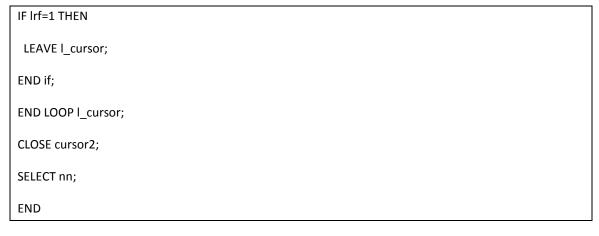
CURSORES

16. obtener_datos. Llamada: CALL obtener_datos("donoso anton");

DELIMITER \$\$
CREATE DEFINER=`root`@`localhost` PROCEDURE `obtener_datos`(id_nombre VARCHAR(25))
BEGIN
DECLARE ed INTEGER;
DECLARE casa TEXT;
SELECT edad, alojamiento INTO ed, casa FROM empleado WHERE nombre=id_nombre;
SELECT ed, casa;
END
17. cursor_demo3. Llamada: CALL cursor_demo3();
DELIMITER \$\$
CREATE DEFINER=`root`@`localhost` PROCEDURE `cursor_demo3`()
BEGIN
DECLARE tmp VARCHAR(200);
DECLARE Irf BOOL;
DECLARE nn INT;
DECLARE cursor2 CURSOR FOR SELECT NOMBRE FROM empleado;
DECLARE CONTINUE HANDLER FOR NOT FOUND SET Irf=1;
SET Irf=0; SET nn=0;
OPEN cursor2;
I_cursor: LOOP
FETCH cursor2 INTO tmp;
SET nn=nn+1;



18. calificación_empleado. Llamada: CALL calificación_empleado ();

```
DELIMITER $$
CREATE DEFINER='root'@'localhost' PROCEDURE 'calificacion_empleado'()
  READS SQL DATA
BEGIN
DECLARE nom TEXT;
DECLARE na_count int;
DECLARE fin BOOL;
DECLARE nn INT;
DECLARE empleado_nom CURSOR FOR SELECT NOMBRE FROM empleado;
DECLARE oficio_em CURSOR FOR SELECT NOMBRE FROM oficioempleado WHERE nombre=nom;
DECLARE CONTINUE HANDLER FOR NOT FOUND SET fin=1;
SET na_count=0;
OPEN empleado_nom;
empl_loop: LOOP
FETCH empleado_nom INTO nom;
IF fin=1 THEN leave empl_loop;
end if;
OPEN oficio_em;
```

```
SET na_count=0;

oficioem_loop: LOOP

fetch oficio_em INTO nom;

IF fin=1 THEN LEAVE oficioem_loop;

END IF;

set na_count=na_count+1;

END LOOP oficioem_loop;

CLOSE oficio_em;

SET fin=0;

SELECT CONCAT("el EMPLEADO ", nom, " tiene ", na_count, " empleos");

END LOOP empl_loop;

CLOSE empleado_nom;

END
```

MANEJO DE ERRORES

19. insertar_registro_en_t. Llamada: CALL insertar_registro_en_t (33,7); CALL insertar_registro_en_t (0,7); Este último da error por existir ya un registro con esa clave 0. EL error es el **1062**

```
DELIMITER $$

CREATE DEFINER=`root`@`localhost` PROCEDURE `insertar_registro_en_t`(in clave int, in valor int)

MODIFIES SQL DATA

BEGIN

INSERT INTO t(C, S1) values (clave, valor);

END
```

20. insertar_registro_en_t_2. Llamada: CALL insertar_registro_en_t_2 (37,7,@es); luego vemos lo que vale @es con: SELECT @es. Probamos con una Primary Key existente y lo vemos también.

```
DELIMITER $$
```

CREATE DEFINER='root'@'localhost' PROCEDURE 'insertar_registro_en_t_2'(in clave int, in valor int,

OUT estado VARCHAR(45))

MODIFIES SQL DATA

```
BEGIN

DECLARE CONTINUE HANDLER FOR 1062 SET estado ="Entrada Duplicada";

SET estado = "OK";

INSERT INTO t(C, S1) values (clave, valor);

END
```

21. insertar_registro_en_t_3. Llamada: CALL insertar_registro_en_t_3 (39,7); luego volver a llamarlo igual y ver los mensajes de error.

```
DELIMITER $$

CREATE DEFINER='root'@'localhost' PROCEDURE 'insertar_registro_en_t_3' (in clave int, in valor int
)

MODIFIES SQL DATA

BEGIN

DECLARE duplicate_key INT DEFAULT 0;

BEGIN

DECLARE CONTINUE HANDLER FOR 1062 SET duplicate_key=1;

INSERT INTO t(C, S1) values (clave, valor);

END;

IF duplicate_key=1 THEN

select concat("error en la inserción de la clave duplicada") as "Resultado";

ELSE select concat ("Registro ",clave ," creado ");

END IF;

END
```