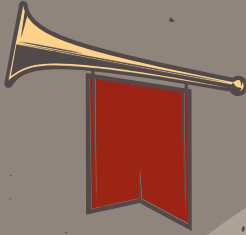


2022



BASE DE DATOS II

Estudiante:

Univ. Marco Antonio Calle Vaquiata

Tabla De Contenido

01 PARTE TEORICA

Se explicara conceptos elementales sobre El lenguaje procedural en DBA.

02 PARTE PRACTICA

Se presentara la aplicacion de la parte teorica para la resolucion de requerimientos.





INTRODUCCION

Las bases de datos son un elemento fundamental en el entorno informático hoy en día y tienen aplicación en la práctica totalidad de campos. Además, el trabajo con un SIG presenta una serie de características que hacen que sea recomendable el uso de bases de datos.

Aunque la realidad es que todavía se efectúa mucho trabajo SIG sin emplear bases de datos, la naturaleza propia de los proyectos SIG y la progresiva implantación de los SIG a niveles más allá del uso personal traen ambas consigo un uso cada vez mayor de las bases de datos, y por tanto una mayor necesidad de conocer el funcionamiento de estas.



Manejo de Conceptos

Defina que es lenguaje procedural en MySQL

Es en síntesis la programación a nivel de DBA, en donde generamos distintas estructuras de control dentro de funciones almacenadas.

Defina que es una función en MySQL

Es un proceso usado o creado para tomar parámetros y transformarlos en salidas; el proceso es rutinario.

¿Qué cosas características debe de tener una función?

- La creación de la función.
- Asignación de un nombre y si es requerido, parámetros de entrada.
- Definición del tipo de dato que se retorna.
- La selección de la función.

¿Cómo crear, modificar y cómo eliminar una función?

CREA:

```
CREATE FUNCTION nombre()
```

MODIFICA:

```
CREATE OR REPLACE FUNCTION nombre()
```

ELIMINA:

```
DROP FUNCTION comparacion;
```

```
DROP FUNCTION IF EXISTS comparacion;
```

Para qué sirve la función CONCAT y como funciona en MYSQL

CONCAT es utilizado para unir todos los datos de tipo cadena.

```
CREATE OR REPLACE FUNCTION ejemplo(cadena TEXT, cadena2 TEXT, cadena3 TEXT)
RETURNS TEXT
BEGIN
    RETURN(CONCAT(cadena, cadena2, cadena3));
end;

SELECT ejemplo( cadena: 'Ronald', cadena2: 'Choque', cadena3: '2022');
```

```
ejemplo('Ronald', 'Choque', '2022')
```

```
1 RonaldChoque2022
```

Para qué sirve la función SUBSTRING y como funciona en MYSQL

SUBSTRING recupera los elementos que se le indican de una cadena principal (conteo).

```
CREATE OR REPLACE FUNCTION ejemplo(cadena TEXT)
RETURNS TEXT
BEGIN
    RETURN(SUBSTR(cadena, 1, 6));
end;

SELECT ejemplo( cadena: 'Ronald Choque P.');
```

```
ejemplo('Ronald Choque P.')
```

```
1 Ronald
```

Para qué sirve la función **STRCMP** y como funciona en **MYSQL**

STRCMP es una función de tipo booleano que presenta dos respuestas:

- 0 en TRUE.
- 1 o -1 en FALSE.

```
CREATE OR REPLACE FUNCTION consulta(c1 TEXT, c2 TEXT, c3 TEXT)
RETURNS TEXT
BEGIN
    IF STRCMP(c1, c2) = 0 OR STRCMP(c1, c3) = 0
    then
        RETURN 'CADENAS IGUALES';
    else
        RETURN 'CADENAS DIFERENTES';
    end if;
end;
```

```
SELECT consulta(c1: 'dba iii', c2: 'dba i', c3: 'dba iii');
```

Para qué sirve la función **CHAR_LENGTH** y **LOCATE** y como funciona en **MYSQL**

- CHAR_LENGTH contabiliza los caracteres de una cadena, devuelve un INT.
- LOCATE realiza la búsqueda de un dato dentro de otro.

```
create or replace function pregunta(p1 text, p2 text)
returns text
begin
    declare position text default '';
    DECLARE con TEXT DEFAULT CONCAT(p1, ' ', p2);
    DECLARE car INT DEFAULT CHAR_LENGTH(con);
    set position = CONCAT('Posicion: ', locate(p1, p2), ' || N de datos: ', car);
    return position;
end;
```

```
select pregunta( p1: 'II', p2: 'Base de Datos II');
```

¿Cual es la diferencia entre las funciones de agregación y funciones creados por el DBA?

- AGREGACION son aquellas que funcionan bajo la cláusula SELECT, aplicado a un grupo de registros y que devuelven un único valor.
- CUSTOM esta definida por el comando CREATE FUNCTION, su sintaxis y uso esta completamente definida por el USER.

¿Busque y defina a qué se referirá cuando se habla de parámetros de entrada y salida en MySQL?

- ENTRADA O INPUT refiere a la introducción de datos.
- SALIDA O OUTPUT refiere a la muestra de los mismos.

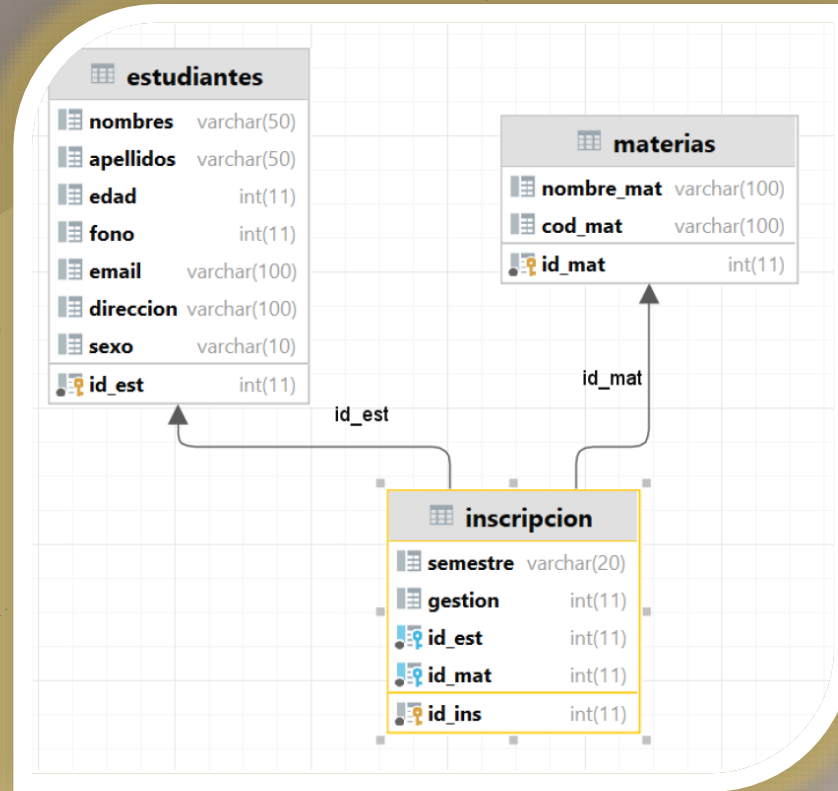
```
CREATE OR REPLACE FUNCTION ejemplo(cadena TEXT)
RETURNS TEXT
BEGIN
    RETURN(SUBSTR(cadena, 1, 6));
end;

SELECT ejemplo(cadena: 'Ronald Choque P.');
```


02 PARTE PRACTICA



Crear la siguiente Base de datos y sus registros.



CREACION DE LA DBA.



```
CREATE DATABASE evaluacion;  
USE evaluacion;
```

El comando CREATE TABLE es el que genera la DBA.

Comando USE para trabajar en el mismo.

PARA LA CREACION DE TABLAS:

- Definimos a los PRIMARY KEY.
- El comando FOREIGN KEY es utilizado para relacionar las tablas mediante los PRIMARY KEY.
- NOT NULL para no tener columnas sin registro.
- AUTO_INCREMENT para generar automáticamente el registro de la columna.

CREACION DE LAS TABLAS.

```
CREATE TABLE estudiantes
(
    id_est INTEGER AUTO_INCREMENT PRIMARY KEY NOT NULL,
    nombres VARCHAR(50) NOT NULL,
    apellidos VARCHAR(50) NOT NULL,
    edad INTEGER NOT NULL,
    gestion INTEGER,
    fono INTEGER NOT NULL,
    email VARCHAR(100) NOT NULL,
    direccion VARCHAR(100) NOT NULL,
    sexo VARCHAR(10) NOT NULL
);
```

```
CREATE TABLE materias
(
    id_mat INTEGER AUTO_INCREMENT PRIMARY KEY NOT NULL,
    nombre_mat VARCHAR(100) NOT NULL,
    cod_mat VARCHAR(100) NOT NULL
);
```

```
CREATE TABLE inscripcion
(
    id_ins INTEGER AUTO_INCREMENT PRIMARY KEY NOT NULL,
    semestre VARCHAR(20) NOT NULL,
    gestion INTEGER NOT NULL,
    id_est INTEGER NOT NULL,
    id_mat INTEGER NOT NULL,
    FOREIGN KEY (id_est) REFERENCES estudiantes (id_est),
    FOREIGN KEY (id_mat) REFERENCES materias (id_mat)
);
```

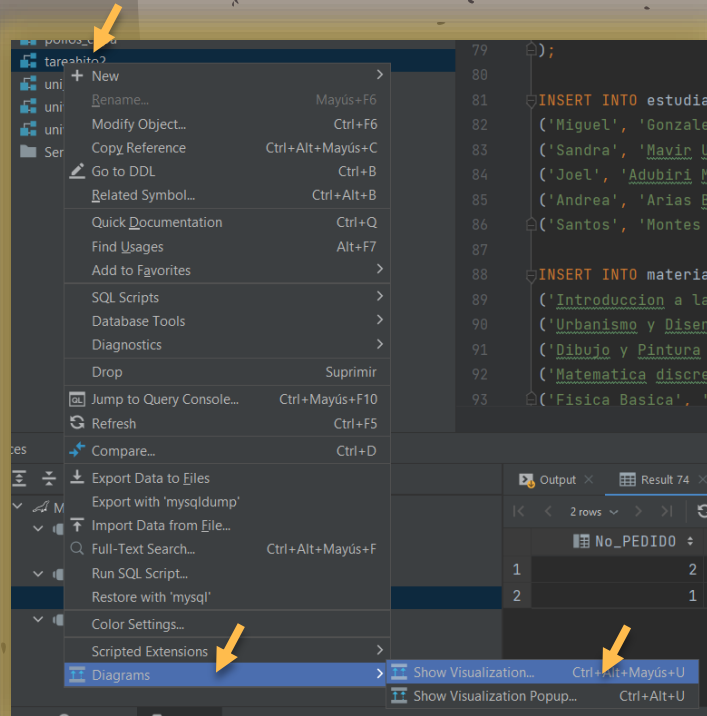
LLENADO DE LAS TABLAS.

```
INSERT INTO estudiantes (nombres, apellidos, edad, fono, email, direccion, sexo) VALUES
('Miguel', 'Gonzales Veliz', 20, 2_832_115, 'miguel@gmail.com', 'Av. 6 de Agosto', 'masculino'),
('Sandra', 'Mavir Uria', 25, 2_832_116, 'sandra@gmail.com', 'Av. 6 de Agosto', 'femenino'),
('Joel', 'Adubiri Mondar', 30, 2_832_117, 'joel@gmail.com', 'Av. 6 de Agosto', 'masculino'),
('Andrea', 'Arias Ballesteros', 21, 2_832_118, 'andrea@gmail.com', 'Av. 6 de Agosto', 'femenino'),
('Santos', 'Montes Valenzuela', 24, 2_832_119, 'santos@gmail.com', 'Av. 6 de Agosto', 'masculino');
```

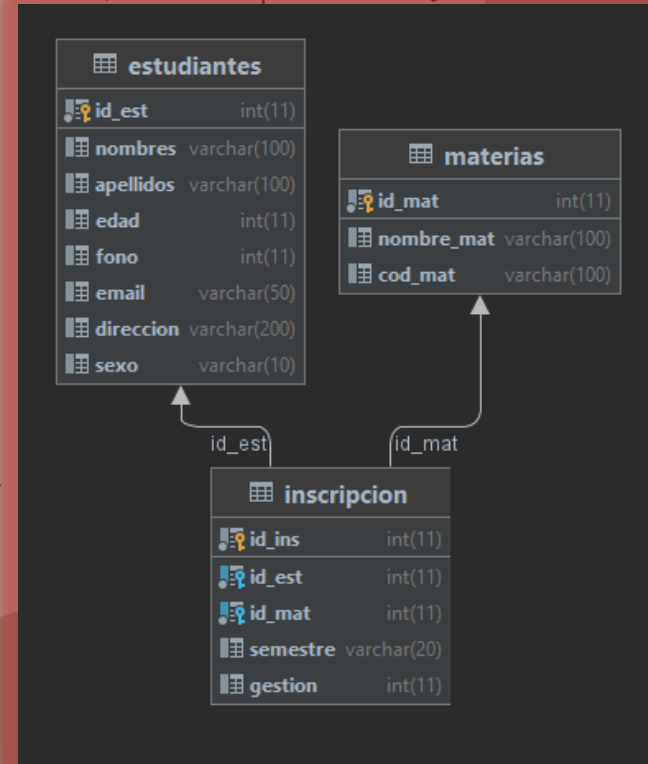
```
INSERT INTO materias (nombre_mat, cod_mat) VALUES
('Introduccion a la Arquitectura', 'ARQ-101'),
('Urbanismo y Diseno', 'ARQ-102'),
('Dibujo y Pintura Arquitectonico', 'ARQ-103'),
('Matematica discreta', 'ARQ-104'),
('Fisica Basica', 'ARQ-105');
```

```
INSERT INTO inscripcion (id_est, id_mat, semestre, gestion) VALUES
(1, 1, '1er Semestre', 2018),
(1, 2, '2do Semestre', 2018),
(2, 4, '1er Semestre', 2019),
(2, 3, '2do Semestre', 2019),
(3, 3, '2do Semestre', 2020),
(3, 1, '3er Semestre', 2020),
(4, 4, '4to Semestre', 2021),
(5, 5, '5to Semestre', 2021);
```

LLENADO DE LAS TABLAS.



Haciendo un clic derecho sobre la carpeta de tablas, vaya y seleccione DIAGRAMS y la opción SHOW VISUALIZATION para poder ver el diagrama.



Crear una función que genere la serie Fibonacci.

```
CREATE OR REPLACE FUNCTION seriefibonacci(limite INT)
RETURNS TEXT
BEGIN
    DECLARE resp TEXT DEFAULT '';
    DECLARE i INT DEFAULT 0;
    DECLARE siguiente INT DEFAULT 1;
    DECLARE uno INT DEFAULT 0;
    DECLARE dos INT DEFAULT 1;

    WHILE i <= limite DO
        SET i = i + 1;
        IF i <= 1
        THEN
            SET siguiente = i;
        ELSE
            SET siguiente = uno + dos;
            SET uno = dos;
            SET dos = siguiente;
        end if;
        SET resp = CONCAT(resp, siguiente, ' , ');
    end while;
    RETURN resp;
end;

SELECT seriefibonacci( limite: 7);
```

EJECUCION:

```
`seriefibonacci(7)`
1 1 , 1 , 2 , 3 , 5 , 8 , 13 , 21 ,
```

Crear una variable global a nivel BASE DE DATOS.

```
SET @userALL = 'SR.SKYNET';  
  
CREATE OR REPLACE FUNCTION variableglobal()  
RETURNS TEXT  
BEGIN  
    RETURN @userALL;  
end;  
  
SELECT variableglobal() AS ADMIN;
```

EJECUCION:

	ADMIN
1	SR.SKYNET

Crear una variable global a nivel BASE DE DATOS.

CONSIGNA:

En base a la edad menor en la tabla estudiantes, obtenga la serie de nueros ya sean pares o impares.

```
CREATE OR REPLACE FUNCTION min_edad_estudiantes()
RETURNS INT
BEGIN
    RETURN
    (
        SELECT MIN(est.edad)
        FROM estudiantes AS est
    );
end;

SELECT min_edad_estudiantes();
```



```
CREATE OR REPLACE FUNCTION serie_edad_parimpar(x INT)
RETURNS TEXT
BEGIN
    DECLARE str TEXT DEFAULT '';
    REPEAT
        IF x % 2 = 0
        THEN
            SET str = CONCAT(str, x, ' , ');
            SET x = x - 2;
        ELSE IF x % 2 = 1
        THEN
            SET str = CONCAT(str, x, ' , ');
            SET x = x - 2;
        end if;
    end if;
    UNTIL x <= 0
    end repeat;

    RETURN str;
end;
```

```
1 `serie_edad_parimpar(min_edad_estudiantes())`
1 20 , 18 , 16 , 14 , 12 , 10 , 8 , 6 , 4 , 2 ,
```

```
SELECT serie_edad_parimpar( x: min_edad_estudiantes());
```


Crear una variable global a nivel BASE DE DATOS.

```
CREATE OR REPLACE FUNCTION separaVocales(par1 TEXT)
RETURNS TEXT
BEGIN
    declare x int default 1;
    declare response text default '';
    declare letra char default '';
    declare limite int default char_length(par1);
    DECLARE a int default 0;
    DECLARE e int default 0;
    DECLARE i int default 0;
    DECLARE o int default 0;
    DECLARE u int default 0;
    while x <= limite do
        set letra = substring(par1, x, 1);
        if letra = 'a'
        then
            set a = a + 1;
        else if letra = 'e'
```

```
        set e = e + 1;
        else if letra = 'i'
        then
            set i = i + 1;
        else if letra = 'o'
        then
            set o = o + 1;
        else if letra = 'u'
        then
            set u = u + 1;
        end if; end if; end if; end if; end if;
        set x = x + 1;
    end while;
    set response = concat('A: ', a, ', ', 'E: ', e, ', ', 'I: ', i, ', ', 'O: ', o, ', ', 'U: ', u);
    return response;
end;

SELECT separaVocales( par1 'taller de base de datos') AS SEPARA_VOCALES;
```

Crear una variable global a nivel BASE DE DATOS.

CONSIGNA:

```
SELECT separaVocales( cadena: 'taller de base de datos') AS SEPARA_VOCALES;
```

Output SEPARA_VOCALES:text x

1 row

SEPARA_VOCALES
1 a: 3, e: 4, i: 0, o: 1, u: 0,

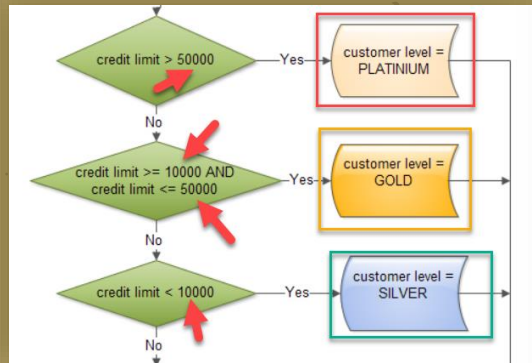
EJECUCION:

```
SEPARA_VOCALES
```

```
1 A: 3 ,E: 4 ,I: 0 ,O: 1 ,U: 0
```

Crear una función que recibe un parámetro INTEGER.

CONSIGNA:



```
CREATE OR REPLACE FUNCTION tipodeusuario(parametro INT)
RETURNS TEXT
BEGIN
    DECLARE credit_number INT DEFAULT 0;
    DECLARE respuesta TEXT DEFAULT '';
    SET credit_number = parametro;
    CASE
        WHEN credit_number > 50000 THEN SET respuesta = 'PLATINIUM';
        WHEN credit_number >= 10000 AND credit_number <= 50000 THEN SET respuesta = 'GOLD';
        WHEN credit_number < 10000 THEN SET respuesta = 'SILVER';
        else SET respuesta = 'usuario no IDENTIFICADO';
    END CASE;
    RETURN respuesta;
end;

SELECT tipodeusuario( parametro: 5000) AS VERIFICAR_TIPO_USUARIO;
```

EJECUCION:

```
VERIFICAR_TIPO_USUARIO
1 SILVER
```

Crear una función que reciba un parámetro TEXT

CONSIGNA:

LETTERS	
1	dbaii, baii, aii, ii, i,

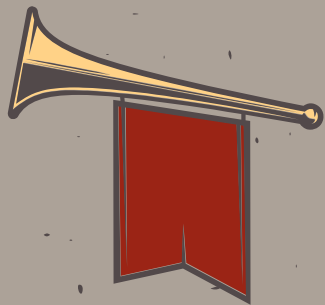
EJECUCION:

descomponerc('DBAII')	
1	I , II , AII , BAI , DBAI ,

```
create OR REPLACE FUNCTION descomponerc(x TEXT)
returns TEXT
begin
    DECLARE str TEXT DEFAULT '';
    DECLARE legth_concat INT DEFAULT CHAR_LENGTH(x);
    DECLARE limite INT DEFAULT 1;
    DECLARE sub INT DEFAULT legth_concat;

    REPEAT
        IF legth_concat >= limite
        THEN
            SET str = CONCAT(str, substr(x, legth_concat, sub - 1), ', ');
            SET legth_concat = legth_concat - 1;
        end if;
        UNTIL legth_concat <= 0
    end repeat;
    RETURN str;
END;

select descomponerc(x: 'DBAII');
```



¡GRACIAS!

