参考：<http://www.sphinxsearch.org/archives/80>

Sphinx中文入门指南——新手可先看此文

2009年02月9日 — admin

本文主要介绍Sphinx的入门使用，新手观看，老鸟指正！

\* 1、简介

\* 1.1.Sphinx是什么  
\* 1.2.Sphinx的特性  
\* 1.3.Sphinx中文分词

\* 2、安装配置实例

\* 2.1 在GNU/Linux/unix系统上安装  
o 2.1.1 sphinx安装  
o 2.1.2.sfc安装(见另文)  
o 2.1.3.coreseek安装（见另文）  
\* 2.2 在windows下安装

\* 3、 配置实例  
\* 4、应用

\* 4.1 在CLI上测试  
\* 4.2 使用API调用

\* 5、附录

1.Sphinx简介  
1.1.Sphinx是什么

Sphinx是由俄罗斯人Andrew Aksyonoff开发的一个全文检索引擎。意图为其他应用提供高速、低空间占用、高结果 相关度的全文搜索功能。Sphinx可以非常容易的与SQL数据库和脚本语言集成。当前系统内置MySQL和PostgreSQL 数据库数据源的支持，也支持从标准输入读取特定格式 的XML数据。通过修改源代码，用户可以自行增加新的数据源（例如：其他类型的DBMS 的原生支持）  
1.2.Sphinx的特性

\* 高速的建立索引(在当代CPU上，峰值性能可达到10 MB/秒);  
\* 高性能的搜索(在2 – 4GB 的文本数据上，平均每次检索响应时间小于0.1秒);  
\* 可处理海量数据(目前已知可以处理超过100 GB的文本数据, 在单一CPU的系统上可 处理100 M 文档);  
\* 提供了优秀的相关度算法，基于短语相似度和统计（BM25）的复合Ranking方法;  
\* 支持分布式搜索;  
\* 支持短语搜索  
\* 提供文档摘要生成  
\* 可作为MySQL的存储引擎提供搜索服务;  
\* 支持布尔、短语、词语相似度等多种检索模式;  
\* 文档支持多个全文检索字段(最大不超过32个);  
\* 文档支持多个额外的属性信息(例如：分组信息，时间戳等);  
\* 支持断词;

1.3.Sphinx中文分词

中文的全文检索和英文等latin系列不一样，后者是根据空格等特殊字符来断词，而中文是根据语义来分词。目前大多数数据库尚未支持中文全文检索，如Mysql。故，国内出现了一些Mysql的中文全文检索的插件，做的比较好的有hightman的中文分词。Sphinx如果需要对中文进行全文检索，也得需要一些插件来补充。其中我知道的插件有 coreseek 和 sfc 。

\* Coreseek是现在用的最多的sphinx中文全文检索，它提供了为Sphinx设计的中文分词包LibMMSeg 。并提供了多个系统的二进制发行版，其中有rpm,deb及windows下的二进制包。另外，coreseek也为sphinx贡献了以下事项：  
o GBK编码的数据源支持  
o 采用Chih-Hao Tsai MMSEG算法的中文分词器  
o 中文使用手册（这份中文手册对国内使用sphinx新手——特别是英语不太好的人来说，提供了极大的便利）  
\* sfc（sphinx-for-chinese）是由网友happy兄提供的另外一个中文分词插件。其中文词典采用的是xdict。据其介绍,经过测试，目前版本在索引速度上(Linux 测试平台)基本上能够达到索引UTF-8英文的一半，即官方宣称速度的一半。（时间主要是消耗在分词上）。 现提供了与sphinx最新版(sphinx 0.9.10)同步的sphinx-for-chinese-0.9.10-dev-r2006.tar.gz 。此版本增加了sql\_attr\_string，经过本人的测试。其安装和配置都非常方便。happy兄在分词方面还有另外一个贡献——php-mmseg，这是php对中文分词的一个扩展库。

在此，对以上二位作者谨以最大的敬意

\* 此外，如果你对中文分词不感兴趣。或者说仅需要实现类似sql中like的功能，如： select \* from product where prodName like ‘%手机%’。sphinx也不会让你失望，这个或许就是官网对中文的简单实现——直接对字索引。并且搜索速度还不错^\_^ 。

本文会对以上三种中文应用进行测试，并以文档的方式记录下来，这也许正是本文档的重点。  
2.安装配置实例  
2.1在GNU/Linux/unix系统上安装

Sphinx在mysql上的应用有两种方式：  
①、采用API调用，如使用PHP、java等的API函数或方法查询。优点是可不必对mysql重新编译，服务端进程“低耦合”，且程序可灵活、方便的调用；  
缺点是如已有搜索程序的条件下，需修改部分程序。推荐程序员使用。  
②、使用插件方式（sphinxSE）把sphinx编译成一个mysql插件并使用特定的sql语句进行检索。其特点是，在sql端方便组合，且能直接返回数据给客户端  
不必二次查询（注）,在程序上仅需要修改对应的sql，但这对使用框架开发的程序很不方便，比如使用了ORM。另外还需要对mysql进行重新编译，且需要mysql-5.1以上版本  
支持插件存储。系统管理员可使用这种方式  
二次查询注：到现在发布版本为止——sphinx-0.9.9，sphinx在检索到结果后只能返回记录的ID，而非要查的sql数据，故需要重新根据这些ID再次从数据库中查询，  
正在开发的sphinx 0.9.10版本已可存储这些文本数据，作者曾试过，性能和存储上的效果都不佳，毕竟还没出正式版

本文采用的是第一种方式

在\*nix系统下安装，首先需要以下一些软件支持

软件环境：

\* 操作系统：Centos-5.2  
\* 数据库：mysql-5.0.77-3.el5 mysql-devel（如果要使用sphinxSE插件存储请使用mysql-5.1以上版本）  
\* 编译软件：gcc gcc-c++ autoconf automake  
\* Sphinx ：Sphinx-0.9.9 (最新稳定版 )

安装：

\* [root@localhost ~]# yum install -y mysql mysql-devel  
\* [root@localhost ~]# yum install -y automake autoconf  
\* [root@localhost ~]# cd /usr/local/src/  
\* [root@localhost src]# wget http://www.sphinxsearch.com/downloads/sphinx-0.9.9.tar.gz  
\* [root@localhost src]# tar zxvf sphinx-0.9.9.tar.gz  
\* [root@localhost local]# cd sphinx-0.9.9  
\* [root@localhost sphinx-0.9.9]# ./configure –prefix=/usr/local/sphinx #注意：这里sphinx已经默认支持了mysql  
\* [root@localhost sphinx-0.9.9]# make && make install # 其中的“警告”可以忽略

安装完毕后查看一下/usr/local/sphinx下是否有 三个目录 bin etc var，如有，则安装无误！  
2.1.2.sfc安装(点击进入)  
2.1.3.coreseek安装（点击进入）  
3.配置实例  
3.1、数据源。

这里我们采用 mysql的数据源。具体情况如下：

Mysql server：192.168.1.10

Mysql db :test

Mysql 表：test.sphinx\_article

mysql> desc sphinx\_article;  
+———–+———————+——+—–+———+—————-+  
| Field | Type | Null | Key | Default | Extra |  
+———–+———————+——+—–+———+—————-+  
| id | int(11) unsigned | NO | PRI | NULL | auto\_increment |  
| title | varchar(255) | NO | | | |  
| cat\_id | tinyint(3) unsigned | NO | MUL | | |  
| member\_id | int(11) unsigned | NO | MUL | | |  
| content | longtext | NO | | | |  
| created | int(11) | NO | MUL | | |  
+———–+———————+——+—–+———+—————-+  
6 rows in set (0.00 sec)  
3.2、配置文件

\* [root@localhost ~]#cd /usr/local/sphinx/etc #进入sphinx的配置文件目录  
\* [root@localhost etc]# cp sphinx.conf.dist sphinx.conf #新建Sphinx配置文件  
\* [root@localhost etc]# vim sphinx.conf #编辑sphinx.conf

具体实例配置文件：

##### 索引源 ###########  
source article\_src  
{  
type = mysql #####数据源类型  
sql\_host = 192.168.1.10 ######mysql主机  
sql\_user = root ########mysql用户名  
sql\_pass = pwd############mysql密码  
sql\_db = test #########mysql数据库名  
sql\_port= 3306 ###########mysql端口  
sql\_query\_pre = SET NAMES UTF8 ###mysql检索编码，特别要注意这点，很多人中文检索不到是数据库的编码是GBK或其他非UTF8  
sql\_query = SELECT id,title,cat\_id,member\_id,content,created FROM sphinx\_article ####### 获取数据的sql

#####以下是用来过滤或条件查询的属性############

sql\_attr\_uint = cat\_id ######## 无符号整数属性  
sql\_attr\_uint = member\_id  
sql\_attr\_timestamp = created ############ UNIX时间戳属性

sql\_query\_info = select \* from sphinx\_article where id=$id ######### 用于命令界面端(CLI)调用的测试

}

### 索引 ###

index article  
{  
source = article\_src ####声明索引源  
path = /usr/local/sphinx/var/data/article #######索引文件存放路径及索引的文件名  
docinfo = extern ##### 文档信息存储方式  
mlock = 0 ###缓存数据内存锁定  
morphology = none #### 形态学（对中文无效）  
min\_word\_len = 1 #### 索引的词最小长度  
charset\_type = utf-8 #####数据编码

##### 字符表，注意：如使用这种方式，则sphinx会对中文进行单字切分，  
##### 即进行字索引，若要使用中文分词，必须使用其他分词插件如 coreseek，sfc

charset\_table = U+FF10..U+FF19->0..9, 0..9, U+FF41..U+FF5A->a..z, U+FF21..U+FF3A->a..z,\  
A..Z->a..z, a..z, U+0149, U+017F, U+0138, U+00DF, U+00FF, U+00C0..U+00D6->U+00E0..U+00F6,\  
U+00E0..U+00F6, U+00D8..U+00DE->U+00F8..U+00FE, U+00F8..U+00FE, U+0100->U+0101, U+0101,\  
U+0102->U+0103, U+0103, U+0104->U+0105, U+0105, U+0106->U+0107, U+0107, U+0108->U+0109,\  
U+0109, U+010A->U+010B, U+010B, U+010C->U+010D, U+010D, U+010E->U+010F, U+010F,\  
U+0110->U+0111, U+0111, U+0112->U+0113, U+0113, U+0114->U+0115, U+0115, \  
U+0116->U+0117,U+0117, U+0118->U+0119, U+0119, U+011A->U+011B, U+011B, U+011C->U+011D,\  
U+011D,U+011E->U+011F, U+011F, U+0130->U+0131, U+0131, U+0132->U+0133, U+0133, \  
U+0134->U+0135,U+0135, U+0136->U+0137, U+0137, U+0139->U+013A, U+013A, U+013B->U+013C, \  
U+013C,U+013D->U+013E, U+013E, U+013F->U+0140, U+0140, U+0141->U+0142, U+0142, \  
U+0143->U+0144,U+0144, U+0145->U+0146, U+0146, U+0147->U+0148, U+0148, U+014A->U+014B, \  
U+014B,U+014C->U+014D, U+014D, U+014E->U+014F, U+014F, U+0150->U+0151, U+0151, \  
U+0152->U+0153,U+0153, U+0154->U+0155, U+0155, U+0156->U+0157, U+0157, U+0158->U+0159,\  
U+0159,U+015A->U+015B, U+015B, U+015C->U+015D, U+015D, U+015E->U+015F, U+015F, \  
U+0160->U+0161,U+0161, U+0162->U+0163, U+0163, U+0164->U+0165, U+0165, U+0166->U+0167, \  
U+0167,U+0168->U+0169, U+0169, U+016A->U+016B, U+016B, U+016C->U+016D, U+016D, \  
U+016E->U+016F,U+016F, U+0170->U+0171, U+0171, U+0172->U+0173, U+0173, U+0174->U+0175,\  
U+0175,U+0176->U+0177, U+0177, U+0178->U+00FF, U+00FF, U+0179->U+017A, U+017A, \  
U+017B->U+017C,U+017C, U+017D->U+017E, U+017E, U+0410..U+042F->U+0430..U+044F, \  
U+0430..U+044F,U+05D0..U+05EA, U+0531..U+0556->U+0561..U+0586, U+0561..U+0587, \  
U+0621..U+063A, U+01B9,U+01BF, U+0640..U+064A, U+0660..U+0669, U+066E, U+066F, \  
U+0671..U+06D3, U+06F0..U+06FF,U+0904..U+0939, U+0958..U+095F, U+0960..U+0963, \  
U+0966..U+096F, U+097B..U+097F,U+0985..U+09B9, U+09CE, U+09DC..U+09E3, U+09E6..U+09EF, \  
U+0A05..U+0A39, U+0A59..U+0A5E,U+0A66..U+0A6F, U+0A85..U+0AB9, U+0AE0..U+0AE3, \  
U+0AE6..U+0AEF, U+0B05..U+0B39,U+0B5C..U+0B61, U+0B66..U+0B6F, U+0B71, U+0B85..U+0BB9, \  
U+0BE6..U+0BF2, U+0C05..U+0C39,U+0C66..U+0C6F, U+0C85..U+0CB9, U+0CDE..U+0CE3, \  
U+0CE6..U+0CEF, U+0D05..U+0D39, U+0D60,U+0D61, U+0D66..U+0D6F, U+0D85..U+0DC6, \  
U+1900..U+1938, U+1946..U+194F, U+A800..U+A805,U+A807..U+A822, U+0386->U+03B1, \  
U+03AC->U+03B1, U+0388->U+03B5, U+03AD->U+03B5,U+0389->U+03B7, U+03AE->U+03B7, \  
U+038A->U+03B9, U+0390->U+03B9, U+03AA->U+03B9,U+03AF->U+03B9, U+03CA->U+03B9, \  
U+038C->U+03BF, U+03CC->U+03BF, U+038E->U+03C5,U+03AB->U+03C5, U+03B0->U+03C5, \  
U+03CB->U+03C5, U+03CD->U+03C5, U+038F->U+03C9,U+03CE->U+03C9, U+03C2->U+03C3, \  
U+0391..U+03A1->U+03B1..U+03C1,U+03A3..U+03A9->U+03C3..U+03C9, U+03B1..U+03C1, \  
U+03C3..U+03C9, U+0E01..U+0E2E,U+0E30..U+0E3A, U+0E40..U+0E45, U+0E47, U+0E50..U+0E59, \  
U+A000..U+A48F, U+4E00..U+9FBF,U+3400..U+4DBF, U+20000..U+2A6DF, U+F900..U+FAFF, \  
U+2F800..U+2FA1F, U+2E80..U+2EFF,U+2F00..U+2FDF, U+3100..U+312F, U+31A0..U+31BF, \  
U+3040..U+309F, U+30A0..U+30FF,U+31F0..U+31FF, U+AC00..U+D7AF, U+1100..U+11FF, \  
U+3130..U+318F, U+A000..U+A48F,U+A490..U+A4CF  
min\_prefix\_len = 0 #最小前缀  
min\_infix\_len = 1 #最小中缀  
ngram\_len = 1 # 对于非字母型数据的长度切割

#加上这个选项，则会对每个中文，英文字词进行分割，速度会慢  
#ngram\_chars = U+4E00..U+9FBF, U+3400..U+4DBF, U+20000..U+2A6DF, U+F900..U+FAFF,\  
#U+2F800..U+2FA1F, U+2E80..U+2EFF, U+2F00..U+2FDF, U+3100..U+312F, U+31A0..U+31BF,\  
#U+3040..U+309F, U+30A0..U+30FF, U+31F0..U+31FF, U+AC00..U+D7AF, U+1100..U+11FF,\  
#U+3130..U+318F, U+A000..U+A48F, U+A490..U+A4CF

}

######### 索引器配置 #####  
indexer  
{  
mem\_limit = 256M ####### 内存限制  
}

############ sphinx 服务进程 ########  
searchd  
{  
#listen = 9312 ### 监听端口，在此版本开始，官方已在IANA获得正式授权的9312端口，以前版本默认的是3312

log = /usr/local/sphinx/var/log/searchd.log #### 服务进程日志 ，一旦sphinx出现异常，基本上可以从这里查询有效信息，轮换（rotate）出的问题一般可在此寻到答案  
query\_log = /usr/local/sphinx/var/log/query.log ### 客户端查询日志，笔者注：若欲对一些关键词进行统计，可以分析此日志文件  
read\_timeout = 5 ## 请求超时  
max\_children = 30 ### 同时可执行的最大searchd 进程数  
pid\_file = /usr/local/sphinx/var/log/searchd.pid #######进程ID文件  
max\_matches = 1000 ### 查询结果的最大返回数  
seamless\_rotate = 1 ### 是否支持无缝切换，做增量索引时通常需要  
}  
3.3、建立索引文件

[root@localhost sphinx]# bin/indexer -c etc/sphinx.conf article ### 建立索引文件的命令  
Sphinx 0.9.9-release (r2117)  
Copyright (c) 2001-2009, Andrew Aksyonoff

using config file ‘etc/sphinx.conf’…  
indexing index ‘article’…  
collected 1000 docs, 0.2 MB  
sorted 0.4 Mhits, 99.6% done  
total 1000 docs, 210559 bytes  
total 3.585 sec, 58723 bytes/sec, 278.89 docs/sec  
total 2 reads, 0.031 sec, 1428.8 kb/call avg, 15.6 msec/call avg  
total 11 writes, 0.032 sec, 671.6 kb/call avg, 2.9 msec/call avg  
[root@localhost sphinx]#  
出现以上代表已经索引成功，若不成功的话请根据提示的错误修改配置文件，或到这里提问，我看到后会尽快解决  
4.应用  
4.1 在CLI上测试

在上一步中，我们建立了索引，现在我们对刚建立的索引进行测试。测试有两种方式：CLI端和API调用

在CLI端上命令测试是使用sphinx自带的搜索命令：search

###### 在article索引上检索 “北京”关键词 ########  
[root@localhost sphinx]# bin/search -c etc/sphinx.conf 北京  
Sphinx 0.9.9-release (r2117)  
Copyright (c) 2001-2009, Andrew Aksyonoff

using config file ‘etc/sphinx.conf’…  
index ‘article’: query ‘北京 ‘: returned 995 matches of 995 total in 0.008 sec

displaying matches:  
1. document=76, weight=2, cat\_id=1, member\_id=2, created=Sat Jan 23 19:05:09 2010  
id=76  
title=??????????  
cat\_id=1  
member\_id=2  
content=????????????????????????????????  
created=1264244709  
2. document=85, weight=2, cat\_id=1, member\_id=2, created=Sat Jan 23 19:05:09 2010  
id=85  
title=????????????  
cat\_id=1  
member\_id=2  
content=??▒????????????▒????????▒????▒?????????????????????????????  
created=1264244709  
…..这里省略….  
20. document=17, weight=1, cat\_id=1, member\_id=2, created=Sat Jan 23 19:05:09 2010  
id=17  
title=????????????  
cat\_id=1  
member\_id=2  
content=??????????????????????????????????????????????????????????  
created=1264244709

words:  
1. ‘北京’: 995 documents, 999 hits

至此，可以看到，我们已经检索出所有有关“北京”的信息

注意：这里我使用的是putty的客户端，在客户端编码设置的是utf-8，这个是测试的前提条件  
4.2 API调用

在本例中，我使用PHP的api来测试，在测试前，先启动sphinx服务进程，并对centos的防火墙做好9312端口的开放

[root@localhost sphinx]# bin/searchd -c etc/sphinx.conf & ### 使sphinx在后台运行  
[1] 5759  
[root@localhost sphinx]# Sphinx 0.9.9-release (r2117)  
Copyright (c) 2001-2009, Andrew Aksyonoff

using config file ‘etc/sphinx.conf’…  
listening on all interfaces, port=9312

[1]+ Done bin/searchd -c etc/sphinx.conf

php测试代码：

窗体顶端

 



窗体底端

SetServer(’192.168.1.150′, 9312); //注意这里的主机  
#$cl->SetMatchMode(SPH\_MATCH\_EXTENDED); //使用多字段模式  
//dump($cl);  
$index=”article”;  
$res = $cl->Query($keyword, $index);  
$err = $cl->GetLastError();  
dump($res);  
function dump($var)  
{  
echo ‘

';

var\_dump($var);

echo '

‘;  
}  
?>

检索“北京”dump后的结果是如下：

array(10) {  
["error"]=>  
string(0) “”  
["warning"]=>  
string(0) “”  
["status"]=>  
int(0)  
["fields"]=>  
array(2) {  
[0]=>  
string(5) “title”  
[1]=>  
string(7) “content”  
}  
["attrs"]=>  
array(3) {  
["cat\_id"]=>  
int(1)  
["member\_id"]=>  
int(1)  
["created"]=>  
int(2)  
}  
["matches"]=>  
array(20) {  
[76]=>  
array(2) {  
["weight"]=>  
string(1) “2″  
["attrs"]=>  
array(3) {  
["cat\_id"]=>  
string(1) “1″  
["member\_id"]=>  
string(1) “2″  
["created"]=>  
string(10) “1264244709″  
}  
}  
…..这里省略…..  
[17]=>  
array(2) {  
["weight"]=>  
string(1) “1″  
["attrs"]=>  
array(3) {  
["cat\_id"]=>  
string(1) “1″  
["member\_id"]=>  
string(1) “2″  
["created"]=>  
string(10) “1264244709″  
}  
}  
}  
["total"]=>  
string(3) “995″  
["total\_found"]=>  
string(3) “995″  
["time"]=>  
string(5) “0.008″  
["words"]=>  
array(1) {  
["北京"]=>  
array(2) {  
["docs"]=>  
string(3) “995″  
["hits"]=>  
string(3) “999″  
}  
}  
}

至此PHP已可调用出结果！

附录

这是本人对sphinx写的入门手册，也为自己存档使用。为写本文，特重新安装了一遍Sphinx，并新建mysql表加入1000条记录，动手重复了一下所有  
过程。如有错误或疑问请到以下地址反馈，谢谢！  
欢迎到sphinx中文站(www.sphinxsearch.org)讨论sphinx的相关问题，交流你我的思想！