

Ingeniería de Software

RoR - Active Record

Juan Pablo Sandoval

RoadMap

- *API - Estudiantes*
 - *crear, actualizar y borrar*
- *Validadores*
- *Asociaciones: ejemplo 1 a n*
- *Otras asociaciones*

RoadMap

- *API - Estudiantes*
 - ***Crear, Actualizar y orrar***
- *Validadores*
- *Asociaciones: ejemplo 1 a n*
- *Otras asociaciones*

create

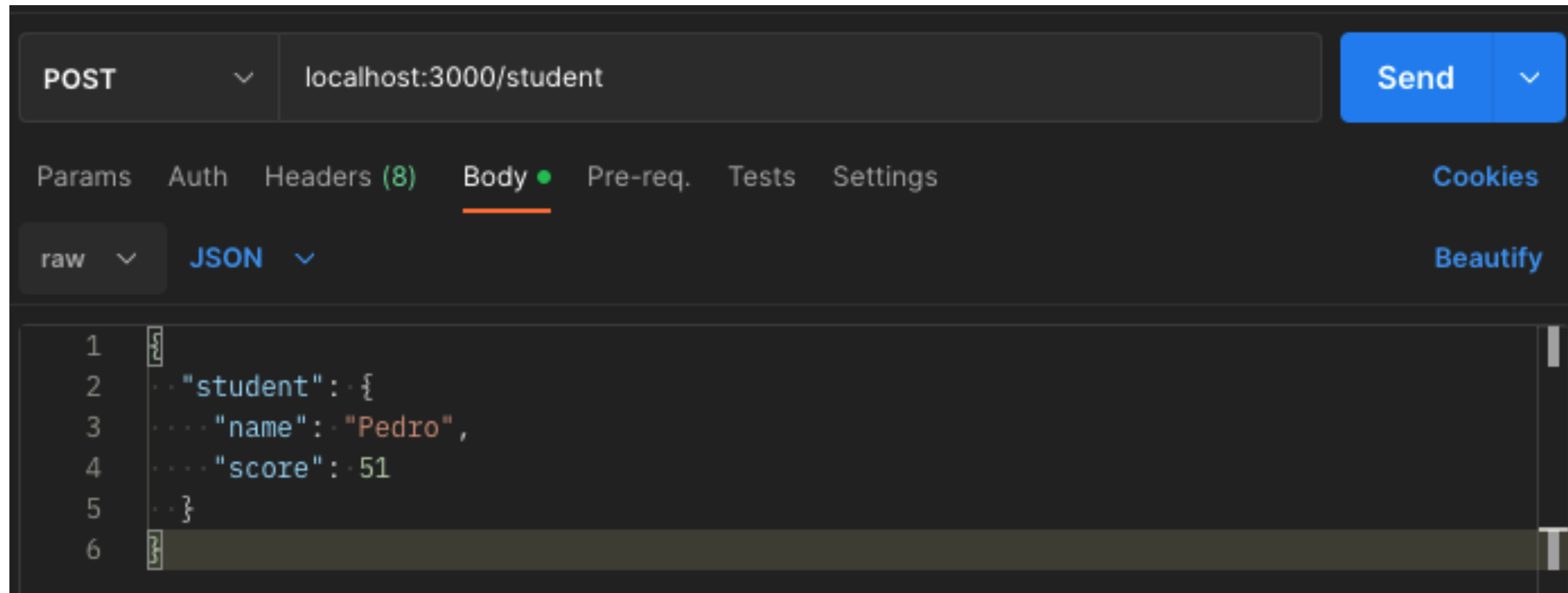
```
class StudentController < ApplicationController
  def create
    @student = Student.new(student_params)
    if @student.save
      render json: @student
    else
      render json: @student.errors, status: :unprocessable_entity
    end
  end

  def student_params
    params.require(:student).permit(:name, :score)
  end
end
```

```
Rails.application.routes.draw do
  ...
  post '/student', to: 'student#create'
end
```

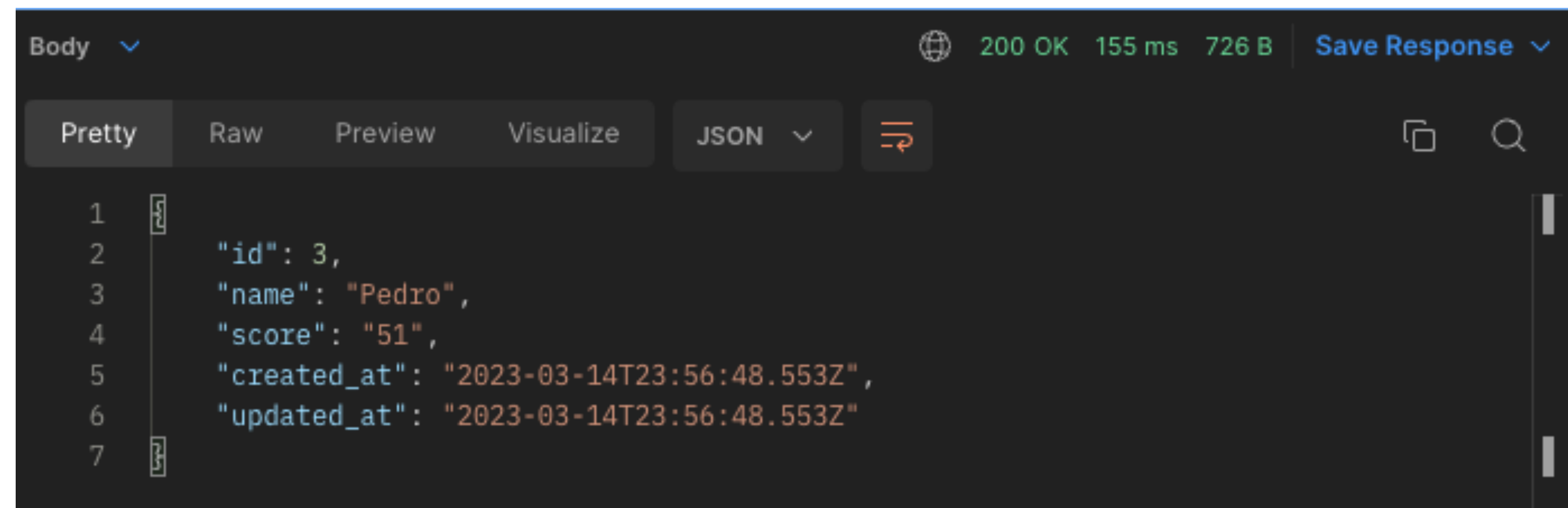


POSTMAN



Request

Response



update

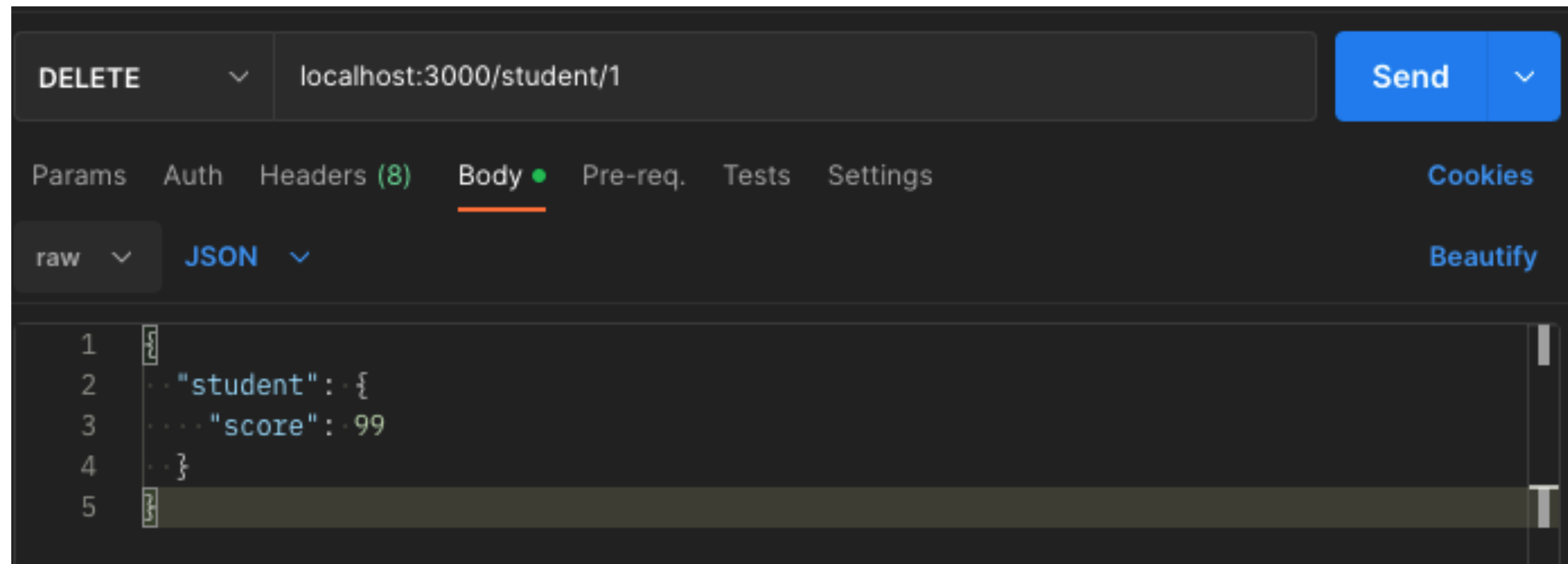
```
class StudentController < ApplicationController
  def update
    @student = Student.find(params[:id])
    if @student.update(student_params)
      render json: @student
    else
      render json: @student.errors, status: :unprocessable_entity
    end
  end

  def student_params
    params.require(:student).permit(:name, :score)
  end
end
```

```
Rails.application.routes.draw do
  ...
  patch 'student/:id', to: 'student#update'
end
```

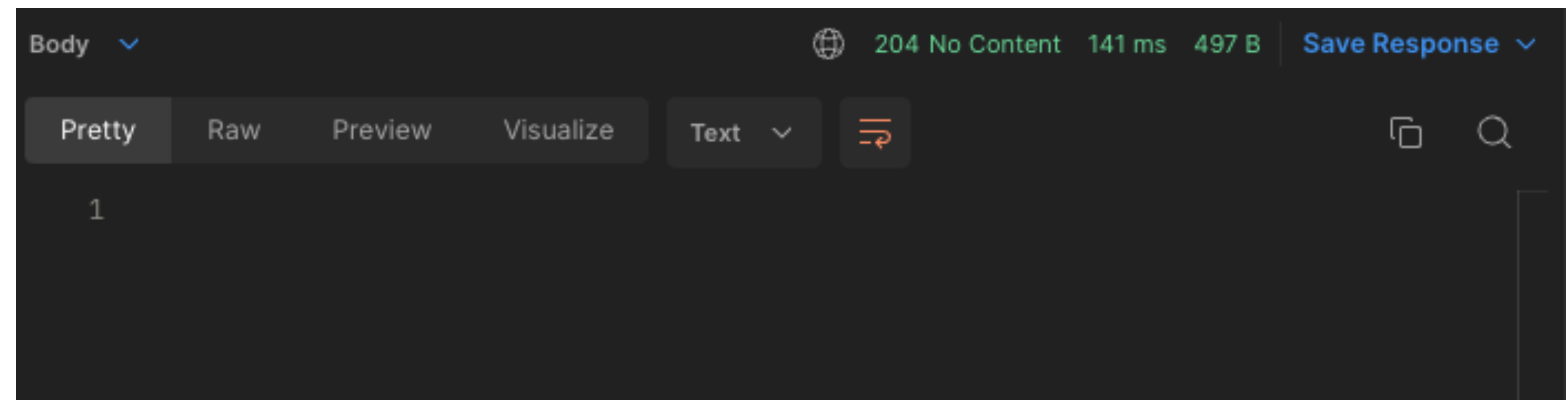


POSTMAN



Request

Response



delete

```
class StudentController < ApplicationController
  # DELETE /student/1
  def destroy
    @student = Student.find(params[:id])
    @student.destroy
  end
end
```

```
Rails.application.routes.draw do
  ...
  delete 'student/:id', to: 'student#destroy'
end
```


Filter

```
class StudentController < ApplicationController
  # GET /filter/:score
  def filter
    @students =
      Student.where("score = ?", params[:score])
    render json: @students
  end
end
```

```
Rails.application.routes.draw do
  ...
  get 'filter/:score', to: 'student#filter'
end
```

RoadMap

- *API - Estudiantes*
 - *Crear, Actualizar y borrar*
- ***Validadores***
- *Asociaciones: ejemplo 1 a n*
- *Otras asociaciones*

Validando que un estudiante si o si tenga nombre

```
class Student < ApplicationRecord
  validates :name, presence: true
end
```

Probando validadores en la consola:

```
>rails console
Loading development environment (Rails 7.0.4.2)
irb(main):001:0> student=Student.new(score:100)
=>
#<Student:0x000000010a193ee8
...
irb(main):002:0> student.save
=> false
```

No es posible guardar porque el estudiante no tiene el atributo name

El atributo errors

Probando validadores en la consola:

```
>rails console
Loading development environment (Rails 7.0.4.2)
irb(main):001:0> student=Student.new(score:100)
=>
#<Student:0x000000010a193ee8
...
irb(main):002:0> student.save
=> false
```

*Rails también guarda los errores de validaciones en un atributo llamado **errors***

```
student.errors
=> #<ActiveModel::Errors [#<ActiveModel::Error attribute=name, type=blank,
options={}>]>
```

valid?

Es posible preguntar a un objeto si esta en un estado valido o invalido

```
irb(main):004:0> student.valid?  
=> false
```

Para averiguar mas sobre validadores pueden visitar

https://guides.rubyonrails.org/active_record_validations.html

valid?

Es posible preguntar a un objeto si esta en un estado valido o invalido

```
irb(main):004:0> student.valid?  
=> false
```

Mas ejemplos

```
class Person < ApplicationRecord
  validates :name, length: { minimum: 2 }
  validates :bio, length: { maximum: 500 }
  validates :password, length: { in: 6..20 }
  validates :registration_number, length: { is: 6 }
end
```

```
class Account < ApplicationRecord
  validates :email, uniqueness: true
end
```

Para saber mas visita:

[**https://guides.rubyonrails.org/active_record_validations.html**](https://guides.rubyonrails.org/active_record_validations.html)

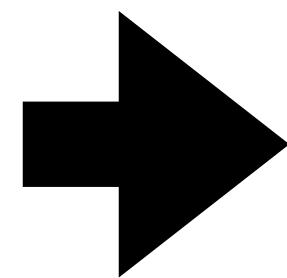
RoadMap

- *API - Estudiantes*
 - *Crear, Actualizar y borrar*
- *Validadores*
- ***Asociaciones: ejemplo 1 a n***
- *Otras asociaciones*

Creando un Modelo y su archivo de migración

```
> rails generate model Student name:text score:integer
```

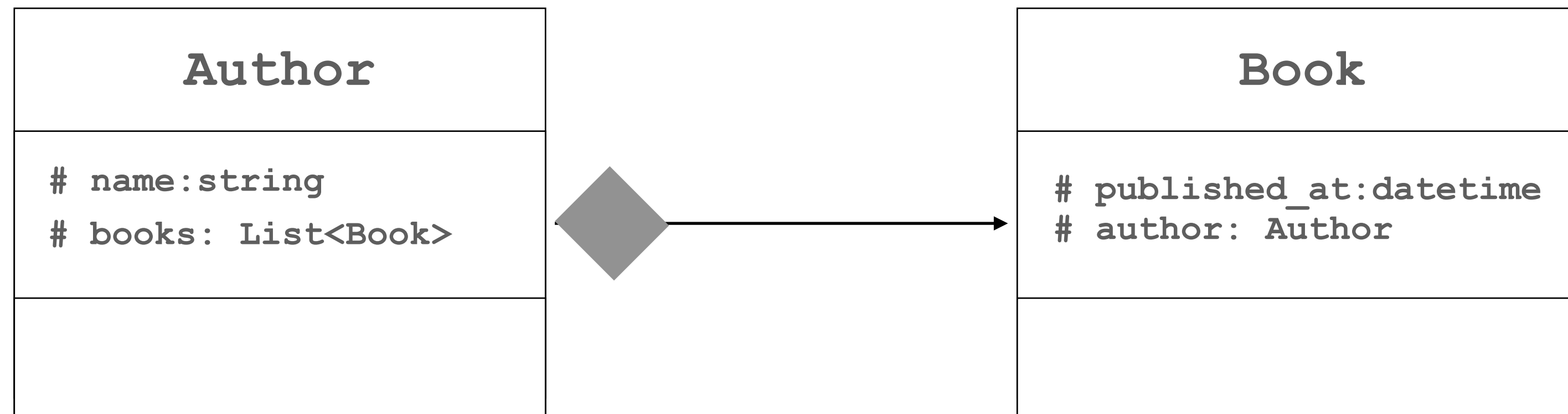
Es posible especificar el tipo de dato de la columna en la tabla.



```
class CreateStudents <
  ActiveRecord::Migration[7.0]
    def change
      create_table :students do |t|
        t.text :name
        t.integer :score
        t.timestamps
      end
    end
end
```

Asociaciones

- Un Author tiene varios libros
- Un libro pertenece a un author



Creamos modelos con los atributos simples:

```
> rails generate model Autor name:string
> rails generate model Book published_at:datetime
```

Clases y Migraciones Generadas

```
class CreateAuthors < ActiveRecord::Migration[7.0]
  def change
    create_table :authors do |t|
      t.string :name

      t.timestamps
    end
  end
end
```

```
class Author < ApplicationRecord
end
```

```
class CreateBooks < ActiveRecord::Migration[7.0]
  def change
    create_table :books do |t|
      t.datetime :published_at

      t.timestamps
    end
  end
end
```

```
class Book < ApplicationRecord
end
```

Agregando las asociaciones

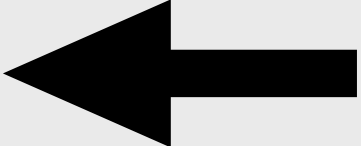
```
class CreateAuthors < ActiveRecord::Migration[7.0]
  def change
    create_table :authors do |t|
      t.string :name

      t.timestamps
    end
  end
end
```

```
class Author < ApplicationRecord
  has_many :books
end
```



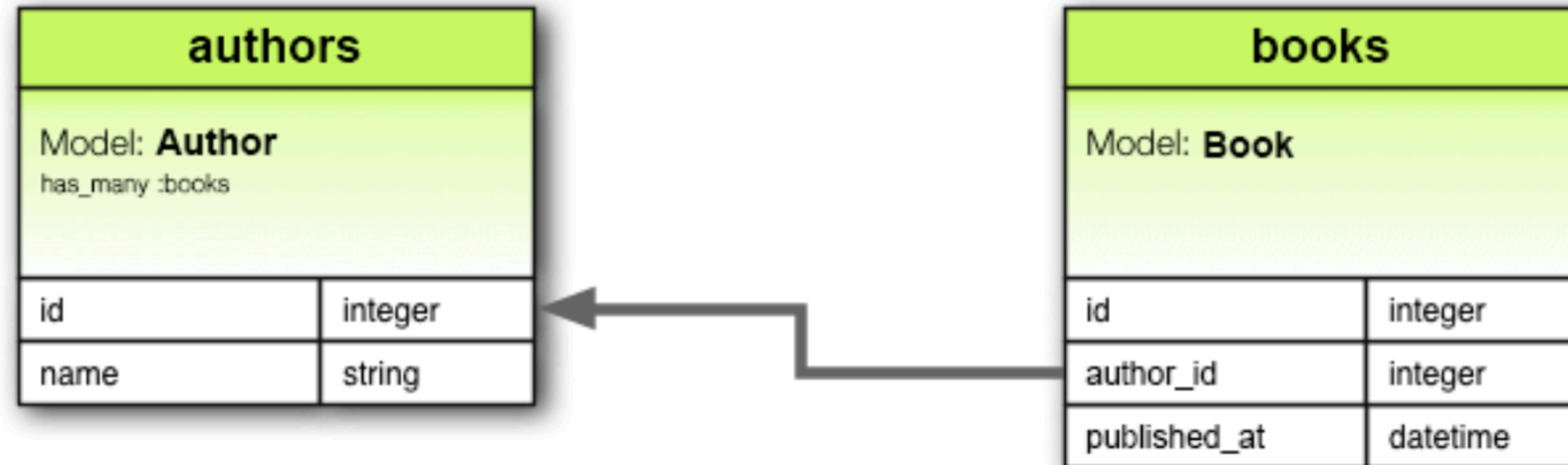
```
class CreateBooks < ActiveRecord::Migration[7.0]
  def change
    create_table :books do |t|
      t.belongs_to :author
      t.datetime :published_at
      t.timestamps
    end
  end
end
```



```
class Book < ApplicationRecord
  belongs_to :author
end
```



> rails db:migrate



- *Tablas y columnas generadas por rails*
- *En bases de datos solo es necesario que cada libro tenga el id del author.*

Probando en rails console

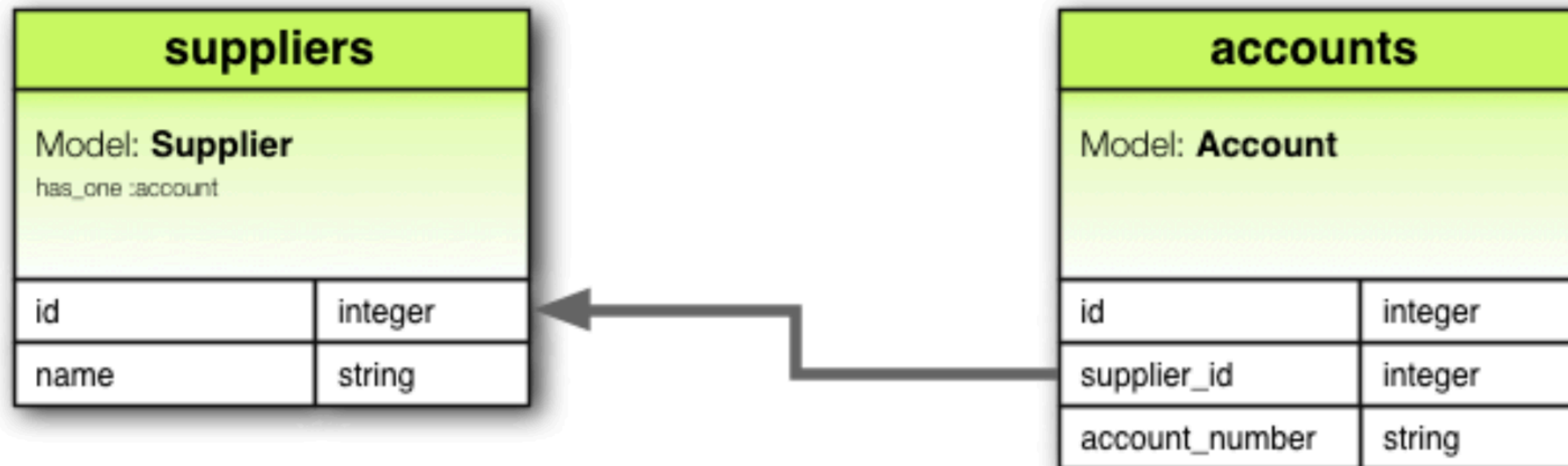
Probando asociaciones en consola

```
>rails console
irb(main):001:0> juan = Author.new(name: 'Juan')
irb(main):002:0> juan.save
irb(main):003:0> book1 = Book.new(published_at:Time.now, author: juan)
irb(main):004:0> book1.save
irb(main):005:0> book2= Book.new(published_at:Time.new, author: juan)
irb(main):006:0> book2.save
irb(main):007:0> juan.books
Book Load (0.4ms) SELECT "books".* FROM "books" WHERE "books"."author_id" = ?  [["author_id", 1]]
=>
[#<Book:0x0000000108553a80
  id: 1,
  author_id: 1,
  published_at: Wed, 15 Mar 2023 01:13:13.325948000 UTC +00:00,
  created_at: Wed, 15 Mar 2023 01:13:18.131497000 UTC +00:00,
  updated_at: Wed, 15 Mar 2023 01:13:18.131497000 UTC +00:00>,
#<Book:0x0000000108551ac8
  id: 2,
  author_id: 1,
  published_at: Wed, 15 Mar 2023 01:13:33.928028000 UTC +00:00,
  created_at: Wed, 15 Mar 2023 01:13:37.886815000 UTC +00:00,
  updated_at: Wed, 15 Mar 2023 01:13:37.886815000 UTC +00:00>]
irb(main):008:0>
```

RoadMap

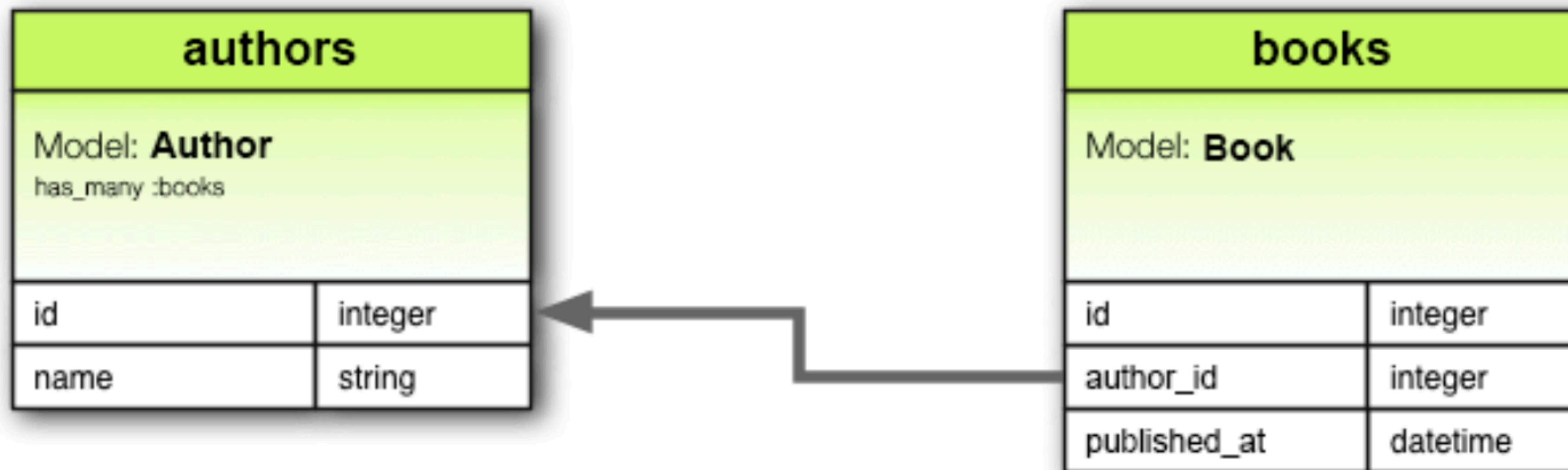
- *API - Estudiantes*
 - *Crear, Actualizar yorrar*
- *Validadores*
- *Asociaciones: ejemplo 1 a n*
- ***Otras asociaciones***

Has One



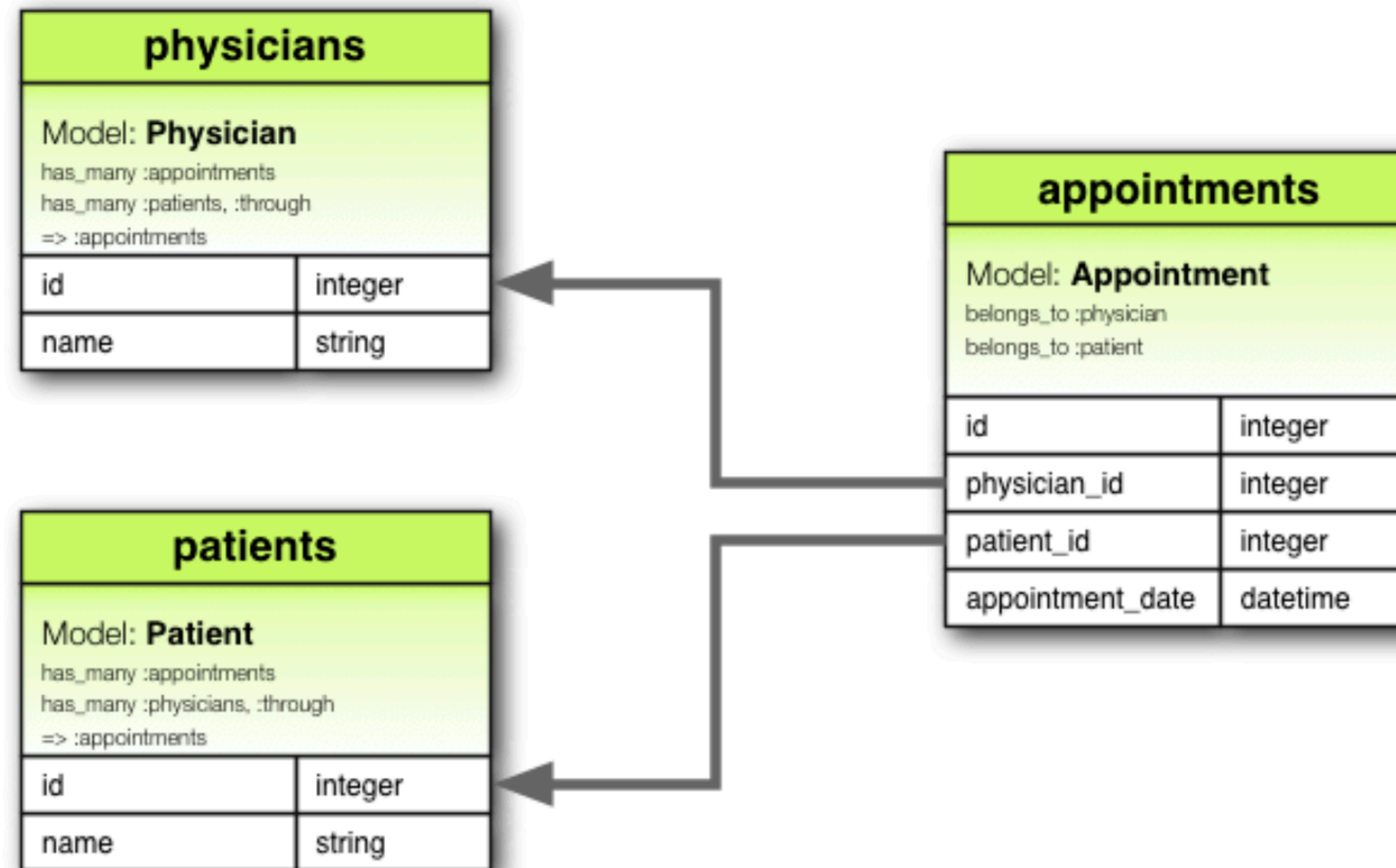
1 a 1

Has Many



1 a N

Has Many :through Association



N a N

Como implementar estas asociaciones?

https://guides.rubyonrails.org/v6.1/association_basics.html

Bonus - debugging

```
logger.debug "... message ..."
```

Mas sobre debugging

https://guides.rubyonrails.org/v6.1/debugging_rails_applications.html

```
>rails console
```

Material Adicional

- Documentación de Ruby on Rails: https://guides.rubyonrails.org/getting_started.html
- Documentación Active Record: https://guides.rubyonrails.org/active_record_basics.html
- Documentación migraciones: https://guides.rubyonrails.org/active_record_migrations.html
- Debugging :https://guides.rubyonrails.org/v6.1/debugging_rails_applications.html