PROJECT REPORT

INDEXING WIKI DUMPS ON CLOUD USING ELK STACK & SOLR

TABLE OF CONTENTS:

PROJECT DESCRIPTION:

The project agenda was loading Wiki Datasets and Indexing them on following Cloud Platforms using Elasticsearch , Logstash and Kibana Stack & SOLR

- Amazon Web Services
- Google Cloud Platform
- Microsoft Azure

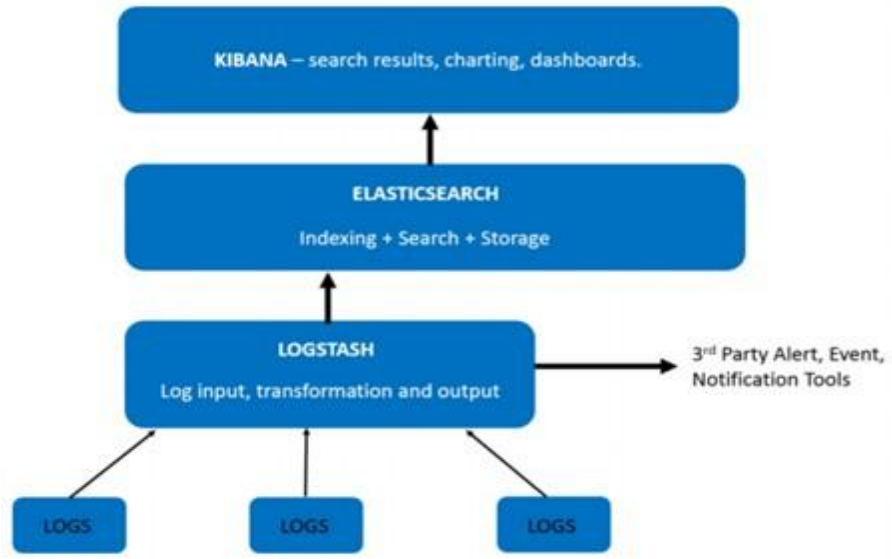
INDEXING:

Indexing is a way to optimize performance of a database by minimizing the number of disk accesses required when a query is processed. An index or database index is a data structure which is used to quickly locate and access the data in a database table.

INDEXING WIKI DUMPS On GOOGLE CLOUD PLATFORM using ELK Stack:

ELASTICSEARCH:

Elastic search is an open source, broadly-distributable, readily-scalable, enterprise-grade search engine based on Lucene and released under the terms of the Apache License. It is Java-based and designed to operate in real time. It can search and index document files in diverse formats. It was designed to be used in distributed environments by providing flexibility and scalability. Now, elastic search is the most popular enterprise search engine followed by Apache Solr, also based on Lucene.



ADVANTAGES OF ELASTIC SEARCH OVER SOLR

Build on top of lucene

Elastic search is built on top of Lucene, which is a full-featured information retrieval library, so it provides the most powerful full-text search capabilities of any open source product.

Document- oriented

Elastic search is document-oriented. It stores real world complex entities as structured JSON documents and indexes all fields by default, with a higher performance result.

Speed

Elastic search is able to execute complex queries extremely fast. It also caches almost all of the structured queries commonly used as a filter for the result set and executes them only once. For every other request which contains a cached filter, it checks the result from the cache. This saves the time parsing and executing the query improving the speed.

Structured search

Elastic Search is schema free, it accepts JSON documents, as well as tries to detect the data structure, index the data, and make it searchable.

Data record

Elastics earch records any changes made in transactions logs on multiple nodes in the cluster to minimize the chance of data loss.

INSTALLATION OF ELASTICSEARCH:

Firewall Creation:

Creating Firewall rules for access to ports 9200 and 5601 for Elastic search and Kibana.

Ingress Egress

Name	Targets	Source filters	Protocols / ports	Action	Priority	Network
elasticsearch	Apply to all	IP ranges: 0.0.0.0/0	tcp:9200	Allow	1000	default
kibana	Apply to all	IP ranges: 0.0.0.0/0	tcp:5601	Allow	1000	default
default-allow-icmp	Apply to all	IP ranges: 0.0.0.0/0	icmp	Allow	65534	default
default-allow-internal	Apply to all	IP ranges: 10.128.0.0/9	tcp:0-65535, udp:0-65535, 1 more	Allow	65534	default
default-allow-rdp	Apply to all	IP ranges: 0.0.0.0/0	tcp:3389	Allow	65534	default
default-allow-ssh	Apply to all	IP ranges: 0.0.0.0/0	tcp:22	Allow	65534	default

To install Java

$ sudo apt-get install default-jre

This will fetch the latest ElasticSearch Version for us

$ wget -qO - https://packages.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -

This will complete the installation

$ sudo apt-get install elasticsearch

Find the line referring to the network.host portion. It will be commented out. Uncomment the file and make it read network.host “0.0.0.0"

$ sudo vi /etc/elasticsearch/elasticsearch.yml

ELASTIC SEARCH is installed and running successfully.

```
venkateshumamaheswaran@instance-1:~$ curl localhost:9200
{
  "name" : "jlH4DEo",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "Oaen330HRfSMhUSo4GYWZw",
  "version" : {
    "number" : "5.6.3",
    "build_hash" : "1a2f265",
    "build_date" : "2017-10-06T20:33:39.012Z",
    "build_snapshot" : false,
    "lucene_version" : "6.6.1"
  },
  "tagline" : "You Know, for Search"
}
venkateshumamaheswaran@instance-1:~$
```

KIBANA

Kibana is an open source analytics and visualization platform designed to work with Elasticsearch. You use Kibana to search, view, and interact with data stored in Elasticsearch indices. You can easily perform advanced data analysis and visualize your data in a variety of charts, tables, and maps.
Kibana makes it easy to understand large volumes of data. Its simple, browser-based interface enables you to quickly create and share dynamic dashboards that display changes to Elasticsearch queries in real time.

The different views in Kibana are as follows:
The **discover** view is used to view a list of documents and search for specific documents.
The **visualize** view is used to create visualizations like graphs from the data. We can add those visualization to **dashboards** to have an overview of you data at a glance.
Timelion was formerly a plugin and is now build in. It's used to make advanced timeseries analysis.

The **management** tab are the settings of Kibana where we can add index patterns and tune some advanced settings.
The **Dev Tools** currently only contain the so called Console, which was formerly known as the Sense plugin in Elasticsearch. We can use it to send JSON directly to Elasticsearch and more meant for developers or advanced users.

INSTALLATION OF KIBANA

This will establish the source for Kibana
$ echo "deb http://packages.elastic.co/kibana/5.3/debian stable main" | sudo tee -a /etc/apt/sources.list.d/kibana-5.3.x.list

Setting up for Kibana installation
$ sudo apt-get update
$ sudo apt-get install kibana

```
venkateshumamaheswaran@instance-1:~$ curl -XGET 'http://localhost:5601/'
<script>var hashRoute = '/app/kibana';
var defaultRoute = '/app/kibana';

var hash = window.location.hash;
if (hash.length) {
  window.location = hashRoute + hash;
} else {
  window.location = defaultRoute;
}</script>venkateshumamaheswaran@instance-1:~$
```

```
venkateshumamaheswaran@instance-1:~$ curl -XGET 'http://localhost:5601/'
<script>var hashRoute = '/app/kibana';
var defaultRoute = '/app/kibana';

var hash = window.location.hash;
if (hash.length) {
  window.location = hashRoute + hash;
} else {
  window.location = defaultRoute;
}</script>venkateshumamaheswaran@instance-1:~$ sudo service kibana status
● kibana.service - no description given
   Loaded: loaded (/lib/systemd/system/kibana.service; disabled; vendor preset: enabled)
   Active: active (running) since Tue 2017-10-24 16:31:54 UTC; 10h ago
 Main PID: 32303 (node)
    Tasks: 9 (limit: 4915)
   CGroup: /system.slice/kibana.service
           └─32303 /opt/kibana/bin/../node/bin/node /opt/kibana/bin/../src/cli
```

$ sudo service kibana start

```
venkateshumamaheswaran@instance-1:~$ sudo service kibana status
● kibana.service - Kibana
   Loaded: loaded (/etc/systemd/system/kibana.service; disabled; vendor preset: enabled)
   Active: active (running) since Sun 2017-12-03 01:47:24 UTC; 39min ago
 Main PID: 1719 (node)
    Tasks: 10 (limit: 4915)
   CGroup: /system.slice/kibana.service
           └─1719 /usr/share/kibana/bin/../node/bin/node --no-warnings /usr/share/kibana/bin/../src/cli -c /etc/kibana/kibana.yml

Dec 03 01:47:39 instance-1 kibana[1719]: {"type":"log","@timestamp":"2017-12-03T01:47:39Z","tags":["status","plugin:console@5.6.3","info"],"pid":1719,"state":"green","message
:"Status changed from uninitialized to green - Ready","prevState":"uninitialized","prevMsg":"uninitialized"}
Dec 03 01:47:39 instance-1 kibana[1719]: {"type":"log","@timestamp":"2017-12-03T01:47:39Z","tags":["status","plugin:elasticsearch@5.6.3","error"],"pid":1719,"state":"red","me
ssage":"Status changed from yellow to red - Unable to connect to Elasticsearch at http://localhost:9200.","prevState":"yellow","prevMsg":"Waiting for Elasticsearch"}
Dec 03 01:47:39 instance-1 kibana[1719]: {"type":"log","@timestamp":"2017-12-03T01:47:39Z","tags":["status","plugin:metrics@5.6.3","info"],"pid":1719,"state":"green","message
:"Status changed from uninitialized to green - Ready","prevState":"uninitialized","prevMsg":"uninitialized"}
Dec 03 01:47:39 instance-1 kibana[1719]: {"type":"log","@timestamp":"2017-12-03T01:47:39Z","tags":["status","plugin:timelion@5.6.3","info"],"pid":1719,"state":"green","messag
":"Status changed from uninitialized to green - Ready","prevState":"uninitialized","prevMsg":"uninitialized"}
Dec 03 01:47:39 instance-1 kibana[1719]: {"type":"log","@timestamp":"2017-12-03T01:47:39Z","tags":["listening","info"],"pid":1719,"message":"Server running at http://0.0.0.0:
601"}
Dec 03 01:47:39 instance-1 kibana[1719]: {"type":"log","@timestamp":"2017-12-03T01:47:39Z","tags":["status","ui settings","error"],"pid":1719,"state":"red","message":"Status
hanged from uninitialized to red - Elasticsearch plugin is red","prevState":"uninitialized","prevMsg":"uninitialized"}
Dec 03 01:47:42 instance-1 kibana[1719]: {"type":"log","@timestamp":"2017-12-03T01:47:42Z","tags":["status","plugin:elasticsearch@5.6.3","error"],"pid":1719,"state":"red","me
ssage":"Status changed from red to red - Service Unavailable","prevState":"red","prevMsg":"Unable to connect to Elasticsearch at http://localhost:9200."}
Dec 03 01:47:44 instance-1 kibana[1719]: {"type":"log","@timestamp":"2017-12-03T01:47:44Z","tags":["status","plugin:elasticsearch@5.6.3","error"],"pid":1719,"state":"red","me
ssage":"Status changed from red to red - Elasticsearch is still initializing the kibana index.","prevState":"red","prevMsg":"Service Unavailable"}
Dec 03 01:47:50 instance-1 kibana[1719]: {"type":"log","@timestamp":"2017-12-03T01:47:50Z","tags":["status","plugin:elasticsearch@5.6.3","info"],"pid":1719,"state":"green","m
ssage":"Status changed from red to green - Kibana index ready","prevState":"red","prevMsg":"Elasticsearch is still initializing the kibana index."}
Dec 03 01:47:50 instance-1 kibana[1719]: {"type":"log","@timestamp":"2017-12-03T01:47:50Z","tags":["status","ui settings","info"],"pid":1719,"state":"green","message":"Status
changed from red to green - Ready","prevState":"red","prevMsg":"Elasticsearch plugin is red"}
```

LOGSTASH:

Logstash is a tool for managing events and logs.The purpose of Logstash is to get events from any number of inputs (could be from a file, a queue, another Logstash instance, etc), apply filters (parse, modify, or perform any number of processing tasks), and finally output to any number of destinations.

INSTALLATION OF LOGSTASH:

This setups installs for logstash in your system
$ sudo apt-get install apt-transport-https

This will establish the source for Logstash
$ echo "deb https://artifacts.elastic.co/packages/5.x/apt stable main" | sudo tee -a /etc/apt/sources.list.d/elastic-5.x.list

Setting up for Logstash installation
$ sudo apt-get update
$ sudo apt-get install logstash

Start the logstash service so we can start shipping logs
$ sudo service logstash start

```
venkateshumamaheswaran@instance-1:~$ sudo service logstash status
● logstash.service - LSB: Starts Logstash as a daemon.
   Loaded: loaded (/etc/init.d/logstash; generated; vendor preset: enabled)
   Active: active (exited) since Wed 2017-10-04 06:31:39 UTC; 2 weeks 6 days ago
     Docs: man:systemd-sysv-generator(8)
    Tasks: 0 (limit: 4915)
   CGroup: /system.slice/logstash.service
```

SIMPLE WIKIPEDIA DATASET:

Elasticsearch only supports JSON documents. We have chosen simplewiki document to index into our Elasticsearch instance.
This is the link for Simple Wikipedia data we are using:
https://dumps.wikimedia.org/other/cirrussearch/20171106/enwikiquote-20171106-cirrussearch-general.json.gz

enwikinews-20171106-cirrussearch-content.json.gz	07-Nov-2017 17:41	51697701
enwikinews-20171106-cirrussearch-general.json.gz	07-Nov-2017 17:56	366382264
enwikiquote-20171106-cirrussearch-content.json.gz	07-Nov-2017 17:57	195231449
enwikiquote-20171106-cirrussearch-general.json.gz	07-Nov-2017 17:58	63055360
enwikisource-20171106-cirrussearch-content.json.gz	07-Nov-2017 18:28	4274507543
enwikisource-20171106-cirrussearch-general.json.gz	07-Nov-2017 18:30	144400036
enwikiversity-20171106-cirrussearch-content.jso..>	07-Nov-2017 18:31	122816349
enwikiversity-20171106-cirrussearch-general.jso..>	07-Nov-2017 18:32	151778222

LOADING WIKI DATA AND INDEXING:

Step 1: Download a wiki dump
Step 2: Get the index ready
Step 3: Prepare the wiki for loading
Step 4: Load the wiki

INDEXING WIKI QUOTE DATA:

We need analysis-icu plugin for Elasticsearch to handle it the index.
bin/plugin install analysis-icu
Then we need jq for some of the json-foo we do next.
sudo apt-get install jq
Then we have to create 3 vim files createindex.sh, chunker.sh, uploader.sh
CREATEINDEX.SH

```
export es=localhost:9200
export site=en.wikiquote.org
export index=enwikiquote

curl -XDELETE $es/$index?pretty

curl -s 'https://'$site'/w/api.php?action=cirrus-settings-dump&format=json&formatversion=2' |
  jq '{
      analysis: .content.page.index.analysis,
      number_of_shards: 1,
      number_of_replicas: 0
      }' |
      curl -XPUT $es/$index?pretty -d @-

curl -s 'https://'$site'/w/api.php?action=cirrus-mapping-dump&format=json&formatversion=2' |
  jq .content |
  sed 's/"index_analyzer"/"analyzer"/' |
  sed 's/"position_offset_gap"/"position_increment_gap"/' |
  curl -XPUT $es/$index/_mapping/page?pretty -d @-
```

CODE EXPLANATION FOR CREATEINDEX.SH:

export es=localhost:9200 sets up $es to be Elasticsearch's address.
export site=en.wikiquote.org sets up $site to be the hostname of the MediaWiki instance that you want to use.
export index=enwikiquote just sets $index to the name of the index you'll be loading.
curl -XDELETE $es/$index?pretty deletes the index if it already exists.

CHUNKER.SH

```
export dump=enwikiquote-20171106-cirrussearch-general.json.gz
export index=enwikiquotes

mkdir chunks
cd chunks
zcat ../$dump | split -a 10 -l 500 - $index
```

CODE EXPLANATION FOR CHUNKER.SH:

The first export line just names the file that you downloaded.
The mkdir and cd lines make a directory to hold the files.
The last line cuts the file into 500 line chunks. 250 of those lines are metadata lines for the _bulk api. 250 lines are the actual documents.

UPLOADER.SH:

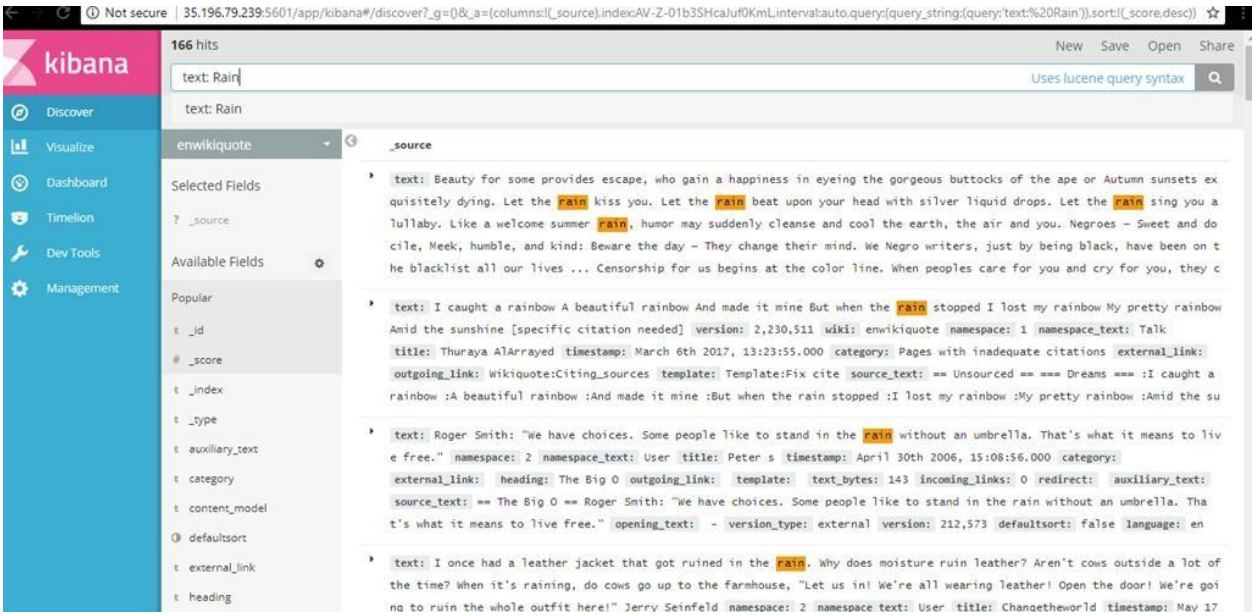
```
export es=localhost:9200
export index=enwikiquote`
cd chunks

for file in *; do
        echo -n "${file}:  "
        took=$(curl -s -XPOST $es/$index/_bulk?pretty --data-binary @$file |
           grep took | cut -d':' -f 2 | cut -d',' -f 1)
        printf '%7s\n' $took
        [ "x$took" = "x" ] || rm $file
done
```

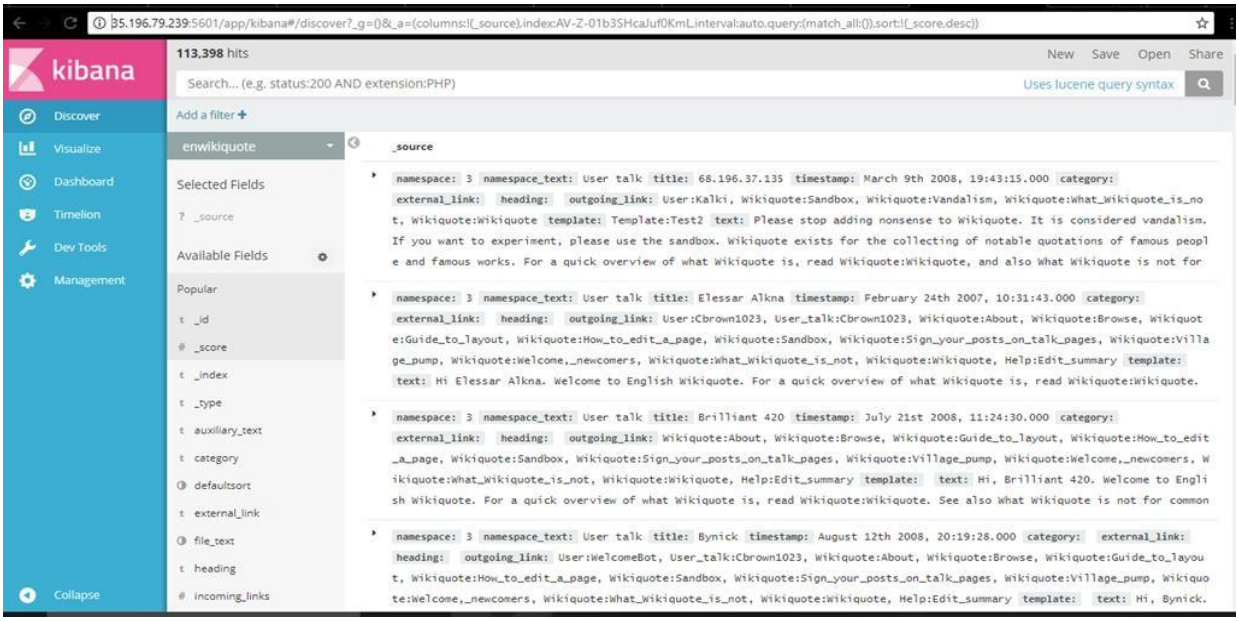
CODE EXPLANATION FOR UPLOADER.SH:

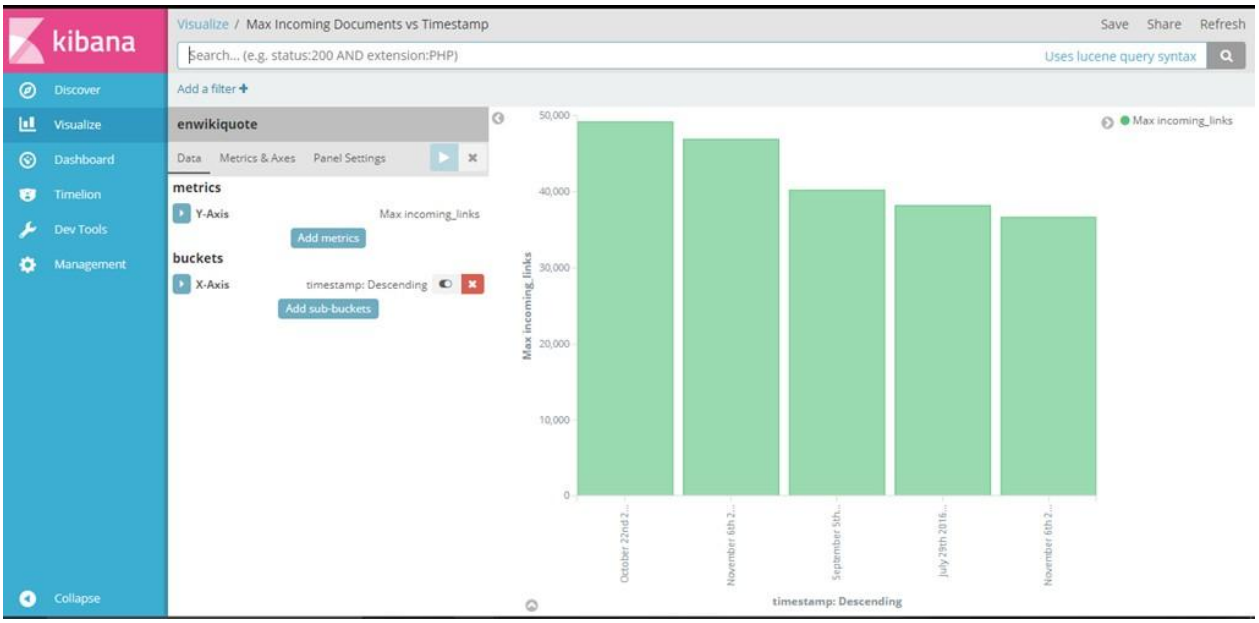
The first three lines should be familiar from above. The loop loads each file and deletes it after it's loaded.
If the file fails to load it isn't deleted and the loop moves on to the next file.

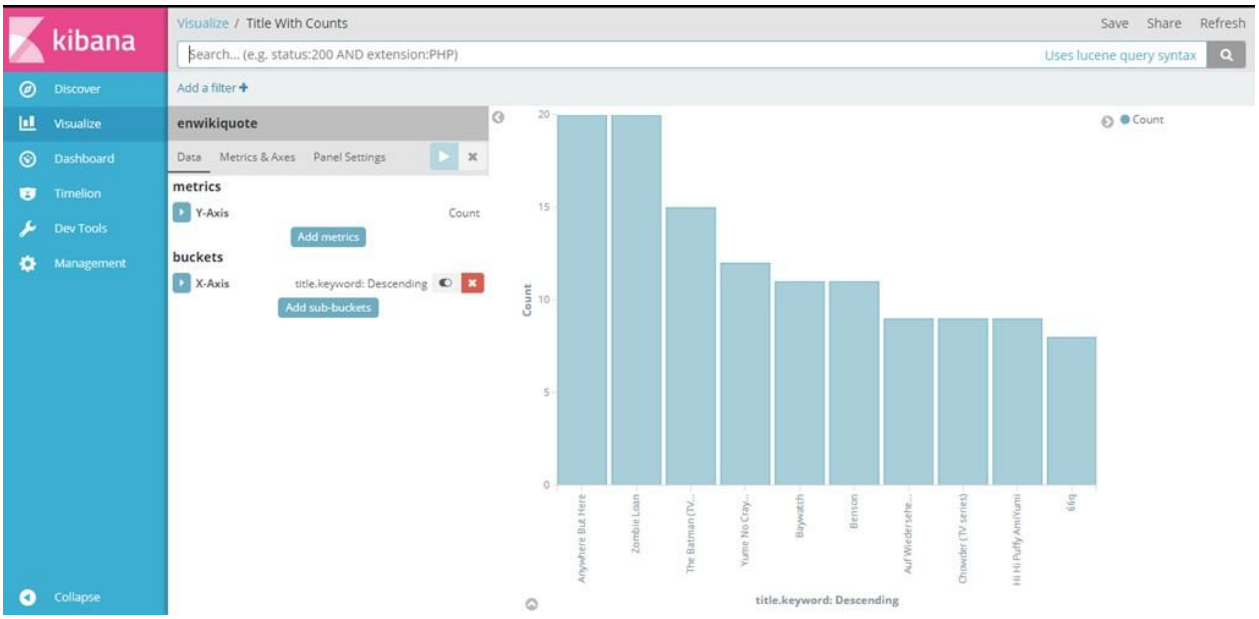
KIBANA VISUALIZATION:

Here are some of the example kibana visualizations we have come up with





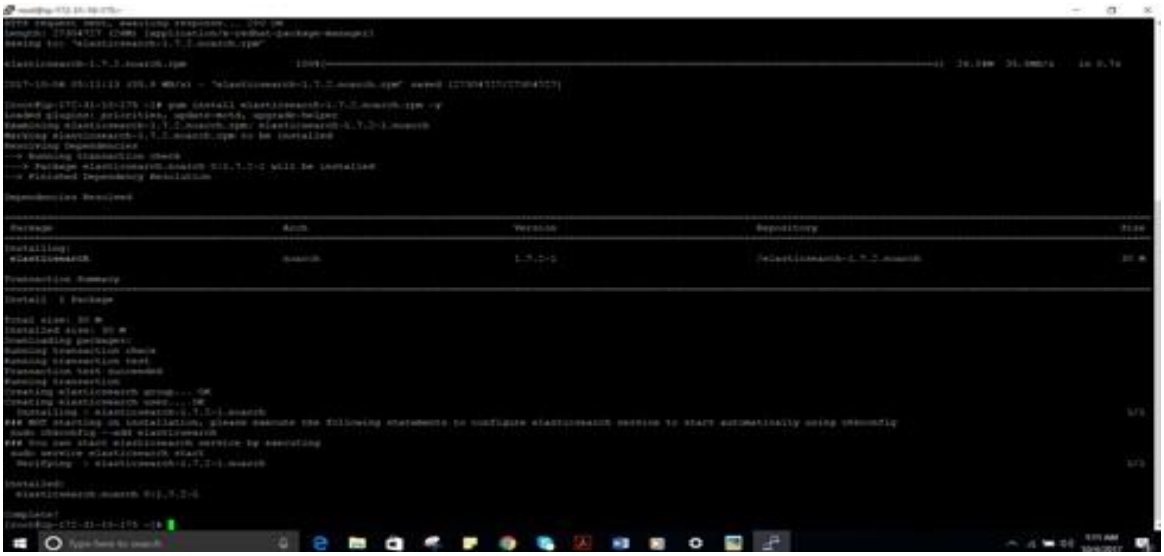




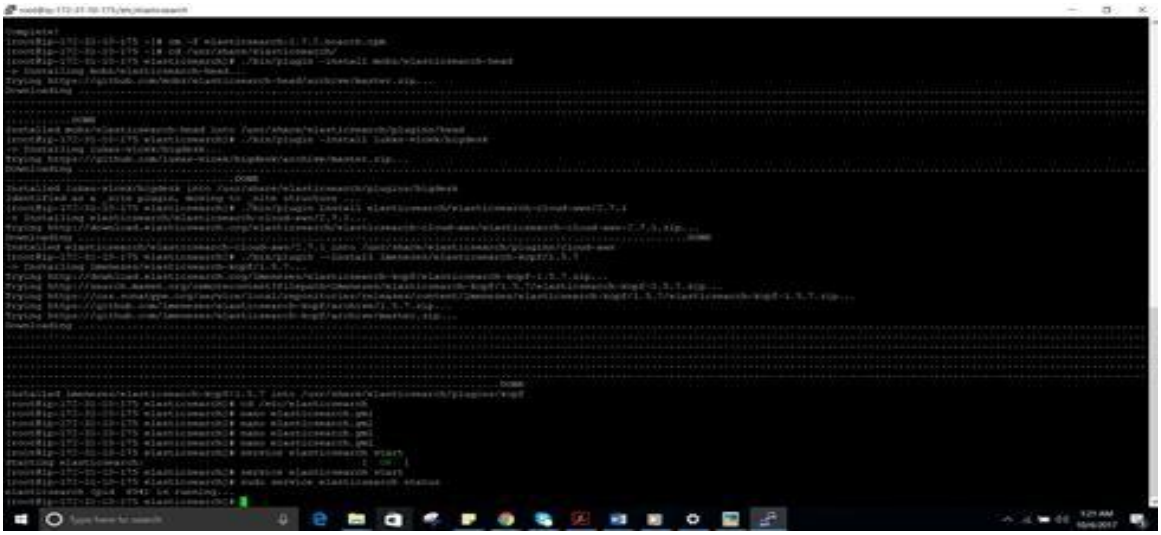
INDEXING WIKI DUMPS On AMAZON WEB SERVICES using ELK Stack:

The same commands are to be executed for installation of Elasticsearch, Logstash and Kibana as used for Google Cloud Platform. Since the commands are the very same, you can see the screenshots of successful installation of ELK stack on AMAZON WEB SERVICES.

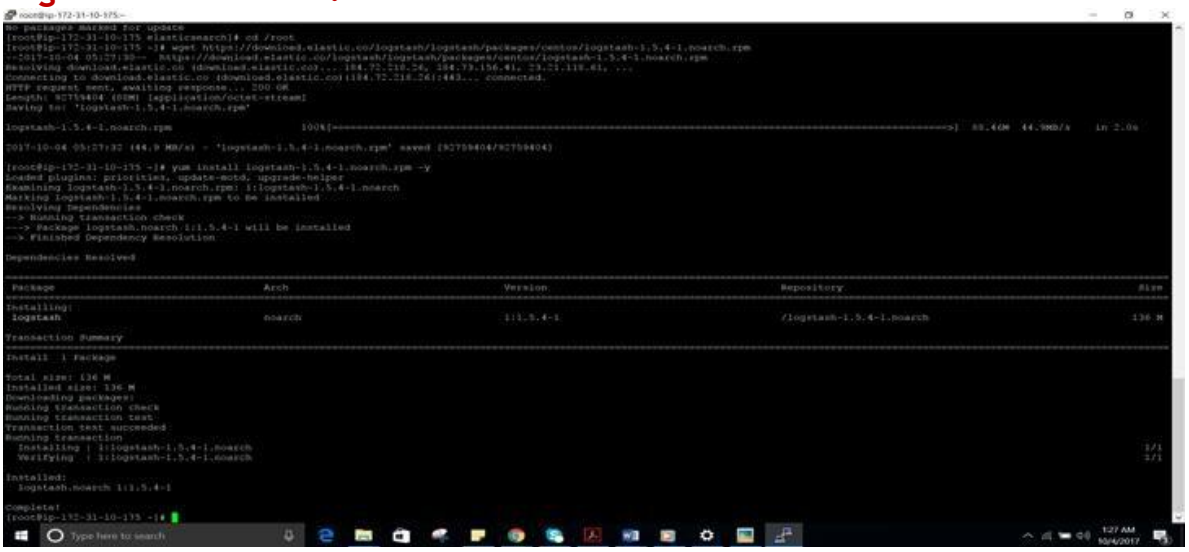
Elasticsearch Installation:



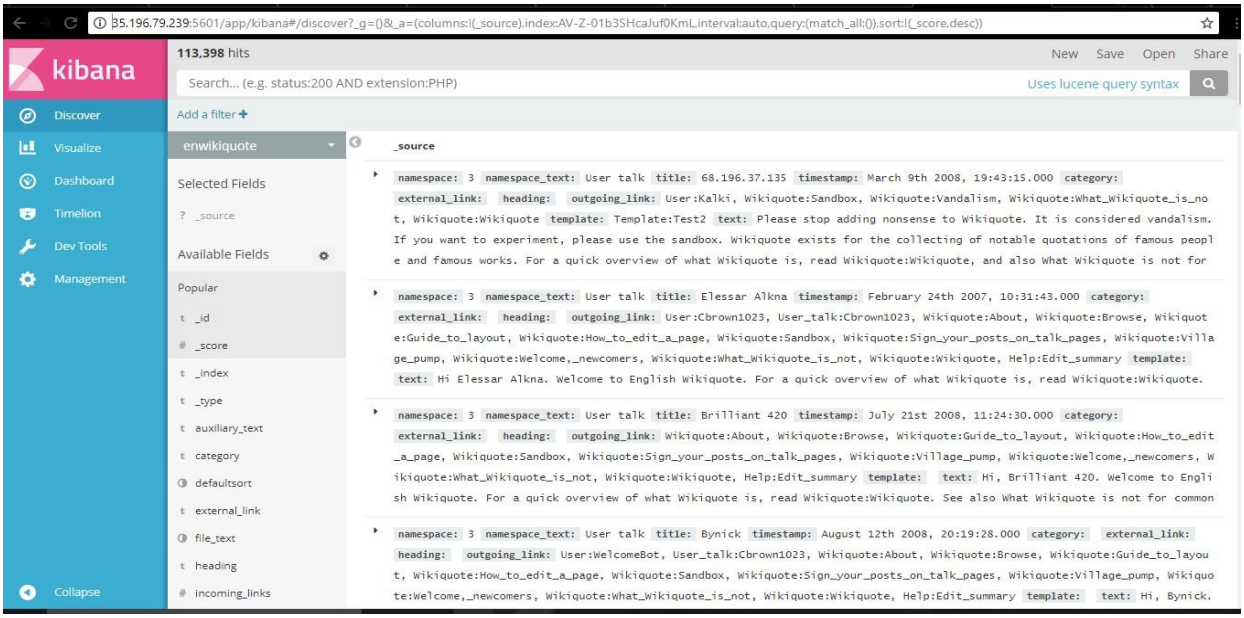
Kibana Installation:



Logstash Installation:

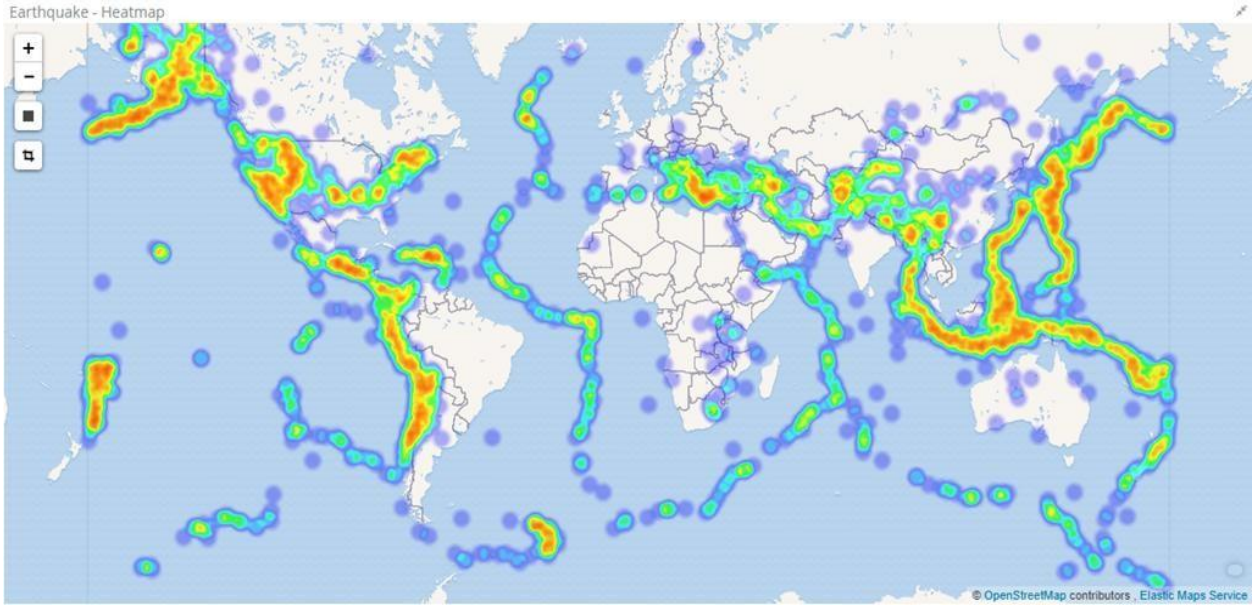


KIBANA VISUALIZATIONS THROUGH AWS:



Earthquakes Data

For more sample visualizations and better hands- on experience on Elasticsearch and Kibana, we tried this data available on github https://github.com/elastic/examples/tree/master/Exploring%20Public%20Datasets/earthquakes



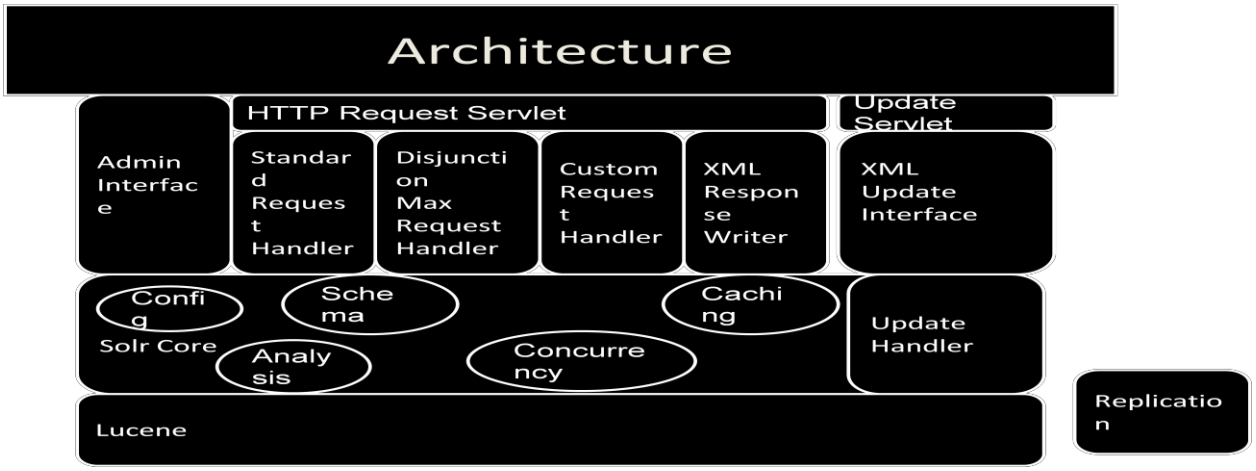
INDEXING WIKI DUMPS ON MICROSOFT AZURE USING SOLR:

SOLR:

Solr is powered by Lucene, a powerful open-source full-text search library, under the hood. Solr is designed for scalability and fault tolerance. Solr is widely used for enterprise search and analytics use cases.

- XML/HTTP Interfaces
- Loose Schema to define types and fields
- Web Administration Interface
- Extensive Caching
- Index Replication
- Extensible Open Architecture

ARCHITECTURE OF SOLR:



INSTALLATION STEPS OF SOLR:

Below are the steps for the installation steps of SOLR clearly explained step by step:

1. Download solr-3.4
2. Download wikipedia dump
3. data-config.xml was used to index Wikipedia dump.

```
<dataConfig>
<dataSource type="FileDataSource" encoding="UTF-8" />
<document>
<entity name="page"
processor="XPathEntityProcessor"
stream="true"
forEach="/mediawiki/page/"
```

4. The relevant portion of schema.xml is below:

```
<field name="id" type="string" indexed="true" stored="true" required="true"/>
<field name="title" type="string" indexed="true" stored="false"/>
<field name="revision" type="sint" indexed="true" stored="true"/>
<field name="user" type="string" indexed="true" stored="true"/>
<field name="userId" type="int" indexed="true" stored="true"/>
<field name="text" type="text" indexed="true" stored="false"/>
<uniqueKey>id</uniqueKey>
<copyField source="title" dest="titleText"/>
```

5. Add Dih request handler in solrconfig.xml file

```
<requestHandler name="/update/dih" startup="lazy">
<lst name="defaults">
<str name="config">dih-config.xml</str>
</lst>
```

6. Restart solr
7. Index some documents using below command_

http://localhost:8983/solr/update/dih?command=full-import

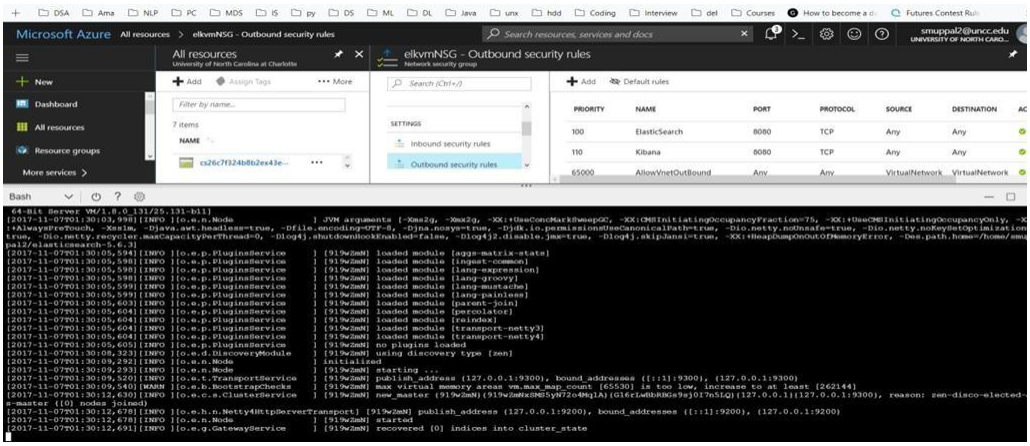
Schema: Analyzers:

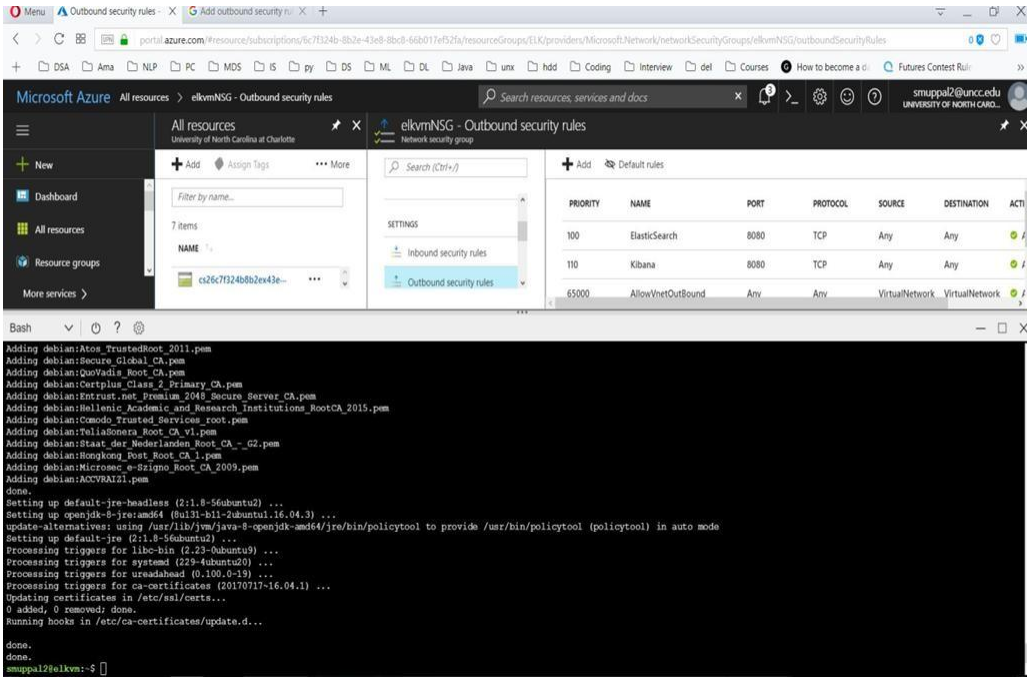
```
<fieldtype name="nametext" class="solr.TextField">
    <analyzer class="org.apache.lucene.analysis.WhitespaceAnalyzer"/>
</fieldtype>
<fieldtype name="text" class="solr.TextField">
    <analyzer>
        <tokenizer class="solr.StandardTokenizerFactory"/>
        <filter class="solr.StandardFilterFactory"/>
        <filter class="solr.LowerCaseFilterFactory"/>
        <filter class="solr.StopFilterFactory"/>
        <filter class="solr.PorterStemFilterFactory"/>
    </analyzer>
```

```
</fieldtype>
<fieldtype name="myfieldtype" class="solr.TextField">
        <analyzer>
                <tokenizer class="solr.WhitespaceTokenizerFactory"/>
                <filter class="solr.SnowballPorterFilterFactory" language="German" />
        </analyzer>
</fieldtype>
```

INSTALLATION OF SOLR IN AZURE:

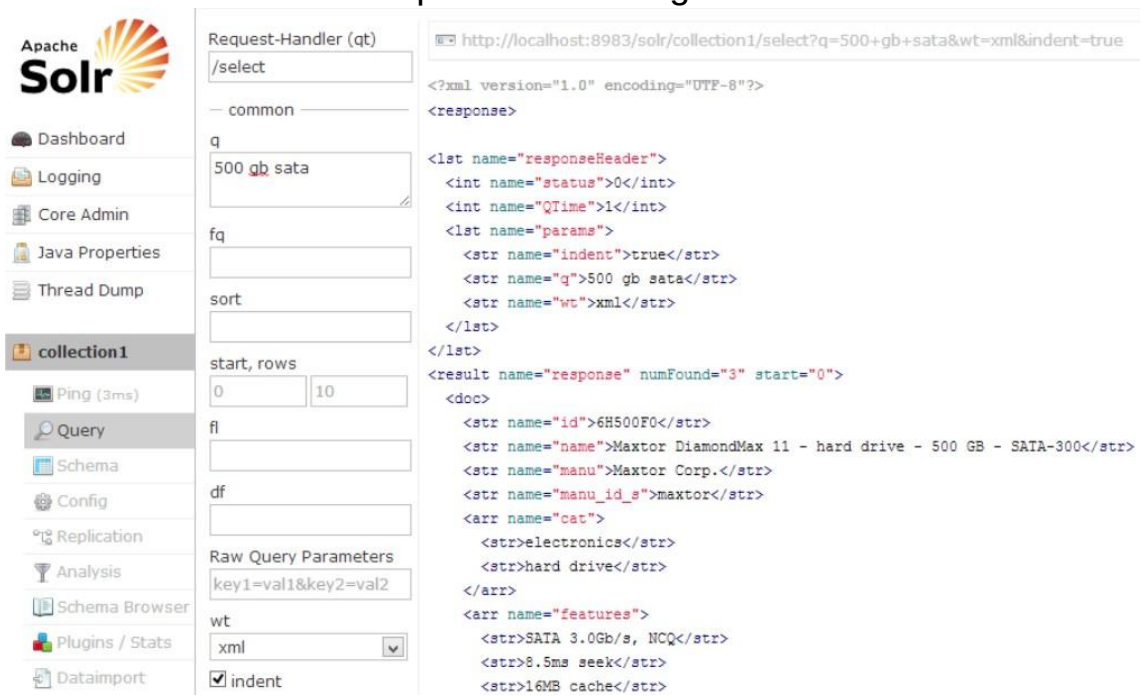
Below are the screenshots for installation of SOLR using AZURE





INDEXED WIKIPEDIA DATA IN SOLR:

Below is the indexed wikipedia data using SOLR



CREDITS:

https://logz.io/blog/elk-stack-google-cloud/
https://dzone.com/articles/how-to-install-the-elk-stack-on-google-cloud-platf-1
https://www.elastic.co/blog/loading-wikipedia
https://dumps.wikimedia.org/other/cirrussearch/current/
https://www.digitalocean.com/community/tutorials/how-to-use-kibana-dashboards-and-visualIzations
https://github.com/elastic/examples/tree/master/Exploring%20Public%20Datasets/earthquakes
https://stackoverflow.com/questions/3846793/running-solr-on-azure