logstash-input-jdbc 安装

1. 安装logstash

cd /opt/soft

tar xvf logstash-5.5.1.tar.gz

运行

./bin/logstash -e 'input { stdin { } } output { stdout {} }'

然后输入

hello world

得到类似的结果

2016-11-24T08:01:55.949Z bogon hello world

安装logstash-input-jdbc插件

1.修改ruby仓库镜像

如果没有安装 gem 的话 安装gem

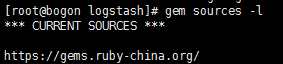
yum install gem

替换国内的镜像

gem sources --add https://gems.ruby-china.org/ --remove https://rubygems.org/

验证是否成功

gem sources -l



修改Gemfile的数据源地址

whereis logstash # 查看logstash安装的位置， 我的在 /usr/share/logstash目录

cd /usr/share/logstash

vim Gemfile

修改 source 的值 为： "https://gems.ruby-china.org/"

vim Gemfile.jruby-1.9.lock # 找到 remote 修改它的值为：

https://gems.ruby-china.org/

或者直接替换源这样你不用改你的 Gemfile 的 source。

gem install bundler

$ bundle config mirror.https://rubygems.org https://gems.ruby-china.org/

然后开始安装

bin/logstash-plugin install logstash-input-jdbc

如果镜像地址没有改错的话应该可以直接安装

或者  进入源码地址的release页面[logstash-input-jdbc](https://github.com/logstash-plugins/logstash-input-jdbc/releases" \t "http://www.cnblogs.com/phpshen/p/_blank)

2.开始同步 mysql 数据

需要建立 两个文件  一个  .conf后缀的 一个 .sql 后缀

 一个 mysql 的[Java](http://lib.csdn.net/base/javaee" \o "Java EE知识库" \t "http://www.cnblogs.com/phpshen/p/_blank) 驱动包  ： mysql-connector-java-5.1.40-bin.jar

filename.conf 内容：

里面的参数可以参考 [logstash-input-jdbc官方参考文档](https://www.elastic.co/guide/en/logstash/current/plugins-inputs-jdbc.html" \t "http://www.cnblogs.com/phpshen/p/_blank)

[IMG_257](http://www.cnblogs.com/phpshen/p/javascript:void(0);)

input {

stdin {

}

jdbc {

# 数据库地址 端口 数据库名

jdbc\_connection\_string => "jdbc:mysql://localhost:3306/shen"

# 数据库用户名

jdbc\_user => "root"

# 数据库密码

jdbc\_password => "rootroot"

# mysql java驱动地址

jdbc\_driver\_library => "/usr/share/logstash/mysql-connector-java-5.1.40-bin.jar"

jdbc\_driver\_class => "com.mysql.jdbc.Driver"

jdbc\_paging\_enabled => "true"

jdbc\_page\_size => "50000"

# sql 语句文件

statement\_filepath => "filename.sql"

schedule => "\* \* \* \* \*"

type => "jdbc"

}

}

output {

stdout {

codec => json\_lines

}

　　　elasticsearch {

hosts => "localhost:9200"

index => "contacts"

　　　　　document\_type => "contact"

document\_id => "%{id}"

}

}

[IMG_258](http://www.cnblogs.com/phpshen/p/javascript:void(0);)

filename.sql

select \* from a

注意： 在你的数据库里 要有一个数据库名字和filename.conf 里的对应就可以了   表明 和 filename.sql 里的对应就可以了   在表中  有一个id字段是为了和filename.conf  中document\_id => "%{id}" 这个参数对应 可以执行修改

然后开始执行

bin/logstash -f fielname.conf

插入多表

input {

jdbc {

jdbc\_driver\_library => "/Users/logstash/mysql-connector-java-5.1.39-bin.jar"

jdbc\_driver\_class => "com.mysql.jdbc.Driver"

jdbc\_connection\_string => "jdbc:mysql://localhost:3306/database\_name"

jdbc\_user => "root"

jdbc\_password => "password"

schedule => "\* \* \* \* \*"

statement => "select \* from table1"

type => "table1"

}

jdbc {

jdbc\_driver\_library => "/Users/logstash/mysql-connector-java-5.1.39-bin.jar"

jdbc\_driver\_class => "com.mysql.jdbc.Driver"

jdbc\_connection\_string => "jdbc:mysql://localhost:3306/database\_name"

jdbc\_user => "root"

jdbc\_password => "password"

schedule => "\* \* \* \* \*"

statement => "select \* from table2"

type => "table2"

}

# add more jdbc inputs to suit your needs

}

output {

elasticsearch {

index => "testdb"

document\_type => "%{type}" # <- use the type from each input

hosts => "localhost:9200"

}

}

2.

[root@5b9dbaaa148a logstash\_jdbc\_test]# cat jdbc.conf

input {

stdin {

}

jdbc {

# mysql jdbc connection string to our backup databse 后面的test对应mysql中的test数据库

jdbc\_connection\_string => "jdbc:mysql://192.168.1.1:3306/test"

# the user we wish to excute our statement as

jdbc\_user => "root"

jdbc\_password => "\*\*\*\*\*\*"

# the path to our downloaded jdbc driver

jdbc\_driver\_library => "/elasticsearch-jdbc-2.3.2.0/lib/mysql-connector-java-5.1.38.jar"

# the name of the driver class for mysql

jdbc\_driver\_class => "com.mysql.jdbc.Driver"

jdbc\_paging\_enabled => "true"

jdbc\_page\_size => "50000"#以下对应着要执行的sql的绝对路径。

statement\_filepath => "/usr/local/logstash/bin/logstash\_jdbc\_test/jdbc.sql"#定时字段 各字段含义（由左至右）分、时、天、月、年，全部为\*默认含义为每分钟都更新（测试结果，不同的话请留言指出）

schedule => "\* \* \* \* \*"#设定ES索引类型

type => "cc\_type"

}

}

filter {

json {

source => "message"

remove\_field => ["message"]

}

}

output {

elasticsearch {#ESIP地址与端口

hosts => "192.168.1.1:9200"#ES索引名称（自己定义的）

index => "cc\_index"#自增ID编号

document\_id => "%{id}"

}

stdout {#以JSON格式输出

codec => json\_lines

}

}

#要执行的sql语句。

选择哪些信息同步到ES中。

[root@5b9dbaaa148a logstash\_jdbc\_test]# cat jdbc.sql

select

\*

from

where cc.modified\_at > :sql\_last\_value

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13
* 14
* 15
* 16
* 17
* 18
* 19
* 20
* 21
* 22
* 23
* 24
* 25
* 26
* 27
* 28
* 29
* 30
* 31
* 32
* 33
* 34
* 35
* 36
* 37
* 38
* 39
* 40
* 41
* 42
* 43
* 44
* 45
* 46
* 47
* 48
* 49
* 50
* 51
* 52
* 53
* 54

## [注意啦！注意啦！注意啦！]

cc.modified\_at, 这个modified\_at是我自己定义的更改时间字段，默认值default是now()当前时间。   
而 :sql\_last\_value如果input里面use\_column\_value => true， 即如果设置为true的话，可以是我们设定的字段的上一次的值。   
默认 use\_column\_value => false， 这样 :sql\_last\_value为上一次更新的最后时刻值。   
也就是说，对于新增的值，才会更新。这样就实现了增量更新的目的。

有童鞋问，如何全量更新呢？ 答案：就是去掉where子句即可。

### 步骤1：

在logstash的bin路径下新建文件夹logstash\_jdbc\_test，并将上两个文件 1）jdbc.conf，2)jdbc.sql.模板拷贝到里面。

### 步骤2：

按照自己的mysql地址、es地址、建立的索引名称、类型名称修改conf，以及要同步内容修改sql。

### 步骤3：

执行logstash, 如下：   
[root@5b9dbaaa148a plugins]# ./logstash -f ./logstash\_jdbc\_test/jdbc.conf

### 步骤4：

验证同步是否成功。   
可以通过： 如下图所示：   
