HITO4-DEFENSA

BASE DE DATOS II

Por: Marco Antonio Calle Vaquiata

INDICE

1: Manejo de Conceptos

2: Parte Practica

Manejo de conceptos

1. Defina que es lenguaje procedural en MySQL.

Lenguajes procuderales o procedimentales: El usuario da órdenes para que se realicen las tareas pertinentes con el objetico de recuperar los datos requeridos. Es la base del lenguaje de consulta SQL.

2. Defina que es una FUNCTION en MySQL

Las **funciones** son piezas de código que reciben datos de entrada, realizan operaciones con ellos y luego devuelven un resultado.

3. Cuál es la diferencia entre funciones y procedimientos almacenados.

Cuando llama al procedimiento almacenado, se debe especificar que es un parámetro externo. Una ventaja de los procedimientos almacenados es que puede obtener varios parámetros mientras que, en las funciones, solo se puede devolver una variable (función escalar) o una tabla (funciones con valores de tabla).

4. Cómo se ejecuta una función y un procedimiento almacenado.

Procedimiento almacenado: Es un objeto que **se** crea con la sentencia CREATE PROCEDURE y **se** invoca con la sentencia CALL . ...

Función almacenada: Es un objeto que **se** crea con la sentencia CREATE **FUNCTION** y **se** invoca con la sentencia SELECT o dentro de una expresión.

5. Defina que es una TRIGGER en MySQL

Un **trigger** o disparador es una regla que se asocia a una tabla. Mediante esta regla, se ejecutan una serie de instrucciones cuando se producen ciertos eventos sobre una tabla. Los eventos son: INSERT, UPDATE o DELETE. Para poder realizar **triggers** es necesario que tengas permisos para ejecutar esas consultas.

6. En un trigger que papel juega las variables OLD y NEW

La variable OLD hace referencia al valor de una columna antes de la incidencia se produzca; La variable NEW hace referencia a una columna afectada por la incidencia, una vez que haya pasado. Puede utilizar expresiones para realizar operaciones de lectura y asignación de valores a las variables de fila.

7. En un trigger que papel juega los conceptos(cláusulas) BEFORE o AFTER

Puede ser BEFORE (antes) o AFTER (despues), **para indicar que el disparador se ejecute antes o después que la sentencia que lo activa**.

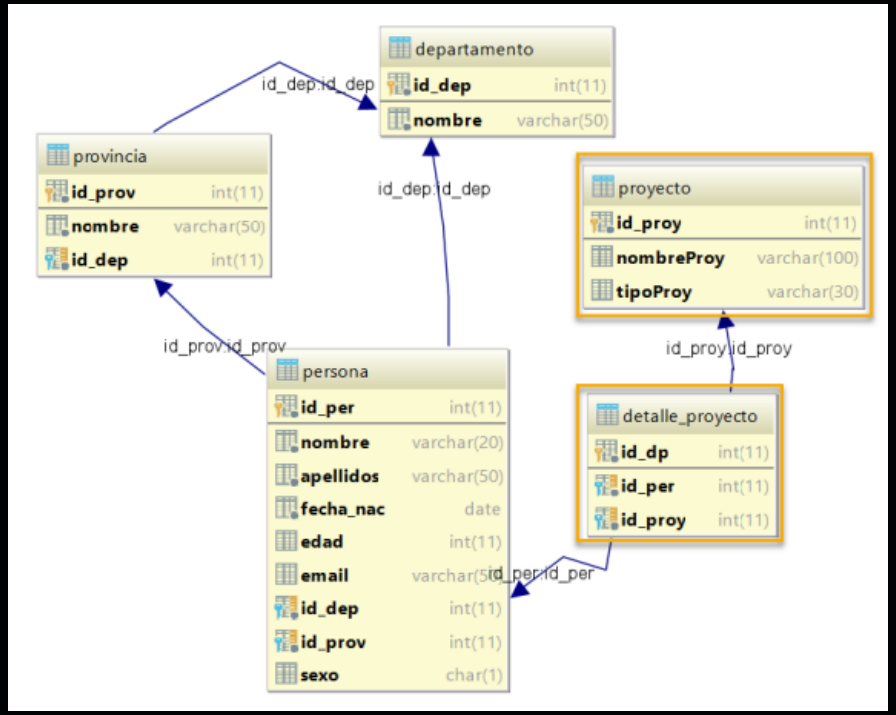
```
CREATE TRIGGER testref AFTER INSERT ON test1

FOR EACH ROW
```

8. A que se refiere cuando se habla de eventos en TRIGGERS

Un **trigger**, también conocido como disparador (Por su traducción al español) es un conjunto de sentencias SQL las cuales se ejecutan de forma automática cuando ocurre algún **evento** que modifique a una tabla.

PARTE PRACTICA

9. Crear la siguiente Base de datos y sus registros.



```sql
CREATE DATABASE DEFESA_HIT04;
USE DEFESA_HIT04;


CREATE TABLE  DEPARTAMENTO
(
    ID_DEP INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
    NOMBRE VARCHAR(50)
);

CREATE TABLE PROVINCIA
(
    ID_PROV INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
    NOMBRE VARCHAR(50),
    ID_DEP INT NOT NULL,
    FOREIGN KEY (ID_DEP) REFERENCES DEPARTAMENTO(ID_DEP)
);

CREATE TABLE PROYECTO
(
    ID_PROY INT NOT NULL PRIMARY KEY AUTO_INCREMENT,
    NOMBRE_PROY VARCHAR(100),
    TIPO_PROY VARCHAR(30)
);
```

```sql
CREATE TABLE DETALLE_PROYECTO
(
    ID_DP INTEGER NOT NULL PRIMARY KEY AUTO_INCREMENT,
    ID_PER INT NOT NULL ,
    ID_PROY INT NOT NULL ,

    FOREIGN KEY (ID_PROY) REFERENCES PROYECTO(ID_PROY),
    FOREIGN KEY (ID_PER) REFERENCES PERSONA(ID_PER)
);

CREATE TABLE PERSONA
(
  ID_PER INT NOT NULL AUTO_INCREMENT PRIMARY KEY ,
  NOMBRE VARCHAR(20),
  APELLIDOS VARCHAR(50),
  FECHA_NAC DATE,
  EDAD INT,
  EMAIL VARCHAR(50),
  ID_DEP INT NOT NULL ,
  ID_PROV INT NOT NULL,
  GENERO VARCHAR(1),
  FOREIGN KEY (ID_PROV) REFERENCES  PROVINCIA(ID_PROV),
  FOREIGN KEY (ID_DEP) REFERENCES DEPARTAMENTO(ID_DEP));
```

```sql
INSERT INTO  DEPARTAMENTO(NOMBRE)
VALUES ('LA PAZ'),
       ('SANTA CRUZ'),
       ('BENI'),
       ('ORURO'),
       ('CHUQUISACA');

INSERT INTO  PROVINCIA(NOMBRE, ID_DEP)
VALUES ('VIACHA','1'),
       ('ROBORE','2'),
       ('MAGDALENA','3'),
       ('CHALLAPATA','4'),
       ('TARABUCO','5');

INSERT INTO PERSONA (NOMBRE, APELLIDOS, FECHA_NAC, EDAD, EMAIL, ID_DEP, ID_PROV, GENERO)
VALUES ('MARCO','CALLE ','2002-11-21',23,'RODRIGO@GMAIL.COM',1,1,'M'),
       ('DANIELA','QUIROGA ','1999-12-16',25,'MARIA@GAMIL.COM',2,2,'F');

INSERT INTO DETALLE_PROYECTO(ID_DP,ID_PER, ID_PROY)
VALUES (1,2,1),
       (2,1,1);

INSERT INTO  PROYECTO(NOMBRE_PROY, TIPO_PROY)
VALUES('ANIMALES','BIOLOGIA'),
      ('PREHISTORIA','ANTROPOLOGIA');
```

10.Crear una función que sume los valores de la serie Fibonacci

```
CREATE OR REPLACE FUNCTION SERIE_FIBONANCI(NUMBER INTEGER)

RETURNS TEXT
BEGIN
    DECLARE A INTEGER DEFAULT 0;
    DECLARE B INTEGER DEFAULT 1;
    DECLARE AUX INTEGER DEFAULT 0;
    DECLARE CONTADOR INTEGER DEFAULT 0;
    DECLARE CADENA TEXT DEFAULT '';
    SET CADENA =CONCAT(A,',',B);

IF NUMBER=1 THEN SET CADENA='0';
ELSEIF NUMBER=2 THEN SET CADENA='0,1';
ELSEIF NUMBER<=0 THEN SET CADENA='EL NUMERO DEBE SER MAYOR A CERO';
ELSE
            REPEAT

            SET AUX=A+B;
            SET CADENA = CONCAT(CADENA,',',AUX);
            SET A=B;
            SET B=AUX;
            SET CONTADOR=CONTADOR+1;
            UNTIL CONTADOR = NUMBER-2 END REPEAT;
```

```
END IF;
    RETURN CADENA;
END;

CREATE OR REPLACE FUNCTION CONTAR_FIBONANCI(SERIE TEXT)
RETURNS INTEGER
BEGIN
    DECLARE SUMA INTEGER DEFAULT 0;
    DECLARE CONT INTEGER DEFAULT 1;
    DECLARE FINAL INTEGER DEFAULT CHAR_LENGTH(SERIE);

  REPEAT
   SET SUMA =SUMA + SUBSTRING(SERIE,CONT,1);
   SET CONT=CONT+2;
  until  CONT> FINAL END REPEAT;

    RETURN SUMA;
end;

SELECT SERIE_FIBONANCI( NUMBER: 5);
SELECT CONTAR_FIBONANCI( SERIE: SERIE_FIBONANCI( NUMBER: 5));
```

11.Manejo de vistas.

```sql
CREATE OR REPLACE VIEW BUSQUEDA AS
SELECT CONCAT(PERSONA.NOMBRE,' ',PERSONA.APELLIDOS) AS NOMBRES_Y_APELLIDOS,
          PERSONA.EDAD AS EDAD,
          PERSONA.FECHA_NAC AS FECHA_DE_NACIMIENTO,
        PROYECTO.NOMBRE_PROY AS NOMBRE_DEL_PROYECTO
FROM PERSONA
INNER JOIN DEPARTAMENTO ON PERSONA.ID_DEP = DEPARTAMENTO.ID_DEP
INNER JOIN DETALLE_PROYECTO ON PERSONA.ID_PER = DETALLE_PROYECTO.ID_PER
INNER JOIN PROYECTO ON DETALLE_PROYECTO.ID_PROY = PROYECTO.ID_PROY

WHERE PERSONA.GENERO='F' AND  DEPARTAMENTO.NOMBRE='EL ALTO' AND PERSONA.FECHA_NAC='2000-10-10' ;

SELECT * FROM BUSQUEDA;
```

```sql
INSERT INTO DEPARTAMENTO(NOMBRE)
VALUES('EL ALTO');

INSERT INTO PERSONA (NOMBRE, APELLIDOS, FECHA_NAC, EDAD, EMAIL, ID_DEP, ID_PROV, GENERO)
VALUES ('CARMEN','CALLE UGARTE','2000-10-10',23,'CARMEN@GMAIL.COM',10,1,'F'),
       ('GABRIELA','BARRA MENDOZA','2000-10-10',23,'GABRIELA@GMAIL.COM',10,1,'F');
INSERT INTO DETALLE_PROYECTO(ID_PER, ID_PROY)
VALUES (6,4),
       (7,5);
```

	NOMBRES_Y_APELLIDOS	EDAD	FECHA_DE_NACIMIENTO	NOMBRE_DEL_PROYECTO
1	CARMEN CALLE UGARTE	23	2000-10-10	REDES Y SISTEMAS
2	GABRIELA BARRA MENDOZA	23	2000-10-10	FLUJO MAGNETICO

12.Manejo de TRIGGERS

```
ALTER TABLE PROYECTO ADD (ESTADO VARCHAR(30));

INSERT INTO PROYECTO(NOMBRE_PROY, TIPO_PROY)
VALUES ('EDUCACION PERSONAS ESPECIALES','EDUCACION'),
       ('PLANTACION DE ARBOLES','FORESTACION'),
       ('LOS AZTECAS','CULTURA');

CREATE OR REPLACE TRIGGER UPDATE_TIP_PROYCT
BEFORE UPDATE ON PROYECTO
FOR EACH ROW
    BEGIN
        IF  NEW.TIPO_PROY='EDUCACION'OR  NEW.TIPO_PROY ='FORESTACION' OR  NEW.TIPO_PROY= 'CULTURA'
            THEN SET NEW.ESTADO='ACTIVO';
        ELSE
            SET NEW.ESTADO='INACTIVO';
        END IF;
    END;

CREATE OR REPLACE TRIGGER ADD_TIP_PROYCT
BEFORE  INSERT ON PROYECTO
FOR EACH ROW
    BEGIN
```

```
        IF  NEW.TIPO_PROY='EDUCACION'OR  NEW.TIPO_PROY ='FORESTACION' OR  NEW.TIPO_PROY= 'CULTURA'
            THEN SET NEW.ESTADO='ACTIVO';
        ELSE
            SET NEW.ESTADO='INACTIVO';
        END IF;
    end;

INSERT INTO PROYECTO(NOMBRE_PROY, TIPO_PROY)
VALUES ('ARBOLESS','EDUCACION');

SELECT *
FROM PROYECTO
```

	ID_PROY	NOMBRE_PROY	TIPO_PROY	ESTADO
1	1	ANIMALES	BIOLOGIA	<null>
2	2	PREHISTORIA	ANTROPOLOGIA	<null>
3	3	MICROBIOS	BIOLOGIA	<null>
4	4	REDES Y SISTEMAS	TECNOLOGIA	<null>
5	5	FLUJO MAGNETICO	FISICA	<null>
6	6	EDUCACION PERSONAS ESPECIALES	EDUCACION	<null>
7	7	PLANTACION DE ARBOLES	FORESTACION	<null>
8	8	LOS AZTECAS	CULTURA	<null>
9	9	ARBOLESS	EDUCACION	ACTIVO
10	10	ARBOLESS	EDUCACION	ACTIVO
11	11	EDUCACION PERSONAS ESPECIALES	EDUCACION	ACTIVO
12	12	PLANTACION DE ARBOLES	FORESTACION	ACTIVO
13	13	LOS AZTECAS	CULTURA	ACTIVO

13.Manejo de Triggers II

```
CREATE OR REPLACE TRIGGER AGREGAR_EDAD
BEFORE INSERT ON PERSONA
FOR EACH ROW
    BEGIN
        SET NEW.EDAD= TIMESTAMPDIFF(YEAR,NEW.FECHA_NAC,CURDATE());
    end;

INSERT INTO  PERSONA(NOMBRE, APELLIDOS, FECHA_NAC, EMAIL, ID_DEP, ID_PROV, GENERO)
VALUES  ('MARIA','GALARSA ORTEGA','1992-12-15','MARIA@GMAIL.COM',3,3,'F');

SELECT *FROM PERSONA;
```

	ID_PER	NOMBRE	APELLIDOS	FECHA_NAC	EDAD	EMAIL	ID_DEP
1	1	RODRIGO	MENODZA UGARTE	1999-11-21	23	RODRIGO@GMAIL.COM	1
2	2	MARIA	LAURA TORREZ	1999-12-16	25	MARIA@GAMIL.COM	2
3	3	AUGUSTO	MEDRANO LOZA	1998-09-12	24	AUGUSTO@GMAIL.COM	3
4	4	MARIANO	FERNANDEZ GUTIERREZ	1995-09-12	27	MARIANO@GMAIL.COM	4
5	5	LORENA	ZAMUDIO LLANOS	2000-09-12	22	LORENA@GMAIL.COM	5
6	6	CARMEN	CALLE UGARTE	2000-10-10	23	CARMEN@GMAIL.COM	10
7	7	GABRIELA	BARRA MENDOZA	2000-10-10	23	GABRIELA@GMAIL.COM	10
8	8	MARIA	GALARSA ORTEGA	1992-12-15	29	MARIA@GMAIL.COM	3

14.Manejo de TRIGGERS III

```sql
CREATE TABLE COPIA_PERSONA
(
  NOMBRE VARCHAR(20),
  APELLIDOS VARCHAR(50),
  FECHA_NAC DATE,
  EDAD INT,
  EMAIL VARCHAR(50),
  ID_DEP INT NOT NULL ,
  ID_PROV INT NOT NULL,
  GENERO VARCHAR(1),
  FOREIGN KEY (ID_PROV) REFERENCES  PROVINCIA(ID_PROV),
  FOREIGN KEY (ID_DEP) REFERENCES DEPARTAMENTO(ID_DEP)
);

CREATE OR REPLACE TRIGGER COPIAR_PERSONA
BEFORE INSERT ON PERSONA
FOR EACH ROW
    BEGIN
        INSERT INTO COPIA_PERSONA(NOMBRE, APELLIDOS, FECHA_NAC, EDAD, EMAIL, ID_DEP, ID_PROV, GENERO)
        VALUES(NEW.NOMBRE,NEW.APELLIDOS,NEW.FECHA_NAC,NEW.EDAD,NEW.EMAIL,NEW.ID_DEP,NEW.ID_PROV,NEW.GENERO);
    end;

INSERT INTO PERSONA(NOMBRE, APELLIDOS, FECHA_NAC, EDAD, EMAIL, ID_DEP, ID_PROV, GENERO)
VALUES ('PDRO','ALIAGA ORTEGA','2000-12-15',22,'ALEJANDRA@GMAIL.COM',5,5,'M');
```

```
SELECT*FROM COPIA_PERSONA;
```

	NOMBRE	APELLIDOS	FECHA_NAC	EDAD	EMAIL	ID_DEP	ID_PROV	GENER
1	PDRO	ALIAGA ORTEGA	2000-12-15	21	ALEJANDRA@GMAIL.COM	5	5	M

EDAD	EMAIL	ID_DEP	ID_PROV	GENERO
21	ALEJANDRA@GMAIL.COM	5	5	M

15.Crear una consulta SQL que haga uso de todas las tablas.

```sql
CREATE OR REPLACE VIEW TODAS_LAS_TABLAS AS
SELECT
    CONCAT(PERSONA.NOMBRE,PERSONA.APELLIDOS) AS NOMBRE_Y_APELLIDOS,
    PERSONA.EDAD AS EDAD,
    DEPARTAMENTO.NOMBRE AS DEPARTAMENTO,
    PROVINCIA.NOMBRE AS PROVINCIA,
    CONCAT(PROYECTO.NOMBRE_PROY,': ',TIPO_PROY) AS PROYECTO

FROM PERSONA
INNER JOIN DEPARTAMENTO ON PERSONA.ID_DEP = DEPARTAMENTO.ID_DEP
INNER JOIN PROVINCIA ON PERSONA.ID_PROV = PROVINCIA.ID_PROV
INNER JOIN DETALLE_PROYECTO ON PERSONA.ID_PER = DETALLE_PROYECTO.ID_PER
INNER JOIN PROYECTO ON DETALLE_PROYECTO.ID_PROY = PROYECTO.ID_PROY;

SELECT * FROM (TODAS_LAS_TABLAS);
```

	NOMBRE_Y_APELLIDOS	EDAD	DEPARTAMENTO	PROVINCIA	PROYECTO
1	RODRIGOMENODZA UGARTE	23	LA PAZ	VIACHA	PREHISTORIA: ANTROPOLOGIA
2	MARIALAURA TORREZ	25	SANTA CRUZ	ROBORE	ANIMALES: BIOLOGIA
3	AUGUSTOMEDRANO LOZA	24	BENI	MAGDALENA	REDES Y SISTEMAS: TECNOLOGIA
4	MARIANOFERNANDEZ GUTIERREZ	27	ORURO	CHALLAPATA	MICROBIOS: BIOLOGIA
5	CARMENCALLE UGARTE	23	EL ALTO	VIACHA	REDES Y SISTEMAS: TECNOLOGIA
6	GABRIELABARRA MENDOZA	23	EL ALTO	VIACHA	FLUJO MAGNETICO: FISICA

GRACIAS!!!