

TEMA 6

CONSTRUCCIÓN DE SCRIPTS

BASE DE DATOS



Daniel Rodríguez Fernández

Contenidos del Tema

1. Lenguajes de Programación
2. Procedimientos y Funciones
3. Parámetros y Variables
4. Instrucciones Condicionales
5. Instrucciones Repetitivas

1. Lenguajes de Programación

TEMA 6: CONSTRUCCIÓN DE SCRIPTS



Daniel Rodríguez Fernández

1. Lenguajes de Programación

Se puede programar una BBDD en multitud de lenguajes: C, C++, Java, etc., pero todos ellos terminan recurriendo a sentencias SQL para insertar o extraer datos de la BBDD.

Para incrementar la potencia de ejecución de las sentencias SQL, Oracle desarrolló el lenguaje PL/SQL, que permite definir variables, estructuras de control, procedimientos, funciones, ...

Los programas PL/SQL pueden ser de dos tipos:

- Procedimientos.
- Funciones.

2. Procedimientos y Funciones

TEMA 6: CONSTRUCCIÓN DE SCRIPTS



Daniel Rodríguez Fernández

2. Procedimientos y Funciones

Son un conjunto de comandos SQL que pueden almacenarse en el servidor.

Una vez almacenado, los clientes no necesitan lanzar cada comando individual sino que pueden en su lugar llamar al procedimiento almacenado como un único comando. El intercambio hace que aumente la carga del servidor.

Diferencias entre procedimientos y funciones:

PROPIEDADES	FUNCIONES	PROCEDIMIENTOS
Valores retornados	CON RETURN	A través de parámetros (OUT o INOUT)
¿Se pueden utilizar en expresiones?	SI	NO

Esquema general de un procedimiento:

Nombre rutina + parámetros (entrada, salida o entrada/salida)

Declaración e inicialización de variables

Procesamiento de datos
Bloques BEGIN/END con instrucciones de control
(condicionales y repetitivas)

Fin
Con la instrucción RETURN para devolver un valor en el caso de funciones almacenadas

Sintaxis y Ejemplos de Rutinas Almacenadas:

Para construir nuestros propios procedimientos necesitaremos :

- Un editor Notepad ++ (o bien directamente desde la consola)
- Programas servidor y cliente de MySQL.

La **sintaxis** resumida para la creación de un procedimiento o función es:

Nombre (parametros) + modificadores

BEGIN

Declaracion (DECLARE)

Establecimiento de variables (SET)

Proceso de datos (instrucciones sql /instrucciones de control)

END

Ejemplo 1:

```
DELIMITER $$  
DROP PROCEDURE IF EXISTS hola_mundo$$  
CREATE PROCEDURE hola_mundo()  
BEGIN  
SELECT 'hola mundo';  
END;$$
```



```
C:\Mysql\bin\mysql.exe  
mysql> source C:\Users\M Carmen\Desktop\hola.sql  
Query OK, 0 rows affected (0.44 sec)  
  
Query OK, 0 rows affected (0.00 sec)  
  
mysql> call hola_mundo();  
-> $$  
+-----+  
| hola mundo |  
+-----+  
| hola mundo |  
+-----+  
1 row in set (0.02 sec)  
  
Query OK, 0 rows affected (0.08 sec)
```

Ejemplo 2:

```
DELIMITER $$  
DROP PROCEDURE IF EXISTS fecha$$  
CREATE PROCEDURE fecha()  
SELECT now();  
$$
```



```
C:\Program Files\MySQL\MySQL Server 5.1\bin\mysql.exe  
mysql> call fecha()$$  
+-----+  
| now() |  
+-----+  
| 2013-05-08 10:36:09 |  
+-----+  
1 row in set (0.16 sec)  
  
Query OK, 0 rows affected (0.21 sec)  
  
mysql>
```

3. Parámetros y Variables

TEMA 6: CONSTRUCCIÓN DE SCRIPTS



Daniel Rodríguez Fernández

3. Parámetros y Variables

Para el manejo de variables utilizamos las siguientes cláusulas:

- **DECLARE:** crea una nueva variable con su nombre y tipo. Tipos: char, varchar, int, float ...
- **DEFAULT:** Establece valores por defecto a las variables, si no se indica, dichos valores serán NULL.

Ejemplo: DECLARE a, b INT DEFAULT 5;

- **SET:** permite asignar valores a las variables usando el operador de igualdad.

Ejemplo: SET variable=30;

Tipos de parámetros:

IN: Parámetros de entrada que se usaran en el procedimiento.

```
DELIMITER $$  
CREATE PROCEDURE proc2(IN p int)  
    SET @x=p  
$$
```

```
mysql> call proc2(12345)$$  
Query OK, 0 rows affected (0.10 sec)  
  
mysql> select @x$$  
+-----+  
| @x    |  
+-----+  
| 12345 |  
+-----+  
1 row in set (0.07 sec)
```

OUT: Parámetros de salida. Son devueltos en la llamada.

```
DELIMITER $$  
CREATE PROCEDURE proc3(OUT p int)  
    SET p=-5  
$$
```

```
mysql> call proc3(@y)$$  
Query OK, 0 rows affected (0.01 sec)  
  
mysql> select @y$$  
+-----+  
| @y    |  
+-----+  
| -5    |  
+-----+  
1 row in set (0.00 sec)
```

INOUT: Permite pasar valores al procedimientos que serán modificados y devueltos en la llamada.

```
DELIMITER $$  
CREATE PROCEDURE  
proc4(INOUT p int)  
    SET p=p-5  
$$
```

```
mysql> SET @y=0$$  
Query OK, 0 rows affected (0.00 sec)  
  
mysql> CALL proc4(@y)$$  
Query OK, 0 rows affected (0.00 sec)  
  
mysql> SELECT @y$$  
+-----+  
| @y    |  
+-----+  
| -5    |  
+-----+  
1 row in set (0.00 sec)
```

Ejercicio:

Realiza la llamada para los siguientes procedimientos:

```
DELIMITER $$  
CREATE PROCEDURE ejercicioA(IN num int, OUT cuadrado int )  
    SET cuadrado=num*num;  
$$
```

```
DELIMITER $$  
CREATE PROCEDURE ejercicioB(IN base int, IN exp INT,  
    OUT potencia int )  
    select pow(base,exp) INTO potencia;  
$$
```

```
DELIMITER $$  
CREATE PROCEDURE ejercicioC (INOUT cad VARCHAR(10), OUT tam int )  
BEGIN  
    select UPPER (cad) INTO cad;  
    select LENGTH (cad) INTO tam;  
END; $$
```

4. Instrucciones Condicionales

TEMA 6: CONSTRUCCIÓN DE SCRIPTS



Daniel Rodríguez Fernández

4. Instrucciones Condicionales

IF – THEN - ELSE

```
IF expr1 THEN
    ...
ELSEIF expr2 THEN
    ...
ELSE
    ...
END IF;
```

CASE

```
CASE expression
    WHEN value THEN statements
    [WHEN value THEN
statements]
    .....
    [ELSE statements]
END CASE;
```

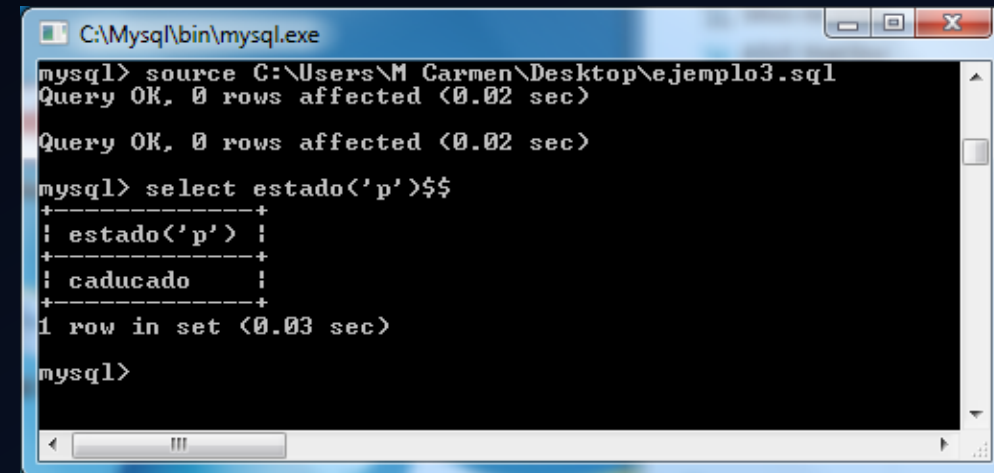
Ejemplo 1:

```
DELIMITER $$
DROP FUNCTION IF EXISTS estado$$
CREATE FUNCTION estado(estado CHAR(1))
    RETURNS VARCHAR(20)
BEGIN
    DECLARE rtdo VARCHAR(20);

    IF estado='P' THEN SET rtdo='caducado';
    ELSEIF estado='O' THEN SET rtdo='activo';
    ELSEIF estado='N' THEN SET rtdo='nuevo';
    END IF;

    RETURN (rtdo);

END;$$
```



```
C:\Mysql\bin\mysql.exe
mysql> source C:\Users\M Carmen\Desktop\ejemplo3.sql
Query OK, 0 rows affected (0.02 sec)

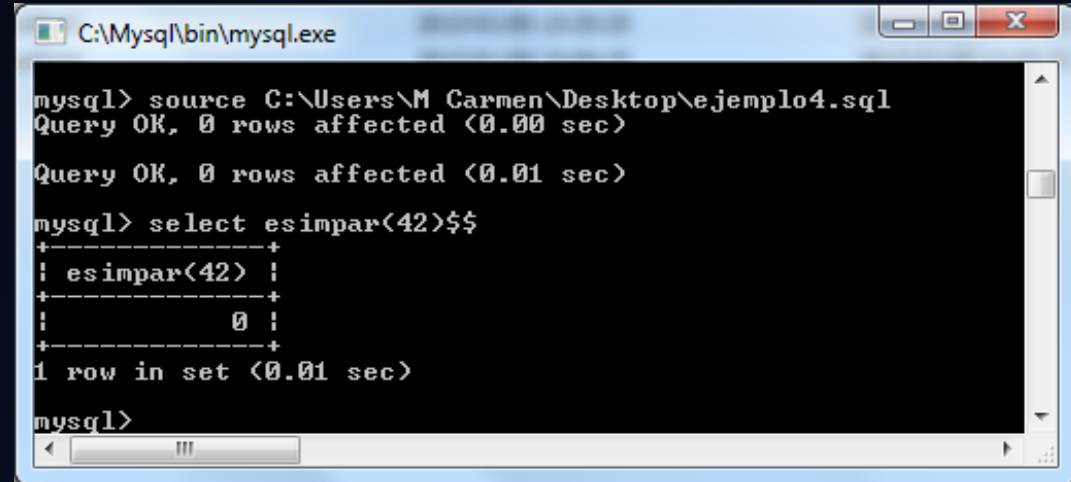
Query OK, 0 rows affected (0.02 sec)

mysql> select estado('p')$$
+-----+
| estado('p') |
+-----+
| caducado    |
+-----+
1 row in set (0.03 sec)

mysql>
```


Ejemplo 2:

```
DELIMITER $$
DROP FUNCTION IF EXISTS esimpar$$
CREATE FUNCTION esimpar(numero int)
    RETURNS int
BEGIN
    DECLARE impar INT;
    IF MOD(numero,2)=0 THEN
        SET impar=FALSE;
    ELSE SET impar=TRUE;
    END IF;
    RETURN(impar);
END; $$
```



A screenshot of a MySQL command prompt window titled "C:\Mysql\bin\mysql.exe". The window shows the following commands and their outputs:

```
mysql> source C:\Users\M Carmen\Desktop\ejemplo4.sql
Query OK, 0 rows affected (0.00 sec)

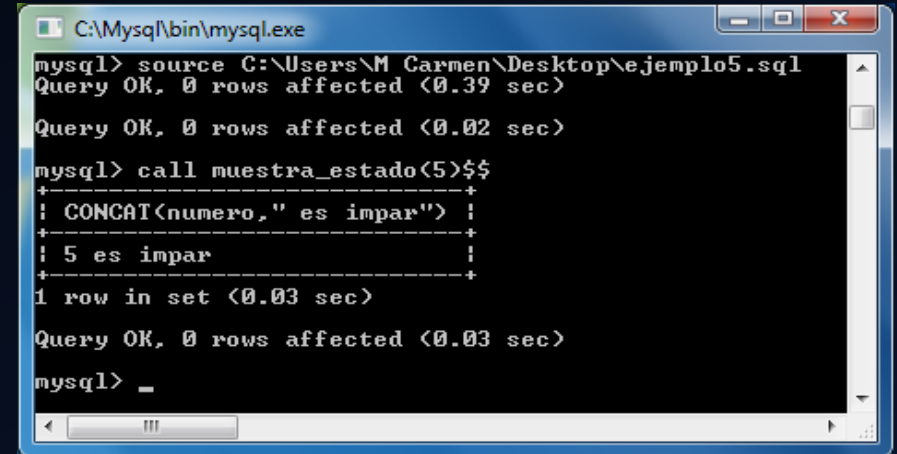
Query OK, 0 rows affected (0.01 sec)

mysql> select esimpar(42)$$
+-----+
| esimpar(42) |
+-----+
|           0 |
+-----+
1 row in set (0.01 sec)

mysql>
```

Ejemplo 3:

```
DELIMITER $$
DROP PROCEDURE IF EXISTS muestra_estado$$
CREATE PROCEDURE muestra_estado(in numero int)
BEGIN
    IF (esimpar(numero))THEN
        SELECT CONCAT(numero," es impar");
    ELSE
        SELECT CONCAT(numero," es par");
    END IF;
END; $$
```



The screenshot shows a MySQL command prompt window titled "C:\Mysql\bin\mysql.exe". The user has executed the following commands:

```
mysql> source C:\Users\M Carmen\Desktop\ejemplo5.sql
Query OK, 0 rows affected (0.39 sec)

Query OK, 0 rows affected (0.02 sec)

mysql> call muestra_estado(5)$$
+-----+
| CONCAT(numero," es impar") |
+-----+
| 5 es impar                  |
+-----+
1 row in set (0.03 sec)

Query OK, 0 rows affected (0.03 sec)

mysql> _
```

Ejercicios:

1. Crea una función que devuelva 1 - 0 si un numero es o no divisible por otro
2. Usa las estructuras condicionales para mostrar el día de la semana según un valor de entrada numérico, 1 para domingos, 2 para lunes, etc...
3. Crea una función que devuelva el mayor de 3 números pasados como parámetros
4. Crea un procedimiento que diga si una palabra, pasada como parámetro, es palíndroma.

Solución Ejercicio 1:

```
mysql> CREATE FUNCTION divisible(num int, den int) RETURNS bool
-> BEGIN
->   DECLARE esdiv bool;
->   DECLARE modulo int;
->
->   SET modulo=num%den;
->
->   IF (modulo=0) THEN SET esdiv=TRUE;
->   ELSE SET esdiv=FALSE;
->   END IF;
->
->   RETURN esdiv;
->
-> END; //
```

Query OK, 0 rows affected (0.11 sec)

```
mysql> DELIMITER ;
mysql> select divisible(6,5);
```

divisible(6,5)
0

1 row in set (0.06 sec)

Solución Ejercicio 2:

A

```
mysql> CREATE FUNCTION diasemana(num int) RETURNS CHAR(10)
-> BEGIN
-> DECLARE dia char(10);
->
-> IF (num=1) THEN SET dia='Domingo';
-> ELSEIF (num=2) THEN SET dia='Lunes';
-> ELSEIF (num=3) THEN SET dia='Martes';
-> ELSEIF (num=4) THEN SET dia='Miercoles';
-> ELSEIF (num=5) THEN SET dia='Jueves';
-> ELSEIF (num=6) THEN SET dia='Viernes';
-> ELSEIF (num=7) THEN SET dia='Sabado';
->
-> ELSE SET dia='ERROR';
-> END IF;
->
-> RETURN dia;
-> END; //
Query OK, 0 rows affected (0.03 sec)

mysql> DELIMITER ;
mysql> select diasemana(3)
-> ;
+-----+
| diasemana(3) |
+-----+
| Martes      |
+-----+
1 row in set (0.01 sec)
```

B

```
mysql> CREATE FUNCTION diasemana2(num int) RETURNS CHAR(10)
-> BEGIN
-> DECLARE dia char(10);
->
-> CASE num
->     WHEN 1 THEN SET dia='Domingo';
->     WHEN 2 THEN SET dia='Lunes';
->     WHEN 3 THEN SET dia='Martes';
->     WHEN 4 THEN SET dia='Miercoles';
->     WHEN 5 THEN SET dia='Jueves';
->     WHEN 6 THEN SET dia='Viernes';
->     WHEN 7 THEN SET dia='Sabado';
->
->     ELSE SET dia='ERROR';
-> END CASE;
->
-> RETURN dia;
-> END; //
Query OK, 0 rows affected (0.05 sec)

mysql> DELIMITER ;
mysql> select diasemana2(5);
+-----+
| diasemana2(5) |
+-----+
| Jueves        |
+-----+
1 row in set (0.08 sec)
```

Solución Ejercicio 3:

```
mysql> CREATE FUNCTION mayor(num1 int,num2 int,num3 int)
-> RETURNS INT
-> BEGIN
->   DECLARE max INT;
->
->   IF (<num1>num2) and (<num1>num3)) then set max=num1;
->   ELSEIF (<num2>num1) and (<num2>num3)) then set max=num2;
->   ELSE set max=num3;
->   END IF;
->
->   RETURN max;
-> END; //
Query OK, 0 rows affected (0.05 sec)

mysql> DELIMITER ;
mysql> select mayor(20,6,100);
+-----+
| mayor(20,6,100) |
+-----+
|             100 |
+-----+
1 row in set (0.03 sec)
```

Solución Ejercicio 4:

```
mysql> CREATE FUNCTION palindroma(palabra char(20))
-> RETURNS BOOL
-> BEGIN
->
-> IF palabra=REVERSE(Palabra) then return TRUE;
-> ELSE return FALSE;
-> END IF;
->
-> END; //
```

Query OK, 0 rows affected (0.03 sec)

```
mysql> DELIMITER ;
mysql> select palindroma('ANA');
```

palindroma('ANA')
1

1 row in set (0.01 sec)

5. Instrucciones Repetitivas

TEMA 6: CONSTRUCCIÓN DE SCRIPTS



Daniel Rodríguez Fernández

5. Instrucciones Repetitivas

Simple LOOP

```
[etiqueta:] LOOP
instrucciones
END LOOP
[etiqueta];
```

Etiqueta: Permite etiquetar el loop para podernos referir a el dentro del bloque.

LEAVE [etiqueta] : Termina el loop

```
DELIMITER //
CREATE PROCEDURE proc9()
BEGIN
    DECLARE cont INT;
    SET cont=0;
    loop_label: LOOP
        INSERT INTO t VALUES(cont);
        SET cont=cont+1;
        IF cont>=5 THEN
            LEAVE loop_label;
        END IF;
    END LOOP;
END;
```

```
mysql> DELIMITER ;
mysql> call proc9();
Query OK, 1 row affected (0.10 sec)

mysql> select * from t;
+-----+
| cod |
+-----+
| 0 |
| 1 |
| 2 |
| 3 |
| 4 |
+-----+
5 rows in set (0.01 sec)
```

WHILE LOOP

```
[etiqueta:] WHILE expresion DO
instrucciones
END WHILE [etiqueta];
```

```
DELIMITER //
CREATE PROCEDURE proc11()
BEGIN
    DECLARE i INT;
    SET i=0;
    loop2: WHILE i<=10 DO
        IF MOD(i,2)<>0 THEN
            select concat(i,' IMPAR');
        END IF;
        SET i=i+1;
    END WHILE loop2;
END;
```

```
mysql> call proc11();//
+-----+
| concat(i,' IMPAR') |
+-----+
| 1 IMPAR            |
+-----+
1 row in set (0.00 sec)

+-----+
| concat(i,' IMPAR') |
+-----+
| 3 IMPAR            |
+-----+
1 row in set (0.02 sec)

+-----+
| concat(i,' IMPAR') |
+-----+
| 5 IMPAR            |
+-----+
1 row in set (0.03 sec)

+-----+
| concat(i,' IMPAR') |
+-----+
| 7 IMPAR            |
+-----+
1 row in set (0.05 sec)
```

Ejercicio:

1. Crea una función que calcule la media de los 10 primeros números impares. Realiza de dos formas con: función y procedimiento.



Solución Ejercicio:

FUNCIÓN:

```
C:\Program Files\MySQL\MySQL Server 5.1\bin\mysql.exe

mysql> DELIMITER //
mysql> DROP FUNCTION ejercicio5//
Query OK, 0 rows affected (0.00 sec)

mysql> CREATE FUNCTION ejercicio5() RETURNS double
-> BEGIN
->   DECLARE num, cont INT;
->   DECLARE sum double;
->   SET num=1;
->   SET sum=0;
->   SET cont=0;
->   WHILE cont<=10 DO
->     IF <MOD(num,2)=1> THEN
->       SET sum=sum+num;
->       SET cont=cont +1;
->     END IF;
->     SET num=num+1;
->   END WHILE;
->   return (sum/cont);
-> END;//
Query OK, 0 rows affected (0.01 sec)

mysql> select ejercicio5() as media//
+-----+
| media |
+-----+
|    11 |
+-----+
1 row in set (0.01 sec)

mysql>
```

PROCEDIMIENTO:

```
C:\Program Files\MySQL\MySQL Server 5.1\bin\mysql.exe

mysql> CREATE Procedure ejercicio6()
-> BEGIN
->   DECLARE num, cont INT;
->   DECLARE sum double;
->   SET num=1;
->   SET sum=0;
->   SET cont=0;
->   WHILE cont<=10 DO
->     IF <MOD(num,2)=1> THEN
->       SET sum=sum+num;
->       SET cont=cont +1;
->     END IF;
->     SET num=num+1;
->   END WHILE;
->   SELECT sum,cont,num;
->   SELECT (sum/cont) as media;
-> END;//
Query OK, 0 rows affected (0.01 sec)

mysql> call ejercicio6();//
+-----+-----+-----+
| sum | cont | num |
+-----+-----+-----+
| 121 | 11 | 22 |
+-----+-----+-----+
1 row in set (0.00 sec)

+-----+
| media |
+-----+
|    11 |
+-----+
1 row in set (0.02 sec)

Query OK, 0 rows affected (0.03 sec)

mysql>
```

Extra:

```
DELIMITER $$
DROP PROCEDURE IF EXISTS cursor_demo$$
CREATE PROCEDURE cursor_demo()
BEGIN
    DECLARE tmp VARCHAR(20);
    DECLARE ufe BOOL;
    DECLARE c1 cursor FOR
        Select nombre from equipos;

    DECLARE CONTINUE HANDLER FOR
        NOT FOUND SET ufe=1;

    SET ufe=0;

    OPEN c1;

    Bucle:LOOP
        FETCH c1 INTO tmp;
        IF ufe=1 THEN
            LEAVE Bucle;
        END IF;
        select tmp;
    END LOOP Bucle;
    CLOSE c1;
END;$$
```



Procedimiento que muestre el DNI , la cuenta y el saldo de los clientes con saldo negativo en alguna de sus cuentas (BDD ebanca)

```
DELIMITER $$
CREATE PROCEDURE saldo_neg()
BEGIN
    DECLARE ufe BOOL;
    DECLARE vsaldo,vcod_cliente,vcuenta INT;
    DECLARE csaldo CURSOR FOR
        SELECT saldo, cod_cliente, cod_cuenta FROM cuentas;
    DECLARE CONTINUE HANDLER FOR NOT FOUND SET ufe=1;
    SET ufe=0;
    OPEN csaldo;
    Bucle:LOOP
        FETCH csaldo INTO vsaldo, vcod_cliente, vcuenta;
        IF vsaldo<0 THEN
            SELECT dni,vsaldo,vcuenta FROM clientes
            WHERE codigo_cliente= vcod_cliente;
        END IF;
        IF ufe=1 THEN LEAVE Bucle;
        END IF;
    END LOOP Bucle;
    CLOSE csaldo;
END;$$
```

Procedimiento que muestre el nombre del autor que mas noticias ha publicado el 19 de noviembre del 2009. (BDD motorblog)

```
DELIMITER $$
CREATE PROCEDURE noticias_mes()
BEGIN
    DECLARE autor VARCHAR(20);
    DECLARE ufe BOOL;
    DECLARE cnombre VARCHAR(20);
    DECLARE cid, numnoticias, maxnoticias INT;
    DECLARE cursorA CURSOR FOR SELECT id_autor, login FROM autores;
    DECLARE CONTINUE HANDLER FOR NOT FOUND SET ufe=1;
    SET ufe=0,maxnoticias=1;
    OPEN cursorA;
    Bucle:LOOP
        FETCH cursorA INTO cid,cnombre;
        select count(*) INTO numnoticias FROM noticias
        WHERE autor_id=cid AND fecha_pub like '2009-11-19%';
        IF numnoticias>maxnoticias THEN
            SET maxnoticias=numnoticias;
            SET autor=cnombre;
        END IF;
        IF ufe=1 THEN LEAVE Bucle;
        END IF;
    END LOOP Bucle;
    CLOSE cursorA;
    select autor;
END;$$
```