Instituto Politécnico Nacional

ESCOM

Buscando Huellitas

Licenciatura en Ciencia de Datos

Bases de datos

Integrante:

- Merlin Matoes Diego Eduardo
- Ulises Abiel Cabello Cardenas

Profesor: Velez Saldaña Ulises

Alcance del Proyecto

a) Perfiles de usuario en el sistema

- **Administrador:** Puede gestionar todas las mascotas registradas y administrar usuarios.
- **Usuario Registrado:** Puede reportar, modificar y eliminar sus propios reportes.
- **Visitante:** Puede consultar las mascotas perdidas o encontradas.

b) Funciones de cada usuario

—p4cm—p10cm—

Perfil	Funciones
Administrador	Gestionar todas las mascotas, consultar historial de reportes y administrar usuarios.
Usuario Registrado	Reportar mascota perdida/encontrada, modificar y eliminar sus propios reportes.
Visitante	Consultar mascotas reportadas sin posibilidad de modificar información.

c) Información que debe contemplar el sistema

- **Mascota:** ID, Nombre, Raza, Tamaño, Género, Collar, Descripción, Chip, Foto.
- **Reporte de pérdida:** Fecha, Ubicación, Estado, Ciudad, Calle, Coordenadas GPS.
- **Reporte de hallazgo:** ID de la mascota, Fecha, Ubicación.
- **Usuario (Opcional):** Nombre, Teléfono, CURP, Dirección.

4. Casos de Uso

4.1 Caso de Uso: Registrar una Mascota Perdida

- **Actor:** Usuario Registrado
- **Descripción:** El usuario reporta una mascota perdida mediante un formulario.
- **Flujo:**
  1. Accede a la vista `alta_perdido.ejs` desde el módulo `perdido`.
  2. Completa los datos requeridos: nombre, descripción, raza, ubicación, foto, etc.
  3. Envía el formulario.
  4. El sistema almacena la información en la base de datos y confirma el registro.
- **Excepciones:** Si faltan datos obligatorios, el sistema muestra un mensaje de error.

4.2 Caso de Uso: Consultar Mascotas Reportadas

- **Actor:** Visitante o Usuario Registrado
- **Descripción:** El usuario puede visualizar la lista de mascotas perdidas o encontradas.
- **Flujo:**

1. Accede a la página principal del sistema desde `inicio/index.ejs`.
2. Filtra por criterios como nombre, raza, estado, especie o ubicación.
3. El sistema consulta la base de datos y despliega los resultados.

4.3 Caso de Uso: Modificar Información de una Mascota

- **Actor:** Usuario Registrado
- **Descripción:** El usuario actualiza información previamente registrada de una mascota.
- **Flujo:**
  1. Accede a la vista `editar_alta.ejs` o `editar_especie.ejs` del módulo `alta_perro`.
  2. El sistema carga la información actual del registro.
  3. El usuario modifica los campos necesarios.
  4. Envía el formulario y el sistema guarda los cambios.
- **Excepciones:** Si el ID no corresponde a una mascota registrada, se notifica al usuario.

4.4 Caso de Uso: Eliminar una Mascota Reportada

- **Actor:** Usuario Registrado o Administrador
- **Descripción:** El usuario elimina un registro de mascota que haya creado o el administrador elimina cualquiera.
- **Flujo:**
  1. Accede a la tabla de mascotas desde `tabla.ejs` en `alta_perro` o `albergue`.
  2. Presiona el botón de "Eliminar" junto al registro.
  3. El sistema solicita confirmación.
  4. Si se confirma, se borra el registro de la base de datos.
- **Excepciones:** Si el registro ya no existe, se muestra un mensaje de error.

4.5 Caso de Uso: Gestionar Reportes (Administrador)

- **Actor:** Administrador
- **Descripción:** El administrador puede visualizar, modificar y eliminar cualquier reporte.
- **Flujo:**
  1. Ingresa al panel de administración desde el módulo `albergue` (`tabla.ejs`).
  2. Consulta la lista de reportes activos.

3. Puede editar (`editar.ejs`) o eliminar registros según sea necesario.

Consultas en Álgebra Relacional y SQL

Consulta 1: Perros registrados en el último mes

Álgebra Relacional:

$$\sigma_{nombre\_especie='Canino' \wedge fecha\_nacimiento \geq FECHA\_MES\_PASADO}(mascota \bowtie especie)$$

SQL:

```
SELECT m.*
FROM mascota m
JOIN especie e ON m.id_especie = e.id_especie
WHERE e.nombre_especie = 'Canino'
  AND m.fecha_nacimiento >= CURRENT_DATE - INTERVAL '1 month';
```

Consulta 2: Perros que requieren vacunación próximamente

Álgebra Relacional:

$\pi_{id\_mascota,\ nombre\_mascota}($

$\sigma_{tipo\_medicamento='Vacuna' \wedge\ intervalo\_revacunación \leq 30}((mascota \bowtie especie \bowtie medicamento\_especie$

$\bowtie medicamento \bowtie dosis\_recomendada))$

SQL:

```
SELECT DISTINCT m.*
FROM mascota m
JOIN especie e ON m.id_especie = e.id_especie
JOIN medicamento_especie me ON e.id_especie = me.id_especie
JOIN medicamento md ON me.id_medicamento = md.id_medicamento
JOIN dosis_recomendada dr ON md.id_medicamento = dr.id_medicamento
WHERE md.tipo_medicamento = 'Vacuna'
  AND dr.intervalo_revacunacion <= 30;
```

Consulta 3: Usuarios con perros San Bernardo en Magdalena Contreras

Álgebra Relacional:

$$\pi_{curp,nombre,apellido\_paterno,apellido\_materno}(\sigma_{nombre\_raza='SanBernardo' \wedge alcaldia='MagdalenaContreras'}$$

$$(\text{Usuarios} \bowtie ubicacion \bowtie propietario\_mascota \bowtie mascota \bowtie raza))$$

SQL:

```
SELECT DISTINCT u.curp, u.nombre, u.apellido_paterno, u.apellido_materno
FROM Usuarios u
JOIN ubicacion ub ON u.id_ubicacion = ub.id_ubicacion
JOIN propietario_mascota pm ON u.curp = pm.curp
JOIN mascota m ON pm.id_mascota = m.id_mascota
JOIN raza r ON m.id_raza = r.id_raza
WHERE LOWER(r.nombre_raza) = 'san bernardo'
  AND LOWER(ub.alcaldia) = 'magdalena contreras';
```

Consulta 4: Perros perdidos a menos de 5 km de ESCOM

Álgebra Relacional:

$$\pi_{curp,\ nombre,\ apellido\_paterno,\ apellido\_materno}($$
$$\sigma_{nombre\_raza='SanBernardo' \wedge\ alcaldía='LaMagdalenaContreras'}((Usuarios \bowtie ubicación$$
$$\bowtie propietario\_mascota \bowtie mascota \bowtie raza))$$

SQL:

```
SELECT m.*
FROM mascota m
JOIN especie e ON m.id_especie = e.id_especie
JOIN reporte_mascota_perdida rmp ON m.id_mascota = rmp.id_mascota
JOIN ubicacion u ON rmp.id_ubicacion = u.id_ubicacion
WHERE e.nombre_especie = 'Canino'
  AND sqrt(
        power(u.latitud - 19.50467, 2) +
        power(u.longitud + 99.14679, 2)
      ) * 111.32 < 5;
```

Consulta 5: Avistamientos que coinciden con reportes de perros perdidos

Álgebra Relacional:

$$\rho_{E1}(reporte\_mascota\_encontrada \bowtie mascota \bowtie raza \bowtie ubicacion) \bowtie$$

$\rho_{E2}(reporte\_mascota\_perdida \bowtie mascota \bowtie raza \bowtie ubicacion)$

SQL:

```
SELECT DISTINCT rme.*
FROM reporte_mascota_encontrada rme
JOIN mascota m1 ON rme.id_mascota = m1.id_mascota
JOIN raza r1 ON m1.id_raza = r1.id_raza
JOIN ubicacion u1 ON rme.id_albergue IS NULL AND u1.id_ubicacion IS NOT NULL
JOIN reporte_mascota_perdida rmp ON EXTRACT(MONTH FROM rme.fecha_reporte)
                                  = EXTRACT(MONTH FROM rmp.fecha_reporte)
JOIN mascota m2 ON rmp.id_mascota = m2.id_mascota
JOIN raza r2 ON m2.id_raza = r2.id_raza
JOIN ubicacion u2 ON rmp.id_ubicacion = u2.id_ubicacion

WHERE r1.nombre_raza = r2.nombre_raza
  AND u1.alcaldia = u2.alcaldia
  AND m1.rasgos_distintivos = m2.rasgos_distintivos;
```

Consulta 6: Dueños que han perdido más de un perro en el año

Álgebra Relacional:

$$\pi_{curp}(\gamma_{curp;\ count(id\_mascota)>1}(propietario\_mascota \bowtie reporte\_mascota\_perdida))$$

SQL:

```
SELECT pm.curp
FROM propietario_mascota pm
JOIN reporte_mascota_perdida rmp ON pm.id_mascota = rmp.id_mascota
WHERE EXTRACT(YEAR FROM rmp.fecha_reporte) = EXTRACT(YEAR FROM CURRENT_DATE)
GROUP BY pm.curp
HAVING COUNT(DISTINCT pm.id_mascota) > 1;
```

Consulta 7: Dueños de perros perdidos al menos dos veces

Álgebra Relacional:

$$\pi_{curp}(\gamma_{curp,id\_mascota;\ count(*)\geq 2}(propietario\_mascota \bowtie reporte\_mascota\_perdida))$$

SQL:

```
SELECT pm.curp
FROM propietario_mascota pm
JOIN reporte_mascota_perdida rmp ON pm.id_mascota = rmp.id_mascota
GROUP BY pm.curp, pm.id_mascota
HAVING COUNT(*) >= 2;
```

Consulta 8: Conteo de perros perdidos, avistados, en resguardo y muertos

SQL:

```
SELECT 'Perros perdidos' AS tipo, COUNT(*) AS total
FROM reporte_mascota_perdida
WHERE fecha_reporte >= CURRENT_DATE - INTERVAL '1 month'

UNION ALL

SELECT 'Perros avistados en calle', COUNT(*)
FROM reporte_mascota_encontrada
WHERE lugar_encontrada ILIKE '%calle%'
  AND fecha_reporte >= CURRENT_DATE - INTERVAL '1 month'

UNION ALL

SELECT 'Perros en resguardo', COUNT(*)
FROM reporte_mascota_encontrada
WHERE id_albergue IS NOT NULL
  AND fecha_reporte >= CURRENT_DATE - INTERVAL '1 month'

UNION ALL

SELECT 'Perros muertos', COUNT(*)
FROM estado_mascota
WHERE estado = 'Cancelada'
  AND fecha_actualizacion >= CURRENT_DATE - INTERVAL '1 month';
```

Consulta 9: Colonias con más perros perdidos

Álgebra Relacional:

$$\gamma_{colonia;\ count(*)}(reporte\_mascota\_perdida \bowtie ubicacion)$$

SQL:

```
SELECT u.colonia, COUNT(*) AS total
FROM reporte_mascota_perdida rmp
JOIN ubicacion u ON rmp.id_ubicacion = u.id_ubicacion
GROUP BY u.colonia
ORDER BY total DESC;
```

Consulta 10: Alcaldías con reportes en todas sus colonias

Álgebra Relacional (División):

$$\pi_{alcaldia}(\sigma_{\forall c \in colonias(alcaldia) \rightarrow \exists rmp(c)}(ubicacion))$$

SQL:

```
SELECT u1.alcaldia
FROM ubicacion u1
WHERE NOT EXISTS (
    SELECT 1
    FROM ubicacion u2
    WHERE u2.alcaldia = u1.alcaldia
      AND NOT EXISTS (
          SELECT 1
          FROM reporte_mascota_perdida rmp
          WHERE rmp.id_ubicacion = u2.id_ubicacion
      )
)
GROUP BY u1.alcaldia;
```

Evaluación de la normalización del modelo

Primera Forma Normal (1FN) – Cumple

- Todas las tablas tienen valores atómicos y no hay atributos multivaluados ni compuestos.
- Las columnas son escalares, no listas ni arreglos.
- Todas las relaciones tienen una clave primaria explícita (ej. `id_mascota`, `id_empresa`, etc.).

Segunda Forma Normal (2FN) – Cumple

- Todas las tablas con claves compuestas (ej. `medicamento_especie`, `servicio_veterinaria`) no presentan dependencias parciales:
  - Cada atributo no clave depende de toda la clave primaria compuesta.
- No hay atributos que dependan solo de una parte de la clave compuesta.

Tercera Forma Normal (3FN) – Cumple

- No se encontraron dependencias transitivas en las tablas. Por ejemplo:
  - En `medicamento`, el `id_empresa` no determina ningún otro atributo de `medicamento`, sino que apunta a otra tabla (`empresa`) que contiene `nombre_empresa`.
- Los atributos no clave dependen directamente de la clave primaria.

Cuarta Forma Normal (4FN) – Cumple

- Las relaciones multivaluadas han sido correctamente modeladas como relaciones independientes. Ejemplos claros:
  - Una `mascota` puede tener múltiples fotos → se modela en `foto_mascota`.
  - Un `medicamento` puede usarse para múltiples especies → `medicamento_especie`.
  - Los `usuarios` pueden tener múltiples teléfonos → `teléfonos`.

Quinta Forma Normal (5FN) – Cumple

- La mayoría de las relaciones ternarias o complejas como:
  - `servicio_mascota` (`id_mascota`, `id_servicio`, `id_veterinaria`)
  - `mascota_adopción` (`id_mascota`, `curp_publicador`, `id_ubicación`)
- Para asegurar 5FN se requeriría demostrar que:
  - Al hacer `JOIN` de proyecciones binarias, no se genera pérdida de información ni combinaciones inválidas.

    La Quinta Forma Normal (5FN) es una forma avanzada de normalización que busca eliminar redundancias derivadas de dependencias de unión en relaciones que involucran múltiples claves. Es especialmente relevante en relaciones ternarias o de mayor aridad, donde podrían generarse combinaciones inválidas al descomponer e intentar reconstruir la información.

    Para estas relaciones, se evaluó si al proyectarlas en pares de atributos (proyecciones binarias) y luego realizar la operación inversa con `JOIN`, se conservaba la información original sin introducir tuplas incorrectas. El resultado fue positivo: no se perdió información y no se generaron combinaciones inválidas.

    Esto indica que dichas relaciones cumplen con los principios de la Quinta Forma Normal, aun sin haber sido descompuestas formalmente. Este cumplimiento garantiza:
    - Integridad de los datos al evitar redundancias y anomalías de unión.
    - Mantenimiento eficiente de la base de datos, sin combinaciones artificiales o inconsistentes.
    - Escalabilidad del modelo, facilitando futuras extensiones del esquema sin comprometer su estructura lógica.