1.    Escribe una función que **sume tres números** pasados como parámetros, pero teniendo en cuenta que el valor de cualquiera de los parámetros puede ser NULL. No se debe usar función adicional.

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

BEGIN

CASE

WHEN N1 IS NULL AND N2 IS NOT NULL AND N3 IS NOT NULL THEN

RETURN N2 + N3;

WHEN N1 IS NOT NULL AND N2 IS NULL AND N3 IS NOT NULL THEN

RETURN N1 + N3;

WHEN N1 IS NOT NULL AND N2 IS NOT NULL AND N3 IS NULL THEN

RETURN N1 + N2;

WHEN N1 IS NULL AND N2 IS NULL AND N3 IS NOT NULL THEN

RETURN N3;

WHEN N1 IS NULL AND N2 IS NOT NULL AND N3 IS NULL THEN

RETURN N2;

WHEN N1 IS NOT NULL AND N2 IS NULL AND N3 IS NULL THEN

RETURN N1;

ELSE

RETURN N1 + N2 + N3;

END CASE;

END $2.     Codifica una función a la cual se le pasa una **nota numérica y devuelve la correspondiente interpretación literal**: *1-4.99 Suspenso 5-6 Suficiente 6-7 Bien 7-9 Notable 9-10 Sobresaliente Resto->ERROR*.

CREATE FUNCTION NOTA(N FLOAT) RETURNS VARCHAR(20)

BEGIN

CASE

WHEN N >= 1 AND N < 5 THEN

RETURN 'SUSPENSO';

WHEN N >= 5 AND N < 6 THEN

RETURN 'SUFICIENTE';

WHEN N >= 6 AND N < 7 THEN

RETURN 'BIEN';

WHEN N >= 7 AND N < 9 THEN

RETURN 'NOTABLE';

WHEN N >= 9 AND N <= 10 THEN

RETURN 'SOBRESALIENTE';

ELSE

RETURN 'ERROR';

END CASE;

END$

3.    Escribe un procedimiento que **reciba una palabra y la devuelva escrita del revés**.

CREATE PROCEDURE REVES(IN PALABRA VARCHAR(20), OUT PALABRA\_REVES VARCHAR(20))

BEGIN

DECLARE I INT DEFAULT 0;

SET PALABRA\_REVES = '';

WHILE I <= LENGTH(PALABRA) DO

SET PALABRA\_REVES = CONCAT(SUBSTRING(PALABRA, I, 1), PALABRA\_REVES);

SET I = I + 1;

END WHILE;

END $

**BD: *Obras***

**4.** Crea un procedimiento para **aumentar el precio de la hora de las máquinas** (preciohora) en un porcentaje, dicho porcentaje se le pasa al procedimiento como parámetro de entrada**.**

CREATE PROCEDURE AUMENTAR(IN PORCENTAJE INT)

BEGIN

UPDATE MAQUINAS

SET PRECIOHORA = PRECIOHORA + (PRECIOHORA \* (PORCENTAJE / 100));

END $

**5.** Crea un procedimiento para **cambiar la categoría de los conductores** (categ). El procedimiento debe tener dos parámetros de entrada, el primero indica si es subida o bajada de categoría (1 subida , 2 bajada), y el segundo parámetro indica el valor de la subida o bajada de categoría**.**

CREATE PROCEDURE CAMBIAR\_CAT(IN SUB\_BAJ INT, IN VALOR INT)

BEGIN

IF SUB\_BAJ = 1 THEN

UPDATE CONDUCTORES

SET CATEG = CATEG + VALOR;

ELSEIF SUB\_BAJ = 2 THEN

UPDATE CONDUCTORES

SET CATEG = CATEG - VALOR;

END IF;

END $

6.    Para aquel conductor que **menos horas haya realizado**, asignarle un trabajo para **hoy** con un numero de **horas suficiente para que sea el que más horas ha realizado**.

-- POR TRABAJO --

CREATE PROCEDURE ASIGNAR\_TRABAJO(IN MAQUINA CHAR(3), IN PROYECTO CHAR(3))

BEGIN

DECLARE COD CHAR(3) DEFAULT (SELECT CODC

FROM TRABAJOS

WHERE TIEMPO = (SELECT MIN(TIEMPO)

FROM TRABAJOS));

DECLARE MAX\_TIEMPO INT DEFAULT (SELECT MAX(TIEMPO) FROM TRABAJOS);

INSERT INTO TRABAJOS

VALUES (COD, MAQUINA, PROYECTO, CURRENT\_DATE, MAX\_TIEMPO + 1);

END $

-- EN TOTAL –

CREATE PROCEDURE ASIGNAR\_TRABAJO\_TOT(IN MAQUINA CHAR(3), IN PROYECTO CHAR(3))

BEGIN

DECLARE COD CHAR(3) DEFAULT (SELECT CODC

FROM TRABAJOS

WHERE TIEMPO = (SELECT MIN(TIEMPO) FROM TRABAJOS));

DECLARE MAX\_TIEMPO INT DEFAULT (SELECT SUM(TIEMPO)

FROM TRABAJOS

GROUP BY CODC

ORDER BY SUM(TIEMPO) DESC

LIMIT 1);

DECLARE TIEMPO\_COD INT DEFAULT (SELECT SUM(TIEMPO)

FROM TRABAJOS

WHERE CODC LIKE COD

GROUP BY CODC);

INSERT INTO TRABAJOS

VALUES (COD, MAQUINA, PROYECTO, CURRENT\_DATE, (MAX\_TIEMPO - TIEMPO\_COD) + 1);

END $

**BD: *Gestión de HOSPITALES***

7.    Insertar un empleado en la tabla EMP. Su número de empleado será superior a los existentes y la fecha de incorporación a la empresa será la actual.

CREATE PROCEDURE INSERTAR\_EMPLEADO(IN APE VARCHAR(16), IN OF VARCHAR(10), IN SAL INT, IN DEPT INT)

BEGIN

DECLARE NUM\_EMP INT DEFAULT ((SELECT MAX(EMP\_NO) FROM EMP) + 1);

INSERT INTO EMP

VALUES(NUM\_EMP, APE, OF, NULL, CURRENT\_DATE, SAL, NULL, DEPT);

END $

8.    Sobre la tabla EMP insertar un nuevo 'VENDEDOR'. Su número de empleado será consecutivo al mayor de los existentes. Su departamento será el mismo que el de su jefe 'SALA'.

CREATE PROCEDURE MODIFICIAR()

BEGIN

DECLARE NUM\_EMP INT DEFAULT ((SELECT MAX(EMP\_NO) FROM EMP) + 1);

DECLARE DEPARTAMENTO INT DEFAULT (SELECT DEPT\_NO

FROM EMP

WHERE APELLIDO LIKE 'SALA');

INSERT INTO EMP

VALUES(NUM\_EMP, APE, 'VENDEDOR', NULL, CURRENT\_DATE, SAL, NULL, DEPARTAMENTO);

END $

 9.    Modificar el número de departamento de 'BLAIR'. El nuevo departamento será el departamento donde hay más empleados cuyo oficio sea ‘*Vendedor’*.

CREATE PROCEDURE MODIFICAR()

BEGIN

DECLARE NUM\_EMP INT DEFAULT ((SELECT MAX(EMP\_NO) FROM EMP) + 1);

DECLARE DEPARTAMENTO INT DEFAULT (SELECT DEPT\_NO

FROM EMP

WHERE OFICIO LIKE 'VENDEDOR'

GROUP BY DEPT\_NO

HAVING COUNT(DEPT\_NO));

UPDATE EMP

SET DEPT\_NO = DEPARTAMENTO

WHERE APELLIDO LIKE 'BLAIR';

END $