daftcode

RUBY ON RAILS 4 BEGINNERS

6. Relacje między modelami

Praca domowa

Asocjacje

Statue

```
$ rails g model statue
```

```ruby
# app/models/statue.rb

class Statue < ApplicationRecord
end

# db/migrate/20181205112806_create_statues

class CreateStatues < ActiveRecord::Migration[5.2]
  def change
    create_table :statues do |t|

      t.timestamps
    end
  end
end
```

1 - 1= 0

```ruby
# app/models/statue.rb

class Statue < ApplicationRecord
  has_one :rock
end
```

```ruby
# app/models/rock.rb

class Rock < ApplicationRecord
  belongs_to :statue
end
```

1 - *

```ruby
# app/models/statue.rb

class Statue < ApplicationRecord
  has_many :rocks
end
```

```ruby
# app/models/rock.rb

class Rock < ApplicationRecord
  belongs_to :statue
end
```

* _ *

```
# app/models/statue.rb

class Statue < ApplicationRecord
  has_and_belongs_to_many :rocks
end
```

```
# app/models/rock.rb

class Rock < ApplicationRecord
  has_and_belongs_to_many :statues
end
```

1 - * | * - 1

```ruby
# app/models/statue.rb

class Statue < ApplicationRecord
  has_many :statue_rocks
  has_many :rocks, through: :statue_rocks
end
```

```ruby
# app/models/rock.rb

class Rock < ApplicationRecord
  has_many :statue_rocks
  has_many :statues, through: :statue_rocks
end
```

```ruby
# app/models/statue_rock.rb

class StatueRock < ApplicationRecord
  belongs_to :statue
  belongs_to :rock
end
```

Migracja

```
# db/migrate/20181205112806_create_statues

class CreateStatues < ActiveRecord::Migration[5.2]
  def change
    create_table :statues do |t|
      t.string :name, null: false
      t.index :name, unique: true
      t.timestamps
    end
  end
end
```

```
# db/migrate/20181205120244_add_statue_to_rock

class AddStatueToRock < ActiveRecord::Migration[5.2]
  def change
    add_reference :rocks, :statue
  end
end
```

Schema

```ruby
# db/schema.rb

ActiveRecord::Schema.define(version: 2018_12_05_120244) do

  create_table "rocks", force: :cascade do |t|
    t.float "weight", null: false
    t.datetime "created_at", null: false
    t.datetime "updated_at", null: false
    t.string "nickname"
    t.integer "statue_id"
    t.index ["statue_id"], name: "index_rocks_on_statue_id"
  end

  create_table "statues", force: :cascade do |t|
    t.string "name", null: false
    t.datetime "created_at", null: false
    t.datetime "updated_at", null: false
    t.index ["name"], name: "index_statues_on_name", unique: true
  end

end
```

Modele

```ruby
# app/models/statue.rb

class Statue < ApplicationRecord
  has_many :rocks, dependent: :destroy
  validates :name,
    presence: true,
    uniqueness: true,
    format: /\w{3,}/
end
```

```ruby
# app/models/rock.rb

class Rock < ApplicationRecord
  belongs_to :statue, optional: true
  validates :weight,
    presence: true,
    numericality: { greater_than: 0 }
end
```

Wykorzystanie

```
> statue = Statue.create(name: 'lilly')
   (0.2ms)  begin transaction
  Statue Exists (0.2ms)  SELECT  1 AS one FROM ...
  Statue Create (0.5ms)  INSERT INTO "statues" ...
   (3.6ms)  commit transaction
 => #<Statue id: 1, name: "lilly", ...>

> statue.rocks
  Rock Load (0.3ms)  SELECT  "rocks".* FROM "rocks"
    WHERE "rocks"."statue_id" = ? ...
 => #<ActiveRecord::Associations::CollectionProxy []>
```

Wykorzystanie

```
> rock = Rock.create(weight: 3.0, statue: statue)
   (0.1ms)  begin transaction
  Rock Create (2.8ms)  INSERT INTO "rocks" ...
   (3.1ms)  commit transaction
 => #<Rock id: 15, weight: 3.0, ..., statue_id: 1>

> rock.statue
 => #<Statue id: 1, name: "lilly", ...>

> statue.rocks
  Rock Load (2.5ms)  SELECT  "rocks".* FROM "rocks" ...
 => #<ActiveRecord::Associations::CollectionProxy [
      #<Rock id: 15, weight: 3.0, ...>]>
```

Nested zasoby

Routing

```ruby
# config/routes.rb

Rails.application.routes.draw do
  resources :statues, param: :name, only: :show do
    resources :rocks, only: [:create, :destroy], controller: :nested_rocks
  end
end
```

```
statue_rocks POST   /statues/:statue_name/rocks         nested_rocks#create
 statue_rock DELETE /statues/:statue_name/rocks/:id     nested_rocks#destroy
      statue GET    /statues/:name                      statues#show
```

Kontroler

```ruby
# app/controllers/statues_controller.rb

class StatuesController < ApplicationController
  def show
    @statue = Statue.find_by!(name: params[:name])
  end
end
```

Kontroler

```ruby
# app/controllers/statues_controller.rb

class NestedRocksController < ApplicationController

  def create
    statue.rocks.create(weight: rand(50) / 10.0)
    redirect_to statue_path(name: statue.name)
  end

  def destroy
    statue.rocks.destroy(params[:id])
    redirect_to statue_path(name: statue.name)
  end

  private

  def statue
    @statue ||= Statue.find_by!(name: params[:statue_name])
  end
end
```

Widok

```
<%# app/views/statues/show.html.erb %>

<br/>Statue: <%= @statue.name %><br/><br/>
<%= link_to 'Add', statue_rocks_path(statue_name: @statue.name), method: :post %>
<table>
  <% @statue.rocks.each do |rock| %>
    <tr>
      <td><%= rock.id %></td>
      <td><%= rock.weight %></td>
      <td>
        <%= link_to 'Remove', statue_rock_path(
          statue_name: @statue.name, id: rock.id), method: :delete %>
      </td>
    </tr>
  <% end %>
</table>
```

Counter cache

Model

```ruby
# app/models/rock.rb

class Rock < ApplicationRecord
  belongs_to :statue,
    optional: true,
    counter_cache: true
  validates :weight,
    presence: true,
    numericality: { greater_than: 0 }
end
```

Statue

```
$ rails g migration AddRocksCountToStatue
```

```ruby
# db/migrate/20181205135116_add_rocks_count_to_statue.rb

class AddRocksCountToStatue < ActiveRecord::Migration[5.2]
  def change
    add_column :statues, :rocks_count, :integer, default: 0
  end
end
```

Widok

```
<%# app/views/statues/show.html.erb %>

<br/>Statue: <%= @statue.name %> (<%= @statue.rocks_count %>)<br/><br/>
...
```

Scope

Model

```ruby
# app/models/rock.rb

class Rock < ApplicationRecord
  belongs_to :statue, optional: true, counter_cache: true

  validates :weight,
    presence: true,
    numericality: { greater_than: 0 }

  scope :sorted, -> { order(weight: :desc, id: :asc) }
  scope :heavier_than, ->(w) { where('weight > ?', w) }
end
```

Kontroler

```ruby
# app/controllers/statues_controller.rb

class StatuesController < ApplicationController
  def show
    @statue = Statue.find_by!(name: params[:name])
    @rocks = @statue.rocks.sorted
    # @rocks = @statue.rocks.sorted.heavier_than(2.5)
  end
end
```

Widok

```
<%# app/views/statues/show.html.erb %>
...
<table>
  <% @rocks.each do |rock| %>
...
```

Thanks!