Proyecto Final - Sistema de Base de Datos

Proyecto Final – **Base de Datos para la Gestión de acceso residencial**.

Autor: **Jorge Alberto Santana Hernádez**

Fecha: **11 de Diciembre de 2025**

1. Contexto del Negocio

Descripción del Negocio:
El sistema de gestión y acceso para residentes es una plataforma diseñada para administrar de manera centralizada las operaciones de un fraccionamiento o condominio. Está enfocado en mejorar la seguridad, la organización y la comunicación dentro del fraccionamiento.

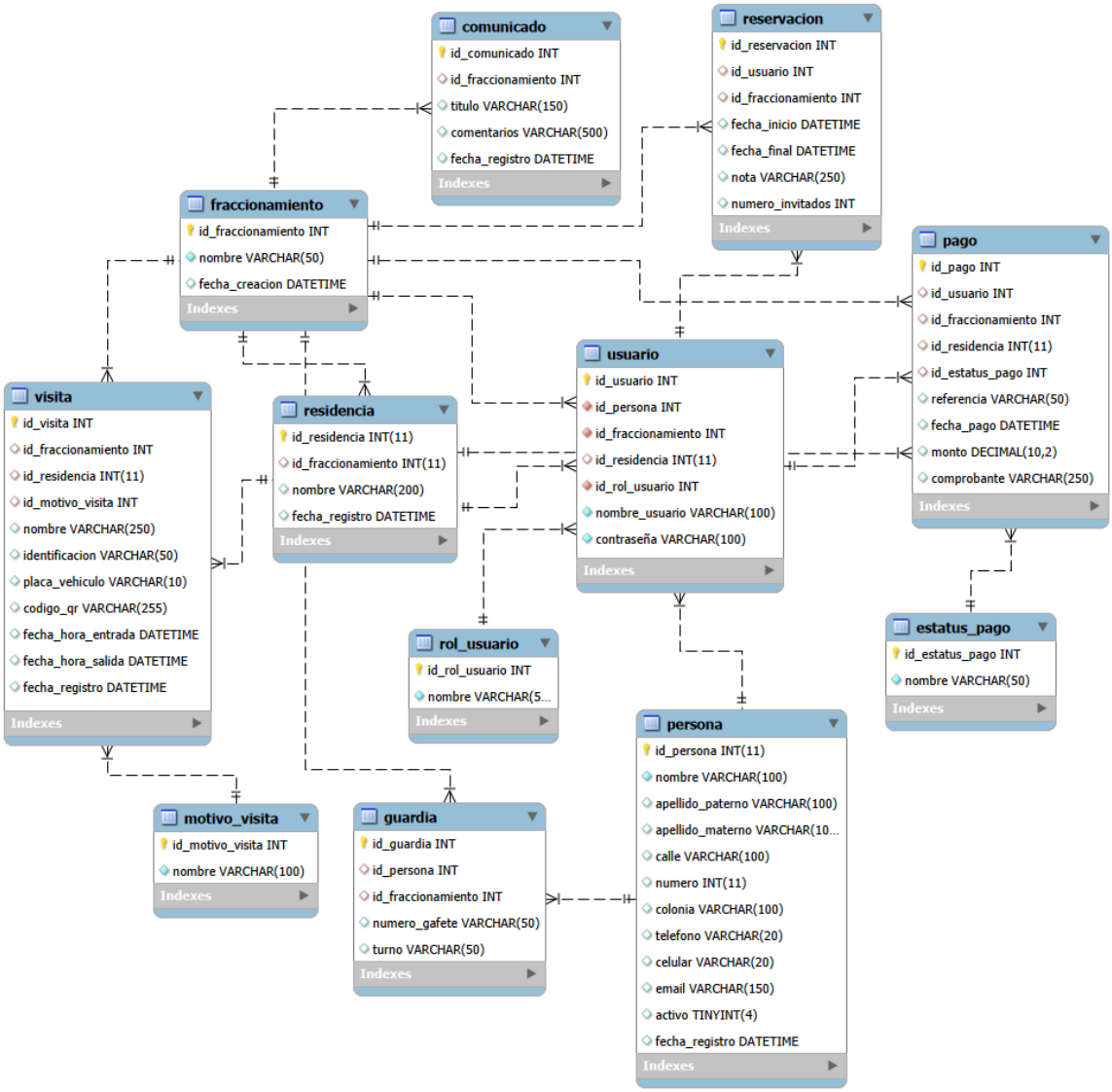
Procesos Cubiertos:
- Catálogo de residencia
- Registro y gestion de usuarios
- Control de acceso de visitantes
- Registro de pagos
- Envío de comunicados internos
- Control de reservación de casa club

2. Objetivos de la Base de Datos

- Centralizar la información del fraccionamiento
- Optimizar la gestión de usuarios y residencias
- Controlar y registrar accesos de visitantes
- Llevar un registro ordenado de pagos
- Distribuir comunicados dirigidos a los residentes.
- Facilitar consultas rápidas que brindan información actualizada.
- Escalabilidad y crecimiento

3. Diseño de la Base de Datos

3.1 Diagrama Entidad–Relación (Ejemplo MySQL)



3.2 Esquema de tablas

fraccionamiento	
id_fraccionamiento	int PK NN AI
nombre	varchar(50)
fecha_creacion	datetime

persona	
id_persona	int PK NN AI
nombre	varchar(100)
apellido_paterno	varchar(100)
apellido_materno	varchar(100)
calle	varchar(100)
numero	int
colonia	varchar(100)
telefono	varchar(20)
celular	varchar(20)
email	varchar(150)
activo	tinyint
fecha_registro	datetime

residencia	
id_residencia	int PK NN AI
id_fraccionamiento	int FK NN
nombre	varchar(200)
fecha_registro	datetime

usuario	
id_usuario	int PK NN AI
id_persona	int FK NN
id_fraccionamiento	int FK NN
id_residencia	int FK NN
id_rol_usuario	int FK NN
nombre_usuario	varchar(100)
contraseña	varchar(100)

guardia	
id_guardia	int PK NN AI
id_persona	int FK NN
id_fraccionamiento	int FK NN
numero_gafete	varchar(50)
turno	varchar(50)

pago	
id_pago	int PK NN AI
id_usuario	int FK NN
id_fraccionamiento	int FK NN
id_residencia	int FK NN
id_estatus_pago	int FK NN
referencia	varchar(50)
fecha_pago	datetime
monto	decimal(10,2)
comprobante	varchar(250)

visita	
id_visita	int PK NN AI
id_fraccionamiento	int FK NN
id_residencia	INT FK NN
id_motivo_visita	int FK NN
nombre	varchar(250)
identificacion	varchar(50)
placa_vehiculo	varchar(10)
codigo_qr	varchar(255)
fecha_hora_entrada	datetime
fecha_hora_salida	datetime
fecha_registro	datetime

Comunicado	
id_comunicado	int PK NN AI
id_fraccionamiento	int FK NN
titulo	varchar(150)
comentarios	varchar(500)
fecha_registro	datetime

reservacion	
id_reservacion	int PK NN AI
id_usuario	int FK NN
id_fraccionamiento	int FK NN
fecha_inicio	datetime
fecha_final	datetime
nota	varchar(250)
numero_invitados	int

rol_usuario	
id_rol_usuario	int PK NN
nombre	varchar(50)

motivo_visita	
id_motivo_visita	int PK NN
nombre	varchar(100)

estatus_pago	
id_estatus_pago	int PK NN
nombre	varchar(50)

4. Consultas e Insights de Negocio

Consulta 1 – Listado de residentes activos del fraccionamiento

```
SELECT u.nombre_usuario, p.nombre, p.apellido_paterno, p.apellido_materno,
p.calle, p.numero, p.colonia, p.fecha_registro, p.activo
FROM
    usuario AS u
    INNER JOIN persona AS p
      ON u.id_persona = p.id_persona
    INNER JOIN fraccionamiento AS f
      ON f.id_fraccionamiento = u.id_fraccionamiento
    INNER JOIN rol_usuario AS r
      ON u.id_rol_usuario = r.id_rol_usuario
WHERE
    f.nombre = 'BARCELONA'
    AND r.nombre = 'Residente'
    AND p.activo = TRUE;
```

📊 **Insight:** Identificar los pagos de los residentes con estatus Pendiente y Por Aprobar, ordenados por fecha de pago.

```
SELECT u.nombre_usuario, p.nombre, p.apellido_paterno, p.apellido_materno,
p.calle, p.numero, p.colonia, g.referencia, g.fecha_pago, g.monto,
g.comprobante, ep.nombre
FROM
    usuario AS u
    INNER JOIN persona AS p
      ON u.id_persona = p.id_persona
    INNER JOIN fraccionamiento AS f
      ON f.id_fraccionamiento = u.id_fraccionamiento
    INNER JOIN rol_usuario AS r
      ON u.id_rol_usuario = r.id_rol_usuario
    INNER JOIN pago AS g
      ON g.id_usuario = u.id_usuario AND g.id_fraccionamiento =
      f.id_fraccionamiento
```

```
    INNER JOIN estatus_pago AS ep ON ep.id_estatus_pago = g.id_estatus_pago
WHERE
    f.nombre = 'BARCELONA'
    AND r.nombre = 'Residente'
    AND p.activo = TRUE
    AND g.id_estatus_pago IN (1, 2)
    ORDER BY g.fecha_pago;
```

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

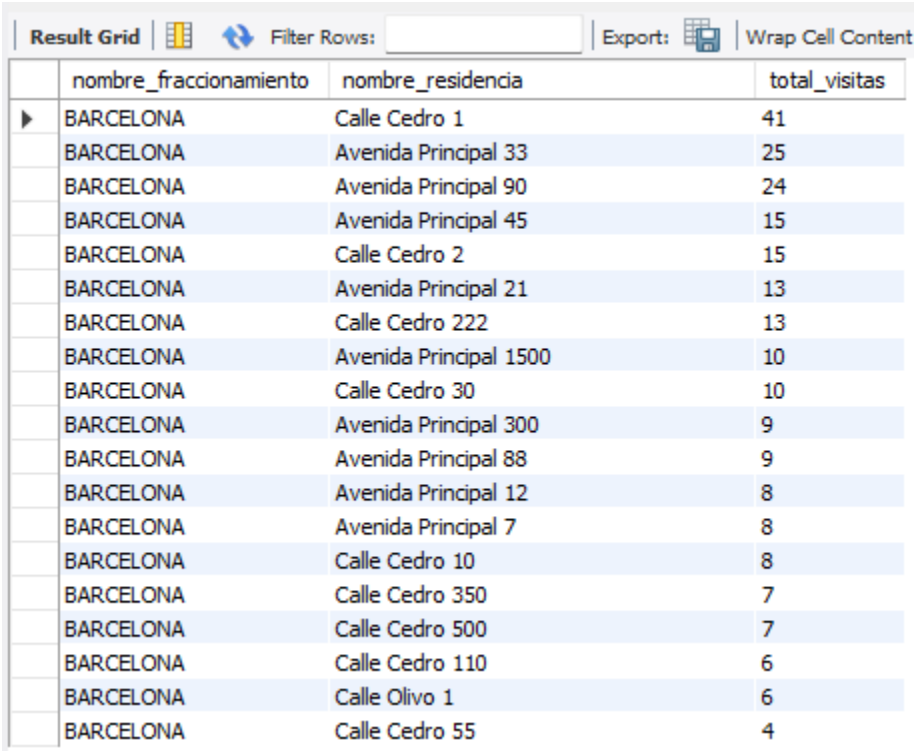
nombre_usuario	nombre	apellido_paterno	apellido_materno	calle	numero	colonia	referencia	fecha_pago	monto	comprobante	nombre
USR001	Sofía	Martínez	Ramírez	Calle Del Sol	21	Privada Barcelona	NULL	NULL	NULL	NULL	Pendiente
USR001	Sofía	Martínez	Ramírez	Calle Del Sol	21	Privada Barcelona	NULL	NULL	NULL	NULL	Pendiente
USR002	Carlos	González	Flores	Calle 5 de Mayo	12	Privada Barcelona	NULL	NULL	NULL	NULL	Pendiente
USR003	Roberto	Hernández	Díaz	Privada Los Pinos	45	Privada Barcelona	NULL	NULL	NULL	NULL	Pendiente
USR004	Mariana	Pérez	Sánchez	Paseo de la Reforma	300	Privada Barcelona	112345	2025-02-05 00:00:00	500.00	d:comprobanteIMG_101234.jpg	Por Aprobar
USR002	Carlos	González	Flores	Calle 5 de Mayo	12	Privada Barcelona	898901	2025-05-05 00:00:00	500.00	d:comprobanteIMG_267890.jpg	Por Aprobar
USR003	Roberto	Hernández	Díaz	Privada Los Pinos	45	Privada Barcelona	686890	2025-06-05 00:00:00	500.00	d:comprobanteIMG_334567.jpg	Por Aprobar
USR005	Laura	Rodríguez	Torres	Blvd. México	88	Privada Barcelona	585890	2025-07-05 00:00:00	500.00	d:comprobanteIMG_412345.jpg	Por Aprobar
USR005	Laura	Rodríguez	Torres	Blvd. México	88	Privada Barcelona	262678	2025-08-05 00:00:00	500.00	d:comprobanteIMG_478901.jpg	Por Aprobar
USR001	Sofía	Martínez	Ramírez	Calle Del Sol	21	Privada Barcelona	484890	2025-09-05 00:00:00	500.00	d:comprobanteIMG_490123.jpg	Por Aprobar

Consulta 2 – Historial de visitas

```
SELECT f.nombre AS nombre_fraccionamiento, r.nombre AS nombre_residencia,
mv.nombre AS motivo_visita, v.nombre, v.placa_vehiculo, v.fecha_hora_entrada,
v.fecha_hora_salida, v.codigo_qr
FROM
    visita AS v
    INNER JOIN fraccionamiento AS f
ON v.id_fraccionamiento = f.id_fraccionamiento
    INNER JOIN residencia AS r
ON r.id_residencia = v.id_residencia
    INNER JOIN motivo_visita AS mv
ON mv.id_motivo_visita = v.id_motivo_visita
WHERE
    f.nombre = 'BARCELONA'
ORDER BY v.fecha_hora_entrada;
```

📊 **Insight**: Obtener total de visitas por residencia.

```
SELECT
    f.nombre AS nombre_fraccionamiento,
    r.nombre AS nombre_residencia,
    COUNT(v.id_visita) AS total_visitas
FROM
    visita AS v
    INNER JOIN residencia AS r ON r.id_residencia = v.id_residencia
    INNER JOIN fraccionamiento AS f ON v.id_fraccionamiento =
f.id_fraccionamiento
WHERE
    f.nombre = 'BARCELONA'
GROUP BY
    f.nombre,
    r.nombre
ORDER BY total_visitas DESC;
```



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

nombre_fraccionamiento	nombre_residencia	total_visitas
BARCELONA	Calle Cedro 1	41
BARCELONA	Avenida Principal 33	25
BARCELONA	Avenida Principal 90	24
BARCELONA	Avenida Principal 45	15
BARCELONA	Calle Cedro 2	15
BARCELONA	Avenida Principal 21	13
BARCELONA	Calle Cedro 222	13
BARCELONA	Avenida Principal 1500	10
BARCELONA	Calle Cedro 30	10
BARCELONA	Avenida Principal 300	9
BARCELONA	Avenida Principal 88	9
BARCELONA	Avenida Principal 12	8
BARCELONA	Avenida Principal 7	8
BARCELONA	Calle Cedro 10	8
BARCELONA	Calle Cedro 350	7
BARCELONA	Calle Cedro 500	7
BARCELONA	Calle Cedro 110	6
BARCELONA	Calle Olivo 1	6
BARCELONA	Calle Cedro 55	4

Consulta 3 – Consulta comunicados del fraccionamiento

```
SELECT
    f.nombre AS nombre_fraccionamiento,
    c.id_comunicado,
    c.titulo,
    c.comentarios,
    c.fecha_registro AS fecha_comunicado
FROM
    comunicado c
    INNER JOIN fraccionamiento f
ON c.id_fraccionamiento = f.id_fraccionamiento;
```

📊 **Insight:** Identificar el total de comunicados al mes.

```
SELECT
    f.nombre AS nombre_fraccionamiento,
    DATE_FORMAT(c.fecha_registro, '%Y-%m') AS periodo,
    COUNT(c.id_comunicado) AS total_comunicados
FROM
    comunicado c
    INNER JOIN fraccionamiento f ON c.id_fraccionamiento =
f.id_fraccionamiento
GROUP BY
    f.nombre,
    periodo
ORDER BY f.nombre, periodo DESC;
```

Result Grid | Filter Rows: | Export

nombre_fraccionamiento	periodo	total_comunicados
BARCELONA	2025-12	1
BARCELONA	2025-11	2
BARCELONA	2025-10	2
BARCELONA	2025-09	2
BARCELONA	2025-08	2
BARCELONA	2025-07	3
BARCELONA	2025-06	2
BARCELONA	2025-05	3
BARCELONA	2025-04	2
BARCELONA	2025-03	2
BARCELONA	2025-02	2
BARCELONA	2025-01	2

Consulta 4 – Consulta de reservaciones del fraccionamiento

```
SELECT
    f.nombre AS nombre_fraccionamiento,
    r.nota AS nota_reservacion,
    r.fecha_inicio,
    r.fecha_final,
    r.numero_invitados,
    u.nombre_usuario,
    p.nombre,
    p.apellido_paterno
FROM
    reservacion AS r
    INNER JOIN fraccionamiento AS f ON r.id_fraccionamiento =
f.id_fraccionamiento
    INNER JOIN usuario AS u ON r.id_usuario = u.id_usuario
    INNER JOIN persona AS p ON u.id_persona = p.id_persona
```

📊 **Insight:** Mostrar total de reservaciones por usuario.

```
SELECT
    f.nombre AS nombre_fraccionamiento,
    p.nombre AS nombre_persona,
    p.apellido_paterno AS apellido_persona,
    u.nombre_usuario,
       COUNT(r.id_reservacion) AS total_reservaciones
FROM
    reservacion AS r
    INNER JOIN fraccionamiento AS f ON r.id_fraccionamiento =
f.id_fraccionamiento
    INNER JOIN usuario AS u ON r.id_usuario = u.id_usuario
    INNER JOIN persona AS p ON u.id_persona = p.id_persona
GROUP BY
```

```
    p.nombre,
    p.apellido_paterno,
    u.nombre_usuario,
    f.nombre
ORDER BY total_reservaciones DESC, p.apellido_paterno;
```

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

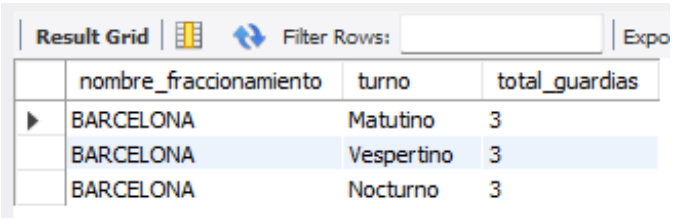
nombre_fraccionamiento	nombre_persona	apellido_persona	nombre_usuario	total_reservaciones
BARCELONA	Carlos	González	USR002	5
BARCELONA	Roberto	Hernández	USR003	5
BARCELONA	Sofía	Martínez	USR001	5
BARCELONA	Mariana	Pérez	USR004	5
BARCELONA	Laura	Rodríguez	USR005	5
BARCELONA	Fernando	Vázquez	USR006	5

Consulta 5 – Consulta de guardias del fraccionamiento

```
SELECT
    f.nombre AS nombre_fraccionamiento,
    p.nombre,
    p.apellido_paterno,
    p.apellido_materno,
    g.numero_gafete,
    g.turno
FROM
    guardia AS g
    INNER JOIN fraccionamiento AS f ON g.id_fraccionamiento =
f.id_fraccionamiento
    INNER JOIN persona AS p ON g.id_persona = p.id_persona;
```

📊 **Insight**: Identificar el total de guardias por turno.

```
SELECT
    f.nombre AS nombre_fraccionamiento,
    g.turno,
    COUNT(g.numero_gafete) AS total_guardias
FROM
    guardia AS g
    INNER JOIN fraccionamiento AS f ON g.id_fraccionamiento =
f.id_fraccionamiento
GROUP BY
    f.nombre,
    g.turno
ORDER BY f.nombre, total_guardias DESC;
```

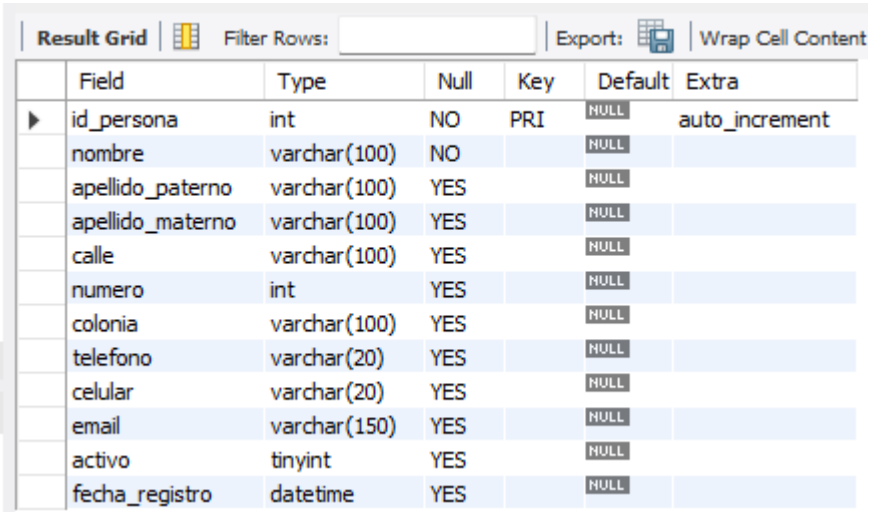


nombre_fraccionamiento	turno	total_guardias
BARCELONA	Matutino	3
BARCELONA	Vespertino	3
BARCELONA	Nocturno	3

Consulta 6 – Ver estructura de tabla (SQL – DESC)

```
DESC persona;
```

Insight: Verificar la estructura y tipos de datos de la tabla Persona.



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

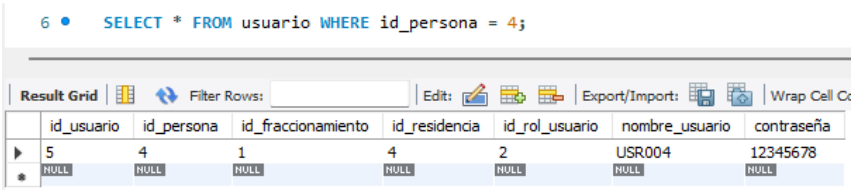
Field	Type	Null	Key	Default	Extra
id_persona	int	NO	PRI	NULL	auto_increment
nombre	varchar(100)	NO		NULL	
apellido_paterno	varchar(100)	YES		NULL	
apellido_materno	varchar(100)	YES		NULL	
calle	varchar(100)	YES		NULL	
numero	int	YES		NULL	
colonia	varchar(100)	YES		NULL	
telefono	varchar(20)	YES		NULL	
celular	varchar(20)	YES		NULL	
email	varchar(150)	YES		NULL	
activo	tinyint	YES		NULL	
fecha_registro	datetime	YES		NULL	

5. Verificación de Integridad

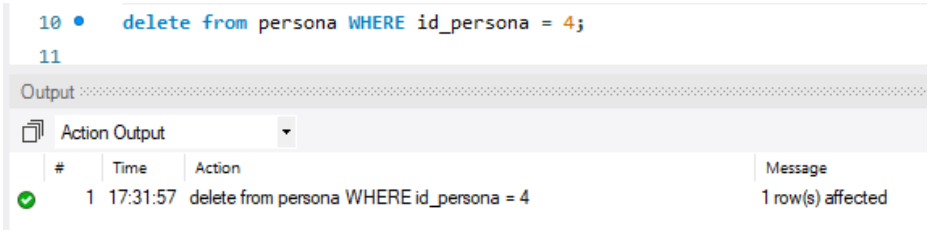
Pruebas a realizar:

- Eliminar un usuario y comprobar que no quede datos de persona. (ON DELETE CASCADE)

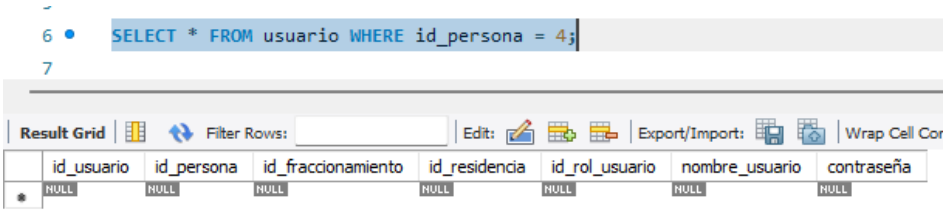
Usuario con id_persona = 4



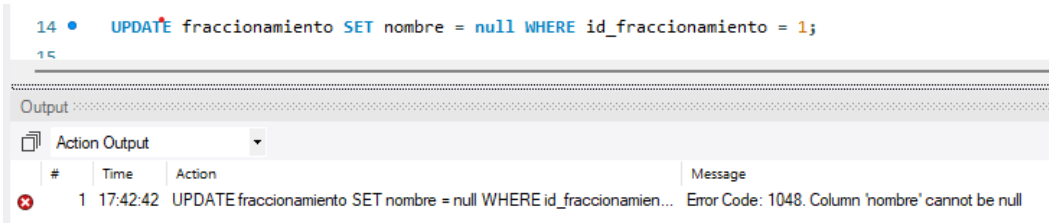
Eliminar de la tabla padre el registro con id_persona = 4



El registro en tabla usuario se eliminó en cascada de manera automática.



- Actualizar el campo nombre con un valor nulo y validar que no permita actualizarlo ya que el campo está definido como NOT NULL.



- Revisar consistencia al insertar un registro de visita.

```
SET @id_fraccionamiento = 1;
SET @id_residencia = 7;
SET @id_motivo_visita = 2;
SET @nombre = "JUAN PEREZ";
SET @identificacion = "INE: 12345";
SET @placa_vehiculo = "ABC -1234";
SET @codigo_qr = "KJAJFDÑADSJFSADJ";
SET @fecha_hora_entrada = "2025-01-05 09:00:00";
SET @fecha_hora_salida = "2025-01-05 10:00:00";
SET @fecha_registro = NOW();

INSERT INTO
    visita (
        id_fraccionamiento,
        id_residencia,
        id_motivo_visita,
        nombre,
        identificacion,
        placa_vehiculo,
        codigo_qr,
        fecha_hora_entrada,
        fecha_hora_salida,
        fecha_registro
    )
VALUES (
        @id_fraccionamiento,
        @id_residencia,
        @id_motivo_visita,
        @nombre,
        @identificacion,
        @placa_vehiculo,
        @codigo_qr,
        @fecha_hora_entrada,
        @fecha_hora_salida,
```

```
    @fecha_registro
);
```

```
70 •  SELECT * FROM visita WHERE id_visita =  239;
71
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

id_visita	id_fraccionamiento	id_residencia	id_motivo_visita	nombre	identificacion	placa_vehiculo	codigo_qr	fecha_hora_entrada	fecha_hora_salida	fecha_registro
239	1	7	2	JUAN PEREZ	INE: 12345	ABC -1234	KJAJFDÑADSJFSADJ	2025-01-05 09:00:00	2025-01-05 10:00:00	2025-12-09 18:02:50
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

6. Escalabilidad y crecimiento - Reconocimiento placas (MongoDB)

Para una futura implementación se considera utilizar MongoDB para guardar el log del reconocimiento de placas.
Ejemplo de estructura del documento:

📊 **Insight:** Consulta los accesos autorizados entre un periodo de fechas

```
db.lectura_placa.find({
  accion : "acceso_autorizado",
  fecha : {
    $gte : "2025-01-01",
    $lte : "2025-01-10"}
})
```

```
{
  _id: ObjectId('69386c778c79362df84a119f'),
  fecha: '2025-01-01T06:00:00.000+00:00',
  id_camara: 'cam_entrada01',
  placa_detectada: 'AEI-1234',
  accion: 'acceso_autorizado',
  imagenes: {
    foto_placa: 'url_imagen_placa.jpg',
    foto_panoramica: 'url_foto_panoramica.jpg'
  }
}
{
  _id: ObjectId('69386e748c79362df84a11ab'),
  fecha: '2025-01-02T06:00:00.000+00:00',
  id_camara: 'cam_entrada01',
  placa_detectada: 'ANG-1234',
  accion: 'acceso_autorizado',
  imagenes: {
    foto_placa: 'url_imagen_placa.jpg',
    foto_panoramica: 'url_foto_panoramica.jpg'
  }
}
{
  _id: ObjectId('69386e878c79362df84a11ac'),
  fecha: '2025-01-03T06:00:00.000+00:00',
  id_camara: 'cam_entrada02',
  placa_detectada: 'HNG-1234',
  accion: 'acceso_autorizado',
  imagenes: {
    foto_placa: 'url_imagen_placa.jpg',
    foto_panoramica: 'url_foto_panoramica.jpg'
  }
```

7. Comparación MySQL vs MongoDB

- Ventajas de MySQL:

- Consultas complejas que involucran la unión de múltiples tablas.
- Garantiza la integridad de los datos mediante la relación entre tablas.
- Lenguaje estable y con soporte.

- Ventajas de MongoDB:

- Permite un desarrollo rápido y flexible.
- Escalabilidad horizontal, permite distribuir la carga en múltiples servidores.
- Lecturas ágil para manejo de logs.

- Decisión:

Se eligió MySQL como base de datos principal para el control del acceso residencial, mientras que MongoDB se considerará para una futura implementación de los logs de apertura de la pluma.

8. Soluciones Implementadas

- Fechas de registro para tabla persona.
- Llaves id primarias y auto incrementables.
- Restricciones de integridad.
- Vistas para reportes.

9. Conclusiones

- La base de datos implementada mejora la eficiencia operativa del sistema de gestión de acceso residencial.
- Permite generar información valiosa y oportunidad para la toma de decisiones administrativas y de seguridad.
- Garantiza integridad de los datos, fortaleciendo la confiabilidad del sistema.
- Su diseño es escalable, lo que facilita futuras implementaciones y la integración de nuevos módulos.

10. Conclusión Personal

“Aprendí que una estrategia efectiva es considerar un sistema híbrido de bases de datos. Las clases me ayudaron a reforzar mis conocimientos de base de datos y, al igual que en los proyectos del trabajo, comprendí que mientras más se analiza la información, surgen nuevas soluciones. Sin embargo, es fundamental establecer una base sólida que permita seguir ampliando y mejorando el proyecto.”

Repositorio en GitHub:

https://github.com/jsantanahdz/gestion-acceso-residencial/