

# APACHE PIG AND HADOOP WITH ELASTICSEARCH: THE ELASTICSEARCH-HADOOP CONNECTOR



Here we show how to retrieve data from ElasticSearch using Apache Pig. The reason for doing that is Pig is much easier to use than Java, Scala, and other tools for doing data extraction and transformation ElasticSearch. (You can read our introduction to Apache Pig [here](#).) Also you can construct complex queries and sets using Pig that you could not with ES alone.

If you look on the internet, most of the examples you see, including those from ElasticSearch, explain how to **write** data to ElasticSearch (ES). For those who understand what ES does, that does not make much sense. ES is usually used together with Kibana and Logstash to store log data from applications. ES is a distributed database that stores documents in JSON format. But Apache Spark would be better suited to that.

*(This article is part of our [ElasticSearch Guide](#). Use the right-hand menu to navigate.)*

The real power of the Hadoop-ElasticSearch plugin is to read data from logs for [cybersecurity](#) and operations purposes. It is common for companies to gather data in ELK for that purpose. But you cannot write complex queries there. But you can do complex queries with Pig and save the data in Hadoop, Spark, or ES and then apply analytics to that.

We won't explain how to install Hadoop and ELK here. You can get instructions for those from Hadoop and ElasticSearch. This article assumes some basic knowledge of ELK.

Instead we are going to load some data in ElasticSearch and then use Apache Pig to query it.

Download the entirety of Shakespeare's plays from [here](#). Granted these are not logs, but they are a good example for sample data and the same that many other tutorials use.

Each line looks like this:

```
{"index":{"_index":"shakespeare","_type":"line","_id":11}}
{"line_id":12,"play_name":"Henry
IV","speech_number":1,"line_number":"1.1.9","speaker":"KING HENRY
IV","text_entry":"Of hostile paces: those opposed ey
es,"}
```

Load that data into ES like this:

```
curl -XPUT localhost:9200/_bulk --data-binary @shakespeare.json
```

Then when you open Kibana you should see the data like this, under the **shakespeare** index.

Discover

Visualize

Dashboard

Timeline

Dev Tools

Management

Collapse

Timeline

Dev Tools

Management

111,394 hits

New

Save

Open

Share

Search... (e.g. status:200 AND extension:PHP)

Uses lucene query syntax

Q

Add a filter

shakespear\*

Selected Fields

? \_source

Available Fields

t \_id

t \_index

# \_score

t \_type

# line\_id

t line\_number

t play\_name

t speaker

t speech\_number

t text\_entry

\_source

line\_id: 1 play\_name: Henry IV speech\_number: line\_number: speaker: text\_entry: ACT I \_id: 0 \_type: act \_index: shake  
spear \_score: 1

line\_id: 15 play\_name: Henry IV speech\_number: 1 line\_number: 1.1.12 speaker: KING HENRY IV text\_entry: Did lately meet in  
the intestine shock \_id: 14 \_type: line \_index: shakespeare \_score: 1

line\_id: 20 play\_name: Henry IV speech\_number: 1 line\_number: 1.1.17 speaker: KING HENRY IV text\_entry: The edge of war, li  
ke an ill-sheathed knife, \_id: 19 \_type: line \_index: shakespeare \_score: 1

line\_id: 23 play\_name: Henry IV speech\_number: 1 line\_number: 1.1.20 speaker: KING HENRY IV text\_entry: whose soldier now,  
under whose blessed cross \_id: 22 \_type: line \_index: shakespeare \_score: 1

line\_id: 25 play\_name: Henry IV speech\_number: 1 line\_number: 1.1.22 speaker: KING HENRY IV text\_entry: Forthwith a power o  
f English shall we levy; \_id: 24 \_type: line \_index: shakespeare \_score: 1

line\_id: 26 play\_name: Henry IV speech\_number: 1 line\_number: 1.1.23 speaker: KING HENRY IV text\_entry: whose arms were mou  
lded in their mothers womb \_id: 25 \_type: line \_index: shakespeare \_score: 1

line\_id: 27 play\_name: Henry IV speech\_number: 1 line\_number: 1.1.24 speaker: KING HENRY IV text\_entry: To chase these paga  
ns in those holy fields \_id: 26 \_type: line \_index: shakespeare \_score: 1

line\_id: 30 play\_name: Henry IV speech\_number: 1 line\_number: 1.1.27 speaker: KING HENRY IV text\_entry: For our advantage o  
n the bitter cross. \_id: 29 \_type: line \_index: shakespeare \_score: 1

line\_id: 41 play\_name: Henry IV speech\_number: 2 line\_number: 1.1.38 speaker: WESTMORELAND text\_entry: whose worst was, tha  
t the noble Mortimer, \_id: 40 \_type: line \_index: shakespeare \_score: 1

line\_id: 42 play\_name: Henry IV speech\_number: 2 line\_number: 1.1.39 speaker: WESTMORELAND text\_entry: Leading the men of H  
erefordshire to fight \_id: 41 \_type: line \_index: shakespeare \_score: 1

line\_id: 45 play\_name: Henry IV speech\_number: 2 line\_number: 1.1.42 speaker: WESTMORELAND text\_entry: A thousand of his pe  
ople butchered; \_id: 44 \_type: line \_index: shakespeare \_score: 1

line\_id: 49 play\_name: Henry IV speech\_number: 2 line\_number: 1.1.46 speaker: WESTMORELAND text\_entry: Without much shame r  
etold or spoken of. \_id: 48 \_type: line \_index: shakespeare \_score: 1

line\_id: 53 play\_name: Henry IV speech\_number: 4 line\_number: 1.1.50 speaker: WESTMORELAND text\_entry: For more uneven and  
unwelcome news \_id: 52 \_type: line \_index: shakespeare \_score: 1

line\_id: 61 play\_name: Henry IV speech\_number: 4 line\_number: 1.1.58 speaker: WESTMORELAND text\_entry: And shape of likelih  
ood, the news was told; \_id: 60 \_type: line \_index: shakespeare \_score: 1

line\_id: 74 play\_name: Henry IV speech\_number: 5 line\_number: 1.1.71 speaker: KING HENRY IV text\_entry: Mordake the Earl of  
Fife, and eldest son \_id: 73 \_type: line \_index: shakespeare \_score: 1

line\_id: 80 play\_name: Henry IV speech\_number: 6 line\_number: 1.1.77 speaker: WESTMORELAND text\_entry: It is a conquest for  
a prince to boast of. \_id: 79 \_type: line \_index: shakespeare \_score: 1

line\_id: 85 play\_name: Henry IV speech\_number: 7 line\_number: 1.1.82 speaker: KING HENRY IV text\_entry: Amongst a grove, th  
e very straightest plant; \_id: 84 \_type: line \_index: shakespeare \_score: 1

line\_id: 90 play\_name: Henry IV speech\_number: 7 line\_number: 1.1.87 speaker: KING HENRY IV text\_entry: That some night-tri  
pping fairy had exchanged \_id: 89 \_type: line \_index: shakespeare \_score: 1

line\_id: 93 play\_name: Henry IV speech\_number: 7 line\_number: 1.1.90 speaker: KING HENRY IV text\_entry: Then would I have h  
is Harry, and he mine. \_id: 92 \_type: line \_index: shakespeare \_score: 1

line\_id: 99 play\_name: Henry IV speech\_number: 8 line\_number: 1.1.96 speaker: WESTMORELAND text\_entry: This is his uncles t  
eaching; this is Worcester, \_id: 98 \_type: line \_index: shakespeare \_score: 1

Now download the last files `elasticsearch-hadoop-5.5.2.jar` and `commons-httpclient-3.1.jar` from Maven.

Then start Pig in local mode (or cluster if that is what you have). (You can make life easier if you run everything as root. Note that you cannot run Elasticsearch as root.)

```
pig -x local
```

This will open the Pig shell. So that those jars come into scope, enter these two commands into the shell:

```
REGISTER /home/walker/Documents/jars/elasticsearch-hadoop-5.5.2.jar
REGISTER /home/hadoop/Documents/jars/commons-httpclient-3.1.jar
```

Now, define a shortcut for ES storage like this:

```
DEFINE EsStorage org.elasticsearch.hadoop.pig.EsStorage();
```

There are lots of options you could pass to that like:

```
DEFINE EsStorage org.elasticsearch.hadoop.pig.EsStorage (
'es.http.timeout= 5m',
'es.index.auto.create = true',
'es.mapping.pig.tuple.use.field.names = true',
'es.mapping.id = id'
);
```

Now load (some of) the data into Pig from ElasticSearch.

```
a = LOAD 'shakespeare' USING
org.elasticsearch.hadoop.pig.EsStorage('es.query=?q=wine');
```

What we have done is use the Lucene (very simple, natural-language) query ability of ES to load every line in the play that has the word **wine** in it. (If you've read much Shakespeare you know they also call it **slack**.)

The result we get is a series of tuples.

ES has no schema since its storage format is JSON. Neither does a tuple.

```
(47371,Julius Caesar,32,2.2.134,CAESAR,Good friends, go in, and taste some
wine with me;)
(64337,Merry Wives of Windsor,83,1.1.165,PAGE,Nay, daughter, carry the wine
in; well drink within.)
(65573,Merry Wives of Windsor,32,3.2.79,FORD, I think I shall drink in pipe
wine first)
```

Now we can load the data with a schema like this:

```
b = LOAD 'shakespeare' USING
org.elasticsearch.hadoop.pig.EsStorage('es.query=?q=wine') as
(line_id:string,chararray, play_name:chararray, speech_number:int,
line_number:chararray, speaker:chararray, text_entry:chararray);
```

Then we ask Pig to show us the schema:

```
describe b
```

```
b: {line_id: chararray,play_name: chararray,speech_number: int,line_number:
chararray,speaker: chararray,text_entry: chararray}
```

When you are done with your dataset running queries and transformations you could load save it into Pig (meaning Hadoop) as when you close the Pig shell you would lose it.