

Rails CRUD & Rails Routing

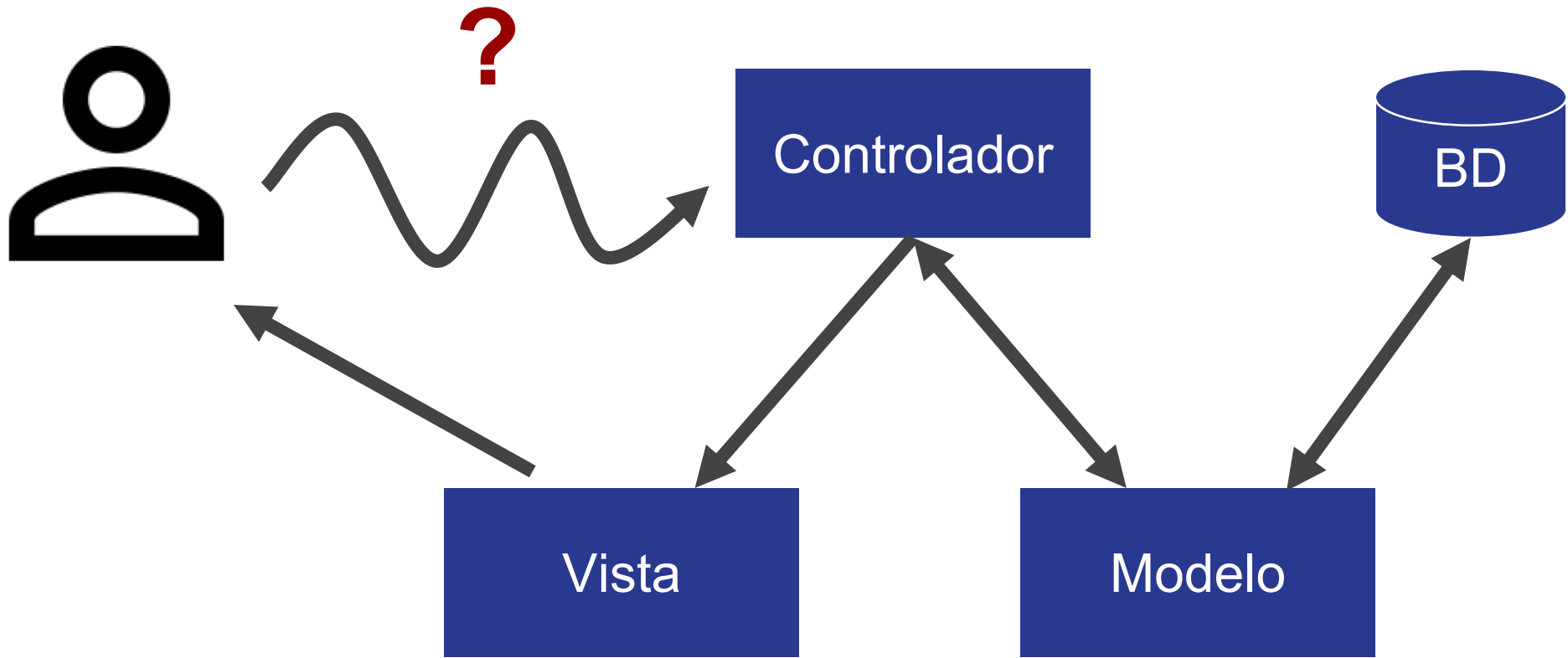
IIC2143 Ingeniería de Software - Ayudantía 4

Hoy:

1. Recordatorio
2. ¿Qué hay en una aplicación de Rails?
3. De ER a aplicación.
4. CRUD
5. Rutas
6. Validaciones

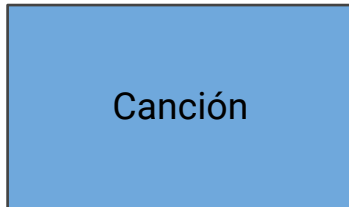
Recordatorio

Modelo-Vista-Controlador (MVC)



ER: Entidades

Representación de un objeto o concepto.



ER: Atributos

Representa una propiedad de interés de una entidad.



ER: Relaciones

Representa y describe la interacción entre dos o más entidades.

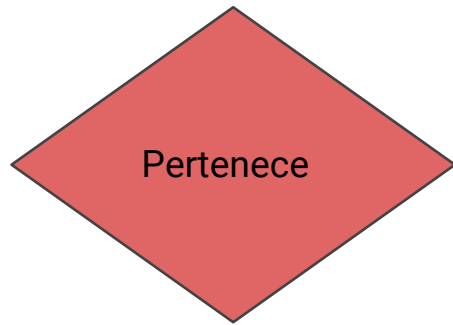
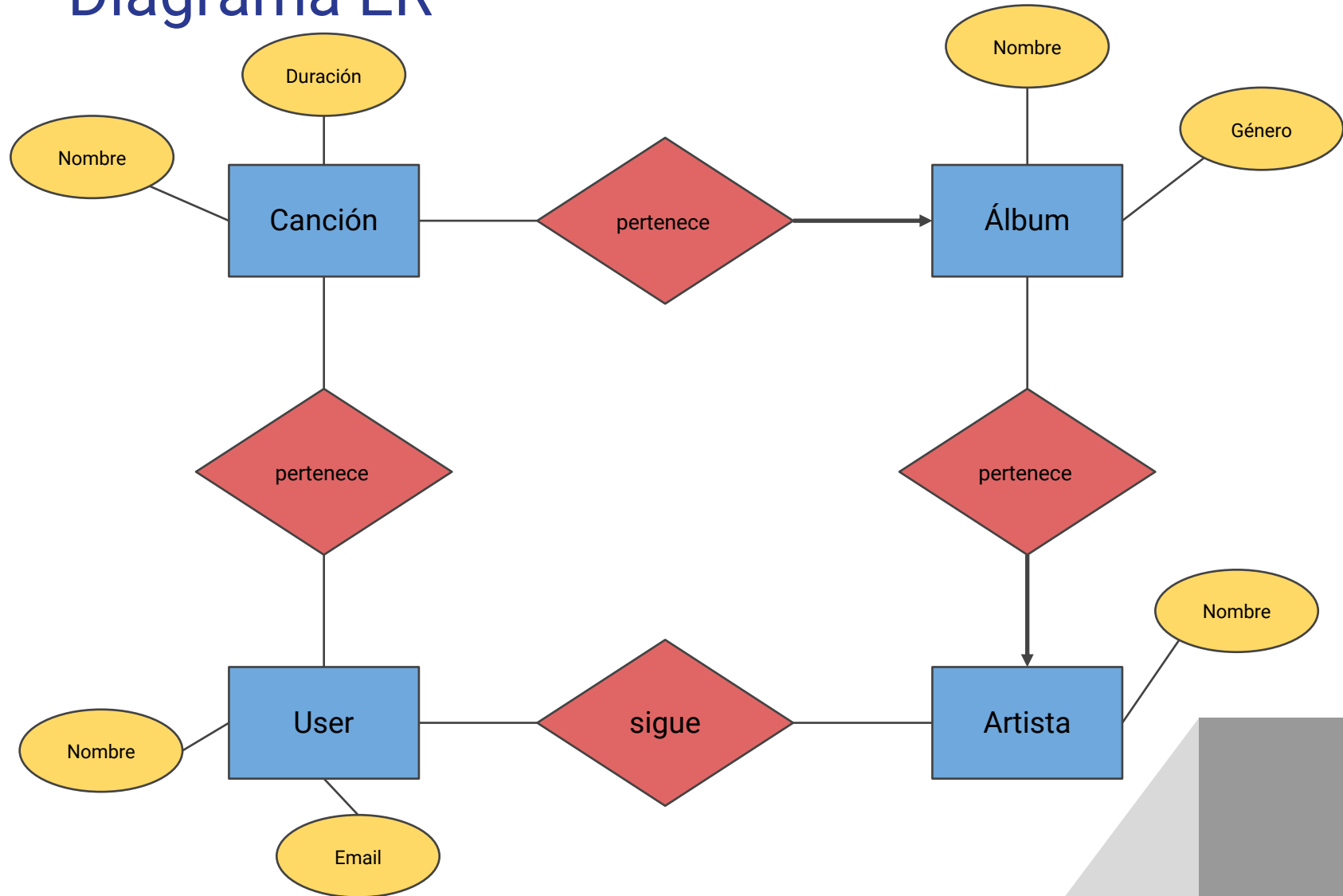


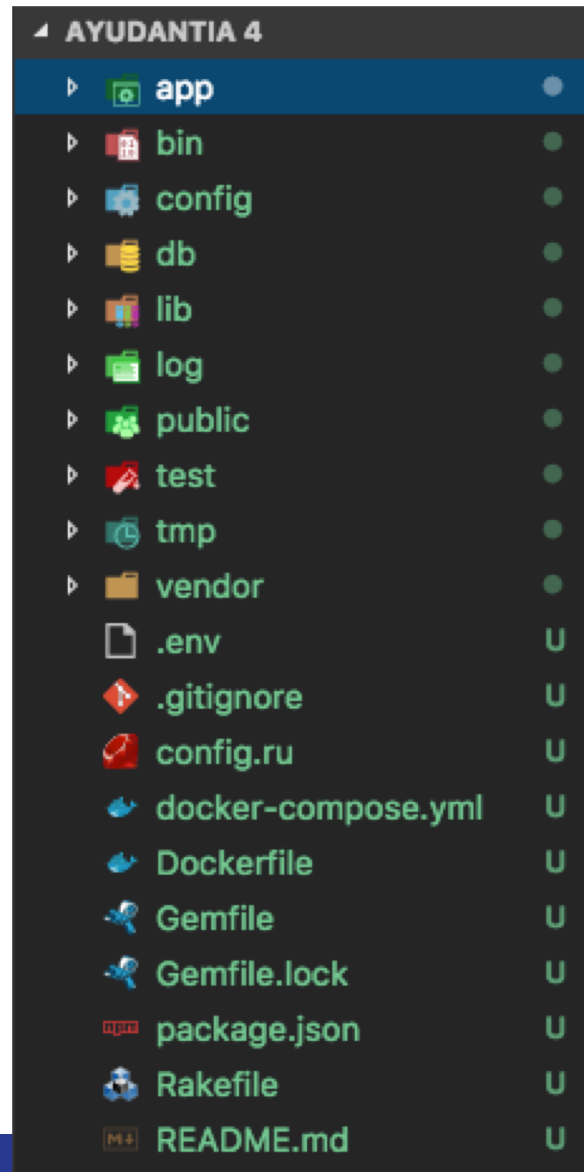
Diagrama ER





¿Qué hay en una aplicación
Rails?

Archivos y directorios

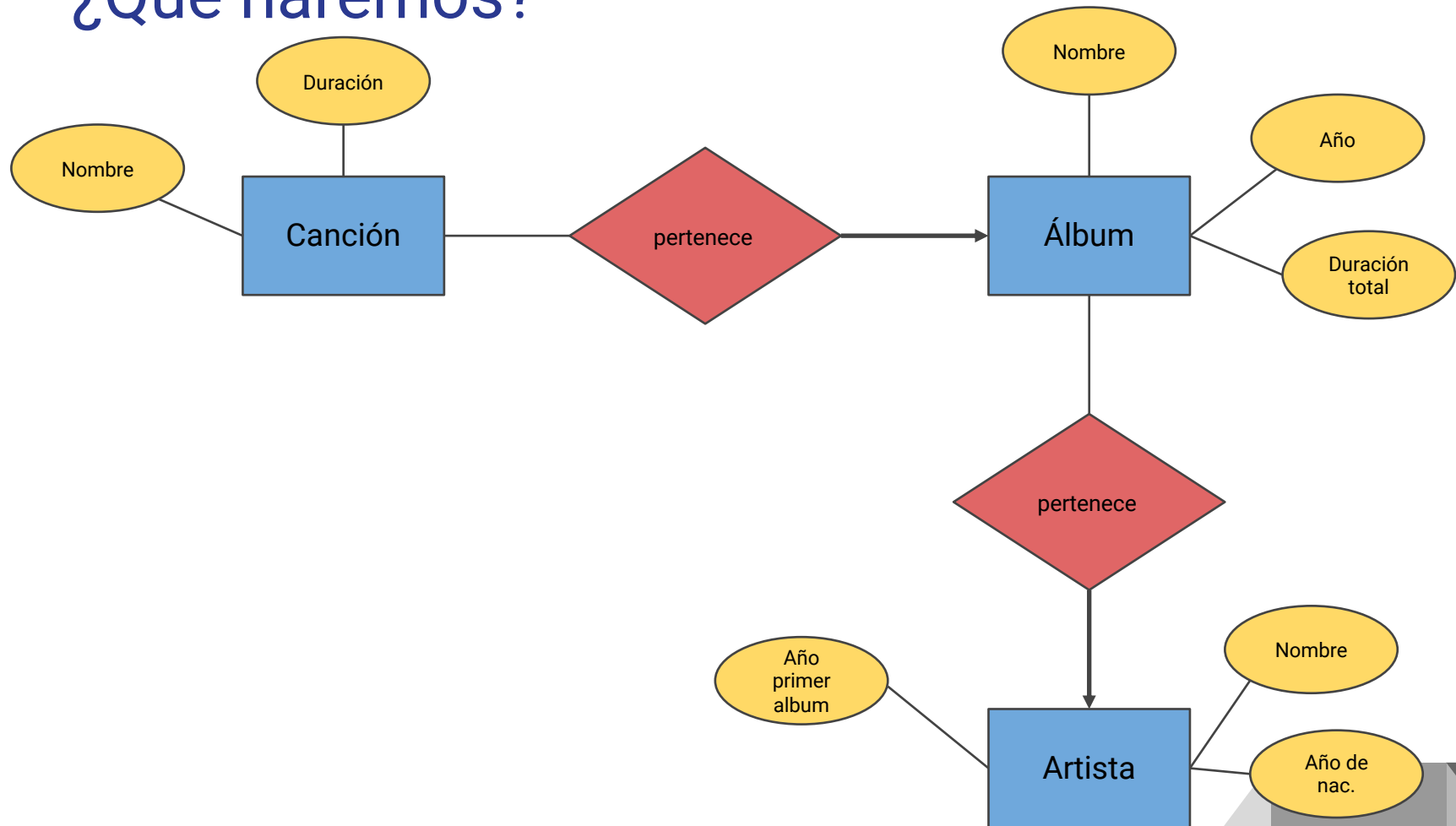




Programemos!

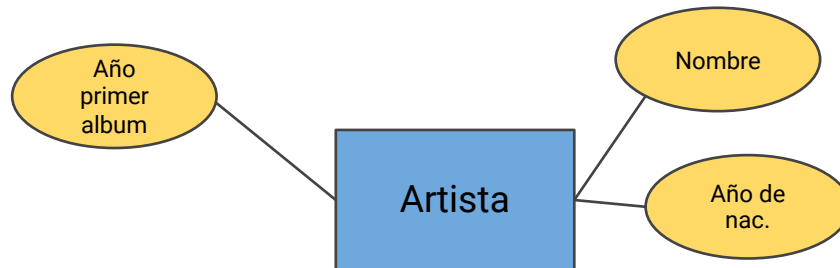
De ER a Rails

¿Qué haremos?



Creando artistas

`$ docker-compose run web rails generate scaffold Artist name:string birth_year:integer first_album_year:integer`

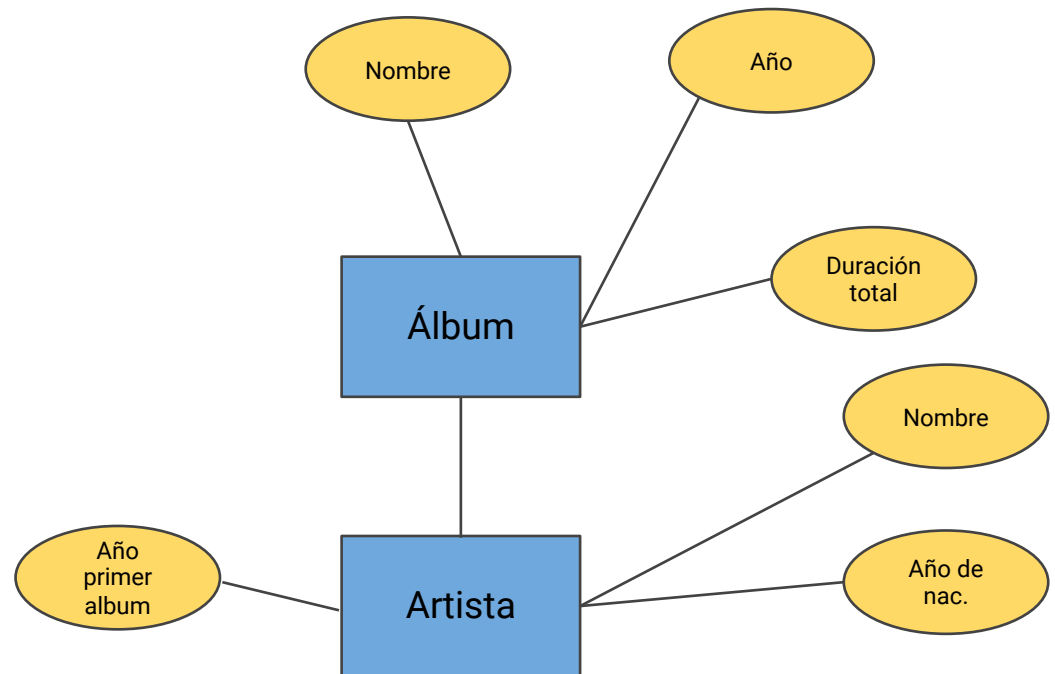


Correr migraciones y jugar un rato :D

Revisar:
https://guides.rubyonrails.org/v3.2.9/getting_started.html#getting-up-and-running-quickly-with-scaffolding

Creando álbumes

\$ docker-compose run web rails generate scaffold Album name:string year:integer
total_duration:integer **artist:references**

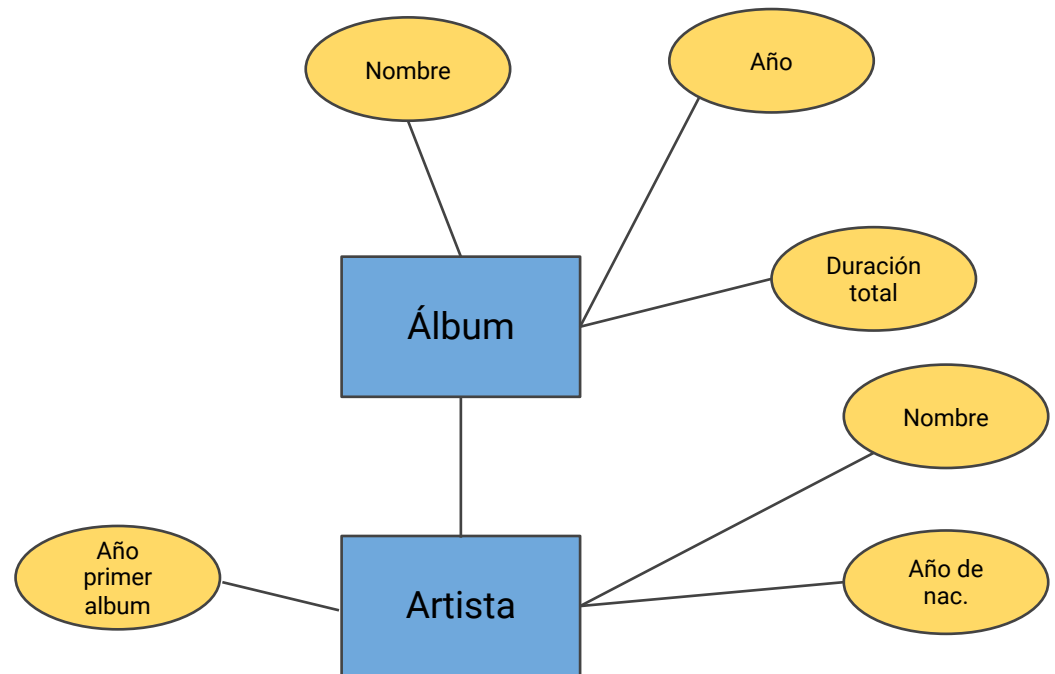
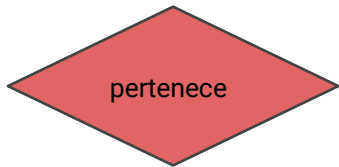


Correr migraciones y jugar un rato :D

Creando álbumes

\$ docker-compose run web rails generate scaffold Album name:string year:integer
total_duration:integer **artist:references**

Pero falta la asociación!



Correr migraciones y jugar un rato :D

Asociaciones

- belongs_to
- has_one
- has_many
- has_and_belongs_to_many

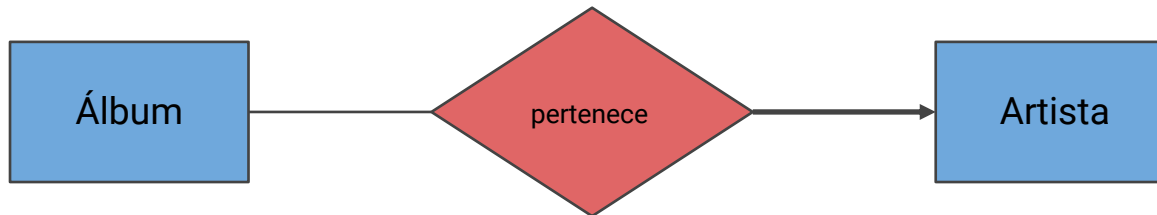
Revisar: https://guides.rubyonrails.org/association_basics.html



Creando asociaciones

- En models/artist.rb agregar ***has_many :albums***
- En models/album.rb agregar ***belongs_to :artist, dependent: :destroy [opcional]***

Y listo, nuestra app luce así:

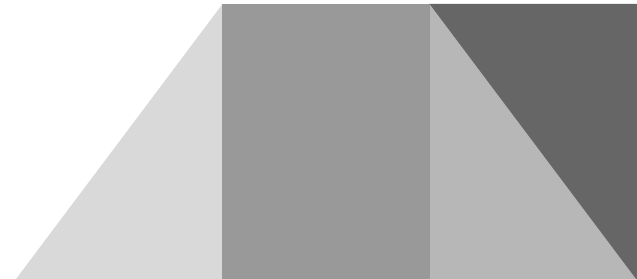


Creando canciones

\$ docker-compose run web rails generate scaffold Song name:string duration:integer
album:references

Agregar asociaciones entre canciones y albums

Correr migraciones y jugar un rato :D



Rails Routing

Rutas

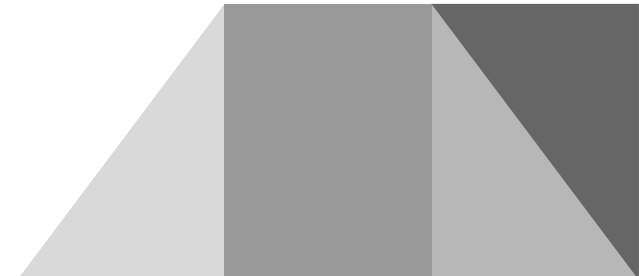
- Dado el path de una request, determina qué controlador la procesará.

GET /songs/4860

POST /playlists

PUT /users/13637542

DESTROY /planets/51

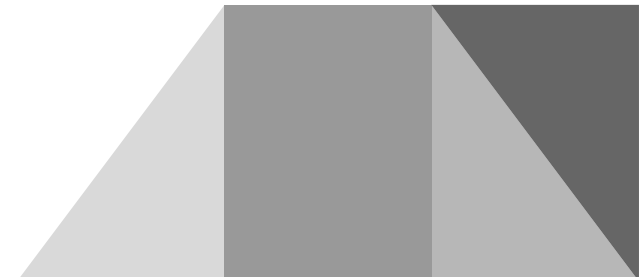


CRUD y HTTP

	SQL	HTTP
C REATE	INSERT	POST
R EAD	SELECT	GET
U PDATE	UPDATE	PUT/PATCH
D ELETE	DELETE	DELETE

Existen muchas formas de generar las rutas para nuestra aplicación y, dependiendo del caso, será diferente el método HTTP a utilizar.

Documentación de Rails Routing:
<https://guides.rubyonrails.org/routing.html>



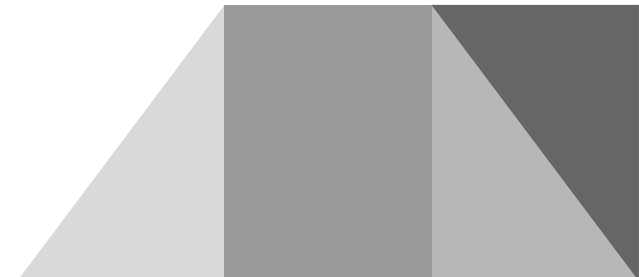
Validaciones

¿Para qué son las validaciones?

- Sirven para cumplir ciertas reglas antes de que los objetos y sus atributos se guarden en nuestra base de datos.
- Ejemplos: número de caracteres de contraseña, nombres no tan largos o con caracteres inválidos, fechas entre algún rango, etc.
- La idea de las validaciones es controlar los recursos que se están creando o modificando antes de que se guarden. Por lo tanto, se crean y ejecutan directamente en los modelos.
- Revisar:
https://guides.rubyonrails.org/active_record_validations.html

Haremos validaciones para el artista

- Al momento de crear al artista queremos que su nombre no esté vacío.
- Y además queremos que su año de nacimiento sea menor que el año en que lanzó su primer álbum (es obvio para nosotros, pero para Rails no 😐)



models/artist.rb

Nombre no vacío:

```
class Artist < ApplicationRecord
  has_many :albums
  validates :name, presence: true
end
```

models/artist.rb

Año de nacimiento menor que lanzamiento primer disco

```
class Artist < ApplicationRecord
  has_many :albums
  validates :name, presence: true
  validate :valid_first_album_year

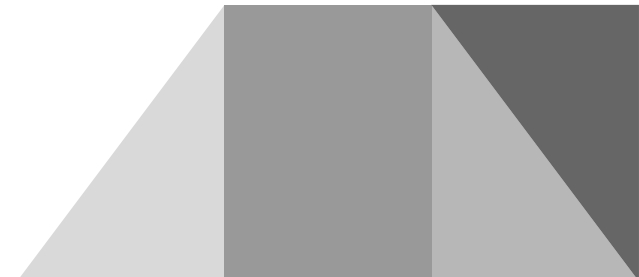
  def valid_first_album_year
    if first_album_year < birth_year
      errors.add(:first_album_year, "no puede ser menor que el año de nacimiento del artista ")
    end
  end
end
```



Bonus Track

before_save

- Un `before_save` se ejecuta justo antes de guardar (ya sea al crear o modificar) recursos en la base de datos.
- Es útil de muchas formas. Para nuestro ejemplo, sabemos que el año de lanzamiento del primer disco siempre será el menor año de todos los discos que ha lanzado un artista. Por lo tanto, haremos de manera automática que se actualice el año cuando agreguemos un nuevo álbum.



before_save: models/album.rb

```
class Album < ApplicationRecord
  belongs_to :artist, dependent: :destroy
  has_many :songs
  before_save :update_artist_first_album_year

  def update_artist_first_album_year
    artist = Artist.find(artist_id)
    if year < artist.first_album_year
      artist.first_album_year = year
      artist.save
    end
  end
end
```