

Standard Assertions

assert	boolean
assert_equal	expected, actual
assert_raise	*args
assert_raises	*args, &block
assert_instance_of	klass, object
assert_nil	object
assert_kind_of	klass, object
assert_respond_to	object, method
assert_match	pattern, string
assert_same	expected, actual
assert_operator	object1, operator, object2
assert_nothing_raised	*args
assert_not_same	expected, actual
assert_not_equal	expected, actual
assert_not_nil	object
assert_no_match	regexp, string
assert_throws	expected_symbol, &proc
assert_nothing_thrown	&proc
assert_in_delta	expected_float, actual_float, delta
assert_send	send_array

Rails Assertions

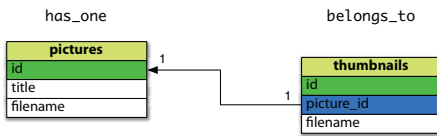
assert_response	type
assert_redirected_to	options = {}
assert_template	expected
assert_recognizes	expected_options, path, extras={}
assert_generates	expected_path, options, defaults={}, extras = {}
assert_routing	path, options, defaults={}, extras={}
assert_tag	*opts
assert_no_tag	*opts
assert_dom_equal	expected, actual
assert_dom_not_equal	expected, actual
assert_valid	record

Test::Rails Assertions

assert_assigned	ivar, value = NOTHING
• deny_assigned	
assert_content_type	type, message = nil
assert_flash	key, content
assert_image	src
assert_error_on	field, type
assert_field	form_action, type, model, column, value = nil
assert_input	form_action, type, name, value = nil
assert_label	form_action, name, include_f = true
assert_links_to	href, content = nil
• deny_links_to	
assert_multipart_form	form_action
assert_post_form	form_action
assert_select	form_action, model, column, options
assert_submit	form_action, value
assert_tag_in_form	form_action, options
• deny	
assert_empty	obj
• deny_empty	
assert_includes	obj, item, message = nil
• deny_includes	

Most also take a message argument as the last parameter. The message will be shown if the test fails.

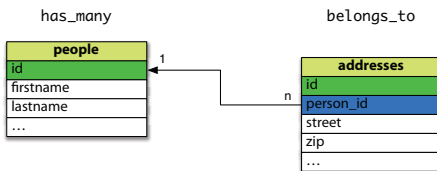
has_one



```
class Picture < ActiveRecord::Base
  has_one :thumbnail
end

class Thumbnail < ActiveRecord::Base
  belongs_to :picture
end
```

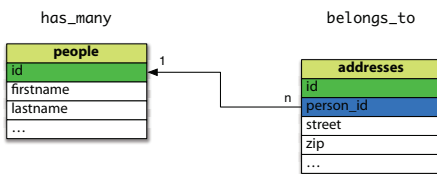
belongs_to



```
class Picture < ActiveRecord::Base
  has_one :thumbnail
end

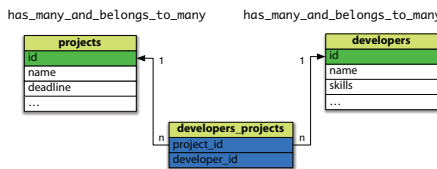
class Thumbnail < ActiveRecord::Base
  belongs_to :picture
end
```

has_many



```
class Person < ActiveRecord::Base
  has_many :addresses
end
```

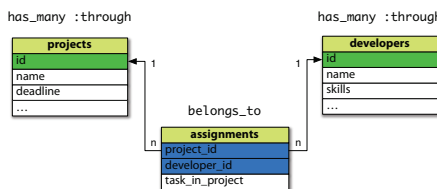
has_many_and_belongs_to



```
class Project < ActiveRecord::Base
  has_and_belongs_to_many :developers
end

class Developer < ActiveRecord::Base
  has_and_belongs_to_many :projects
end
```

has_many :through



```
class Project < ActiveRecord::Base
  has_many :developers, :through => assignments
end

class Assignment < ActiveRecord::Base
  belongs_to :developer
  belongs_to :project
end

class Developer < ActiveRecord::Base
  has_many :projects, :through => assignments
end
```


ERROR: undefined
OFFENDING COMMAND: eexec

STACK:

/quit
-dictionary-
-mark-