



One-to-Many Association

One-to-Many Association

- One person *has one or more* jobs
- One job entry *belongs to* exactly one person
- The “*belongs to*” side is the one with a foreign key
- **Convention:** Default name for the foreign key is {master_table_singular}_id, e.g. person_id

Create Job Model and Migration

```
hazink1-ml1:advanced_ar hazink1$ rails g model job title company position_id person:references
  invoke  active_record
  create  db/migrate/20140619202236_create_jobs.rb
  create  app/models/job.rb
  invoke  test_unit
  create  test/models/job_test.rb
  create  test/fixtures/jobs.yml
hazink1-ml1:advanced_ar hazink1$ rake db:migrate
== 20140619202236 CreateJobs: migrating =====
-- create_table(:jobs)
   -> 0.0060s
== 20140619202236 CreateJobs: migrated (0.0061s) =====
```

OPEN FILES

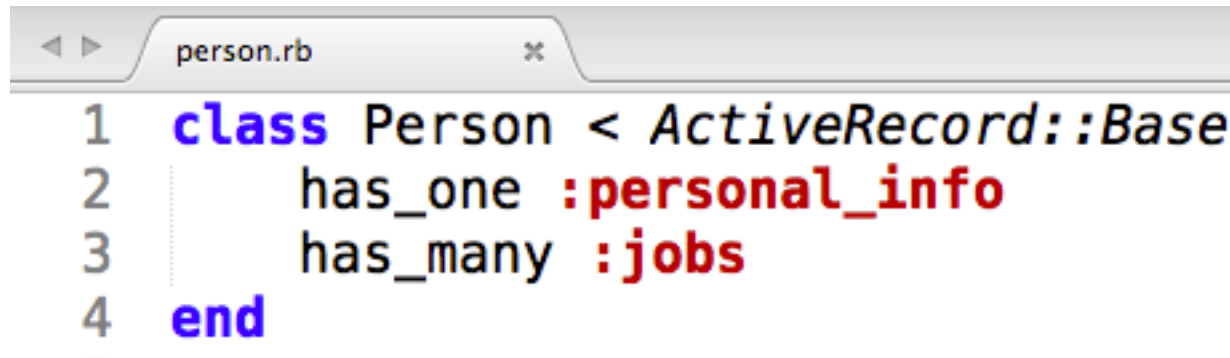
FOLDERS

- ▼ advanced_ar
 - ▶ app
 - ▶ bin
 - ▶ config
 - ▼ db
 - ▼ migrate
 - 20140617030159_create_people.r
 - 20140617034221_add_login_and_
 - 20140617202721_create_persona
 - 20140619202236_create_jobs.rb
 - development.sqlite3
 - schema.rb
 - seeds.rb
 - ▶ lib

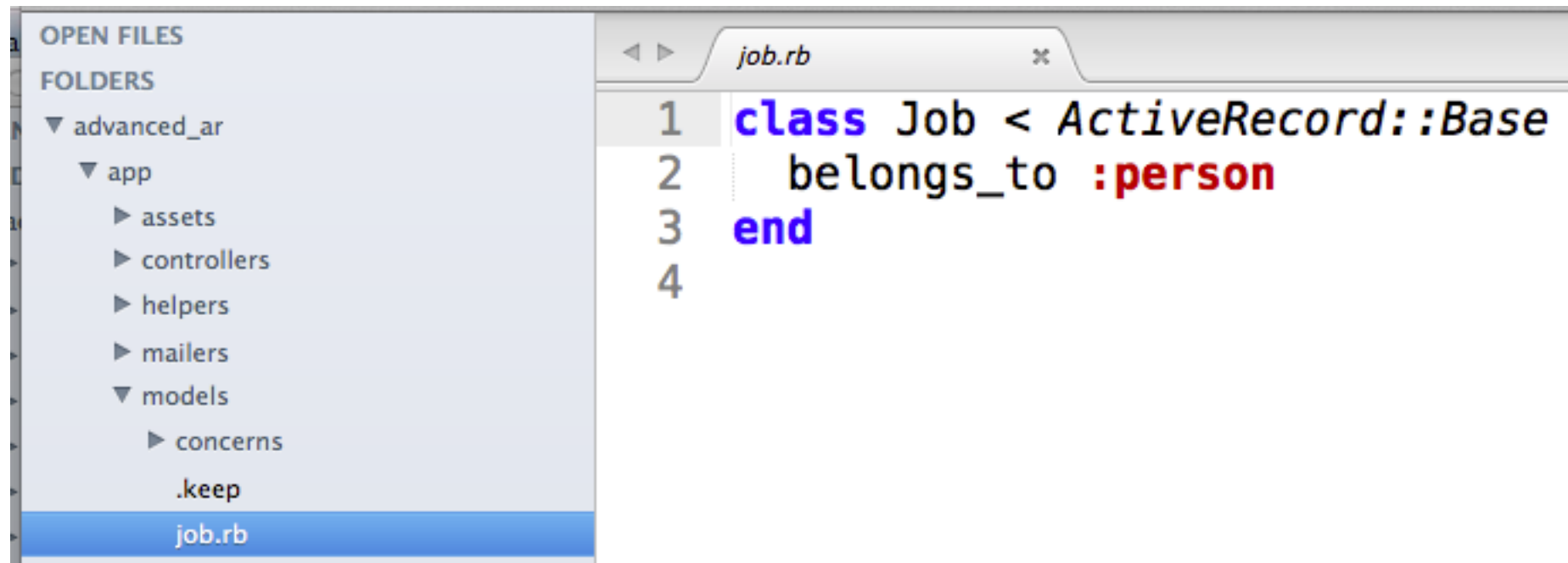
20140619202236_create_jobs.rb ✕

```
1 class CreateJobs < ActiveRecord::Migration
2   def change
3     create_table :jobs do |t|
4       t.string :title
5       t.string :company
6       t.string :position_id
7       t.references :person, index: true
8
9       t.timestamps
10    end
11  end
12 end
```

Modifying Person and Job models



```
1 class Person < ActiveRecord::Base
2   has_one :personal_info
3   has_many :jobs
4 end
```



OPEN FILES

FOLDERS

- ▼ advanced_ar
 - ▼ app
 - ▶ assets
 - ▶ controllers
 - ▶ helpers
 - ▶ mailers
 - ▼ models
 - ▶ concerns
 - .keep
- job.rb

```
1 class Job < ActiveRecord::Base
2   belongs_to :person
3 end
4
```

Person and Job in action

```
hazink1:~/advanced_ar$ rails c
Loading development environment (Rails 4.1.1)
irb(main):001:0> ActiveRecord::Base.logger = nil
=> nil
irb(main):002:0> Job.create company: "MS", title: "Developer", position_id: "#1234"
=> #<Job id: 2, title: "Developer", company: "MS", position_id: "#1234", person_id: nil,
  created_at: "2014-07-11 17:34:47", updated_at: "2014-07-11 17:34:47">
irb(main):003:0> p1 = Person.first
=> #<Person id: 8, first_name: "Kalman", last_name: "Smith", age: 33, created_at: "2014-
06-17 03:51:50", updated_at: "2014-06-17 03:51:50", login: "kalman", pass: "abc123">
irb(main):004:0> p1.jobs
=> #<ActiveRecord::Associations::CollectionProxy []>
irb(main):005:0> p1.jobs << Job.first
=> #<ActiveRecord::Associations::CollectionProxy [#<Job id: 2, title: "Developer", compa
ny: "MS", position_id: "#1234", person_id: 8, created_at: "2014-07-11 17:34:47", updated
_at: "2014-07-11 17:35:51">]>
irb(main):006:0> Job.first.person
=> #<Person id: 8, first_name: "Kalman", last_name: "Smith", age: 33, created_at: "2014-
06-17 03:51:50", updated_at: "2014-06-17 03:51:50", login: "kalman", pass: "abc123">
```

More methods

- `person.jobs = jobs`
 - Replaces existing jobs with a new array
 - As opposed to `person.jobs << job(s)` where the jobs are appended
- `person.jobs.clear`
 - Disassociates jobs from this person by setting the foreign key to NULL
- `create` and `where` methods for jobs become scoped to the `person`!

Scoped jobs

```
< > seeds.rb x
1 Job.destroy_all
2 Person.destroy_all
3
4 Person.create [
5   { first_name: "Kalman", last_name: "Smith", age: 33, login: "kalman", pass: "abc123" },
6   { first_name: "John", last_name: "Whatever", age: 27, login: "johnw", pass: "123abc" },
7   { first_name: "Michael", last_name: "Smith", age: 15, login: "michael", pass: "password" },
8   { first_name: "Josh", last_name: "Oreck", age: 57, login: "josh", pass: "not_telling" },
9   { first_name: "John", last_name: "Smith", age: 27, login: "john", pass: "secret" },
10  { first_name: "Bill", last_name: "Gates", age: 75, login: "bill", pass: "no_idea" },
11  { first_name: "LeBron", last_name: "James", age: 30, login: "lebron", pass: "lejames" }
12 ]
13
14 Person.first.jobs.create [
15   { title: "Developer", company: "MS", position_id: "#1234" },
16   { title: "Developer", company: "MS", position_id: "#1235" }
17 ]
18
19 Person.last.jobs.create [
20   { title: "Sr. Developer", company: "MS", position_id: "#1234" },
21   { title: "Sr. Developer", company: "MS", position_id: "#1235" }
22 ]
```

There is also a build version,
which does not automatically save
to DB

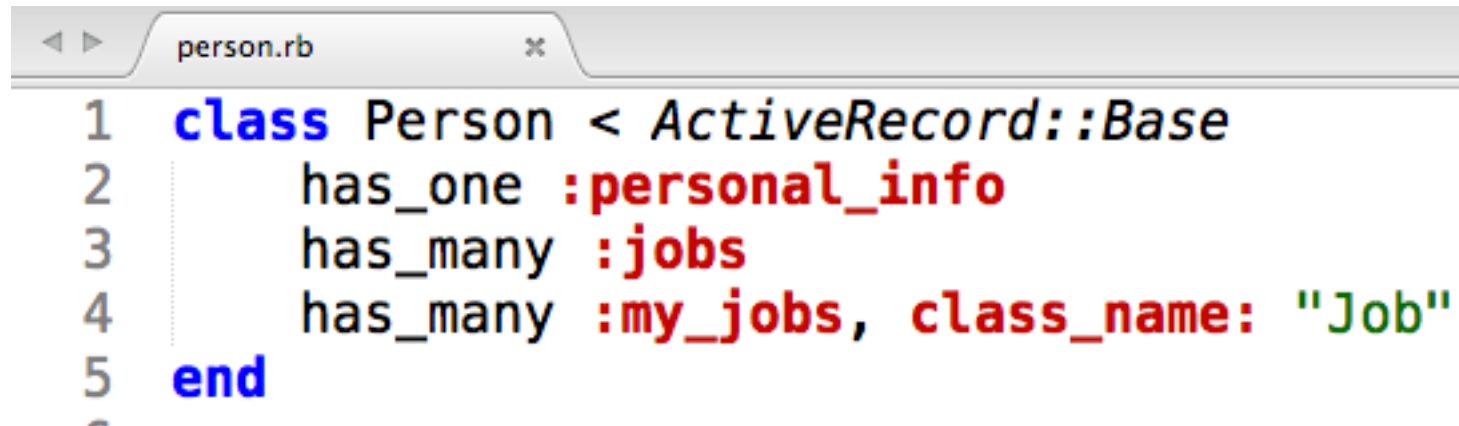
```
hazink1:~/advanced_ar$ rake db:seed
hazink1:~/advanced_ar$ _
```


Scoped jobs - where

```
hazink1:~/advanced_ar$ rails c
Loading development environment (Rails 4.1.1)
irb(main):001:0> ActiveRecord::Base.logger = nil
=> nil
irb(main):002:0> Person.first.jobs.where(company: "MS").to_a
=> [#<Job id: 3, title: "Developer", company: "MS", position_id: "#1234", person_id: 15, created_at: "2014-07-11 17:55:16", updated_at: "2014-07-11 17:55:16">, #<Job id: 4, title: "Developer", company: "MS", position_id: "#1235", person_id: 15, created_at: "2014-07-11 17:55:16", updated_at: "2014-07-11 17:55:16">]
irb(main):003:0> Person.last.jobs.where(company: "MS").to_a
=> [#<Job id: 5, title: "Sr. Developer", company: "MS", position_id: "#1234", person_id: 21, created_at: "2014-07-11 17:55:16", updated_at: "2014-07-11 17:55:16">, #<Job id: 6, title: "Sr. Developer", company: "MS", position_id: "#1235", person_id: 21, created_at: "2014-07-11 17:55:16", updated_at: "2014-07-11 17:55:16">]
irb(main):004:0> Job.where(company: "MS").order(title: :desc).to_a
=> [#<Job id: 5, title: "Sr. Developer", company: "MS", position_id: "#1234", person_id: 21, created_at: "2014-07-11 17:55:16", updated_at: "2014-07-11 17:55:16">, #<Job id: 6, title: "Sr. Developer", company: "MS", position_id: "#1235", person_id: 21, created_at: "2014-07-11 17:55:16", updated_at: "2014-07-11 17:55:16">, #<Job id: 3, title: "Developer", company: "MS", position_id: "#1234", person_id: 15, created_at: "2014-07-11 17:55:16", updated_at: "2014-07-11 17:55:16">, #<Job id: 4, title: "Developer", company: "MS", position_id: "#1235", person_id: 15, created_at: "2014-07-11 17:55:16", updated_at: "2014-07-11 17:55:16">]
```


Options for `has_many :class`

```
class_name: 'Modelname'
```



```
< > person.rb x
1 class Person < ActiveRecord::Base
2   has_one :personal_info
3   has_many :jobs
4   has_many :my_jobs, class_name: "Job"
5 end
```

```
irb(main):002:0> Person.last.my_jobs.to_a
=> [#<Job id: 5, title: "Sr. Developer", company: "MS", position_id: "#1234", person_id: 21, created_at: "2014-07-11 17:55:16", updated_at: "2014-07-11 17:55:16">, #<Job id: 6, title: "Sr. Developer", company: "MS", position_id: "#1235", person_id: 21, created_at: "2014-07-11 17:55:16", updated_at: "2014-07-11 17:55:16">]
```

:dependent

- `has_many`, `has_one` **and** `belongs_to` support `:dependent` option which lets you specify the fate of the association when the parent gets deleted / destroyed
 1. `:delete` – remove associated object(s)
 2. `:destroy` – same as above, but remove the association by calling `destroy` on it
 3. `:nullify` – set the FK to NULL (leave the association alone – just disassociate)

:dependent - Example

```
person.rb
1 class Person < ActiveRecord::Base
2   has_one :personal_info, dependent: :destroy
3   has_many :jobs
4   has_many :my_jobs, class_name: "Job"
5 end
```

:dependent in action

```
hazink1:~/advanced_ar$ rails c
Loading development environment (Rails 4.1.1)
irb(main):001:0> ActiveRecord::Base.logger = nil
=> nil
irb(main):002:0> michael = Person.find_by first_name: "Michael"
=> #<Person id: 17, first_name: "Michael", last_name: "Smith", age: 15, created_at: "2014-07-11 17:55:16", updated_at: "2014-07-11 17:55:16", login: "michael", pass: "password">
irb(main):003:0> michael.personal_info
=> #<PersonalInfo id: 3, height: 6.5, weight: 230.5, person_id: 17, created_at: "2014-07-13 03:10:49", updated_at: "2014-07-13 03:10:49">
irb(main):004:0> michael.destroy
=> #<Person id: 17, first_name: "Michael", last_name: "Smith", age: 15, created_at: "2014-07-11 17:55:16", updated_at: "2014-07-11 17:55:16", login: "michael", pass: "password">
irb(main):005:0> michael = Person.find_by first_name: "Michael"
=> nil
irb(main):006:0> PersonalInfo.pluck(:id)
=> [1, 2]
```