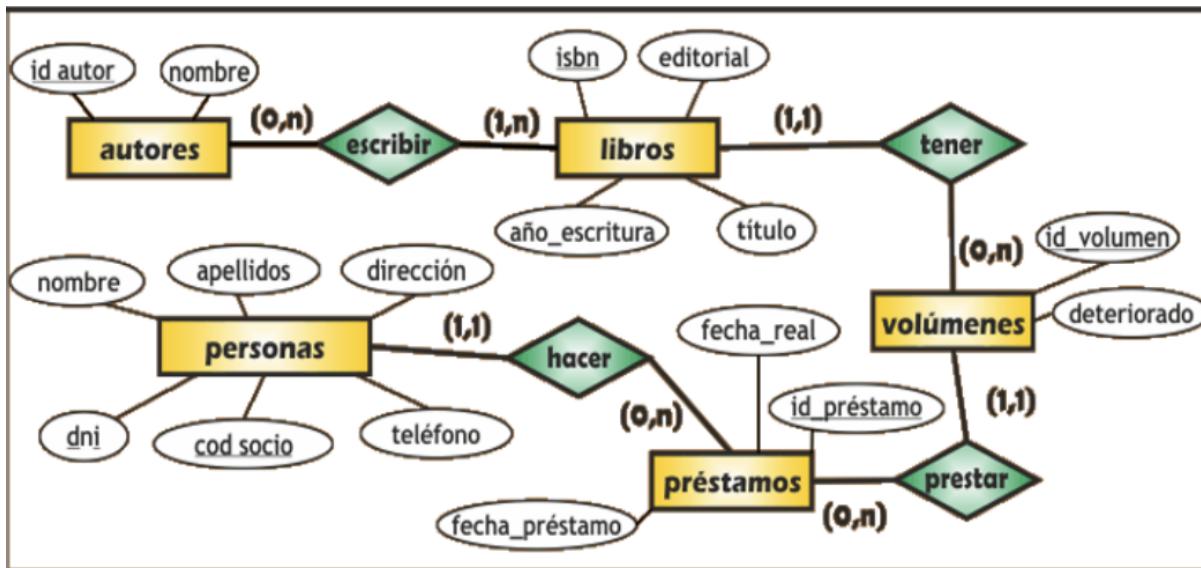


Ejercicio 1. Biblioteca v1



Personas(cod_socio:number,dni:string,nombre:string,apellidos:string,direccion:string,telefono:string)

Prestamos(id:number,fecha_real:date,fecha_prestamo:date,persona:id_persona(FK),volume n:id_volumen(FK))

Volumenes(id:number,deteriorado:boolean,libro:id_libro(FK))

Libros(isbn:number,editorial:string,título:string,año_escritura:number)

Autores(id:number,nombre:string)

autores_libros(autor:id_autor,libro:id_libro(FK))

DDL y DML

```
create database ex1;
```

```
use ex1;
```

```
CREATE TABLE Personas (
    cod_socio int auto_increment primary key,
    dni VARCHAR(10) UNIQUE NOT NULL,
    nombre VARCHAR(100) NOT NULL,
    apellidos VARCHAR(100) NOT NULL,
    direccion VARCHAR(200),
    telefono VARCHAR(20)
);
```

```
CREATE TABLE Libros (
    isbn varchar(30) PRIMARY KEY,
    editorial VARCHAR(100) NOT NULL,
    título VARCHAR(200) NOT NULL,
    año_escritura int
);
```

```
CREATE TABLE Autores (
    id int auto_increment PRIMARY KEY,
```

```

    nombre VARCHAR(100) NOT NULL
);

CREATE TABLE autores_libros (
    autor int,
    libro varchar(30),
    PRIMARY KEY (autor, libro),
    FOREIGN KEY (autor) REFERENCES Autores(id)
    on delete cascade
    on update cascade,
    FOREIGN KEY (libro) REFERENCES Libros(isbn)
    on delete cascade
    on update cascade
);
;

CREATE TABLE Volumenes (
    id int auto_increment PRIMARY KEY,
    deteriorado BOOLEAN,
    libro varchar(30),
    FOREIGN KEY (libro) REFERENCES Libros(isbn)
    on delete cascade
    on update cascade
);
;

CREATE TABLE Prestamos (
    id int auto_increment PRIMARY KEY,
    fecha_real DATE,
    fecha_prestamo DATE,
    persona int,
    volumen int,
    FOREIGN KEY (persona) REFERENCES Personas(cod_socio)
    on delete cascade
    on update cascade,
    FOREIGN KEY (volumen) REFERENCES Volumenes(id)
    on delete cascade
    on update cascade
);
;

-- Inserción de datos en la tabla Personas
INSERT INTO Personas (cod_socio, dni, nombre, apellidos, direccion, telefono)
VALUES (1, '12345678A', 'Juan', 'Perez', 'Calle Mayor 123', '555-1234');

-- Inserción de datos en la tabla Libros
INSERT INTO Libros (isbn, editorial, titulo, ano_escritura)
VALUES ('978-0123456789', 'Editorial X', 'Libro de Prueba', 2023);

-- Inserción de datos en la tabla Autores
INSERT INTO Autores (id, nombre)
VALUES (1, 'Autor Prueba');

-- Inserción de datos en la tabla autores_libros
INSERT INTO autores_libros (autor, libro)

```

```

VALUES (1, '978-0123456789');

-- Inserción de datos en la tabla Volumenes
INSERT INTO Volumenes (id, deteriorado, libro)
VALUES (1, FALSE, '978-0123456789');

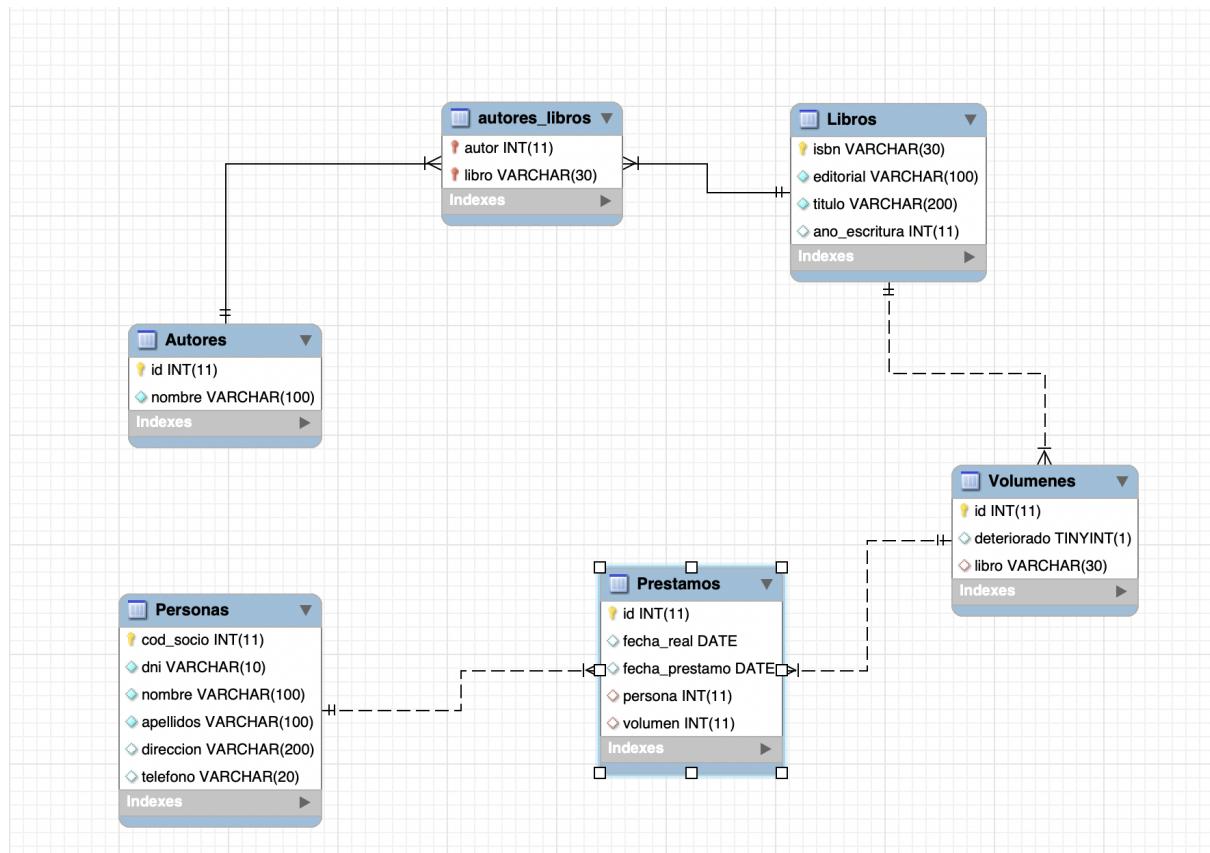
-- Inserción de datos en la tabla Prestamos
INSERT INTO Prestamos (id, fecha_real, fecha_prestamo, persona, volumen)
VALUES (1, '2023-07-28', '2023-07-20', 1, 1);

-- Modificación de un préstamo
UPDATE Prestamos
SET fecha_real = '2023-07-28'
WHERE id = 1;

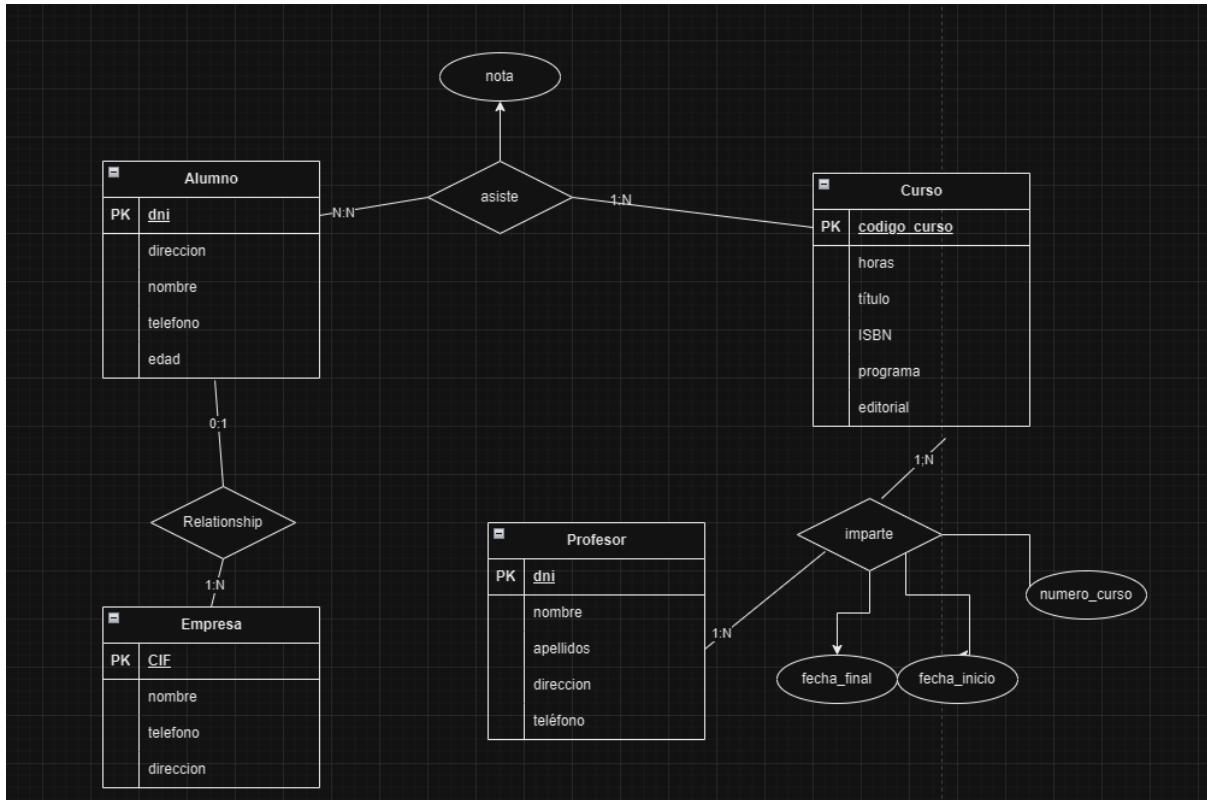
-- Eliminación de un autor y su relación en autores_libros
DELETE FROM autores_libros
WHERE autor = 1;

DELETE FROM Autores
WHERE id = 1;

```



Ejercicio 2



Empresa(cif: string, nombre: string, telefono: string, direccion: string)

Alumno(dni: string, direccion: string, nombre: string, telefono: string, edad: number, empresa: string (FK))

- *empresa es FK que hace referencia a Empresa(CIF)*

Curso(codigo_curso: number, programa: string, horas: date, titulo: string, editorial: string, ISBN: string)

Asistencia(dni_alumno_FK:string, codigo_curso_FK: string, nota: number)

- *Dni_alumno y codigo_curso son a su vez FK que hacen referencia a Curso(codigo_curso) y Alumno(dni).*

Profesor(dni:number, nombre: string, apellidos: string, direccion: string, telefono: string)

Imparte(num_curso: number, dni_profesor_FK:string, codigo_curso_FK:number, fecha_inicio: date, fecha_final: date)

- *Dni_profesor es FK que hace referencia a professor(dni)*
- *Codigo_curso es FK que hace referencia a Curso(codigo_curso)*

DDL y DML

```
create database ex2;
```

```
use ex2;
```

```
CREATE TABLE Empresa (
    cif VARCHAR(20) PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    telefono VARCHAR(20),
    direccion VARCHAR(200)
);
```

```
CREATE TABLE Alumno (
    dni VARCHAR(20) PRIMARY KEY,
    direccion VARCHAR(200),
    nombre VARCHAR(100) NOT NULL,
    telefono VARCHAR(20),
    edad int,
    empresa VARCHAR(20),
    FOREIGN KEY (empresa) REFERENCES Empresa(cif)
    on delete cascade
    on update cascade
);
```

```
CREATE TABLE Curso (
    codigo_curso int auto_increment PRIMARY KEY,
    programa VARCHAR(200),
    horas DATE,
    titulo VARCHAR(200),
    editorial VARCHAR(100),
    ISBN VARCHAR(20)
);
```

```
CREATE TABLE Asistencia (
    dni_alumno_FK VARCHAR(20),
    codigo_curso_FK int,
    nota int,
    PRIMARY KEY (dni_alumno_FK, codigo_curso_FK),
    FOREIGN KEY (dni_alumno_FK) REFERENCES Alumno(dni)
    on delete cascade
    on update cascade,
    FOREIGN KEY (codigo_curso_FK) REFERENCES Curso(codigo_curso)
    on delete cascade
    on update cascade
);
```

```
CREATE TABLE Profesor (
    dni VARCHAR(20) PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    apellidos VARCHAR(100) NOT NULL,
    direccion VARCHAR(200),
    telefono VARCHAR(20)
);
```

```

CREATE TABLE Imparte (
    num_curso int auto_increment,
    dni_profesor_FK VARCHAR(20),
    codigo_curso_FK int,
    fecha_inicio DATE,
    fecha_final DATE,
    primary key(num_curso),
    FOREIGN KEY (dni_profesor_FK) REFERENCES Profesor(dni)
    on delete cascade
    on update cascade,
    FOREIGN KEY (codigo_curso_FK) REFERENCES Curso(codigo_curso)
    on delete cascade
    on update cascade
);
-- Inserción de datos en la tabla Empresa
INSERT INTO Empresa (cif, nombre, telefono, direccion)
VALUES ('A12345678', 'Empresa 1', '555-1234', 'Calle Principal 123');

-- Inserción de datos en la tabla Alumno
INSERT INTO Alumno (dni, direccion, nombre, telefono, edad, empresa)
VALUES ('11111111A', 'Calle Secundaria 456', 'Juan Pérez', '555-5678', 25, 'A12345678');

-- Inserción de datos en la tabla Curso
INSERT INTO Curso (programa, horas, titulo, editorial, ISBN)
VALUES ('Programa Curso 1', '2023-07-28', 'Título del Curso 1', 'Editorial X', '978-0123456789');

-- Inserción de datos en la tabla Asistencia
INSERT INTO Asistencia (dni_alumno_FK, codigo_curso_FK, nota)
VALUES ('11111111A', 1, 8);

-- Inserción de datos en la tabla Profesor
INSERT INTO Profesor (dni, nombre, apellidos, direccion, telefono)
VALUES ('22222222B', 'Profesor', 'García', 'Calle Maestra 789', '555-9876');

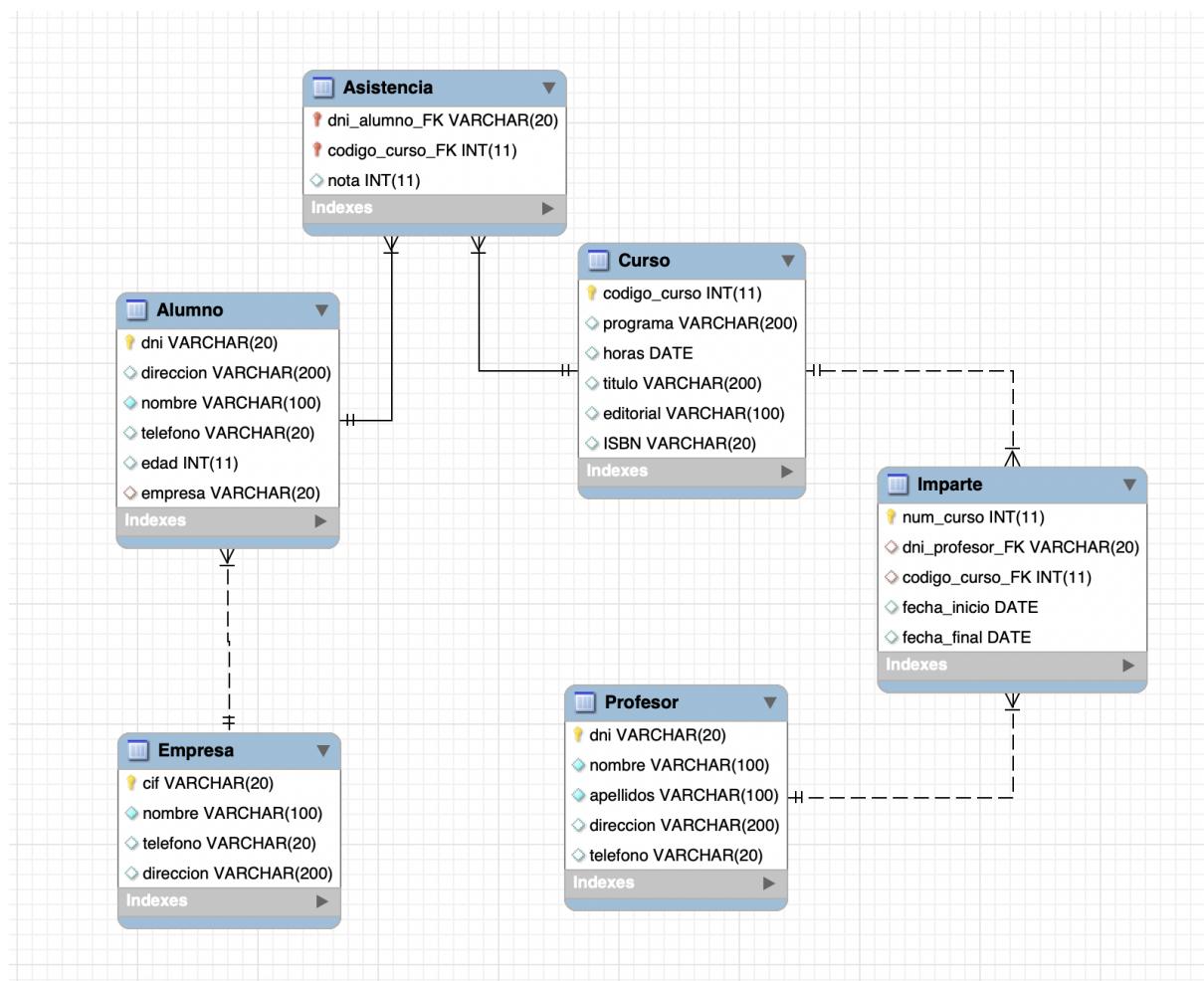
-- Inserción de datos en la tabla Imparte
INSERT INTO Imparte (dni_profesor_FK, codigo_curso_FK, fecha_inicio, fecha_final)
VALUES ('22222222B', 1, '2023-07-01', '2023-07-31');

-- Actualizar la edad del alumno con DNI '11111111A'
UPDATE Alumno
SET edad = 26
WHERE dni = '11111111A';

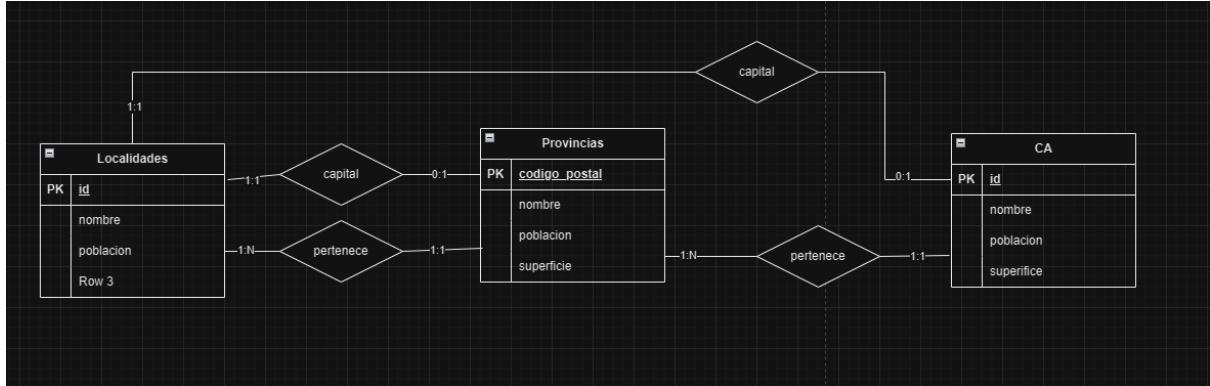
-- Eliminar la empresa con CIF 'A12345678'
DELETE FROM Empresa WHERE cif = 'A12345678';

-- Asignar al profesor con DNI '33333333C' al curso con código 1 en la tabla Imparte
UPDATE Profesor
SET nombre = 'Nuevo nombre'
WHERE dni = '22222222B';

```



Ejercicio 3



Localidades(**id**: number, nombre: string, poblacion: number)

Provincias(**codigo_postal**: number, nombre: string, poblacion: number, superficie: number, **id_capital**: number(FK))

- *id_capital es FK que hace referencia a Localidades(id)*

Pertenencia_localidad_provincia(**id_localidad**: number(FK), **codigo_postal_provincia**: number(FK))

- *Id_localidad es FK que hace referencia a Localidades(id) y codigo_postal_provincia es FK que hace referencia a Provincias(codigo_postal)*

Comunidades_autonomas(**id**: number, nombre: string, poblacion: number, superficie: number, **id_capital**: number(FK))

- *id_capital es FK que hace referencia a Localidades(id)*

Pertenencia_provincia_CA(**codigo_postal_provincia**: number(FK), **id_CA**: number(FK))

- *codigo_postal_provincia es FK que hace referencia a Provincias(codigo_postal)*
- *id_CA es FK que hace referencia a Comunidades_autonomas(id)*

DDL y DML

```
create database ex3;
use ex3;
```

```
CREATE TABLE Localidades (
    id int auto_increment PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    poblacion int
);
```

```
CREATE TABLE Provincias (
    codigo_postal int PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    poblacion int,
    superficie int,
    id_capital int,
    FOREIGN KEY (id_capital) REFERENCES Localidades(id)
    on delete cascade
    on update cascade
);
```

```
CREATE TABLE Pertenencia_localidad_provincia (
    id_localidad int,
    codigo_postal_provincia int,
    PRIMARY KEY (id_localidad, codigo_postal_provincia),
    FOREIGN KEY (id_localidad) REFERENCES Localidades(id)
    on delete cascade
    on update cascade,
    FOREIGN KEY (codigo_postal_provincia) REFERENCES Provincias(codigo_postal)
    on delete cascade
    on update cascade
);
```

```
CREATE TABLE Comunidades_autonomas (
    id int auto_increment PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    poblacion int,
    superficie int,
    id_capital int,
    FOREIGN KEY (id_capital) REFERENCES Localidades(id)
    on delete cascade
    on update cascade
);
```

```
CREATE TABLE Pertenencia_provincia_CA (
    codigo_postal_provincia int,
    id_CA int,
    PRIMARY KEY (codigo_postal_provincia, id_CA),
    FOREIGN KEY (codigo_postal_provincia) REFERENCES Provincias(codigo_postal)
    on delete cascade
    on update cascade,
    FOREIGN KEY (id_CA) REFERENCES Comunidades_autonomas(id)
```

```

on delete cascade
on update cascade
);

-- Inserción de datos en la tabla Localidades
INSERT INTO Localidades (id, nombre, poblacion)
VALUES (1, 'Localidad 1', 10000);

-- Inserción de datos en la tabla Provincias
INSERT INTO Provincias (codigo_postal, nombre, poblacion, superficie, id_capital)
VALUES (12345, 'Provincia 1', 500000, 10000, 1);

-- Inserción de datos en la tabla Pertenencia_localidad_provincia
INSERT INTO Pertenencia_localidad_provincia (id_localidad, codigo_postal_provincia)
VALUES (1, 12345);

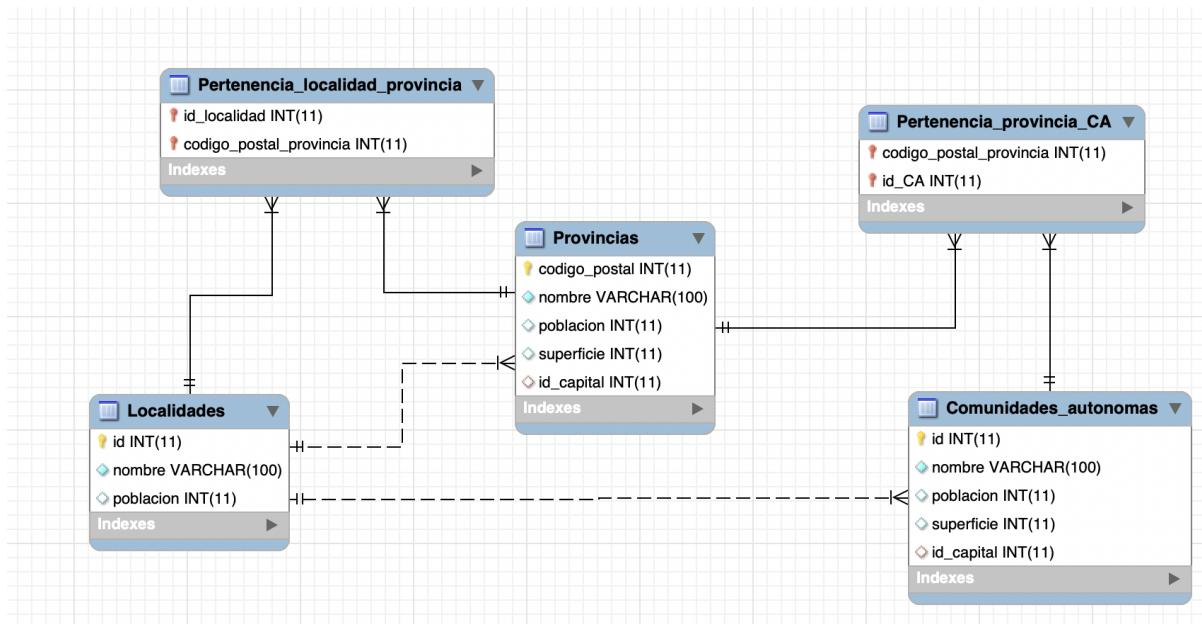
-- Inserción de datos en la tabla Comunidades_autonomas
INSERT INTO Comunidades_autonomas (id, nombre, poblacion, superficie, id_capital)
VALUES (101, 'Comunidad Autónoma 1', 1000000, 50000, 1);

-- Inserción de datos en la tabla Pertenencia_provincia_CA
INSERT INTO Pertenencia_provincia_CA (codigo_postal_provincia, id_CA)
VALUES (12345, 101);

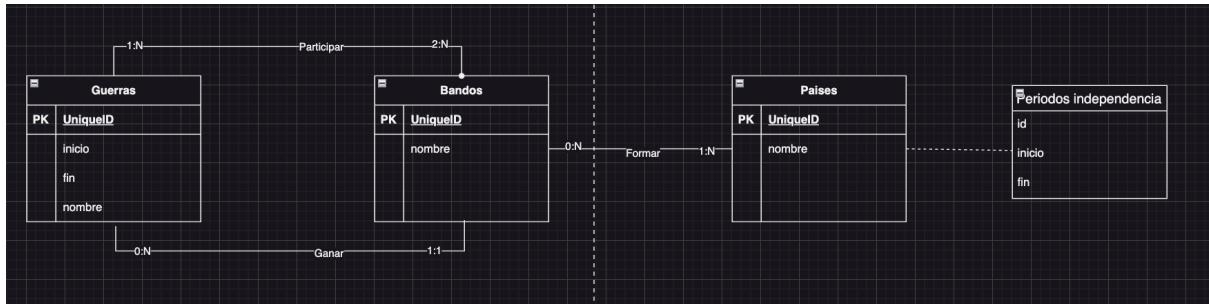
-- Actualizar el nombre de la localidad con ID 1
UPDATE Localidades
SET nombre = 'Nueva Localidad'
WHERE id = 1;

-- Eliminar la provincia con código postal 12345 y su pertenencia en Pertenencia_localidad_provincia
DELETE FROM Pertenencia_localidad_provincia WHERE codigo_postal_provincia = 12345;
DELETE FROM Provincias WHERE codigo_postal = 12345;

```



Ejercicio 4. Guerras



Bandos(**id**:number, nombre:string)

Guerras(**id**:number, inicio:number, fin:number, nombre:string, ganador:**id_bando**(FK))

Participaciones(**guerra**:**id_guerra**(FK), **bando**:**id_bando**(FK))

Paises(**id**:number, nombre:string)

Bandos_paises(**bando**:**id_bando**(FK),**pais**:**id_paises**(FK))

Periodos_independencia(**pais**:**id_paises**(FK), inicio:number, fin:number)

DDL y DML

```
create database ex4;
```

```
use ex4;
```

```
CREATE TABLE Bandos (
    id int auto_increment PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL
);
```

```
CREATE TABLE Guerras (
    id int auto_increment PRIMARY KEY,
    inicio int,
    fin int,
    nombre VARCHAR(100) NOT NULL,
    ganador int,
    FOREIGN KEY (ganador) REFERENCES Bandos(id)
    on delete cascade
    on update cascade
);
```

```
CREATE TABLE Participaciones (
    guerra int,
    bando int,
    PRIMARY KEY (guerra, bando),
    FOREIGN KEY (guerra) REFERENCES Guerras(id)
    on delete cascade
    on update cascade,
    FOREIGN KEY (bando) REFERENCES Bandos(id)
    on delete cascade
    on update cascade
);
```

```
CREATE TABLE Paises (
```

```

    id int auto_increment PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL
);

CREATE TABLE Bandos_paises (
    bando int,
    pais int,
    PRIMARY KEY (bando, pais),
    FOREIGN KEY (bando) REFERENCES Bandos(id)
    on delete cascade
    on update cascade,
    FOREIGN KEY (pais) REFERENCES Paises(id)
    on delete cascade
    on update cascade
);

CREATE TABLE Periodos_independencia (
    pais int,
    inicio int,
    fin int,
    PRIMARY KEY (pais, inicio),
    FOREIGN KEY (pais) REFERENCES Paises(id)
    on delete cascade
    on update cascade
);

-- Inserción de datos en la tabla Bandos
INSERT INTO Bandos (id, nombre)
VALUES (1, 'Bando 1'),
       (2, 'Bando 2');

-- Inserción de datos en la tabla Guerras
INSERT INTO Guerras (id, inicio, fin, nombre, ganador)
VALUES (1, 1900, 1910, 'Guerra 1', 1),
       (2, 1950, 1960, 'Guerra 2', 2);

-- Inserción de datos en la tabla Participaciones
INSERT INTO Participaciones (guerra, bando)
VALUES (1, 1),
       (1, 2),
       (2, 2);

-- Inserción de datos en la tabla Paises
INSERT INTO Paises (id, nombre)
VALUES (1, 'Pais 1'),
       (2, 'Pais 2');

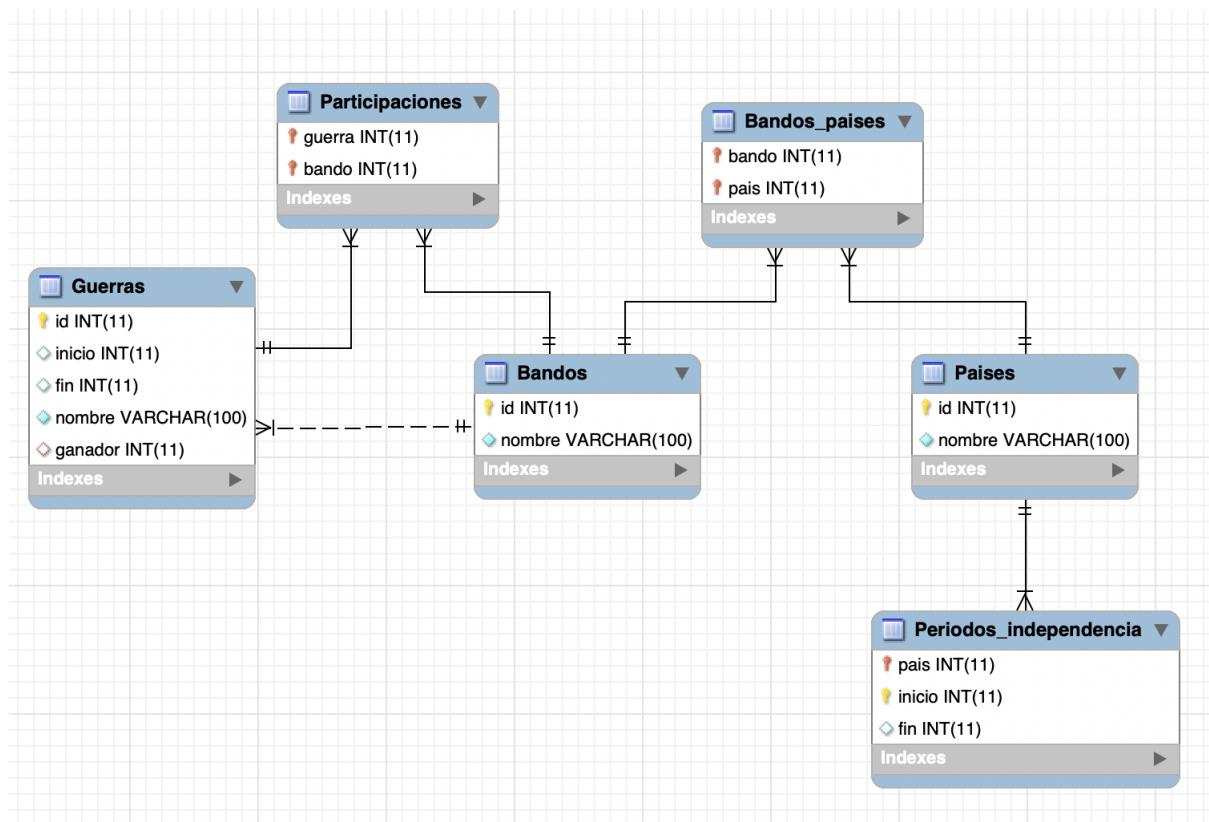
-- Inserción de datos en la tabla Bandos_paises
INSERT INTO Bandos_paises (bando, pais)
VALUES (1, 1),
       (2, 2);

```

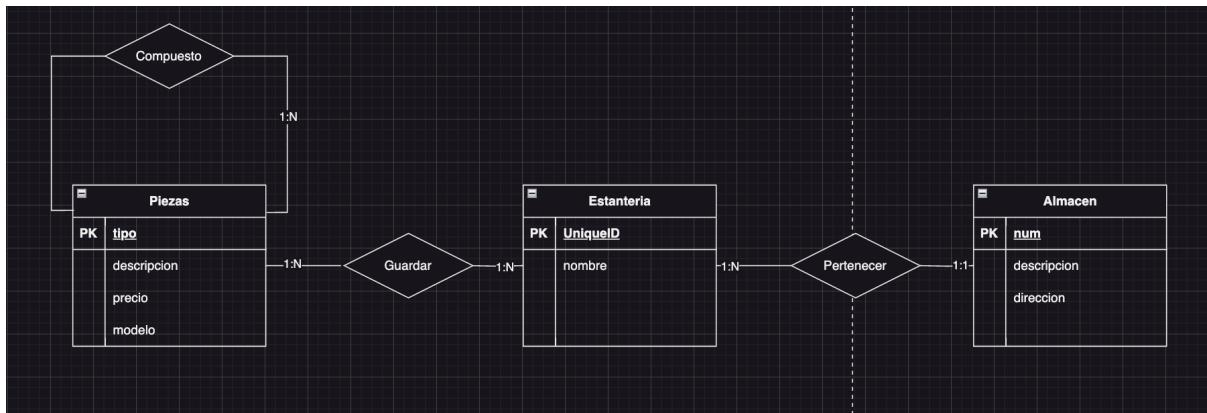
```
-- Inserción de datos en la tabla Periodos_independencia
INSERT INTO Periodos_independencia (pais, inicio, fin)
VALUES (1, 1800, 1900),
(2, 1850, 1950);
```

```
-- Ejemplo de UPDATE: Actualizar el nombre de un bando en la tabla Bandos
UPDATE Bandos
SET nombre = 'Nuevo Bando'
WHERE id = 2;
```

```
-- Ejemplo de DELETE: Eliminar un país y sus períodos de independencia en las tablas Paises y
Periodos_independencia
DELETE FROM Periodos_independencia WHERE pais = 2;
DELETE FROM Bandos_paises WHERE pais = 2;
DELETE FROM Paises WHERE id = 2;
```



Ejercicio 5. Almacén v1



Pieza(tipo:string, descripcion:string, precio:number, modelo:string)

Pieza_compuesta(pieza1:id_piezas(FK), pieza2:id_piezas(FK))

Almacenes(num:number, descripcion:string, direccion:string)

Estanterias(id:number, nombre:string, id_almacen:id_almacenes(FK))

Pieza_estanteria(pieza:id_piezas(FK),estanteria:id_estanterias(FK))

DDL y DML

CREATE DATABASE ex5;

USE ex5;

```

CREATE TABLE Pieza (
    tipo VARCHAR(100),
    descripcion VARCHAR(200),
    precio int,
    modelo VARCHAR(100),
    PRIMARY KEY (tipo)
);
  
```

```

CREATE TABLE Pieza_compuesta (
    pieza1 VARCHAR(100),
    pieza2 VARCHAR(100),
    PRIMARY KEY (pieza1, pieza2),
    FOREIGN KEY (pieza1) REFERENCES Pieza(tipo)
    on delete cascade
    on update cascade,
    FOREIGN KEY (pieza2) REFERENCES Pieza(tipo)
    on delete cascade
    on update cascade
);
  
```

```

CREATE TABLE Almacenes (
    num int auto_increment PRIMARY KEY,
    descripcion VARCHAR(200),
    direccion VARCHAR(200)
);
  
```

```

CREATE TABLE Estanterias (
    id int auto_increment PRIMARY KEY,
    nombre VARCHAR(100),
    id_almacen int,
    FOREIGN KEY (id_almacen) REFERENCES Almacenes(num)
    on delete cascade
    on update cascade
);
;

CREATE TABLE Pieza_estanteria (
    pieza VARCHAR(100),
    estanteria int,
    PRIMARY KEY (pieza, estanteria),
    FOREIGN KEY (pieza) REFERENCES Pieza(tipo)
    on delete cascade
    on update cascade,
    FOREIGN KEY (estanteria) REFERENCES Estanterias(id)
    on delete cascade
    on update cascade
);
;

-- Inserción de datos en la tabla Pieza
INSERT INTO Pieza (tipo, descripcion, precio, modelo)
VALUES ('Tuerca', 'Tuerca para ensamblaje', 0.50, 'ABC123'),
       ('Perno', 'Perno de acero', 0.75, 'XYZ456');

-- Inserción de datos en la tabla Pieza_compuesta
INSERT INTO Pieza_compuesta (pieza1, pieza2)
VALUES ('Tuerca', 'Perno');

-- Inserción de datos en la tabla Almacenes
INSERT INTO Almacenes (num, descripcion, direccion)
VALUES (1, 'Almacén principal', 'Calle Mayor 123'),
       (2, 'Almacén secundario', 'Calle Secundaria 456');

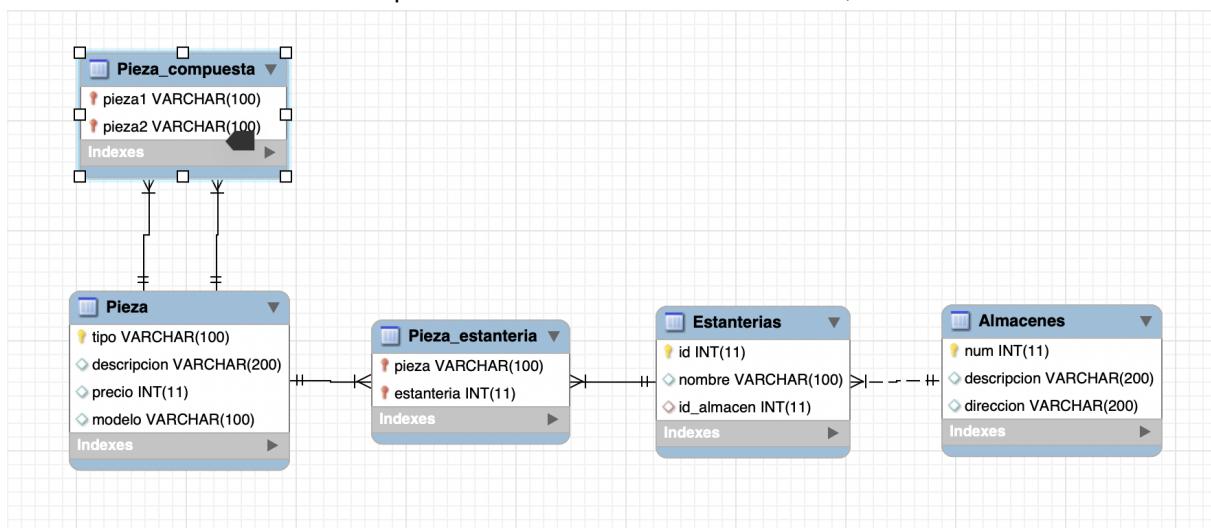
-- Inserción de datos en la tabla Estanterias
INSERT INTO Estanterias (id, nombre, id_almacen)
VALUES (101, 'Estantería 1', 1),
       (102, 'Estantería 2', 2);

-- Inserción de datos en la tabla Pieza_estanteria
INSERT INTO Pieza_estanteria (pieza, estanteria)
VALUES ('Tuerca', 101),
       ('Perno', 102);

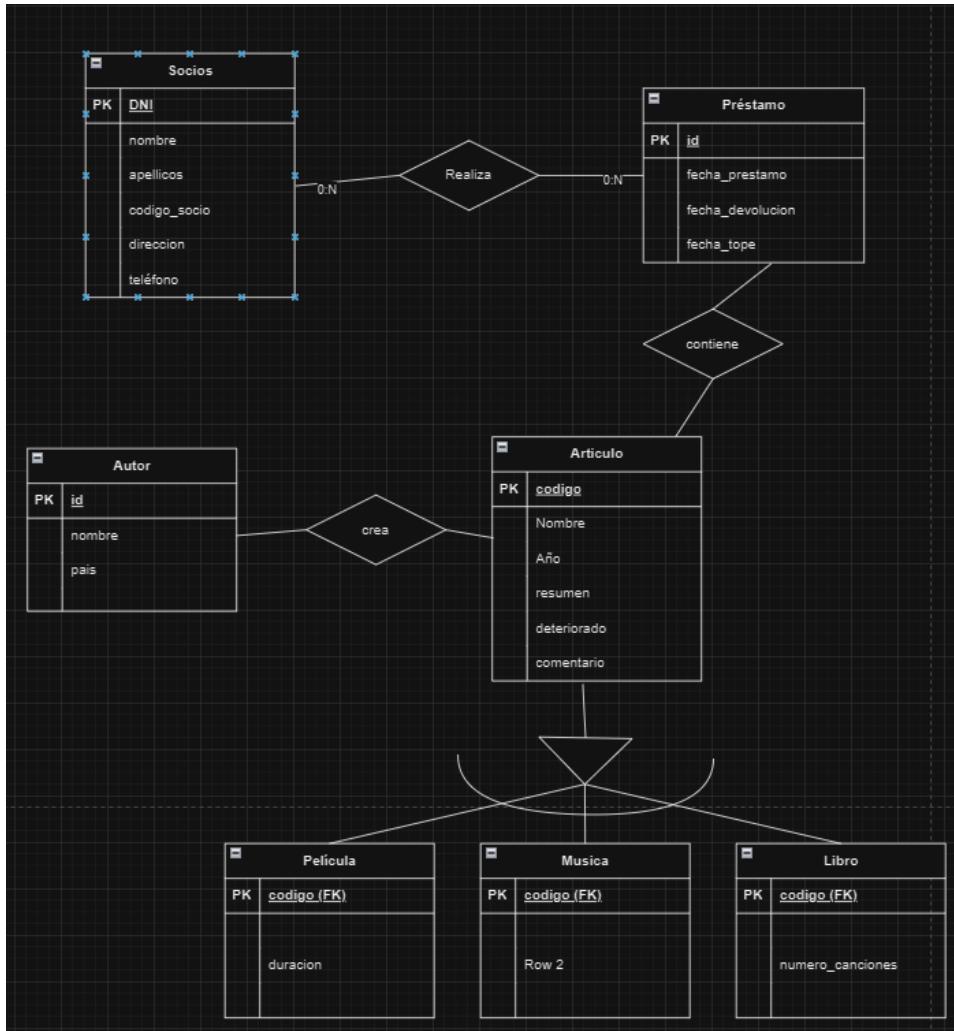
-- Ejemplo de UPDATE: Actualizar el precio de una pieza en la tabla Pieza
UPDATE Pieza
SET precio = 1.00
WHERE tipo = 'Tuerca' AND modelo = 'ABC123';

-- Ejemplo de DELETE: Eliminar una pieza y sus referencias en la tabla Pieza_compuesta y
Pieza_estanteria
;
```

```
DELETE FROM Pieza_estanteria WHERE pieza = 'Perno';
DELETE FROM Pieza_compuesta WHERE pieza1 = 'Tuerca' AND pieza2 = 'Tuerca';
DELETE FROM Pieza WHERE tipo = 'Tuerca' AND modelo = 'ABC123';
```



Ejercicio 6



Socios(dni:string, nombre: string, apellidos: string, codigo_socio: number, direccion: string, telefono: string)

Prestamo(id : number, fecha_prestamo: date, fecha_devolucion: date, fecha_tope: date)

Socios_prestamo(dni_socio: string(FK), id_prestamo: number(FK))

Autor(id: number, nombre: string, pais: string)

Pelicula(codigo: number, nombre: string, año: number, resumen: string, deteriorado: bool, comentario: string, id_autor: number(FK), duracion: number)

Musica(codigo: number, nombre: string, año: number, resumen: string, deteriorado: bool, comentario: string, id_autor: number(FK), num_canciones: number)

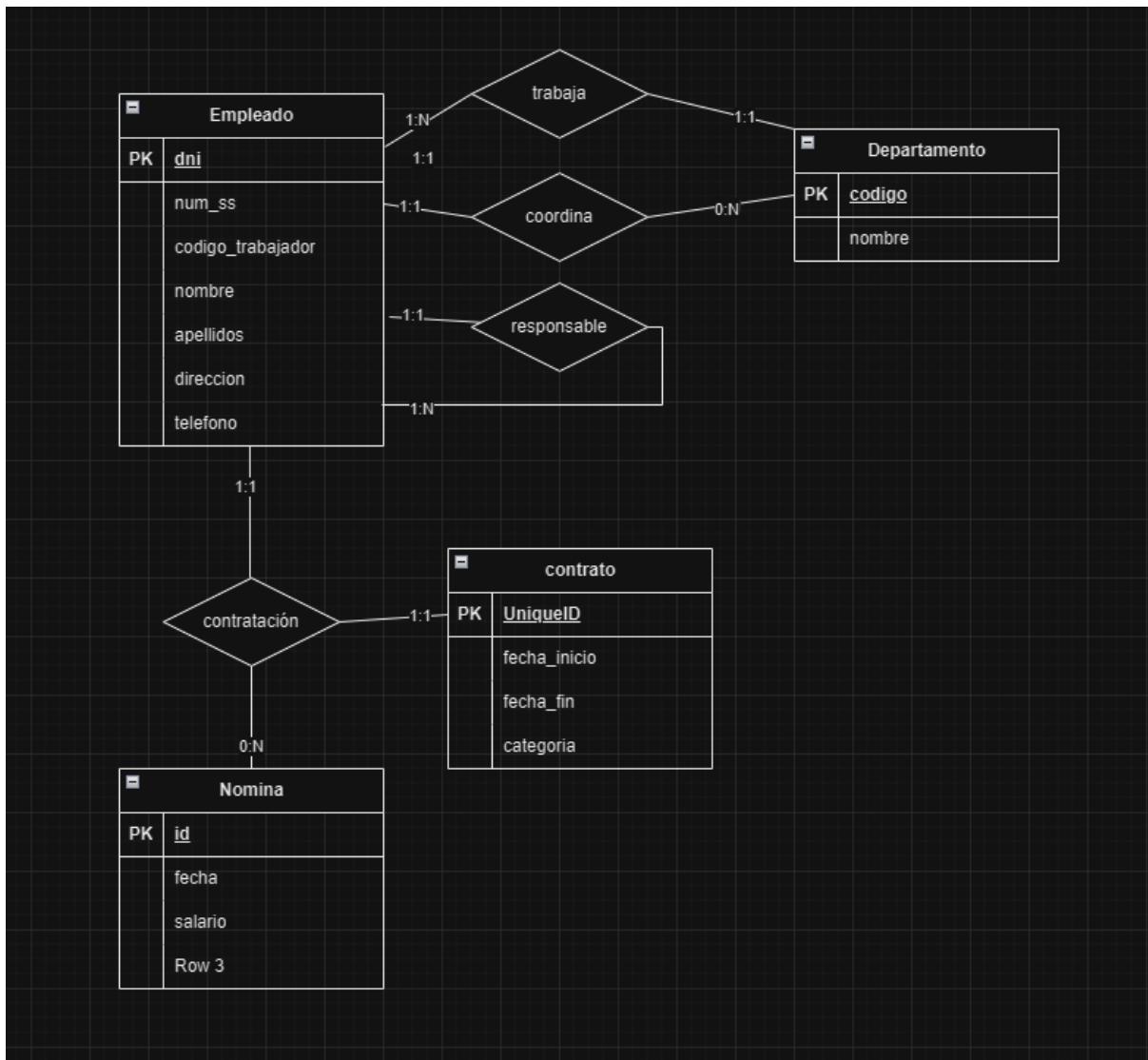
Libro(codigo: number, nombre: string, año: number, resumen: string, deteriorado: bool, comentario: string, id_autor: number(FK), numero_paginas: number)

Prestamo_pelicula(id_prestamo: number(FK), codigo_pelicula: number(FK))

Prestamo_libro(id_prestamo: number(FK), codigo_libro: number(FK))

Prestamo_musica(id_prestamo: number(FK), codigo_musica: number(FK))

Ejercicio 7



Departamento(codigo: number, nombre: string)

Empleado(dni: string, num_ss: number, codigo_trabajador: number, nombre: string,

apellidos: string, direccion: string, telefono: string, codigo_departamento: number (FK))

Departamento_empleado(dni_coordinador: string(FK), codigo_departamento: number(FK))

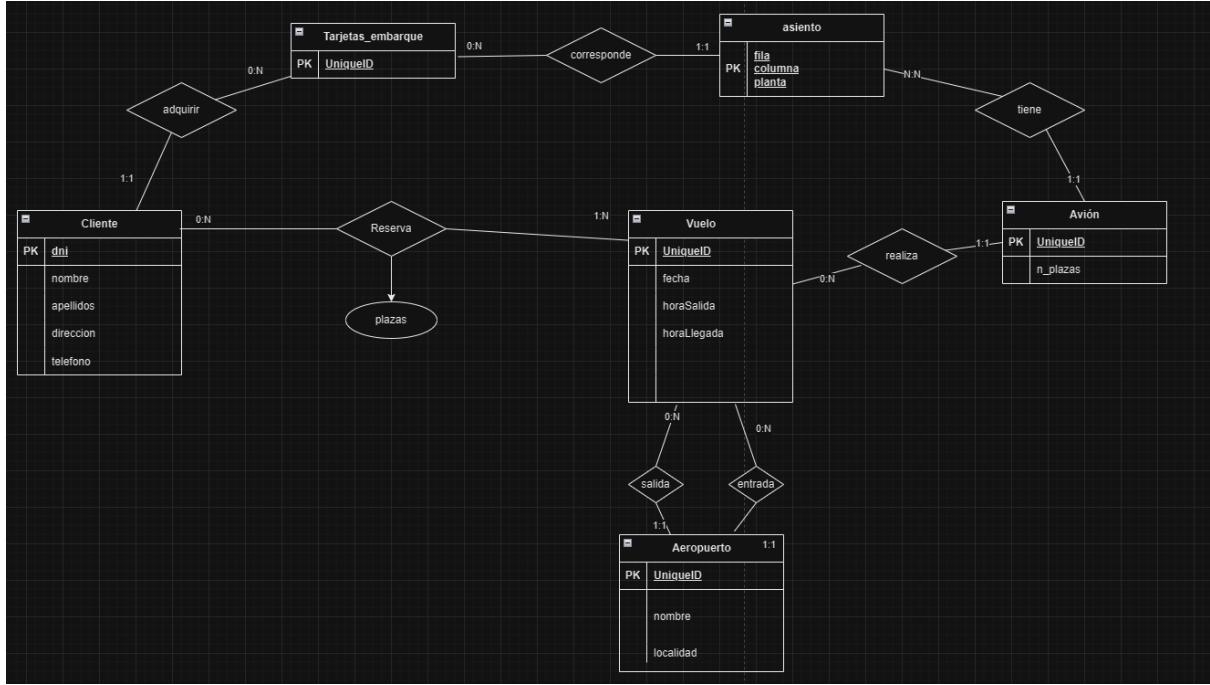
(Tabla intermedia para que no haya referencias entre sí entre Empleados y Departamentos.)

Contrato(id: number, fecha_inicio: date, fecha_fin: date, categoria: string)

Nomina(id: number, fecha: date, salario: number)

Contratacion(id_contrato: number, id_nomina: number, id_empleado: number)

Ejercicio 8. Vuelos



Cliente(dni: string, nombre:string, apellidos: string, direccion: string, telefono: string)

Aeropuerto(id: number, nombre: string, localidad: string)

Vuelo(id: number, fecha: date, hora_salida:date, horaLlegada:date, aeropuerto_salida_id: number(FK), aeropuerto_entrada:number(FK), id_avion: number(FK))

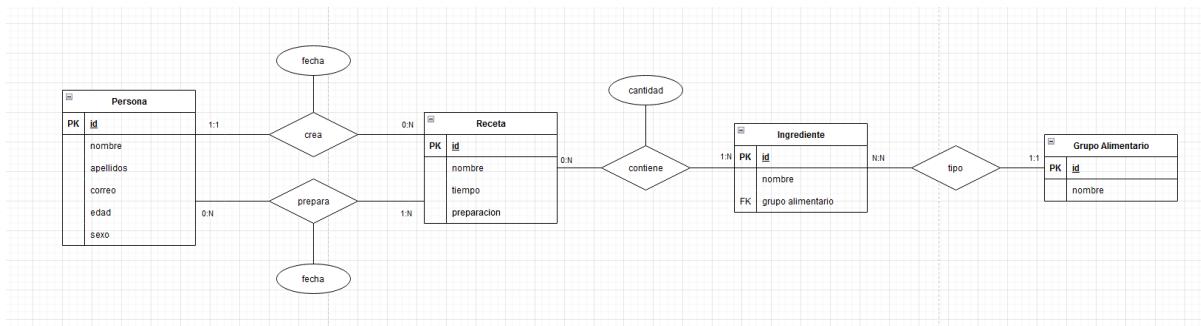
Reserva(dni_cliente:string(FK), id_vuelo:number(FK), **numero_reserva**: number, plazas: number)

Avion(id: number, num_plazas: number)

Asiento(fila: number, columna: number, planta: number, id_avion: number(FK))

Tarjetas_embarque(id: number, dni_cliente: string(FK), fila: number(FK), columna: number(FK), planta: number(FK), id_avion: number(FK))

Ejercicio 9. Recetas de cocina



Personas(**id**:number, nombre:string , apellidos:string, correo:string, edad:number, sexo: string)

Recetas(**id**:number, nombre:string, tiempo:number, preparacion:string, fecha:date,
id_persona_FK:number)

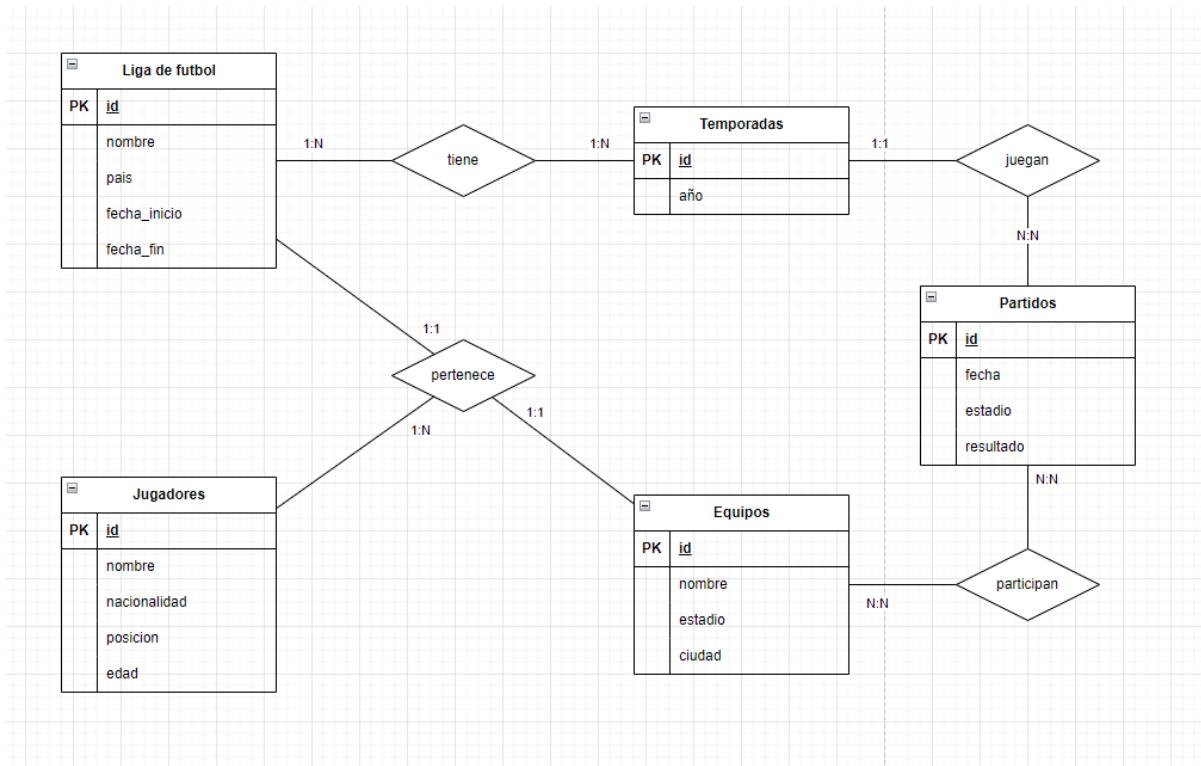
Cocineros(**id_persona_FK**:number, **id_receta_FK**:number, fecha:date)

Grupo_alimentario(**id**:number, nombre:string)

Ingredientes(**id**:number, nombre:string, **id_grupo_alimentario_FK**:string)

Contiene(**id_receta_FK**:number, **id_ingrediente_FK**:number, cantidad:number)

Ejercicio 10



Jugadores(id:number,nombre:string,nacionalidad:string,posicion:string,edad:number)

Equipos(id:number,nombre:string,estadio:string,ciudad:string)

Ligas(id:number,nombre:string,pais:string,fecha_inicio:date,fecha_fin:date)

Temporadas(id:number,año:number)

Partidos(id:number,fecha:date,estadio:string,resultado:string,temporada:id_temporada(FK))

jugadores_equipos_ligas(jugador:id_jugador(FK),equipo:id_equipo(FK),liga:id_liga(FK))

ligas_temporadas(liga:id_liga(FK),temporada:id_temporada(FK))

equipos_partidos(equipo:id_equipo(FK),partido:id_partido(FK))

DDL y DML

create database ex10;

use ex10;

CREATE TABLE Jugadores (

```

    id int auto_increment PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    nacionalidad VARCHAR(100),
    posicion VARCHAR(50),
    edad int
);
```

CREATE TABLE Equipos (

```

    id int auto_increment PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    estadio VARCHAR(200),
```

```
ciudad VARCHAR(100)
);

CREATE TABLE Ligas (
    id int auto_increment PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    pais VARCHAR(100),
    fecha_inicio DATE,
    fecha_fin DATE
);

CREATE TABLE Temporadas (
    id int auto_increment PRIMARY KEY,
    ano int
);

CREATE TABLE Partidos (
    id int auto_increment PRIMARY KEY,
    fecha DATE,
    estadio VARCHAR(200),
    resultado VARCHAR(50),
    temporada int,
    FOREIGN KEY (temporada) REFERENCES Temporadas(id)
);

CREATE TABLE jugadores_equipos_ligas (
    jugador int,
    equipo int,
    liga int,
    PRIMARY KEY (jugador, equipo, liga),
    FOREIGN KEY (jugador) REFERENCES Jugadores(id)
        on delete cascade
        on update cascade,
    FOREIGN KEY (equipo) REFERENCES Equipos(id)
        on delete cascade
        on update cascade,
    FOREIGN KEY (liga) REFERENCES Ligas(id)
        on delete cascade
        on update cascade
);

CREATE TABLE ligas_temporadas (
    liga int,
    temporada int,
    PRIMARY KEY (liga, temporada),
    FOREIGN KEY (liga) REFERENCES Ligas(id)
        on delete cascade
        on update cascade,
    FOREIGN KEY (temporada) REFERENCES Temporadas(id)
        on delete cascade
        on update cascade
);
```

```

CREATE TABLE equipos_partidos (
    equipo int,
    partido int,
    PRIMARY KEY (equipo, partido),
    FOREIGN KEY (equipo) REFERENCES Equipos(id)
    on delete cascade
    on update cascade,
    FOREIGN KEY (partido) REFERENCES Partidos(id)
    on delete cascade
    on update cascade
);
-- Inserción de datos en la tabla Jugadores
INSERT INTO Jugadores (id, nombre, nacionalidad, posicion, edad)
VALUES (1, 'Jugador 1', 'Español', 'Delantero', 25),
       (2, 'Jugador 2', 'Francés', 'Defensa', 28);

-- Inserción de datos en la tabla Equipos
INSERT INTO Equipos (id, nombre, estadio, ciudad)
VALUES (101, 'Equipo 1', 'Estadio 1', 'Ciudad 1'),
       (102, 'Equipo 2', 'Estadio 2', 'Ciudad 2');

-- Inserción de datos en la tabla Ligas
INSERT INTO Ligas (id, nombre, pais, fecha_inicio, fecha_fin)
VALUES (201, 'Liga 1', 'España', '2023-01-01', '2023-12-31'),
       (202, 'Liga 2', 'Francia', '2023-02-01', '2023-12-31');

-- Inserción de datos en la tabla Temporadas
INSERT INTO Temporadas (id, año)
VALUES (301, 2023),
       (302, 2024);

-- Inserción de datos en la tabla Partidos
INSERT INTO Partidos (id, fecha, estadio, resultado, temporada)
VALUES (401, '2023-07-01', 'Estadio 1', '2-1', 301),
       (402, '2023-07-15', 'Estadio 2', '0-0', 301);

-- Inserción de datos en la tabla jugadores_equipos_ligas
INSERT INTO jugadores_equipos_ligas (jugador, equipo, liga)
VALUES (1, 101, 201),
       (2, 102, 202);

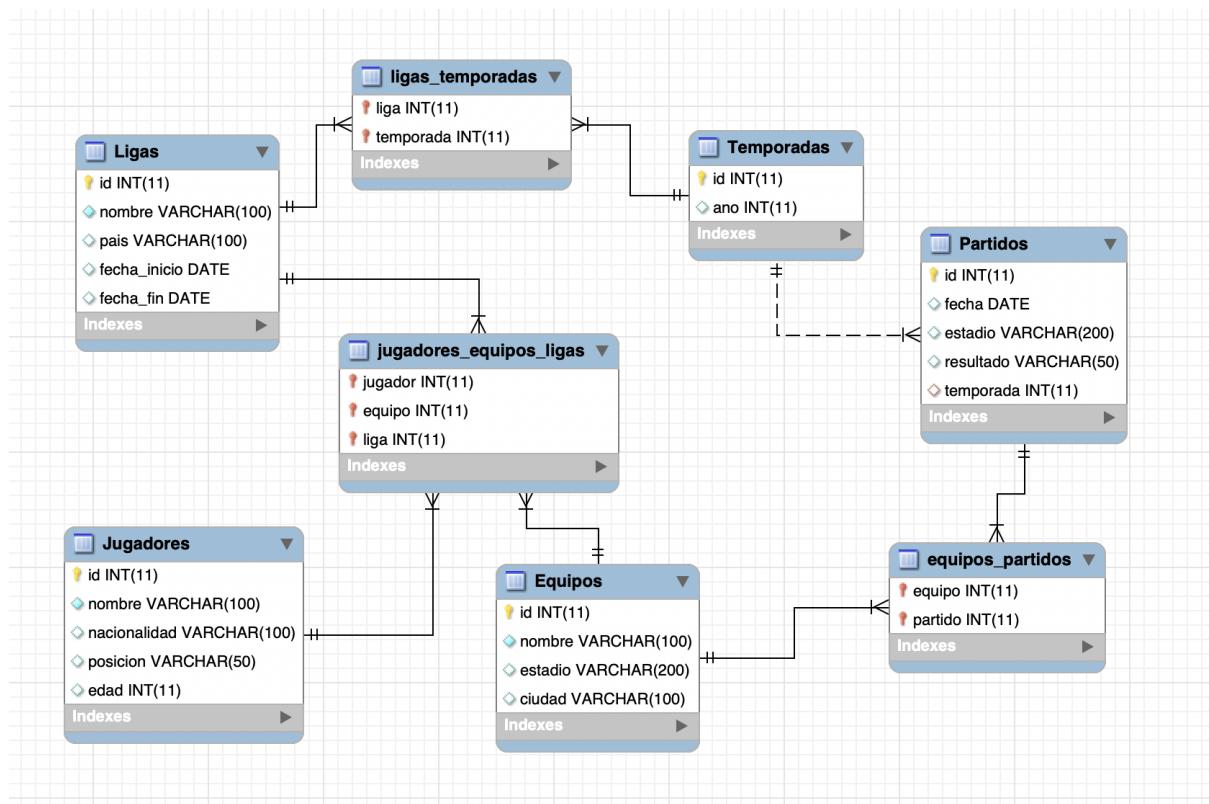
-- Inserción de datos en la tabla ligas_temporadas
INSERT INTO ligas_temporadas (liga, temporada)
VALUES (201, 301),
       (202, 302);

-- Inserción de datos en la tabla equipos_partidos
INSERT INTO equipos_partidos (equipo, partido)
VALUES (101, 401),
       (102, 402);

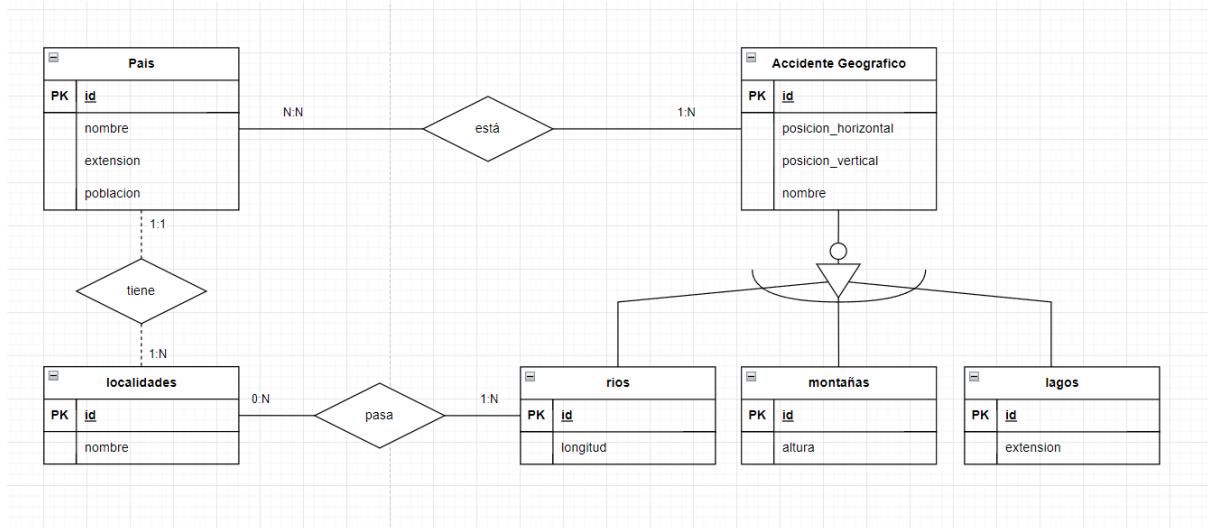
```

```
-- Ejemplo de UPDATE: Actualizar el nombre de un equipo en la tabla Equipos
UPDATE Equipos
SET nombre = 'Nuevo Equipo'
WHERE id = 102;
```

```
-- Ejemplo de DELETE: Eliminar un jugador y sus referencias en la tabla jugadores_equipos_ligas
DELETE FROM jugadores_equipos_ligas WHERE jugador = 2;
DELETE FROM Jugadores WHERE id = 2;
```



Ejercicio 11



Paises(id:number,nombre:string,extension:number,poblacion:number)
Accidentes_geograficos(id:number, pos_h:number, pos_v:number, nombre:string)
Rios(id:number, longitud:number, accidente_geografico:id_accidentesgeograficos(FK))
Montañas(id:number, altura:number, accidente_geografico:id_accidentesgeograficos(FK))
Lagos(id:number, extension:number, accidente_geografico:id_accidentesgeograficos(FK))
Localidades(id:number, nombre:string, pais:id_paises(FK))
Paises_accidentesgeograficos(pais:id_paises(FK),accidente_geografico:id_accidentesgeograficos(FK))
Localidades_rios(localidad:id_localidades(FK),rio:id_rios(FK))

DDL y DML

```
create database ex11;
use ex11;
```

```
CREATE TABLE Paises (
    id int auto_increment PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    extension int,
    poblacion int
);
```

```
CREATE TABLE Accidentes_geograficos (
    id int auto_increment PRIMARY KEY,
    pos_h int,
    pos_v int,
    nombre VARCHAR(100)
);
```

```
CREATE TABLE Rios (
    id int auto_increment PRIMARY KEY,
    longitud int,
    accidente_geografico int,
    FOREIGN KEY (accidente_geografico) REFERENCES Accidentes_geograficos(id)
```

```

on delete cascade
on update cascade
);

CREATE TABLE Montanas (
    id int auto_increment PRIMARY KEY,
    altura int,
    accidente_geografico int,
    FOREIGN KEY (accidente_geografico) REFERENCES Accidentes_geograficos(id)
    on delete cascade
    on update cascade
);

CREATE TABLE Lagos (
    id int auto_increment PRIMARY KEY,
    extension int,
    accidente_geografico int,
    FOREIGN KEY (accidente_geografico) REFERENCES Accidentes_geograficos(id)
    on delete cascade
    on update cascade
);

CREATE TABLE Localidades (
    id int auto_increment PRIMARY KEY,
    nombre VARCHAR(100),
    pais int,
    FOREIGN KEY (pais) REFERENCES Paises(id)
    on delete cascade
    on update cascade
);

CREATE TABLE Paises_accidentesgeograficos (
    pais int,
    accidente_geografico int,
    PRIMARY KEY (pais, accidente_geografico),
    FOREIGN KEY (pais) REFERENCES Paises(id)
    on delete cascade
    on update cascade,
    FOREIGN KEY (accidente_geografico) REFERENCES Accidentes_geograficos(id)
    on delete cascade
    on update cascade
);

CREATE TABLE Localidades_rios (
    localidad int,
    rio int,
    PRIMARY KEY (localidad, rio),
    FOREIGN KEY (localidad) REFERENCES Localidades(id)
    on delete cascade
    on update cascade,
    FOREIGN KEY (rio) REFERENCES Rios(id)
    on delete cascade
);

```

```

on update cascade
);

-- Inserción de datos en la tabla Paises
INSERT INTO Paises (id, nombre, extension, poblacion)
VALUES (1, 'España', 505990, 47000000),
       (2, 'Francia', 643801, 67000000);

-- Inserción de datos en la tabla Accidentes_geograficos
INSERT INTO Accidentes_geograficos (id, pos_h, pos_v, nombre)
VALUES (101, 40, 3, 'Accidente 1'),
       (102, 45, 2, 'Accidente 2');

-- Inserción de datos en la tabla Rios
INSERT INTO Rios (id, longitud, accidente_geografico)
VALUES (201, 1000, 101),
       (202, 800, 102);

-- Inserción de datos en la tabla Montañas
INSERT INTO Montanas (id, altura, accidente_geografico)
VALUES (301, 3000, 101),
       (302, 2500, 102);

-- Inserción de datos en la tabla Lagos
INSERT INTO Lagos (id, extension, accidente_geografico)
VALUES (401, 200, 101),
       (402, 150, 102);

-- Inserción de datos en la tabla Localidades
INSERT INTO Localidades (id, nombre, pais)
VALUES (501, 'Madrid', 1),
       (502, 'París', 2);

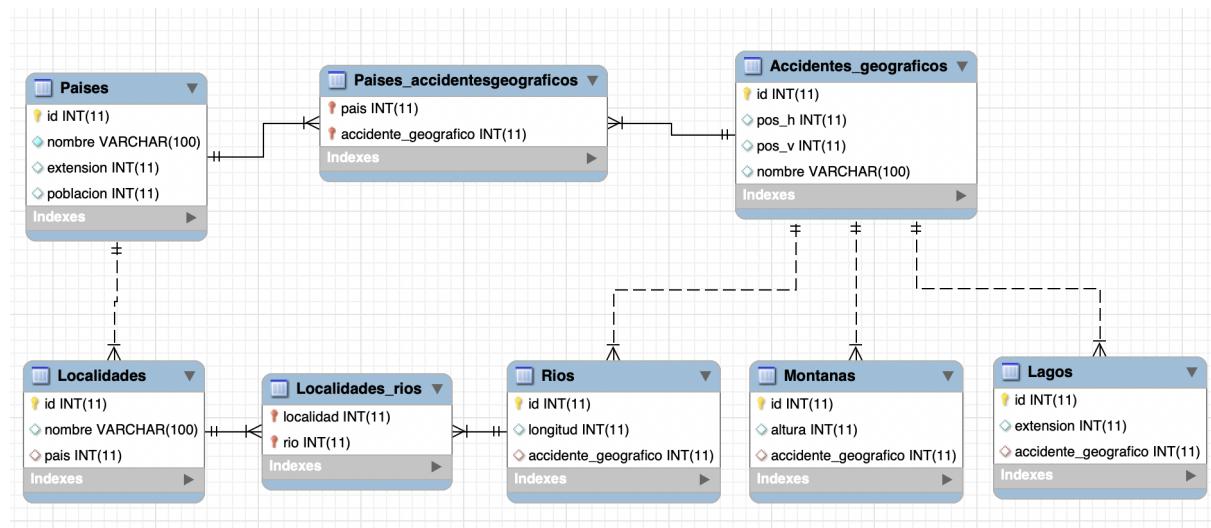
-- Inserción de datos en la tabla Paises_accidentesgeograficos
INSERT INTO Paises_accidentesgeograficos (pais, accidente_geografico)
VALUES (1, 101),
       (2, 102);

-- Inserción de datos en la tabla Localidades_rios
INSERT INTO Localidades_rios (localidad, rio)
VALUES (501, 201),
       (502, 202);

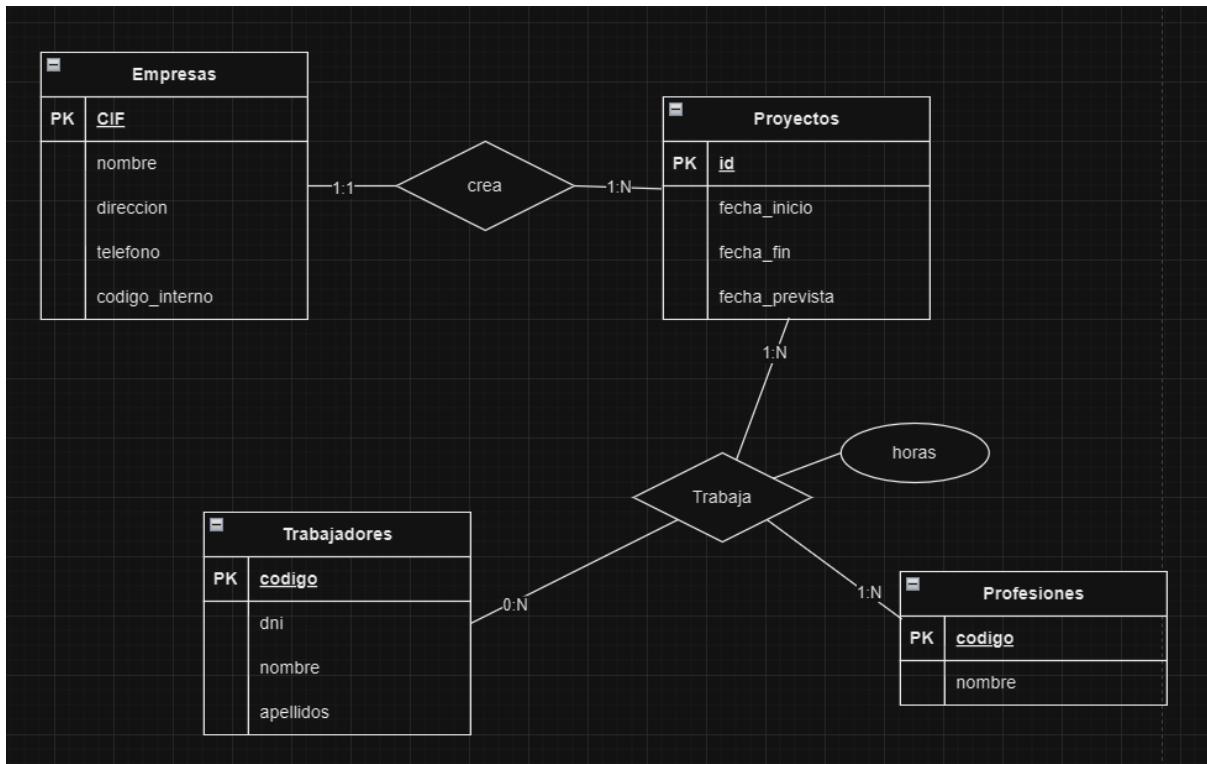
-- Ejemplo de UPDATE: Actualizar el nombre de un país en la tabla Paises
UPDATE Paises
SET nombre = 'Nuevo País'
WHERE id = 2;

-- Ejemplo de DELETE: Eliminar una montaña y sus referencias en la tabla Montañas y
-- Paises_accidentesgeograficos
DELETE FROM Paises_accidentesgeograficos WHERE accidente_geografico = 102;
DELETE FROM Montanas WHERE id = 302;

```



Ejercicio 12



Empresas(CIF:string,nombre:string,direccion:string,telefono:string,codigo_interno:string)
**Proyectos(id:number,fecha_inicio:date,fecha_fin:date,fecha_prevista:date,empresa:id_empr
esas(FK))**
Trabajadores(codigo:number,dni:string,nombre:string,apellidos:string)
Profesiones(id:number,nombre:string)
**Trabajadores_proyectos_profesiones(trabajador:id_trabajadores(FK),proyecto:id_proyectos(
FK),profesion:id_profesiones(FK),horas:number)**

DDL y DML

```

create database ex12;
use ex12;
  
```

```

CREATE TABLE Empresas (
    CIF VARCHAR(20) PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    direccion VARCHAR(200),
    telefono VARCHAR(20),
    codigo_interno VARCHAR(50)
);
  
```

```

CREATE TABLE Proyectos (
    id int auto_increment PRIMARY KEY,
    fecha_inicio DATE,
    fecha_fin DATE,
    fecha_prevista DATE,
    empresa VARCHAR(20),
    FOREIGN KEY (empresa) REFERENCES Empresas(CIF)
  
```

```
on delete cascade
on update cascade
);

CREATE TABLE Trabajadores (
    codigo int auto_increment PRIMARY KEY,
    dni VARCHAR(20) UNIQUE NOT NULL,
    nombre VARCHAR(100) NOT NULL,
    apellidos VARCHAR(100) NOT NULL
);

CREATE TABLE Profesiones (
    id int auto_increment PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL
);

CREATE TABLE Trabajadores_proyectos_profesiones (
    trabajador int,
    proyecto int,
    profesion int,
    horas int,
    PRIMARY KEY (trabajador, proyecto, profesion),
    FOREIGN KEY (trabajador) REFERENCES Trabajadores(codigo)
        on delete cascade
        on update cascade,
    FOREIGN KEY (proyecto) REFERENCES Proyectos(id)
        on delete cascade
        on update cascade,
    FOREIGN KEY (profesion) REFERENCES Profesiones(id)
        on delete cascade
        on update cascade
);
;

-- Inserción de datos en la tabla Empresas
INSERT INTO Empresas (CIF, nombre, direccion, telefono, codigo_interno)
VALUES ('12345678A', 'Empresa 1', 'Calle Mayor 123', '987654321', 'INT001'),
       ('87654321B', 'Empresa 2', 'Avenida Principal 456', '654321987', 'INT002');

-- Inserción de datos en la tabla Proyectos
INSERT INTO Proyectos (id, fecha_inicio, fecha_fin, fecha_prevista, empresa)
VALUES (1, '2023-01-01', '2023-12-31', '2023-06-30', '12345678A'),
       (2, '2023-02-01', '2023-11-30', '2023-08-31', '87654321B');

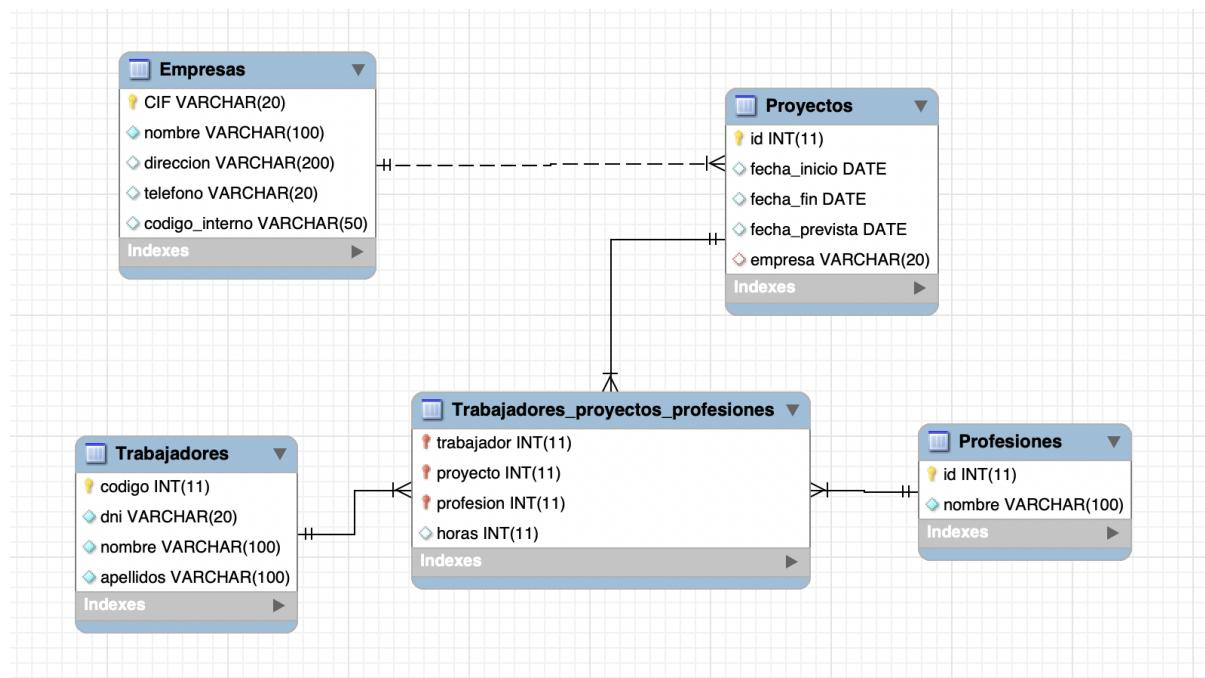
-- Inserción de datos en la tabla Trabajadores
INSERT INTO Trabajadores (codigo, dni, nombre, apellidos)
VALUES (101, '11111111A', 'Trabajador 1', 'Apellido 1'),
       (102, '22222222B', 'Trabajador 2', 'Apellido 2');

-- Inserción de datos en la tabla Profesiones
INSERT INTO Profesiones (id, nombre)
VALUES (201, 'Programador'),
       (202, 'Diseñador');
```

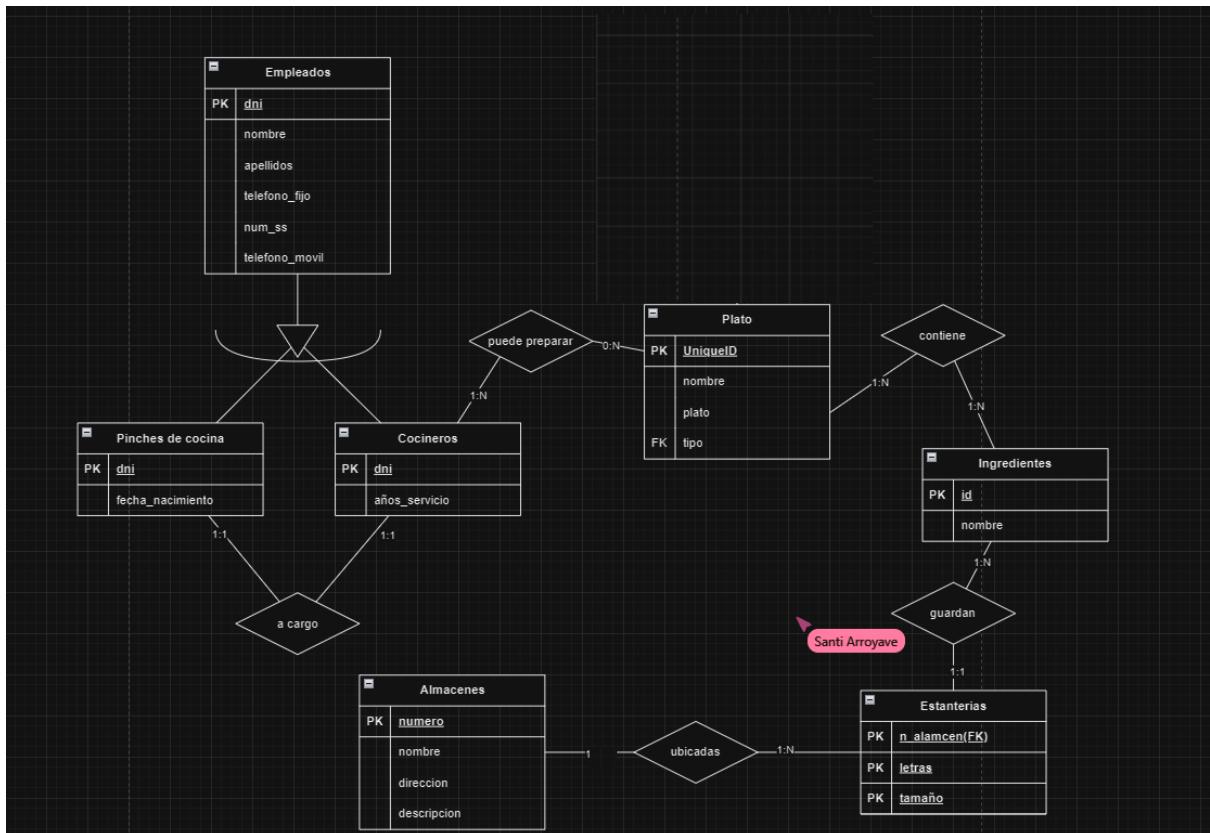
```
-- Inserción de datos en la tabla Trabajadores_proyectos_profesiones
INSERT INTO Trabajadores_proyectos_profesiones (trabajador, proyecto, profesion, horas)
VALUES (101, 1, 201, 160),
       (102, 2, 202, 120);
```

```
-- Ejemplo de UPDATE: Actualizar el nombre de una empresa en la tabla Empresas
UPDATE Empresas
SET nombre = 'Nueva Empresa'
WHERE CIF = '87654321B';
```

```
-- Ejemplo de DELETE: Eliminar un trabajador y sus referencias en la tabla
Trabajadores_proyectos_profesiones
DELETE FROM Trabajadores_proyectos_profesiones WHERE trabajador = 102;
DELETE FROM Trabajadores WHERE codigo = 102;
```



Ejercicio 13



Pinches_cocina(dni: string, nombre: string, apellidos: string, telefono_fijo: string, num_ss: number, telefono_movil: string, fecha_nacimiento(date), dni_encargado: number(FK))

Cocineros(dni: string, nombre: string, apellidos: string, telefono_fijo: string, num_ss: number, telefono_movil: string, años_servicio: number)

plato(id: number, plato: string, tipo: string)

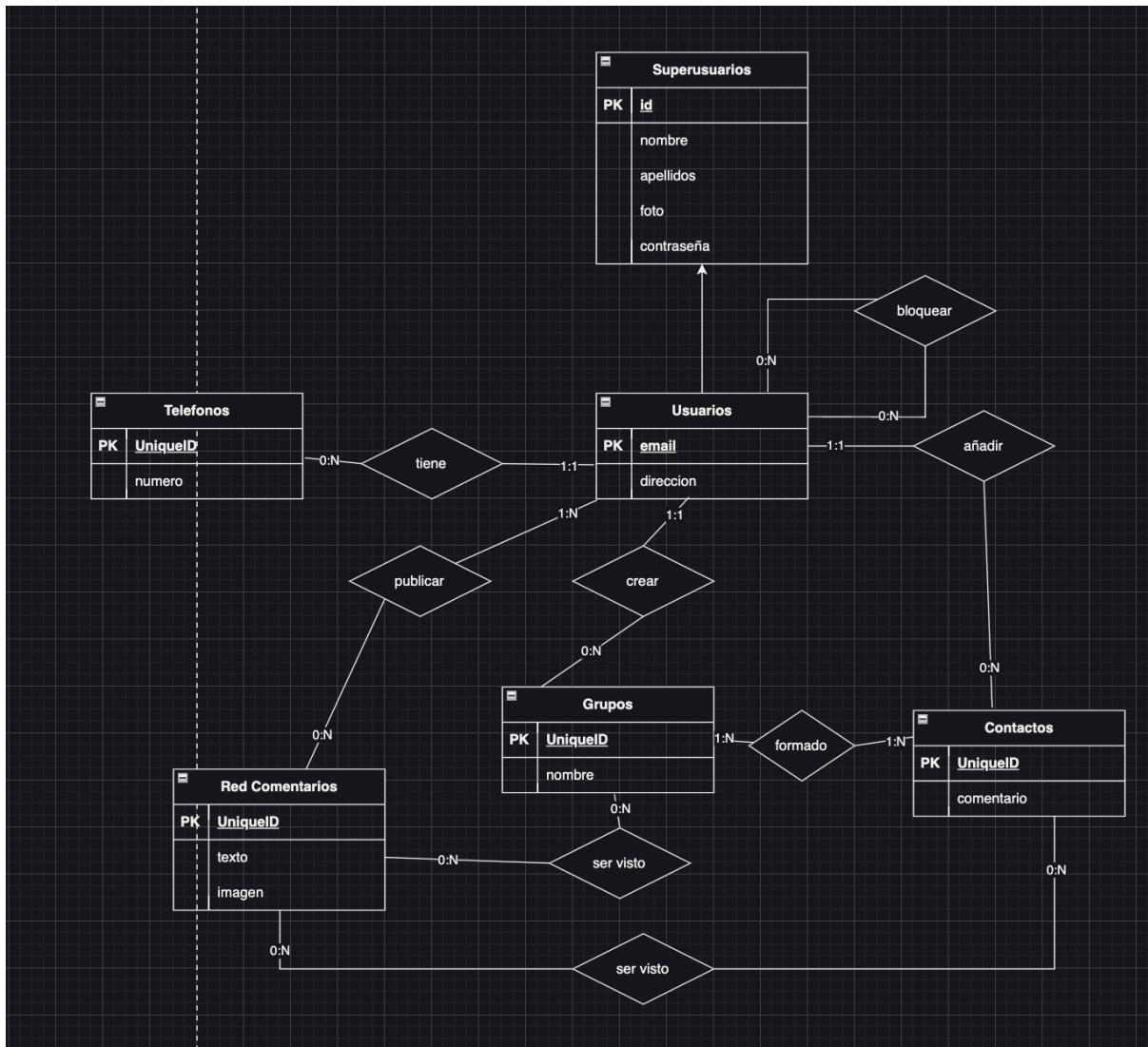
cocineros_platos(dni_cocinero(FK), id_plato(FK))

Almacenes(numero: id, nombre: string, direccion: string, descripcion: string)

Estanterias(n_almacen: number(FK), letras: string, tamaño: number)

Ingredientes(id: number, nombre: string, n_almacen(FK), letra_estanteria(FK), tamaño_estanteria(FK))

Ejercicio 14. Red Social



Superusuarios(id:number,nombre:string,apellidos:string,foto:string,contraseña:string)

Usuarios(email:string,direccion:string,superusuario:id_superusuarios(FK))

Telefonos(id:number,numero:string,usuario:id_usuarios(FK))

Contactos(id:number,comentario:string,usuario:id_usuarios(FK))

Grupos(id:number,nombre:string,usuario:id_usuario(FK))

RedComentarios(id:number,texto:string,imagen:string)

Grupos_contactos(grupo:id_grupos(FK),contacto:id_contactos(FK))

Usuarios_redcomentarios(usuario:id_usuarios(FK),red_comentario:id_redcomentarios(FK))

Redcomentarios_grupos(red_comentario:id_redcomentarios(FK),grupo:id_grupos(FK))

Redcomentarios_contactos(grupo:id_grupos(FK),comentario:id_comentarios(FK))

Usuarios_bloqueados(bloqueante:id_usuarios(FK),bloqueado:id_usuarios(FK))

DDL y DML

```
create database ex14;
use ex14;
```

```
CREATE TABLE Superusuarios (
    id int auto_increment PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    apellidos VARCHAR(100) NOT NULL,
    foto VARCHAR(200),
    contraseña VARCHAR(100) NOT NULL
);
```

```
CREATE TABLE Usuarios (
    email VARCHAR(100) PRIMARY KEY,
    direccion VARCHAR(200),
    superusuario int,
    FOREIGN KEY (superusuario) REFERENCES Superusuarios(id)
    on delete cascade
    on update cascade
);
```

```
CREATE TABLE Telefonos (
    id int auto_increment PRIMARY KEY,
    numero VARCHAR(20),
    usuario VARCHAR(100),
    FOREIGN KEY (usuario) REFERENCES Usuarios(email)
    on delete cascade
    on update cascade
);
```

```
CREATE TABLE Contactos (
    id int auto_increment PRIMARY KEY,
    comentario VARCHAR(200),
    usuario VARCHAR(100),
    FOREIGN KEY (usuario) REFERENCES Usuarios(email)
    on delete cascade
    on update cascade
);
```

```
CREATE TABLE Grupos (
    id int auto_increment PRIMARY KEY,
    nombre VARCHAR(100),
    usuario VARCHAR(100),
    FOREIGN KEY (usuario) REFERENCES Usuarios(email)
    on delete cascade
    on update cascade
);
```

```
CREATE TABLE RedComentarios (
    id int auto_increment PRIMARY KEY,
    texto VARCHAR(200),
    imagen VARCHAR(200)
```

```
);

CREATE TABLE Grupos_contactos (
    grupo int,
    contacto int,
    PRIMARY KEY (grupo, contacto),
    FOREIGN KEY (grupo) REFERENCES Grupos(id)
        on delete cascade
        on update cascade,
    FOREIGN KEY (contacto) REFERENCES Contactos(id)
        on delete cascade
        on update cascade
);
```

```
CREATE TABLE Usuarios_redcomentarios (
    usuario VARCHAR(100),
    red_comentario int,
    PRIMARY KEY (usuario, red_comentario),
    FOREIGN KEY (usuario) REFERENCES Usuarios(email)
        on delete cascade
        on update cascade,
    FOREIGN KEY (red_comentario) REFERENCES RedComentarios(id)
        on delete cascade
        on update cascade
);
```

```
CREATE TABLE Redcomentarios_grupos (
    red_comentario int,
    grupo int,
    PRIMARY KEY (red_comentario, grupo),
    FOREIGN KEY (red_comentario) REFERENCES RedComentarios(id)
        on delete cascade
        on update cascade,
    FOREIGN KEY (grupo) REFERENCES Grupos(id)
        on delete cascade
        on update cascade
);
```

```
CREATE TABLE Redcomentarios_contactos (
    red_comentario int,
    comentario int,
    PRIMARY KEY (red_comentario, comentario),
    FOREIGN KEY (red_comentario) REFERENCES RedComentarios(id)
        on delete cascade
        on update cascade,
    FOREIGN KEY (comentario) REFERENCES Contactos(id)
        on delete cascade
        on update cascade
);
```

```
CREATE TABLE Usuarios_bloqueados (
    bloqueante VARCHAR(100),
```

```
bloqueado VARCHAR(100),
PRIMARY KEY (bloqueante, bloqueado),
FOREIGN KEY (bloqueante) REFERENCES Usuarios(email)
on delete cascade
on update cascade,
FOREIGN KEY (bloqueado) REFERENCES Usuarios(email)
on delete cascade
on update cascade
);

-- Inserción de datos en la tabla Superusuarios
INSERT INTO Superusuarios (id, nombre, apellidos, foto, contraseña)
VALUES (1, 'Superusuario 1', 'Apellido 1', 'foto1.jpg', 'contraseña1'),
       (2, 'Superusuario 2', 'Apellido 2', 'foto2.jpg', 'contraseña2');

-- Inserción de datos en la tabla Usuarios
INSERT INTO Usuarios (email, direccion, superusuario)
VALUES ('usuario1@example.com', 'Calle Mayor 123', 1),
       ('usuario2@example.com', 'Avenida Principal 456', 2);

-- Inserción de datos en la tabla Telefonos
INSERT INTO Telefonos (id, numero, usuario)
VALUES (101, '987654321', 'usuario1@example.com'),
       (102, '654321987', 'usuario2@example.com');

-- Inserción de datos en la tabla Contactos
INSERT INTO Contactos (id, comentario, usuario)
VALUES (201, 'Contacto 1', 'usuario1@example.com'),
       (202, 'Contacto 2', 'usuario2@example.com');

-- Inserción de datos en la tabla Grupos
INSERT INTO Grupos (id, nombre, usuario)
VALUES (301, 'Grupo 1', 'usuario1@example.com'),
       (302, 'Grupo 2', 'usuario2@example.com');

-- Inserción de datos en la tabla RedComentarios
INSERT INTO RedComentarios (id, texto, imagen)
VALUES (401, 'Comentario 1', 'imagen1.jpg'),
       (402, 'Comentario 2', 'imagen2.jpg');

-- Inserción de datos en la tabla Grupos_contactos
INSERT INTO Grupos_contactos (grupo, contacto)
VALUES (301, 201),
       (302, 202);

-- Inserción de datos en la tabla Usuarios_redcomentarios
INSERT INTO Usuarios_redcomentarios (usuario, red_comentario)
VALUES ('usuario1@example.com', 401),
       ('usuario2@example.com', 402);

-- Inserción de datos en la tabla Redcomentarios_grupos
INSERT INTO Redcomentarios_grupos (red_comentario, grupo)
```

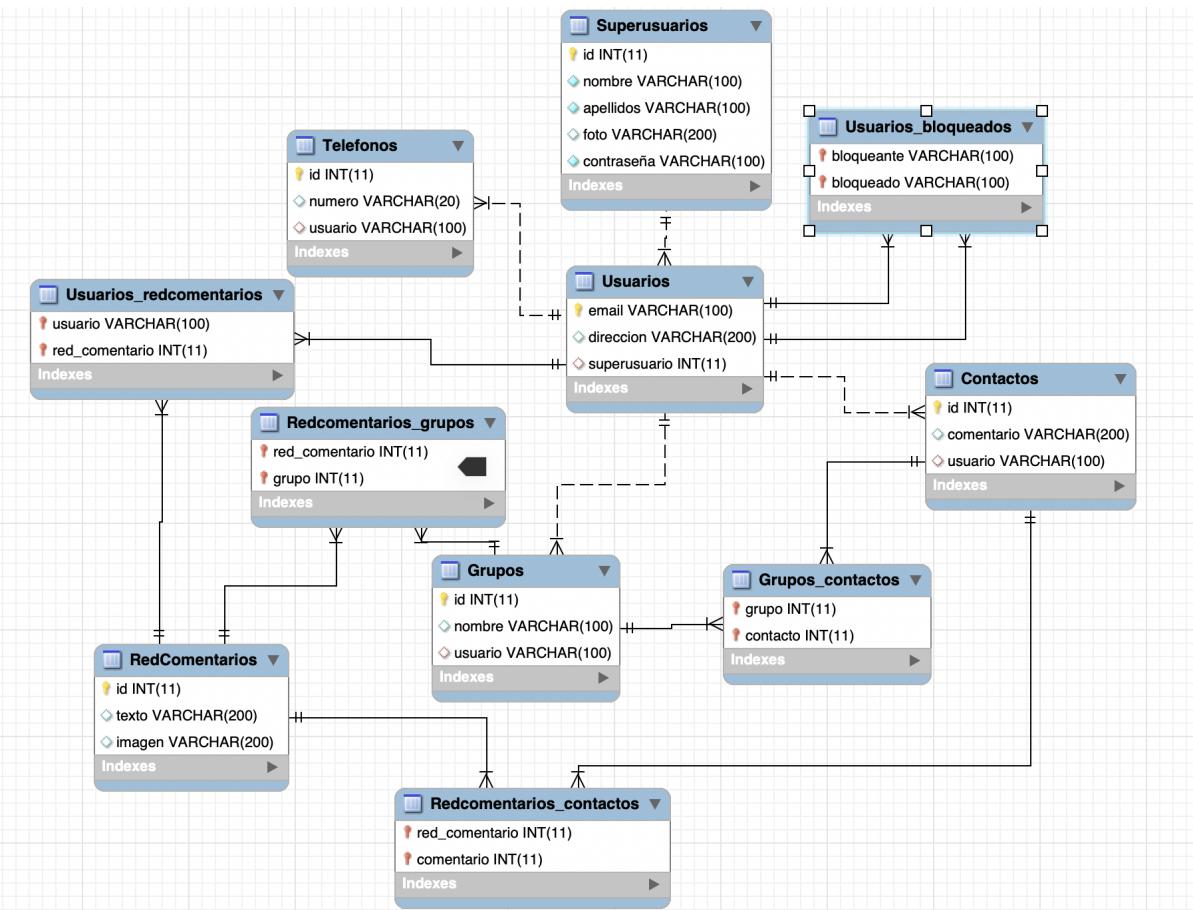
```
VALUES (401, 301),
(402, 302);
```

```
-- Inserción de datos en la tabla Redcomentarios_contactos
INSERT INTO Redcomentarios_contactos (red_comentario, comentario)
VALUES (401, 201),
(402, 202);
```

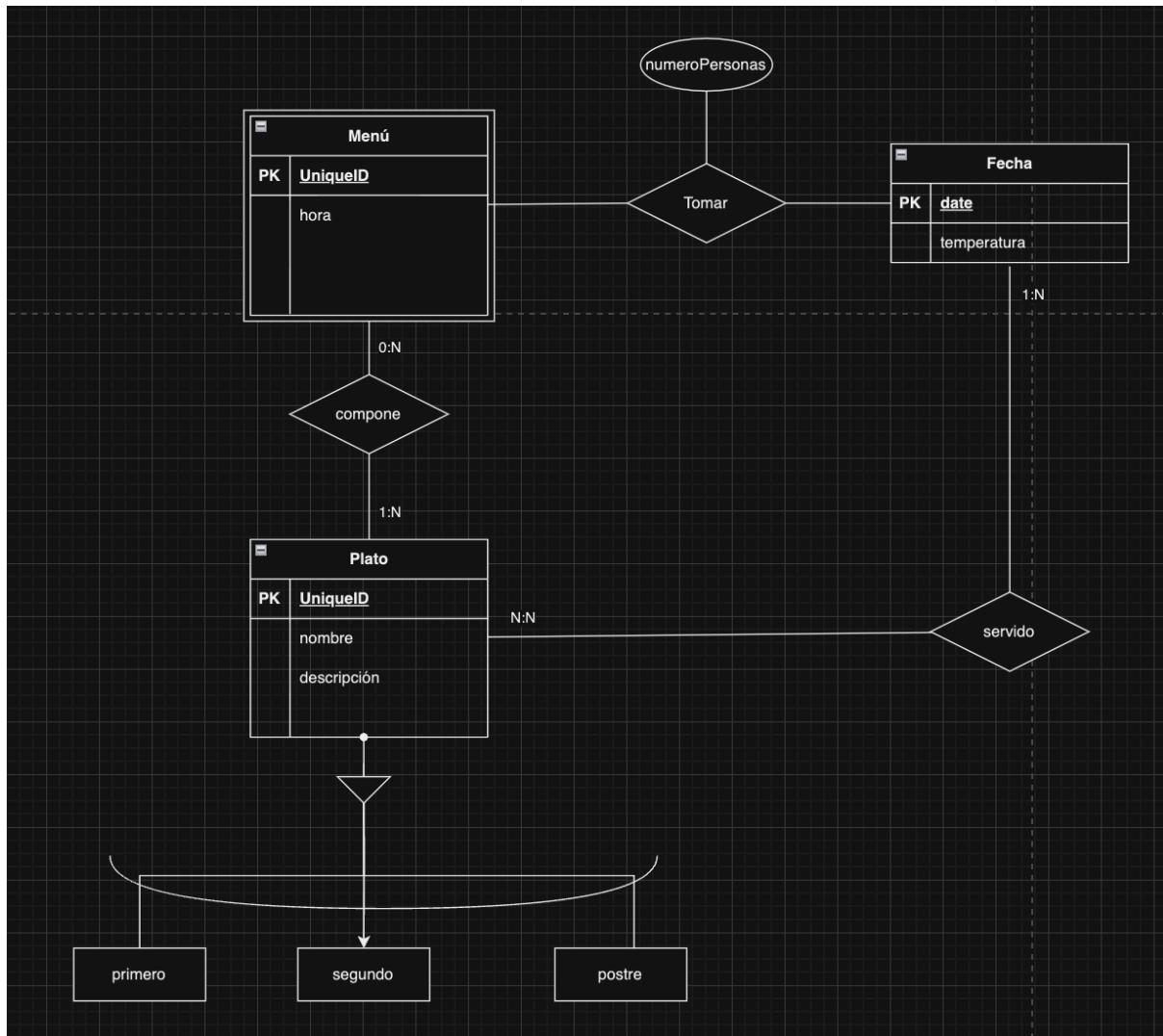
```
-- Inserción de datos en la tabla Usuarios_bloqueados
INSERT INTO Usuarios_bloqueados (bloqueante, bloqueado)
VALUES ('usuario1@example.com', 'usuario2@example.com');
```

```
-- Ejemplo de UPDATE: Actualizar el nombre de un superusuário en la tabla Superusuarios
UPDATE Superusuarios
SET nombre = 'Nuevo Superusuário'
WHERE id = 2;
```

```
-- Ejemplo de DELETE: Eliminar un grupo y sus referencias en las tablas Grupos, Grupos_contactos
y Redcomentarios_grupos
DELETE FROM Redcomentarios_grupos WHERE grupo = 302;
DELETE FROM Grupos_contactos WHERE grupo = 302;
DELETE FROM Grupos WHERE id = 302;
```



Ejercicio 15. Menú diario



Plato(id:number,nombre:string,descripcion:string)
Primero(id_plato(FK))
Segundo(id_plato(FK))
Postre(id_plato(FK))
Menu(id:number,hora:string)
Fecha(date:date,temperatura:string)
Menu_Plato(id_menu(FK),id_plato(FK))
Menu_Fecha(id_menu(FK),date(FK),num_personas:number)
Plato_Fecha(id_plato(FK),date(FK))

DDL y DML

```
CREATE DATABASE ex15;  
use ex15;  
create table plato(  
    id int auto_increment primary key,  
    nombre varchar(25),  
    descripcion varchar(25)
```

```
);

INSERT INTO plato (nombre, descripcion) VALUES
('Plato 1', 'Descripción del Plato 1'),
('Plato 2', 'Descripción del Plato 2'),
('Plato 3', 'Descripción del Plato 3');
```

```
create table primero(
    id int primary key,
    FOREIGN KEY(id) REFERENCES plato(id)
    ON DELETE cascade
    ON UPDATE cascade
);
```

```
INSERT INTO primero (id) VALUES (1);
```

```
create table segundo(
    id int primary key,
    FOREIGN KEY(id) REFERENCES plato(id)
    ON DELETE cascade
    ON UPDATE cascade
);
```

```
INSERT INTO segundo (id) VALUES (2);
```

```
create table postre(
    id int primary key,
    FOREIGN KEY(id) REFERENCES plato(id)
    ON DELETE cascade
    ON UPDATE cascade
);
```

```
INSERT INTO postre (id) VALUES (3);
```

```
create table menu(
    id int auto_increment primary key,
    hora varchar(10)
);
```

```
INSERT INTO menu (hora) VALUES ('12:00 PM'), ('7:00 PM');
```

```
create table fechas(
    fecha date primary key,
    temperatura int
);
```

```
INSERT INTO fechas (fecha, temperatura) VALUES
```

```
('2023-07-26', 28),
('2023-07-27', 30),
('2023-07-28', 26);
```

```
create table menu_plato(
    menu int,
    plato int,
    primary key(menu,plato),
    foreign key(menu) references menu(id)
    on delete cascade
    on update cascade,
    foreign key(plato) references plato(id)
    on delete cascade
    on update cascade
);
```

```
INSERT INTO menu_plato (menu, plato) VALUES
(1, 1),
(1, 2),
(1, 3),
(2, 1),
(2, 2);
```

```
create table menu_fecha(
    menu int,
    fecha date,
    num_personas int,
    primary key(menu,fecha),
    foreign key(menu) references menu(id)
    on delete cascade
    on update cascade,
    foreign key(fecha) references fechas(fecha)
    on delete cascade
    on update cascade
);
```

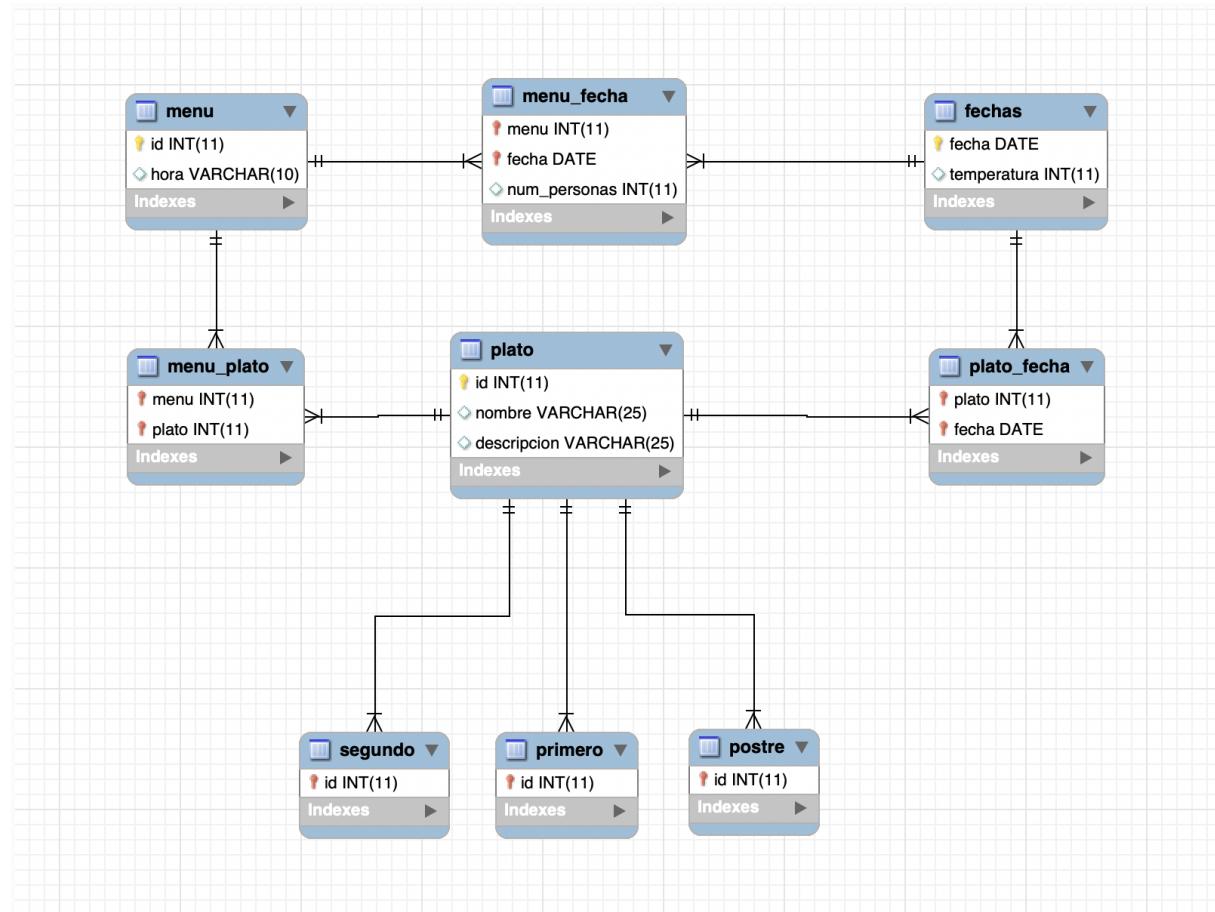
```
INSERT INTO menu_fecha (menu, fecha, num_personas) VALUES
(1, '2023-07-26', 4),
(1, '2023-07-27', 6),
(1, '2023-07-28', 3),
(2, '2023-07-26', 5),
(2, '2023-07-27', 8);
```

```
create table plato_fecha(
    plato int,
    fecha date,
    primary key(plato,fecha),
    foreign key(plato) references plato(id)
    on delete cascade
    on update cascade,
```

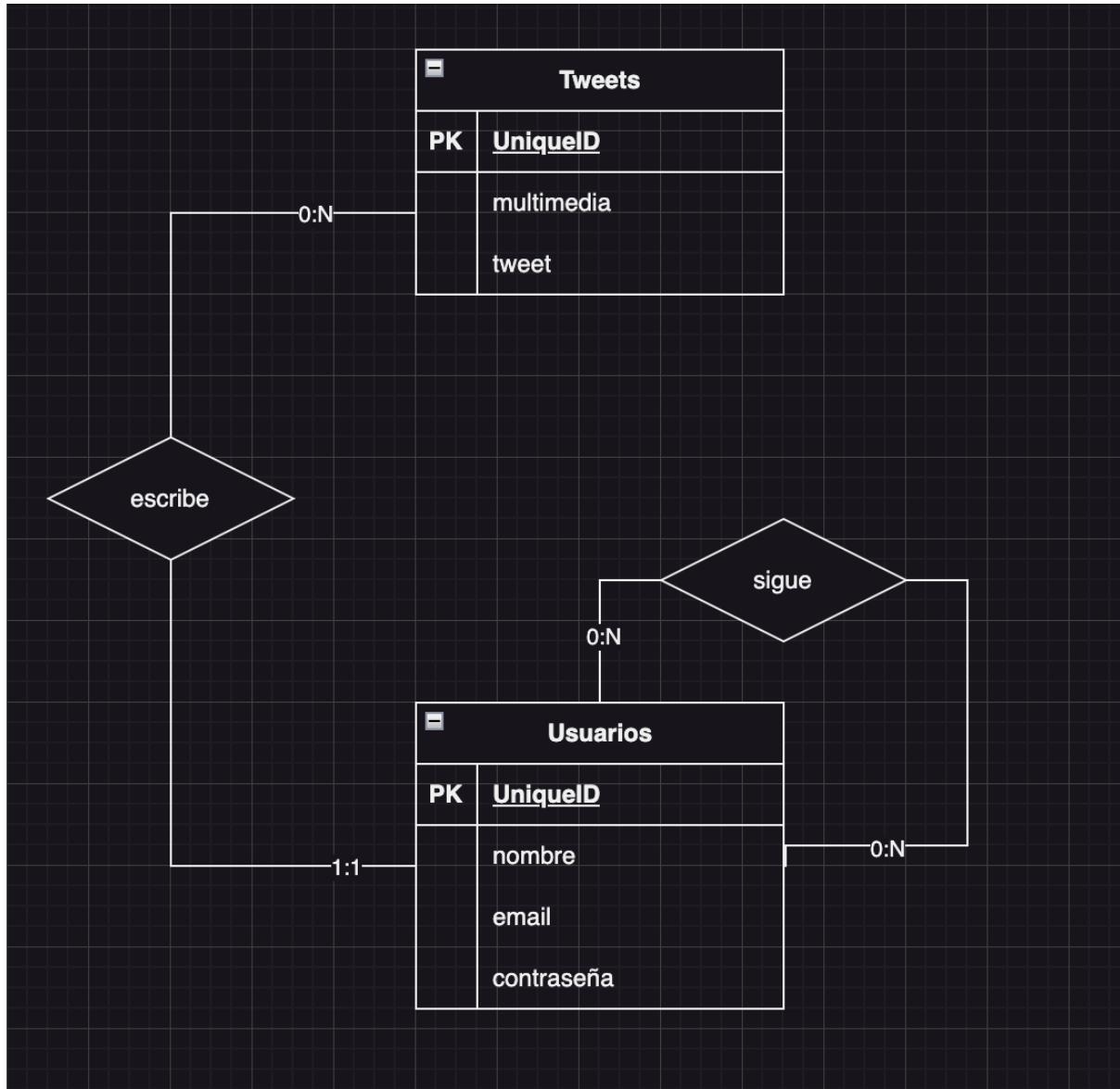
```
foreign key(fecha) references fechas(fecha)
on delete cascade
on update cascade
);
```

```
INSERT INTO plato_fecha (plato, fecha) VALUES  
(1, '2023-07-26'),  
(2, '2023-07-26'),  
(3, '2023-07-26'),  
(1, '2023-07-27'),  
(2, '2023-07-27');
```

```
-- Supongamos que queremos cambiar la descripción del 'Plato 1'.
UPDATE plato SET descripcion = 'Nueva descripción' WHERE id = 1;
-- Supongamos que queremos eliminar el plato asociado al 'primero' con ID = 1.
DELETE FROM primero WHERE id = 1;
-- Supongamos que queremos cambiar la hora del 'menu' con ID = 2.
UPDATE menu SET hora = '8:30 PM' WHERE id = 2;
-- Supongamos que queremos eliminar la fecha '2023-07-28' de la tabla 'fechas'.
DELETE FROM fechas WHERE fecha = '2023-07-28';
-- Supongamos que queremos cambiar el número de personas para el 'menu' con ID = 1 y fecha '2023-07-26'.
UPDATE menu_fecha SET num_personas = 5 WHERE menu = 1 AND fecha = '2023-07-26';
-- Supongamos que queremos eliminar la relación entre el 'plato' con ID = 1 y la fecha '2023-07-26'.
DELETE FROM plato_fecha WHERE plato = 1 AND fecha = '2023-07-26';
```



Ejercicio 16. Twitter



Usuarios(id, nombre, email, contraseña)

Tweets(id, multimedia, tweet)

Seguidores(seguido:id_usuario(FK), seguidor:id_usuario(FK))

Usuario_Tweet(tweet:id_tweet(FK), usuario:id_usuario(FK))

DDL

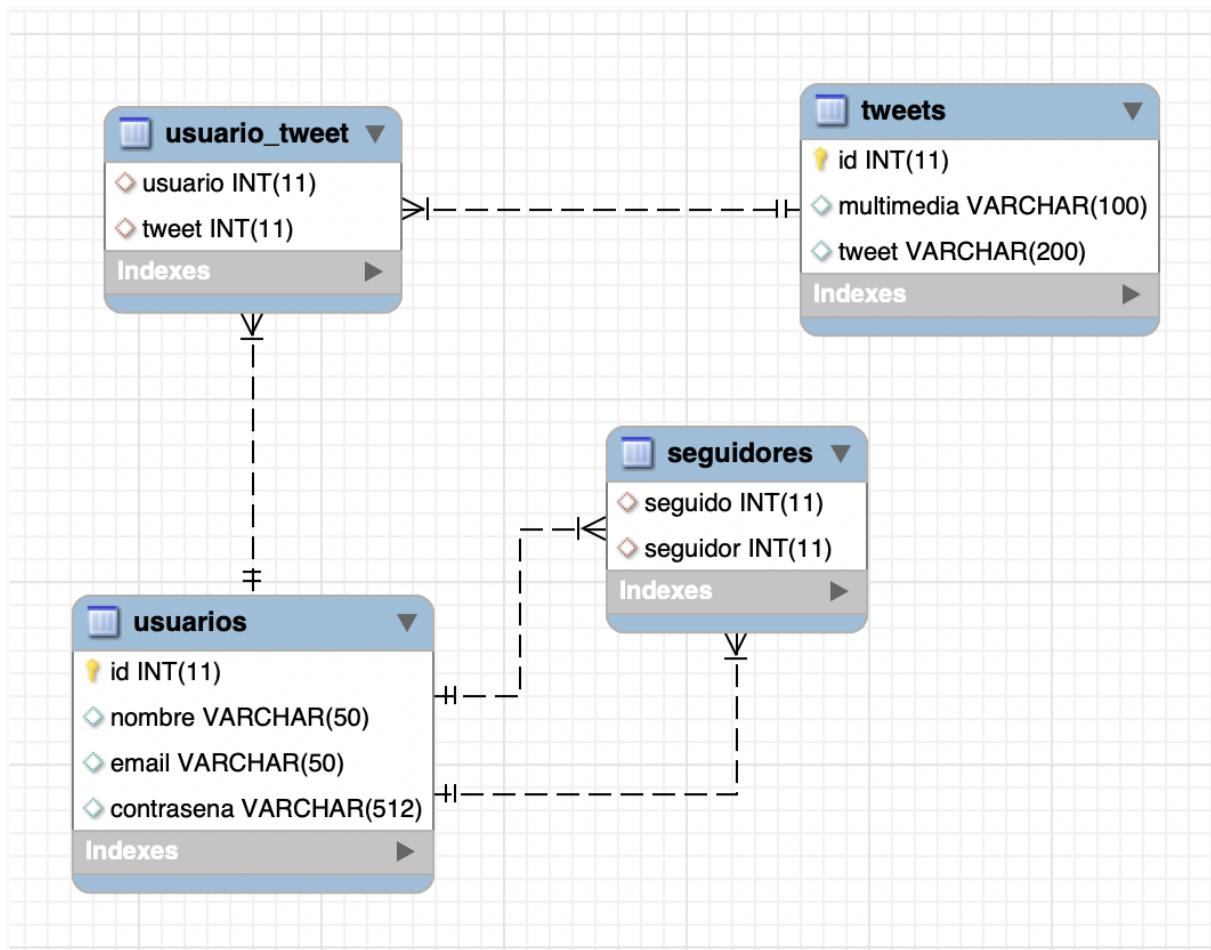
```
create database ex16;
use ex16;

create table usuarios(
    id int auto_increment primary key,
    nombre varchar(50),
    email varchar(50),
    contrasena varchar(512)
);

create table tweets(
    id int auto_increment primary key,
    multimedia varchar(100),
    tweet varchar(200)
);

create table seguidores(
    seguido int,
    seguidor int,
    foreign key(seguido) references usuarios(id)
    on delete cascade
    on update cascade,
    foreign key(seguidor) references usuarios(id)
    on delete cascade
    on update cascade
);

create table usuario_tweet(
    usuario int,
    tweet int,
    foreign key(usuario) references usuarios(id)
    on delete cascade
    on update cascade,
    foreign key(tweet) references tweets(id)
    on delete cascade
    on update cascade
);
```



DML

```

INSERT INTO usuarios (nombre, email, contrasena) VALUES
('Usuario1', 'usuario1@example.com', 'contraseña123'),
('Usuario2', 'usuario2@example.com', 'clave456'),
('Usuario3', 'usuario3@example.com', 'pass789');

```

```

INSERT INTO tweets (multimedia, tweet) VALUES
('imagen1.jpg', 'Este es el primer tweet.'),
('video1.mp4', 'Compartiendo un video interesante.'),
(NULL, 'Un tweet sin multimedia.'),
('imagen2.png', '¡Seguimos compartiendo contenido!');

```

```

INSERT INTO seguidores (seguido, seguidor) VALUES
(1, 2),
(1, 3);

```

```

INSERT INTO seguidores (seguido, seguidor) VALUES
(2, 1);

```

```

INSERT INTO seguidores (seguido, seguidor) VALUES
(3, 1);

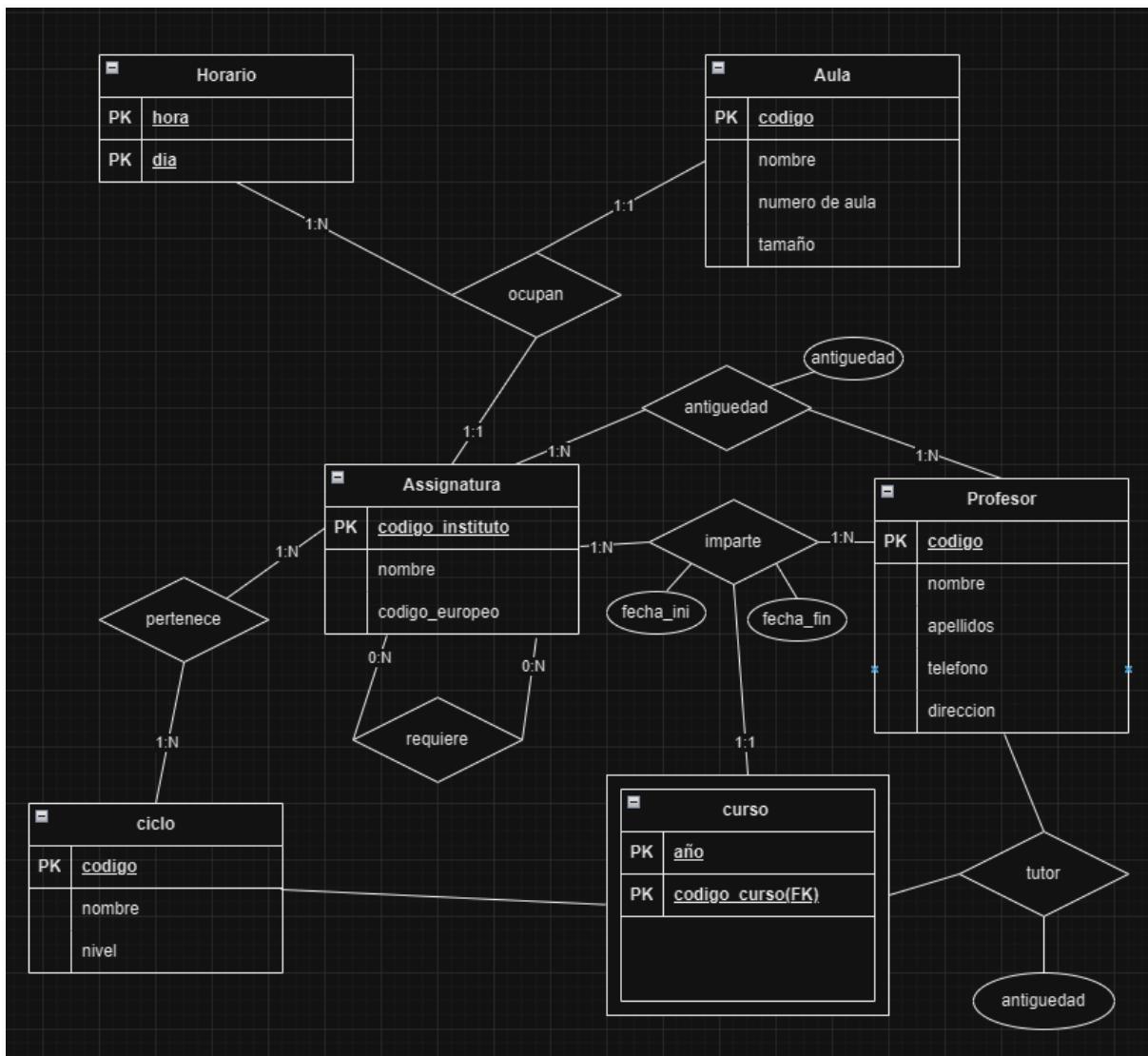
```

```
INSERT INTO usuario_tweet (usuario, tweet) VALUES  
(1, 1),  
(1, 3);
```

```
INSERT INTO usuario_tweet (usuario, tweet) VALUES  
(2, 2),  
(2, 4);
```

```
INSERT INTO usuario_tweet (usuario, tweet) VALUES  
(3, 3),  
(3, 4);
```

Ejercicio 17. Horario escolar



Horario(**hora**: string, **dia**: string)

Aula(**codigo**: number, nombre: string, numero_aula: number, tamaño: number)

Assignatura(**codigo_instituto**: number, codigo_europeo: number, nombre: string)

ocupacion(**hora**: string, **dia**: string, **codigo_aula**: number, **codigo_assignatura**: number)

Requerimiento_assignaturas(**id_assignatura**: number(FK),

id_assignatura_requerida:number (FK))

Profesor(**codigo**:number, nombre: string, apellidos: string, telefono: string, direccion: string)

Ciclo(**codigo**: number, nombre: string, nivel: number)

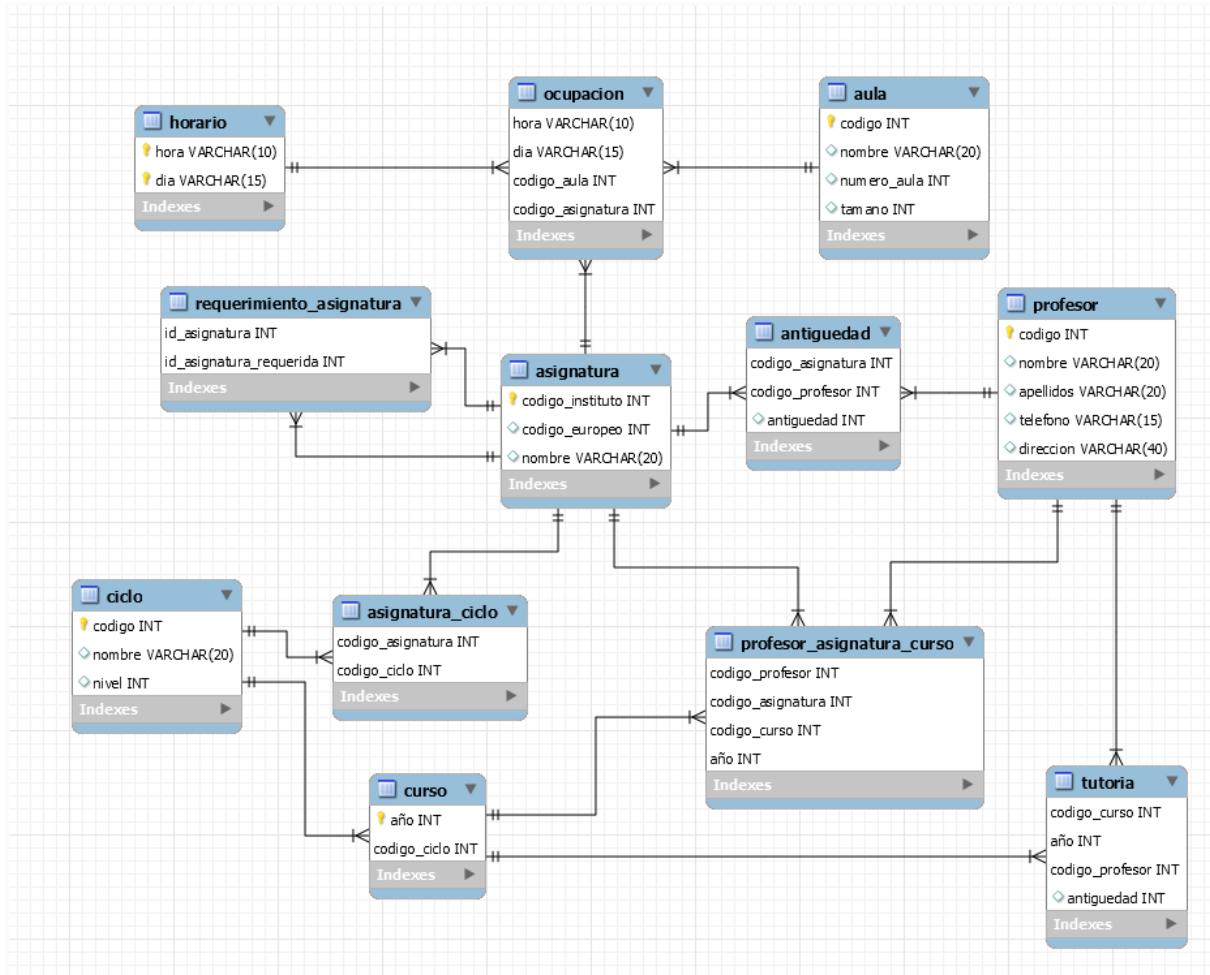
Curso(**año**: number, **codigo_ciclo**: number(FK))

Antiguedad(**codigo_assignatura**:number(FK), **codigo_profesor**:number(FK), antiguedad: number)

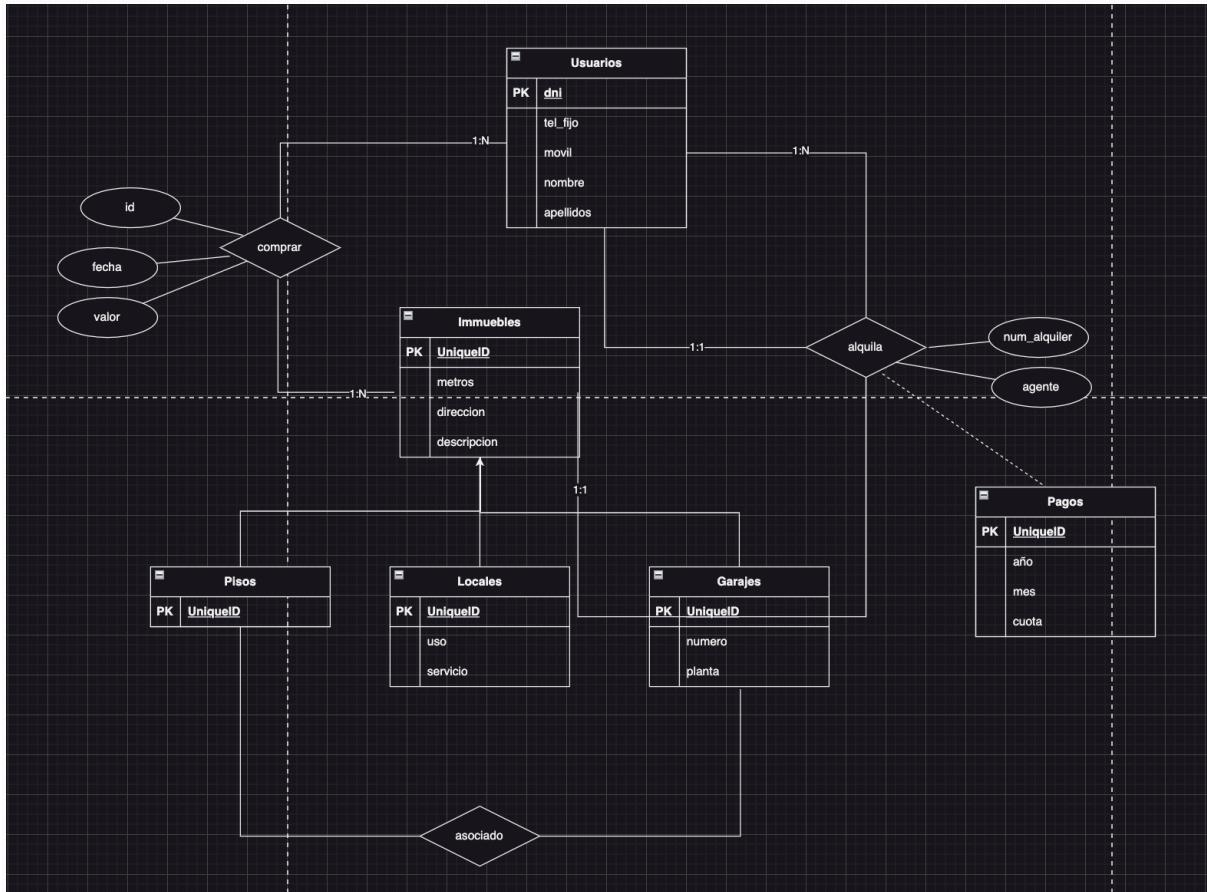
Profesor_asignatura_curso(**codigo_profesor**: number(FK), **codigo_asignatura**: number(FK), **codigo_curso**: number (FK), **año**: number (FK))

tutoria(codigo_curso: number(FK), año: number(FK), codigo_profesor: number(FK), antiguedad:number)

Assignatura_ciclo(codigo_asignatura:number(FK), codigo_ciclo: number(FK))



Ejercicio 18. Inmuebles



Usuarios(**dni**: string, **tel_fijo**: number, **movil**: number, **nombre**: string, **apellidos**: string)

Pisos(**id**: number, **metros**: number, **direccion**: string, **descripcion**: string)

Locales(**id**: number, **metros**: number, **direccion**: string, **descripcion**: string, **uso**: string, **servicio**: string)

Garajes(**id**: number, **metros**: number, **direccion**: string, **descripcion**: string, **numero**: string, **planta**: string)

Pagos(**id**: number, **fecha**: date, **cuota**: number)

compra_piso(**id**: number, **id_piso**: number(FK), **dni_usuario**: string(FK), **fecha**: date, **valor**: number)

compra_local(**id**: number, **id_local**: number(FK), **dni_usuario**: string(FK), **fecha**: date, **valor**: number)

compra_garaje(**id**: number, **id_garaje**: number(FK), **dni_usuario**: string(FK), **fecha**: date, **valor**: number)

alquiler_piso(**id_piso**: number(FK), **dni_inquilino**: string(FK),
dni_propietario(FK), **id_pago**(FK), **num_alquiler**: number, **agente**: string)

alquiler_local(**id_local: number(FK), dni_inquilino: string(FK),**
dni_propietario(FK), id_pago(FK), num_alquiler: number, agente: string)
alquiler_garaje(**id_garaje: number(FK), dni_inquilino: string(FK),**
dni_propietario(FK), id_pago(FK), num_alquiler: number, agente: string)