

UNIVERSIDAD PRIVADA FRANZ TAMAYO

EVALUACIÓN PROCESUAL - HITO 4



ESTUDIANTE:

DERECK DANNER HECTOR FERNANDEZ
MENDOZA

ASIGNATURA:

BASE DE DATOS II

CARRERA:

INGENIERÍA DE SISTEMAS

PARALELO:

BDA - 312

DOCENTE:

WILLIAM RODDY BARRA PAREDES

FECHA:

30/11/2022

GITHUB:

<https://github.com/DereckFM>

Manejo de conceptos.

1. Defina que es lenguaje procedural en MySQL.

Son las instrucciones escritas por el usuario para realizar una determinada tarea.

2. Defina que es una FUNCTION en MySQL.

Son el código escrito por el usuario que reciben datos y realizan operaciones con ellos para luego devolver un resultado.

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

Las funciones solo retornan un valor individual, no un conjunto de registros mientras que un procedimiento puede tomar varios argumentos de entrada y mostrar valores como resultados.

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

Se ejecutan con los comandos SELECT y EXEC respectivamente.

5. Defina que es una TRIGGER en MySQL.

Es un desencadenador que ejecuta una serie de instrucciones cuando se producen ciertos eventos sobre la tabla a la que el trigger está ligado. Los eventos son: INSERT, UPDATE y DELETE.

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

Estas variables permiten acceder a los registros antiguos y nuevos de las columnas de las tablas.

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

Son instrucciones que hacen que el trigger se ejecute antes o después de que se registre un evento en la tabla a la que está asociado.

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

Como se mencionó anteriormente, los eventos son aquellos que afectan el contenido de la tabla y esto son:

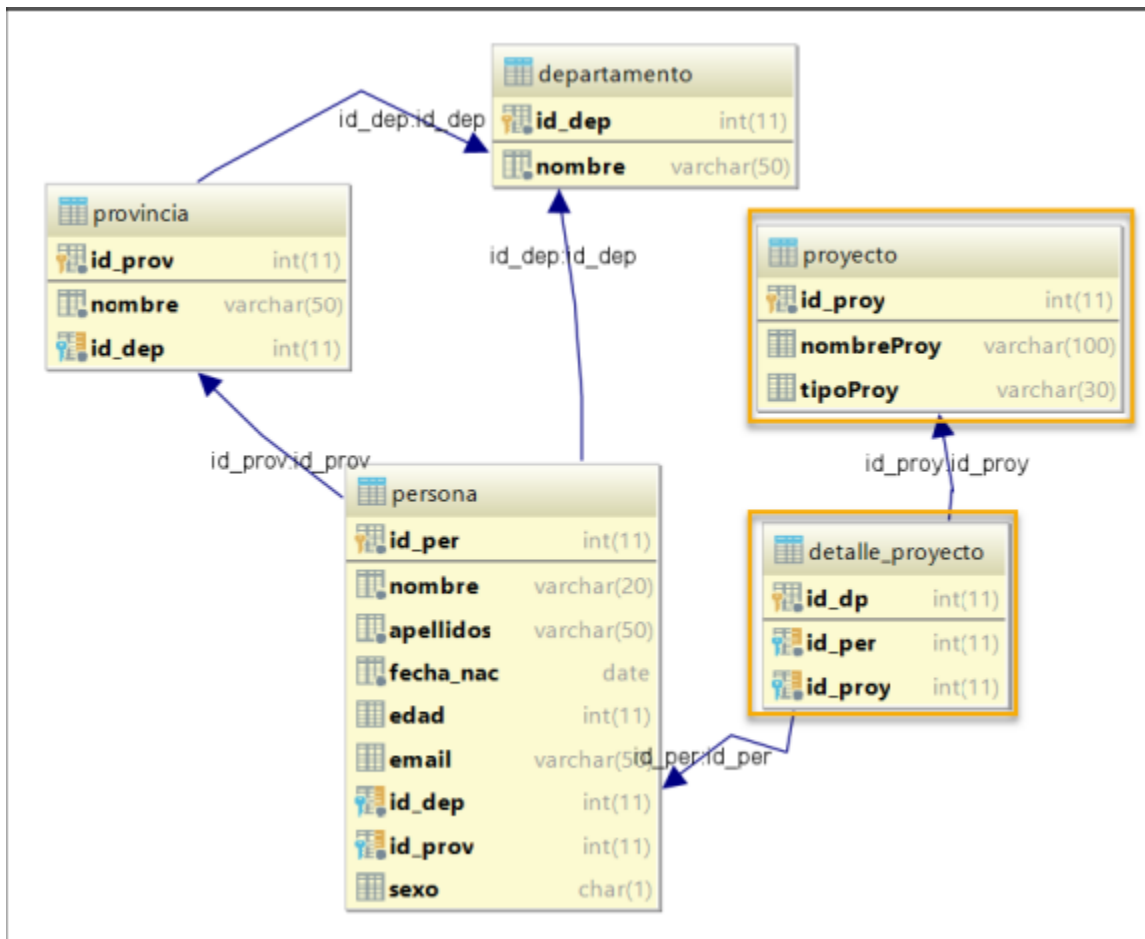
INSERT: Insertar.

UPDATE: Actualizar.

DELETE: Borrar.

Parte práctica

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



```
create database EvaluacionHito4;
```

```
use EvaluacionHito4;
```

```
create table departamento
```

```
(  
    id_dep int primary key auto_increment not null,  
    nombre varchar(50) not null  
);
```

```
insert into departamento(nombre) values
```

```
('La Paz'),  
('Cochabamba');
```

```
create table proyecto
```

```
(  
    id_proyecto int primary key auto_increment not null,  
    nombre_Proj varchar (100) not null,  
    tipo_Proj varchar(30) not null  
);
```

```
insert into proyecto(nombre_Proy, tipo_Proy) values
('Proyecto1','Tipo1'),
('Proyecto2','Tipo2');
```

```
create table provincia
(
    id_prov int primary key auto_increment not null,
    nombre varchar(30),
    id_dep int not null,
    foreign key (id_dep) references departamento(id_dep)
);
```

```
insert into provincia(nombre, id_dep) values
('Provincia1', 1),
('Provincia2', 2);
```

```
create table persona
(
    id_per int primary key auto_increment not null,
    nombre varchar(20) not null,
    apellido varchar(50) not null,
    fecha_nac date not null,
    edad varchar(11) not null,
    email varchar(50) not null,
    id_dep int not null,
    id_prov int not null,
    sexo char(1) not null,
    foreign key (id_dep) references departamento(id_dep),
    foreign key (id_prov) references provincia(id_prov)
);
```

```
insert into persona(nombre, apellido, fecha_nac, edad, email, id_dep, id_prov, sexo) values
('Persona1','Apellido1','2001-11-13','21','Persona1@gmail.com',1,1,'M'),
('Persona2','Apellido2','2003-08-17','19','Persona2@gmail.com',2,2,'F');
```

```
create table detalle_proyecto
(
    id_dp int primary key auto_increment not null,
    id_per int not null,
    id_proy int not null,
    foreign key (id_per) references persona(id_per),
    foreign key (id_proy) references proyecto(id_proyecto)
);
```

```
insert into detalle_proyecto(id_per, id_proy) values
(1,1),
(2,2);
```

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

- El objetivo es sumar todos los números de la serie fibonacci desde una cadena.
- Es decir usted tendrá solo la cadena generada con los primeros N números de la serie fibonacci y a partir de ellos deberá sumar los números de esa serie.

○ Ejemplo: `suma_serie_fibonacci(mi_metodo_que_retorna_la_serie(10))`

■ Note que previamente deberá crear una función que retorne una cadena con la serie fibonacci hasta un cierto valor.

1. Ejemplo: 0,1,1,2,3,5,8,.....

■ Luego esta función se deberá pasar como parámetro a la función que suma todos los valores de esa serie generada.



- Adjuntar el código SQL generado y una imagen de su correcto funcionamiento.

```
create or replace function fibonacci(numero int)
returns text
BEGIN
declare A int default 0;
declare B int default 1;
declare aux int default 0;
declare count int default 0;
declare chain text default '';
set chain = CONCAT(A, ',', B);

if numero = 1 THEN
set chain = '0';
elseif numero = 2 THEN
set chain = '0,1';
elseif numero <= 0 THEN
set chain = 'EL NUMERO DEBE SER MAYOR A CERO';
else
REPEAT

set AUX = A + B;
set chain = CONCAT(chain, ',', AUX);
set A = B;
set B = AUX;
set count = count + 1;
UNTIL count = numero - 2 END REPEAT;
```

```

    END IF;
    RETURN chain;
END;

CREATE OR REPLACE FUNCTION sumifibonacci(serie TEXT)
    RETURNS INTEGER
BEGIN
    DECLARE suma INTEGER DEFAULT 0;
    DECLARE contador INTEGER DEFAULT 1;
    DECLARE fin INTEGER DEFAULT CHAR_LENGTH(serie);

    REPEAT
        SET suma = suma + SUBSTRING(serie, contador, 1);
        SET contador = contador + 2;
    until contador > fin END REPEAT;

    RETURN suma;
end;

SELECT fibonacci(6);
SELECT sumifibonacci(fibonacci(6));

```

```

SELECT fibonacci(6);
SELECT sumifibonacci(fibonacci(6));

```

fibonacci(6):int(11)
0,1,1,2,3,5

```

SELECT fibonacci(6);
SELECT sumifibonacci(fibonacci(6));

```

sumifibonacci(fibonacci(6)):int(11)
12

11. Manejo de vistas.

○ Crear una consulta SQL para lo siguiente.

■ La consulta de la vista debe reflejar como campos:

1. nombres y apellidos concatenados

2. la edad

3. fecha de nacimiento.

4. Nombre del proyecto

○ Obtener todas las personas del sexo femenino que hayan nacido en el departamento de El Alto en donde la fecha de nacimiento sea: 1. fecha_nac = '2000-10-10'

create or replace view Busqueda as

```
SELECT CONCAT(P.NOMBRE,' ',P.APELLIDO) AS NOMBRES_Y_APELLIDOS,
```

```
       P.EDAD AS EDAD,
```

```
       P.FECHA_NAC AS FECHA_DE_NACIMIENTO,
```

```
       PROYECTO.NOMBRE_PROY AS NOMBRE_DEL_PROYECTO
```

```
FROM PERSONA as P
```

```
INNER JOIN DEPARTAMENTO ON P.ID_DEP = DEPARTAMENTO.ID_DEP
```

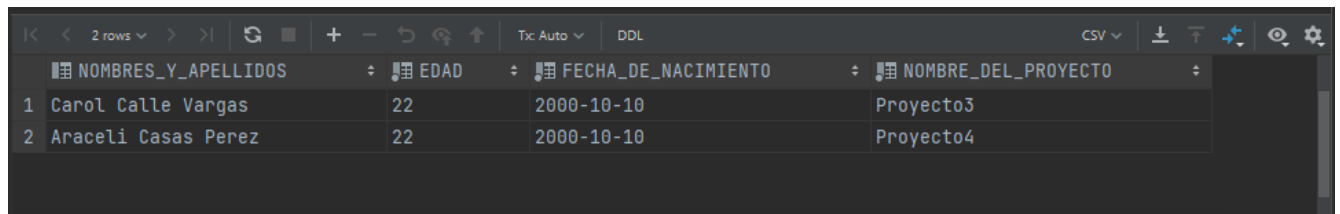
```
INNER JOIN DETALLE_PROYECTO ON P.ID_PER = DETALLE_PROYECTO.ID_PER
```

```
INNER JOIN PROYECTO ON DETALLE_PROYECTO.ID_PROY = PROYECTO.ID_PROYECTO
```

```
WHERE P.SEXO='F' and DEPARTAMENTO.NOMBRE='EL ALTO' and P.FECHA_NAC='2000-10-10' ;
```

```
SELECT *
```

```
FROM Busqueda;
```



The screenshot shows a database interface with a query result table. The table has four columns: NOMBRES_Y_APELLIDOS, EDAD, FECHA_DE_NACIMIENTO, and NOMBRE_DEL_PROYECTO. There are two rows of data displayed.

	NOMBRES_Y_APELLIDOS	EDAD	FECHA_DE_NACIMIENTO	NOMBRE_DEL_PROYECTO
1	Carol Calle Vargas	22	2000-10-10	Proyecto3
2	Araceli Casas Perez	22	2000-10-10	Proyecto4

12. Manejo de TRIGGERS I.

- Crear TRIGGERS Before or After para INSERT y UPDATE aplicado a la tabla PROYECTO
- Debera de crear 2 triggers minimamente.
- Agregar un nuevo campo a la tabla PROYECTO.
- El campo debe llamarse ESTADO 6
- Actualmente solo se tiene habilitados ciertos tipos de proyectos.
- EDUCACION, FORESTACION y CULTURA
- Si al hacer insert o update en el campo tipoProy llega los valores EDUCACION, FORESTACIÓN o CULTURA, en el campo ESTADO colocar el valor ACTIVO. Sin embargo si se llega a un tipo de proyecto distinto colocar INACTIVO
- Adjuntar el código SQL generado y una imagen de su correcto funcionamiento.

```
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_PROY
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_PROY
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');
```


	id_proyecto	nombre_Proj	tipo_Proj	ESTADO
1	1	Proyecto1	Tipo1	<null>
2	2	Proyecto2	Tipo2	<null>
3	3	Proyecto3	Tipo3	<null>
4	4	Proyecto4	Tipo4	<null>
5	5	EDUCACION PERSONAS ESPECIALES	EDUCACION	<null>
6	6	PLANTACION DE ARBOLES	FORESTACION	<null>
7	7	LOS AZTECAS	CULTURA	<null>
8	8	ARBOLESS	EDUCACION	ACTIVO

13. Manejo de Triggers II.

- El trigger debe de llamarse calculaEdad.
- El evento debe de ejecutarse en un BEFORE INSERT.
- Cada vez que se inserta un registro en la tabla PERSONA, el trigger debe de calcular la edad en función a la fecha de nacimiento.
- Adjuntar el código SQL generado y una imagen de su correcto funcionamiento.

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

```
INSERT INTO persona(NOMBRE, APELLIDO, FECHA_NAC, EMAIL, ID_DEP, ID_PROV, SEXO)
VALUES ('Maria','Galarza Ortega','1992-12-15','Maria@gmail.com',2,2,'F');
```

El INSERT no lleva el parámetro edad incluido

	id_per	nombre	apellido	fecha_nac	edad	email	id_dep	id_prov
1	1	Persona1	Apellido1	2001-11-13	21	Persona1@gmail.com	1	1
2	2	Persona2	Apellido2	2003-08-17	19	Persona2@gmail.com	2	2
3	3	Carol	Calle Vargas	2000-10-10	22	Carol@gmail.COM	3	3
4	4	Araceli	Casas Perez	2000-10-10	22	Araceli@gmail.COM	3	4
5	5	Maria	Galarza Ortega	1992-12-15	29	Maria@gmail.com	2	2

14. Manejo de TRIGGERS III.

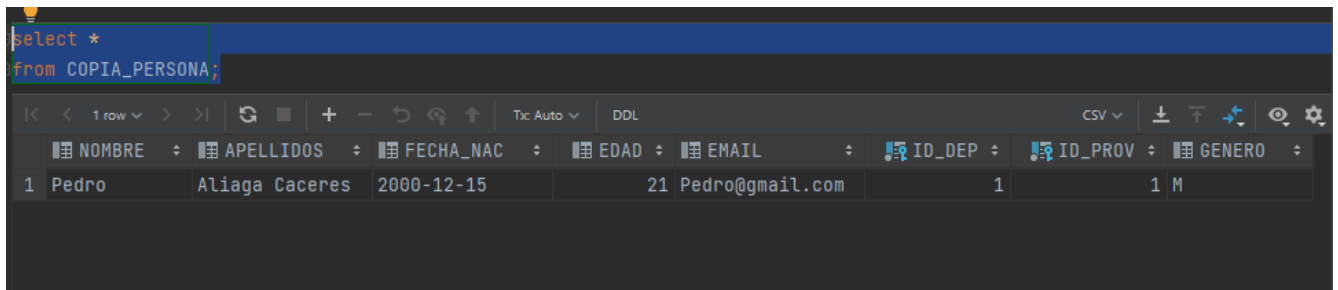
- Crear otra tabla con los mismos campos de la tabla persona (Excepto el primary key id_per).
- No es necesario que tenga PRIMARY KEY.
- Cada vez que se haga un INSERT a la tabla persona estos mismos valores deben insertarse a la tabla copia.
- Para resolver esto deberá de crear un trigger before insert para la tabla PERSONA.
- Adjuntar el código SQL generado y una imagen de su correcto funcionamiento.

```
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,
    GÉNERO 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.APELLIDO,NEW.FECHA_NAC,NEW.EDAD,NEW.EMAIL,NEW.ID_DEP,NEW.ID
_PROV,NEW.SEXO);
end;
```

```
INSERT INTO PERSONA(NOMBRE, APELLIDO, FECHA_NAC, EDAD, EMAIL, ID_DEP, ID_PROV, SEXO)
VALUES ('Pedro','Aliaga Caceres','2000-12-15',22,'Pedro@gmail.com',1,1,'M');
```



	NOMBRE	APELLIDOS	FECHA_NAC	EDAD	EMAIL	ID_DEP	ID_PROV	GENERO
1	Pedro	Aliaga Caceres	2000-12-15	22	Pedro@gmail.com	1	1	M

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

- La consulta generada convertirlo a VISTA

```
CREATE OR REPLACE VIEW TODAS_LAS_TABLAS AS
SELECT CONCAT(P.NOMBRE, ' ', P.APELLIDO) AS NOMBRE_Y_APELLIDOS,
       P.EDAD AS EDAD,
       D.NOMBRE AS DEPARTAMENTO,
       PR.NOMBRE AS PROVINCIA,
       CONCAT(PROY.NOMBRE_PROY, ': ', TIPO_PROY) AS PROYECTO
FROM PERSONA as P
     INNER JOIN DEPARTAMENTO as D ON P.ID_DEP = D.ID_DEP
     INNER JOIN PROVINCIA as PR ON P.ID_PROV = PR.ID_PROV
     INNER JOIN DETALLE_PROYECTO as DP ON P.ID_PER = DP.ID_PER
     INNER JOIN PROYECTO as PROY ON DP.ID_PROY = PROY.ID_PROYECTO;

SELECT * FROM (TODAS_LAS_TABLAS);
```



	NOMBRE_Y_APELLIDOS	EDAD	DEPARTAMENTO	PROVINCIA	PROYECTO
1	Persona1 Apellido1	21	La Paz	Provincia1	Proyecto1: Tipo1
2	Persona1 Apellido1	21	La Paz	Provincia2	Proyecto1: Tipo1
3	Persona1 Apellido1	21	La Paz	Provincia3	Proyecto1: Tipo1
4	Persona1 Apellido1	21	La Paz	Provincia4	Proyecto1: Tipo1
5	Persona2 Apellido2	19	Cochabamba	Provincia1	Proyecto2: Tipo2
6	Persona2 Apellido2	19	Cochabamba	Provincia2	Proyecto2: Tipo2
7	Persona2 Apellido2	19	Cochabamba	Provincia3	Proyecto2: Tipo2
8	Persona2 Apellido2	19	Cochabamba	Provincia4	Proyecto2: Tipo2
9	Carol Calle Vargas	22	El Alto	Provincia1	Proyecto3: Tipo3
10	Carol Calle Vargas	22	El Alto	Provincia2	Proyecto3: Tipo3
11	Carol Calle Vargas	22	El Alto	Provincia3	Proyecto3: Tipo3
12	Carol Calle Vargas	22	El Alto	Provincia4	Proyecto3: Tipo3
13	Araceli Casas Perez	22	El Alto	Provincia1	Proyecto4: Tipo4
14	Araceli Casas Perez	22	El Alto	Provincia2	Proyecto4: Tipo4
15	Araceli Casas Perez	22	El Alto	Provincia3	Proyecto4: Tipo4
16	Araceli Casas Perez	22	El Alto	Provincia4	Proyecto4: Tipo4