REFACTORING DATABASES WITH DELEGATES

@andrewfnewman

RAILS DELEGATES

```ruby
class Greeter < ActiveRecord::Base
  def hello() 'hello' end
  def goodbye() 'goodbye' end
end

class Foo < ActiveRecord::Base
  belongs_to :greeter
  delegate :hello, to: :greeter
end

Foo.new.hello   # => "hello"
Foo.new.goodbye # => NoMethodError
```

ADDRESS

id
street
city_name
postcode

MAP CITY TO POSTCODE

```
...
    let(:address) {
      Address1.create(street: "12 Smith Street",
        city_name: "Burpengary")
    }

    it { address.postcode.must_equal "4505" }

    it "set valid" do
      address.city_name = "Enoggera"
      address.postcode.must_equal "4051"
      address.save(reload: true)
      address.postcode.must_equal "4051"
   end
...
```

IMPLEMENTATION

```
class Address1 < ActiveRecord::Base
  validates :street, :city_name, :postcode, presence: true
  validates :postcode, length: {is: 4}
  validates :postcode, numericality: {only_integer: true}
  attr_accessible :street, :city_name

  CITY_POSTCODE = {"Burpengary" => "4505", "Enoggera" => "4051"}

  def city_name=(name)
    if CITY_POSTCODE.key?(name)
      self[:postcode] = CITY_POSTCODE[name]
      self[:city_name] = name
    end
  end
end
```

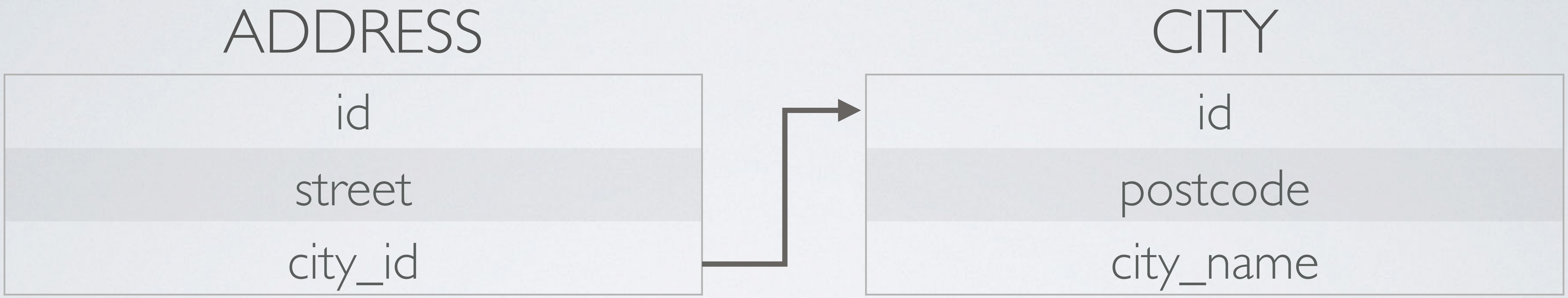
SORTA WORKS

```
...
    let(:address) {
      Address1.create(street: "12 Smith Street",
        city_name: "Burpengary")
    }

    it "can't sorta access postcode with create" do
      address.postcode = "1234"
      address.postcode.must_equal "1234"
      address.reload.postcode.must_equal “4505"
   end
...
```

CAN GO WRONG

```
...
    let(:address) {
      Address1.new(street: "12 Smith Street",
        city_name: “Burpengary”)
    }

    it { address.postcode.must_equal "4505" }

    it "can access postcode" do
      address.postcode = "1234"
      address.save(reload: true)
      address.postcode.must_equal "1234"
    end
...
```

SPLIT TABLE



ADDRESS
id
street
city_id

CITY
id
postcode
city_name

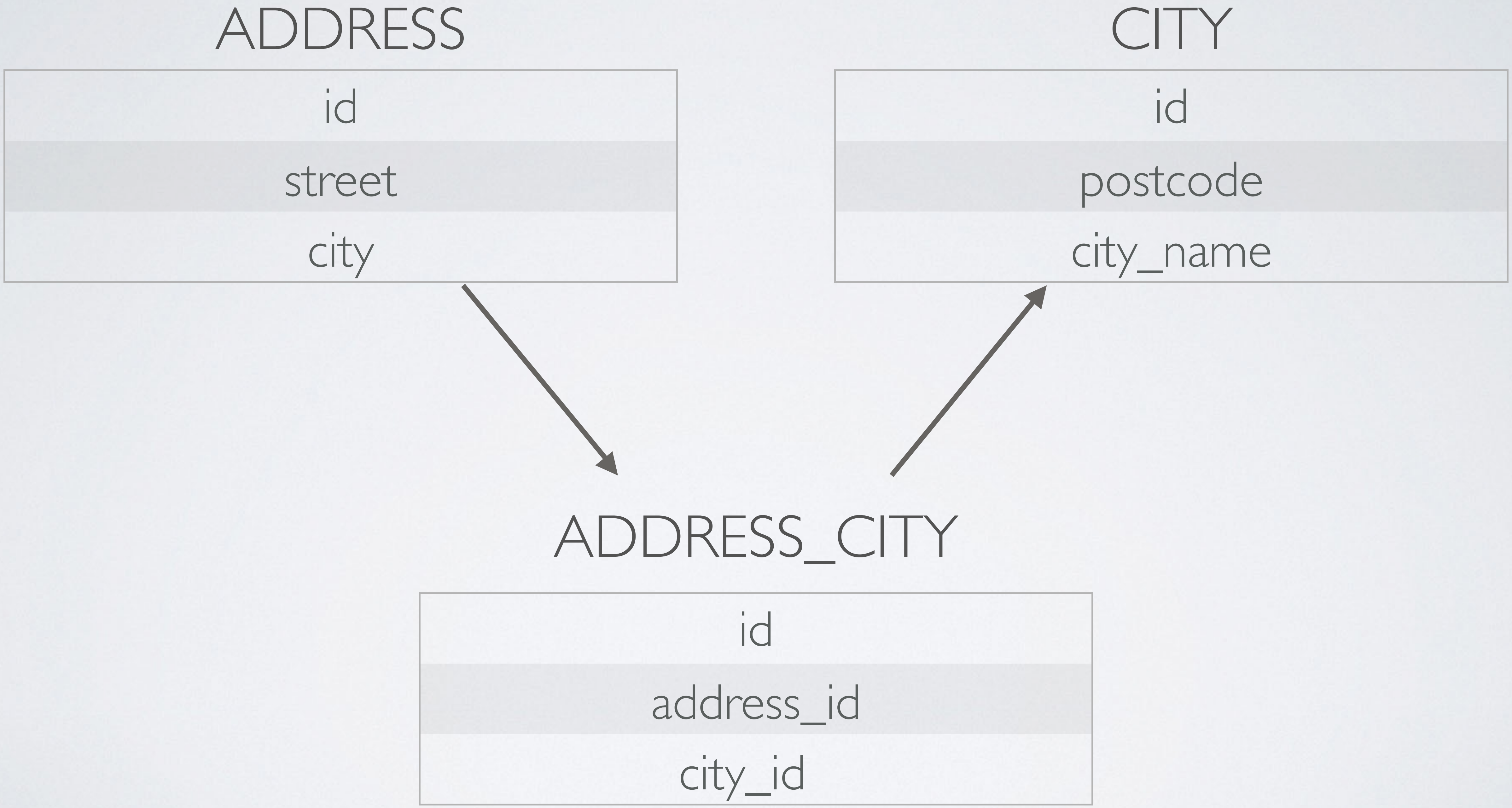
CITY IN ADDRESS

```
class Address2 < ActiveRecord::Base
  belongs_to :city2

  validates :street, :postcode, :city_name, presence: true
  delegate :city_name, :city_name=, :postcode, to: :city2,
    allow_nil: false

  def city_name=(name)
    city = City2.where(city_name: name).first
    self.city2 = city if city
  end
end
```

CITY WITH POSTCODE

```
class City2 < ActiveRecord::Base
  has_one :address2

  validates :postcode, :city_name, presence: true
  validates :postcode, length: {is: 4}
  validates :postcode, numericality: {only_integer: true}
end
```

ANOTHER WAY



ADDRESS
id
street
city

CITY
id
postcode
city_name

ADDRESS_CITY
id
address_id
city_id

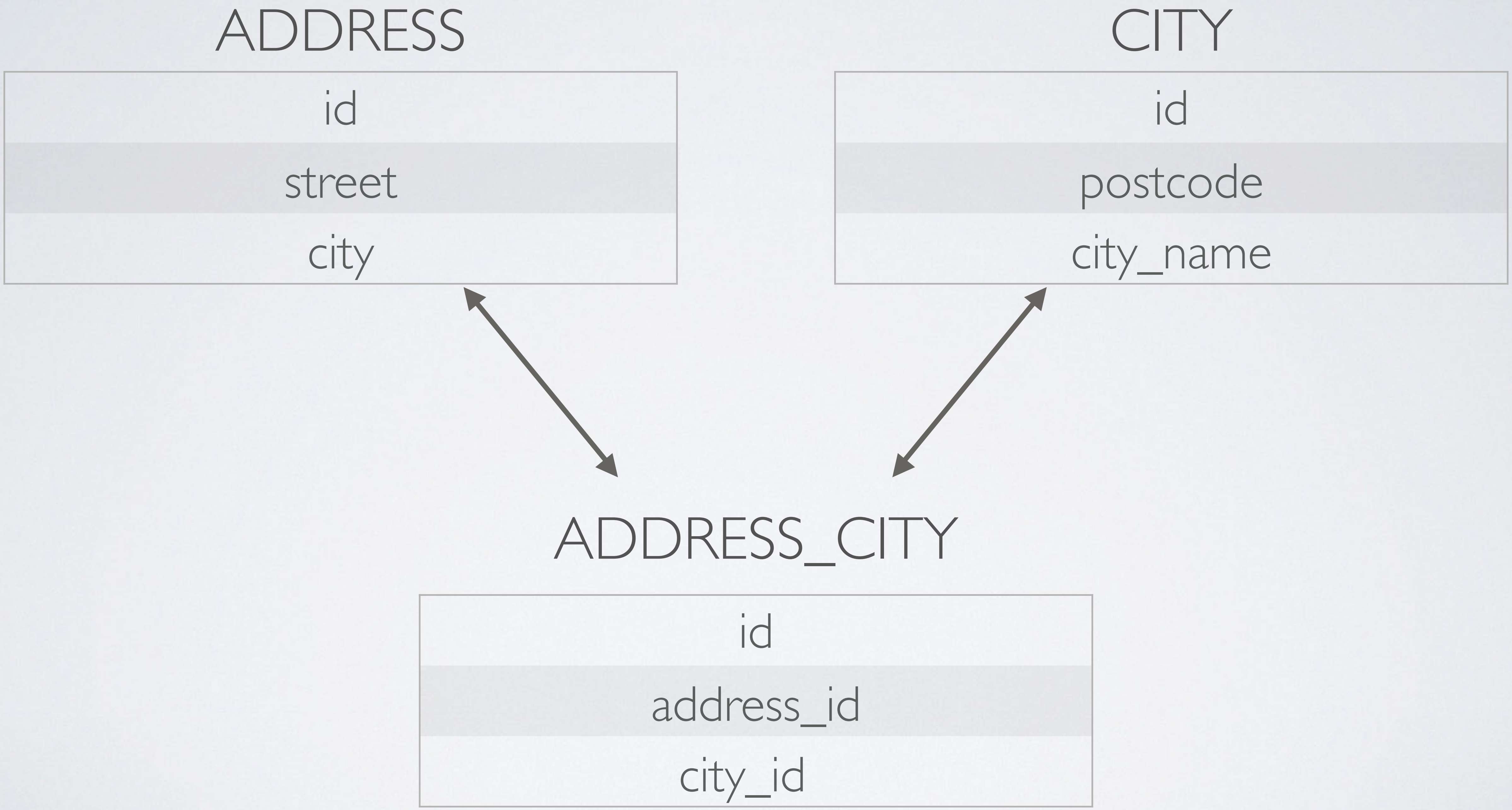
REFACTORED ADDRESS

```ruby
class Address3 < ActiveRecord::Base
  has_one :address3s_city3
  has_one :city3, through: :address3s_city3

  validates :street, :postcode, :city_name, presence: true
  delegate :city_name, :city_name=, :postcode, to: :city3,
    allow_nil: false

  def city_name=(name)
    city = City3.where(city_name: name).first
    self.city3 = city if city
  end
end
```

CITY AND LINK

```
class City3 < ActiveRecord::Base
  has_one :address3s_city3
  has_one :address3, through: :address3s_city3

  validates :postcode, :city_name, presence: true
  validates :postcode, length: {is: 4}
  validates :postcode, numericality: {only_integer: true}
end

class Address3sCity3 < ActiveRecord::Base
  belongs_to :address3
  belongs_to :city3
end
```

REALLY



ADDRESS
id
street
city

CITY
id
postcode
city_name

ADDRESS_CITY
id
address_id
city_id

LINKS

Rails Documentation

http://apidock.com/rails/Module/delegate

Understanding Ruby and Rails: Delegate

http://simonecarletti.com/blog/2009/12/inside-ruby-on-rails-delegate/

Multiple ways in implementing delegation patter in ruby

http://gdakram.com/past/2010/12/2/multiple_ways_in_implementing_delegation_pattern_in_ruby/