将 MySQL 的全量数据导入到 Apache Solr 中

关于分页方式导入全量数据请参照《将 MySQL 的全量数据以分页的形式导入到 Apache Solr 中》。

在前面几篇文章中我们介绍了如何通过 Solr 的 post 命令将各种各样的文件导入到已经创建好的 Core 或 Collection 中。但有时候我们需要的数据并不在文件里面，而是在别的系统中，比如 MySql 里面。不过高兴的是，Solr 针对这些数据也提供了强大的数据导入工具，这就是 DataImportHandler。

DataImportHandler 是 Solr 中最重要的工具之一，利用它我们可以轻易地将数据从数据库、XML 文件以及 HTTP 数据源导入 Solr 中，同时支持全量和增量的数据导入。DataImportHandler 的实现在 solr-dataimporthandler-x.x.x.jar 和 solr-dataimporthandler-extras-x.x.x.jar 两个文件中。下面我们将利用这个工具从 MySQL 导数据到 Solr 中。

其实在 Solr 的安装包里面提供了这个工具的使用例子（$SORE_HOME/example/example-DIH），我们可以通过 solr -e dih 命令来加载这个例。不过基于学习的心态，我不打算直接使用这个例子下的文件，而是从头开始创建。下面跟我一步一步地查看如何实现。

下载相关依赖

在进行相关设置之前，我们准备好运行环境，因为我们这用到了 MySQL，自然需要准备好 MySQL 相关驱动。我们使用下面的命令下载这个驱动，并将它移到 $SOLR_HOME/dist 目录下：

```
wget http://central.maven.org/maven2/mysql/mysql-connector-java/8.0.8-dmr/mysql-connector-java-8.0.8-dmr.jar
mv ~/mysql-connector-java-8.0.8-dmr.jar $SOLR_HOME/dist
```

准备配置文件并创建 Core

MySQL 驱动包准备好之后，我们需要准备配置相关文件了。Solr 安装包默认都自带了名为 _default 的配置，对于的配置文件夹是 $SORE_HOME/server/solr/configsets/_default ，我们直接从这个文件夹复制一份新的配置：

```
cp -R _default iteblog
```

然后到 $SORE_HOME/server/solr/configsets/iteblog/conf/solrconfig.xml 里面添加 MySQL

驱动的路径：

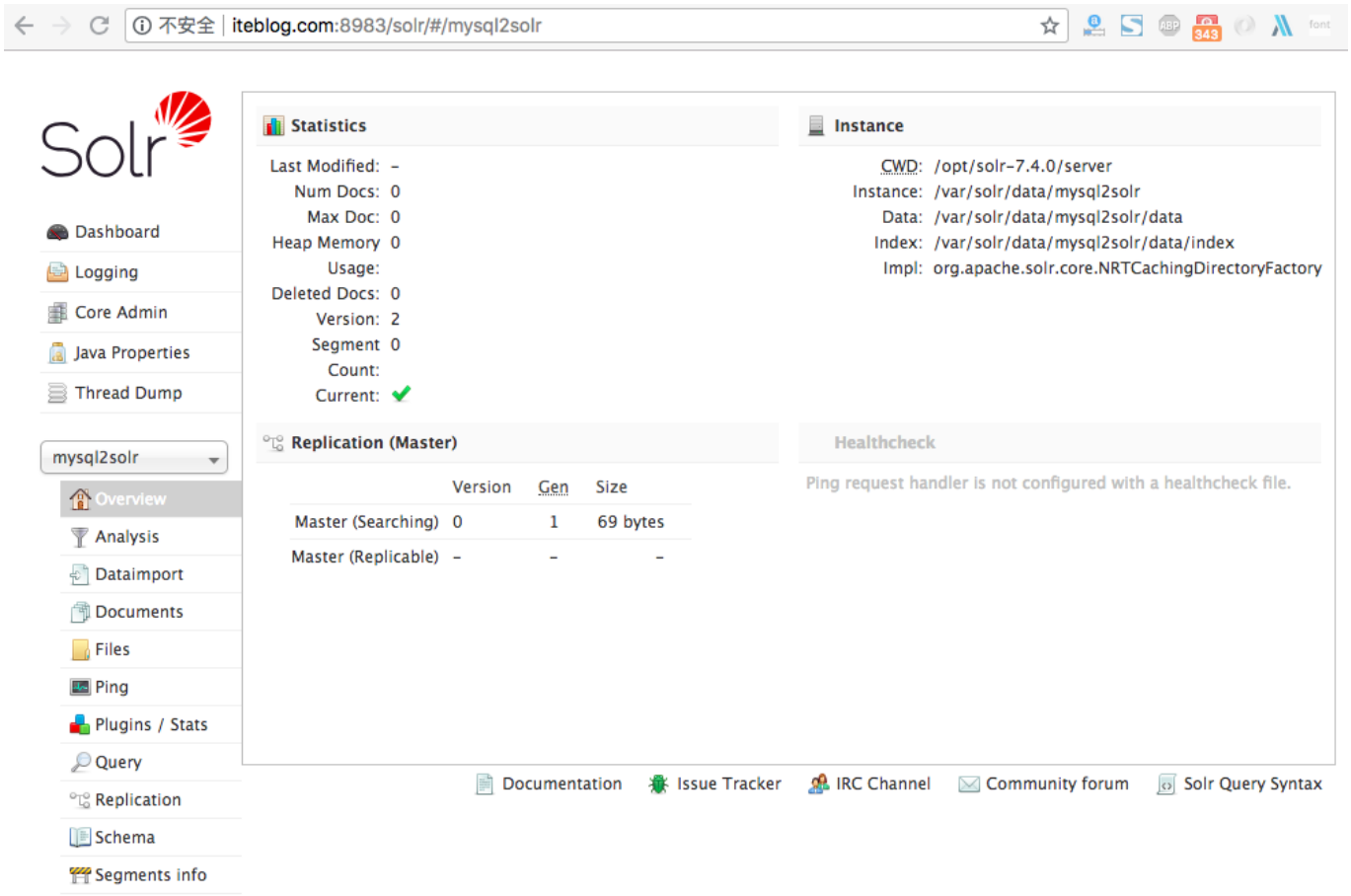
<lib dir="${solr.install.dir:../../../..}/dist/" regex="mysql-connector-java-₩d.*₩.jar" />

配置文件准备好之后，我们就可以创建 Core/Collection 了，命令如下：

[root@iteblog.com /opt/solr]$ bin/solr create -c mysql2solr -d /opt/solr/server/solr/configsets/iteblog/
INFO - 2018-08-06 22:54:42.684; org.apache.solr.util.configuration.SSLCredentialProviderFactory; Processing SSL Credential Provider chain: env;sysprop

Created new core 'mysql2solr'

注意，一定要注意加上 -d /opt/solr/server/solr/configsets/iteblog/ 参数。创建完之后，我们就可以在 Admin UI 界面看到我们创建好的 Core：



如果想及时了

解Spark、Hadoop或者Hbase相关的文章，欢迎关注微信公共帐号：iteblog_hadoop

设置DataImportHandler

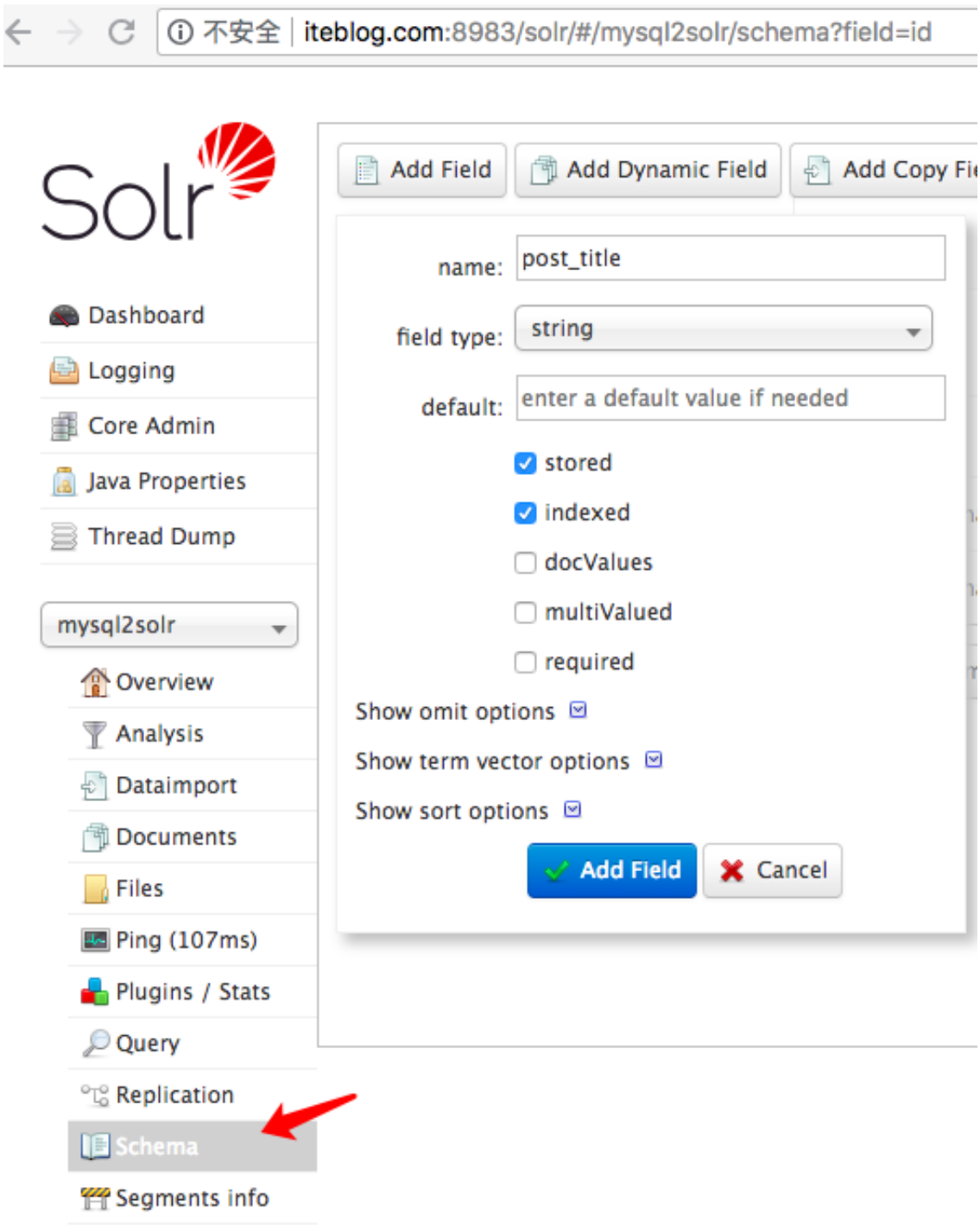
按照上面步骤创建的 Core 是无法从 MySQL 导数到 Solr 的，我们可以到 http://iteblog.com:8983/solr/#/mysql2solr/dataimport/ 路径下显示的页面看到 The solrconfig.xml file for this index does not have an operational DataImportHandler defined! 相关错误提示，因为我们没有定义 DataImportHandler。到 mysql2solr 的 solrconfig.xml 配置文件里面添加如下内容：

```
<requestHandler name="/dataimport" class="solr.DataImportHandler">
   <lst name="defaults">
    <str name="config">db-data-config.xml</str>
   </lst>
</requestHandler>
```

并在 solrconfig.xml 配置文件同一目录下创建名为 db-data-config.xml 的配置文件，在这个文件里面添加如下内容：

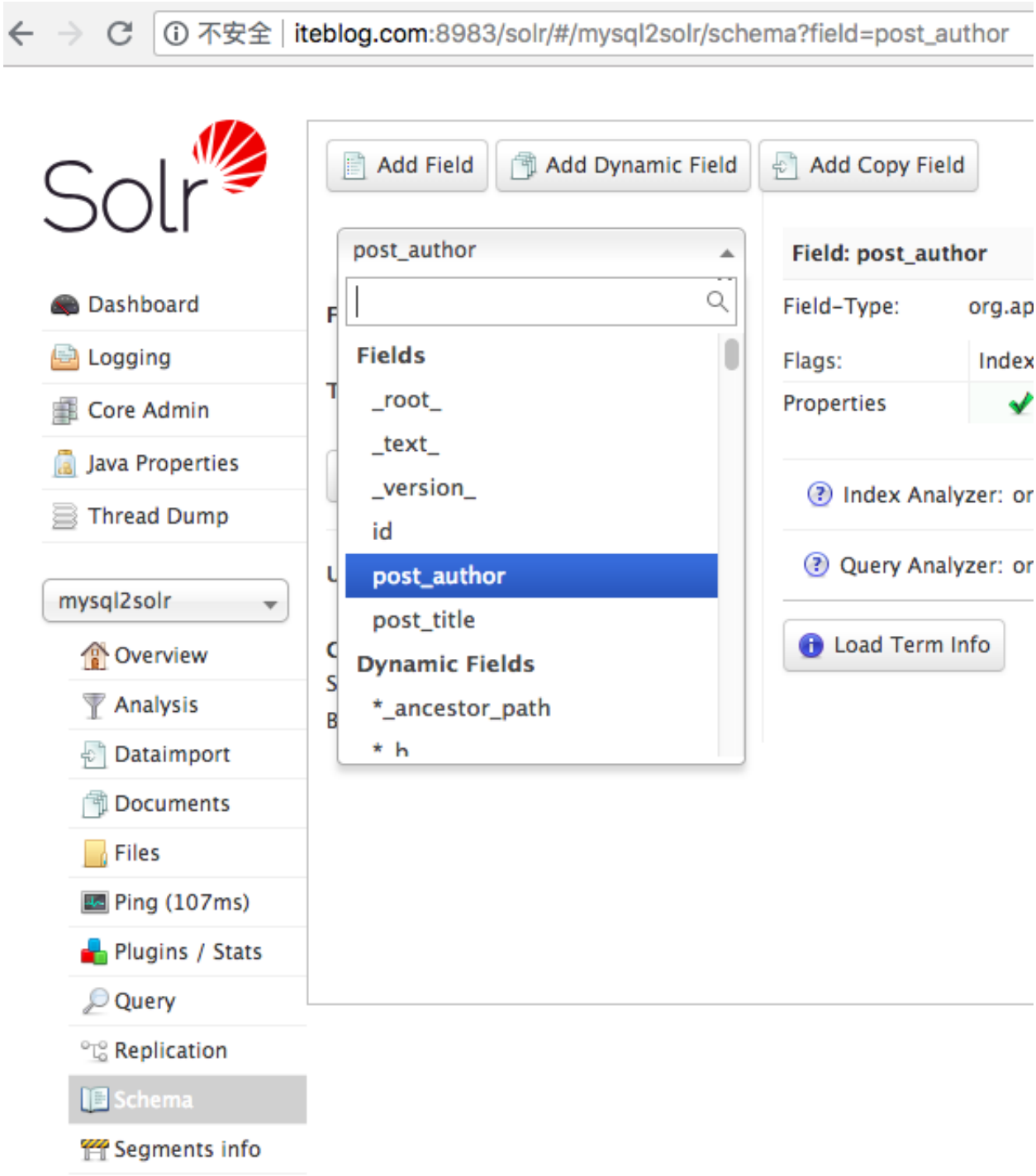
```
<dataConfig>
   <dataSource driver="com.mysql.jdbc.Driver"
       url="jdbc:mysql://localhost:3306/iteblog" user="root" password="xxx" />
      <document>
      <entity name="iteblog" query="select id,post_title,post_author from wp_posts">
   <field column="id" name="id" />
   <field column="post_title" name="post_title" />
   <field column="post_author" name="post_author" />
  </entity>
 </document>
</dataConfig>
```

这里将我博客的 iteblog 数据库下面的 wp_posts 表里面的数据导入到 Solr，其中只使用 id,post_title,post_author 三个字段，其含义分别是文章id、文章标题以及文章作者等信息。现在我们需要到 http://iteblog.com:8983/solr/#/mysql2solr/schema 里面添加 id,post_title,post_author 三个字段的模式相关定义，如下：



如果想及时了
解Spark、Hadoop或者Hbase相关的文章，欢迎关注微信公共帐号：iteblog_hadoop

这里以 post_title 为例进行了介绍，其他字段和这个步骤类似，不再赘述。添加完字段以后我们就可以看到如下界面的信息：



如果想及时了
解Spark、Hadoop或者Hbase相关的文章，欢迎关注微信公共帐号：iteblog_hadoop

到这里 id,post_title,post_author 三个字段的模式已经全部定义完毕。

需要注意的是：

- 因为 MySQL 里面正好有一个名为 id 的字段用于标记为唯一键，而 Solr 里面也正好默认存在名为 id 的字段，定义如下：

<field name="id" type="string" multiValued="false" indexed="true" required="true" stored="true"/>

如果需要导的 MySQL 表唯一键不叫 id，比如叫 iteblog_id，那么我们需要定义它，而且还需要将 Solr 里面的 uniqueKey 属性设置为它，如下：

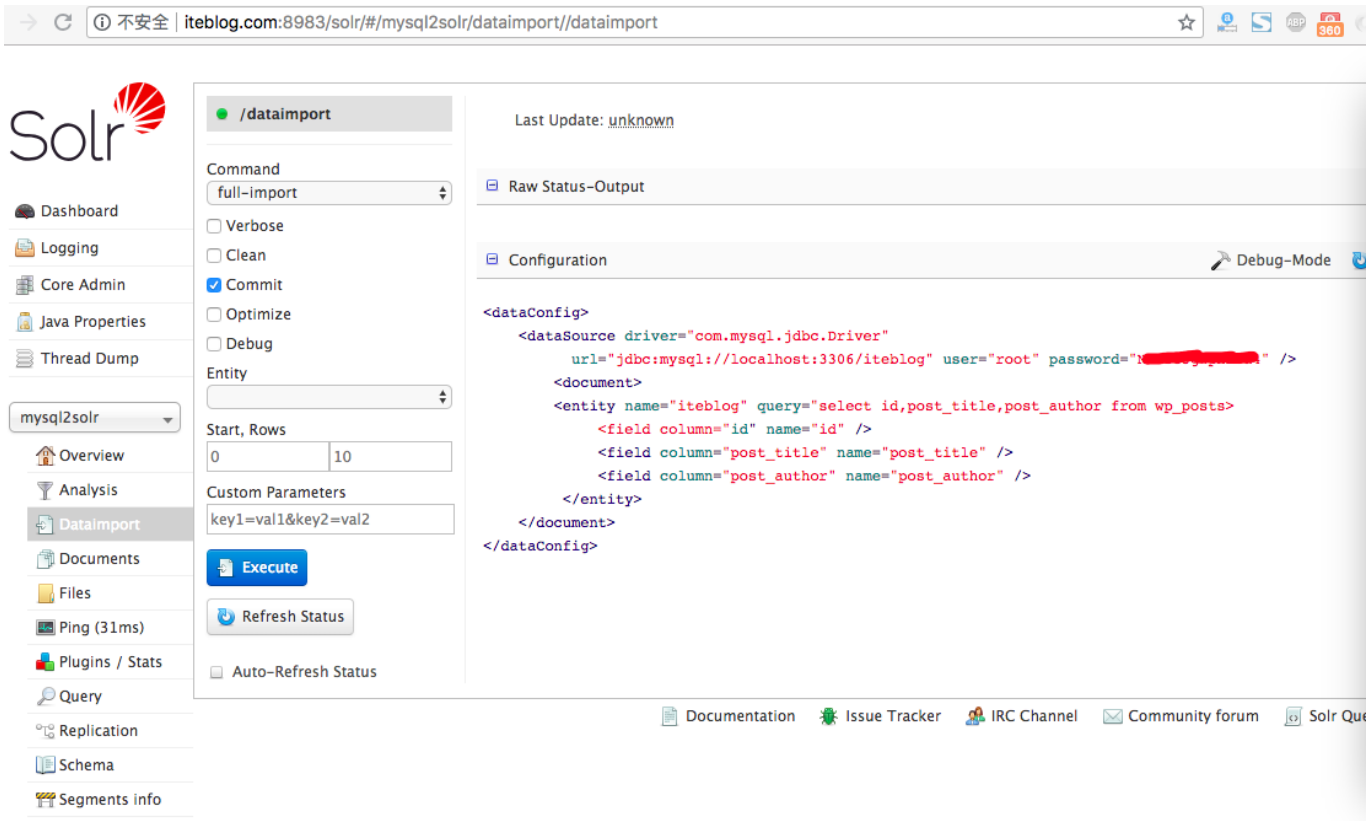
```
<field name="iteblog_id" type="string" multiValued="false"
       indexed="true" required="true" stored="true"/>
<uniqueKey>iteblog_id</uniqueKey>
```

- 其实我们还可以直接到 mysql2solr 的 managed-schema 配置文件添加下面的定义：

```
<field name="post_author" type="string" indexed="true" stored="true"/>
<field name="post_title" type="string" indexed="true" stored="true"/>
```

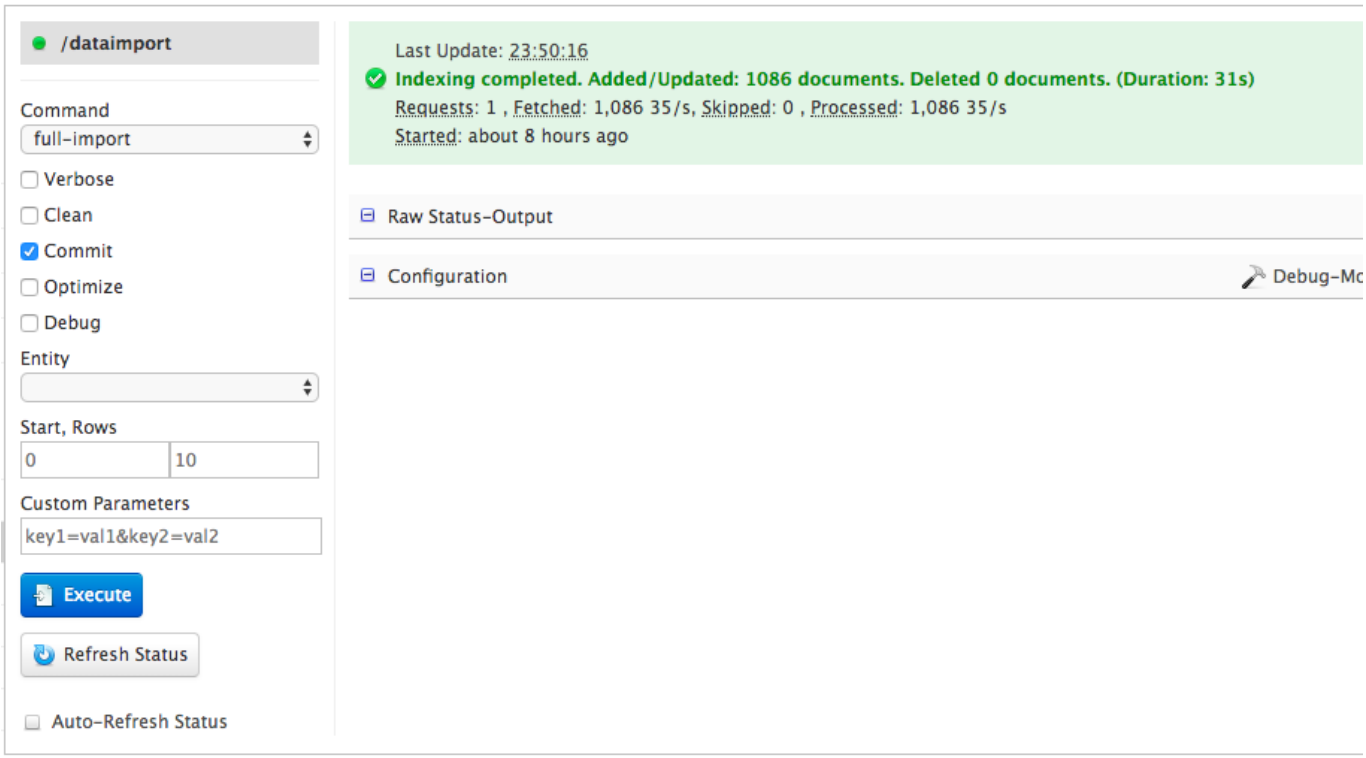
其效果和直接在 Admin UI 界面添加类似。

一切准备好之后，我们需要重启 Solr，然后就可以看到如下界面：



如果想及时了
解Spark、Hadoop或者Hbase相关的文章，欢迎关注微信公共帐号：iteblog_hadoop

好了，现在我们可以点击这个界面里面的 Execute 按钮开始导入 MySQL
的全量数据了，结果如下：



如果想及时了
解Spark、Hadoop或者Hbase相关的文章，欢迎关注微信公共帐号：iteblog_hadoop

如果你看到上面的界面说明导数据成功了，上面界面显示我们一共成功导入了1086篇文章，花费时间为 31s。其实点击 Execute 按钮相当于执行下面的 URL 请求：

http://iteblog.com:8983/solr/mysql2solr/dataimport?core=mysql2solr&indent=on&commit=true&name=dataimport&clean=false&wt=json&command=full-import
输出


```
{
  "responseHeader": {
    "status": 0,
    "QTime": 16
  },
  "initArgs": [
    "defaults",
    [
      "config",
      "db-data-config.xml"
    ]
  ],
  "command": "full-import",
  "status": "idle",
  "importResponse": "",
  "statusMessages": {
```

```
      "Total Requests made to DataSource": "1",
      "Total Rows Fetched": "1086",
      "Total Documents Processed": "1086",
      "Total Documents Skipped": "0",
      "Full Dump Started": "2018-08-06 23:30:00",
      "": "Indexing completed. Added/Updated: 1086 documents. Deleted 0 documents.",
      "Committed": "2018-08-06 23:30:01",
      "Time taken": "0:0:1.79"
   }
}
```


现在我们可以从 Solr 里面查询数据了：


```
curl "http://iteblog.com:8983/solr/mysql2solr/select?q=*:*&rows=2"
{
   "responseHeader": {
      "status": 0,
      "QTime": 5,
      "params": {
         "q": "*:*",
         "rows": "2"
      }
   },
   "response": {
      "numFound": 1086,
      "start": 0,
      "docs": [
         {
            "post_title": "联系我",
            "post_author": "1",
            "id": "2",
            "_version_": 1608065325379616800
         },
         {
            "post_title": "Spark 2.0介绍：Catalog API介绍和使用",
            "post_author": "1",
            "id": "1701",
            "_version_": 1608065330188386300
         }
      ]
   }
}
```

到这里我们已经成功的将 MySQL 里面的数据全量导入 Solr 了。现在有几个问题：

- 如果全量数据很大咋办？一次性导入肯定会出问题，可以分页导入吗？
- 增量数据如何导入？

这些问题我们在后面文章进行介绍吧，很晚了，睡觉去了。