# solr开发指南

## 1.1 solr简介

### 1.1.1官网介绍

Solr是一个基于Lucene的Java搜索引擎服务器。Solr 提供了层面搜索、命中醒目显示并且支持多种输出格式（包括 XML/XSLT 和 JSON 格式）。它易于安装和配置，而且附带了一个基于 HTTP 的管理界面。Solr已经在众多大型的网站中使用，较为成熟和稳定。Solr 包装并扩展了 Lucene，所以Solr的基本上沿用了Lucene的相关术语。更重要的是，Solr 创建的索引与 Lucene 搜索引擎库完全兼容。通过对 Solr 进行适当的配置，某些情况下可能需要进行编码，Solr 可以阅读和使用构建到其他 Lucene 应用程序中的索引。此外，很多 Lucene 工具（如Nutch、 Luke）也可以使用 Solr 创建的索引。

**总结一下**：solr是一个java搜索引擎服务器（是一套war程序），内部集成了Lucene(apache提供的一些对搜索引擎做支持的jar包)。

### 1.2.2 solr功能

* 保存数据
* 建立索引，维护索引
* 数据检索(全文搜索，高亮显示，精确搜索等)

### 1.2.3 solr依赖环境

* Jdk 1.7+
* TOMCAT 7+
* 课程选用版本：solr4.9.1

## 1.2 solr服务器搭建

### 1.2.1初始配置solr

截止到文档编写前，solr目前的最新版本为6.5.1，在本次项目开发中，我们选用solr比较成熟稳定的版本solr 4.9.1。

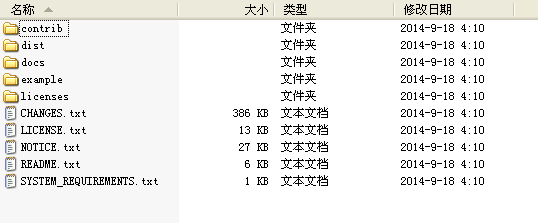
1. 官网下载solr4.9.1的程序安装包。

下载地址：<http://archive.apache.org/dist/lucene/solr/>

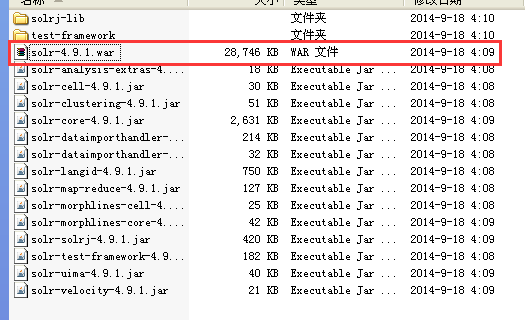


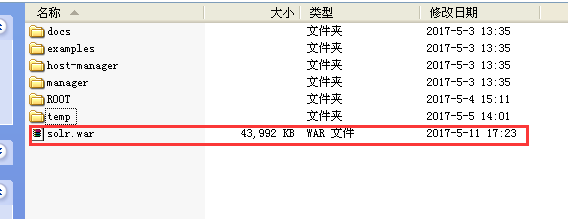


1. 解压solr的zip包，目录如下

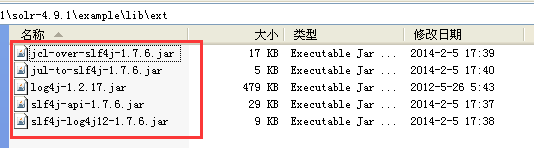


1. 将dist\solr-4.9.1.war文件复制到tomcat的webapps目录下，并将文件命名为solr.war

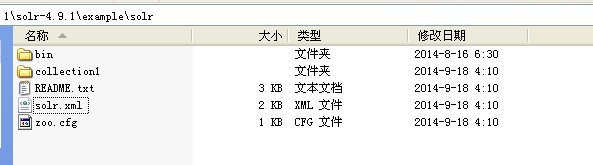


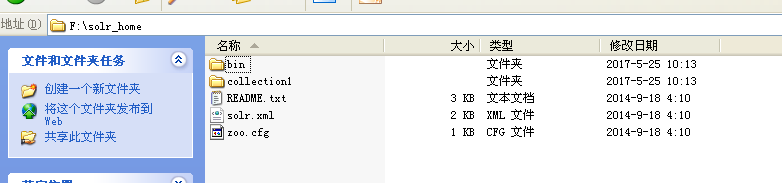


1. 复制 solr解压包下example\lib\ext 下所有的jar 到tomcat 的lib目录下



1. 在计算机本地新建一个文件夹solr\_home（当然你可以随便起名字）, 然后复制solr-4.9.1\example\solr 下的所有文件到 solr\_home下

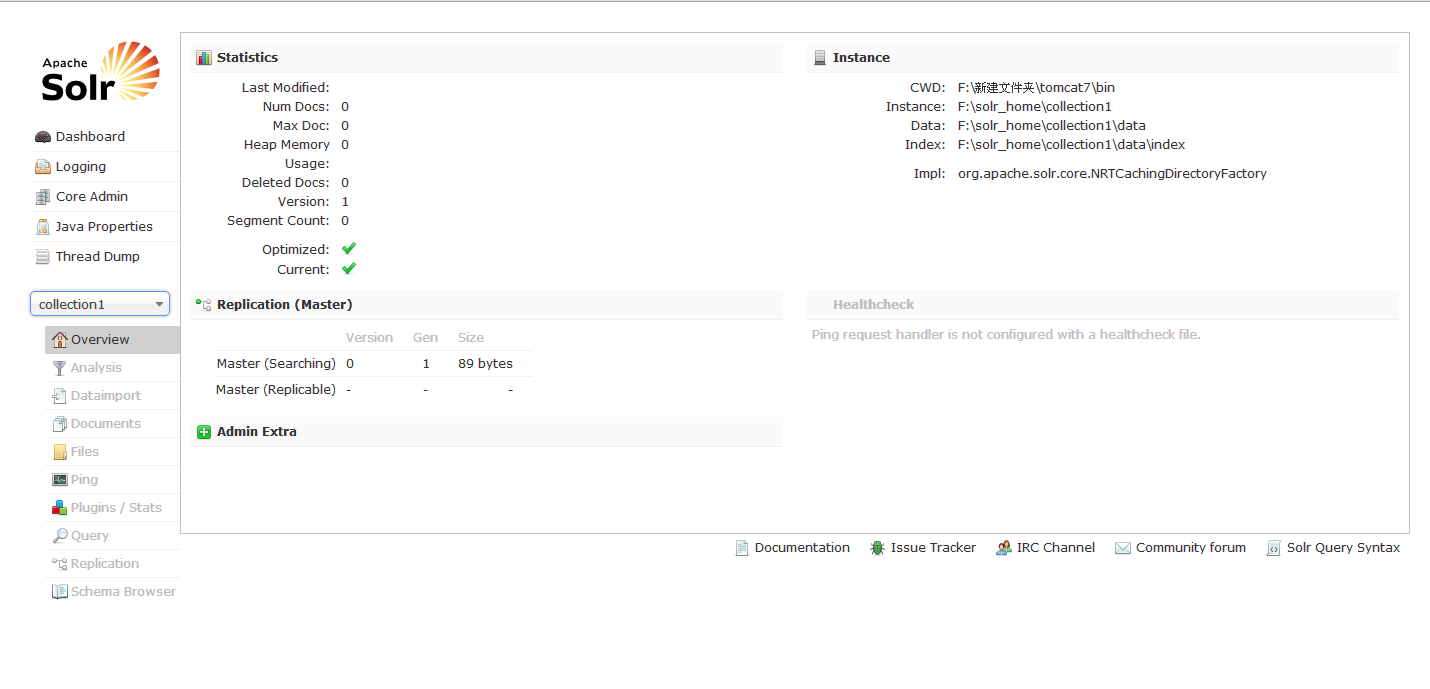




1. 启动tomcat，待tomcat启动成功后，关闭tomcat。打开tomcat的webapps目录。注意，此时solr的war包以及被解压成solr文件夹。删除tomcat 的webapps目录下的solr的war包，保留solr文件夹。
2. 修改配置文件 apache-tomcat-7.0.67\webapps\solr\WEB-INF\web.xml

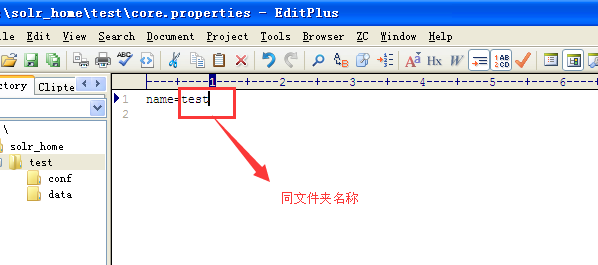
|  |
| --- |
| <env-entry>  <env-entry-name>solr/home</env-entry-name>  <env-entry-value> F:/solr\_home</env-entry-value>  <env-entry-type>java.lang.String</env-entry-type>  </env-entry> |

1. 访问solr(<http://localhost>:端口号/solr/),如出现以下界面则solr部署成功。

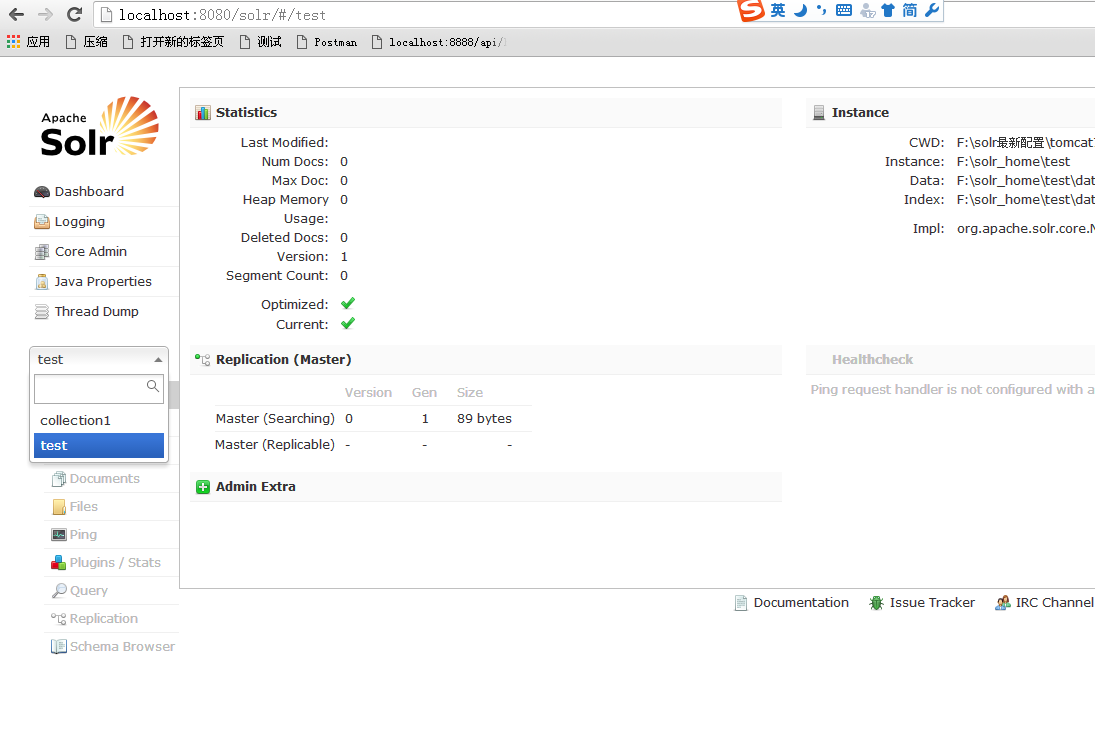


### 1.2.2新建数据配置core

1. 新建core(solr中把配置的每一个模块都叫core),在solr\_home目录下，拷贝collection1文件夹，并起名为test。打开test文件夹，修改core.properties文件,将name修改为test



1. 重新启动tomcat，并访问solr,如出现以下界面,则表示新建test core成功。



1. 重新启动tomcat，并访问solr,如出现以下界面,则表示新建test core成功。

### 1.2.3新增数据库配置

到目前为止，我们已经完成了solr的基础配置，并且创建了test core，接下来我们需要把数据的数据和搜索引擎连接起来，让搜索引擎可以读取数据库的数据。

1. 拷贝数据库连接jar包(mysql-connector-java-5.1.18.jar)到tomcat的lib目录
2. 以创建test core的方式新建hotel core
3. 打开hotel的conf文件夹中的solrconfig.xml文件，在requestHandler name="/select" class="solr.SearchHandler">前面上加上一个dataimport的处理的Handler

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

1. 在hotel的的conf文件夹下并新建data-config.xml文件，配置如下

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>  <dataConfig>  <dataSource type="JdbcDataSource" driver="com.mysql.jdbc.Driver" url="jdbc:mysql://127.0.0.1:3306/itripdb" user="root" password="root" />  <document name="hotel\_doc">  <entity name="hotel" pk="id" query=" select id,hotelName,address from itrip\_hotel">  <field column="id" name="id"/>  <field column="hotelName" name="hotelName"/>  <field column="address" name="address"/>  </entity>  </document>  </dataConfig> |

* dataSource是数据库数据源。
* Entity就是一张表对应的实体，pk是主键，query是查询语句。
* Field对应一个字段，column是数据库里的column名，后面的name属性对应着Solr的Filed的名字。

1. 打开hotel的conf目录下的schema.xml文件

（1）保留\_version\_ 这个field

（2）添加索引字段：这里每个field的name要和data-config.xml里的entity的field的name一样，一一对应。

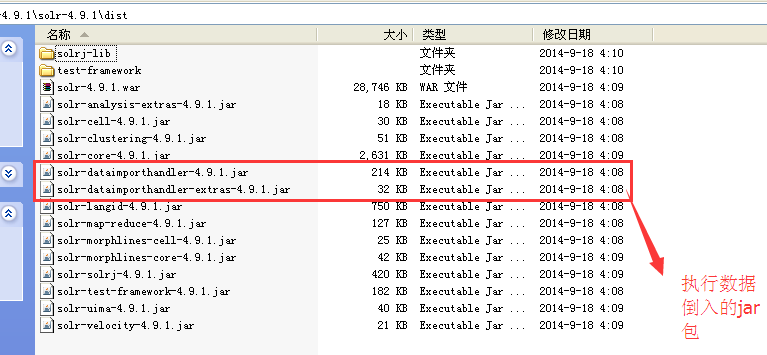
|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>  <dataConfig>  <dataSource type="JdbcDataSource" driver="com.mysql.jdbc.Driver" url="jdbc:mysql://127.0.0.1:3306/itripdb" user="root" password="root" />  <document name="hotel\_doc">  <entity name="hotel" pk="id" query=" select id,hotelName,address from itrip\_hotel">  <field column="id" name="id"/>  <field column="hotelName" name=" hotelName"/>  <field column="address" name=" address"/>  </entity>  </document>  </dataConfig> |

修改同目录下的schema.xml(schema.xml 是solr对数据库里的数据进行索引管理和数据字段展示管理的配置文件)

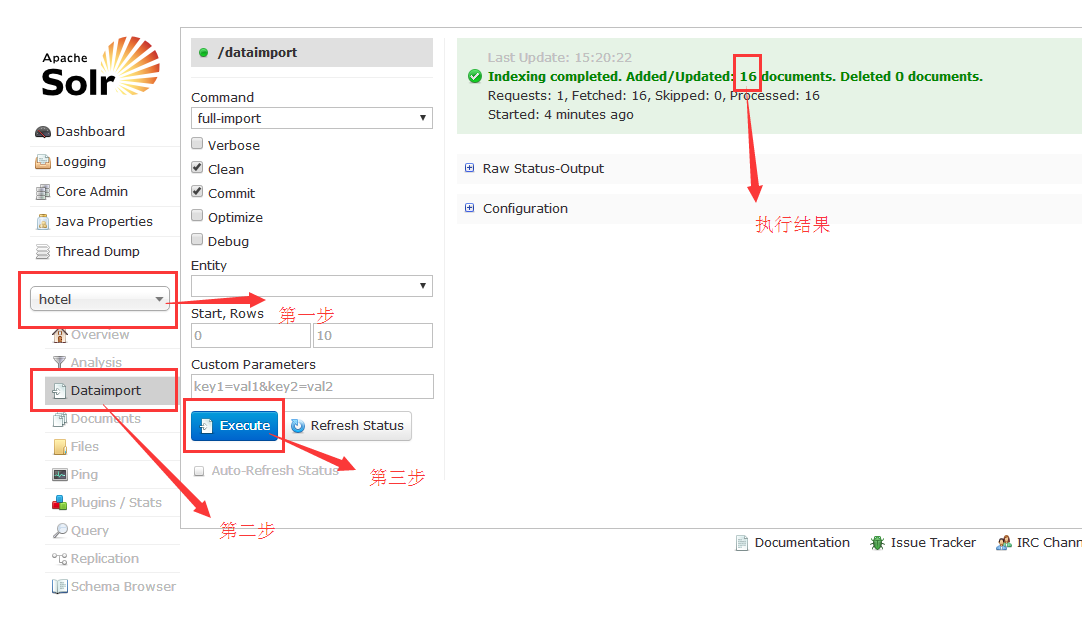
* 删除多余的field，保留\_version\_ 和test这两个field（注意不要删除fieldType）
* 添加索引字段：这里每个field的name要和data-config.xml里的entity的field的name一样，一一对应。红色加粗部分为新增内容。

|  |
| --- |
| <field name="\_version\_" type="long" indexed="true" stored="true"/>  **<field name="id" type="string" indexed="true" stored="true"/>**  **<field name="hotelName" type="string" indexed="true" stored="true"/>**  **<field name="address" type="string" indexed="true" stored="true"/>**  <uniqueKey>id</uniqueKey>  <field name="text" type="text\_general" indexed="true" stored="false" multiValued="true"/> |

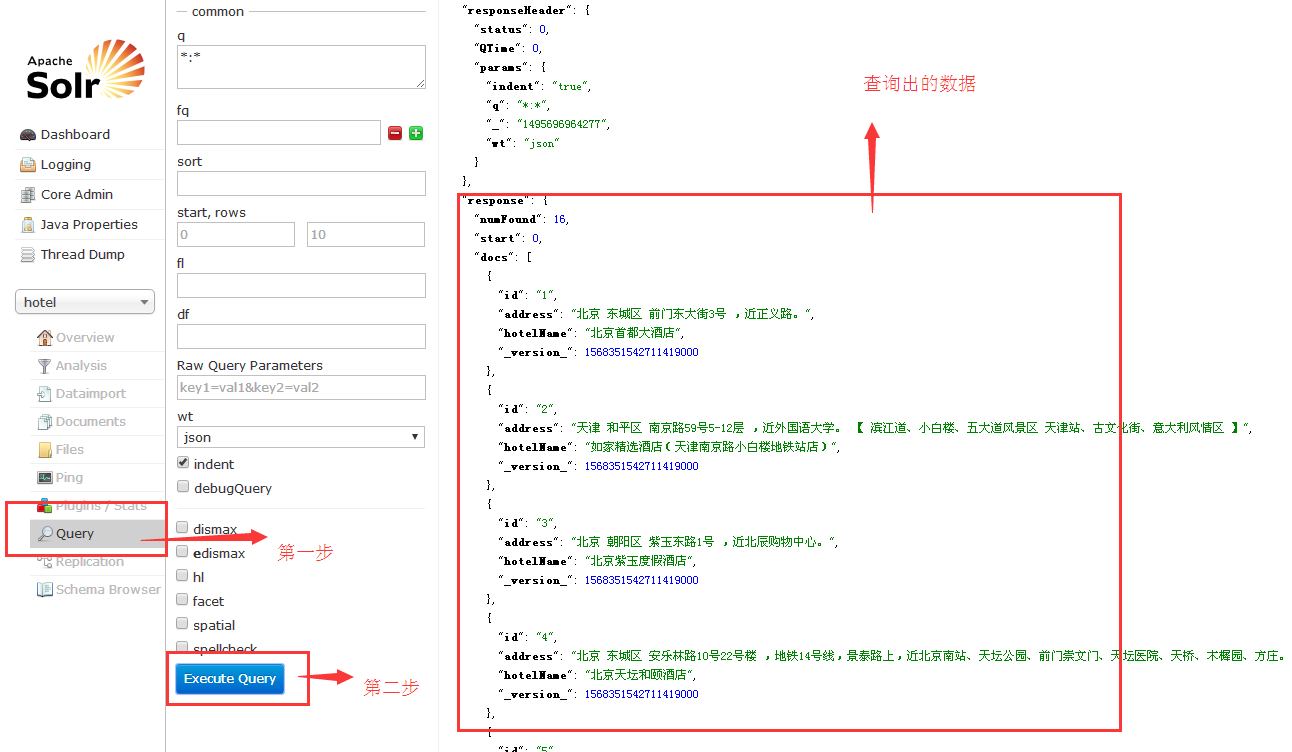
1. 将导入数据的JAR包拷贝到webapps/solr的lib目录下



1. 启动Tomcat，执行数据导入。



1. 查询数据



到目前为止，我们已经将数据库的数据导入到solr当中了，并且已经查询成功。

### 1.2.4配置增量更新

以上的步骤我们实现了如何将数据库的数据导入到solr中，接下来需要配置solr的增量更新，即定时将数据库的数据导入到solr中。

1. 将资料中提供的apache-solr-dataimports-cheduler.jar包添加至solr的lib目录下

**注：apachesolrdataimportscheduler.jar的jar包是apache提供的用于增量更新的jar包，但apache提供的原jar包中，代码有BUG。该bug在教学资料提供的jar包中已经被修复，具体可参考提供的apachesolrdataimportscheduler的源码，在此不再赘述。**

1. 将资料中提供的apache-solr-dataimports-cheduler.jar包添加至solr的lib目录下。
2. 增加增量更新配置文件,在solr\_home文件夹下新建conf文件夹，并新建名为dataimport.properties的配置文件，配置如下,标红的地方为需要修改的地方

|  |
| --- |
| #################################################  # #  # dataimport scheduler properties #  # #  #################################################  # to sync or not to sync  # 1 - active; anything else - inactive  **syncEnabled=1**  # which cores to schedule  # in a multi-core environment you can decide which cores you want syncronized  # leave empty or comment it out if using single-core deployment  **syncCores=test,hotel**  # solr server name or IP address  # [defaults to localhost if empty]  **server=localhost**  # solr server port  # [defaults to 80 if empty]  **port=8080**  # application name/context  # [defaults to current ServletContextListener's context (app) name]  **webapp=solr**  # 增量索引的参数  # URL params [mandatory]  # remainder of URL  params=/dataimport?command=delta-import&clean=false&commit=true  # 重做增量索引的时间间隔  # schedule interval  # number of minutes between two runs  # [defaults to 30 if empty]  **interval=1**  # 重做全量索引的时间间隔，单位分钟，默认7200，即5天;  # 为空,为0,或者注释掉:表示永不重做索引  #reBuildIndexInterval=7200  # 重做索引的参数  reBuildIndexParams=/dataimport?command=full-import&clean=true&commit=true  # 重做索引时间间隔的计时开始时间，第一次真正执行的时间=reBuildIndexBeginTime+reBuildIndexInterval\*60\*1000；  # 两种格式：2012-04-11 03:10:00 或者 03:10:00，后一种会自动补全日期部分为服务启动时的日期  reBuildIndexBeginTime=03:10:00 |

1. 新增增量更新数据的监听器，在solr的web.xml中加入以下监听器

|  |
| --- |
| <listener>  <listener-class>  org.apache.solr.handler.dataimport.scheduler.ApplicationListener  </listener-class>  </listener> |

1. 修改导入数据查询SQL

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>  <dataConfig>  <dataSource type="JdbcDataSource" driver="com.mysql.jdbc.Driver" url="jdbc:mysql://127.0.0.1:3306/itripdb" user="root" password="root" />  <document name="hotel\_doc">  <entity name="hotel" pk="id" query="select id,hotelName,address from itrip\_hotel"  **deltaImportQuery="select id,hotelName,address from itrip\_hotel where id ='${dih.delta.id}'"**  **deltaQuery="SELECT id as id FROM itrip\_hotel where modifyDate > '${dih.last\_index\_time}'">**  <field column="id" name="id"/>  <field column="hotelName" name="hotelName"/>  <field column="address" name="address"/>  </entity>  </document>  </dataConfig> |

说明：deltaQuery是根据dataimport.properties配置文件中的更新时间，从数据库中查询出，修改日期在最后一次更新日期之后的酒店数据，并记录其id，而deltaImportQuery的目的是将deltaQuery查询出的数据导入到solr中。

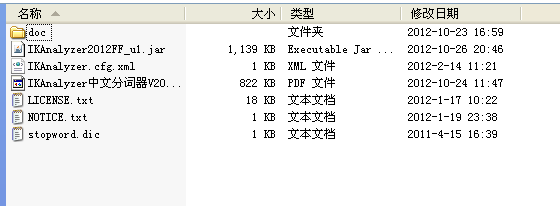
1. 启动Tomcat进行测试

* 启动Tomcat，访问hotel模块
* 修改数据库中的酒店数据并同时修改该数据的modifyDate时间,
* 1分钟后查询酒店数据，确定数据是否更新

### 1.2.5配置分词器

* 分词器: 是从用户输入的一段文本中提取关键词，用于其它业务操作。
* 常见的JAVA分词器: word分词器、Ansj分词器、Stanford分词器、IKAnalyzer分词器
* 课程选用分词器：IKAnalyzer分词器
* solr如果是3.x版本的用IKAnalyzer2012\_u6.zip如果是4.x版本的用IK Analyzer 2012FF\_hf1.zip，一定要对应上，要不然会配置失败。
* IK分词器下载地址: <http://download.csdn.net/download/tjcyjd/8420639>

1. 首先，下载IKAnalyzer 。
2. 将ik的所有jar文件 拷贝到 webapps\solr\WEB-INF\lib 目录下



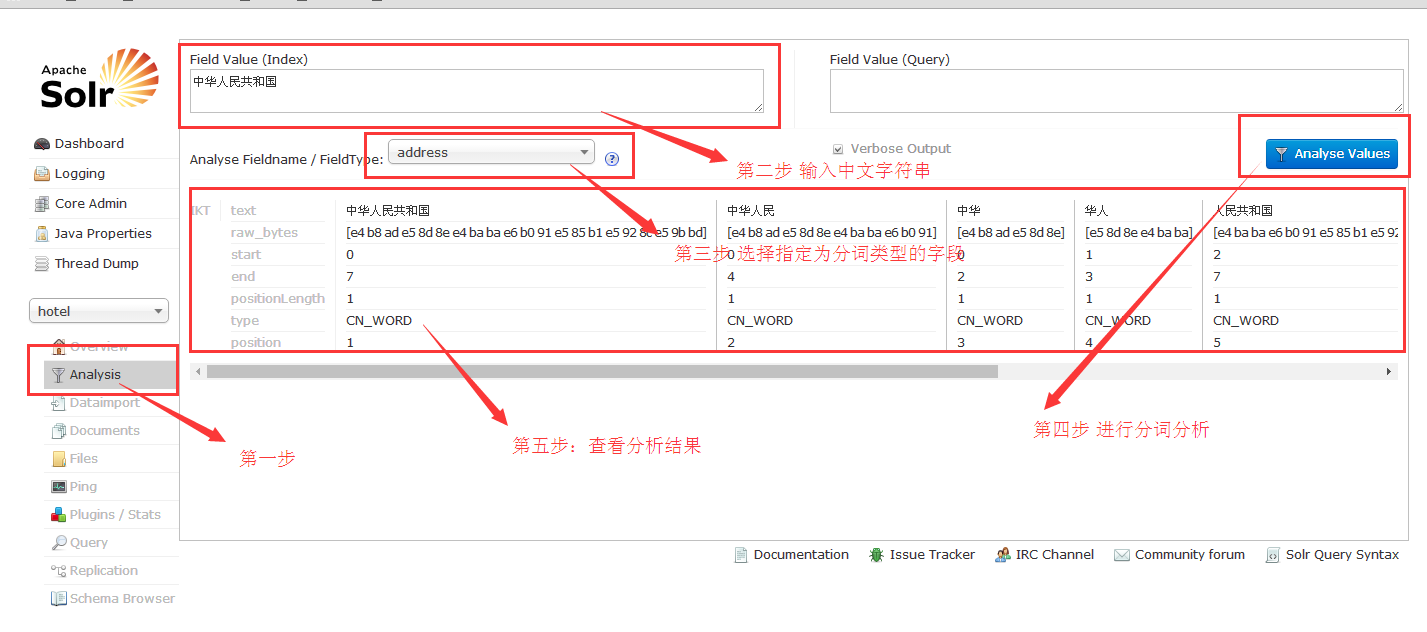
1. 在webapps\solr\WEB-INF\下新建classes文件夹，将IKAnalyzer.cfg.xml和stopword.dic文件拷贝到改文件夹下。
2. 在 solr\_home\hotel\conf\schema.xml 增加如下配置

|  |
| --- |
| <fieldType name="text\_ik" class="solr.TextField">  <analyzer type="index" isMaxWordLength="false" class="org.wltea.analyzer.lucene.IKAnalyzer"/>  <analyzer type="query" isMaxWordLength="true" class="org.wltea.analyzer.lucene.IKAnalyzer"/>  </fieldType> |

1. 修改solr\_home\hotel\conf\schema.xml将hotelName和address指定成为text\_ik类型

|  |
| --- |
| <field name="hotelName" type="text\_ik" indexed="true" stored="true"/>  <field name="address" type="text\_ik" indexed="true" stored="true"/> |

1. 重启Tomcat，访问solr测试分词器



1. 如果分词器出现上边的显示结果，则表示分词器配置成功。

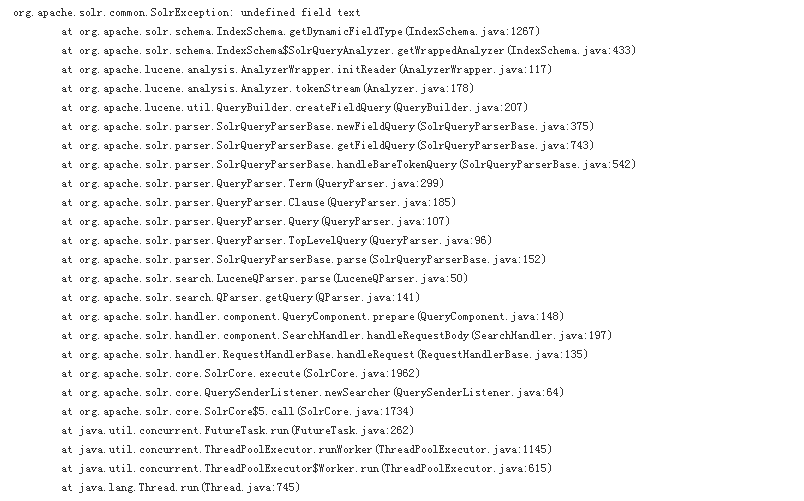
### 1.2.6上传服务器

在本地配置好solr和solr\_home后，可以将装载有solr的tomcat和solr\_home文件夹直接拷贝到Linux服务器。此处注意要修改solr中的web.xml的solr\_home的地址修改为solr\_home在服务器的实际目录。

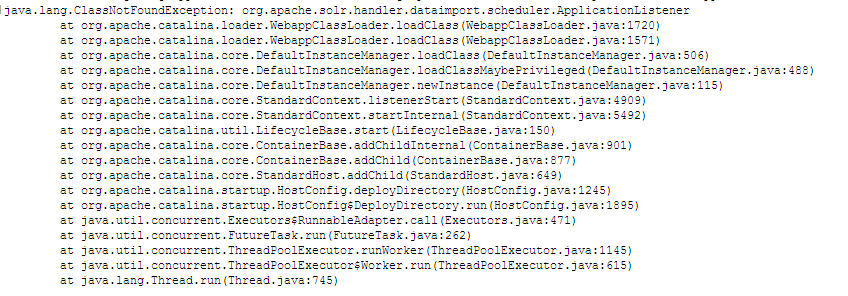
详细部署文档请参考《itrip安装配置及备份方案》。

## 1.4常见异常

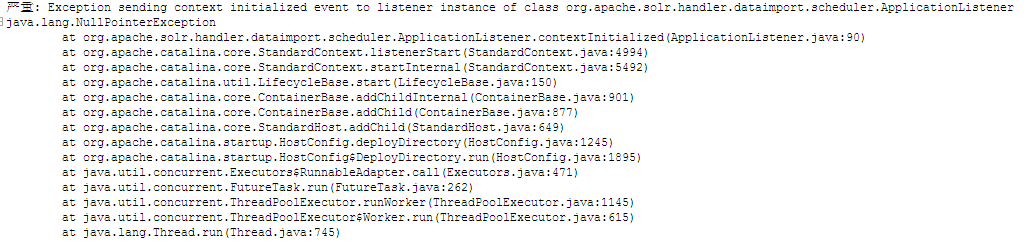
### 1.4.1删除了默认字段(text或者\_version\_)



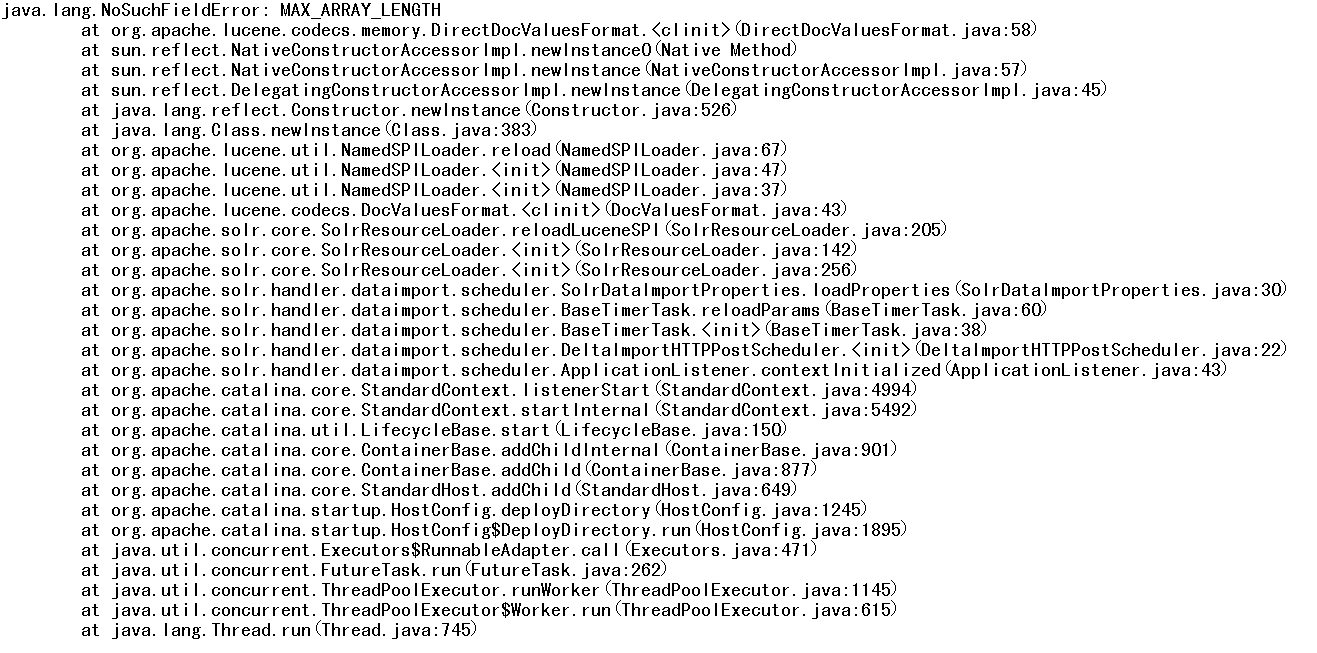
### 1.4.2没有加入增量更新的jar包



### 1.4.3没有加入增量更新的配置文件

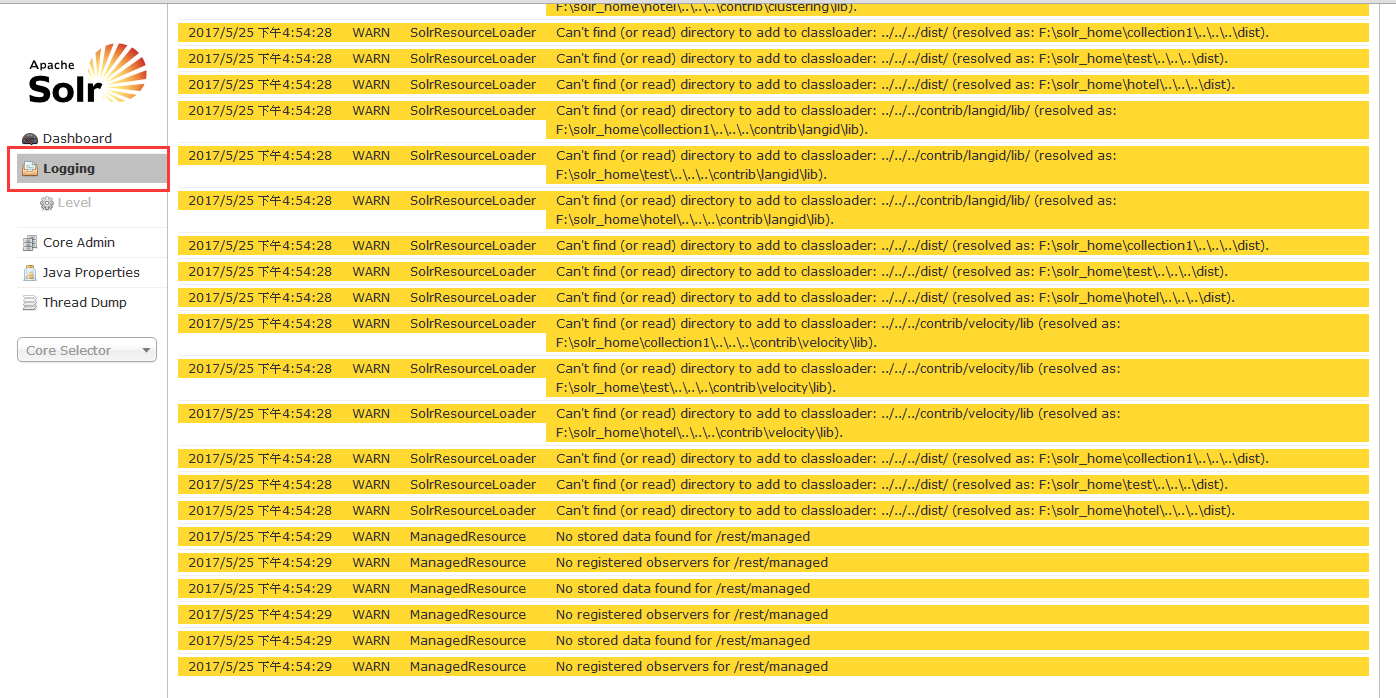


### 1.4.4 solr long字段报错



## 1.5调试技巧

### 1.5.1solr日志



### 1.5.2 tomcat日志

