana”);
CALL introAdulto(25,”Antonio”);
CALL introAdulto(15,”Vanesa Francisca”);
```

```
MariaDB [p17]> CALL introAdulto(25,"Pedro Manuel");
Query OK, 1 row affected (0.01 sec)

MariaDB [p17]> CALL introAdulto(45,"Carlota");
Query OK, 1 row affected (0.00 sec)

MariaDB [p17]> CALL introAdulto(17,"Susana");
+---------------------------------+
| ERROR - Edad inferior a 18 años |
+---------------------------------+
| ERROR - Edad inferior a 18 años |
+---------------------------------+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

MariaDB [p17]> CALL introAdulto(25,"Antonio");
Query OK, 1 row affected (0.00 sec)

MariaDB [p17]> CALL introAdulto(15,"Vanesa Francisca");
+---------------------------------+
| ERROR - Edad inferior a 18 años |
+---------------------------------+
| ERROR - Edad inferior a 18 años |
+---------------------------------+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
```

Las instrucciones que tengan un valor menor de 18 no se introducirán en la tabla y mostrara el mensaje de error.

# PRACTICA 17

6. Crea la función:

```
CREATE FUNCTION divide(dividendo int,divisor int) returns int
begin
   declare aux int;
   declare contador int;
   declare resto int;
   SET contador = 0;
   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;
```

```
MariaDB [p17]> DELIMITER //
MariaDB [p17]> CREATE FUNCTION divide(dividendo int,divisor int) returns int
    -> begin
    ->     declare aux int;
    ->     declare contador int;
    ->     declare resto int;
    ->     SET contador = 0;
    ->     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//
Query OK, 0 rows affected (0.00 sec)

MariaDB [p17]> DELIMITER ;
```

# PRACTICA 17

7. Contesta a las preguntas siguientes:

✓ Cuál es el nombre de la función.
El nombre de la función es divide

✓ Que parámetros usa.
Usa dos parámetros, dividendo y divisor

✓ Qué valor devuelve y de qué tipo es.
Devuelve el valor de contador, que es de tipo int

✓ Que variables locales usa.
Usa las variables aux, contador y resto

✓ Di como llamas a la función con los valores 20 y 3 y di que escribe y porqué.
Se llama poniendo SELECT divide(20,3);

```
MariaDB [p17]> SELECT divide(20,3);
+--------------+
| divide(20,3) |
+--------------+
|            6 |
+--------------+
1 row in set (0.01 sec)
```

✓ Y si cambiamos la instrucción "**RETURN** contador;" por "**RETURN** resto;" ¿Que sucede?
Que la función devuelve el resto de la división
En este caso 2

✓ Llama a la función con los valores 4 y 18 ¿qué sucede?

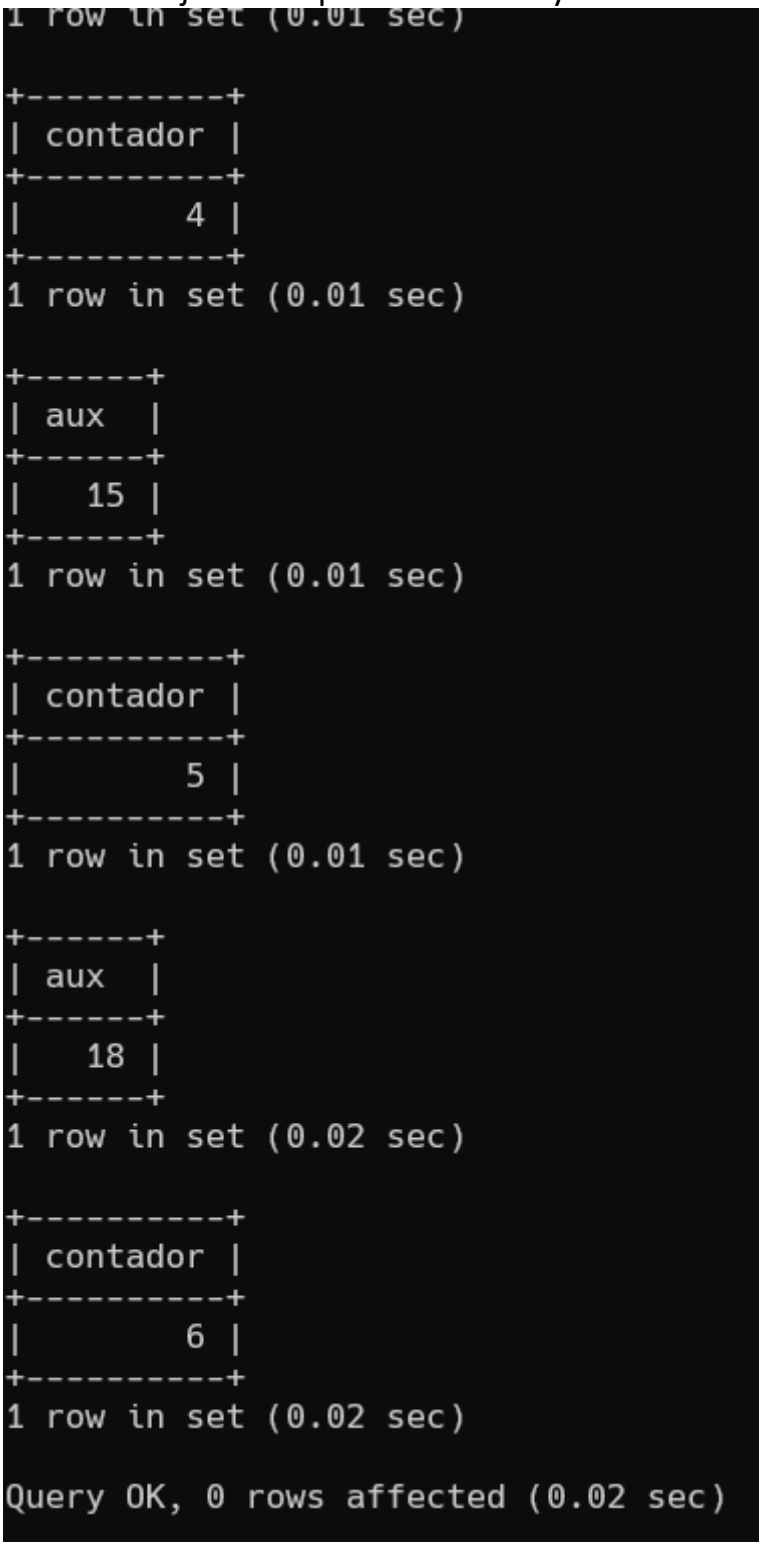
```
MariaDB [p17]> SELECT divide(4,18);
+--------------+
| divide(4,18) |
+--------------+
|            0 |
+--------------+
1 row in set (0.00 sec)
```

El resultado es 0 porque es mayor el divisor que el dividendo

✓ Y si cambiamos la instrucción "**RETURN** contador;" por "**RETURN** resto;"
El resultado es 4 porque no se puede efectuar la división por la misma razón que antes

# PRACTICA 17

8. Modifica la función anterior añadiendo una instrucción antes de END WHILE que muestre el contenido de las variables **aux** y **contador**. Ejecuta el procedimiento y muestra el resultado.



```
1 row in set (0.01 sec)

+----------+
| contador |
+----------+
|        4 |
+----------+
1 row in set (0.01 sec)

+------+
| aux  |
+------+
|   15 |
+------+
1 row in set (0.01 sec)

+----------+
| contador |
+----------+
|        5 |
+----------+
1 row in set (0.01 sec)

+------+
| aux  |
+------+
|   18 |
+------+
1 row in set (0.02 sec)

+----------+
| contador |
+----------+
|        6 |
+----------+
1 row in set (0.02 sec)

Query OK, 0 rows affected (0.02 sec)
```