# **solr**

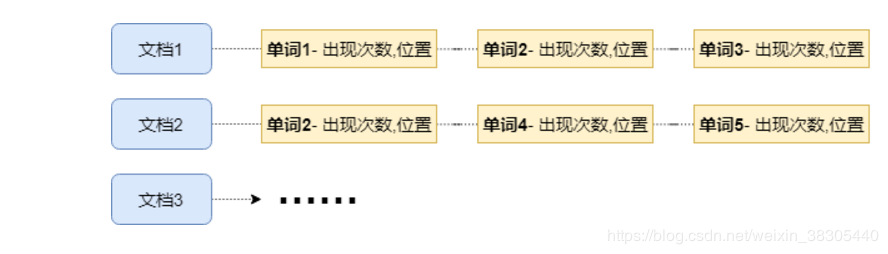
Solr是一个高性能，基于Lucene的全文搜索服务器。同时对其进行了扩展，提供了比Lucene更为丰富的查询语言，同时实现了可配置、可扩展，并对查询性能进行了优化，并且提供了一个完善的功能管理界面，是一款非常优秀的全文搜索引擎。

# **lucene**

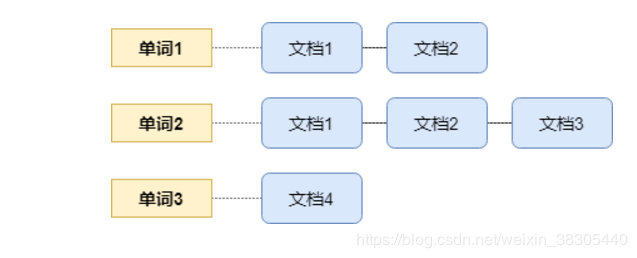
Lucene是apache jakarta项目的一个子项目，是一个开放源代码的全文检索引擎开发工具包，但它不是一个完整的全文检索引擎，而是一个全文检索引擎的架构，提供了完整的查询引擎和索引引擎，部分文本分析引擎。Lucene的目的是为软件开发人员提供一个简单易用的工具包，以方便的在目标系统中实现全文检索的功能，或者是以此为基础建立起完整的全文检索引擎。

## **倒排索引**

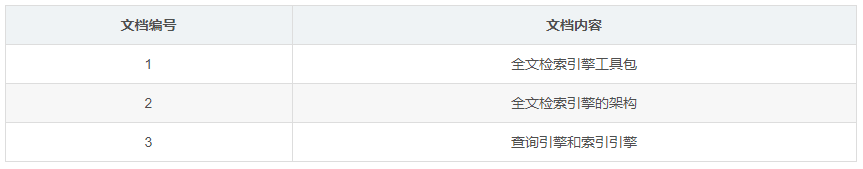
我们一般情况下,先找到文档,再在文档中找出包含的词;



倒排索引则是这个过程反过来,用词,来找出它出现的文档.



## **实际举例**



分词结果



倒排索引



## **lucene API 介绍**

### **创建索引**

#### **新建 maven 项目,添加依赖**

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>cn.tedu</groupId>

<artifactId>lucene-demo</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>luceme-demo</name>

<dependencies>

<dependency>

<groupId>org.apache.lucene</groupId>

<artifactId>lucene-core</artifactId>

<version>8.1.1</version>

</dependency>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.12</version>

</dependency>

<dependency>

<groupId>org.apache.lucene</groupId>

<artifactId>lucene-analyzers-smartcn</artifactId>

<version>8.1.1</version>

</dependency>

</dependencies>

</project>

#### **创建测试类,添加以下代码**

package test;

import java.io.File;

import org.apache.lucene.analysis.cn.smart.SmartChineseAnalyzer;

import org.apache.lucene.document.Document;

import org.apache.lucene.document.Field.Store;

import org.apache.lucene.document.LongPoint;

import org.apache.lucene.document.StoredField;

import org.apache.lucene.document.TextField;

import org.apache.lucene.index.IndexWriter;

import org.apache.lucene.index.IndexWriterConfig;

import org.apache.lucene.store.FSDirectory;

import org.junit.Test;

public class Test1 {

String[] a = {

"3, 华为 - 华为电脑, 爆款",

"4, 华为手机, 旗舰",

"5, 联想 - Thinkpad, 商务本",

"6, 联想手机, 自拍神器"

};

@Test

public void test1() throws Exception {

//存储索引文件的路径

File path = new File("d:/abc/");

FSDirectory d = FSDirectory.open(path.toPath());

//lucene提供的中文分词器

SmartChineseAnalyzer analyzer = new SmartChineseAnalyzer();

//通过配置对象来指定分词器

IndexWriterConfig cfg = new IndexWriterConfig(analyzer);

//索引输出工具

IndexWriter writer = new IndexWriter(d, cfg);

for (int i = 0; i < a.length; i++) {

String[] strs = a[i].split(",");

//创建文档,文档中包含的是要索引的字段

Document doc = new Document();

doc.add(new LongPoint("id", Long.parseLong(strs[0])));

doc.add(new StoredField("id", Long.parseLong(strs[0])));

doc.add(new TextField("title", strs[1], Store.YES));

doc.add(new TextField("sellPoint", strs[2], Store.YES));

//将文档写入磁盘索引文件

writer.addDocument(doc);

}

writer.close();

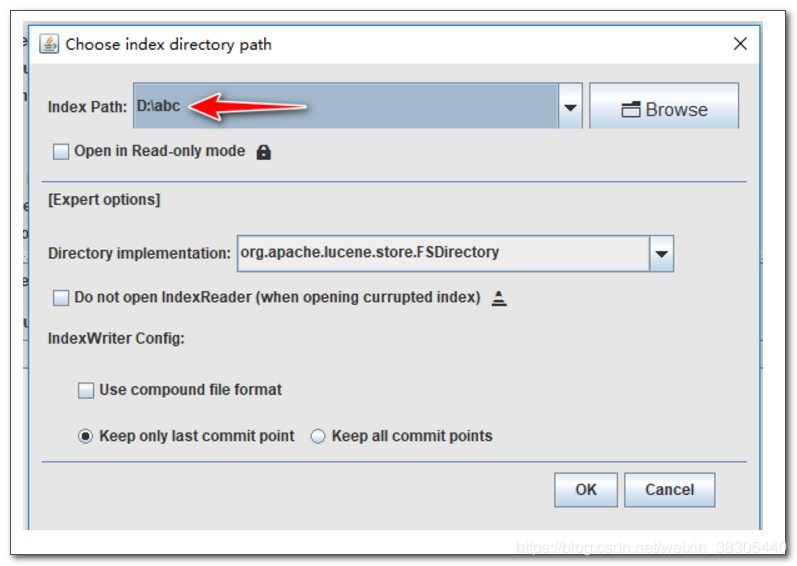
}

}

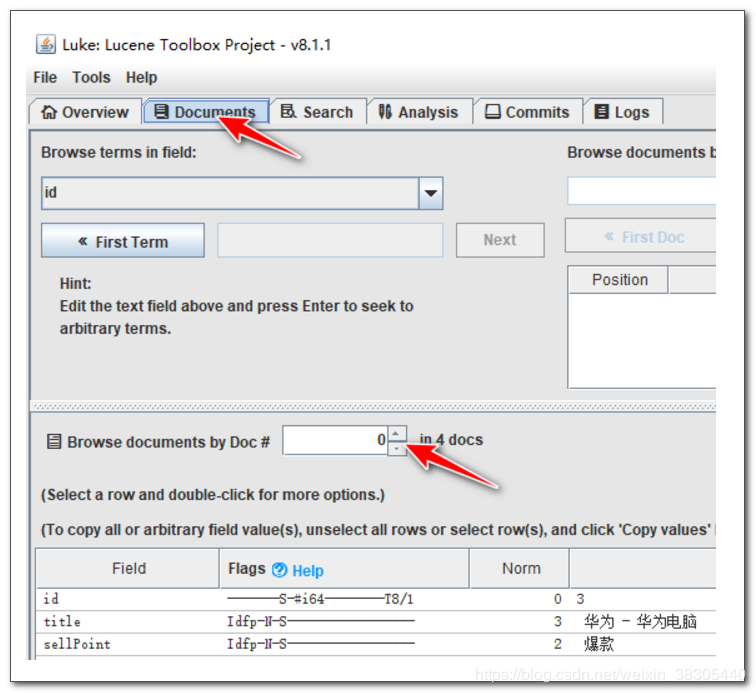
### **查看索引**

#### **运行 luke**

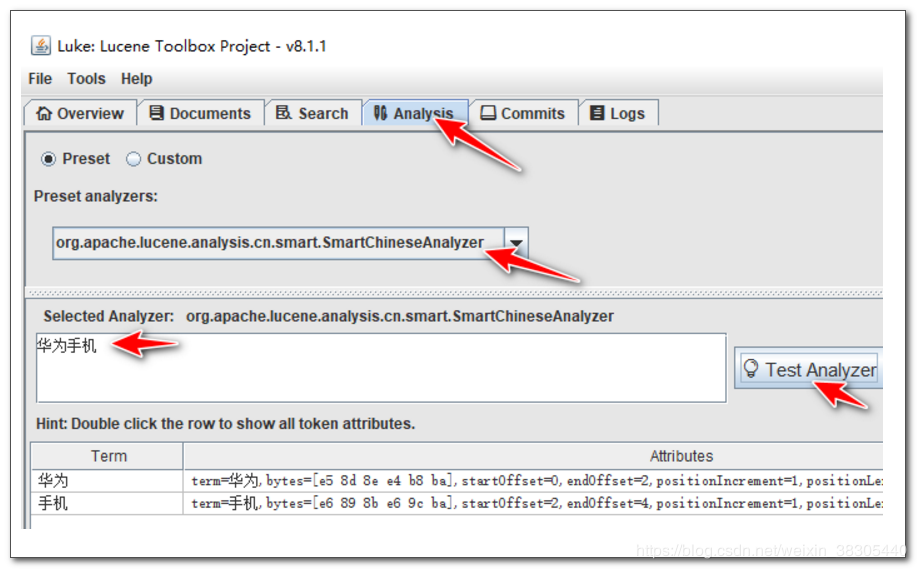
运行lucene 8.1.1中的luke应用程序,指定索引的存放目录



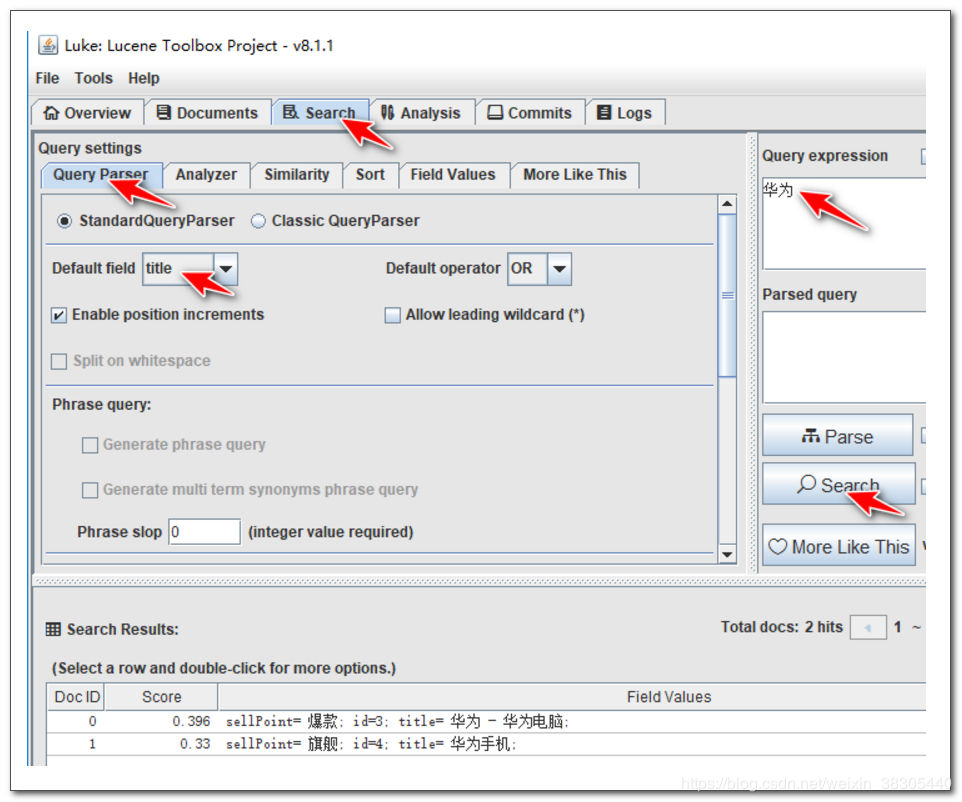
#### **查看文档**



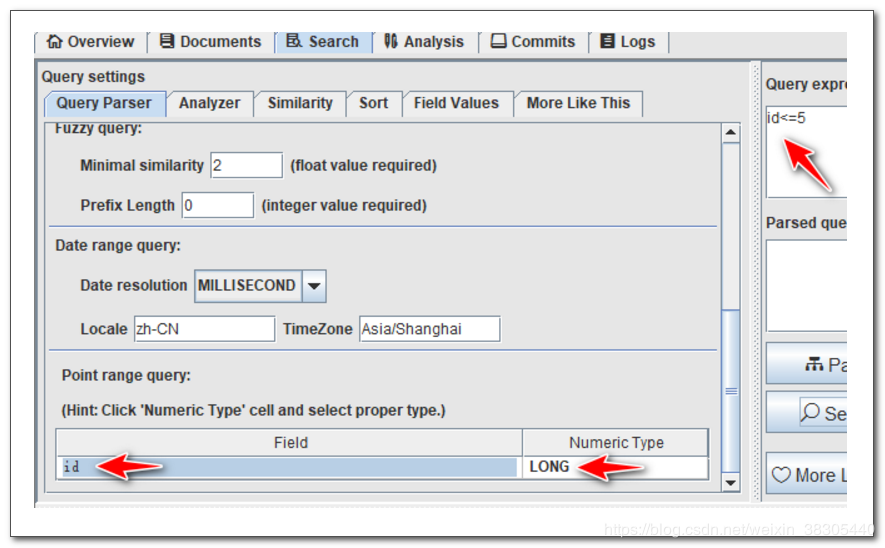
#### **指定分词器,并测试分词**



#### **查询测试**



id的查询



### **从索引查询**

在测试类中添加 test2() 测试方法

package test;

import java.io.File;

import org.apache.lucene.analysis.cn.smart.SmartChineseAnalyzer;

import org.apache.lucene.document.Document;

import org.apache.lucene.document.Field.Store;

import org.apache.lucene.document.LongPoint;

import org.apache.lucene.document.StoredField;

import org.apache.lucene.document.TextField;

import org.apache.lucene.index.DirectoryReader;

import org.apache.lucene.index.IndexWriter;

import org.apache.lucene.index.IndexWriterConfig;

import org.apache.lucene.index.Term;

import org.apache.lucene.search.IndexSearcher;

import org.apache.lucene.search.ScoreDoc;

import org.apache.lucene.search.TermQuery;

import org.apache.lucene.search.TopDocs;

import org.apache.lucene.store.FSDirectory;

import org.junit.Test;

public class Test1 {

String[] a = {

"3, 华为 - 华为电脑, 爆款",

"4, 华为手机, 旗舰",

"5, 联想 - Thinkpad, 商务本",

"6, 联想手机, 自拍神器"

};

@Test

public void test1() throws Exception {

//存储索引文件的路径

File path = new File("d:/abc/");

FSDirectory d = FSDirectory.open(path.toPath());

//lucene提供的中文分词器

SmartChineseAnalyzer analyzer = new SmartChineseAnalyzer();

//通过配置对象来指定分词器

IndexWriterConfig cfg = new IndexWriterConfig(analyzer);

//索引输出工具

IndexWriter writer = new IndexWriter(d, cfg);

for (int i = 0; i < a.length; i++) {

String[] strs = a[i].split(",");

//创建文档,文档中包含的是要索引的字段

Document doc = new Document();

doc.add(new LongPoint("id", Long.parseLong(strs[0])));

doc.add(new StoredField("id", Long.parseLong(strs[0])));

doc.add(new TextField("title", strs[1], Store.YES));

doc.add(new TextField("sellPoint", strs[2], Store.YES));

//将文档写入磁盘索引文件

writer.addDocument(doc);

}

writer.close();

}

@Test

public void test2() throws Exception {

//索引数据的保存目录

File path = new File("d:/abc");

FSDirectory d = FSDirectory.open(path.toPath());

//创建搜索工具对象

DirectoryReader reader = DirectoryReader.open(d);

IndexSearcher searcher = new IndexSearcher(reader);

//关键词搜索器,我们搜索 "title:华为"

TermQuery q = new TermQuery(new Term("title", "华为"));

//执行查询,并返回前20条数据

TopDocs docs = searcher.search(q, 20);

//遍历查询到的结果文档并显示

for (ScoreDoc scoreDoc : docs.scoreDocs) {

Document doc = searcher.doc(scoreDoc.doc);

System.out.println(doc.get("id"));

System.out.println(doc.get("title"));

System.out.println(doc.get("sellPoint"));

System.out.println("--------------");

}

}

}

# **solr 安装**

下面我们来安装 solr 服务器

## **把 solr-8.1.1.tgz 传到服务器**

先切换到 /usr/local 目录

cd /usr/local

把文件传到 /usr/local 目录下

## **解压 solr**

cd /usr/local

# 上传 solr-8.1.1.tgz 到 /usr/local 目录

# 并解压缩

tar -xzf solr-8.1.1.tgz

## **启动 solr**

cd /usr/local/solr-8.1.1

# 不建议使用管理员启动 solr,加 -force 强制启动

bin/solr start -force

# 开放 8983 端口

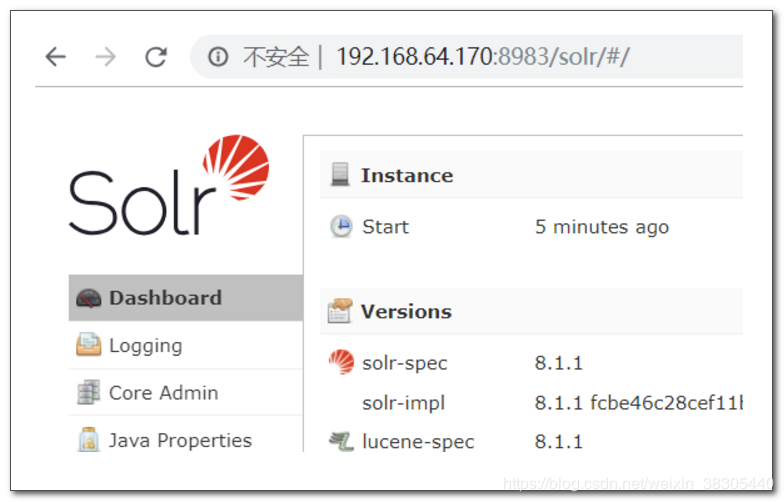
firewall-cmd --zone=public --add-port=8983/tcp --permanent

firewall-cmd --reload

## **浏览器访问 solr 控制台**

http://192.168.64.170:8983

* 注意修改地址



# **创建 core**

数据库中 pd\_item 表中的商品数据, 在 solr 中保存索引数据, 一类数据, 在 solr 中创建一个 core 保存索引数据

创建一个名为 pd 的 core, 首先要准备以下目录结构:

# solr目录/server/solr/

# pd/

# conf/

# data/

cd /usr/local/solr-8.1.1

mkdir server/solr/pd

mkdir server/solr/pd/conf

mkdir server/solr/pd/data

conf 目录是 core 的配置目录, 存储一组配置文件, 我们以默认配置为基础, 后续逐步修改

## **复制默认配置**

cd /usr/local/solr-8.1.1

cp -r server/solr/configsets/\_default/conf server/solr/pd

## **创建名为 pd 的 core**

