Use find_by instead of where.first to remove unnecessary sorting

sincen where.first will issue query with order by primary key which is actually unnecessary.

redmine/app/helpers/application_helper.rb:909:        u = User.visible.where(:id => oid, :type => 'User').first
redmine/app/helpers/application_helper.rb:911:        if repository && (changeset =
Changeset.visible.where("repository_id = ? AND scmid LIKE %", repository.id, "#(name)%").first)
redmine/app/helpers/application_helper.rb:966:        if p = Project.visible.where("identifier = ? OR LOWER(name) = ?", p => name.downcase).first
redmine/app/helpers/application_helper.rb:970:        u = User.visible.where(:login => name, :type => 'User').first
redmine/app/models/token.rb:115:    token = Token.where(:action => action, :value => key).first
redmine/app/models/user.rb:196:    user = where(:identity_url => url).first
redmine/app/models/user.rb:489:        user = where("LOWER(login) = ?", login.downcase).first
redmine/app/models/user.rb:560:      h[project_id] = memberships.where(:project_id => project_id).first
redmine/app/models/wiki.rb:59:    redirect = redirects.where("LOWER(title) = LOWER(?)", title).first
redmine/app/models/attachment.rb:271:    attachment = Attachment.where(:id => attachment_id, :digest =>
attachment_digest).first
redmine/app/models/time_entry_query.rb:150:    issue = Issue.where(:id => value.first.to_i).first
redmine/app/models/repository.rb:252:      changesets.where("revision = ?", s).first
redmine/app/models/repository.rb:254:      changesets.where("revision LIKE %", s + %).first
redmine/app/models/role.rb:296:      role = unscoped.where(:builtin => builtin).first
redmine/app/models/repository/git.rb:92:      changesets.where(revision => name.to_s).first ||
redmine/app/models/repository/git.rb:93:      changesets.where("scmid LIKE %", "(#(name)%)").first
redmine/app/models/repository/mercurial.rb:100:    cs = changesets.where("scmid => s").first
redmine/app/models/repository/mercurial.rb:102:    cs = changesets.where(revision => s).first
redmine/app/models/mercurial.rb:105:    changesets.where("scmid LIKE %", "(#(name)%)").first
redmine/app/models/mercurial.rb:107:    changesets.where("scmid LIKE %", "(#(name)%)").first
redmine/app/controllers/imports_controller.rb:100:    @import = Import.where(:user_id => User.current.id, :filename =>
params[:id]).first
Use find_by instead of where.first to remove unnecessary sorting (#26747).

Patch by Yuichi HARADA.

History

#1 - 2017-08-23 06:58 - Toshi MARUYAMA
- Category set to Code cleanup/refactoring
- Status changed from New to Needs feedback

Please post by attachment patch file.

#2 - 2017-08-23 07:15 - Toshi MARUYAMA

jwjw yy wrote:

sicne where.first will issue query with order by primary key which is actually unnecessary.

Really? Do all your description "where"s call "order by"?

#3 - 2017-09-12 19:21 - jwjw yy

Toshi MARUYAMA wrote:

jwjw yy wrote:

sicne where.first will issue query with order by primary key which is actually unnecessary.

Really? Do all your description "where"s call "order by"?

yes

#4 - 2018-07-16 22:30 - jwjw yy

Removed advertisement -- Holger Just

#5 - 2018-07-29 03:25 - Go MAEDA
- Subject changed from use find_by to replace where.first to Use find_by instead of where.first to remove unnecessary sorting
- Category changed from Code cleanup/refactoring to Performance
- Status changed from Needs feedback to New
- Target version set to Candidate for next major release
Toshi MARUYAMA wrote:

> *jwjw yy wrote:*

> sicne where.first will issue query with order by primary key which is actually unnecessary.

Really? Do all your description "where"s call "order by"?

It is documented in the Rails API reference.

https://api.rubyonrails.org/classes/ActiveRecord/FinderMethods.html#method-i-first

Find the first record (or first N records if a parameter is supplied). **If no order is defined it will order by primary key.**

> User.where(:login => 'admin').first

User Load (0.4ms)  SELECT "users".* FROM "users" WHERE "users"."type" IN ('User', 'AnonymousUser') AND "users"."login" = ? ORDER BY "users"."id" ASC LIMIT ?  
[
 ["login", "admin"], 
 ["LIMIT", 1]
 ]

> User.find_by(:login => 'admin')

User Load (0.4ms)  SELECT "users".* FROM "users" WHERE "users"."type" IN ('User', 'AnonymousUser') AND "users"."login" = ? LIMIT ?  
[
 ["login", "admin"], 
 ["LIMIT", 1]
 ]

---

#6 - 2018-08-12 11:25 - Go MAEDA

find_by is significantly faster than where.first.

```ruby
require 'benchmark/ips'

Benchmark.ips do |x|
  x.report('where.first') { User.where(:login => 'admin').first }
  x.report('find_by') { User.find_by(:login => 'admin') }
end

$ bin/rails r bench-26747.rb

Warming up -----------------------------
  where.first 80.000 i/100ms
  find_by 165.000 i/100ms
Calculating ---------------------------
  where.first 961.953 (± 5.5%) i/s -  4.800k in  5.005696s
  find_by 1.664k (± 4.6%) i/s -  8.415k in  5.066908s
```
I changed `where.first` to `find_by` as follows.

1. "Primary key" is specified as a condition
2. "Unique index column" is specified as a condition
3. Only one registration like `AnonymousUser`

Committed. Thank you for improving Redmine.