

Analisis 3 - Entrega

Table of Contents

- [1. Narrativa](#)
- [2. Analisis de Entidad-Relacion](#)
- [3. Entidades Identificadas](#)
- [4. Visiones](#)
- [5.Codigo Sql](#)
- [6. Diagrama Pata de gallo](#)
- [7. Marco Teorico Normalizacion](#)
 - [7.1. Primera Forma Normal \(1FN\)](#)
 - [7.2. Segunda Forma Normal \(2FN\)](#)
 - [7.3. Tercera Forma Normal \(3FN\)](#)
 - [7.4. Cuarta Forma Normal \(4FN\)](#)
 - [7.5. Quinta Forma Normal \(5FN\)](#)
- [8. Diagrama de clases](#)
- [9. Diagrama Entidad-Relacion](#)

1. Narrativa

La creación de este sistema surge como una herramienta que pueda servir de intermediario para que los usuarios puedan contactar de manera rápida y efectiva con distintos profesionales que se encarguen de brindar servicios, tales como: Electricista, gasista, plomero, etc. En primera medida permite el registro de los profesionales en la aplicación, donde podrán dejar su currículum, que tipo de servicios se encarga de proveer, disponibilidad horaria, tarifa, de esta manera será mucho más fácil encontrar de forma inmediata los servicios que requiere el cliente, el profesional deberá entregar los certificados necesarios para avalar sus habilidades en el tipo de servicio que brindará. El administrador del sistema podrá gestionar los servicios disponibles dentro de la aplicación, podrá agregar, modificar y eliminar servicios. Las publicaciones van a estar basadas única y exclusivamente en estos servicios. El profesional tendrá acceso a un sistema de tareas donde deberá realizar una cierta cantidad de trabajos y ganará nivel, el cual le permitirá ser recomendado en trabajos futuros. El profesional podrá realizar publicaciones especificando el servicio, la ubicación de trabajo y la tarifa. Luego se podrán registrar los propios usuarios, que tendrán una cuenta dentro del sistema que le permitirá acceder a todas sus funcionalidades, el registro podrá ser con una cuenta en el mismo sistema o mediante una red social, además podrán eliminar o modificar los datos de su cuenta en caso de algún error o imprevisto. El sistema cuenta con un servicio de filtrado, donde el cliente podrá elegir que tipo de servicio está buscando, que tarifas es capaz de manejar y en qué horarios podría acordar una cita, de esta manera la aplicación se torna mucho más flexible y eficiente, donde se le otorga un gran poder de elección al usuario. Una vez que selecciona el tipo de servicio aparecerán todos las publicaciones que se encuentran disponibles en ese momento, donde podrá observar de los mismos sus trabajos previos, calificaciones y comentarios de opinión.. Además, la aplicación cuenta con un sistema de mensajería interno que permite la comunicación directa y en tiempo real con los profesionales. Otra manera de contactarlos es mediante un sistema donde el usuario podrá ponerse en “Alerta”, qué hará saber a los profesionales que necesita un servicio en especial, de ésta manera será visto en el mapa y se le pondrá enviar distintas propuestas de trabajos con su tarifa, es así que el cliente tendrá una lista con los trabajos que le proponen sobre cierto servicio social y elegir el que le sea más conveniente, ya sea por tarifa o experiencia del trabajador. Una vez que el usuario selecciona la publicación que desea, se genera un precontrato, que requerirá de la confirmación por parte del profesional para transformarse en contrato.

Los usuarios deberán pagar la cita dentro de la aplicación, ya sea con tarjeta o depósito en efectivo, es aquí donde el sistema cobra un plus sobre la tarifa original del profesional. El sistema se encargará de retener el dinero por un tiempo determinado, esto lo hará con la finalidad de prever posibles reclamos por parte de los clientes antes situaciones como: Servicio de mala calidad, mal trato por parte del profesional, no cumplió con sus expectativas, etc. Es así como el usuario tendrá un tiempo específico para solicitar la devolución del dinero de manera exitosa, en caso contrario el pago será realizado al profesional y quedará fuera del alcance del sistema poder satisfacer las necesidades del cliente con respecto a la solicitud de reclamo. Cada día llegará un recordatorio al cliente de las citas que tiene pendiente junto con los datos tales como: tarifa, tipo de servicio, hora, fecha, lugar, de ésta manera el usuario podrá saber cuáles son las citas que tiene pendientes. Una vez culminado el servicio por parte del profesional, el usuario podrá evaluar su experiencia tanto con la aplicación como con los servicios brindados, podrá dejar comentarios que ayuden a otros clientes en el uso de este sistema.

2. Analisis de Entidad-Relacion

En este apartado se realizará un análisis exhaustivo sobre las distintas entidades que forman parte de la aplicación que se encarga de proveer servicios esenciales, haremos un esquema con las entidades que intervienen en el mismo, así como también las relaciones que existen entre las mismas. Además, haremos visiones de contexto que nos permitan comprender de una manera mucho más sencilla que relaciones existen entre las distintas entidades y observar el sistema desde otro punto de vista.

3. Entidades Identificadas

1. Profesionales
2. Administradores
3. Usuarios
4. Reseñas
5. Servicios-Publicacion
6. Estado
7. Formas-Pago
8. Pre-Contratos
9. Pre-Contratos-Confirmaciones
10. Comprobantes-De-Pago
11. Contratos
12. Recordatorios
13. Publicaciones
14. Permisos
15. Administradores-Permisos
16. Cambios
17. Administradores-Cambios
18. Categorías

4. Visiones

1. Se sabe que para los profesionales tendrán un código único que los identifique, los mismos podrán realizar muchas publicaciones al mismo tiempo, además tendrán un email y teléfonos únicos de contacto, tendrán la posibilidad de confirmar o rechazar los pre-Contratos generados por los usuarios interesados en dicha publicación.
2. Los usuarios tendrán un DNI, que los identificará inequívocamente del resto, también contarán con un email y teléfono de contacto únicos.
3. Las categorías contarán con una descripción única, que permitirán saber de que tipo es el servicio que se brindará.
4. El administrador, tendrá un Número que lo identificará, además contará con varios permisos, los cuales le permitirán realizar distintos cambios dentro del sistema.
5. Para los permisos se saben que cuentan con una descripción única que detallará que acción podrá realizar el administrador que posea el mismo.
6. Se sabe que las publicaciones se basarán en un único servicio, contarán con una descripción, fecha de inicio, estado actual, que profesional realizó dicha publicación y horas estimadas de trabajo por parte del profesional.
7. Se sabe de la reseña que tendrá un comentario, los usuarios podrán realizar cuantas reseñas deseen, pero la reseña se puede hacer por un solo usuario, sobre una publicación específica.
8. Los comprobantes de pago, contarán con un código, el método por el cual se realizó el mismo, y además la publicación que se pagó con dicho comprobante.
9. Los precontratos contarán con un código único, además los usuarios podrán estar involucrados en varios PreContratos, así como también los profesionales que se encuentran en la publicación en la cual se basa el pre-Contrato.
10. Los contratos, contarán con un código único, el usuario y publicación, el profesional, el comprobante de pago de dicho contrato y el pre-contrato en el cual se basa el mismo.
11. Los reclamos tendrán un código único, el usuario que lo realiza, de que contrato, el motivo, la fecha y el estado de este reclamo. Visiones de Contexto

5. Codigo Sql

```
create database App_Servicios;
use App_Servicios;
create table Formas_Pago(
Num_Forma_Pago int primary key,
descripcion nvarchar (40) not null unique,
Beneficios nvarchar (30) not null,
);
create table Categorías_Servicios(
Num_Categoría int primary key,
Nombre nvarchar (20) not null unique,
Descripcion nvarchar (30) not null,
);
create table Permisos(
Num_Permiso int primary key,
Descripcion nvarchar (30) not null unique,
);
create table Estados(
Num_Estado int primary key,
Descripcion nvarchar (30) not null unique,
);
create table Profesionales(
Num_Profesional int primary key,
Dni int not null,
```

```

Email nvarchar (30) not null unique,
Telefono int not null unique,
Nombre nvarchar (25) not null,
Apellido nvarchar (25) not null,
);
create table Credenciales(
Num_credencial int primary key,
Titulo nvarchar (50) not null,
Imagen_Titulo nvarchar (150) not null,
Especializacion nvarchar (40) not null,
Num_Profesional int foreign key references Profesionales (Num_Profesional),

);
create table Usuarios(
Num_Usuario int primary key,
Nombre nvarchar (25) not null,
Apellido nvarchar (25) not null,
Email nvarchar (30) not null unique,
Telefono int not null unique,
);
create table Administradores(
Num_Administrador int primary key,
Dni int not null,
Nombre nvarchar (25) not null,
Apellido nvarchar (25) not null,
Email nvarchar (30) not null unique,
Telefono int not null unique,
);
create table Administradores_Permisos(
primary key (Num_Administrador,Num_Permiso),
Num_Administrador int foreign key references Administradores (Num_Administrador),
Num_Permiso int foreign key references Permisos (Num_Permiso),
);
create table Servicios_Publicaciones(
Num_Servicio int primary key,
Descripcion nvarchar (50) not null,
Num_Categoria int foreign key references Categorias_Servicios (Num_Categoria)
);
create table Publicaciones(
Num_Publicacion int primary key,
unique (Num_Profesional,Num_Servicio),
Num_Profesional int foreign key references Profesionales (Num_Profesional),
Num_Servicio int foreign key references Servicios_Publicaciones (Num_Servicio),
Fecha_Publicacion_Inicial datetime not null,
Tarifa decimal (5,2) not null,
Horas_De_Trabajo decimal (2,1) not null,
);
create table Reseñas(
Num_Reseña int primary key,
Calificacion int not null,
unique (Num_Usuario,Num_Publicacion),
Num_Usuario int foreign key references Usuarios (Num_Usuario),
Comentario nvarchar (100) not null,
Num_Publicacion int foreign key references Publicaciones (Num_Publicacion),
Fecha_Reseña datetime not null,
);
create table Comprobantes_Pago(
Num_Comprobante int primary key,
Num_Forma_Pago int foreign key references Formas_Pago (Num_Forma_Pago),
Fecha_Emision datetime not null,
);
create table Pre_Contratos(
Num_Pre_Contrato int primary key,
Num_Usuario int foreign key references Usuarios (Num_Usuario),
Num_Publicacion int foreign key references Publicaciones (Num_Publicacion),
);
create table Pre_Contratos_Confirmaciones(
Cod_Aceptacion int primary key,
Num_Precontrato int foreign key references Pre_Contratos (Num_Pre_Contrato),
Num_Estado int foreign key references Estados (Num_Estado),
Fecha_Aceptacion datetime,
);
create table Administradores_Cambios(
Num_Administrador int foreign key references Administradores (Num_Administrador),
Cambio_Realizado nvarchar (200) not null,
Fecha_Cambio datetime not null,
);
create table Contratos(
Num_Contrato int primary key,
Num_Comprobante int foreign key references Comprobantes_Pago (Num_Comprobante) not null unique,
Num_Profesional int foreign key references Profesionales (Num_Profesional),
Num_Usuario int foreign key references Usuarios (Num_Usuario),
Num_Publicacion int foreign key references Publicaciones (Num_Publicacion),
Num_Pre_Contrato int foreign key references Pre_Contratos (Num_Pre_Contrato),
Fecha_Generacion datetime not null,
);
create table Reclamos(
Num_reclamo int primary key,

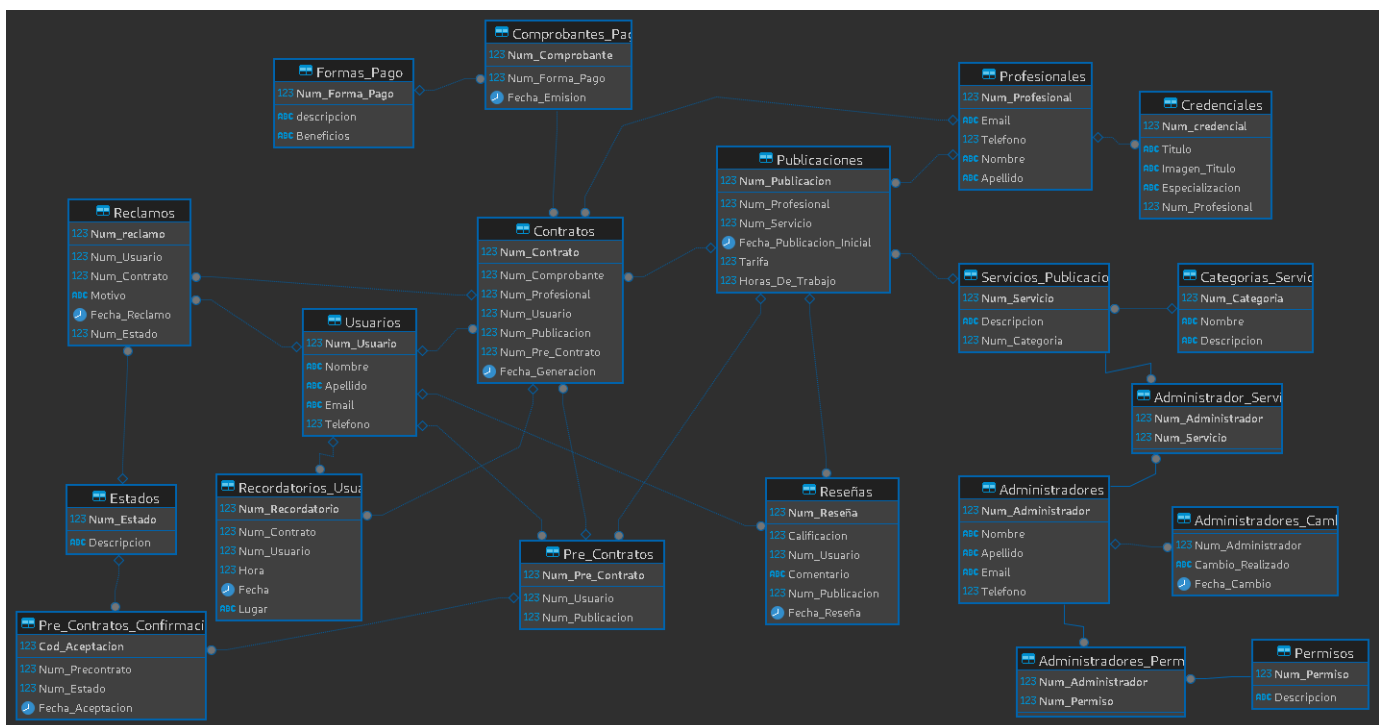
```

```

unique (Num_Usuario,Num_Contrato),
Num_Usuario int foreign key references Usuarios (Num_Usuario),
Num_Contrato int foreign key references Contratos (Num_Contrato),
Motivo nvarchar (100) not null,
Fecha_Reclamo datetime not null,
Num_Estado int foreign key references Estados (Num_Estado),
);
create table Administrador_Servicio(
Primary key (Num_Administrador,Num_Servicio),
Num_Administrador int foreign key references Administradores (Num_Administrador),
Num_Servicio int foreign key references Servicios_Publicaciones (Num_Servicio)
);
create table Recordatorios_Usuario(
Num_Recordatorio int primary key,
Num_Contrato int foreign key references Contratos (Num_Contrato),
Num_Usuario int foreign key references Usuarios (Num_Usuario),
Hora decimal (10,2) not null,
Fecha datetime not null,
Lugar nvarchar (50) not null
);

```

6. Diagrama Pata de gallo



7. Marco Teorico Normalizacion

La normalización es un proceso fundamental en el diseño de bases de datos que tiene como objetivo reducir la redundancia y mejorar la integridad de los datos. Este proceso implica organizar los datos en tablas y columnas para asegurar que las dependencias entre datos estén correctamente definidas y las anomalías de actualización se minimicen. Las formas normales (NF) son los estándares utilizados para evaluar el nivel de normalización de una base de datos.

A continuación, desarrollaremos las cinco formas normales (1FN, 2FN, 3FN, 4FN y 5FN) aplicadas a nuestro proyecto para poder tener una visión más práctica de lo que las mismas implican.

7.1. Primera Forma Normal (1FN)

La Primera Forma Normal establece que los datos deben estar organizados en tablas de manera que cada columna contenga valores atómicos, y cada fila sea única.

Por ejemplo, la tabla 'Profesionales' tiene columnas con valores indivisibles y cada campo contiene solo un valor y no se permite la repetición de filas.

```

create table Profesionales(
Num_Profesional int primary key,
Email nvarchar (30) not null unique,
Telefono int not null unique,
Nombre nvarchar (25) not null,

```

```
Apellido nvarchar (25) not null,  
DNI int not null unique  
);
```

7.2. Segunda Forma Normal (2FN)

La Segunda Forma Normal requiere que la base de datos cumpla con 1FN y que todos los atributos no clave dependan completamente de la clave primaria. Esto significa que no debe haber dependencias parciales en una tabla. Las tablas como `Credenciales` cumplen con esta forma normal, ya que todos sus atributos dependen completamente de la clave primaria `Num_credencial`.

```
create table Credenciales(  
    Num_credencial int primary key,  
    Titulo nvarchar (50) not null,  
    Imagen_Titulo nvarchar (150) not null,  
    Especializacion nvarchar (40) not null,  
    Num_Profesional int foreign key references Profesionales (Num_Profesional)  
);
```

7.3. Tercera Forma Normal (3FN)

La Tercera Forma Normal requiere que la base de datos cumpla con 2FN y que no haya dependencias transitivas, es decir, los atributos no clave deben depender solo de la clave primaria y no de otros atributos no clave. Las tablas como `Publicaciones` cumplen con esta forma normal, ya que todos sus atributos dependen directamente de la clave primaria `Num_Publicacion`.

```
create table Publicaciones(  
    Num_Publicacion int primary key,  
    unique (Num_Profesional, Num_Servicio),  
    Num_Profesional int foreign key references Profesionales (Num_Profesional),  
    Num_Servicio int foreign key references Servicios_Publicaciones (Num_Servicio),  
    Fecha_Publicacion_Inicial datetime not null,  
    Tarifa decimal (5,2) not null,  
    Horas_De_Trabajo decimal (2,1) not null  
);
```

7.4. Cuarta Forma Normal (4FN)

La Cuarta Forma Normal se ocupa de las dependencias multivaluadas. Una tabla está en 4FN si está en 3FN y no tiene dependencias multivaluadas no triviales. En nuestro caso las tablas no presentan dependencias multivaluadas no triviales, lo que significa que cumplen con esta forma normal. Por ejemplo, la tabla `Administradores_Permisos` maneja correctamente las relaciones muchos a muchos sin introducir redundancias indebidas.

```
create table Administradores_Permisos(  
    primary key (Num_Administrador, Num_Permiso),  
    Num_Administrador int foreign key references Administradores (Num_Administrador),  
    Num_Permiso int foreign key references Permisos (Num_Permiso)  
);
```

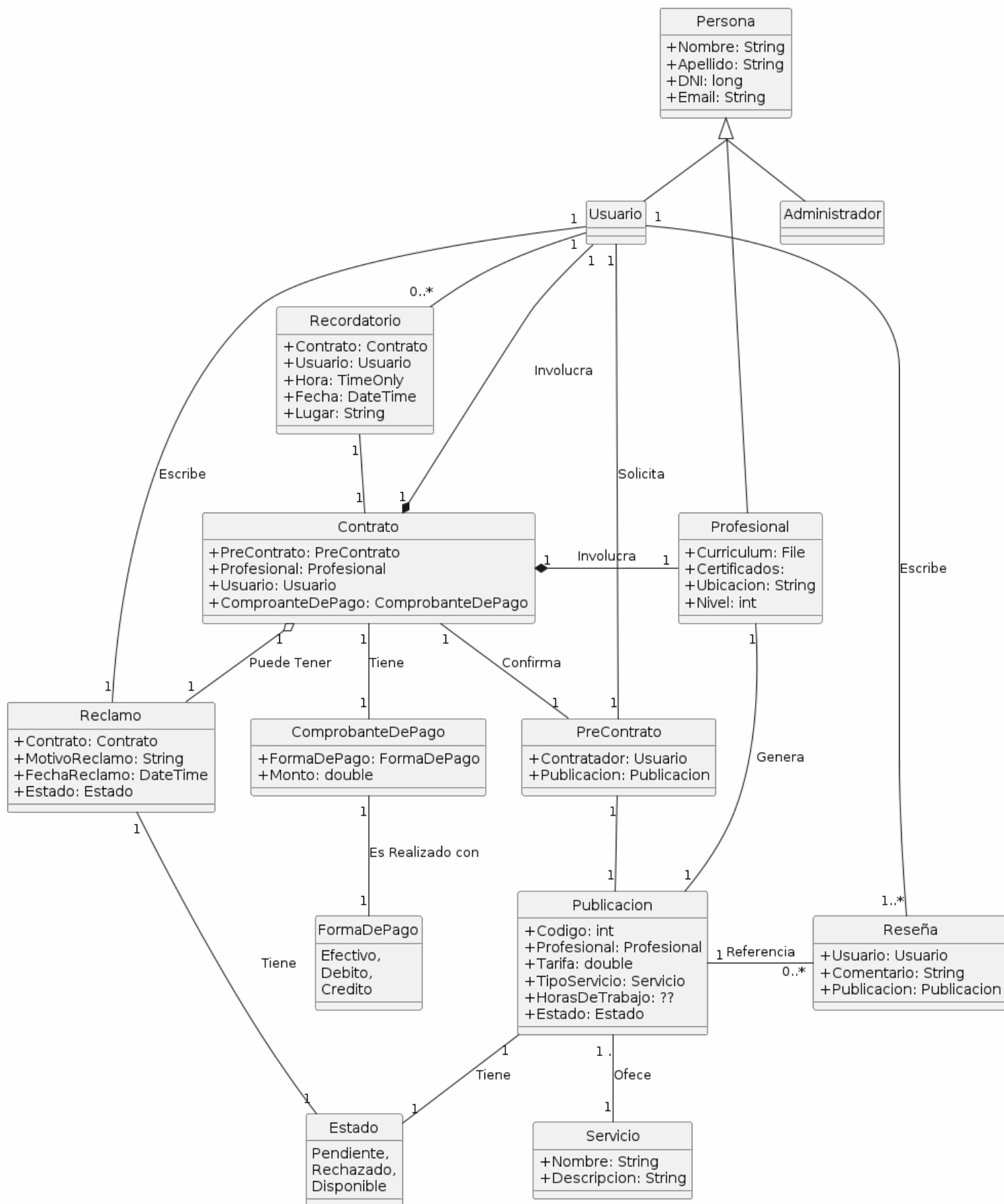
7.5. Quinta Forma Normal (5FN)

La Quinta Forma Normal requiere que la base de datos esté en 4FN y que cualquier dependencia de unión no trivial esté implícita en las claves candidatas. Esto significa que una tabla no debe poder descomponerse en dos o más tablas más pequeñas que se puedan unir sin pérdida de información.

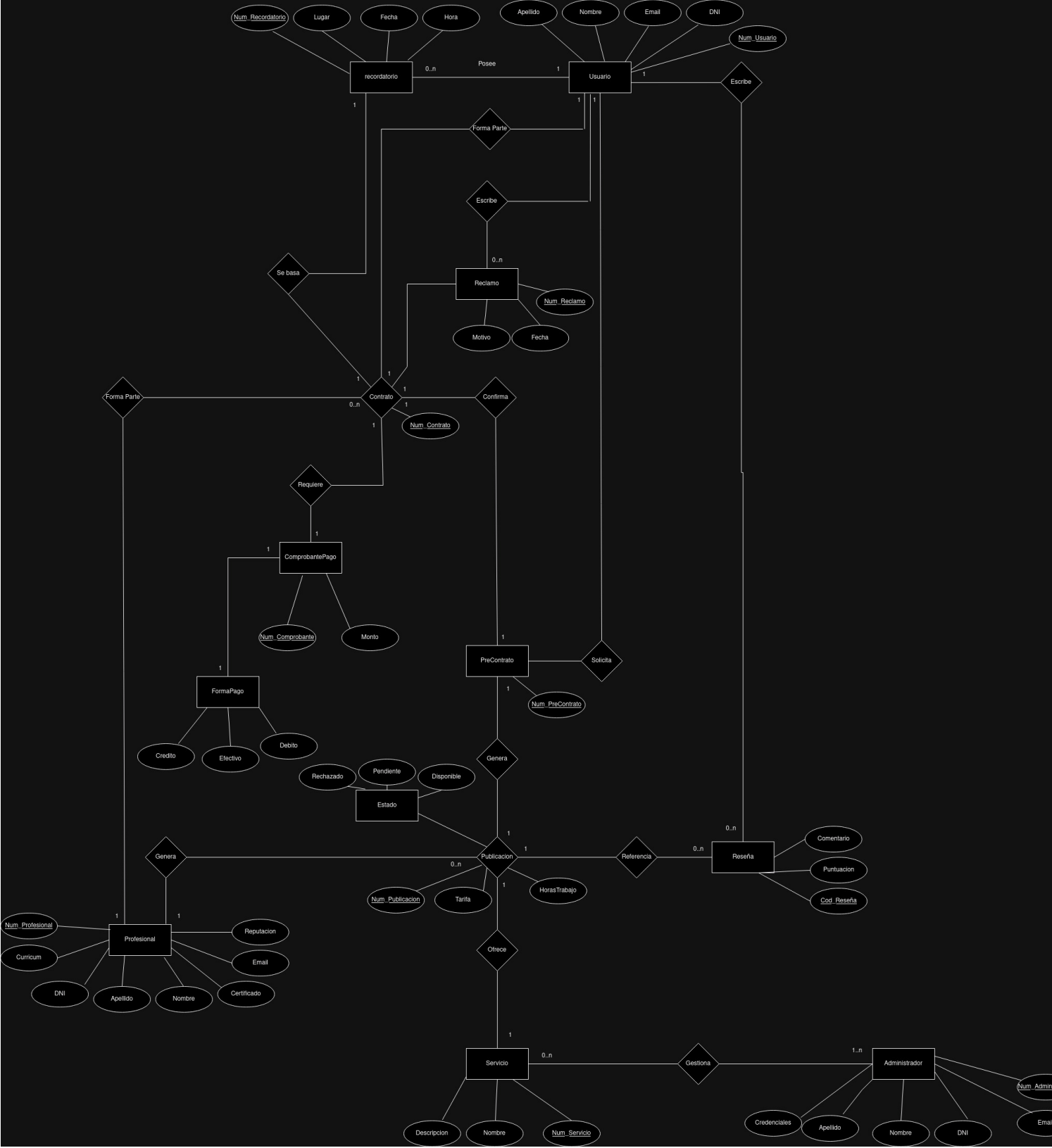
Por ejemplo, la tabla `Contratos` no puede descomponerse sin pérdida de información:

```
create table Contratos(  
    Num_Contrato int primary key,  
    Fecha_Generacion datetime not null,  
    Num_Comprobante int foreign key references Comprobantes_Pago (Num_Comprobante) not null unique,  
    Num_Profesional int foreign key references Profesionales (Num_Profesional),  
    Num_Usuario int foreign key references Usuarios (Num_Usuario),  
    Num_Publicacion int foreign key references Publicaciones (Num_Publicacion),  
    Num_Pre_Contrato int foreign key references Pre_Contratos (Num_Pre_Contrato)  
);
```

8. Diagrama de clases



9. Diagrama Entidad-Relacion



Author: Grupo 1
Created: 2024-07-07 dom 20:02