Foreman - Bug #688

Reports tab unusable with large number of reports

02/23/2011 03:04 PM - Kal McFate

Status:	Closed		
Priority:	Normal		
Assignee:			
Category:			
Target version:			
Difficulty:		**Fixed in Releases:**	
Triaged:		**Found in Releases:**	
Bugzilla link:		**Red Hat JIRA:**	
Pull request:			

Description

On the reports tab the table query takes a very long time to sort (approx 3 minutes for me)

SELECT * FROM `reports` WHERE (status != 0) ORDER BY reports.created_at DESC LIMIT 0, 20

Should have a composite index on status and created_at or similar.

explain results:

```
+----+-------------+---------+-------+-------------------------+-------------------------+--------
-+------+--------+-----------------------------+
| id | select_type | table   | type  | possible_keys           | key                     | key_len
 | ref  | rows   | Extra                       |
+----+-------------+---------+-------+-------------------------+-------------------------+--------
-+------+--------+-----------------------------+
|  1 | SIMPLE      | reports | range | index_reports_on_status | index_reports_on_status | 5
 | NULL | 516521 | Using where; Using filesort |
+----+-------------+---------+-------+-------------------------+-------------------------+--------
-+------+--------+-----------------------------+
```

History

#1 - 02/23/2011 03:23 PM - Kal McFate

https://foreman.wc1.dfw1.stabletransit.com/reports is much faster.

It uses 'describe SELECT * FROM `reports` ORDER BY reports.created_at DESC LIMIT 0, 20;'

Which i would actually expect to be slower, as it has (in my case) 10x's as many rows to sort.

#2 - 02/23/2011 04:57 PM - Ohad Levy

- Status changed from New to Feedback

can you have a look at this thread: https://groups.google.com/group/foreman-users/browse_thread/thread/2729d0b6583037eb?hl=en ?

i think that the suggested patch (that adds the indexes) should solve your issue.

if it does, let me know and I'll merge it in.

thanks

#3 - 02/24/2011 04:57 PM - Kal McFate

After investigating this further:

This is all improved by tuning the innodb_buffer_pool_size, but I do have notes for additional improvement.

Adding the index reports(status, created_at) was pointless as the optimizer never used it.

Adding the index reports(created_at) and altering it to:
SELECT id,status FROM `reports` HAVING (status != 0) ORDER BY reports.created_at DESC LIMIT 0, 20;
avoids the filesort and is magnitudes faster. However i am unsure of portability.

WHERE operates before the sort forcing a filesort and HAVING is just a filter afterwards allowing it to use the reports(create_at) index.

Plus the reports(created_at) index would also used by:
SELECT id,status FROM `reports` ORDER BY reports.created_at DESC LIMIT 0, 20;

SELECT count(*) AS count_all FROM `reports`;
Is just slow with innodb, however very dependent on your innodb buffer pool size.

Just ensuring my buffer_pool was large enough to hold at least the entire reports table helped tremendously.

#4 - 03/15/2011 03:48 AM - Ohad Levy

- Status changed from Feedback to Closed