Project Report

Colin Legge, Jacob Ericson, Suqian Wang

Our goal for this milestone is to build a search engine for the 131k documents we crawled in task 1. We used Solr, a full-text search server based on Lucene, the library because Solr has JSON Interfaces which is perfect for our documents.

First, we worked on setting up the Solr server:

- download Solr from http://apache.mesi.com.ar/lucene/solr/7.2.1/
- launch Solr from bin folder: ./solr start
- create a core in order to be index and search and assigned a port number for it
- use bin/post tool and indexed our documents on the server

Then, we tested our Solr server by performing searching operation on it:

- searching all documents

http://localhost:8983/solr/SpotifyProject/select?q=*:*


```
{
  "responseHeader":{
    "status":0,
    "QTime":220,
    "params":{
      "q":"*:*",
      "_":"1521586659352"}},
  "response":{"numFound":131243,"start":0,"docs":[
      {
        "added_at":["2015-12-16T16:16:36Z"],
        "added_by.external_urls.spotify":["https://open.spotify.com/user/cabal76"],
        "added_by.href":["https://api.spotify.com/v1/users/cabal76"],
```


- searching certain artist

http://localhost:8983/solr/SpotifyProject/select?q=track.artists.name:"Katy Perry"


```
{
  "responseHeader":{
    "status":0,
    "QTime":612,
    "params":{
      "q":"track.artists.name:\"Katy Perry\"",
      "_":"1521586659352"}},
  "response":{"numFound":141,"start":0,"docs":[
      {
        "added_at":["2010-12-16T17:06:39Z"],
        "track.album.album_type":["single"],
        "track.album.artists.external_urls.spotify":["https://open.spotify.com/artist/6jJ0s89eD6GaHleKKya26X"],
        "track.album.artists.href":["https://api.spotify.com/v1/artists/6jJ0s89eD6GaHleKKya26X"],
        "track.album.artists.id":["6jJ0s89eD6GaHleKKya26X"],
        "track.album.artists.name":["Katy Perry"],
        "track.album.artists.type":["artist"],
        "track.album.artists.uri":["spotify:artist:6jJ0s89eD6GaHleKKya26X"],
```

- searching a certain song

http://localhost:8983/solr/SpotifyProject/select?q=track.name:"I knew you were trouble"


```
{
  "responseHeader":{
    "status":0,
    "QTime":938,
    "params":{
      "q":"track.name:\"I knew you were trouble\"",
      "_":"1521586659352"}},
  "response":{"numFound":3,"start":0,"docs":[
      {
        "added_at":["2017-10-24T14:49:32Z"],
        "track.album.album_type":["album"],
        "track.album.artists.external_urls.spotify":["https://open.spotify.com/artist/06HL4z0CvFAxyc27GXpf02"],
        "track.album.artists.href":["https://api.spotify.com/v1/artists/06HL4z0CvFAxyc27GXpf02"],
        "track.album.artists.id":["06HL4z0CvFAxyc27GXpf02"],
        "track.album.artists.name":["Taylor Swift"],
        "track.album.artists.type":["artist"],
        "track.album.artists.uri":["spotify:artist:06HL4z0CvFAxyc27GXpf02"],
```


- searching using wild-card query

http://localhost:8983/solr/SpotifyProject/select?q=track.artists.name:ade*


```
{
  "responseHeader":{
    "status":0,
    "QTime":431,
    "params":{
      "q":"track.artists.name:ade*",
      "_":"1521586659352"}},
  "response":{"numFound":187,"start":0,"docs":[
      {
        "added_at":["2017-03-22T20:29:28Z"],
        "track.album.album_type":["single"],
        "track.album.artists.external_urls.spotify":["https://open.spotify.com/artist/4dpARuHxo51G3z768sgnrY"],
        "track.album.artists.href":["https://api.spotify.com/v1/artists/4dpARuHxo51G3z768sgnrY"],
        "track.album.artists.id":["4dpARuHxo51G3z768sgnrY"],
        "track.album.artists.name":["Adele"],
        "track.album.artists.type":["artist"],
```


During the process, we ran into a few problems:
In the first task, we got 131k documents each represent a different song's data. When trying to indexing all of them, the server gives error saying argument list is too long. Therefore, we have to separate them and repeat the indexing command. However, indexing 131k documents is not very time efficient, so we tried to use one big JSON file to save the Solr indexing time, but we failed due Solr's file size limit. To solve this problem, we get a smaller amount of documents which group the information by playlists which inspired us to create two nodes, one for all documents with different songs and the other for different playlists that contain the same song information. Another problem is, when we indexing our files, the number of indexed file didn't match the number of documents on our core, it turns out that only some of them will show on our core, this happened when we tried to index all those playlists documents. We don't know the reason yet except for suspecting it has something to do with the date format.