PROCEDIMIENTOS CON EXPRESIONES REPETITIVAS

/*Sintaxis PL/MySQL: */

```
WHILE Expression DO
        <conjunto de instrucciones>
END WHILE;


REPEAT
        <conjunto de instrucciones>
UNTIL expresion
END REPEAT;
```

EJERCICIO Nº 1:

/* A. Procedimiento "numeros" que muestre los 20 primeros números enteros */

```
/*-- Opción 1: WHILE */
DROP PROCEDURE IF EXISTS numeros;
DELIMITER //

CREATE PROCEDURE numeros()
BEGIN
        DECLARE i INTEGER; -- Opción alternativa: DECLARE i INTEGER DEFAULT 1;
        SET i = 1;
        WHILE i <= 20 DO
                SELECT i;
                SET i = i + 1;
        END WHILE;
END//
DELIMITER ;
CALL numeros();
```

REALIZA LA OPCIÓN CON REPEAT

/* B- Realiza un Procedimiento "numeros" que muestre los 20 primeros números enteros PERO CON LA OPCIÓN DE REPEAT */

/* REALIZA EXPLICA QUE SE ESTÁ REALIZANDO EN ESTE CÓDIGO */

/*C- Opción 3: con cadenas */

```
DROP PROCEDURE IF EXISTS numeros;
DELIMITER //
CREATE PROCEDURE numeros()
BEGIN
	DECLARE i INTEGER DEFAULT 1;
	DECLARE nums VARCHAR(100) DEFAULT "";

	WHILE i <= 20 DO
		SET nums = CONCAT(nums, " ", i);
		SET i = i + 1;
	END WHILE;

	SELECT nums AS numeros;
END//
DELIMITER ;
```

/* EJERCICIO Nº 2. Procedimiento "impares" que muestre los 10 primeros números impares. Usar función MOD. */

/* A: OPCIÓN DE REPEAT */

```
DROP PROCEDURE IF EXISTS impares;
DELIMITER //
CREATE PROCEDURE impares()
BEGIN
	DECLARE i INTEGER; -- Opción alternativa: DECLARE i INTEGER DEFAULT 0;
	SET i = 0;
	REPEAT
		SET i = i + 1;
		IF MOD(i,2) <> 0 THEN
			SELECT i;
		END IF;
	UNTIL i >= 20
	END REPEAT;
END//
DELIMITER ;
```

/* B: REALIZA LA OPCIÓN DE WHILE: Procedimiento "impares" que muestre los 10 primeros números impares. Usar función MOD */

/*-- Opción 3: usando cadena*/

/* EXPLICA LO QUE AQUÍ SE EXPRESA EN EL SIGUIENTE CÓDIGO:*/

```
DROP PROCEDURE IF EXISTS impares;

DELIMITER //
CREATE PROCEDURE impares()
BEGIN
	DECLARE i INTEGER;
	DECLARE impar VARCHAR(100) DEFAULT "";
	SET i = 1;

	WHILE i <= 20 DO
		IF MOD(i,2) <> 0 THEN
			SET impar = CONCAT(impar," ",i);
		END IF;
		SET i = i + 1;
	END WHILE;

	-- Mostrar resultado
	SELECT impar AS numeros_impares;
END//
DELIMITER ;
```

EJERCICIO Nº 3. Procedimiento "generar_cuentas" que dado un número n y su nombre de dominio permita obtener n cuentas del tipo usu_1@dominio, usu_2@dominio, ... Se deberá comprobar que el dominio no sea vacío y el número n sea entero positivo.

Ejemplo:

CALL generar_cuentas(3, 'ies.org')

***/**

-- Opción 1

DROP PROCEDURE IF EXISTS generar_cuentas;

DELIMITER //

OPCIÓN A:

```
CREATE PROCEDURE generar_cuentas(n INTEGER, dominio VARCHAR(50))
BEGIN
	DECLARE i INTEGER DEFAULT 0;
	IF (n > 0) THEN
		IF (dominio IS NOT NULL AND CHAR_LENGTH(dominio) > 0)
THEN
			WHILE (i < n) DO
				SET i = i + 1;
				SELECT CONCAT('usu_', i, '@', dominio) AS email;
			END WHILE;
		ELSE
			SIGNAL SQLSTATE '45000' SET message_text='El
dominio no debe ser vacío';
		END IF;
```

```
        ELSE
                SIGNAL SQLSTATE '45000' SET message_text='N debe ser un
valor positivo';
        END IF;
END//
DELIMITER ;
```

OPCIÓN B:

Repite el procedimiento de generar cuenta pero en vez de realizarlo con un WHILE lo vamos a realizar con un REPEAT

EJERCICIO Nº 4.

```
/* REALIZA UN Procedimiento "nimpares" que dado un parámetro N
muestre los N primeros números impares en el siguiente formato "1, 3, 5,
7, ...". */

/*PODRÍAS HACERLO MEDIANTE UNA SENTENCIA WHILE O MEDIANTE UN
REPEAT*/

DROP PROCEDURE IF EXISTS nimpares;
```