# Freelancing Gods 2009

12 Jun 2008

## A Concise Guide to Using Thinking Sphinx

Tags:

- ruby
- rails
- plugins
- search
- sphinx

Okay, it's well past time for the companion piece to my Sphinx Primer – let's go through the basic process of using Thinking Sphinx with Rails.

Just to recap: Sphinx is a search engine that indexes data, and then you can query it with search terms to find out which documents are relevant. Why do you want to use it with Rails? Because it saves having to write messy SQL, and it's so damn fast.

(If you're getting a feeling of deja-vu, then it's probably because you've read an old post on this blog that dealt with an old version of Thinking Sphinx. I've had a few requests for an updated article, so this is it.)

### Installation

So: first step is to install Sphinx. This may be tricky on some systems – but I've never had a problem with it with Mac OS X or Ubuntu. My process is thus:

```
curl -O http://sphinxsearch.com/downloads/sphinx-0.9.8-rc2.tar.gz
tar zxvf sphinx-0.9.8-rc2.tar.gz
cd sphinx-0.9.8-rc2
./configure
make
sudo make install
```

If you're using Windows, you can just grab the binaries.

Once that's taken care of, you then want to take your Rails app, and install the plugin. If you're running edge or 2.1, this is a piece of cake:

```
script/plugin install git://github.com/freelancing-god/thinking-sphinx.git
```

Otherwise, you've got a couple of options. The first is, if you have git installed, just clone to your vendor/plugins directory:

```
git clone git://github.com/freelancing-god/thinking-sphinx.git
  vendor/plugins/thinking-sphinx
```

If you're not yet using git, then the easiest way is to download the tar file of the code. Try the following:

```
curl -L http://github.com/freelancing-god/thinking-sphinx/tarball/master
  -o thinking-sphinx.tar.gz
tar -xvf thinking-sphinx.tar.gz -C vendor/plugins
mv vendor/plugins/freelancing-god-thinking-sphinx* vendor/plugins/thinking-sphinx
```

Oh, and it's worth noting: if you're not using MySQL or PostgreSQL, then you're out of luck – Sphinx doesn't talk to any other relational databases.

## Configuration

Next step: let's get a model or two indexed. It might be worth refreshing your memory on what fields and attributes are for – can I recommend my Sphinx article (because I'm not at all biased)?

Ok, now let's work with a simple Person model, and add a few fields:

```
class Person < ActiveRecord::Base
  define_index do
    indexes [first_name, last_name], :as => :name
    indexes location
  end
end
```

Nothing too scary – we've added two fields. The first is the first and last names of a person combined to one field with the alias 'name'. The second is simply location.

Adding attributes is just as easy:

```
define_index do
  # ...

  has birthday
end
```

This attribute is the datetime value birthday (so you can now sort and filter your results by birthdays).

## Managing Sphinx

We've set up a basic index – now what? We tell Sphinx to index the data, and then we can start searching. Rake is our friend for this:

```
rake thinking_sphinx:index
rake thinking_sphinx:start
```

## Searching

Now for the fun stuff:

```
Person.search "Melbourne"
```

Or with some sorting:

```
Person.search "Melbourne", :order => :birthday
```

Or just people born within a 10 year window:

```
Person.search "Melbourne", :with => {:birthday => 25.years.ago..15.years.ago}
```

If you want to keep certain search terms to specific fields, use `:conditions`:

```
Person.search :conditions => {:location => "Melbourne"}
```

Just remember: `:conditions` is for fields, `:with` is for attributes (and `:without` for exclusive attribute filters).

## Change

Your data changes – but unfortunately, Sphinx doesn't update your indexes to match automatically. So there's two things you need to do. Firstly, run `rake thinking_sphinx:index` regularly (using cron or something similar). 'Regularly' can mean whatever time frame you want – weekly, daily, hourly.

The second step is optional, but it's needed to have your indexes always up to date. First, add a boolean column to your model, named 'delta', and have it default to false. Then, tell your index to use that delta field to keep track of changes:

```
define_index do
  # ...

  set_property :delta => true
end
```

Then you need to tell Sphinx about the updates:

```
rake thinking_sphinx:stop
rake thinking_sphinx:index
rake thinking_sphinx:start
```

Once that's done, a delta index will be created – which holds any recent changes (since the last proper indexing), and gets re-indexed whenever a model is edited or created. This doesn't mean you can stop the regular indexing, as that's needed to keep delta indexes as small (and fast) as possible.

## String Sorting

If you remember the details about fields and attributes, you'll know that you can't sort by fields. Which is a pain, but there's ways around this – and it's kept pretty damn easy in Thinking Sphinx. Let's say we wanted to make our `name` field sortable:

```
define_index do
  indexes [first_name, last_name], :as => :name, :sortable => true

  # ...
end
```

Re-index and restart Sphinx, and sorting by name will work.

How is this done? Thinking Sphinx creates an attribute under the hood, called name_sort, and uses that, as Sphinx is quite fine with sorting by strings if they're converted to ordinal values (which happens automatically when they're attributes).

## Pagination

Sphinx paginates automatically – in fact, there's no way of turning that off. But that's okay… as long as you can use your will_paginate helper, right? Never fear, Thinking Sphinx plays nicely with will_paginate, so your views don't need to change at all:

```
<%= will_paginate @search_results %>
```

## Associations

Sometimes you'll want data in your fields (or attributes) from associations. This is a piece of cake:

```
define_index do
  indexes photos.caption, :as => :captions
  indexes friends.photos.caption, :as => :friends_photos

  # ...
end
```

Polymorphic associations are fine as well – but keep in mind, the more complex your index fields and attributes, the slower it will be for Sphinx to index (and you'll definitely need some database indexes on foreign key columns to help it stay as speedy as possible).

### Gotchas

In case things aren't working, here's some things to keep in mind:

- Added an attribute, but can't sort or filter by it? Have you reindexed and restarted Sphinx? It doesn't automatically pick up these changes.
- Sorting not working? If you're specifying the attribute to sort by as a string, you'll need to include the direction to sort by, just like with SQL: "birthday ASC".
- Using `name` or `id` columns in your fields or attributes? Make sure you specify them using symbols, as they're core class methods in Ruby.

```
define_index do
  indexes :name

  # ...

  has photos(:id), :as => :photo_ids
end
```

### And Next?

I realise this article is pretty light on details – but if you want more information, the first stop should be the extended usage page on the Thinking Sphinx site, quickly followed by the documentation. There's also an email list to ask questions on.

← View Previous ArticleThe End of Charity
View Next Article →Rails Camp UK

## Comments

30 responses to this article

13 Jun 2008
*Andrew Zielinski said:*

> Nice Introduction to Thinking Sphinx. Thanks!

13 Jun 2008
*Benny said:*

> Can you make a quick rundown why Thinking Sphinx is to be preferred other other Sphinx helpers like Ultrasphinx ?
>
> Which one has covered most of Sphinx's features, etc.

13 Jun 2008
*Marston A said:*

> Great article. I'm also curious if you've compared this to UltraSphinx and how they stack up?

13 Jun 2008
*pat said:*

> Andrew: no problems.
>
> Benny and Marston: Ultrasphinx is more complex – which is useful if you want to tweak Sphinx quite a bit. Thinking Sphinx's goal is convention over configuration – although you can set most things to whatever you wish, it makes assumptions on file locations and such, so you can just start using Sphinx as quickly as possible.
>
> Thinking Sphinx also allows you to set all the settings you wish through Ruby and YAML - so you don't need to learn about the configuration syntax of Sphinx if you don't wish to.