TAREA HITO 4 BASE DE DATOS

Estudiante: Aaron Álvaro Huanca Salazar

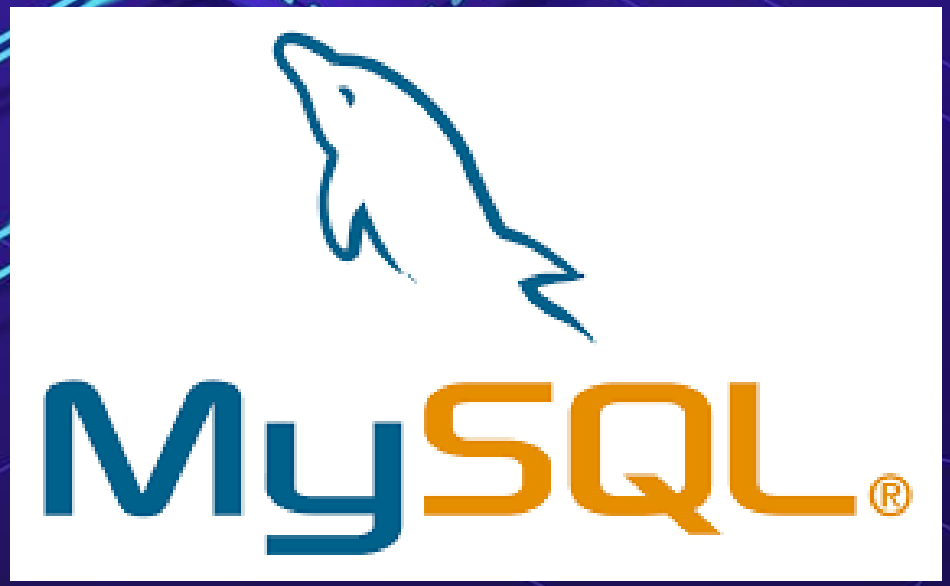
Carrera: Ingeniería de sistemas

Docente: Ing. William Roddy Barra

Gestión: 2022

1. Defina que es lenguaje procedural en MySQL

El usuario da órdenes para que se realicen las tareas pertinentes con el objetico de recuperarlos datos requeridos. Es la base del lenguaje de consulta SQL.



2. Defina que es una función 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.

Lógica para cada una:

```
CREATE PROCEDURE
HelloWorldprocedure
AS
PRINT 'Hello World'
```

```
CREATE FUNCTION
dbo.helloworldfunction()
RETURNS varchar(20)
AS
BEGIN
   RETURN 'Hello world'
END
```

5. Defina que es una TRIGGER en MySQL.

El trigger MySQL es un objeto de la base de datos que está asociado con una tabla. Se activará cuando una acción definida se ejecute en la tabla. El trigger puede usarse para ejecutar una de las siguientes sentencias MySQL en la tabla: INSERT, UPDATE y DELETE. Se puede invocar antes o después del evento.

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

Cuando trabajamos con trigger a nivel de fila, Oracle provee de dos tablas temporales a las cuales se puede acceder, que contienen los antiguos y nuevos valores de los campos del registro afectado por la sentencia que disparó el trigger. El nuevo valor es ":new" y el viejo valor es ":old". Para referirnos a ellos debemos especificar su campo separado por un punto ":new.CAMPO" y ":old.CAMPO".

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

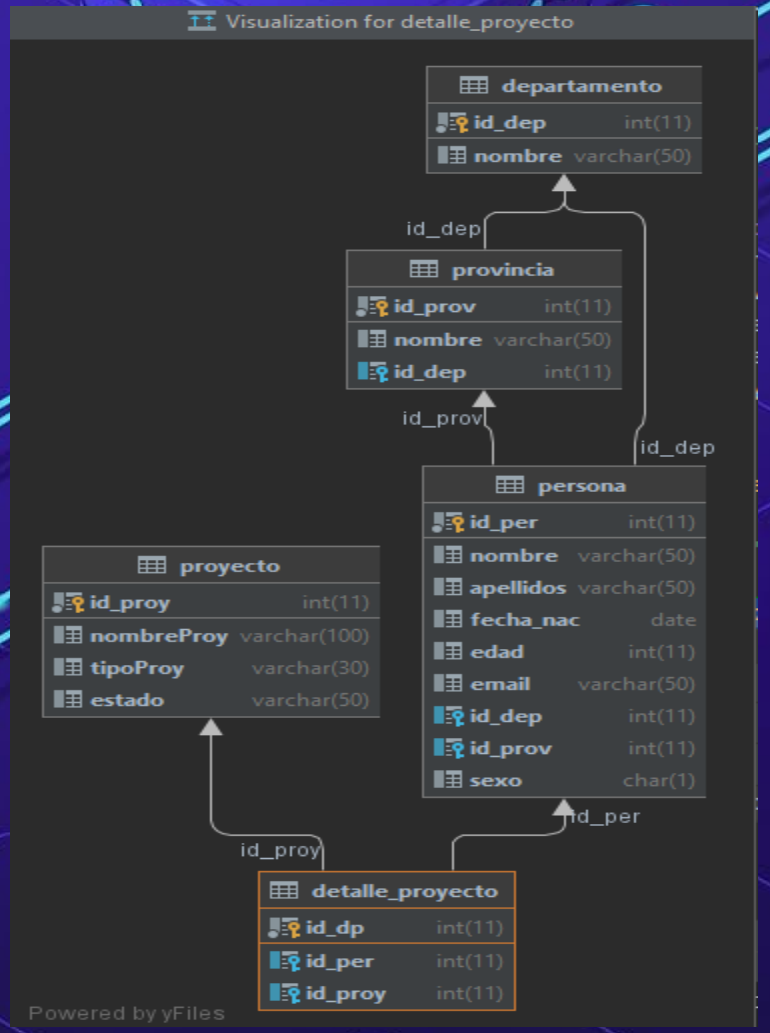
Puede ser **BEFORE** (antes) o **AFTER** (despues), para indicar que el disparador se ejecute antes o después que la sentencia que lo activa.

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

Un "trigger" (disparador o desencadenador) es un bloque de código que se ejecuta automáticamente cuando ocurre algún evento (como inserción, actualización o borrado) sobre una determinada tabla (o vista); es decir, cuando se intenta modificar los datos de una tabla (o vista) asociada al disparador.

Parte practica

9. Crear la siguiente Base de datos y sus registros



Código:

```sql
CREATE TABLE departamento
(
   id_dep INTEGER (11) AUTO_INCREMENT PRIMARY KEY NOT NULL,
   nombre VARCHAR(50)
);
CREATE TABLE proyecto
(
   id_proy INTEGER (11) AUTO_INCREMENT PRIMARY KEY NOT NULL,
   nombreProy VARCHAR(100),
   tipoProy VARCHAR (30),
   estado VARCHAR(30)
);
CREATE TABLE provincia(
   id_prov int(11) AUTO_INCREMENT PRIMARY KEY,
   nombre VARCHAR(50),
   id_dep int(11),
   FOREIGN KEY (id_dep) REFERENCES departamento (id_dep)
);
CREATE TABLE persona
(
   id_per INTEGER (11) AUTO_INCREMENT PRIMARY KEY NOT NULL,
   nombre VARCHAR(50),
   apellidos VARCHAR(50),
   fecha_nac date,
   edad INTEGER(11),
   email VARCHAR(50),
   id_dep int(11),
   id_prov int(11),
   sexo CHAR(1),
   FOREIGN KEY (id_prov) REFERENCES provincia (id_prov),
   FOREIGN KEY (id_dep) REFERENCES departamento (id_dep)
);
CREATE TABLE detalle_proyecto
(
   id_dp INTEGER (11) AUTO_INCREMENT PRIMARY KEY NOT NULL,
   id_per int(11),
   id_proy INTEGER(11),
   FOREIGN KEY (id_proy) REFERENCES proyecto (id_proy),
   FOREIGN KEY (id_per) REFERENCES persona (id_per)
);
```

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

Código:

```
drop function if exists SerieFibonnaci;
create function SerieFibonnaci(limiti integer) returns text
begin
   declare respuesta text default '';
   declare x integer default 0;
   declare y integer default 1;
   declare cont integer default 0;

   while cont!= limiti do
      set respuesta= concat(respuesta,x,', ');
      set x=x+y;
      set y=x-y;
      set cont=cont+1;
   end while;

   return respuesta;
end;
Select SerieFibonnaci(10);

drop function if exists SUMA_SerieFibonnaci;
create function SUMA_SerieFibonnaci(Limita integer)
returns text
   begin
      declare entrada text default '';
      declare espacio text default ' ';
      declare x int default 1;
      declare nVeces int default 0;
      declare letra char default '';
      declare limite int default 0;
      declare a int default 0;
      declare b int default 1;
      declare cont int default 0;
      declare aux int default 0;
      declare sumar int default 0;

      set entrada = SerieFibonnaci(limita);
      set limite = char_length(entrada);

      while x <= limite do
       set letra = substring(entrada, x, 1);
       if letra = espacio
         then
          set nVeces = nVeces + 1;
       end if;
       set x = x + 1;
      end while;
      while cont < nVeces do
            if cont = 0
               then
                  set sumar = 0;
               else
                  set sumar = sumar + b;
                  set aux = a;
                  set a = b;
                  set b = aux + a;
            end if;
            set cont = cont + 1;
      end while;
      return sumar;
   end;

Select SUMA_SerieFibonnaci(10);
```

11. Manejo de vistas.

Código:

```
create view nueva_vista as
    select concat(per.nombre, ' ', per.apellidos) as persona,
           per.edad as edad,per.fecha_nac as nacimiento_la_fecha
,proy.nombreProy as proyecto
    from departamento as depa
    inner join provincia as pro on depa.id_dep = pro.id_dep
    inner join persona as per on depa.id_dep = per.id_dep
    inner join detalle_proyecto as dep on per.id_per = dep.id_per
    inner join proyecto as proy on dep.id_proy = proy.id_proy
    where per.sexo = 'Femenino' AND depa.nombre = 'ElAlto' AND
per.fecha_nac = '2000-10-10';
```

12. .Manejo de TRIGGERS I.

Código:

```
create trigger tipoproy
    before update on proyecto
    for each row
    begin
        if new.tipoProy = 'Educacion' or new.tipoProy = 'Forestacion' or new.tipoProy = 'Cultura' then
            set new.estado = 'Activo';
        else
            set new.estado = 'Inactivo';
        end if;
    end;

update proyecto
set tipoProy = 'Educacion'
where id_proy = 1;
```

13. Manejo de Triggers II.

Código:

```
create trigger calcula_edad_para_persona
   before insert on persona
   for each row
begin
   declare edad_calc integer;
   set edad_calc=timestampdiff(year, new.fecha_nac,curdate());

   set new.edad=edad_calc;
end;

INSERT INTO persona (nombre,fecha_nac) VALUES ('Martin','1990-02-15');
```

14. Manejo de TRIGGERS III.

Código:

```
CREATE TABLE persona1(
   nombre VARCHAR(50),
   apellidos VARCHAR(50),
   fecha_nac date,
   edad INTEGER(11),
   email VARCHAR(50),
   sexo CHAR(1)
);

create trigger copeear
  before insert on persona
  for each row
begin
  insert into persona1(nombre, apellidos, fecha_nac, edad, email,sexo) VALUES
      (new.nombre,new.apellidos,new.fecha_nac,new.edad,new.email,new.sexo);
end;

insert into persona(id_per,nombre, apellidos, fecha_nac, edad, email,sexo)
VALUES(9,'Mdgfa','Maamanti','2010-10-11',8,'aarddfdondgmail.com','F');
```

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

Código:

```
select per.nombre, per.apellidos,depa.nombre,
pro.nombre,proy.nombreProy,proy.tipoProy
    from departamento as depa
    inner join provincia as pro on depa.id_dep = pro.id_dep
    inner join persona as per on depa.id_dep = per.id_dep
    inner join detalle_proyecto as dep on per.id_per = dep.id_per
    inner join proyecto as proy on dep.id_proy = proy.id_proy
```


GRACIAS POR LA
REVISIÓN