redmineorg-copy202205 - Vote #76056

Ability to limit search to last week/month/year

2022/05/09 17:33 - 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:	Search engine_16	**予定工数:**	0.00時間
対象バージョン:		**作業時間:**	0.00時間
Redmineorg_URL:	https://www.redmine.org/issues/19108	**status_id:**	1
category_id:	16	**tracker_id:**	3
version_id:	0	**plus1:**	0
issue_org_id:	19108	**affected_version:**	
author_id:	26986	**closed_on:**	
assigned_to_id:	0	**affected_version_id:**	
comments:	5		

説明

Problem

Redmine search doesn't support or use full text searching features of mysql, postgres, lucene or elasticsearch, etc. which means it unfortunately doesn't scale well. The @like %keyword%@ query basically does a table scan which can easily take 10+ seconds on a medium database with decent hardware (Amazon EC c3.large - high CPU instance).

We tried the "elastic search plugin":https://github.com/Undev/redmine_elasticsearch (it's still in early beta) for Redmine but it gave unreliable results and turned out to degrade our user confidence in the search.

Solution

After looking at what other ticketing/issue/case management solutions do, I see many of them include a "past week/month/year" in their search and it usually has a default value of past year or month.

This patch adds a dropdown to select last week/month/year/all time on the search screen and defaults to last year as shown in attached screen shot. I was able to use existing translations except for the newly added "label_last_year". I followed the conventions of the rest of your code base as far as possible. The patch was generated on your 2.6 stable branch.

journals

I've updated the patch slightly, please remove my previous patch. Any comments or suggestions are welcome.

After some refactoring, search requests will be much faster with Redmine 3.0. Here is an example for searching a keyword in the issues of the redmine.org database (14k issues and 45k comments) on my dev machine with PostgreSQL. The search request is:

/search?q=foo&all_words=1&issues=1

With Redmine 2.6:

Completed 200 OK in 2402.4ms (Views: 15.6ms ¦ ActiveRecord: 2371.2ms)

With current trunk (Redmine 3.0):

Completed 200 OK in 905ms (Views: 31.2ms ¦ ActiveRecord: 842.4ms)

The most important thing is that the new queries will now take advantage of indexes if they exist (wasn't the case before 3.0). Here is the indexes for issues and journals tables:

```
CREATE   INDEX   issues_subject_description_idx
    ON   issues
```

```
    USING gin
    (subject COLLATE pg_catalog."default" gin_trgm_ops, description COLLATE pg_catalog."default" gin_trgm_ops);

CREATE INDEX journals_notes_idx
    ON journals
    USING gin
    (notes COLLATE pg_catalog."default" gin_trgm_ops);
```

After creating these indexes, we get pretty decent response times:

Completed 200 OK in 89ms (Views: 20.0ms ¦ ActiveRecord: 34.0ms)

Thank you for the feedback, that is good news. The change proposed by this patch can still be useful, whenever you search for something and it shows there are 100+ results, it would have been better if it was limited to the last x amount of time as it is sorted by date and you probably won't paginate 10+ pages, if you need older results you just set it to "All time", it makes the "next" button faster. We could also make the default option "All time" and make it configurable. Additionally if the search always searches everything, then it will only get slower as you build a bigger database.

As a side note: would you advise moving from mysql to postgres in preparation for Redmine 3?

Pierre Pretorius wrote:

> if you need older results you just set it to "All time", it makes the "next" button faster

I forgot to say that result ids are now cached, making pagination (next/previous links) really, really fast in Redmine in 3.0.

> As a side note: would you advise moving from mysql to postgres in preparation for Redmine 3?

The test suite runs with both as you can see "here": http://www.redmine.org/builds/index.html but all the development is done with PostgreSQL. I know PostgreSQL much better and it's my favorite, so I'd say yes. But I didn't try MySQL indexes to improve search request response time.

履歴

#1 - 2022/05/10 17:09 - Admin Redmine

- カテゴリ を Search engine_16 にセット