全文检索服务器

# Solr介绍

## 什么是solr

Solr是一个独立的[企业级搜索](https://baike.baidu.com/item/%E4%BC%81%E4%B8%9A%E7%BA%A7%E6%90%9C%E7%B4%A2" \t "_blank)应用服务器，它对外提供类似于Web-service的API接口。用户可以通过http请求，向搜索引擎服务器提交一定格式的XML文件，生成索引；也可以通过Http Get操作提出查找请求，并得到XML格式的返回结果。

Solr是apache的顶级开源项目，它是使用java开发 ，基于lucene的全文检索服务器。

Solr比lucene提供了更多的查询语句，而且它可扩展、可配置，同时它对lucene的性能进行了优化。

Solr是如何实现全文检索的呢？

索引流程：solr客户端（浏览器、java程序）可以向solr服务端发送POST请求，请求内容是包含Field等信息的一个xml文档，通过该文档，solr实现对索引的维护（增删改）

搜索流程：solr客户端（浏览器、java程序）可以向solr服务端发送GET请求，solr服务器返回一个xml/json文档。

## Solr和lucene的区别

Lucene是一个全文检索引擎工具包，它只是一个jar包，不能独立运行，对外提供服务。

Solr是一个全文检索服务器，它可以单独运行在servlet容器，可以单独对外提供搜索和索引功能。Solr比lucene在开发全文检索功能时，更快捷、更方便。



# Solr安装配置

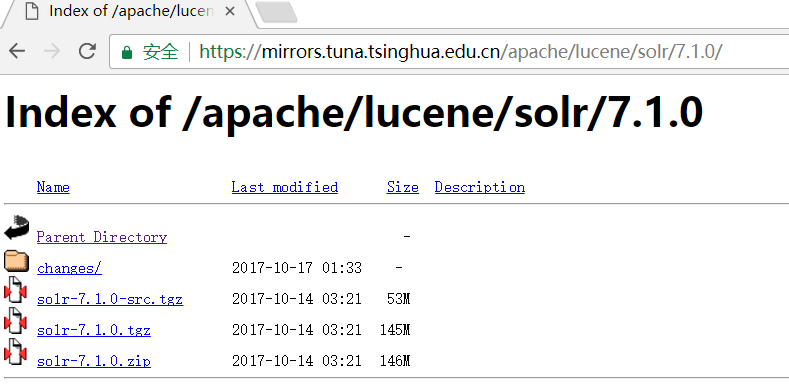
## 下载solr

Solr和lucene的版本是同步更新的，最新的版本是[8.1.0/](https://mirrors.tuna.tsinghua.edu.cn/apache/lucene/solr/7.1.0/)

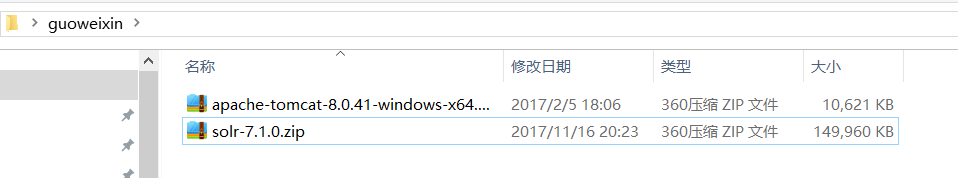
本课程使用的版本：7.1.0

下载地址：https://mirrors.tuna.tsinghua.edu.cn/apache/lucene/solr/

下载版本：7.1.0

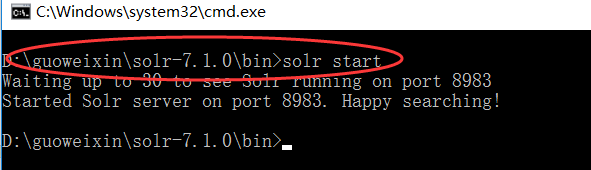


1. src.tgz：带src表示是带源码文件的压缩包，无src是已经编译过的压缩包   
2. .tgz：Linux相关操作系统使用的压缩包   
3. .zip：Windows操作系统使用的压缩包



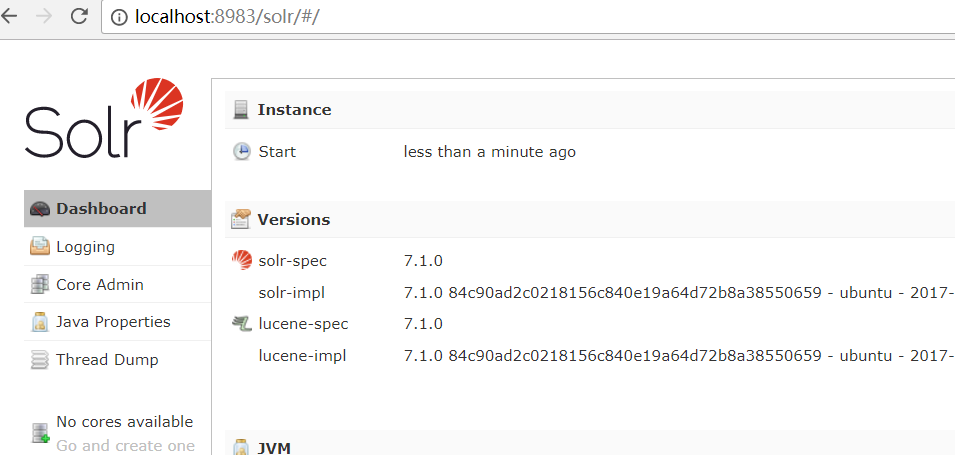
**Jetty** 是一个开源的servlet容器，它为基于Java的web容器，例如JSP和servlet提供运行环境。Jetty是使用[Java语言](https://baike.baidu.com/item/Java%E8%AF%AD%E8%A8%80" \t "_blank)编写的，它的API以一组JAR包的形式发布。开发人员可以将Jetty容器实例化成一个对象，可以迅速为一些独立运行（stand-alone）的Java应用提供网络和web连接

## 启动Solr:



solr的启动、停止、查看命令：   
1. 启动：bin\solr.cmd start   
2. 停止：bin\solr stop 或bin\solr stop -all   
3. 查看：bin\solr status

**访问地址：http://localhost:8983/solr/#/**

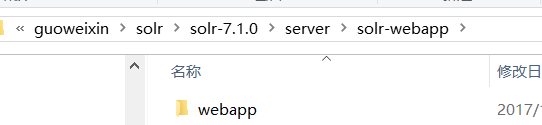


## Tomcat安装运行环境

* Jdk：1.8及以上
* Solr：7.1
* Web服务器：tomcat 8（改端口9080）

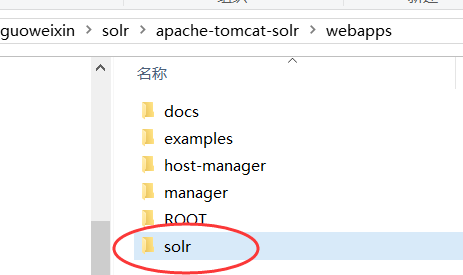
### 操作步骤

**1、找到如下目录内容：**

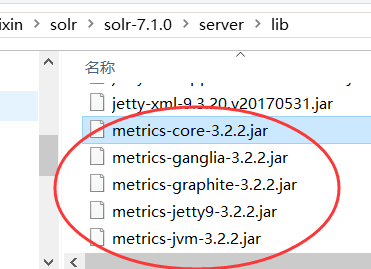
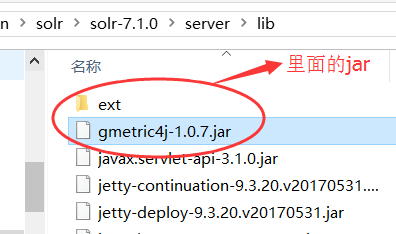
D:\guoweixin\solr\solr-7.1.0\server\solr-webapp

2、复制到Tomcat下，并命令为solr

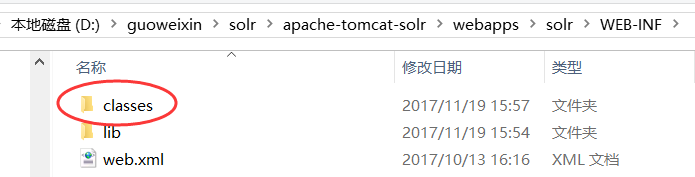
D:\guoweixin\solr\apache-tomcat-solr\webapps

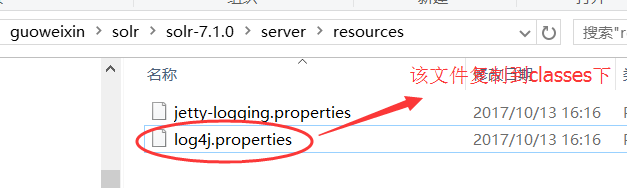


**2、将solr-7.1.0\server\lib\ext下的所有jar包，以及solr-7.1.0\server\lib下以metrics开头的jar、gmetric4j-1.0.7.jar复制到apache-tomcat-8\webapps\solr\WEB-INF\lib下**



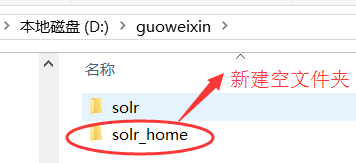
**3、在tomcat\webapps\solr\WEB-INF中，新建classes文件夹，将solr-7.1.0\server\resources下的log4j.properties文件拷贝到里面**





**创建Solr CoreAdmin管理（Solr Home）**

**4、创建SOLR HOME目录（solr\_home）**

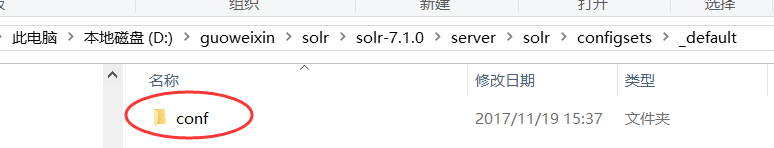


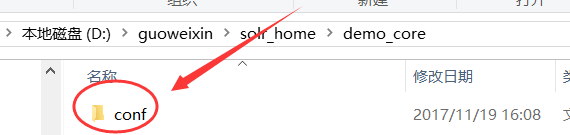
**5、拷贝solr-7.1.0\server\solr\下所有文件、文件夹复制到solr\_home目录下**

**6、拷贝solr-7.1.0 下contrib和dist文件夹至solr\_home目录下**

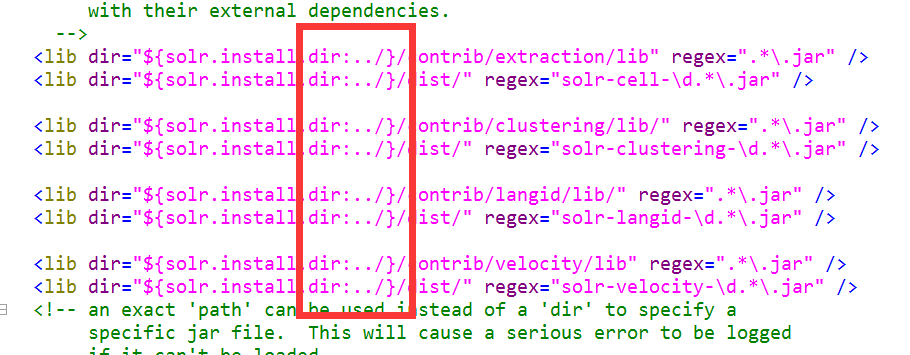
**7、在solr\_home目录下新建demo\_core文件夹；**

**并拷贝D:\guoweixin\solr\solr\_home\configsets\sample\_techproducts\_configs\目录下conf文件夹至solr\_home\demo\_core下**



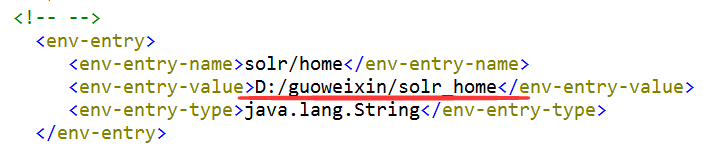


**8(需要)、修改solr\_home\demo\_core\conf\solrconfig.xml文件，如下(75行处)**

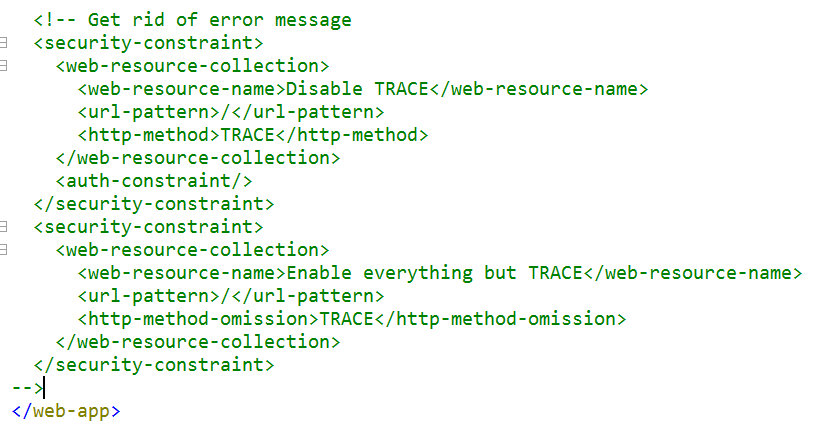


**9、修改tomcat\webapps\solr\WEB-INF中的web.xml文件**

新增如下部分，默认是注释掉的。 (40行处)  
其中env-entry-value值为solr\_home对应值，即solr home目录



**10、修改tomcat\webapps\solr\WEB-INF中的web.xml文件**（注释如下部分内容）最下面162行

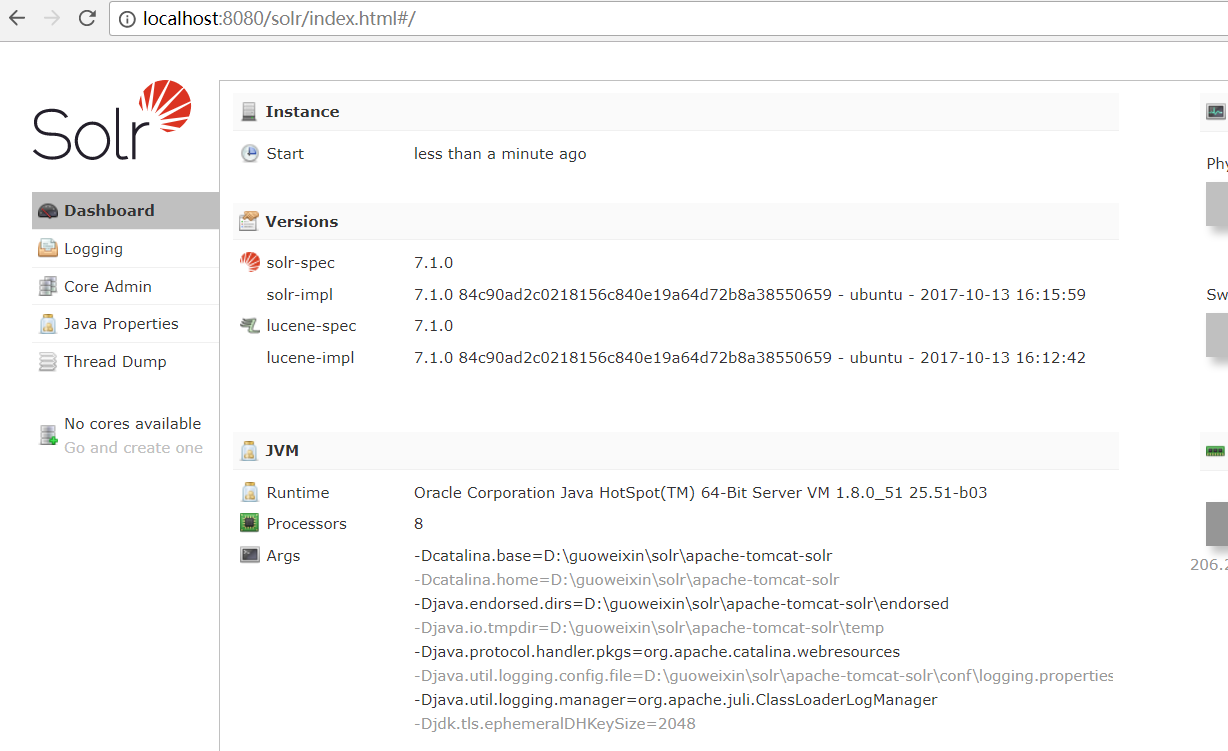


**11、启动Tomcat**

启动tomcat，访问[http://localhost:8080/solr/index.html](http://localhost:8080/solr/index.html" \t "_blank)

## solr界面介绍

启动solr服务

http://localhost:8080/solr/index.html

### Dashboard

仪表盘，显示了该Solr实例开始启动运行的时间、版本、系统资源、jvm等信息。

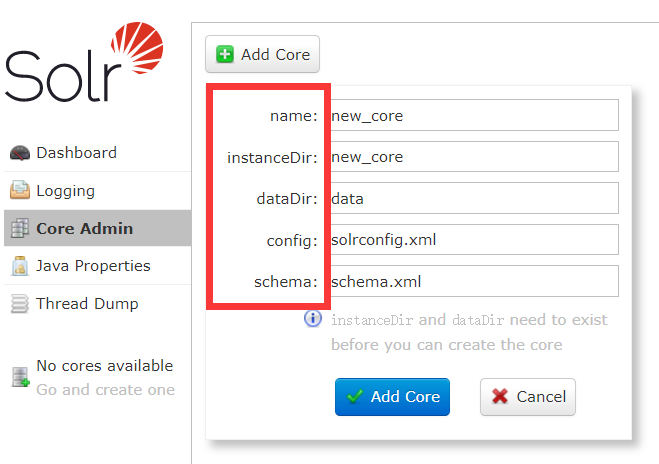
### Logging

显示solr运行出现的异常或错误

### Core Admin

Solr Core的管理界面。在这里可以添加SolrCore实例。

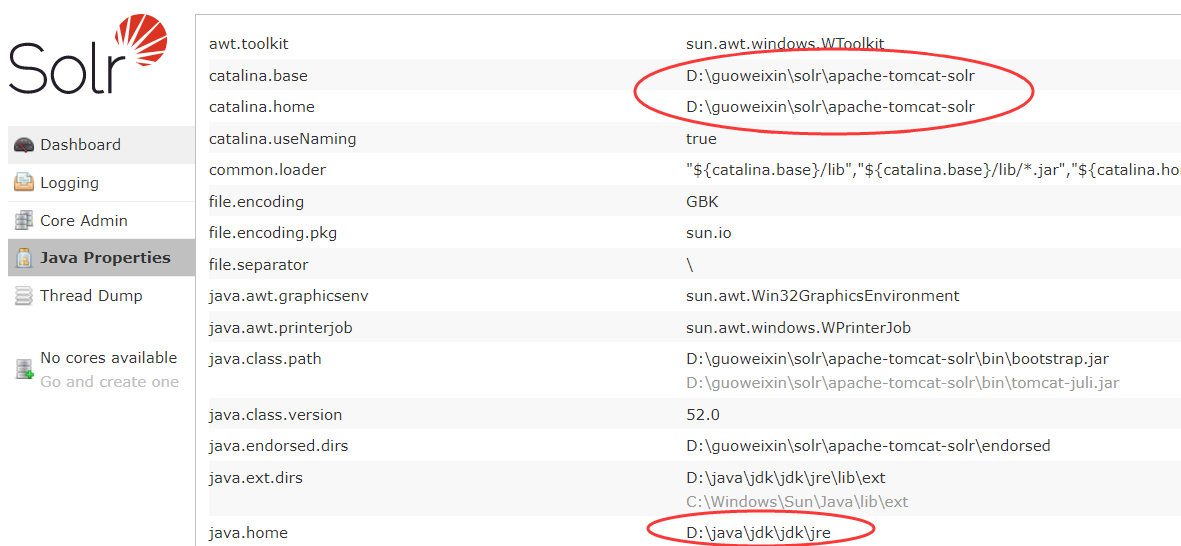
主要有Add Core(添加核心)， Unload（卸载核心），Rename（重命名核心），Reload(重新加载核心)，Optimize（优化索引库）   
Add Core是添加core：主要是在instanceDir对应的文件夹里生成一个core.properties文件



name：给core起的名字；   
instanceDir：与我们在配置solr到tomcat里时的solr\_home里新建的core文件夹名一致；   
dataDir:确认Add Core时，会在new\_core目录下生成名为data的文件夹   
config：new\_core下的conf下的config配置文件(solrconfig.xml)   
schema: new\_core下的conf下的schema文件(schema.xml)

### java properties

Solr在JVM 运行环境中的属性信息，包括类路径、文件编码、jvm内存设置等信息。可查看到java相关的一些属性的信息



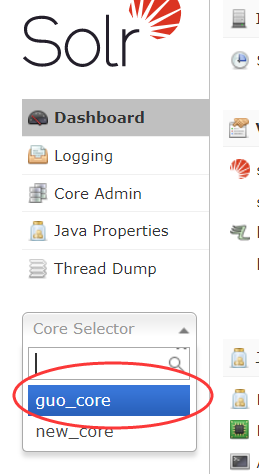
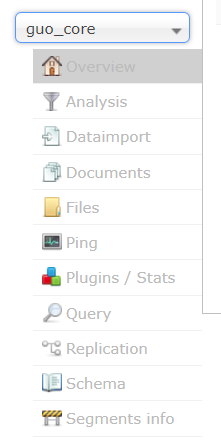
### Tread Dump

显示Solr Server中当前活跃线程信息，同时也可以跟踪线程运行栈信息。

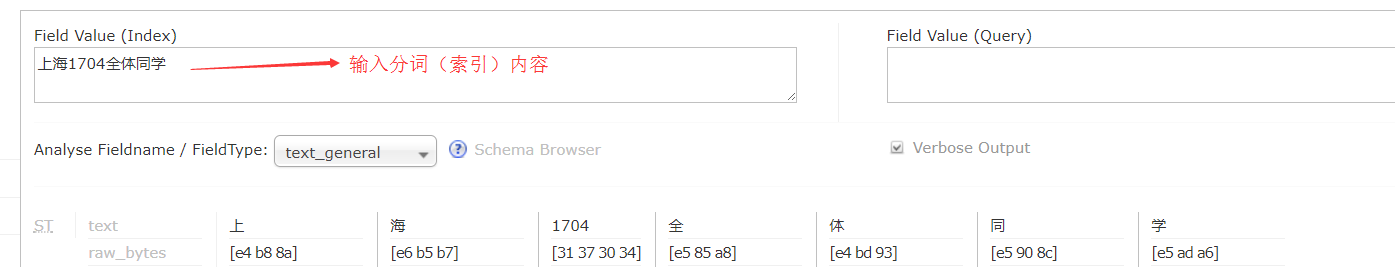
### Core selector（重点）

需要在Core Admin里添加了core后才有可选项。

选择一个SolrCore进行详细操作，如下：

#### Analysis（重点）



通过此界面可以测试索引分析器和搜索分析器的执行情况。

注：solr中，**分析器是绑定在域的类型中的**。

#### dataimport

可以定义数据导入处理器，从关系数据库将数据导入到Solr索引库中。

默认没有配置，需要手工配置。

#### Document

通过/update表示更新索引，**solr默认根据id（唯一约束）域来更新Document的内容，如果根据id值搜索不到id域则会执行添加操作，如果找到则更新**。

通过此菜单可以**创建索引、更新索引、删除索引**等操作

#### Query（重点）

通过/select执行搜索索引，必须指定“q”查询条件方可搜索。



## Solr core的配置（创建）

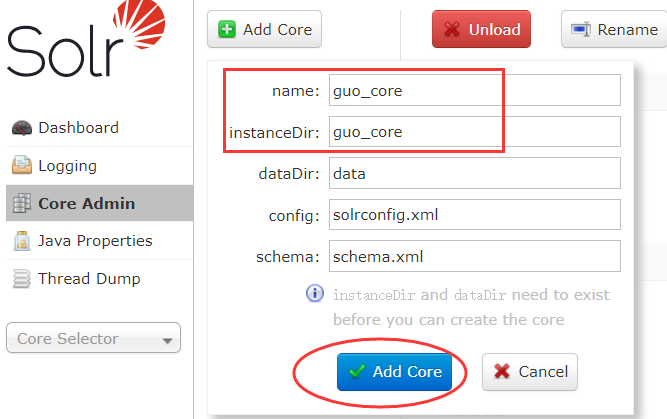
**1、新建core(**创建guo\_core文件夹**)**

新建的solrhome文件的位置为：D:\guoweixin\solr\solr\_home\guo\_core

**2.D:\guoweixin\solr\solr\_home\configsets\sample\_techproducts\_configs\目录下conf文件夹复制到solr\_home\guo\_core下**

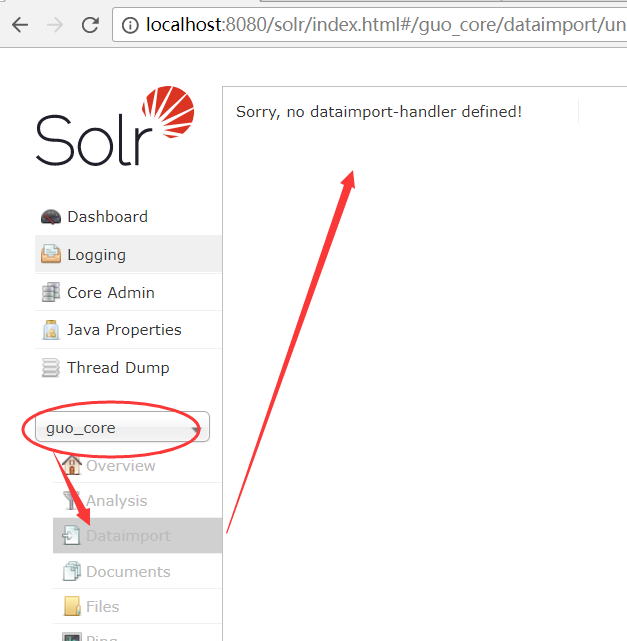
**3.在http://localhost:8080/solr/index.html管理界面中添加guo\_core**

点击 **add Core**按钮进行添加



# Solr导入数据

## 查看core



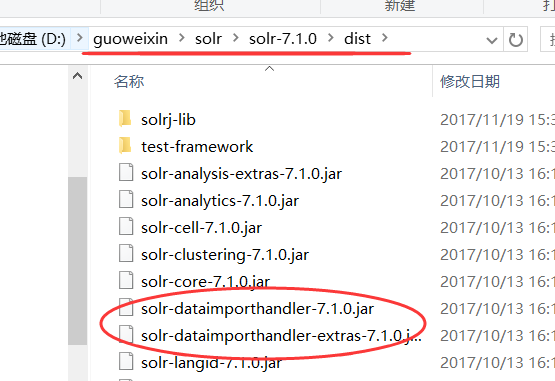
通过此查看创建的guo\_core。则说明没有相关配置文件，需要添加并修改

## DataImport导入数据

**该功能是将数据库中数据通过Sql语句方式导入到Solr索引库中。**

### 第一步：添加jar包

D:\guoweixin\solr\solr-7.1.0\dist 下的jar



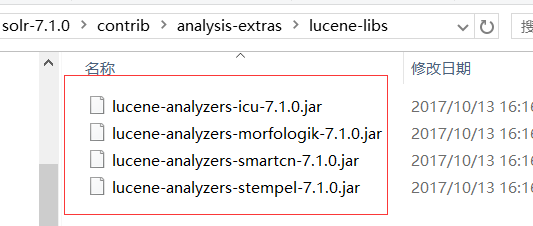
mysql jar:



--------------------------------------------------------------------------------------------------------------------------

配置Solr:找到自带的中文分词器

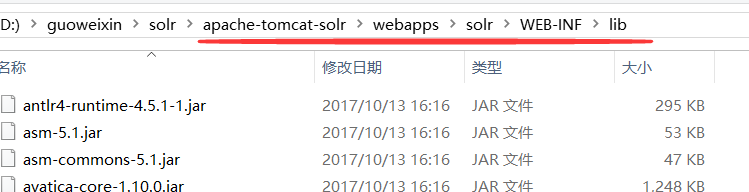
D:\guoweixin\solr\solr-7.1.0\contrib\analysis-extras\lucene-libs



---------------------------------------------------------------------------------------------------------------------------------

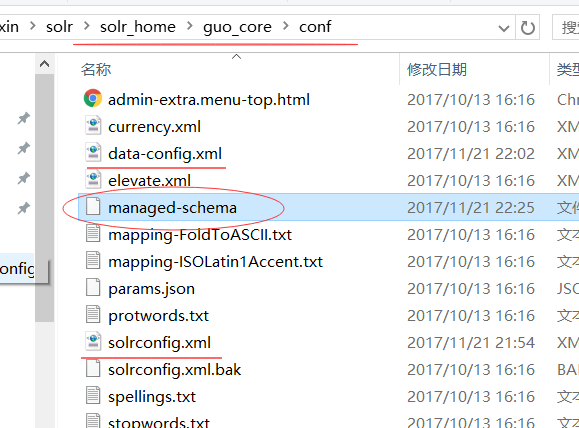
---------------------------------------------------------------------------------------------------------------------------------

以上所有jar全部复制到 webapps/solr/WEB-INF/lib下

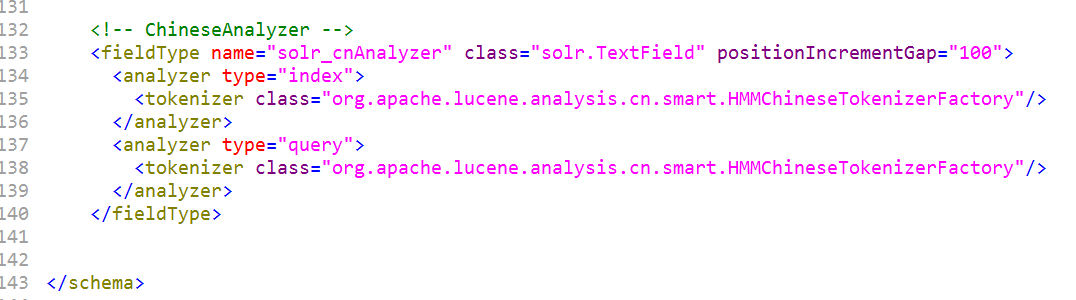


### 第二步：配置中文分词器

修改同目录下的managed-schema文件



**1、在文件最后方插入中文分词器**：



|  |
| --- |
| <!-- ChineseAnalyzer -->  <fieldType name="solr\_cnAnalyzer" class="solr.TextField" positionIncrementGap="100">  <analyzer type="index">  <tokenizer class="org.apache.lucene.analysis.cn.smart.HMMChineseTokenizerFactory"/>  </analyzer>  <analyzer type="query">  <tokenizer class="org.apache.lucene.analysis.cn.smart.HMMChineseTokenizerFactory"/>  </analyzer>  </fieldType> |

### managed-schema文件详解

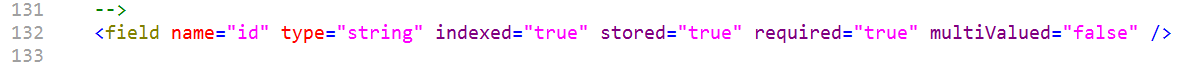
**找到solr\_home\core\_meeting\conf下managed-schema文件**

在managed-schema.xml文件中，主要配置SolrCoe数据信息，包括：Field和FieldType的定义等信息，**在solr中，Field和FieldType都需要先定义后使用**

#### Field详解：

学习详细参考：<http://blog.csdn.net/mine_song/article/details/58065323>

https://www.cnblogs.com/gslblog/p/6582008.html



name：指定域的名称(自定义)

type：指定域的类型

indexed：是否索引

是：(将分好的词进行索引，索引的目的，就是为了搜索)

否：不索引，也就是不对该field域进行搜索。

stored：是否存储

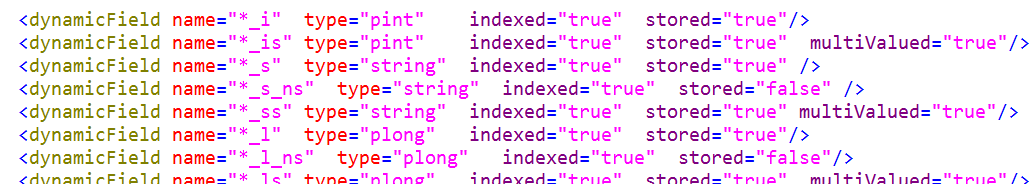
是：将field域中的内容存储到文档域中。存储的目的，就是为了搜索页面显示取值用的

否：不将field域中的内容存储到文档域中。不存储，则搜索页面中没法获取该field域的值。

required：是否必须

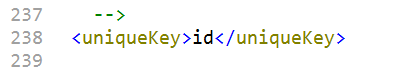
multiValued：是否多值，比如查询数据需要关联多个字段数据，一个Field存储多个值信息，必须将multiValued设置为true。

#### dynamicField详解(199行)：



name为\*\_i，定义它的type为int，那么在使用这个字段的时候，任何以\_i结果的字段都被认为符合这个定义

#### uniqueKey



其中的id是在Field标签中已经定义好的域名，而且该域设置为required为true。

一个managed-schema文件中必须有且仅有一个唯一键

#### copyField

复制域

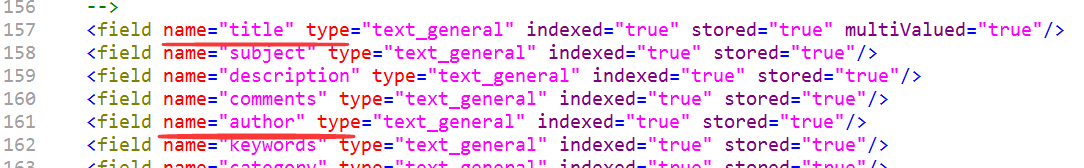
  应用场景：我们在搜索时比如输入java，一篇文章分为标题、简介、内容等很多字段，输入的关键字需要制定solr中的域进行检索，不可能从一个表中将所有字段进行索引，因为有些字段不需要索引，所以出现copyField域,把多个域的关键词复制到同一个域，多个域时，可以放到一个域中。就不用定义那么多域了。搜索比较方便

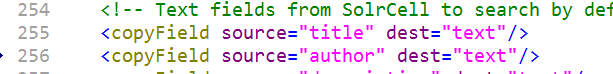
实例：<copyField source="projectName" dest="keywords"/>

Source:是Field域的名称

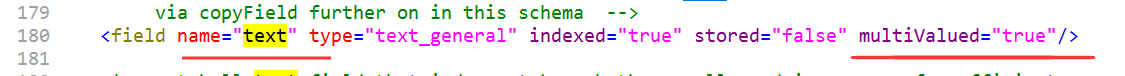
Dest：是destination的缩写 目标域

使用案例：

1. 两个普通域 ：title和author，
2. 使用复制域，将两个域进行索引检索

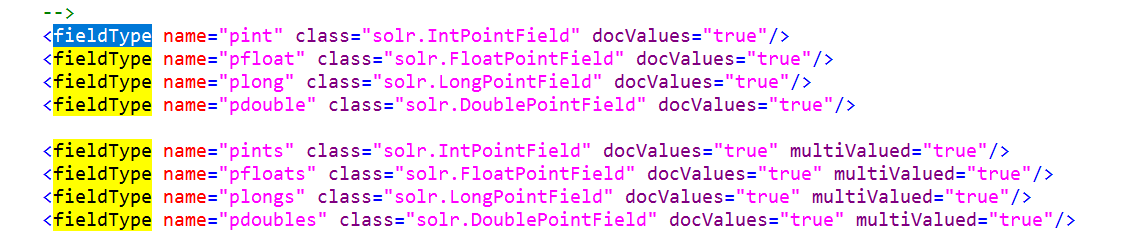


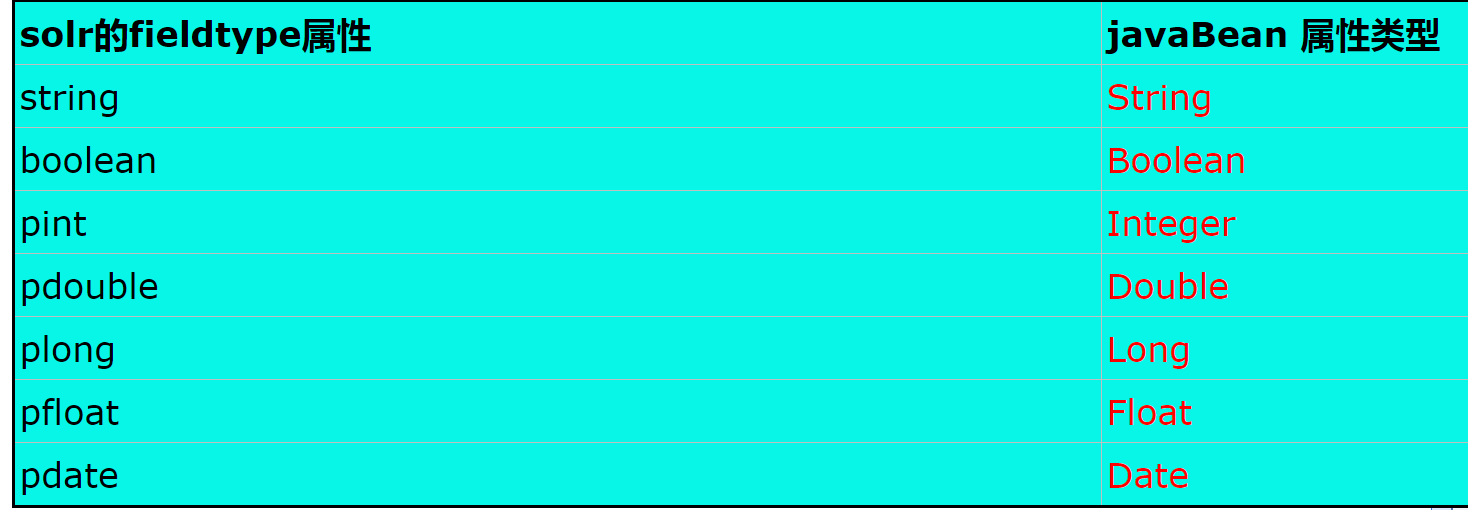
1. 该域名field name=”text”即是复制域



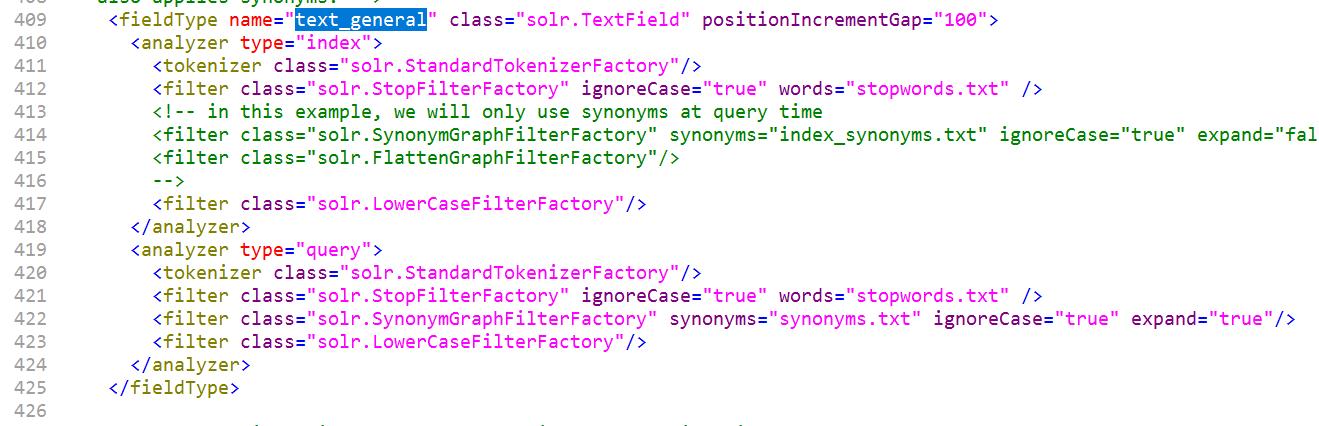
#### fieldType

域类型：





**分词器**：



Name：指定域类型的名称

Class：指定该域类型对应的solr的类型

Analyzer：指定分析器

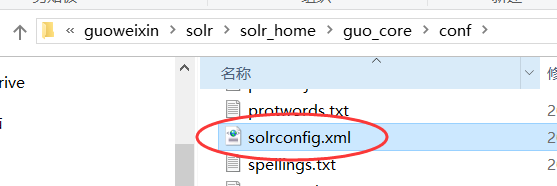
Type：index、query，分别指定搜索和索引时的分析器

Tokenizer：指定分词器

Filter：指定过滤器

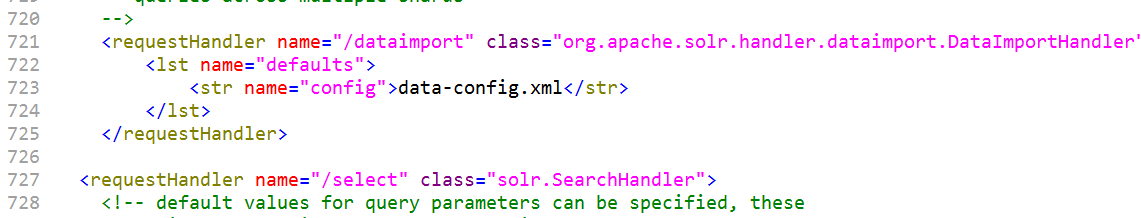
### 第三步：修改 solrconfig.xml文件

找到D:\guoweixin\solr\solr\_home\guo\_core\conf文件夹下的solrconfig.xml，添加dataimport



首先查询是否存在dataimport的requestHandler，如果不存在，因此需要手动添加。为了以后便于维护此文件，我们就在requestHandler起始位置，约为720行处，添加如下内容

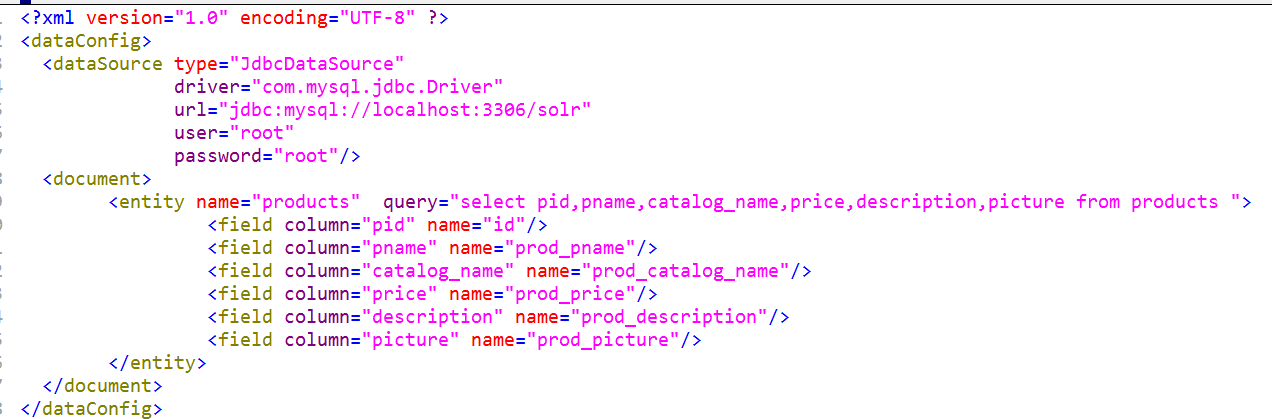
|  |
| --- |
| <requestHandler name="/dataimport" class="org.apache.solr.handler.dataimport.DataImportHandler">  <lst name="defaults">  <str name="config">data-config.xml</str>  </lst>  </requestHandler> |



### 第四步：创建data-config.xml

**data-config.xml作用：数据库连接相关信息、SQL以及查询结果映射对应域中**

在solrconfig.xml同级目录下，创建data-config.xml



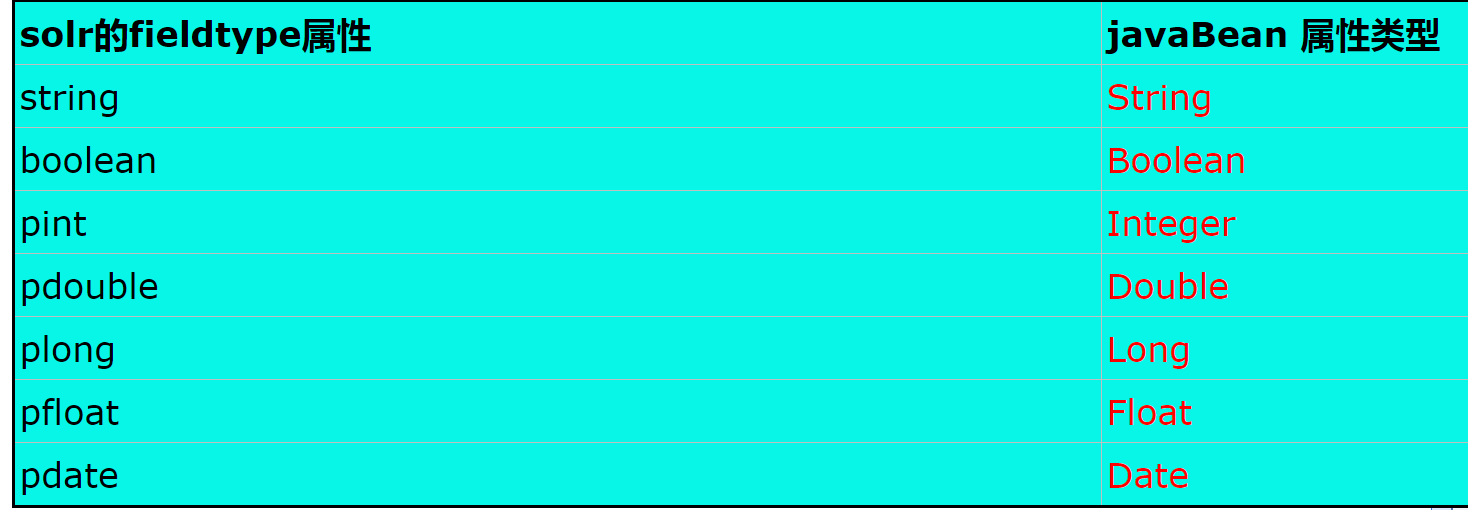
Query代表：查询的SQL语句

Column代表：SQL查询的列名

Name代表：Solr 域Filed中的域名

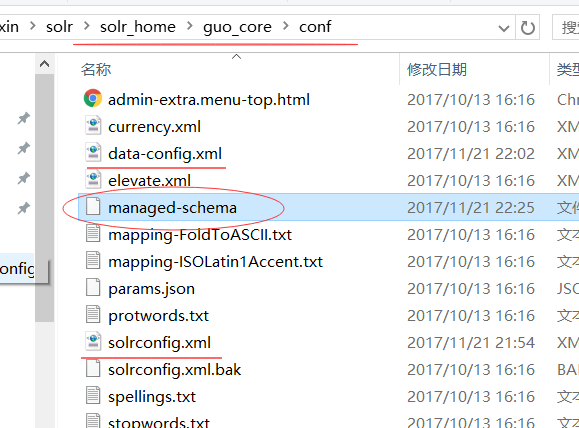
|  |
| --- |
| <?xml version="1.0" encoding="UTF-8" ?>  <dataConfig>  <dataSource type="JdbcDataSource"  driver="com.mysql.jdbc.Driver"  url="jdbc:mysql://localhost:3306/solr"  user="root"  password="root"/>  <document>  <entity name="products" query="select pid,pname,catalog\_name,price,description,picture from products ">  <field column="pid" name="id"/>  <field column="pname" name="prod\_pname"/>  <field column="catalog\_name" name="prod\_catalog\_name"/>  <field column="price" name="prod\_price"/>  <field column="description" name="prod\_description"/>  <field column="picture" name="prod\_picture"/>  </entity>  </document>  </dataConfig> |



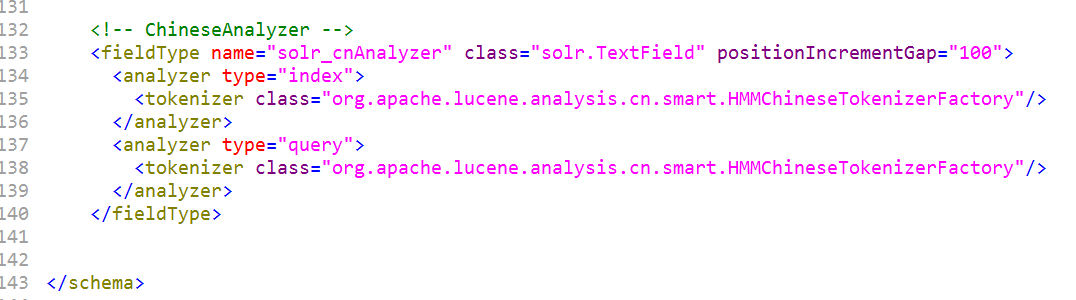


### 第四步：定义域

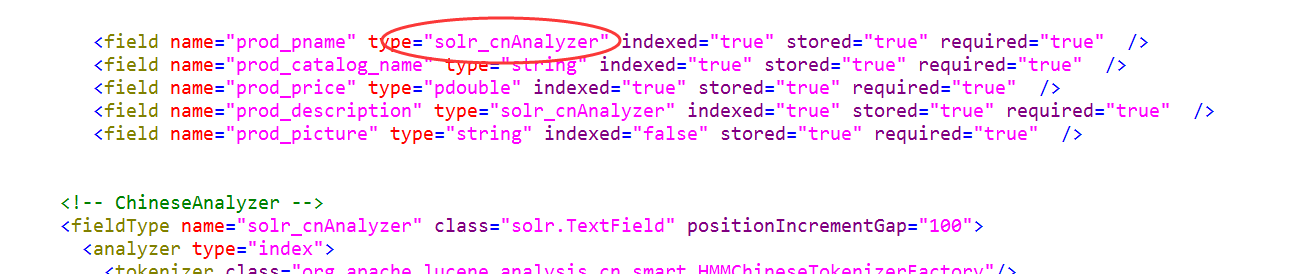
修改同目录下的managed-schema文件



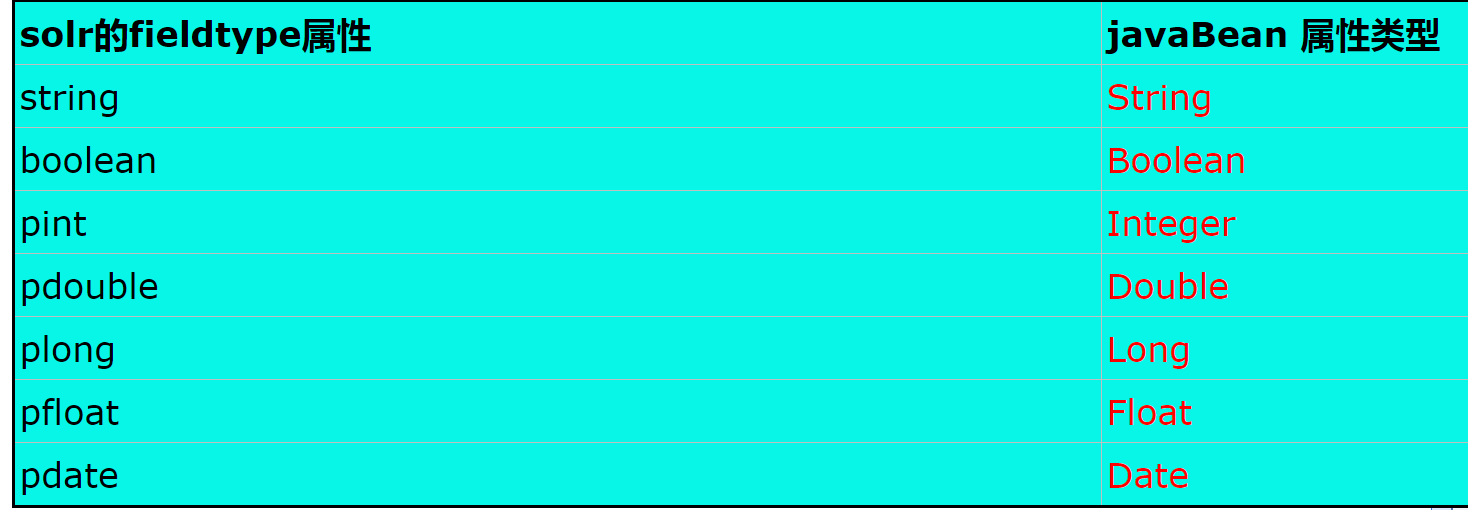
**1、在文件最后方插入中文分词器**：



**同时还要配置其Field域：**来配置字段【这里的名字要与data-config中的域名一模一样】



|  |
| --- |
| <field name="prod\_pname" type="solr\_cnAnalyzer" indexed="true" stored="true" required="true" />  <field name="prod\_catalog\_name" type="string" indexed="true" stored="true" required="true" />  <field name="prod\_price" type="pdouble" indexed="true" stored="true" required="true" />  <field name="prod\_description" type="solr\_cnAnalyzer" indexed="true" stored="true" required="true" />  <field name="prod\_picture" type="string" indexed="false" stored="true" required="true" /> |
| **其中prod\_pname需要模糊查询（中文分词），故将其类型改为solr\_cnAnalyzer** |

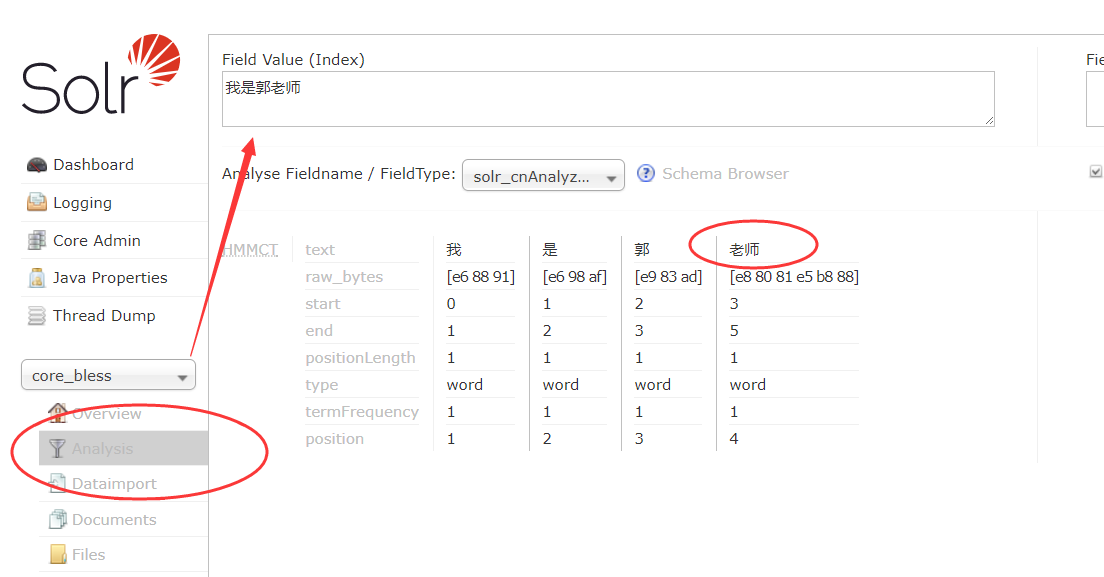


### 第五步：启动服务

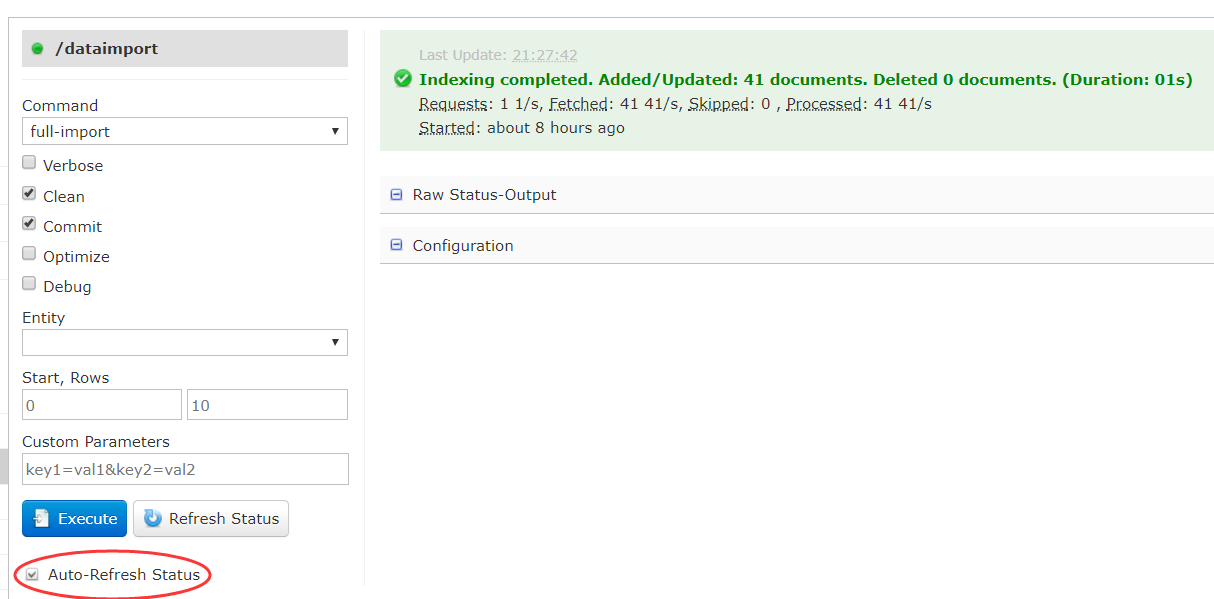
1、启动Tomcat服务器

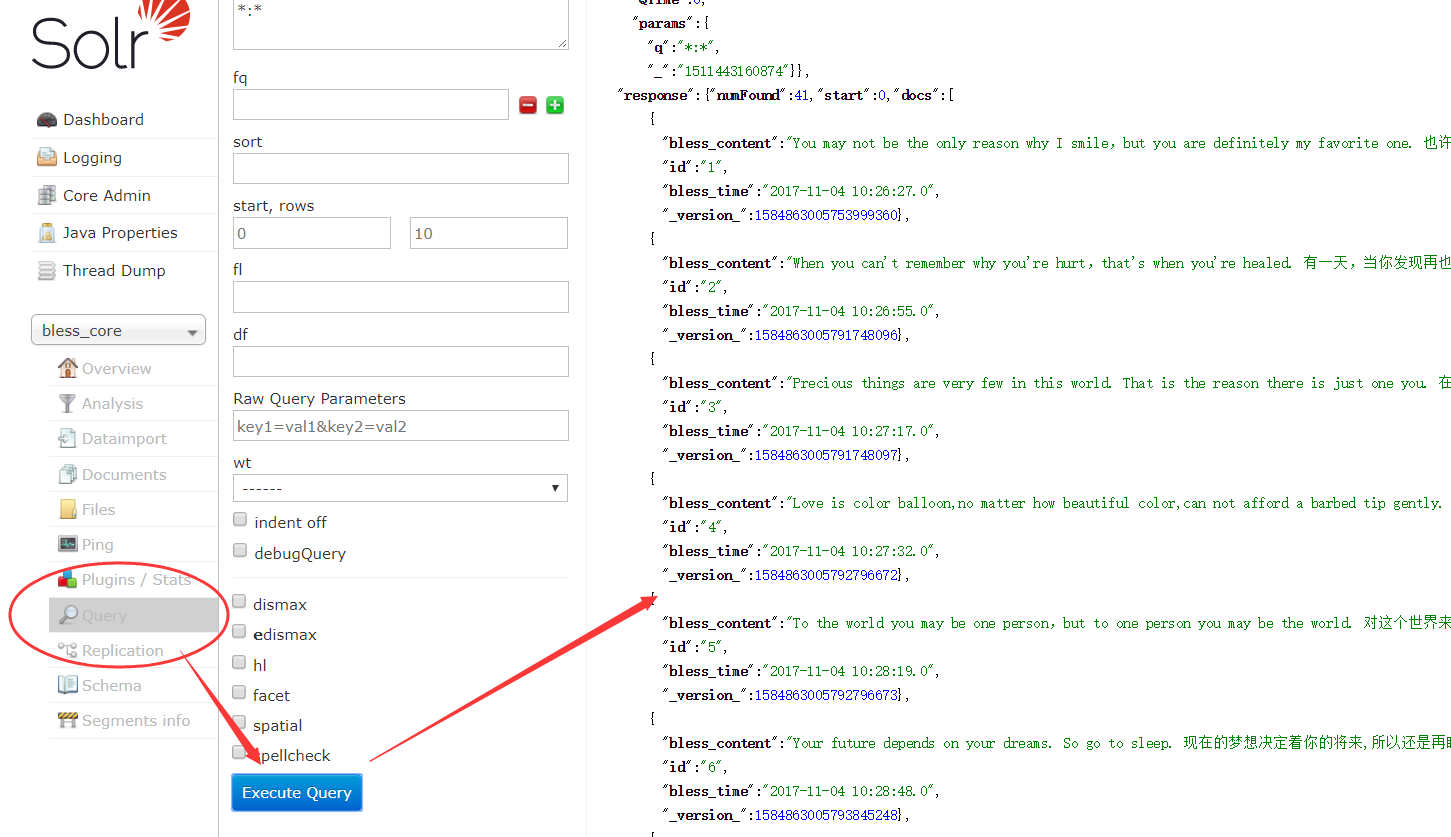
**2、查看数据导入结果：**

**重启Tomcat服务器，进入 guo\_core ,测试中文分词器效果：**



**重启Tomcat服务器，进入 guo\_core 中 DataImport 执行 execute**





**查询描述**

　　q  查询的关键字，此参数最为重要，例如，q=id:1，默认为q=\*:\*，

　　fl  指定返回（回显）哪些字段，用逗号或空格分隔，注意：字段区分大小写，例如，fl= id,title,sort

　　start  返回结果的第几条记录开始，一般分页用，默认0开始

　　rows  指定返回结果最多有多少条记录，默认值为 10，配合start实现分页

　　sort  排序方式，例如id  desc 表示按照 “id” 降序

　　wt  (writer type)指定输出格式，有 xml, json等

　　fq  （filter query）过虑查询，提供一个可选的筛选器查询。返回在q查询符合结果中同时符合的fq条件的查询结果，例如：q=id:1&fq=sort:[1 TO 5]，找关键字id为1 的，并且sort是1到5之间的。

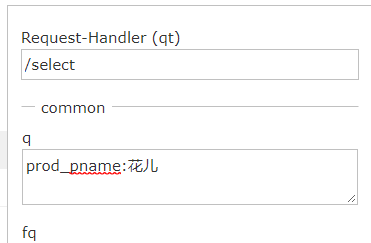
　　df   默认的查询字段，一般默认指定。

　　h1  是否高亮，

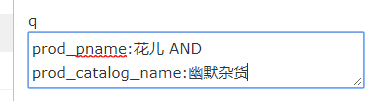
## 客户端查询语法

1. q查询关键字，查询所有使用\*:\*。

请求的q是字符串

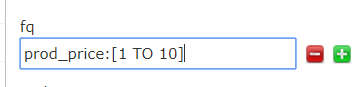


多个条件可以：之间用 AND 或 OR 关联



1. fq （filter query）过滤查询，作用：在q查询符合结果中同时是fq查询符合的，例如：

请求fq是一个数组（多个值）

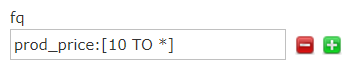


过滤查询价格从1到1的记录。

也可以在“q”查询条件中使用product\_price:[1 TO 20]，如下：



也可以使用“\*”表示无限，例如：

10以上：

10以下：

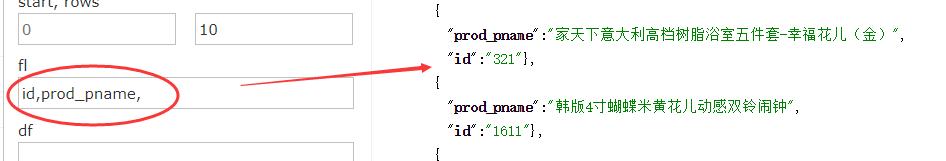
1. sort 排序，格式：sort=<field name>+<desc|asc>[,<field name>+<desc|asc>]…. 示例：

按价格降序

1. start - 分页显示使用，开始记录下标，从0开始
2. rows - 指定返回结果最多有多少条记录，配合start来实现分页。

实际开发时，知道当前页码和每页显示的个数最后求出开始下标。

1. fl - 指定返回那些字段内容，用逗号或空格分隔多个(回显)



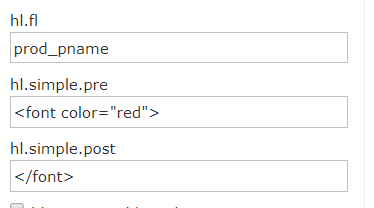
1. df-指定一个搜索默认Field

🡪

1. wt - (writer type)指定输出格式，



1. hl 是否高亮 ,设置高亮Field，设置格式前缀和后缀。



hl.fl 指定高亮域的名称