

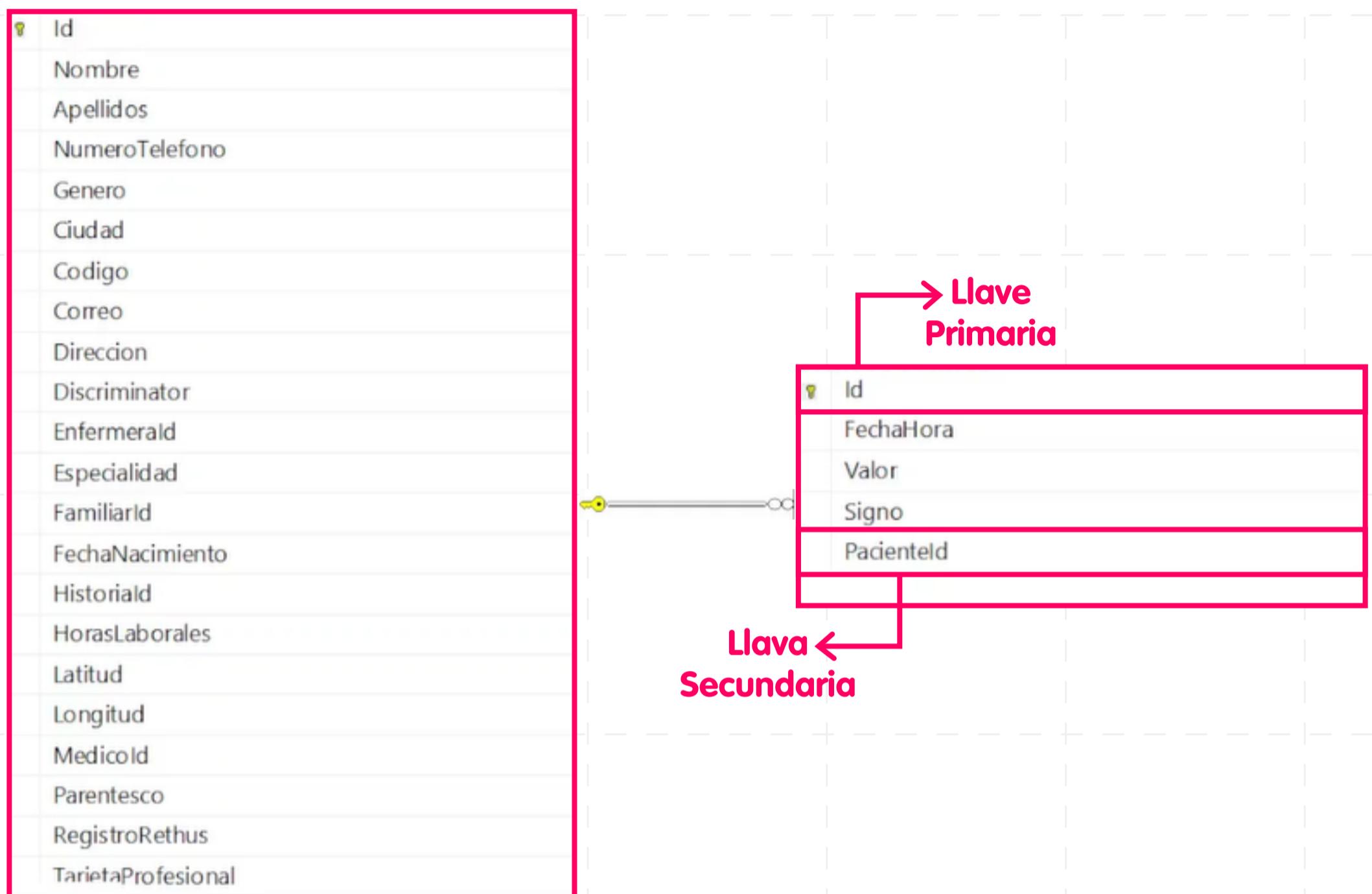
# Entidades Relacionadas Maestro-Detalle

# Hola:

El patrón maestro-detalle es bastante utilizado en el desarrollo de aplicaciones web y para implementarlo se deben tener en cuenta diferentes aspectos en todas las capas arquitectónicas de una aplicación. En un patrón maestro-detalle se tiene la entidad Paciente, que tiene el rol de maestro al que se le pueden asignar varios signos vitales a través de la entidad Detalle. En el caso de ejemplo, el modelo establece que serán necesarios mínimo 5 signos vitales para cada Paciente. Este diseño de patrón maestro-detalle es una relación de entidades de “uno a muchos”. Sin embargo, en esta ocasión se realizará la implementación de las 3 capas de la arquitectura, es decir, en la capa de presentación, la capa de dominio y en la capa de persistencia.



En las tablas la relación maestro-detalle se da de la siguiente manera: En la tabla de signos vitales se encuentra el identificador paciente y en la tabla de signos vitales se encuentran todos los signos de todos los pacientes y la forma de identificarlos es con el ID del paciente al que pertenecen. Este campo de la tabla se conoce como una llave secundaria, ya que permite identificar los registros en la otra tabla, para este caso, la tabla personas. La llave principal de la tabla signos vitales es su identificador, pero el paciente ID es su llave secundaria, que hace referencia a la tabla personas.



La implementación de la relación maestro-detalle en la capa del dominio entre la entidad Paciencia y la entidad signos vitales se define con una propiedad de tipo lista, que significa que cada paciente tiene o está relacionado con muchos signos vitales. En la clase persona se declara una lista de signos vitales. Esto significa que persona, en este caso paciente, hereda una lista de signos vitales.

```
public class Paciente : Persona
{
    /**
     * 
     * 
     * 
     *     Referencia a la lista de signos vitales de un Paciente
     * 
     */
    public List<SignoVital> SignosVitales { get; set; }
}
```

La forma de representarlo en el *frontend* es la siguiente: el maestro es un formulario donde se muestran los datos del Paciente seleccionado y, en este caso, en la parte inferior hay un botón llamado ‘signos vitales’, que despliega otro formulario web con la lista de signos vitales que le pertenecen al determinado paciente.

## Edición de los datos del Paciente

Nombre

Carmenza

Apellidos

Zuluaga

NumeroTelefono

5001646

Ciudad

Pereira

Direccion

Calle 908 No Xy-40

Grabar

Signos Vitales

## Listado de Signos Vitales

TemperaturaCorporal	36	12/09/2021 18:50:00
SaturacionOxigeno	95	12/09/2021 18:50:00
FrecuenciaCardica	90	12/09/2021 18:50:00

Regresar

La representación de la información es diferente en cada capa y hacer la transformación e intercambio de una capa a la otra se hace de la siguiente manera: asp.net core trabaja directamente con el *entity framework* que se encarga del mapeo de entidades a tablas. En la búsqueda de un paciente por su identificador y con la instrucción *include* se le está indicando que también cargue los signos vitales de ese paciente.

```
IEnumerable<SignoVital> IRepositoryPaciente.GetSignosPaciente(int idPaciente)
{
    var paciente = _appContext.Pacientes.Where(s => s.Id==idPaciente)
        .Include(s=>s.SignosVitales)
        .FirstOrDefault();

    return paciente.SignosVitales;
}
```

El frontend es de especial atención, ya que allí es donde se deben enlazar los datos para garantizar que se permita este enlace cuando se pasa de un formulario a otro. Un ejemplo sería el siguiente: en la implementación del botón del formulario paciente se le debe indicar que se debe dirigir a la página signos/list, además de asignar el identificador del paciente. El formulario listará los signos vitales y, a su vez, tomará este parámetro al definir en la clase lista el enlace a Paciente y recibir el identificador con el método onget.

En conclusión, se ha presentado el patrón maestro-detalie y la forma en que se representa en las diferentes capas de la arquitectura. También lo que es el entity framework y como este es el que se encarga de gran parte del trabajo.



Universidad de Caldas