

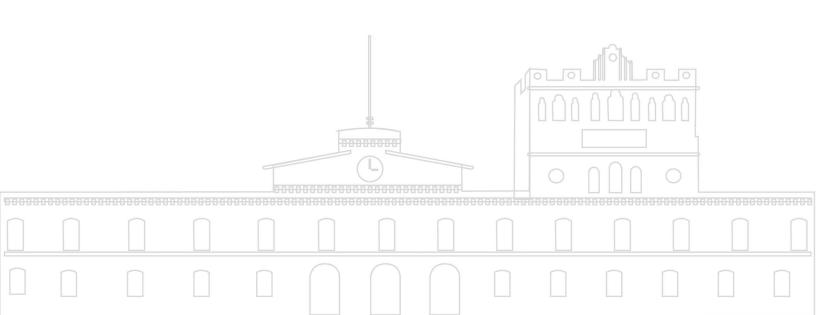


# REPORTE DE PRÁCTICA 1.5

NOMBRE DE LA PRÁCTICA: 1.5 Práctica 0

ALUMNO: Axel Damian Ortiz Simon

Dr. Eduardo Cornejo-Velázquez



# 1. Marco teórico

En el ámbito de las bases de datos, es fundamental comprender y manejar diversos elementos que permiten una gestión eficiente y segura de la información. Entre estos elementos destacan los procedimientos almacenados, las funciones, las estructuras de control condicionales y repetitivas, y los disparadores. Cada uno de estos componentes desempeña un papel crucial en la manipulación y administración de datos, facilitando operaciones complejas y automatizando tareas dentro del sistema de gestión de bases de datos.

# 1. Procedimientos almacenados (Procedure)

Los procedimientos almacenados son conjuntos de instrucciones SQL que se almacenan en el servidor de bases de datos y se pueden ejecutar cuando se necesiten. Estos procedimientos permiten encapsular lógica de negocio en la base de datos, mejorando la eficiencia y seguridad al reducir el tráfico entre la aplicación y el servidor.

Ejemplo 1: Procedimiento para actualizar el estado de un vehículo

Ejemplo 2: Procedimiento para registrar una nueva ruta

```
Listing 2: RegistrarNuevaRuta
```

# 2. Functions (Function)

DELIMITER ;

Ejemplo 1: Función para calcular la edad de un usuario

Listing 3: CalcularEdad

```
DELIMITER //
```

```
CREATE FUNCTION CalcularEdad (fechaNacimiento DATE) RETURNS INT DETERMINISTIC
BEGIN
    RETURN TIMESTAMPDIFF(YEAR, fechanacimiento, CURDATE());
END //
DELIMITER ;
```

## Ejemplo 2: Función para obtener el saldo de un usuario

```
Listing 4: ObtenerSaldo
```

```
DELIMITER //
CREATE FUNCTION ObtenerSaldo(idUsuario INT) RETURNS DECIMAL(10,2) DETERMINISTIC
    DECLARE saldo DECIMAL(10,2);
    SELECT balance INTO saldo FROM Cuentas WHERE usuario_id = idUsuario;
    RETURN saldo;
END //
DELIMITER ;
```

# 3. Estructuras de control condicionales y repetitivas

#### Ejemplo 1: Uso de IF en un procedimiento

DELIMITER //

DELIMITER ;

```
Listing 5: UsoDeIF
CREATE PROCEDURE Verificar Saldo (IN id Usuario INT, OUT mensaje VARCHAR (50))
```

```
BEGIN
    DECLARE saldo DECIMAL(10,2);
    SELECT balance INTO saldo FROM Cuentas WHERE usuario_id = idUsuario;
    IF saldo > 1000 THEN
        SET mensaje = 'Saldo suficiente';
    ELSE
        SET mensaje = 'Saldo insuficiente';
    END IF;
END //
```

#### Ejemplo 2: Uso de WHILE en un procedimiento

```
Listing 6: UsoDeWhile
```

```
DELIMITER //
CREATE PROCEDURE ContarHastaN (IN limite INT)
BEGIN
    DECLARE contador INT DEFAULT 0:
    WHILE contador < limite DO
        SET contador = contador + 1;
    END WHILE;
END //
DELIMITER ;
```

## 4. Disparadores (Triggers)

# Ejemplo 1: Trigger para auditar inserciones en una tabla

```
Listing 7: AuditarInserciones
```

CREATE TRIGGER InsertarLog AFTER INSERT ON Usuarios

```
FOR EACH ROW
```

BEGIN

 $\label{eq:insert} \mbox{INSERT INTO Log (accion, usuario\_id, fecha) VALUES ('INSERT', NEW.id, NOW());} \\ END;$ 

# Ejemplo 2: Trigger para prevenir eliminación de datos importantes

Listing 8: PrevenirEliminacion

CREATE TRIGGER PrevenirEliminacion BEFORE DELETE ON Cuentas FOR EACH ROW BEGIN

SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = 'No se permite eliminar cuentas'; END;

# Referencias Bibliográficas

# References

- [1] Coronel, C., Morris, S., Rob, P. (2020). Database Systems: Design, Implementation, and Management (13<sup>a</sup> ed.). Cengage Learning.
- [2] Elmasri, R., Navathe, S. (2016). Fundamentals of Database Systems (7ª ed.). Pearson.
- [3] DuBois, P. (2008). MySQL: The Complete Reference (2ª ed.). McGraw-Hill Education.
- [4] Ghiani, G., Laporte, G., Musmanno, R. (2013). Introduction to Logistics Systems Management ( $2^{\underline{a}}$  ed.). Wiley.
- [5] Hernandez, M. J. (2013). Database Design for Mere Mortals (3ª ed.). Addison-Wesley.