redmineorg-copy202205 - Vote #77633

Improve Update/Create issue speed

2022/05/09 18:01 - Admin Redmine

ステータス:	New	**開始日:**	2022/05/09
優先度:	通常	**期日:**	
担当者:		**進捗率:**	0%
カテゴリ:	Performance_53	**予定工数:**	0.00時間
対象バージョン:	Candidate for next major release_32	**作業時間:**	0.00時間
Redmineorg_URL:	https://www.redmine.org/issues/23328	**status_id:**	1
category_id:	53	**tracker_id:**	3
version_id:	32	**plus1:**	0
issue_org_id:	23328	**affected_version:**	
author_id:	118650	**closed_on:**	
assigned_to_id:	1	**affected_version_id:**	
comments:	7		

説明

Hi guys,
When Redmine look for what members it should send e-mail, they interate one by one fetching principal.
This is a N + 1 Query problem.

When we have more then 5K users in one project it is a problem. So with a single line change I drop the time for update issue from 5 to 2 seconds.

I hope this help you.

```
Date: Tue Jul 12 19:37:14 2016 -0300

    improve update/create speed

diff --git a/app/models/project.rb b/app/models/project.rb
index 660a486..88bd8eb 100644
--- a/app/models/project.rb
+++ b/app/models/project.rb
@@ -524,7 +524,7 @@ class Project < ActiveRecord::Base
     # Returns the users that should be notified on project events
     def notified_users
       # TODO: User part should be extracted to User#notify_about?
-      members.select {|m| m.principal.present? && (m.mail_notification? || m.principal.mail_notification == 'all')}.collect {|
ncipal}
+      members.includes(:principal).select {|m| m.principal.present? && (m.mail_notification? || m.principal.mail_notification =
)}.collect {|m| m.principal}
     end

     # Returns a scope of all custom fields enabled for project issues
```

journals

I resolved this issue a bit different. I changed the *includes* to *eager_load* to explicitly eager load the *principal* association and added a *find_each* to save memory when the quantity of members is to big.

```
members.eager_load(:principal).find_each()
                  .select {|m| m.principal.present? && (m.mail_notification? || m.principal.mail_notification == 'all')}
                  .collect {|m| m.principal}
```

I've made some tests and these are the results:
_. # of project members	_. current method	_. after patch
>. 6024	>. 6.13s	>. 1.15s
>. 7933	>. 7.57s	>. 1.40s

¦>. 7935 ¦>. 7.46s ¦>. 1.32s ¦

Yes, for memory it's a better solution.
=)

Thx for this patch

Redmine 3.3.0 uses preload method in Project#notified_users. Please see r15518.
Could you test Redmine 3.3.0?

Go MAEDA wrote:

> Redmine 3.3.0 uses preload method in Project#notified_users. Please see r15518.
> Could you test Redmine 3.3.0?

Hi Go MAEDA,
What is the policy for update redmine stable branch? When 3.3-stable was lunch I update my redmine for it. When I read your comments I realided that there is a lot off new commits, with new features (redmine.lib changed a lot), performance issues fixed, etc.

About this issue, why preload and not eager_load? And I think the Lucas's idea with find_each is good to prevent memory problems.

Thanks for the quick feedback.

Victor Campos wrote:

> What is the policy for update redmine stable branch? When 3.3-stable was lunch I update my redmine for it. When I read your comments I realided that there is a lot off new commits, with new features (redmine.lib changed a lot), performance issues fixed, etc.

I am not a commiter, so I can't explain about the policy. But as I know, the branch was used to prepare releasing of 3.3.0. Many revisions were merged from trunk before 3.3.0 is released.

> About this issue, why preload and not eager_load? And I think the Lucas's idea with find_each is good to prevent memory problems.

I would like Jean-Philippe Lang to make a judgment. Setting assignee to Jean-Philippe.

履歴

#1 - 2022/05/10 17:07 - Admin Redmine

- カテゴリ を Performance_53 にセット

- 対象バージョン を Candidate for next major release_32 にセット