Guía de ejercicios - Relaciones 1 a N en los modelos

 ¡Hola! Te damos la bienvenida a esta nueva guía de estudio.

¿En qué consiste esta guía?

La siguiente guía de estudio tiene como objetivo practicar y ejercitar los contenidos que hemos visto en clase. Trabajaremos en una nueva aplicación llamada Centro médico, basada en un proyecto real. Empezaremos desde 0 para repasar conceptos que pueden estar colgando hacia el abismo del olvido.

¡Vamos con todo!

Tabla de contenidos

¡Comencemos!

Actividad guiada: Centro médico

A continuación, empezaremos a trabajar el proyecto de Centro médico. Daniela tiene muy mala memoria y siempre olvida sus notas en casa, por lo que te ha pedido desarrollar una aplicación para poder llevar un registro de los tratamientos que le ha asignado a un paciente, para una primera entrega nos da los siguientes requisitos:

- Diseño amigable para los niños (Nunca hay que dejar de lado el front-end de una aplicación, muchas veces por muy complicado y lujoso que pueda ser el back-end, si la vista no es lo primero que se conquista el proyecto puede no llegar a buen puerto).
- Crear pacientes a los cuales les asignará un tratamiento.
- Mostrar el último tratamiento al cual ha sido expuesto el paciente.
- Al borrar un paciente, su tratamiento, al ser personalizado, debe borrarse de la lista de intervenciones que ha hecho el médico.

Desarrollo:

- **Paso 1:** Creamos la aplicación llamada `MedicalCenter` con base de datos PostgreSql.

```
rails new MedicalCenter -d postgresql
```

- **Paso 2:** Creamos la base de datos.

```
rails db:create
```

- **Paso 3:** Identificamos los modelos a integrar.

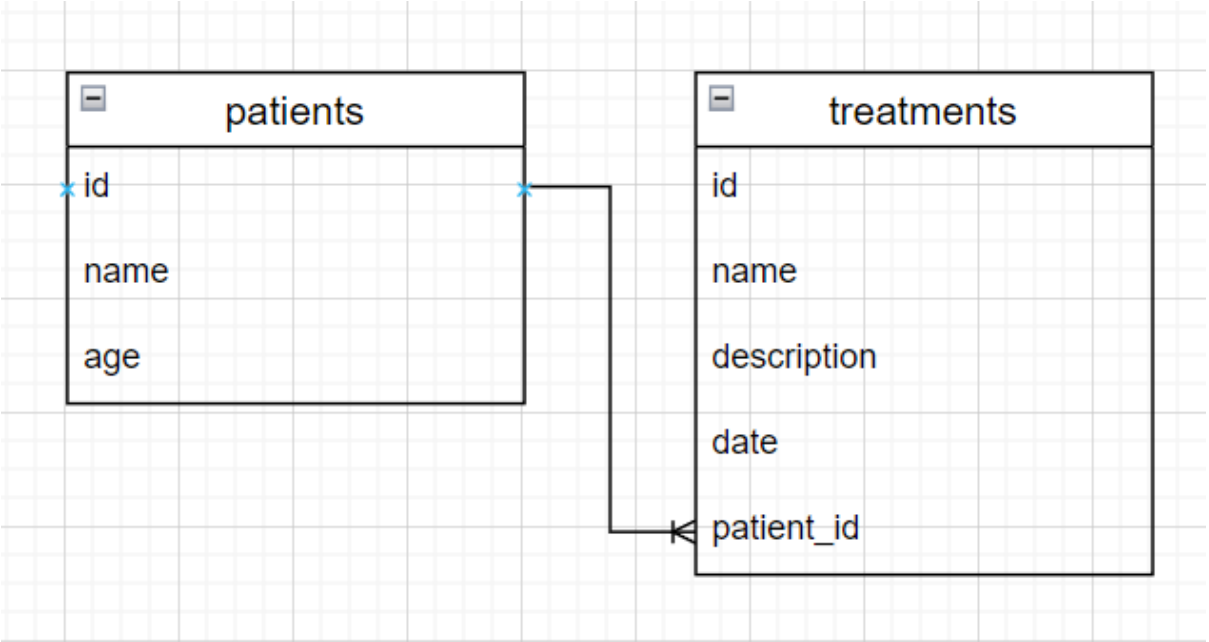

Imagen 1. Modelos identificados
Fuente: Desafío Latam

- **Paso 4:** Creamos modelos en base de datos mediante scaffold (opcional, puede ser hecho a mano si el estudiante lo desea).

    a. Realizamos scaffold de pacientes.

```
rails g scaffold patient name age:integer
```

    b. Realizamos scaffold de tratamientos

```
rails g scaffold treatment name description:text date:date
```

    c. Agregamos la clave foránea a tratamientos

```
rails g migration AddPatientToTreatments patient:references
```

    d. Corremos las migraciones en base de datos

```
rails db:migrate
```

- **Paso 5:** Agregamos Bootstrap a nuestro proyecto.
    a. Como prerrequisito debemos tener instalado yarn. Puedes revisar este tutorial de instalación.
    b. Instalamos la gema 'cssbundling-rails' dentro de nuestro gemfile. Link de Github

```
bundle add cssbundling-rails
```

    c. Utilizamos el comando dentro de la documentación para instalar bootstrap. (En este punto podemos utilizar las apariencias de bootstrap, pero dependencias de Jquery como dropdown no funcionarían).

```
rails css:install:bootstrap
```

    d. Instalamos la gema 'jsbundling-rails' dentro de nuestro gemfile. Link de Github

```
bundle add jsbundling-rails
```

    e. Utilizamos el comando dentro de la documentación para instalar esbuild. (ESBuild es el empaquetador más recientemente popular de JavaScript. Su principal característica y ventaja frente al resto de empaquetadores es su velocidad de compilación)

```
rails javascript:install:esbuild
```

    f. Arreglamos la instalación de la gema 'jsbundling-rails'.
        - Primero debemos instalar los paquetes de 'turbo-rails' y 'stimulus'.

```
yarn add @hotwired/turbo-rails
```

```
yarn add @hotwired/stimulus
```

        - Ajustaremos el path de 'application.js' y cambiaremos las llamadas antiguas de 'stimulus' en index.js

```
app/javascript/application.js
// Configure your import map in config/importmap.rb. Read more:
```

```
https://github.com/rails/importmap-rails
import "@hotwired/turbo-rails"
import "./controllers"
import * as bootstrap from "bootstrap"
```

```
app/javascript/controllers/index.js
// Import and register all your controllers from the importmap under
controllers/*

// import { application } from "controllers/application"

// Eager load all controllers defined in the import map under
controllers/**/*_controller
// import { eagerLoadControllersFrom } from "@hotwired/stimulus-loading"
// eagerLoadControllersFrom("controllers", application)

// Lazy load controllers as they appear in the DOM (remember not to
preload controllers in import map!)
// import { lazyLoadControllersFrom } from "@hotwired/stimulus-loading"
// lazyLoadControllersFrom("controllers", application)
import { application } from "./application";
```

- En 'application.html.erb' sacamos 'javascript_importmap_tags', ya que no es necesario.

```
app/views/layouts/application.html.erb
<!DOCTYPE html>
<html>
  <head>
    <title>MedicalCenter</title>
    <meta name="viewport" content="width=device-width,initial-scale=1">
    <%= csrf_meta_tags %>
    <%= csp_meta_tag %>

    <%= stylesheet_link_tag "application", "data-turbo-track": "reload"
%>
    <%= javascript_include_tag "application", "data-turbo-track":
"reload", defer: true %>
  </head>

  <body>
    <%= yield %>
  </body>
```

```
</html>
```

- Por último, en 'manifest.js' solo dejaremos 'builds' e 'images'.

```
app/assets/config/manifest.js
//= link_tree ../images
//= link_tree ../builds
```

g. Utilizamos el comando dentro de la documentación para instalar esbuild nuevamente.

```
rails javascript:install:esbuild
```

- **Paso 6:** Modificamos la vista parcial de paciente y tratamiento. (Iconos sacados desde tablericons)
  a. index y vista parcial de paciente

```
#app/views/patients/index.html.erb
<p style="color: green"><%= notice %></p>

<div class="container">

<h1>Patients</h1>
  <div id="patients">
    <table class="table table-striped">
      <thead>
        <tr>
          <th scope="col">id</th>
          <th scope="col">Name</th>
          <th scope="col">Age</th>
          <th scope="col" colspan="2"></th>
        </tr>
      </thead>
      <tbody>
          <% @patients.each do |patient| %>
            <%= render patient %>
          <% end %>
      </tbody>
    </table>

    <%= link_to new_patient_path do %>
```

```
        <svg xmlns="http://www.w3.org/2000/svg" class="icon icon-tabler
icon-tabler-circle-plus" width="32" height="32" viewBox="0 0 24 24"
stroke-width="1.5" stroke="#009988" fill="none" stroke-linecap="round"
stroke-linejoin="round">
          <path stroke="none" d="M0 0h24v24H0z" fill="none"/>
          <circle cx="12" cy="12" r="9" />
          <line x1="9" y1="12" x2="15" y2="12" />
          <line x1="12" y1="9" x2="12" y2="15" />
        </svg>
    <% end %>
  </div>
</div>
```

```
#app/views/patients/_patient.html.erb
<div id="<%= dom_id patient %>">
  <tr>
    <th scope="row"><%= patient.id %></th>
    <td><%= patient.name %></td>
    <td><%= patient.age %></td>
    <td><%= link_to patient, class: 'btn' do  %>
          <svg xmlns="http://www.w3.org/2000/svg" class="icon
icon-tabler icon-tabler-eye-check" width="20" height="20" viewBox="0 0
24 24" stroke-width="1.5" stroke="#00abfb" fill="none"
stroke-linecap="round" stroke-linejoin="round">
            <path stroke="none" d="M0 0h24v24H0z" fill="none"/>
            <circle cx="12" cy="12" r="2" />
            <path d="M12 19c-4 0 -7.333 -2.333 -10 -7c2.667 -4.667 6 -7
10 -7s7.333 2.333 10 7c-.42 .736 -.858 1.414 -1.311 2.033" />
            <path d="M15 19l2 2l4 -4" />
          </svg>
        <% end %>
    </td>
    <td><%= link_to edit_patient_path(patient), class: 'btn' do %>
        <svg xmlns="http://www.w3.org/2000/svg" class="icon icon-tabler
icon-tabler-edit" width="20" height="20" viewBox="0 0 24 24"
stroke-width="1.5" stroke="#ffec00" fill="none" stroke-linecap="round"
stroke-linejoin="round">
          <path stroke="none" d="M0 0h24v24H0z" fill="none"/>
          <path d="M9 7h-3a2 2 0 0 0 -2 2v9a2 2 0 0 0 2 2h9a2 2 0 0 0 2
-2v-3" />
          <path d="M9 15h3l8.5 -8.5a1.5 1.5 0 0 0 -3 -3l-8.5 8.5v3" />
          <line x1="16" y1="5" x2="19" y2="8" />
        </svg>
```

```
      <% end %>
    </td>
    <td><%= button_to patient, method: :delete, class: 'btn' do %>
        <svg xmlns="http://www.w3.org/2000/svg" class="icon icon-tabler
icon-tabler-eraser" width="20" height="20" viewBox="0 0 24 24"
stroke-width="1.5" stroke="#ff4500" fill="none" stroke-linecap="round"
stroke-linejoin="round">
          <path stroke="none" d="M0 0h24v24H0z" fill="none"/>
          <path d="M19 19h-11l-4 -4a1 1 0 0 1 0 -1.41l10 -10a1 1 0 0 1
1.41 0l5 5a1 1 0 0 1 0 1.41l-9 9" />
          <line x1="18" y1="12.3" x2="11.7" y2="6" />
        </svg>
      <% end %>
    </td>
  </tr>
</div>
```

b. index y vista parcial de tratamiento

```
#app/views/treatments/index.html.erb
<p style="color: green"><%= notice %></p>

<div class="container">
  <h1>Treatments</h1>

<div id="treatments">
  <table class="table table-striped">
      <thead>
        <tr>
          <th scope="col">id</th>
          <th scope="col">Name</th>
          <th scope="col">Description</th>
          <th scope="col">Date</th>
          <th scope="col">Patient</th>
          <th scope="col" colspan="2"></th>
        </tr>
      </thead>
      <tbody>
    <% @treatments.each do |treatment| %>
      <%= render treatment %>
    <% end %>
    </tbody>
  </table>
```

```
</div>

  <%= link_to new_treatment_path do %>
  <svg xmlns="http://www.w3.org/2000/svg" class="icon icon-tabler
icon-tabler-circle-plus" width="32" height="32" viewBox="0 0 24 24"
stroke-width="1.5" stroke="#009988" fill="none" stroke-linecap="round"
stroke-linejoin="round">
    <path stroke="none" d="M0 0h24v24H0z" fill="none"/>
    <circle cx="12" cy="12" r="9" />
    <line x1="9" y1="12" x2="15" y2="12" />
    <line x1="12" y1="9" x2="12" y2="15" />
  </svg>
  <% end %>

</div>
```

```
#app/views/treatments/_treatment.html.erb
<div id="<%= dom_id treatment %>">
  <tr>
    <td><%= treatment.id %></td>
    <td><%= treatment.name %></td>
    <td><%= treatment.description %></td>
    <td><%= treatment.date %></td>
    <td><%= treatment.patient.name %></td>
    <td><%= link_to treatment, class: 'btn' do  %>
          <svg xmlns="http://www.w3.org/2000/svg" class="icon
icon-tabler icon-tabler-eye-check" width="20" height="20" viewBox="0 0
24 24" stroke-width="1.5" stroke="#00abfb" fill="none"
stroke-linecap="round" stroke-linejoin="round">
            <path stroke="none" d="M0 0h24v24H0z" fill="none"/>
            <circle cx="12" cy="12" r="2" />
            <path d="M12 19c-4 0 -7.333 -2.333 -10 -7c2.667 -4.667 6 -7
10 -7s7.333 2.333 10 7c-.42 .736 -.858 1.414 -1.311 2.033" />
            <path d="M15 19l2 2l4 -4" />
          </svg>
        <% end %>
    </td>
    <td><%= link_to edit_treatment_path(treatment), class: 'btn' do %>
        <svg xmlns="http://www.w3.org/2000/svg" class="icon icon-tabler
icon-tabler-edit" width="20" height="20" viewBox="0 0 24 24"
stroke-width="1.5" stroke="#ffec00" fill="none" stroke-linecap="round"
stroke-linejoin="round">
          <path stroke="none" d="M0 0h24v24H0z" fill="none"/>
```

```
          <path d="M9 7h-3a2 2 0 0 0 -2 2v9a2 2 0 0 0 2 2h9a2 2 0 0 0 2
-2v-3" />
          <path d="M9 15h3l8.5 -8.5a1.5 1.5 0 0 0 -3 -3l-8.5 8.5v3" />
          <line x1="16" y1="5" x2="19" y2="8" />
        </svg>
      <% end %>
    </td>
    <td><%= button_to treatment, method: :delete, class: 'btn' do %>
        <svg xmlns="http://www.w3.org/2000/svg" class="icon icon-tabler
icon-tabler-eraser" width="20" height="20" viewBox="0 0 24 24"
stroke-width="1.5" stroke="#ff4500" fill="none" stroke-linecap="round"
stroke-linejoin="round">
          <path stroke="none" d="M0 0h24v24H0z" fill="none"/>
          <path d="M19 19h-11l-4 -4a1 1 0 0 1 0 -1.41l10 -10a1 1 0 0 1
1.41 0l5 5a1 1 0 0 1 0 1.41l-9 9" />
          <line x1="18" y1="12.3" x2="11.7" y2="6" />
        </svg>
      <% end %>
    </td>
  </tr>
</div>
```

- **Paso 7:** Definimos una ruta root.

```
#config/routes.rb
root "patients#index"
```

- **Paso 8:** Agregamos los métodos de relación a los modelos.

```
class Patient < ApplicationRecord
  has_many :treatments
end
```

```
class Treatment < ApplicationRecord
  belongs_to :patient
end
```

- **Paso 9:** Probamos relación en Rails console.

```
❯ rails c
Loading development environment (Rails 7.0.4)
```

```
3.1.1 :001 > Patient.new.treatments
 => []
3.1.1 :002 > Treatment.new.patient
 => nil
3.1.1 :003 >
```

- **Paso 10:** Al momento de crear o editar un tratemiento, asignaremos un paciente.
    a. Agregamos 'patient_id' a los 'strong params' de tratamientos.

```
def treatment_params
      params.require(:treatment).permit(:name, :description, :date,
:patient_id)
end
```

    b. Agregamos un select al formulario de tratamiento con los pacientes

```
<div>
    <%= form.label :patient_id, style: "display: block" %>
    <%= form.select :patient_id, @patients %>
</div>
```

    c. Agregamos '@patients' a los controladores necesarios

```
def new
    @treatment = Treatment.new
    @patients = Patient.all.pluck :name, :id
  end

  # GET /treatments/1/edit
  def edit
    @patients = Patient.all.pluck :name, :id
  end

  # POST /treatments or /treatments.json
  def create
    @treatment = Treatment.new(treatment_params)
    @patients = Patient.all.pluck :name, :id
    respond_to do |format|
      if @treatment.save
        format.html { redirect_to treatment_url(@treatment), notice:
"Treatment was successfully created." }
        format.json { render :show, status: :created, location:
@treatment }
      else
```

```
        format.html { render :new, status: :unprocessable_entity }
        format.json { render json: @treatment.errors, status:
:unprocessable_entity }
      end
    end
  end

  # PATCH/PUT /treatments/1 or /treatments/1.json
  def update
    @patients = Patient.all.pluck :name, :id
    respond_to do |format|
      if @treatment.update(treatment_params)
        format.html { redirect_to treatment_url(@treatment), notice:
"Treatment was successfully updated." }
        format.json { render :show, status: :ok, location: @treatment }
      else
        format.html { render :edit, status: :unprocessable_entity }
        format.json { render json: @treatment.errors, status:
:unprocessable_entity }
      end
    end
  end

  # DELETE /treatments/1 or /treatments/1.json
  def destroy
    @patients = Patient.all.pluck :name, :id
    @treatment.destroy

    respond_to do |format|
      format.html { redirect_to treatments_url, notice: "Treatment was
successfully destroyed." }
      format.json { head :no_content }
    end
  end
```

- **Paso 11:** En la vista `index` de pacientes mostraremos el último tratamiento dado al paciente.
    a. Agregamos 'Treatment' a las cabeceras de la tabla

```
<thead>
        <tr>
          <th scope="col">id</th>
          <th scope="col">Name</th>
          <th scope="col">Age</th>
```

```
          <th scope="col">Treatment</th>
          <th scope="col" colspan="3"></th>
        </tr>
</thead>
```

b. Añadimos el último tratamiento a la vista parcial.

```
<td><%= patient.treatments.last.name %></td>
```

Ya tenemos un registro funcional ¡Felicidades!

En la plataforma tendrás acceso al código de este ejercicio con el nombre "Códigos - Actividad guiada: Centro médico".

¡Manos a la obra! - Requerimientos de última hora

Daniela nos ha pedido que al ingresar al perfil de un paciente, podamos mostrar su último tratamiento y en caso de equivocarse poder editarlo.

- Editar vista show del paciente para mostrar su último tratamiento.
- Crear link al formulario de edición del tratamiento en caso de errores al ingresarlo.

¡Manos a la obra! - ¡Quiero verlo!

Daniela nos ha dicho que le gustó mucho nuestra implementación, pero que necesita tener la aplicación en su computador para hacer pruebas.

- Realizar deploy a heroku.

Solución Manos a la Obra 1

- Editar vista `show` del paciente para mostrar su último tratamiento.

Simplemente, modificamos un poco la vista show dándole formato al igual que el index de una tabla.

```
<p style="color: green"><%= notice %></p>

<table class="table table-striped">
  <thead>
    <tr>
      <th scope="col">id</th>
      <th scope="col">Name</th>
      <th scope="col">Age</th>
      <th scope="col">Treatment</th>
      <th scope="col" colspan="3"></th>
    </tr>
  </thead>
  <tbody>
        <%= render @patient %>
  </tbody>
</table>
```

- Crear link al formulario de edición del tratamiento en caso de errores al ingresarlo.

Debemos cambiar el parcial de paciente agregándole un link para ir al edit de tratamientos.

```
<td><%= patient.treatments.last.name %>
      <%= link_to edit_treatment_path(patient.treatments.last), class:
'btn' do %>
        <svg xmlns="http://www.w3.org/2000/svg" class="icon icon-tabler
icon-tabler-edit" width="20" height="20" viewBox="0 0 24 24"
stroke-width="1.5" stroke="#ffec00" fill="none" stroke-linecap="round"
stroke-linejoin="round">
          <path stroke="none" d="M0 0h24v24H0z" fill="none"/>
          <path d="M9 7h-3a2 2 0 0 0 -2 2v9a2 2 0 0 0 2 2h9a2 2 0 0 0 2
-2v-3" />
          <path d="M9 15h3l8.5 -8.5a1.5 1.5 0 0 0 -3 -3l-8.5 8.5v3" />
          <line x1="16" y1="5" x2="19" y2="8" />
        </svg>
      <% end %>
```

```
    </td>
```

Solución Manos a la Obra 2

- Realizar deploy a heroku.
    a. Tenemos que logearnos en heroku

```
heroku login
```

    b. Creamos una aplicación

```
heroku create
```

    c. Subimos los cambios a heroku

```
git push heroku main
```

    d. Corremos las migraciones para el correcto funcionamiento

```
heroku run rake db:migrate
```

Preguntas de proceso

Reflexiona:

- ¿Qué te pareció el ejercicio MedicalCenter?
- ¿Por qué crees que esta versión más complicada de agregar bootstrap es la más popular?
- ¿Crees que es estrictamente necesario utilizar esta forma de integración?

Preguntas de cierre

- ¿Por qué para ir a la función destruir se debe usar un botón y no un link?
- ¿Qué diferencia tiene el `bundle add` con agregar la gema al `gemfile`?
- ¿Qué es `yarn`?
- ¿Es posible reemplazar `@patients` con un método de `set_patient`?