

Ingeniería de Software

1.1 - ActiveRecord

IIC2143-3

Josefa España

jpespana@uc.cl

Contenidos

- API - Pokémon
  - CRUD (**C**reate, **R**ead, **U**pdate, **D**elete)
- Validaciones
- Asociaciones



API



Para crear un proyecto Api:

```
rails new PokemonApi --database=postgresql --api
```

Create

app/models:

PokemonApi - pokemon.rb

```ruby
# == Schema Information
#
# Table name: pokemons
#
# id              :bigint           not null, primary key
# name            :string           not null
# pokemon_type    :string           not null
class Pokemon < ApplicationRecord
end
```

En consola:

```
rails generate migration CreatePokemon name:string pokemon_type:string
      invoke  active_record
      create    db/migrate/20240310134012_create_pokemon.rb
```

Create

app/controllers:

PokemonApi - pokemons_controller.rb

```ruby
class PokemonsController < ApplicationController
  def create
    @pokemon = Pokemon.new(pokemon_params)
    if @pokemon.save
      render json: @pokemon, status: :created
    else
      render json: @pokemon.errors, status: :unprocessable_entity
    end
  end

  def pokemon_params
    params.require(:pokemon).permit(:name, :type)
  end
end
```

config/routes:

PokemonApi - routes.rb

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

Create

POSTMAN

POST localhost:3000/pokemons Send

Params Authorization Headers (9) Body Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL JSON Beautify

```
{
    "pokemon": {
        "name": "Pikachu",
        "pokemon_type": "Eléctrico"
    }
}
```

Body Cookies Headers (13) Test Results 201 Created 198 ms 751 B Save as example

Pretty Raw Preview Visualize JSON

```
{
    "id": 1,
    "name": "Pikachu",
    "pokemon_type": "Eléctrico",
    "created_at": "2024-03-10T13:48:40.310Z",
    "updated_at": "2024-03-10T13:48:40.310Z"
}
```



Update

app/models:

PokemonApi - pokemons_controller.rb

```ruby
class PokemonsController < ApplicationController
  def update
    @pokemon = Pokemon.find(params[:id])
    if @pokemon.update(pokemon_params)
      render json: @pokemon
    else
      render json: @pokemon.errors, status: :unprocessable_entity
    end
  end

  def pokemon_params
    params.require(:pokemon).permit(:name, :pokemon_type)
  end
end

```

Update

config/routes:

PokemonApi - routes.rb

```
1  Rails.application.routes.draw do
2    patch '/pokemons/:id', to: 'pokemons#update'
3  end
```

Update

POSTMAN

PATCH localhost:3000/pokemons/1 Send

Params Authorization Headers (9) Body Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL JSON Beautify

```
{
    "pokemon": {
        "name": "Jolteon",
        "pokemon_type": "Eléctrico"
    }
}
```

Body Cookies Headers (13) Test Results 200 OK 40 ms 783 B Save as example

Pretty Raw Preview Visualize JSON

```
{
    "name": "Jolteon",
    "pokemon_type": "Eléctrico",
    "id": 1,
    "created_at": "2024-03-10T14:26:47.130Z",
    "updated_at": "2024-03-10T14:26:50.134Z"
}
```



Delete

app/models:

PokemonApi -

```
1  class PokemonsController < ApplicationController
2    def destroy
3      @pokemon = Pokemon.find(params[:id])
4      @pokemon.destroy
5      head :no_content
6    end
7  end
```

Delete

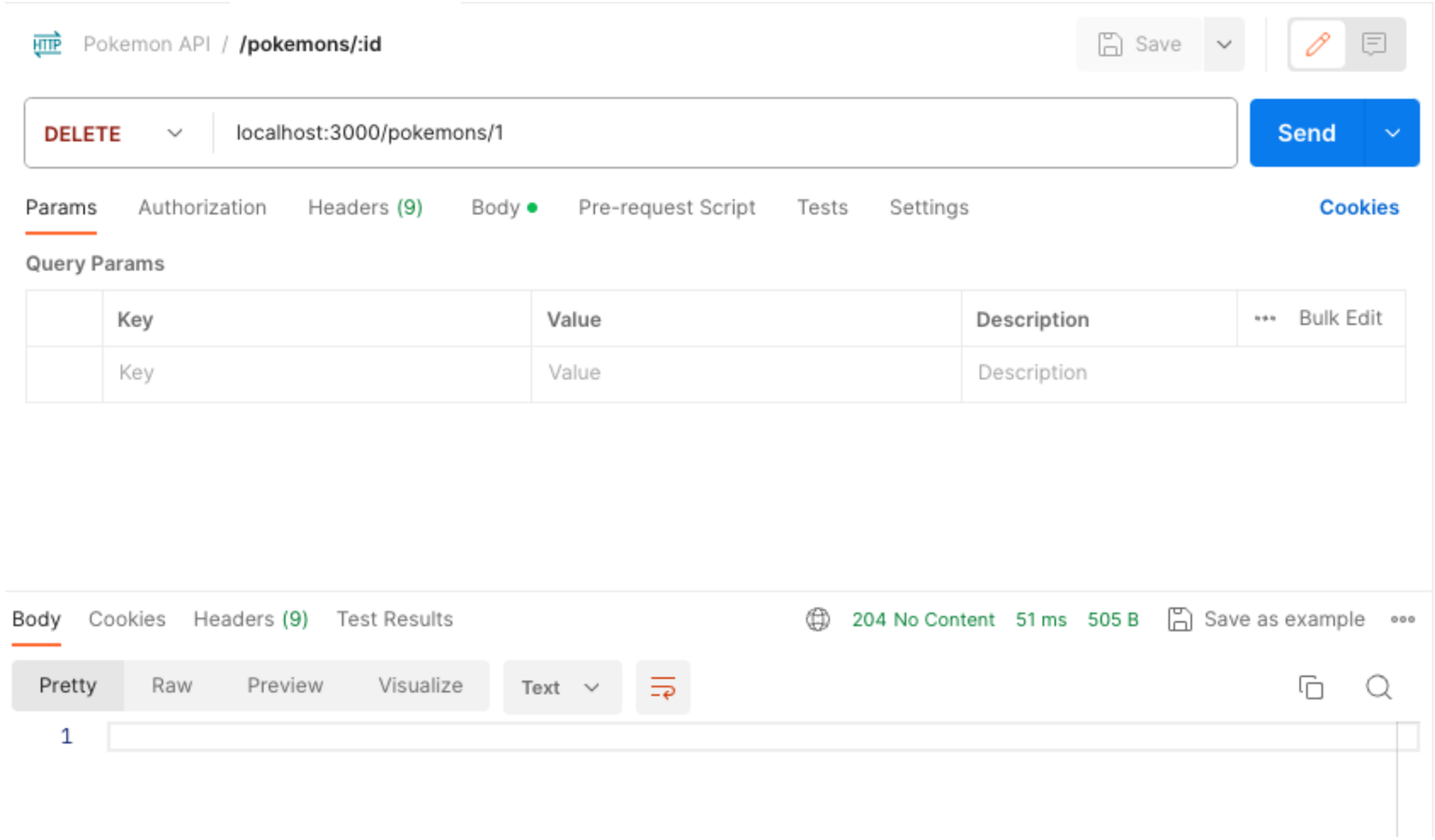
config/routes:

PokemonApi - routes.rb

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

Delete

POSTMAN



Filter

app/models:

PokemonApi - pokemons_controller.rb

```
1  class PokemonsController < ApplicationController
2    def filter_by_type
3      @pokemons = Pokemon.where(pokemon_type: params[:pokemon_type])
4      render json: @pokemons
5    end
6  end
```

Filter

config/routes:

PokemonApi - routes.rb

```ruby
Rails.application.routes.draw do
  get '/pokemons/filter/:pokemon_type', to: 'pokemons#filter_by_type'
end

```

Filter

POSTMAN

GET localhost:3000/pokemons/filter/Eléctrico Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL JSON Beautify

```
1
```

Body Cookies Headers (13) Test Results 200 OK 14 ms 748 B Save as example

Pretty Raw Preview Visualize JSON


```
[
    {
        "id": 2,
        "name": "Pikachu",
        "pokemon_type": "Eléctrico",
        "created_at": "2024-03-10T14:13:24.676Z",
        "updated_at": "2024-03-10T14:13:24.676Z"
    }
]
```




Validaciones

Los pokemon tienen que tener nombre

app/models:

PokemonApi - pokemon.rb

```ruby
# == Schema Information
#
# Table name: pokemons
#
# id              :bigint           not null, primary key
# name            :string           not null
# pokemon_type    :string           not null
class Pokemon < ApplicationRecord
  validates :name, presence: true
end
```

Los pokémon tienen que tener nombre

consola:

```
josefapaz@Josefas-MacBook-Pro PokemonApi % rails c
Loading development environment (Rails 7.1.3.2)
irb(main):001> blastoise = Pokemon.new(pokemon_type: "Agua")
⇒ #<Pokemon:0x000000011da95380 id: nil, name: nil, pokemon_type: "Agua", created_at: nil, updated_at: nil>
irb(main):002> blastoise.save
⇒ false
irb(main):003>
```

Como agregamos la validación de que deben tener nombre los pokémon, no nos deja guardar a blastoise con nombre **nil (nulo)**.

Los pokémon tienen que tener nombre

consola:

```
irb(main):003> blastoise.errors
⇒ #<ActiveModel::Errors [#<ActiveModel::Error attribute=name, type=blank, options={}>]>
irb(main):004> blastoise.valid?
⇒ false
```

- Los objetos guardan los errores de validación en el atributo **errors.**
- Podemos preguntar si el objeto tiene estado válido o inválido, muy útil para solo guardar cuando cumple los requisitos.

Pueden encontrar más validaciones en la guía oficial: https://guides.rubyonrails.org/active_record_validations.html

Más validaciones

```
13  class Person < ApplicationRecord
14    validates :name, presence: true, length: { minimum: 2 }
15    validates :age, presence: true, numericality: { only_integer: true, greater_than: 0 }
16    validates :password, presence: true, length: { in: 6..20 }
17    validates :email, presence: true, uniqueness: true
18  end
```

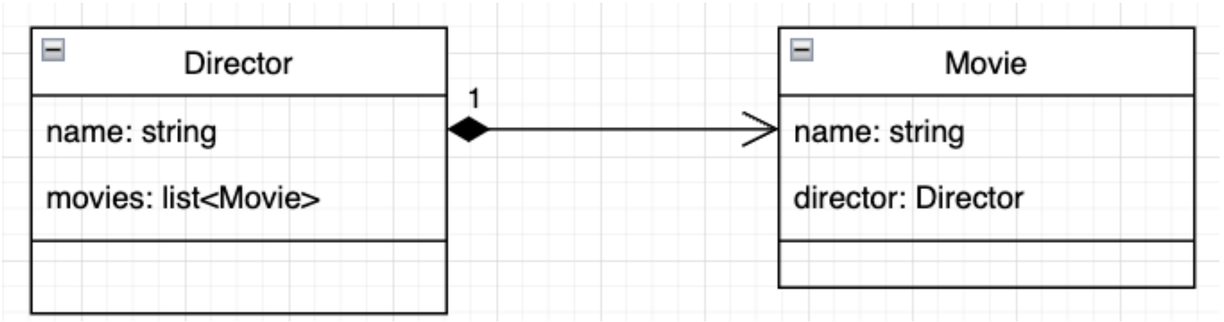
Asociaciones

Proyecto: Películas

En consola:

```
rails new peliculas --database=postgresql --api
```

Asociaciones



- Un director puede haber dirigido varias películas.
- Una película pertenece a un director.

En consola:

```
rails generate model Director name:string
rails generate model Movie name:string
```

Clases y Migraciones generales

peliculas - 20240310145226_create_directors.rb

```ruby
class CreateDirectors < ActiveRecord::Migration[7.1]
  def change
    create_table :directors do |t|
      t.string :name

      t.timestamps
    end
  end
end

```

peliculas - director.rb

```ruby
class Director < ApplicationRecord
  has_many :movies
end

```

- Agregamos la asociación al modelo

Clases y Migraciones generales

peliculas - 20240310145239_create_movies.rb

```rb
class CreateMovies < ActiveRecord::Migration[7.1]
  def change
    create_table :movies do |t|
      t.string :name
      t.belongs_to :director
      t.timestamps
    end
  end
end

```

peliculas - movie.rb

```rb
class Movie < ApplicationRecord
  belongs_to :director
end

```

- Agregamos la asociación al modelo y a la migración

Probando asociaciones

```
rails c
Loading development environment (Rails 7.1.3.2)
irb(main):001> tarantino = Director.new(name: 'Tarantino')
=> #<Director:0x00000001220c53a0 id: nil, name: "Tarantino", created_at: nil, updated_at: nil>
irb(main):002> tarantino.save!
  TRANSACTION (0.3ms)  BEGIN
  Director Create (1.8ms)  INSERT INTO "directors" ("name", "created_at", "updated_at") VALUES
($1, $2, $3) RETURNING "id"  [["name", "Tarantino"], ["created_at", "2024-03-10 15:08:52.853588"],
["updated_at", "2024-03-10 15:08:52.853588"]]
  TRANSACTION (1.4ms)  COMMIT
=> true
irb(main):003> pulp_fiction = Movie.new(name: 'Pulp Fiction', director: tarantino)
=> #<Movie:0x0000000121fe5408 id: nil, name: "Pulp Fiction", director_id: 2, created_at: nil, upda
ted_at: nil>
irb(main):004> pulp_fiction.save!
  TRANSACTION (0.2ms)  BEGIN
  Movie Create (5.7ms)  INSERT INTO "movies" ("name", "director_id", "created_at", "updated_at") V
ALUES ($1, $2, $3, $4) RETURNING "id"  [["name", "Pulp Fiction"], ["director_id", 2], ["created_a
t", "2024-03-10 15:09:27.131769"], ["updated_at", "2024-03-10 15:09:27.131769"]]
  TRANSACTION (0.4ms)  COMMIT
=> true
```

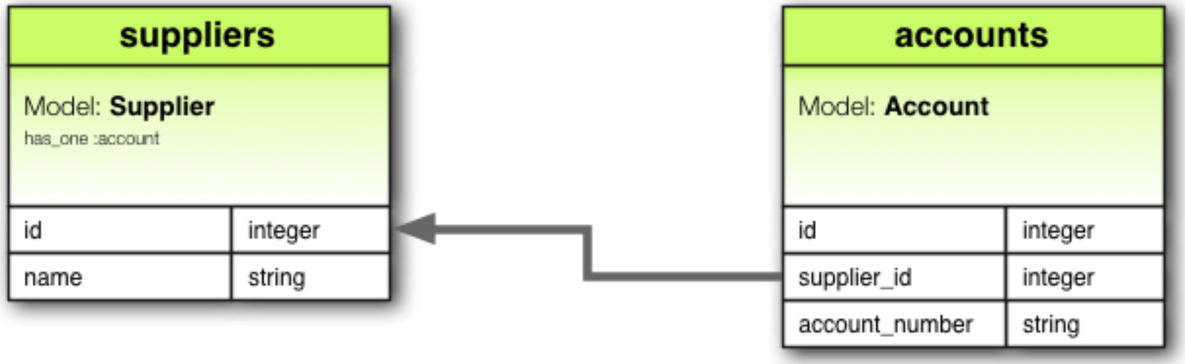
Probando asociaciones

```
irb(main):005> kill_bill = Movie.new(name: 'Kill Bill', director: tarantino)
=> #<Movie:0x000000012230b6a0 id: nil, name: "Kill Bill", director_id: 2, created_at: nil, updated
_at: nil>
irb(main):006> kill_bill.save!
  TRANSACTION (0.3ms)  BEGIN
  Movie Create (2.2ms)  INSERT INTO "movies" ("name", "director_id", "created_at", "updated_at") V
ALUES ($1, $2, $3, $4) RETURNING "id"  [["name", "Kill Bill"], ["director_id", 2], ["created_at",
"2024-03-10 15:10:07.693933"], ["updated_at", "2024-03-10 15:10:07.693933"]]
  TRANSACTION (0.5ms)  COMMIT
=> true
irb(main):007> tarantino.movies
  Movie Load (0.2ms)  SELECT "movies".* FROM "movies" WHERE "movies"."director_id" = $1 /* loading
for pp */ LIMIT $2  [["director_id", 2], ["LIMIT", 11]]
=>
[#<Movie:0x00000001223ab060
  id: 1,
  name: "Pulp Fiction",
  director_id: 2,
  created_at: Sun, 10 Mar 2024 15:09:27.131769000 UTC +00:00,
  updated_at: Sun, 10 Mar 2024 15:09:27.131769000 UTC +00:00>,
 #<Movie:0x00000001223aaf20
  id: 2,
  name: "Kill Bill",
  director_id: 2,
  created_at: Sun, 10 Mar 2024 15:10:07.693933000 UTC +00:00,
  updated_at: Sun, 10 Mar 2024 15:10:07.693933000 UTC +00:00>]
irb(main):008>
```

Otras asociaciones

Asociación 1 a 1:

- Una cuenta tiene un Provedor
- en rails: **has_one**

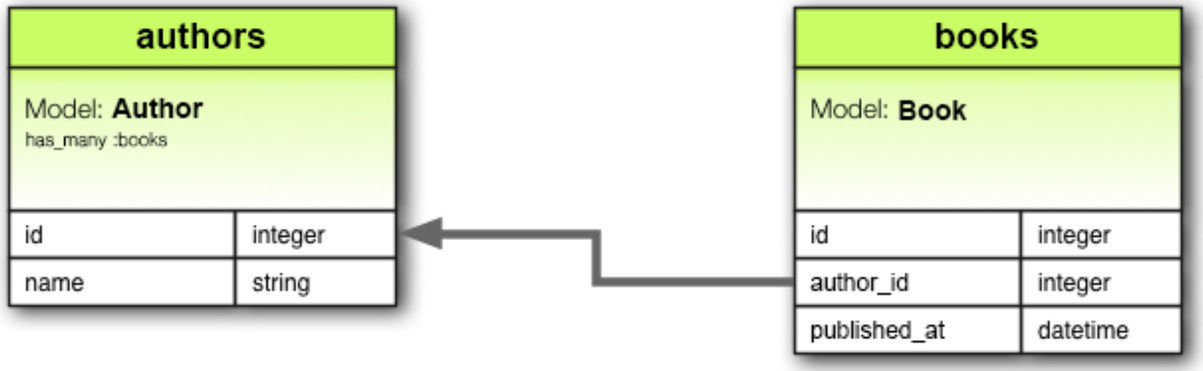



```
class Supplier < ApplicationRecord
  has_one :account
end
```

Otras asociaciones

Asociación 1 a 1:

- Un autor tiene muchos libros
- en rails: **has_many**

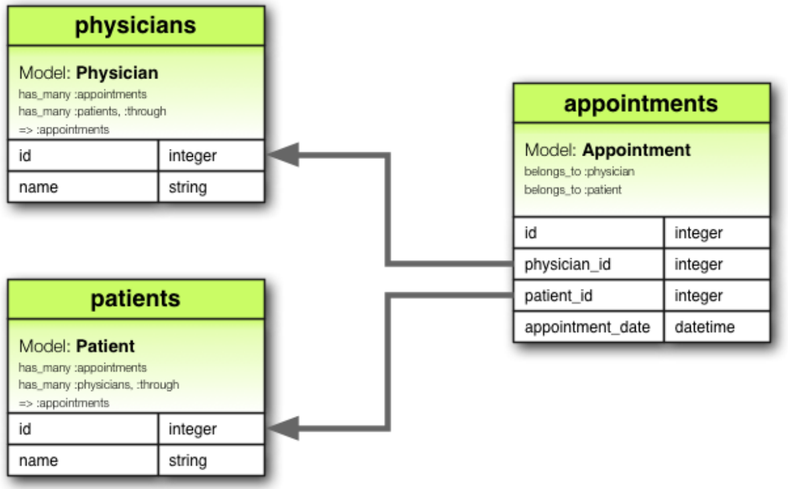



```
class Author < ApplicationRecord
  has_many :books
end
```

Otras asociaciones

Asociación 1 a 1:

- Un paciente tiene muchos médicos a través de una hora agendada.
- en rails: **has_many :through**




```
class Physician < ApplicationRecord
  has_many :appointments
  has_many :patients, :through => :appointments
end

class Appointment < ApplicationRecord
  belongs_to :physician
  belongs_to :patient
end

class Patient < ApplicationRecord
  has_many :appointments
  has_many :physicians, :through => :appointments
end
```

Material Adicional

- Doc ActiveRecord: https://guides.rubyonrails.org/active_record_basics.html
- Doc asociaciones: https://guides.rubyonrails.org/association_basics.html
- Doc migraciones: https://guides.rubyonrails.org/active_record_migrations.html
- Debugging en rails: https://guides.rubyonrails.org/v6.1/debugging_rails_applications.html
-



Ingeniería de Software

1.1 - ActiveRecord

IIC2143-3

Josefa España

jpespana@uc.cl