Facilitador(a): Ing. Carmen Ortega Asignatura: Base de Datos II

Fecha: 30/10/2020 Grupo: \_\_\_1IL131

Estudiantes: Jacky He, 3-740-114

Yui Lo, 8-929-854

Maria José Martínez PE-15-1146

Dhiraj Lakhwani 3-743-1798

Richie Ieong 8-970-668

1. **TÍTULO** **DE LA EXPERIENCIA**: APLIQUEMOS CONCEPTOS DEL CAPITULO III – INTEGRIDAD DE LA BD
2. **TEMAS:**

CAPITULO III – INTEGRIDAD DE LA BD

1. **OBJETIVO(S):**

Realizar investigación sobre conceptos de Integridad de la BD y aplicarlo en un ejemplo práctico

La actividad le permite al estudiante enriquecer su experiencia en el desarrollo y construcción de conocimientos del curso de Base de Datos II.

1. **METODOLOGÍA:**
2. Divida el salón en grupos.
3. Distribuya el taller a cada grupo.
4. Cada equipo contará con un período de laboratorio (2 períodos de 45 min) para dar respuesta a las preguntas.
5. Primero realice el taller de forma individual y una vez desarrollado haga una reunión video llamada en equipo para que colaborativamente den respuesta en equipo las preguntas del ejercicio.
6. Se deben recibir en sección tareas de la plataforma de clases virtuales las respectivas asignación resuelta o realizar el test online del taller.
7. **PROCEDIMIENTO O ENUNCIADO DE LA EXPERIENCIA:**

**Caso 1. Crear Triggers – SQL Server**

Una librería almacena los datos de sus libros en una tabla denominada "libros" y controla las acciones que los empleados realizan sobre dicha tabla almacenando en la tabla "control" el nombre del usuario y la fecha, cada vez que se modifica el "precio" de un libro.

1. Crear las tablas con las siguientes estructuras:
   1. Tabla Libros
      1. codigo number(6)
      2. titulo varchar2(40)
      3. autor varchar2(30)
      4. editorial varchar2(20)
      5. precio number(6,2)
   2. Tabla Control
      1. usuario varchar2(30)
      2. fecha date
2. Ingresemos algunos registros en "libros":

insert into libros values(100,'Uno','Richard Bach','Planeta',25);

insert into libros values(103,'El aleph','Borges','Emece',28);

insert into libros values(105,'Matematica estas ahi','Paenza','Nuevo siglo',12);

insert into libros values(120,'Aprenda PHP','Molina Mario','Nuevo siglo',55);

insert into libros values(145,'Alicia en el pais de las maravillas','Carroll','Planeta',35);

1. Establecer el formato de fecha para que muestre "DD/MM/YYYY HH24:MI"
2. Crear un desencadenador a nivel de fila que se dispare cada vez que se actualiza el campo "precio"; el trigger debe ingresar en la tabla "control", el nombre del usuario, la fecha y la hora en la cual se realizó un "update" sobre "precio" de "libros"

**Caso 2. Crear Triggers MySQL**

1.Vamos a crear una base de datos sencilla, con sólo dos tablas. En una tabla guardaremos datos de personas, y en la otra anotaremos cuando se ha introducido cada dato. La estructura básica sería ésta:

**Nombre de Base de Datos** à Ejemplotriggers

**Estructura Tabla Persona:**

codigo varchar(10)

nombre varchar(50)

edad decimal(3)

PRIMARY KEY (`codigo`)

**Estructura Tabla NuevosDatos**

codigo varchar(10)

cuando date

tipo char(1)

2. Para que se añada un dato en la segunda tabla cada vez que insertemos en la primera, creamos un **TRIGGER** que saltará con un **AFTER INSERT**. Para indicar los pasos que debe hacer, se usa la expresión "**FOR EACH ROW**" (para cada fila)

(Los datos que introduciremos serán: el código de la persona, la fecha actual y una letra "i" para indicar que el cambio ha sido la "inserción" de un dato nuevo).

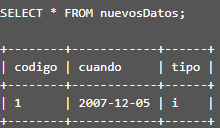
3. Si ahora introducimos un dato en la tabla personas:

**INSERT INTO persona**

**VALUES ('1','Juan',20);**

4. La tabla de **"nuevosDatos**" habrá cambiado:

SELECT \* FROM nuevosDatos;



**Nota 1**: Si en vez de monitorizar los INSERT, queremos controlar los UPDATE, el valor actual del nombre es "NEW.nombre", pero también podemos saber el valor anterior con "OLD.nombre", de modo que podríamos almacenar en una tabla todos los detalles sobre el cambio que ha hecho el usuario.

**Nota 2**: Si no queremos guardar sólo la fecha actual, sino la fecha y la hora, el campo debería ser de tipo DATETIME, y sabríamos el instante actual con "NOW()"

CREATE TRIGGER modificacion

AFTER INSERT ON persona

FOR EACH ROW

INSERT INTO nuevosDatos

VALUES (NEW.codigo, CURRENT\_DATE, 'i');

5.Si queremos indicar que se deben dar secuencias de pasos más largas, deberemos tener en cuenta dos cosas: cuando sean varias órdenes, deberán encerrarse entre BEGIN y END; además, como cada una de ellas terminará en punto y coma, deberemos cambiar momentáneamente el "delimitador" (DELIMITER) de MySQL, para que no piense que hemos terminado en cuanto aparezca el primer punto y coma:

DELIMITER |

CREATE TRIGGER validacionPersona BEFORE INSERT ON persona

FOR EACH ROW BEGIN

SET NEW.codigo = UPPER(NEW.codigo);

SET NEW.edad = IF(NEW.edad = 0, NULL, NEW.edad);

END;

|

DELIMITER ;

Si añadimos un dato que tenga un código en minúsculas y una edad 0, y pedimos que se nos muestre el resultado, veremos ésto:

INSERT INTO persona

VALUES ('p','Pedro',0)

+--------+---------+------+

| codigo | nombre | edad |

+--------+---------+------+

| 1 | Juan | 20 |

| P | Pedro | NULL |

+--------+---------+------+

**Ejercicios Propuestos**

* Amplía esta base de datos de ejemplo, para que antes de cada borrado, se anote en una tabla de "copia de seguridad" el dato que se va a borrar.
* Amplía esta base de datos de ejemplo, para que se antes de cada modificación se anote en una tabla "historico" el valor que antes tenía el registro que se va a modificar, junto con la fecha y hora actual.

**Caso 3. Crear Triggers MySQL**

Crea una base de datos llamada test que contenga una tabla llamada alumnos con las siguientes columnas.

**Tabla alumnos:**

id (entero sin signo)

nombre (cadena de caracteres)

apellido1 (cadena de caracteres)

apellido2 (cadena de caracteres)

nota (número real)

Una vez creada la tabla escriba dos triggers con las siguientes características:

**Trigger 1: trigger\_check\_nota\_before\_insert**

Se ejecuta sobre la tabla alumnos.

Se ejecuta antes de una operación de inserción.

Si el nuevo valor de la nota que se quiere insertar es negativo, se guarda como 0.

Si el nuevo valor de la nota que se quiere insertar es mayor que 10, se guarda como 10.

**Trigger2 : trigger\_check\_nota\_before\_update**

Se ejecuta sobre la tabla alumnos.

Se ejecuta antes de una operación de actualización.

Si el nuevo valor de la nota que se quiere actualizar es negativo, se guarda como 0.

Si el nuevo valor de la nota que se quiere actualizar es mayor que 10, se guarda como 10.

Una vez creados los triggers escriba varias sentencias de inserción y actualización sobre la tabla alumnos y verifica que los triggers se están ejecutando correctamente.

**Caso 4. Crear Triggers MySQL**

Crea una tabla que se llame notificaciones que tenga las siguientes columnas:

id (entero sin signo, autoincremento y clave primaria)

fecha\_hora: marca de tiempo con el instante del pago (fecha y hora)

total: el valor del pago (real)

codigo\_cliente: código del cliente que realiza el pago (entero)

Escriba un trigger que nos permita llevar un control de los pagos que van realizando los clientes. Los detalles de implementación son los siguientes:

Nombre: trigger\_notificar\_pago

Se ejecuta sobre la tabla pago.

Se ejecuta después de hacer la inserción de un pago.

Cada vez que un cliente realice un pago (es decir, se hace una inserción en la tabla pago), el trigger deberá insertar un nuevo registro en una tabla llamada notificaciones.

Escriba algunas sentencias SQL para comprobar que el trigger funciona correctamente.

**NOTA: DESARROLLE SUS RESPUESTAS EN LA SECCION G. RESULTADOS DE ESTA PLANTILLA.**

1. **RECURSOS:**

Laptop o PC con los recursos, Internet, material de clases preparado por el docente.

1. **RESULTADOS:**

**Caso 1:**

1. **Creación de tabla**

CREATE TABLE libros(

codigo DECIMAL(6),

titulo VARCHAR(40),

autor VARCHAR (30),

editorial VARCHAR (20),

precio DECIMAL(6,2)

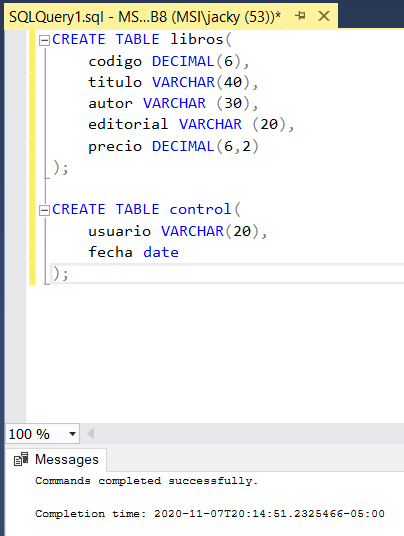
);

CREATE TABLE control(

usuario VARCHAR(20),

fecha date

);



1. **Inserción de datos en la tabla libros**

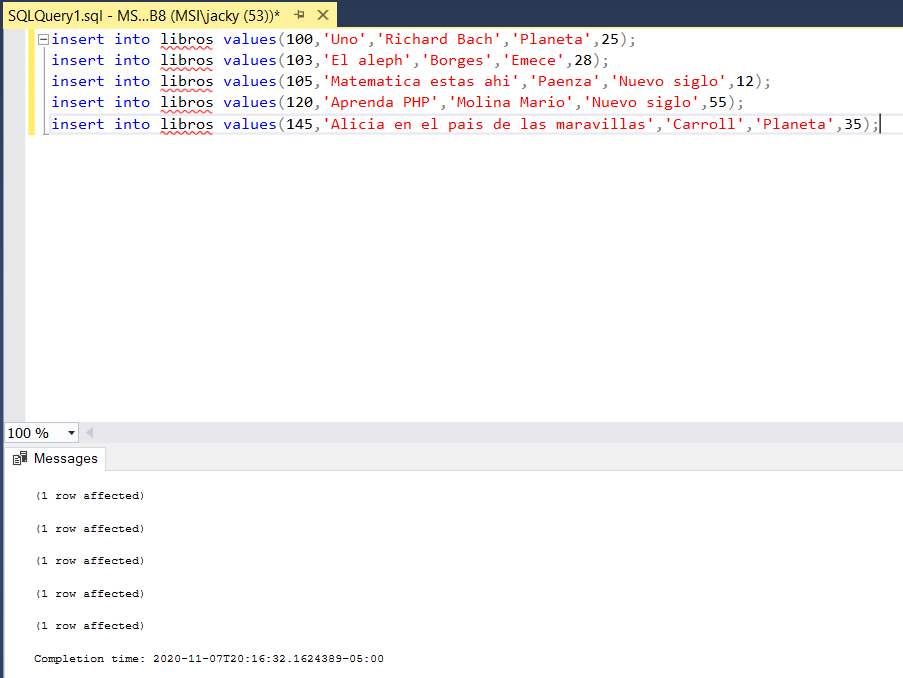
insert into libros values(100,'Uno','Richard Bach','Planeta',25);

insert into libros values(103,'El aleph','Borges','Emece',28);

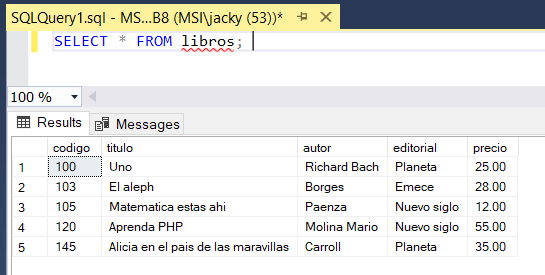
insert into libros values(105,'Matematica estas ahi','Paenza','Nuevo siglo',12);

insert into libros values(120,'Aprenda PHP','Molina Mario','Nuevo siglo',55);

insert into libros values(145,'Alicia en el pais de las maravillas','Carroll','Planeta',35);



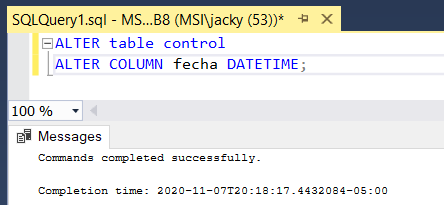
SELECT \* FROM libros;



1. **Establecer formato de fecha**

ALTER table control

ALTER COLUMN fecha DATETIME;



1. **Creación del trigger**

GO

CREATE OR ALTER TRIGGER tg\_LibroCam

ON libros

AFTER UPDATE

AS

BEGIN

IF UPDATE(precio)

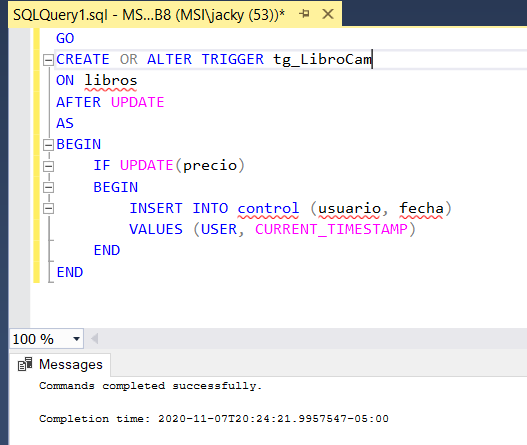
BEGIN

INSERT INTO control (usuario, fecha)

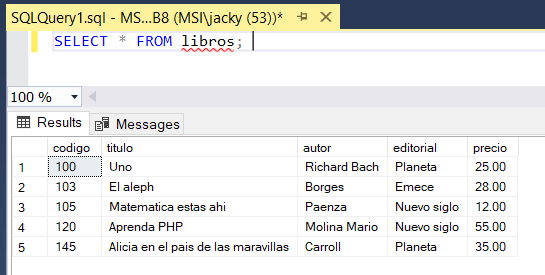
VALUES (USER, CURRENT\_TIMESTAMP)

END

END



**ANTES**



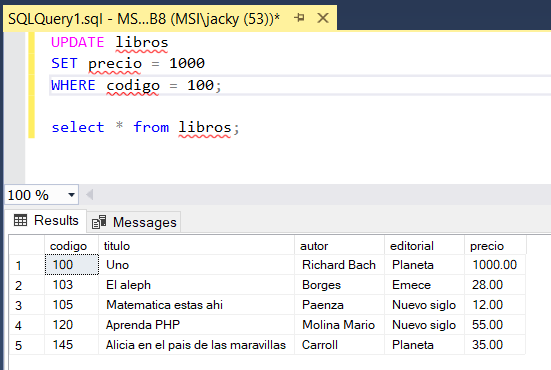
**DESPUES**

UPDATE libros

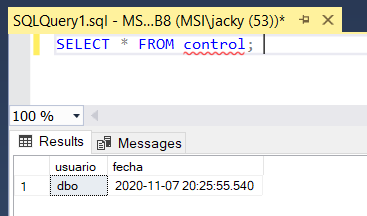
SET precio = 1000

WHERE codigo = 100;

select \* from libros;



SELECT \* FROM control;



**Caso 2:**

1. **Creación de database y tablas**

create database Ejemplotriggers;

use Ejemplotriggers;

create table Persona(

codigo varchar(10) not null PRIMARY KEY,

nombre varchar(50),

edad decimal(3)

);

create table NuevosDatos(

codigo varchar(10),

cuando date,

tipo char(1)

);

1. **Creación del trigger modificación**

DELIMITER $$

create trigger modificacion

after insert on persona

for each row

insert into NuevosDatos

values (NEW.codigo, current\_date, 'i')

$$

1. **Inserción de valores a la tabla persona**

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

1. **Tabla de NuevosDatos**

Interfaz de usuario gráfica, Tabla

Descripción generada automáticamente

1. **Secuencias de Pasos mas Largos**

DELIMITER |

CREATE TRIGGER validacionPersona BEFORE INSERT ON persona

FOR EACH ROW BEGIN

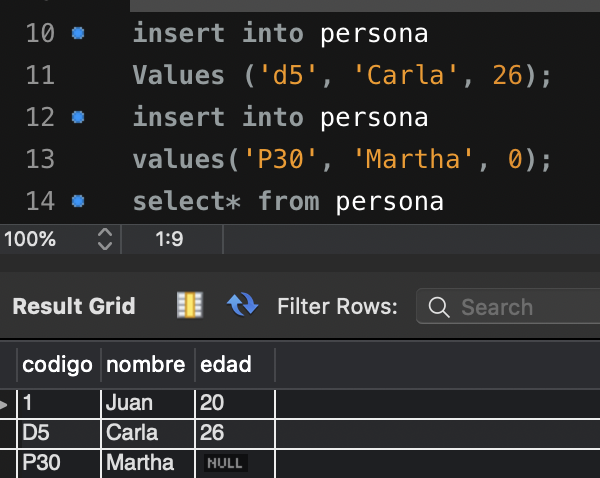
SET NEW.codigo = UPPER(NEW.codigo);

SET NEW.edad = IF(NEW.edad = 0, NULL, NEW.edad);

END;

|

DELIMITER ;



1. Problemas Propuestos

**Creación de la tabla de copia de seguridad**

create table backup(

codigo varchar(10) not null PRIMARY KEY,

nombre varchar(50),

edad decimal(3)

);

**Creación del trigger**

CREATE TRIGGER copiaseg

Before DELETE ON persona

for each row

insert into backup values(old.codigo, old.nombre, old.edad)

Antes de eliminar: Despues de eliminar una persona:

Table

Description automatically generated A picture containing graphical user interface

Description automatically generated

Tabla de copia de seguridad:

Graphical user interface, application

Description automatically generated

**Creacion de la tabla historico**

create table historico(

codigo varchar(3),

nombre varchar(40),

edad int,

modificado datetime default current\_timestamp);

**Creacion de trigger**

CREATE TRIGGER historic

Before UPDATE ON persona

for each row

insert into historico(codigo,nombre,edad) values(old.codigo, old.nombre, old.edad);

**Tabla persona despues de un update**

A screenshot of a cell phone

Description automatically generated

Tabla Historico luego de la modificacion

Graphical user interface, application

Description automatically generated

**Caso 3:**

-- crear tabla alumnos

drop table if exists alumnos;

CREATE TABLE alumnos (

id INT UNSIGNED AUTO\_INCREMENT PRIMARY KEY,

nombre VARCHAR(50) NOT NULL,

apellido1 VARCHAR(50) NOT NULL,

apellido2 VARCHAR(50),

nota FLOAT

);

-- Trigger 1: trigger\_check\_nota\_before\_insert

DELIMITER $$

DROP TRIGGER IF EXISTS trigger\_check\_nota\_before\_insert$$

CREATE TRIGGER trigger\_check\_nota\_before\_insert

BEFORE INSERT

ON alumnos FOR EACH ROW

BEGIN

IF NEW.nota < 0 THEN

set NEW.nota = 0;

ELSEIF NEW.nota > 10 THEN

set NEW.nota = 10;

END IF;

END $$

-- Trigger 2: trigger\_check\_nota\_before\_update

DELIMITER $$

DROP TRIGGER IF EXISTS trigger\_check\_nota\_before\_update$$

CREATE TRIGGER trigger\_check\_nota\_before\_update

BEFORE UPDATE

ON alumnos FOR EACH ROW

BEGIN

IF NEW.nota < 0 THEN

set NEW.nota = 0;

ELSEIF NEW.nota > 10 THEN

set NEW.nota = 10;

END IF;

END $$

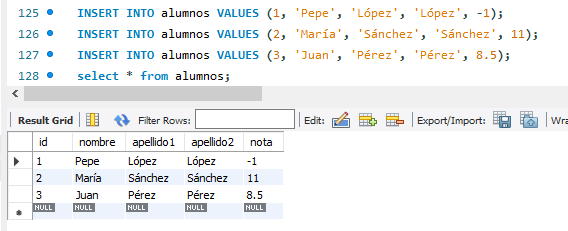
DELIMITER ;

INSERT INTO alumnos VALUES (1, 'Pepe', 'López', 'López', -1);

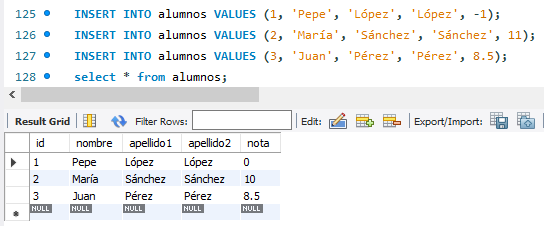
INSERT INTO alumnos VALUES (2, 'María', 'Sánchez', 'Sánchez', 11);

INSERT INTO alumnos VALUES (3, 'Juan', 'Pérez', 'Pérez', 8.5);

**Sin trigger:**



**Con trigger:**

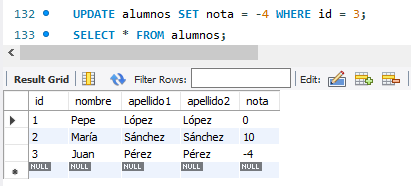


UPDATE alumnos SET nota = -4 WHERE id = 3;

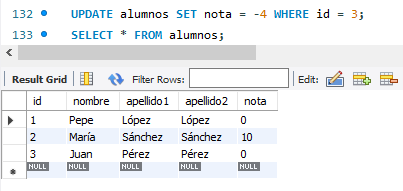
UPDATE alumnos SET nota = 14 WHERE id = 3;

UPDATE alumnos SET nota = 9.5 WHERE id = 3;

**Sin trigger:**



**Con trigger:**



**Caso 4:**

**-- tabla pago**

create table pago (

codigo\_cliente int primary key not null auto\_increment,

total real

);

**-- tabla de notificaciones**

create table notificaciones (

id int primary key auto\_increment,

fecha\_hora timestamp default current\_timestamp,

total real ,

codigo\_cliente int,

foreign key (codigo\_cliente) references pago(codigo\_cliente)

);

**-- trigger de notificaciones**

DELIMITER //

create trigger trigger\_notificar\_pago

After insert on pago

for each row

begin

Insert into notificaciones(total, codigo\_cliente) values (new.total, new.codigo\_cliente);

END//

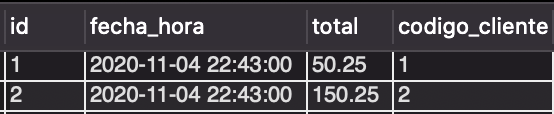
**-- pruebas para verificar el trigger**

Insert into pago(total) value (50.25);

Insert into pago(total) value (150.25);

**-- se realiza la consulta**

**select\* from notificaciones;**



1. **CONSIDERACIONES FINALES – CONCLUSIONES:**

Opinión del estudiante(s) sobre el logro del objetivo y el desarrollo de la Asignación.

Opinión del estudiante(s) sobre el valor de esta herramienta para apoyar los procesos de aprendizaje.

En este trabajo logramos aplicar lo aprendido sobre los triggers de múltiples maneras en MySQL, entre nuestro grupo de trabajo nos repartimos el trabajo y nos apoyamos entre nosotros cuando uno tenía dudas sobre los triggers, ya que algunos no teníamos demasiada experiencia trabajando con estos.

En conclusión, el desarrollo de este laboratorio fue una muy buena manera poner en práctica lo aprendido durante las clases virtuales.

1. **BIBLIOGRAFIA:**

* Material didáctico suministrado por la profesora
* Internet

1. **RÚBRICAS:**

|  |  |
| --- | --- |
| ***Detalle - Evaluar*** | ***Valor de la respuesta*** |
| ***Sección G – (100 PUNTOS)*** | ***100 pts*** |