20 FEBRUARY 2016 **ENGINEERING**

Loading Wikipedia's Search Index For Testing

By Nik Everett

Share

A month ago I published a **blog** post testing query speeds with data from English Wikipedia and mentioned that the loading process deserved its own blog post. Well here it is!

Step 0: Why

This gets you a copy of the search index used to power on site search for Wikipedia. The index is quite large because it's used to support all the funky things that Elasticsearch is used for there. It may have mistakes, old data, strange leftovers from previous versions of the indexing code that have yet to be cleaned up. In short: it's imperfect, just like any real production deploy.

Step 1: Download a smaller wiki

The Wikimedia Foundation makes everything about Wikipedia and the other wikis it powers public. Well, everything that is safe to make public anyway. That includes a dump of the search index. Head **here** and you'll get a list of dates when the dump

runs began. Click on the latest date and download the file that looks like `enwikiquote-$date-cirrussearch-content.json.gz`. The naming scheme is `$wiki-$date-cirrussearch-$indexType.json.gz` where:

- `$wiki` is usually made of two parts: `$langCode$project`
  - `$langCode` is the language - mostly **ISO 639-1** codes. I use English because it is nice and large and I can read it.
  - `$project` is the kind of wiki:
    - `wiki` is **Wikipedia**, an encyclopedia
    - `wiktionary` is **Wikitionary**, a dictionary
    - `wikisource` is **Wikisource**, a library of free content
    - `wikivoyage` is **Wikivoyage**, a collaborative travel guide
    - `wikibooks` is **Wikibooks**, a repository of open content text books
    - `wikiquote` is **Wikiquote**, a collection of quotations
    - `wikiversity` is **Wikiversity**, an open learning community
    - `wikinews` is **Wikinews**, a collaborative news site
  - There are some special wiki codes like `commonwiki` which is for **Wikimedia Commons**, a repository of free multimedia stuff.
- `$indexType` is the type of search index. Normally it's either `content`, pages someone reading the wiki usually wants to search, and `general`, pages that are useful for building the wiki like templates discussions, and scripts. Wikimedia Commons, being a multimedia repository, has it a `file $indexType` that contains data about the files like description. Sometimes it contains text that can be extracted from the files, like **this** issue of Scientific American. Some of these extracts are fairly large like **this** biography.

By now you've figured out that I didn't have you download English Wikipedia's search index. I had you fetch English Wikiquote. You can follow my instructions below to load English Wikiquote and then start over to load any other wiki you'd like from the dumps page. If one particular wiki caught your eye, go and start downloading it now while you follow along to the end of the blog post using English Wikiquote. If all goes well it'll be ready by the time you want it.

Step 2: Get the index ready

While the dump downloads you can setup Elasticsearch to handle it the index. You'll need the `analysis-icu` plugin. You can do that by this into your bash shell:

```
bin/plugin install analysis-icu
```

You'll have to restart Elasticsearch after installing that for it to take.

Then you'll need **jq** for some of the json-foo you'll do next. For me it's just `sudo apt-get install jq` but your command may vary.

Finally you can create you index with these bash shell commands:

```
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
```

Read More

Let me walk you through that:

- `export es=localhost:9200` sets up `$es` to be Elasticsearch's address. I do that so I don't have to type `localhost:9200` over and over again. And it'll make copying and pasting my text easier if your Elasticsearch doesn't listen on 9200. Win/win.
- `export site=en.wikiquote.org` sets up `$site` to be the hostname of the MediaWiki instance that you want to use. For English Wikipedia you'd use `en.wikipedia.org`. For German Wikipedia you'd use `de.wikipedia.org`. For Wikimedia Commons, you'd use `commons.wikimedia.org`.
- `export index=enwikiquote` just sets `$index` to the name of the index you'll be loading. I usually use the same name as the wiki.
- `curl -XDELETE $es/$index?pretty` deletes the index if it already exists. It fails if the index doesn't exist but we don't care. You'll see this at the start of most curl issue reproductions filed with Elasticsearch because it ensures a clean slate.
- The next sequence of pipes fetches the settings that en.wikiquote.org uses for its index and uses them as a template to create a new index in our test cluster. We don't just copy all of the configuration because we want to force ourselves to 0 replicas and 1 shard because we're doing this test on one machine. Feel free to change this if you have more machines in your test Elasticsearch cluster.
- The last sequence of pipes fetches that mapping for the content index, upgrades it from 1.7 style to 2.x style, and creates a `page` type with that mapping in our test index.

Clear as mud? Ok.

Step 3: Prepare the wiki for loading

Crack open the file with zless and have a look at it. Don't worry, you can do that before it finishes downloading. Its contents are conveniently in the format that Elasticsearch uses for bulk loading. Hurray! We'll unzip and cut this file into smaller chunks so it'll work properly with curl and the `_bulk` API. It'll also you a way to pause and resume the process. After it has finished downloading you can do that with these bash shell commands:

```
export dump=enwikiquote-20160201-cirrussearch-
content.json.gz
export index=enwikiquote

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

- The first `export` line just names the file that you downloaded. Change it to whatever file you fetched. The second export line is the same as one used in the previous section.
- The `mkdir` and `cd` lines make a directory to hold the files we're going to make.
- 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.

For English Wikiquote that should finish in a few seconds. English Wikipedia takes longer than a coffee break.

Step 4: Load the wiki

The last step is to load the actual data with these bash commands:

```
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
```

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.

I find setting the `refresh_interval` to `-1` will speed this process up some. You can apply it by running this in a different terminal:

```
curl -XPUT "$es/$index/_settings?pretty" -d '{
    "index" : {
        "refresh_interval" : -1
    }
}'
```

You can monitor the progress with:

```
date; curl $es/$index/_refresh?pretty; curl
$es/$index/_count?pretty
```

The `_refresh` makes the current process visible regardless of the `refresh_interval`. The `_count` just counts how many documents you've loaded.

Loading English Wikiquote is an excellent opportunity to have a coffee break. Loading English Wikipedia takes all night. When it is all done you should flush the index with `curl $es/$index/_flush` to make sure that the next time you restart Elasticsearch it doesn't have a large translog to replay.

Step 5: Now do it with a bigger wiki

Now that you've loaded a smaller wiki you can just repeate the process with other `export` statements to load a larger wiki. While loading English Wikiquote took something like 20 minutes total you'll find loading English Wikipedia to be an overnight job.

Regarding space: I find that gzip gives English Wikipedia about a 1:6 compession ratio and when the Elasticsearch index is comlete it's only slightly bigger. Since the load process removes that files after they are loaded into Elasticsearch it should be safe if you budget 10 time the file size for this process. When I last loaded English Wikipedia the Elasticsearch index ended up being 7 times the size of the gzip and I kept the gzip on disk in case I wanted to load it again. Its small enough not to be a big deal on a spinning disk but large enough to be a pain on an SSD.

Step 6: Now what?

After you've loaded the index you can do whatever you want with it: presumably you'll want to test some queries or

something. You can do that now. You'll notice a huge performance boost if you `_optimize?max_num_segments=1` but that is cheating to some degree because it causes trouble if you ever modify the index. For testing maybe it's fine but it doesn't simulate a system that is constantly being updated.