Associations between models

Luigi De Russis

Lab 6 – Warm Up

POLITECNICO
DI TORINO

e-Lite

Associations between models

Associations

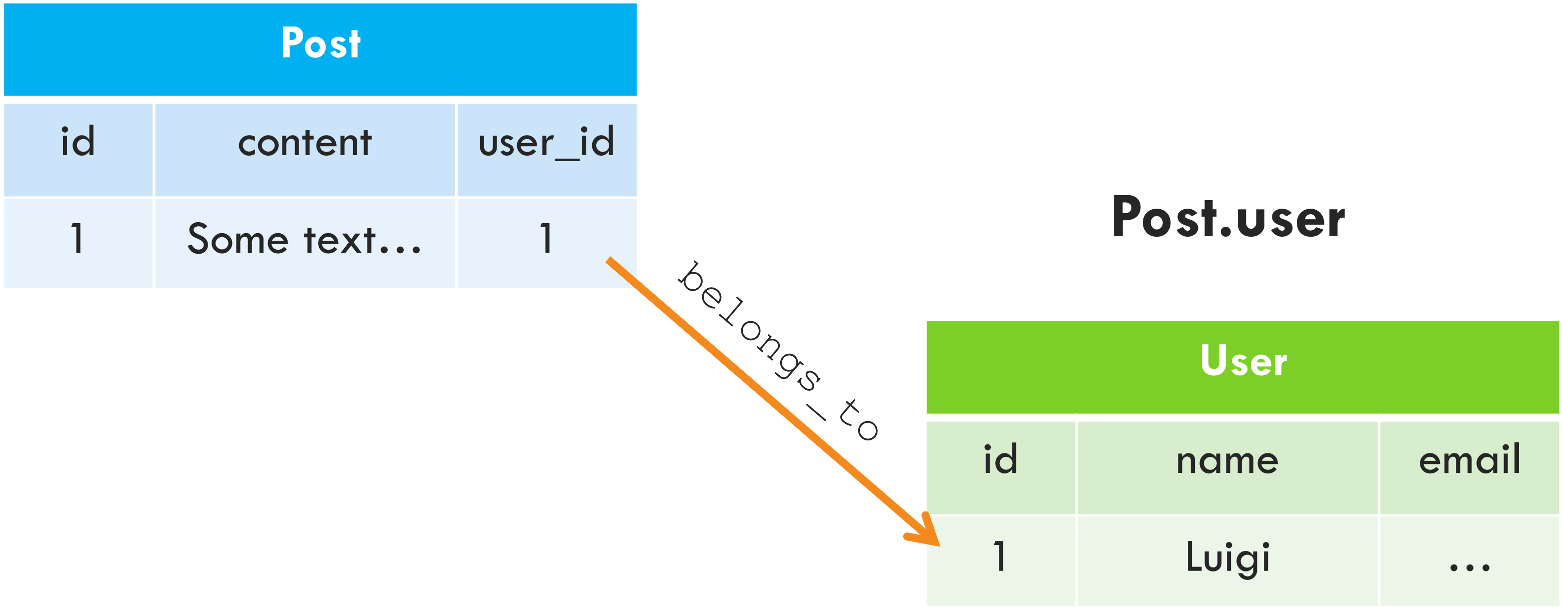
- `belong_to`
  - sets up a one-to-one connection with another model, such that each instance of the declaring model "belongs to" one instance of the other model
- `has_one`
  - sets up a one-to-one connection with another model, such that each instance of a model contains or possesses one instance of another model
- `has_many`
  - one-to-many connection between models
  - typically on the other side of a belong_to association

Associations

- `has_many, has_one :through`
  - similar to `has_many` and `has_one`
  - but the declaring model can be matched with one instance of another model by proceeding *through* a third model
- `has_and_belongs_to_many`
  - creates a direct many-to-many connection with another model, with no intervening model
- For each of them, Rails generates some methods

In our case…

- 1 post belongs to 1 user
  - in the Post model we'll have: `belongs_to :user`



Post		
id	content	user_id
1	Some text…	1

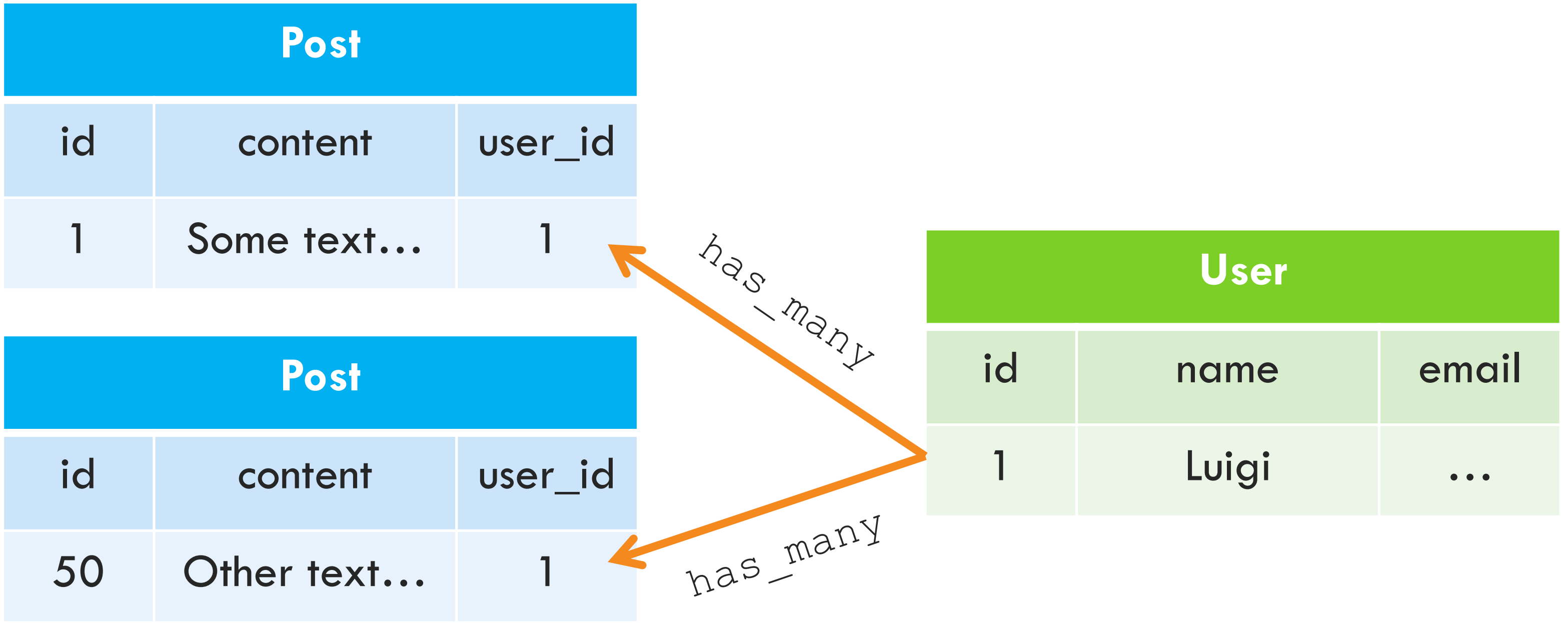
Post.user

User		
id	name	email
1	Luigi	…

In our case…

- 1 user can have many post
  - in the User model we'll have: `has_many :posts`

User.posts



Post		
id	content	user_id
1	Some text…	1

Post		
id	content	user_id
50	Other text…	1

User		
id	name	email
1	Luigi	…

In our case…

belongs_to	post.user	return the User object associated with the post
has_many	user.posts	return an Array of the user's posts
	user.posts.create(arg)	create a post (user_id = user.id)
	user.posts.create!(arg)	create a post, exception on failure
	user.posts.build(arg)	return a new Post object (user_id = user.id)

Some tips…

default_scope

- □ Use this macro in your model to set a default scope for all operations on the model
- □ Example

```
class Post < ActiveRecord::Base
   # descending order for getting the posts
   default_scope order: 'posts.created_at
        DESC`
end


Post.all #=> SELECT * FROM posts ORDER BY
   created_at DESC
```

Generate fake posts

- □ With the Faker gem

```
users = User.all(limit: 10)
50.times do
       post_title = Faker::Lorem.sentence(3)
       post_content = Faker::Lorem.paragraph
       users.each { |user| user.posts.create!(title:
             post_title, content: post_content )}
end
```

Generate fake posts

- With the Faker gem

```
users = User.all(limit: 10)
50.times do
     post_title = Faker::Lorem.sentence(3)
     post_content = Faker::Lorem.paragraph
     users.each { |user| user.posts.create!(title:
              post_title, content: post_content )}
end
```

First 10 users, only

50 posts

Lorem Ipsum sentences

License