Bases de Datos

Pavel Miron

UD4 - Práctica 4.

Índice

Contenido

1. Crea los siguientes triggers. Muestra el código empleado así como una captura del funcionamiento de un insert que haga saltar a cada trigger:

a. Vuelve al instituto, chaval. Muestra un mensaje de error si intentamos crear un empleado que tenga menos de 16 años

```
DELIMITER $$
CREATE TRIGGER edad_empleado
BEFORE INSERT ON employees
FOR EACH ROW
BEGIN
    IF TIMESTAMPDIFF(YEAR, NEW.birth_date, CURDATE()) < 16 THEN
       SIGNAL SQLSTATE '50001' SET MESSAGE_TEXT = 'Error. La edad debe ser
    superior a 16';
    END IF;
END $$

INSERT INTO employees (emp_no, birth_date, first_name, last_name, gender, hire_date)
VALUES (1001, '2015-05-01', 'Juan', 'Pérez', 'M', '2025-04-14');
```

b. El nombre de moda. Muestra un mensaje de error si intentamos registrar a un empleado/a que no se llame Lucía o Martín (según el INE, son los dos nombres más populares entre los recién nacidos en España)

```
DELIMITER $$
CREATE TRIGGER validar_nombre
BEFORE INSERT ON employees
FOR EACH ROW
BEGIN
    IF NEW.first_name NOT IN ('Lucia', 'Martin') THEN
        SIGNAL SQLSTATE '50001' SET MESSAGE_TEXT = 'Error. El nombre del empleado debe ser Lucia o Martin';
    END IF;
END $$

INSERT INTO employees (emp_no, birth_date, first_name, last_name, gender, hire_date)
VALUES (1002, '1990-07-10', 'Ana', 'Gómez', 'F', '2025-04-14');
```

c. Regreso al futuro. Escoge alguna de las tablas que controlan fecha inicio y fecha fin (v. gr. dept_emp) y evita que la fecha inicio pueda ser posterior a la de fin

```
DELIMITER $$
CREATE TRIGGER validar_fecha_dept
BEFORE INSERT ON dept_emp
FOR EACH ROW
BEGIN
    IF NEW.from_date > to_date THEN
        SIGNAL SQLSTATE '50001' SET MESSAGE_TEXT = 'Error. La fecha de inicio no puede ser posterior a la fecha de fin';
    END IF;
END $$

INSERT INTO dept_emp (emp_no, dept_no, from_date, to_date)
VALUES (1001, 'd001', '2025-05-01', '2025-04-01');
```

d. Jefatura del siglo pasado. Asegúrate de que cualquier empleado que sea nombrado manager lleva al menos 10 años trabajando en la empresa

```
DELIMITER $$

CREATE TRIGGER valida_antiguedad_manager
BEFORE INSERT ON dept_manager
FOR EACH ROW
BEGIN
    IF TIMESTAMPDIFF(YEAR, (SELECT hire_date FROM employees WHERE emp_no = NEW.emp_no), CURDATE()) < 10 THEN
        SIGNAL SQLSTATE '50001' SET MESSAGE_TEXT = 'El empleado debe tener al menos 10 años de antigüedad para ser nombrado manager';
    END IF;
END $$

INSERT INTO dept_manager (emp_no, dept_no, from_date, to_date)
VALUES (1001, 'd001', '2025-04-14', '9999-01-01');
```

2. Crea los siguientes triggers y sus correspondientes tablas. Muestra el código empleado así como una captura del funcionamiento de un insert que haga saltar a cada trigger:

a. Ricachones. Crea una tabla en la que guardarás los detalles de todos los empleados que reciban un salario superior a 100.000 €

```
CREATE TABLE ricachones (
    emp_no INT NOT NULL,
    salary INT NOT NULL,
    from_date DATE NOT NULL,
    to_date DATE NOT NULL,
    PRIMARY KEY (emp_no, from_date),
    FOREIGN KEY (emp_no) REFERENCES employees (emp_no) ON DELETE CASCADE
);

DELIMITER $$
CREATE TRIGGER inserta_ricachones
AFTER INSERT ON salaries
FOR EACH ROW
BEGIN
    IF NEW.salary > 100000 THEN
        INSERT INTO ricachones (emp_no, salary, from_date, to_date)
        VALUES (NEW.emp_no, NEW.salary, NEW.from_date, NEW.to_date);
    END IF;
END $$

INSERT INTO salaries (emp_no, salary, from_date, to_date)
VALUES (1001, 120000, '2025-01-01', '2025-12-31');
```

b. Los reyes del mambo. Crea otra tabla en la que guardes los detalles de un jefe (dept_manager) en el momento que deja de serlo (esto es, el to_date deja de apuntar al fin de los días (9999-01-01)

```
CREATE TABLE ex_managers (
    emp_no INT NOT NULL,
    dept_no CHAR(4) NOT NULL,
    from_date DATE NOT NULL,
    to_date DATE NOT NULL,
    PRIMARY KEY (emp_no, dept_no),
    FOREIGN KEY (emp_no) REFERENCES employees (emp_no) ON DELETE CASCADE,
    FOREIGN KEY (dept_no) REFERENCES departments (dept_no) ON DELETE CASCADE
);

DELIMITER $$
CREATE TRIGGER inserta_ex_manager
BEFORE UPDATE ON dept_manager
FOR EACH ROW
BEGIN
    IF NEW.to_date = '9999-01-01' THEN
        INSERT INTO ex_managers (emp_no, dept_no, from_date, to_date)
        VALUES (NEW.emp_no, NEW.dept_no, NEW.from_date, NEW.to_date);
    END IF;
END $$

UPDATE dept_manager
SET to_date = '9999-01-01'
WHERE emp_no = 1001 AND dept_no = 'd001';
```

c. Esto lo tiene que aprobar dirección. Crea una tabla que almacene el nuevo nombre de los departamentos y su código siempre que alguien intente renombrarlos

```
CREATE TABLE dept_name_changes (
    dept_no CHAR(4) NOT NULL,
    old_name VARCHAR(40) NOT NULL,
    new_name VARCHAR(40) NOT NULL,
    change_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    PRIMARY KEY (dept_no, change_date),
    FOREIGN KEY (dept_no) REFERENCES departments (dept_no) ON DELETE CASCADE
);

DELIMITER $$

CREATE TRIGGER guarda_cambio_nombre_departamento
BEFORE UPDATE ON departments
FOR EACH ROW
BEGIN
    IF NEW.dept_name != OLD.dept_name THEN
        INSERT INTO dept_name_changes (dept_no, old_name, new_name)
        VALUES (OLD.dept_no, OLD.dept_name, NEW.dept_name);
    END IF;
END $$

UPDATE departments
SET dept_name = 'New Department Name'
WHERE dept_no = 'd001';
```