



**Universidad  
Tecnológico**

**Campus**  
Campus Cúmbres

**Materia**  
Base de Datos

**Actividad**  
Actividad 4

**Maestra**  
Edwige Jazzmín Novelo Villegas

---

**Alumnos**

Juan Porfirio Torres Rojas  
#07099471

Jesus David Marroquin Peña  
#07101520

# Índice

Índice	2
Instrucciones	3
La base de datos	4
Datos	6
EJERCICIOS CASE	9
EER Diagrama	17
Liga GitHub	17

# Instrucciones

Esta actividad se puede realizar de forma individual o en equipos de hasta tres personas.

Actividad: Realicen los ejercicios de CASE y de Procedimientos que se encuentran en nuestro Drive, el documento se llama Actividad4\_Ejercicios a entregar.

El código SQL de la creación de los CASE y Procedimientos (con su ejecución y comprobación).

Capturas de pantalla de los resultados de la ejecución de cada consulta SQL y la explicación pertinente.

Enlace de Código: Proporcionen un enlace a su código SQL subido en una plataforma como GitHub.

Archivos .sql: Incluyan los archivos .sql.

# La base de datos

```
CREATE DATABASE IF NOT EXISTS veterinaria;  
USE veterinaria;
```

```
CREATE TABLE propietarios (  
    id_propietario INT PRIMARY KEY AUTO_INCREMENT,  
    nombre VARCHAR(100) NOT NULL,  
    apellido VARCHAR(100) NOT NULL,  
    telefono VARCHAR(15),  
    email VARCHAR(100),  
    direccion TEXT,  
    fecha_registro DATE DEFAULT (CURRENT_DATE)  
);
```

```
CREATE TABLE mascotas (  
    id_mascota INT PRIMARY KEY AUTO_INCREMENT,  
    nombre VARCHAR(50) NOT NULL,  
    especie VARCHAR(30) NOT NULL,  
    raza VARCHAR(50),  
    edad INT,  
    peso DECIMAL(5,2),  
    color VARCHAR(30),  
    id_propietario INT NOT NULL,  
    fecha_registro DATE DEFAULT (CURRENT_DATE),  
    FOREIGN KEY (id_propietario) REFERENCES propietarios(id_propietario)  
);
```

```
CREATE TABLE veterinarios (  
    id_veterinario INT PRIMARY KEY AUTO_INCREMENT,  
    nombre VARCHAR(100) NOT NULL,  
    apellido VARCHAR(100) NOT NULL,  
    cedula_profesional VARCHAR(20) UNIQUE,  
    especialidad VARCHAR(100),  
    telefono VARCHAR(15),  
    email VARCHAR(100)  
);
```

```
CREATE TABLE citas (  
    id_cita INT PRIMARY KEY AUTO_INCREMENT,  
    fecha_cita DATE NOT NULL,  
    hora_cita TIME NOT NULL,  
    motivo TEXT,  
    estado VARCHAR(20) DEFAULT 'Programada',  
    id_mascota INT NOT NULL,  
    id_veterinario INT NOT NULL,
```

```
FOREIGN KEY (id_mascota) REFERENCES mascotas(id_mascota),  
FOREIGN KEY (id_veterinario) REFERENCES veterinarios(id_veterinario)  
);
```

```
CREATE TABLE tratamientos (  
    id_tratamiento INT PRIMARY KEY AUTO_INCREMENT,  
    nombre_tratamiento VARCHAR(100) NOT NULL,  
    descripcion TEXT,  
    precio DECIMAL(10,2),  
    duracion_minutos INT  
);
```

```
CREATE TABLE historial_medico (  
    id_historial INT PRIMARY KEY AUTO_INCREMENT,  
    fecha_consulta DATE NOT NULL,  
    diagnostico TEXT,  
    observaciones TEXT,  
    peso_actual DECIMAL(5,2),  
    temperatura DECIMAL(4,1),  
    id_mascota INT NOT NULL,  
    id_veterinario INT NOT NULL,  
    id_tratamiento INT,  
    FOREIGN KEY (id_mascota) REFERENCES mascotas(id_mascota),  
    FOREIGN KEY (id_veterinario) REFERENCES veterinarios(id_veterinario),  
    FOREIGN KEY (id_tratamiento) REFERENCES tratamientos(id_tratamiento)  
);
```

```
CREATE TABLE personas (  
    id_persona INT PRIMARY KEY AUTO_INCREMENT,  
    nombre VARCHAR(100) NOT NULL,  
    apellido VARCHAR(100) NOT NULL  
);
```

```
CREATE TABLE invitados (  
    id_invitado INT PRIMARY KEY AUTO_INCREMENT,  
    id_persona INT NOT NULL,  
    metodo_invitacion VARCHAR(20),  
    FOREIGN KEY (id_persona) REFERENCES personas(id_persona)  
);
```

```
CREATE TABLE confirmaciones (  
    id_confirmacion INT PRIMARY KEY AUTO_INCREMENT,  
    id_persona INT NOT NULL,  
    estado VARCHAR(20),  
    FOREIGN KEY (id_persona) REFERENCES personas(id_persona)  
);
```

## Datos

```
INSERT INTO propietarios (nombre, apellido, telefono, email, direccion) VALUES
('Juan', 'Pérez', '555-1234', 'juan.perez@email.com', 'Av. Principal 123'),
('María', 'González', '555-5678', 'maria.gonzalez@email.com', 'Calle Secundaria 456'),
('Carlos', 'López', '555-9012', 'carlos.lopez@email.com', 'Colonia Centro 789');
```

[illegible]

```
INSERT INTO veterinarios (nombre, apellido, cedula_profesional, especialidad, telefono, email) VALUES ('Dr. Ana', 'Martínez', 'VET001', 'Medicina General', '555-1111', 'ana.martinez@vet.com'), ('Dr. Luis', 'Rodríguez', 'VET002', 'Cirugía', '555-2222', 'luis.rodriquez@vet.com');
```

id_veterinario	nombre	apellido	cedula_profesional	especialidad	telefono	email
1	Dr. Ana	Martínez	VET001	Medicina General	555-1111	ana.martinez@vet.com
2	Dr. Luis	Rodríguez	VET002	Cirugía	555-2222	luis.rodriguez@vet.com
NULL	NULL	NULL	NULL	NULL	NULL	NULL

```
INSERT INTO mascotas (nombre, especie, raza, edad, peso, color, id_propietario) VALUES
('Max', 'Perro', 'Labrador', 3, 25.5, 'Dorado', 1),
('Luna', 'Gato', 'Siamés', 2, 4.2, 'Blanco', 1),
('Rocky', 'Perro', 'Bulldog', 5, 18.0, 'Marrón', 2),
('Mimi', 'Gato', 'Persa', 1, 3.8, 'Gris', 3);
```

[illegible]

```
INSERT INTO tratamientos (nombre_tratamiento, descripcion, precio, duracion_minutos)
VALUES
('Consulta General', 'Revisión médica completa', 300.00, 30),
('Vacunación', 'Aplicación de vacunas', 150.00, 15),
('Desparasitación', 'Tratamiento antiparasitario', 200.00, 20),
('Cirugía Menor', 'Procedimientos quirúrgicos menores', 800.00, 60);
```

id_tratamiento	nombre_tratamiento	descripcion	precio	duracion_minutos
1	Consulta General	Revisión médica completa	300.00	30
2	Vacunación	Aplicación de vacunas	150.00	15
3	Desparasitación	Tratamiento antiparasitario	200.00	20
4	Cirugía Menor	Procedimientos quirúrgicos menores	800.00	60
NULL	NULL	NULL	NULL	NULL

```
INSERT INTO personas (nombre, apellido) VALUES
('Miguel', 'González'),
('María', 'López'),
('José', 'Martínez'),
('Ana', 'Hernández'),
('Carlos', 'García'),
('Laura', 'Rodríguez'),
('Roberto', 'Pérez'),
('Sofía', 'Sánchez'),
('Diego', 'Ramírez'),
('Valentina', 'Torres');
```

id_persona	nombre	apellido
1	Miguel	González
2	María	López
3	José	Martínez
4	Ana	Hernández
5	Carlos	García
6	Laura	Rodríguez
7	Roberto	Pérez
8	Sofía	Sánchez
9	Diego	Ramírez
10	Valentina	Torres
NULL	NULL	NULL

```
INSERT INTO invitados (id_persona, metodo_invitacion) VALUES
(1, 'WhatsApp'),
(2, 'WhatsApp'),
(3, 'Email'),
(4, 'WhatsApp'),
(5, 'Llamada'),
(6, 'WhatsApp'),
(7, 'En persona'),
(8, 'Email'),
(9, 'WhatsApp'),
(10, 'Email');
```

id_invitado	id_persona	metodo_invitacion
1	1	WhatsApp
2	2	WhatsApp
3	3	Email
4	4	WhatsApp
5	5	Llamada
6	6	WhatsApp
7	7	En persona
8	8	Email
9	9	WhatsApp
10	10	Email
NULL	NULL	NULL

```
INSERT INTO confirmaciones (id_persona, estado) VALUES
(1, 'Confirma'),
(2, 'Confirma'),
(3, 'No puede'),
(4, 'Confirma'),
(5, 'Tal vez'),
(6, 'Confirma');
```

id_confirmacion	id_persona	estado
1	1	Confirma
2	2	Confirma
3	3	No puede
4	4	Confirma
5	5	Tal vez
6	6	Confirma
NULL	NULL	NULL



# EJERCICIOS CASE

-- Ejercicio 1: CASE SIMPLE

```
SELECT
  nombre,
  apellido,
  metodo_invitacion,
  CASE metodo_invitacion
    WHEN 'WhatsApp' THEN '🟢 Mensaje'
    WHEN 'Email' THEN '🟡 Correo'
    WHEN 'Llamada' THEN '🟠 Teléfono'
    WHEN 'En persona' THEN '🔴 Cara a cara'
    ELSE '🔴 Desconocido'
  END as tipo_invitacion
FROM personas p
JOIN invitados i ON p.id_persona = i.id_persona;
```

nombre	apellido	metodo_invitacion	tipo_invitacion
Miguel	González	WhatsApp	🟢 Mensaje
María	López	WhatsApp	🟢 Mensaje
José	Martínez	Email	🟡 Correo
Ana	Hernández	WhatsApp	🟢 Mensaje
Carlos	García	Llamada	🟠 Teléfono
Laura	Rodríguez	WhatsApp	🟢 Mensaje
Roberto	Pérez	En persona	🔴 Cara a cara
Sofía	Sánchez	Email	🟡 Correo
Diego	Ramírez	WhatsApp	🟢 Mensaje
Valentina	Torres	Email	🟡 Correo

-- Ejercicio 2: CASE SEARCHED

```
SELECT
  p.nombre,
  p.apellido,
  c.estado as confirmacion,
  CASE
    WHEN c.estado = 'Confirma' THEN '✅ Viene seguro'
    WHEN c.estado = 'No puede' THEN '❌ No puede venir'
    WHEN c.estado = 'Tal vez' THEN '🟡 Tal vez viene'
    WHEN c.estado IS NULL THEN '🟡 Sin respuesta'
    ELSE '💡 Estado raro'
  END as situacion
FROM personas p
LEFT JOIN confirmaciones c ON p.id_persona = c.id_persona
WHERE p.id_persona IN (SELECT id_persona FROM invitados);
```

nombre	apellido	confirmacion	situacion
Miguel	González	Confirma	✅ Viene seguro
María	López	Confirma	✅ Viene seguro
José	Martínez	No puede	❌ No puede venir
Ana	Hernández	Confirma	✅ Viene seguro
Carlos	García	Tal vez	🟡 Tal vez viene
Laura	Rodríguez	Confirma	✅ Viene seguro
Roberto	Pérez	NULL	🟡 Sin respuesta
Sofía	Sánchez	NULL	🟡 Sin respuesta
Diego	Ramírez	NULL	🟡 Sin respuesta
Valentina	Torres	NULL	🟡 Sin respuesta

-- Ejercicio 3: CASE SEARCHED en WHERE

```
SELECT
  p.nombre,
  p.apellido,
  i.metodo_invitacion,
  c.estado
FROM personas p
LEFT JOIN invitados i ON p.id_persona =
i.id_persona
LEFT JOIN confirmaciones c ON p.id_persona =
c.id_persona
WHERE
  CASE
    WHEN i.metodo_invitacion = 'WhatsApp' THEN c.estado = 'Confirma'
    WHEN i.metodo_invitacion = 'Email' THEN c.estado IN ('Confirma', 'Tal vez')
    ELSE TRUE
  END;
```

nombre	apellido	metodo_invitacion	estado
Miguel	González	WhatsApp	Confirma
María	López	WhatsApp	Confirma
Ana	Hernández	WhatsApp	Confirma
Carlos	García	Llamada	Tal vez
Laura	Rodríguez	WhatsApp	Confirma
Roberto	Pérez	En persona	NULL

-- PROCEDIMIENTOS ALMACENADOS

-- Ejercicio 4: Procedimiento sin parámetros

DELIMITER //

CREATE PROCEDURE mostrar\_todas\_mascotas()

BEGIN

SELECT

m.id\_mascota as 'ID Mascota',

m.nombre as 'Nombre Mascota',

m.especie as 'Especie',

m.raza as 'Raza',

m.edad as 'Edad',

m.peso as 'Peso (kg)',

CONCAT(p.nombre, ' ', p.apellido) as 'Propietario',

p.telefono as 'Teléfono Propietario'

FROM mascotas m

INNER JOIN propietarios p ON m.id\_propietario = p.id\_propietario

ORDER BY m.nombre;

END //

DELIMITER ;

-- Ejecutar procedimiento sin parámetros (Ejercicio 4)

CALL mostrar\_todas\_mascotas();

result grid   Filter Rows:   Export:   Wrap Cell Content:								
	ID Mascota	Nombre Mascota	Especie	Raza	Edad	Peso (kg)	Propietario	Teléfono Propietario
	2	Luna	Gato	Siamés	2	4.20	Juan Pérez	555-1234
	1	Max	Perro	Labrador	3	25.50	Juan Pérez	555-1234
	4	Mimi	Gato	Persa	1	3.80	Carlos López	555-9012
	3	Rocky	Perro	Bulldog	5	18.00	María González	555-5678



```

-- Ejercicio 5: Procedimiento con IN
DELIMITER //
CREATE PROCEDURE buscar_mascota_por_id(IN p_id INT)
BEGIN
    IF p_id <= 0 THEN
        SELECT 'Error: El ID debe ser mayor a 0' as Mensaje;
    ELSE
        SELECT
            id_mascota as 'ID',
            nombre as 'Nombre',
            especie as 'Especie',
            raza as 'Raza',
            edad as 'Edad (años)',
            peso as 'Peso (kg)'
        FROM mascotas
        WHERE id_mascota = p_id;
    END IF;
END //
DELIMITER ;

```

-- Ejecutar procedimiento con IN (Ejercicio 5)

CALL buscar\_mascota\_por\_id(1);

Result Grid |  Filter Rows:  | Export:  | Wrap Cell Conte

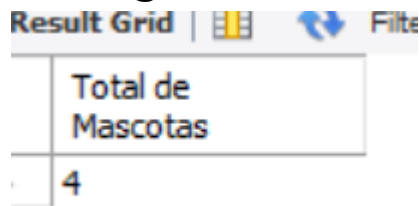
ID	Nombre	Especie	Raza	Edad (años)	Peso (kg)
1	Max	Perro	Labrador	3	25.50

CALL buscar\_mascota\_por\_id(0);

Result Grid		Filter Rows:
Mensaje		
▶	Error: El ID debe ser mayor a 0	

```
-- Ejercicio 6: Procedimiento con OUT
DELIMITER //
CREATE PROCEDURE contar_mascotas(OUT p_total INT)
BEGIN
    SELECT COUNT(*) INTO p_total FROM mascotas;
END //
DELIMITER ;
```

```
-- Ejecutar procedimiento con OUT (Ejercicio 6)
SET @resultado = 0;
CALL contar_mascotas(@resultado);
SELECT @resultado as "Total de Mascotas";
```



The screenshot shows a window titled "Result Grid" with a toolbar containing icons for grid view, table view, and a filter icon. Below the toolbar is a table with one column and one row.

Total de Mascotas
4

```

-- Ejercicio 7: Procedimiento con IN y OUT
DELIMITER //
CREATE PROCEDURE actualizar_peso_mascota(IN p_id INT, INOUT p_peso
DECIMAL(5,2))
BEGIN
    DECLARE peso_anterior DECIMAL(5,2) DEFAULT 0;
    DECLARE mascota_existe INT DEFAULT 0;

    SELECT COUNT(*) INTO mascota_existe
    FROM mascotas
    WHERE id_mascota = p_id;

    IF mascota_existe = 0 THEN
        SET p_peso = -1;
    ELSE
        SELECT peso INTO peso_anterior
        FROM mascotas
        WHERE id_mascota = p_id;

        UPDATE mascotas
        SET peso = p_peso
        WHERE id_mascota = p_id;

        SET p_peso = peso_anterior;
    END IF;
END //
DELIMITER ;

-- Ejecutar procedimiento con INOUT (Ejercicio 7)
SET @nuevo_peso = 27.8;
CALL actualizar_peso_mascota(1, @nuevo_peso);
SELECT @nuevo_peso as "Peso anterior";

```

Result Grid	
Peso anterior	
25.50	



```

-- Ejercicio 9: Procedimiento de validación
DELIMITER //
CREATE PROCEDURE validar_edad_mascota(IN p_id_mascota INT, OUT p_es_valida
VARCHAR(50))
BEGIN
    DECLARE v_edad INT DEFAULT 0;
    DECLARE mascota_existe INT DEFAULT 0;

    SELECT COUNT(*) INTO mascota_existe
    FROM mascotas
    WHERE id_mascota = p_id_mascota;

    IF mascota_existe > 0 THEN
        SELECT edad INTO v_edad
        FROM mascotas
        WHERE id_mascota = p_id_mascota;
    END IF;

    IF mascota_existe = 0 THEN
        SET p_es_valida = 'Error: Mascota no encontrada';
    ELSEIF v_edad <= 0 THEN
        SET p_es_valida = 'Error: Edad inválida';
    ELSE
        IF v_edad <= 25 THEN
            SET p_es_valida = 'Edad válida';
        ELSE
            SET p_es_valida = 'Edad muy alta (máximo 25 años)';
        END IF;
    END IF;
END //
DELIMITER ;

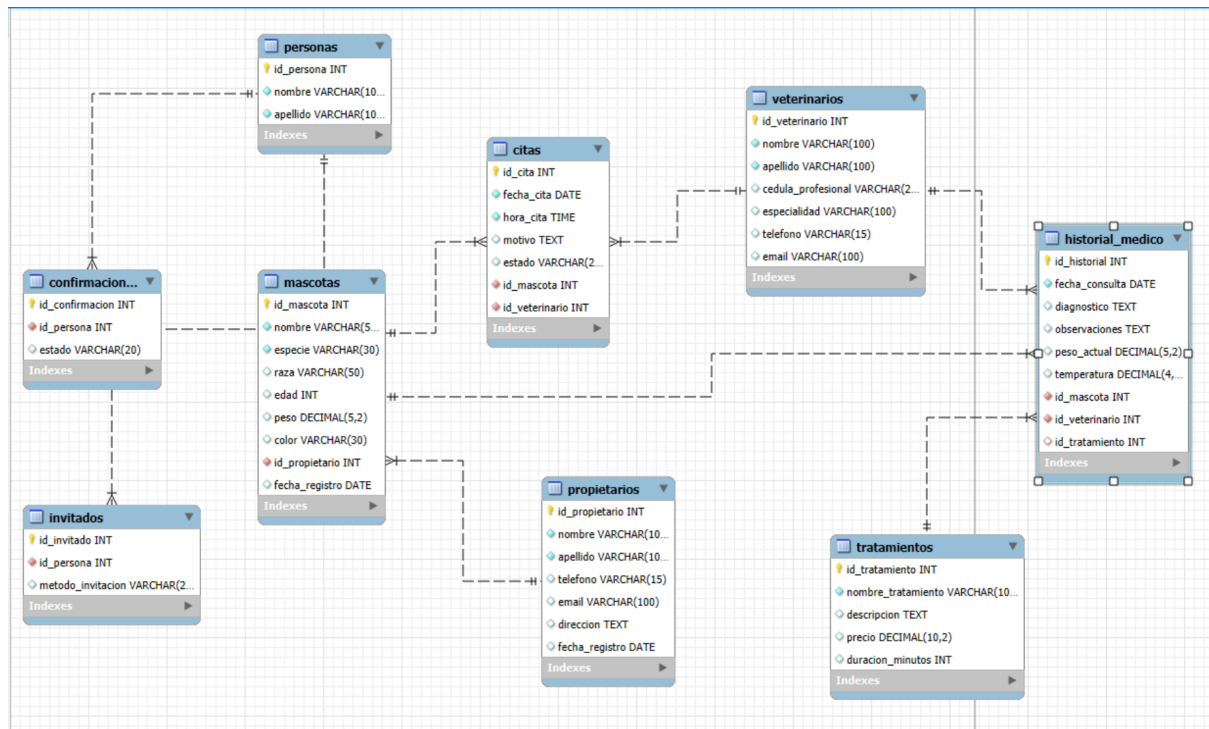
-- Ejecutar procedimiento de validación (Ejercicio 9)
SET @resultado = "";
CALL validar_edad_mascota(3, @resultado);
SELECT @resultado as "Resultado de validación";

```

Resultado de validación
Edad válida



# EER Diagrama



## Liga GitHub

<https://github.com/enderkai00-png/Actividad-4-Base-de-datos/upload>