Redmine - Patch #6431

Allow parent task field to be used in filters

2010-09-18 03:20 - Stuart Cianos

Status:	Closed	**Start date:**	2010-09-18
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:	Issues	**Estimated time:**	0.00 hour
Target version:			

Description

This patch allows the parent task field to be used in filters.

The patch also corrects a potential bug in the SQL code generation for grouping of columns with both a specified sort order and a default sort order, based on Redmine's current implementation of subtasking as a sort order.

This patch resolves #6397.

Related issues:

Related to Redmine - Feature #6397: Add Parent Task field as a sort criterium...	**Closed**	**2010-09-15**
Related to Redmine - Feature #6118: Filter by parent task or subtasks	**Closed**	**2010-08-12**

History

#1 - 2010-09-18 03:21 - Stuart Cianos

One additional note: This also implements the ability to specify a custom caption when defining filter fields.

#2 - 2010-09-28 23:22 - Gray Wizard

I found a small "feature" in the submitted patch.

"root_id" => { :type => :integer, :order => 16, :caption => "field_parent_issue" },

should be:

"parent_id" => { :type => :integer, :order => 16, :caption => "field_parent_issue" },

#3 - 2010-09-29 04:46 - Stuart Cianos

Reason for the use of root_id was based on the previous specification in current production source. Reference is:

Line 124 (app/models/query.rb): QueryColumn.new(:parent, :sortable => ["#{Issue.table_name}.root_id", "#{Issue.table_name}.lft ASC"], :default_order => 'desc', :caption => :field_parent_issue)

Is there a reason parent_id is reccomended as apposed to the above spec from the original querycolumn definition?

Gray Wizard wrote:

> I found a small "feature" in the submitted patch.
>
> "root_id" => { :type => :integer, :order => 16, :caption => "field_parent_issue" },
>
> should be:
>
> "parent_id" => { :type => :integer, :order => 16, :caption => "field_parent_issue" },

#4 - 2010-09-30 03:01 - Gray Wizard

Stuart Cianos wrote:

> Reason for the use of root_id was based on the previous specification in current production source. Reference is:
>
> Line 124 (app/models/query.rb): QueryColumn.new(:parent, :sortable => ["#{Issue.table_name}.root_id", "#{Issue.table_name}.lft ASC"], :default_order => 'desc', :caption => :field_parent_issue)
>
> Is there a reason parent_id is reccomended as apposed to the above spec from the original querycolumn definition?

> Gray Wizard wrote:
>
> > I found a small "feature" in the submitted patch.
> >
> > "root_id" => { :type => :integer, :order => 16, :caption => "field_parent_issue" },
> >
> > should be:
> >
> > "parent_id" => { :type => :integer, :order => 16, :caption => "field_parent_issue" },

I must confess I don't know the DB structure of Redmine very much at all but the "root_id" is not linked (in our instance) to the parent issue at all. As a matter of fact I am not sure what the root_id refers to as the values in that column exceed the id values I have in my projects, versions, and ticket tables. I assume that they must relate somehow to the parent issue but it is not a direct link, at least not in the latest "Bitnami Redmine Stack for Windows"

#5 - 2010-09-30 23:48 - Stuart Cianos

Gray Wizard wrote:

> I must confess I don't know the DB structure of Redmine very much at all but the "root_id" is not linked (in our instance) to the parent issue at all. As a matter of fact I am not sure what the root_id refers to as the values in that column exceed the id values I have in my projects, versions, and ticket tables. I assume that they must relate somehow to the parent issue but it is not a direct link, at least not in the latest "Bitnami Redmine Stack for Windows"

On further review, I think we're both wrong... The root_id points to the issue id of the top branch within the "tree of subtasks". They are storing the tree in the database (i.e. the lft, rgt columns and henceforth their use of those columns with the root_id for ordering, as described above). This makes for an interesting dilemna in some respects, since this means that using the parent_id would require a user to have an "or" condition to include children more than one branch away... I'll have to play around with the code to see if there is a better way (anyone else with ideas, please chime in!)

Regarding your strange ID#'s:

You noted the root_id is pointing to nonexistent id's in your db. What version of Redmine is included in the Bitnami stack (from Administration->Information)? In 1.0.1 (current production in my organization) and SVN trunk I show the root_id linked to either the root parent or, if parent_id is null, itself. Example:

'id', 'parent_id', 'root_id'
'414', '478', '478'
'415', NULL, '415'
'416', '478', '478'
'417', '437', '437'
'418', '437', '437'
'419', '437', '437'
'420', '437', '437'
'421', '437', '437'
'422', '436', '436'

In this case, row 415 has no parent_id, so root_id links to itself. For 416-421, 437 the parent_id and root_id is set accordingly (which is correct).

#6 - 2010-10-01 00:34 - Gray Wizard

Stuart Cianos wrote:

> Gray Wizard wrote:
>
> > I must confess I don't know the DB structure of Redmine very much at all but the "root_id" is not linked (in our instance) to the parent issue at all. As a matter of fact I am not sure what the root_id refers to as the values in that column exceed the id values I have in my projects, versions, and ticket tables. I assume that they must relate somehow to the parent issue but it is not a direct link, at least not in the latest "Bitnami Redmine Stack for Windows"
>
> On further review, I think we're both wrong... The root_id points to the issue id of the top branch within the "tree of subtasks". They are storing the tree in the database (i.e. the lft, rgt columns and henceforth their use of those columns with the root_id for ordering, as described above). This makes for an interesting dilemna in some respects, since this means that using the parent_id would require a user to have an "or" condition to include children more than one branch away... I'll have to play around with the code to see if there is a better way (anyone else with ideas, please chime in!)
>
> Regarding your strange ID#'s:
>
> You noted the root_id is pointing to nonexistent id's in your db. What version of Redmine is included in the Bitnami stack (from Administration->Information)? In 1.0.1 (current production in my organization) and SVN trunk I show the root_id linked to either the root parent or, if parent_id is null, itself. Example:

'id', 'parent_id', 'root_id'
'414', '478', '478'
'415', NULL, '415'
'416', '478', '478'
'417', '437', '437'
'418', '437', '437'
'419', '437', '437'
'420', '437', '437'
'421', '437', '437'
'422', '436', '436'

In this case, row 415 has no parent_id, so root_id links to itself. For 416-421, 437 the parent_id and root_id is set accordingly (which is correct).

Upon further review you nailed it on the head, the root_id was the id if there was no parent_id, I had not double checked my numbers well enough (just re-ran the query). In our case we only have one level of sub-task by policy so it is not a huge problem for our implementation, but it would indeed break for anything deeply nested. Is there a problem with with changing the query to be "parent_id=X OR root_id=X and ID != X" for "child tasks"?

#7 - 2010-10-01 00:35 - Gray Wizard

Forgot to mention we are running 1.0.1 (stable) on mySQL

#8 - 2010-10-01 01:22 - Stuart Cianos

Gray Wizard wrote:

> Upon further review you nailed it on the head, the root_id was the id if there was no parent_id, I had not double checked my numbers well enough (just re-ran the query). In our case we only have one level of sub-task by policy so it is not a huge problem for our implementation, but it would indeed break for anything deeply nested. Is there a problem with with changing the query to be "parent_id=X OR root_id=X and ID != X" for "child tasks"?

I'll have to check it out to see if the current implementation can handle multiple field references... if not, will be a weekend project since the organization I work for needs this feature :)

- Stu

#9 - 2011-04-18 22:51 - Daniel Thornton

I am unable to get this patch working in version 1.1.1.stable or in svn trunk.

#10 - 2011-04-18 22:56 - Stuart Cianos

Thanks for the heads up... We're running 1.1.2 right now without issue.

What errors are you getting/did the patch not apply cleanly?

#11 - 2012-03-05 22:12 - William Lafleur

Would it be a good idea to apply this patch to version 1.3.0?

#12 - 2013-01-28 11:21 - shravan kumar

Getting internal with this patch.

I have redmine 1.4.4

When i make change in /app/views/quires/_filters.html.erb

<%= select_tag 'add_filter_select', options_for_select([["",""] + query.available_filters.sort{|a,b| a[1][:order]<=>b[1][:order]}.collect{|field| [field[1][:name] || (field[1][:caption] == nil ? l(("field_"+field[0].to_s.gsub(/_id$/, "")).to_sym) : l(field[1][:caption].to_sym)), field[0]] unless query.has_filter?(field[0])}.compact),

I get Internal error.

Can please hepl to resolve this issue

#13 - 2014-03-06 11:00 - Luis Serrano Aranda

+1

#14 - 2018-02-15 23:02 - @ go2null

As #6118 is implemented, shouldn't this be closed?

#15 - 2018-02-16 02:15 - Mischa The Evil

- Status changed from New to Closed

@ go2null wrote:

> As #6118 is implemented, shouldn't this be closed?

You are right. Thanks for your comment. I'll close this issue.

Files

scvmc_redmine_6397_parent_task.diff	3.6 KB	2010-09-18	Stuart Cianos