

Redmine - Patch #38198

Improve MySQL query plan for Project#project\_condition

2023-01-19 15:25 - Holger Just

Status:	Closed	Start date:	
Priority:	Normal	Due date:	
Assignee:	Go MAEDA	% Done:	0%
Category:	Performance	Estimated time:	0.00 hour
Target version:	5.1.0		

**Description**

On a large installations (>30k projects, >500k issues), rendering the issue statistics on a project overview page can take a long time. We observed runtimes for the SQL queries of > 5 seconds for the queries generated in ProjectsController#show

```
@open_issues_by_tracker = Issue.visible.open.where(cond).group(:tracker).count
@total_issues_by_tracker = Issue.visible.where(cond).group(:tracker).count

@total_hours = TimeEntry.visible.where(cond).sum(:hours).to_f
@total_estimated_hours = Issue.visible.where(cond).sum(:estimated_hours).to_f
```

As an example, the query plan for one query for @total\_estimated\_hours was as follows:

id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	PRIMARY	issues	ALL	issues_project_id	NULL	NULL	NULL	448033	Using where
1	PRIMARY	projects	eq_ref	PRIMARY, index_projects_on_lft, index_projects_on_rgt	PRIMARY	4	redmine.issues.project_id	1	Using where
3	SUBQUERY	members	range	index_members_on_user_id_and_project_id, index_members_on_user_id, index_members_on_project_id	index_members_on_user_id_and_project_id	4	NULL	8	Using where; Using index
2	DEPENDENT SUBQUERY	em	ref	enabled_modules_project_id	enabled_modules_project_id	5	redmine.projects.id	4	Using where

This query took more than 5 seconds in MySQL. The query used both the projects.id as well as the projects.lft / project.rgt columns. This caused MySQL to perform a table scan on the (large) time\_entries or issues tables followed by an index-scan on the projects table.

With the change in the attached patch change, MySQL first filters the projects followed by the issues/ time entries. This allows MySQL to use the project\_id index on the issues table after performing a table scan on the (smaller) projects table. The query plan for this improved query is:

id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	PRIMARY	projects	ALL	PRIMARY, index_proj	NULL	NULL	NULL	39606	Using where

				ects_on_lft ,index_proj ects_on_rg t					
1	PRIMARY	issues	ref	issues_pro ject_id	issues_pro ject_id	4	hostedred mine.proje cts.id	14	Using where
3	SUBQUER Y	members	range	index_me mbers_on _user_id_a nd_project _id,index_ members_ on_user_id ,index_me mbers_on _project_id	index_me mbers_on _user_id_a nd_project _id	4	NULL	8	Using where; Using index
2	DEPENDEN T SUBQUER Y	em	ref	enabled_m odules_pro ject_id	enabled_m odules_pro ject_id	5	hostedred mine.proje cts.id	4	Using where

The attached patch improves the query plan selected by MySQL and results in a query which finishes in about 50ms (100 times faster).

The new query is equivalent to Project.self\_and\_descendants (in lib/redmine/nested\_set/traversing.rb), as was semantically the old one.

Associated revisions

Revision 22069 - 2023-01-21 09:50 - Go MAEDA

Improve index usability for Project#project\_condition (#38198).

Patch by Holger Just.

History

#1 - 2023-01-20 09:47 - Go MAEDA

- Category changed from Database to Performance
- Target version set to 5.1.0

Setting the target version to 5.1.0.

#2 - 2023-01-21 09:50 - Go MAEDA

- Status changed from New to Closed
- Assignee set to Go MAEDA

Committed the patch. Thank you for your contribution.

Files

0001-Improve-index-usability-for-Project-project_conditio.patch	2.19 KB	2023-01-19	Holger Just
---	---------	------------	-------------