

Redmine - Defect #37268

Performance problem with Redmine 4.2.7 and 5.0.2

2022-06-21 20:40 - Alexander Meindl

Status:	Closed	Start date:	
Priority:	Normal	Due date:	
Assignee:	Marius BĂLTEANU	% Done:	0%
Category:	Issues list	Estimated time:	0.00 hour
Target version:	4.2.8	Affected version:	4.2.7
Resolution:	Fixed		
Description With r21646 especially this change: https://www.redmine.org/projects/redmine/repository/revisions/21646/diff/trunk/app/models/issue_custom_field.rb <pre>"((#{sql}) AND (#{tracker_condition}) AND (#{project_condition}))"</pre> to <pre>"((#{sql}) AND (#{tracker_condition}) AND (#{project_condition}) AND (#{Issue.visible_condition(user)}))"</pre> we got massive performance problems on issue list. Some information to the system environment: <ul style="list-style-type: none">• Redmine 5.0.2 (for this issue I selected 5.0.1, because there is no 5.0.2 to select)• Ruby 3.0.2• PostgreSQL 14• 150.000 issues• 500 projects• 40 custom fields for issues			
Related issues:			
Has duplicate Redmine - Defect #37539: Issue index filtering with custom fiel...		Closed	
Has duplicate Redmine - Patch #37565: Performance problem when filtering issu...		Closed	

Associated revisions

Revision 21833 - 2022-09-23 07:51 - Marius BĂLTEANU

Fix perfomance issue when objects are filtered by custom field values (#37268).

Revision 21847 - 2022-09-26 23:44 - Marius BĂLTEANU

Merge r21833 to 5.0-stable (#37268).

Revision 21848 - 2022-09-26 23:44 - Marius BĂLTEANU

Merge r21833 to 4.2-stable (#37268).

History

#1 - 2022-06-21 20:43 - Alexander Meindl

Sorry for syntax error in my code, I cannot edit it.

Just an info to execution time: if I change this line above back to the old version, database query needs 3 seconds. With the new security fix same query needs 59 seconds.

I found the problem: the created subquery with Issue.visible_condition(user) needs a relation to the main query, which is missing right know. We need main_query_issue.id = subquery_issue.id in subquery. This would fix the problem.
I am not sure, how this could be solved with Issue.visible_condition abstraction.

#2 - 2022-06-21 23:24 - Holger Just

- Description updated

Hello,

[holger mareck](#): thanks for fixing my broken syntax

More details to the problem: it appears, if a custom field is used as a filter.

We have a custom field with format "list" and used this filter with one selected value (operator => "="). If this filter is used, most of the time the database query needs forever (at least some minutes).

This is the created sql query:

```
SELECT COUNT(*)
FROM "issues"
INNER JOIN "projects" ON "projects"."id" = "issues"."project_id"
INNER JOIN "issue_statuses" ON "issue_statuses"."id" = "issues"."status_id"
WHERE (PROJECTS.STATUS <> 9
      AND EXISTS
        (SELECT 1 AS ONE
         FROM ENABLED_MODULES EM
         WHERE EM.PROJECT_ID = PROJECTS.ID
              AND EM.NAME = 'issue_tracking'))
AND ((ISSUES.STATUS_ID IN
      (SELECT ID
       FROM ISSUE_STATUSES
       WHERE IS_CLOSED = FALSE))
AND ISSUES.ID IN
 (SELECT ISSUES.ID
  FROM ISSUES
  LEFT OUTER JOIN CUSTOM_VALUES ON CUSTOM_VALUES.CUSTOMIZED_TYPE = 'Issue'
  AND CUSTOM_VALUES.CUSTOMIZED_ID = ISSUES.ID
  AND CUSTOM_VALUES.CUSTOM_FIELD_ID = 61
  WHERE (CUSTOM_VALUES.VALUE IN ('Alignment'))
  AND ((1 = 1)
      AND (ISSUES.TRACKER_ID IN
          (SELECT TRACKER_ID
           FROM
CUSTOM_FIELDS_TRACKERS
          WHERE CUSTOM_FIELD_ID
= 61))
      AND (EXISTS
          (SELECT 1
           FROM CUSTOM_FIELDS IFA
           WHERE IFA.IS_FOR_ALL =
TRUE
              AND IFA.ID = 61)
          OR ISSUES.PROJECT_ID IN
            (SELECT PROJECT_ID
             FROM
CUSTOM_FIELDS_PROJECTS
             WHERE CUSTOM_FIELD_ID
= 61))
      AND (PROJECTS.STATUS <> 9
          AND EXISTS
            (SELECT 1 AS ONE
             FROM ENABLED_MODULES
             WHERE EM.PROJECT_ID =
PROJECTS.ID
              AND EM.NAME =
'issue_tracking'))))))))
```

and this is a test to fix it manually by sql (with "AND main_query.ID = ISSUES.ID"):

```
SELECT COUNT(*)
FROM "issues" AS main_query
INNER JOIN "projects" ON "projects"."id" = main_query."project_id"
INNER JOIN "issue_statuses" ON "issue_statuses"."id" = main_query."status_id"
WHERE (PROJECTS.STATUS <> 9
      AND EXISTS
        (SELECT 1 AS ONE
         FROM ENABLED_MODULES EM
         WHERE EM.PROJECT_ID = PROJECTS.ID
              AND EM.NAME = 'issue_tracking'))
AND ((main_query.STATUS_ID IN
```

```

        (SELECT ID
          FROM ISSUE_STATUSES
          WHERE IS_CLOSED = FALSE))
AND main_query.ID IN
  (SELECT ISSUES.ID
   FROM ISSUES
   LEFT OUTER JOIN CUSTOM_VALUES ON CUSTOM_VALUES.CUSTOMIZED_TYPE = 'Issue'
   AND main_query.ID = ISSUES.ID
   AND CUSTOM_VALUES.CUSTOMIZED_ID = ISSUES.ID
   AND CUSTOM_VALUES.CUSTOM_FIELD_ID = 61
   WHERE (CUSTOM_VALUES.VALUE IN ('Alignment'))
        AND ((1 = 1)
              AND (ISSUES.TRACKER_ID IN
                    (SELECT TRACKER_ID
                     FROM
CUSTOM_FIELDS_TRACKERS
                     WHERE CUSTOM_FIELD_ID
= 61))
              AND (EXISTS
                    (SELECT 1
                     FROM CUSTOM_FIELDS IFA
                     WHERE IFA.IS_FOR_ALL =
TRUE
                     AND IFA.ID = 61)
                    OR ISSUES.PROJECT_ID IN
                    (SELECT PROJECT_ID
                     FROM
CUSTOM_FIELDS_PROJECTS
                     WHERE CUSTOM_FIELD_ID
= 61))
              AND (PROJECTS.STATUS <> 9
                    AND EXISTS
                    (SELECT 1 AS ONE
                     FROM ENABLED_MODULES
                     WHERE EM.PROJECT_ID =
PROJECTS.ID
                     AND EM.NAME =
'issue_tracking'))))))

```

#4 - 2022-06-22 07:55 - Alexander Meindl

- File *custom_field_visibility_fix.patch* added

Hi,

I attached a fix, which solves the problem. Question is, are there any other problems with custom field filters.

#5 - 2022-06-22 07:56 - Alexander Meindl

- File *custom_field_visibility_fix_v2.patch* added

Here is the right version of the patch.

#6 - 2022-06-22 08:02 - Alexander Meindl

- File *custom_field_visibility_fix_v3.patch* added

And this version uses an abstract name for subquery (because it is not only issue related).

#8 - 2022-06-22 18:52 - Alexander Meindl

- File *custom_field_visibility_fix_v4.patch* added

This version fixes errors with VersionCustomFields for issues.

Some tests for queries are broken, because filter operator "none" (!*) does not work anymore with this patch.

#9 - 2022-06-23 08:24 - Simon Hori

We also got a serious performance issue after upgrade from **4.2.6** to **4.2.7** as well.

The same change has been applied to 4.2.7.

Thank you very much for your time for this issue.

#10 - 2022-06-23 11:05 - Marius BĂLTEANU

- Target version set to 4.2.8
- Affected version changed from 5.0.1 to 4.2.6

#11 - 2022-06-23 14:51 - Holger Just

Simon, which database type (and version) are you using?

#12 - 2022-06-24 01:52 - Go MAEDA

- Affected version changed from 4.2.6 to 4.2.7

#13 - 2022-06-25 16:06 - Marius BĂLTEANU

- Subject changed from Performance problem with Redmine 5.0.2 to Performance problem with Redmine 4.2.7 and 5.0.2
- Status changed from New to Confirmed

I've confirmed the performance issue with:

- 5000 issues
- 500 projects
- 40 custom fields for issues

4.2.7: Query runs in ~ 5 seconds
4.2.6: Query runs in ~ 0,5 seconds.

Holger, Felix, should we revert the fix until we have this fixed properly?

#14 - 2022-06-26 20:01 - Holger Just

The fix was security related and should not be reverted, lest we re-introduce the underlying security issue. However, we may want to fix the performance issue.

Marius: Could you please confirm which database you observed the issue on? Could you please also show the emitted SQL queries for both versions, in addition to an EXPLAIN of both queries?

#15 - 2022-06-27 09:30 - Marius BĂLTEANU

Holger Just wrote:

The fix was security related and should not be reverted, lest we re-introduce the underlying security issue. However, we may want to fix the performance issue.

Marius: Could you please confirm which database you observed the issue on? Could you please also show the emitted SQL queries for both versions, in addition to an EXPLAIN of both queries?

I've tested with postgres:14-alpine, I'll be back with the explains and the scripts used to generate the necessary fixtures.

#16 - 2022-06-27 21:13 - Marius BĂLTEANU

- File explain_with_visible_condition.csv added
- File sql_with_visible_condition.sql added
- File explain_without_visible_condition.csv added
- File sql_without_visible_condition.sql added
- File performance_fixtures.rb added

Holger, I've attached the requested details:

1. SQL and explain from PGAdmin for the query with visible condition
2. SQL and explain from PGAdmin for the query before the security fix was implemented (without visible condition).

I made very basic performance tests using the Chrome Dev Tool for /issues page filtered by a custom field:

Database	With visible condition	Without visible condition
----------	------------------------	---------------------------

postgres 14	~11s	~123ms
mysql 8 ¹	~5s	~5s

From my tests, the degradation appears only when postgres is used as database.

I've used the attached script to generate the fixtures on top of the test fixtures.

[1]: I expected to obtain same results on mysql 5.7.

#17 - 2022-07-12 06:52 - Marius BĂLTEANU

Holger, Felix, are you working on a fix for this? Or should I start doing it?

#18 - 2022-07-12 09:02 - Felix Schäfer

I have started working my way into the explains, unfortunately I think I will have to tinker around with it more as well as looking at the explains with the proposed patch to understand what difference it makes for the optimiser. I will only be back at my main computer with enough time to look into it by the weekend probably though.

#19 - 2022-07-17 10:42 - salman mp

Some of my issue queries takes thousands of seconds!

#20 - 2022-08-03 01:41 - Go MAEDA

- Has duplicate Defect #37539: Issue index filtering with custom field became unusable with Redmine 4.2.7 added

#21 - 2022-08-06 05:20 - ashraf alzyoud

Any update?!

#22 - 2022-08-11 00:35 - Mischa The Evil

- Has duplicate Patch #37565: Performance problem when filtering issues by custom-field value added

#23 - 2022-08-13 11:25 - Felix Schäfer

I am sorry for the delay, I was ill with COVID-19.

The patch proposed in [#37268#note-8](#) sadly does not work in MySQL.

Query::StatementInvalid: Mysql2::Error: Unknown column 'issues.id' in 'on clause'

The following added line causes this error " AND #{queried_table_name}.id=cc_sub.id ". In the case of issues this line leads to issues.id=cc_sub.id where cc_sub is also issues. I think the result of this query is not correct.

#24 - 2022-08-13 11:33 - Felix Schäfer

From what I gathered the additional visible condition trips the query optimiser for PostgreSQL and leads to a much more costly merge mechanism.

I am not sure how to alleviate this issue for PostgreSQL. As MySQL does not have this problem I fear this is a PostgreSQL-specific problem that would require PostgreSQL knowledge to solve. I will however continue researching this issue and explore the other proposed options such as [#37565](#).

#25 - 2022-08-15 14:25 - Felix Schäfer

Additionally joining on projects for Query#sql_for_custom_field will solve the performance issue. This is only required for IssueCustomField. The following patch implements this solution:

```
diff --git a/app/models/query.rb b/app/models/query.rb
index 1a614f1753..db977859f8 100644
--- a/app/models/query.rb
+++ b/app/models/query.rb
@@ -1159,11 +1159,15 @@ class Query < ActiveRecord::Base
   if /[<>]/.match?(operator)
     where = "(#{where}) AND #{db_table}.#{db_field} <> '"
   end
+
+  additional_joins = customized_class == Issue ? "LEFT OUTER JOIN #{Project.table_name} ON #{customized_class.table_name}.project_id = #{Project.table_name}.id" : ""
+
+  "#{queried_table_name}.#{customized_key} #{not_in} IN (" \
+    "SELECT #{customized_class.table_name}.id FROM #{customized_class.table_name}" \
```

```

" LEFT OUTER JOIN #{db_table} ON #{db_table}.customized_type='#{customized_class}' " \
" AND #{db_table}.customized_id=#{customized_class.table_name}.id" \
" AND #{db_table}.custom_field_id=#{custom_field_id}" \
+ " #{additional_joins}" \
" WHERE (#{where}) AND (#{filter[:field].visibility_by_project_condition}))"
end

```

This should probably be added through a `IssueCustomField#visibility_by_project_condition_joins` method.

#26 - 2022-08-26 04:12 - Ko Nagase

I haven't checked the details, but I noticed that Ruby bug tracker site (<https://bugs.ruby-lang.org/>) seems to apply their own patch for PostgreSQL DB, so just for your information.

- Twitter post: <https://twitter.com/nalsh/status/1561995447748947968>
- GitHub commit: <https://github.com/ruby/b.r-l.o/commit/91e830ef03d9290465d40eb2437308f9b33dd1cf>

```

diff --git a/app/models/query.rb b/app/models/query.rb
index 1a614f1753..e098a7eb46 100644
--- a/app/models/query.rb
+++ b/app/models/query.rb
@@ -1159,12 +1159,13 @@ def sql_for_custom_field(field, operator, value, custom_field_id)
  if /[<>]/.match?(operator)
    where = "(#{where}) AND #{db_table}.#{db_field} <> '"
  end
- "#{queried_table_name}.#{customized_key} #{not_in} IN (" \
- "SELECT #{customized_class.table_name}.id FROM #{customized_class.table_name}" \
+ "#{not_in} EXISTS (" \
+ "SELECT * FROM #{customized_class.table_name} i" \
+ " LEFT OUTER JOIN #{db_table} ON #{db_table}.customized_type='#{customized_class}' " \
+ " AND #{db_table}.customized_id=#{customized_class.table_name}.id" \
+ " AND #{db_table}.custom_field_id=#{custom_field_id}" \
- " WHERE (#{where}) AND (#{filter[:field].visibility_by_project_condition}))"
+ " WHERE #{queried_table_name}.#{customized_key} = i.id AND " \
+ " (#{where}) AND (#{filter[:field].visibility_by_project_condition}))"
end

def sql_for_chained_custom_field(field, operator, value, custom_field_id, chained_custom_field_id)

```

#27 - 2022-09-13 23:08 - Marius BĂLTEANU

- File 37268.patch added

Ko Nagase wrote:

I haven't checked the details, but I noticed that Ruby bug tracker site (<https://bugs.ruby-lang.org/>) seems to apply their own patch for PostgreSQL DB, so just for your information.

- Twitter post: <https://twitter.com/nalsh/status/1561995447748947968>
 - GitHub commit: <https://github.com/ruby/b.r-l.o/commit/91e830ef03d9290465d40eb2437308f9b33dd1cf>
- [...]

Thanks for pointing this out.

I've fixed an issue on that patch and the results look great, all tests pass and the performance issue is fixed as well. I'm going to commit this patch in the following days.

#28 - 2022-09-23 07:53 - Marius BĂLTEANU

- Status changed from Confirmed to Resolved
- Assignee set to Marius BĂLTEANU
- Resolution set to Fixed

Fix committed, sorry for the delay!

#29 - 2022-09-26 23:47 - Marius BĂLTEANU

- Status changed from Resolved to Closed

Fix merged to stable branches and the new releases will be published this weekend.

Files

custom_field_visibility_fix.patch	5.69 KB	2022-06-22	Alexander Meindl
custom_field_visibility_fix_v2.patch	960 Bytes	2022-06-22	Alexander Meindl
custom_field_visibility_fix_v3.patch	954 Bytes	2022-06-22	Alexander Meindl
custom_field_visibility_fix_v4.patch	921 Bytes	2022-06-22	Alexander Meindl
sql_with_visible_condition.sql	3.2 KB	2022-06-27	Marius BĂLTEANU
explain_with_visible_condition.csv	20.5 KB	2022-06-27	Marius BĂLTEANU
sql_without_visible_condition.sql	2.89 KB	2022-06-27	Marius BĂLTEANU
explain_without_visible_condition.csv	16.9 KB	2022-06-27	Marius BĂLTEANU
performace_fixtures.rb	423 Bytes	2022-06-27	Marius BĂLTEANU
37268.patch	1.24 KB	2022-09-13	Marius BĂLTEANU