Assignment 2 Report:
Simple Search Engine using Hadoop MapReduce

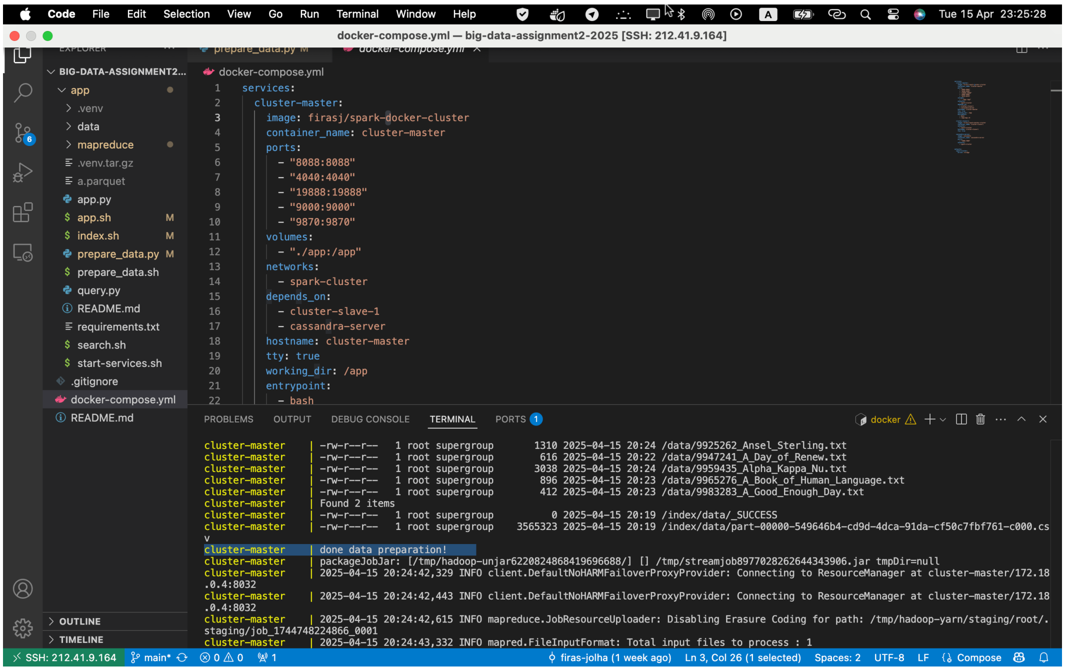
Liana Mardanova
DS-01
l.mardanova@innopolis.university

S25 - Big Data Course
April 15, 2025

Overview

I created a simple search engine using MapReduce for indexing, Cassandra for storing statistics and Spark RDD for ranking using BM25. Initially, I prepared 1000 documents (from a.parquet) and stored them in HDFS. Afterwards, I ran MapReduce to get the info to calculate BM25 for the query and stored it in Cassandra. Next, I used PySpark to read this data and calculate BM25. Each part is described in more detail below.

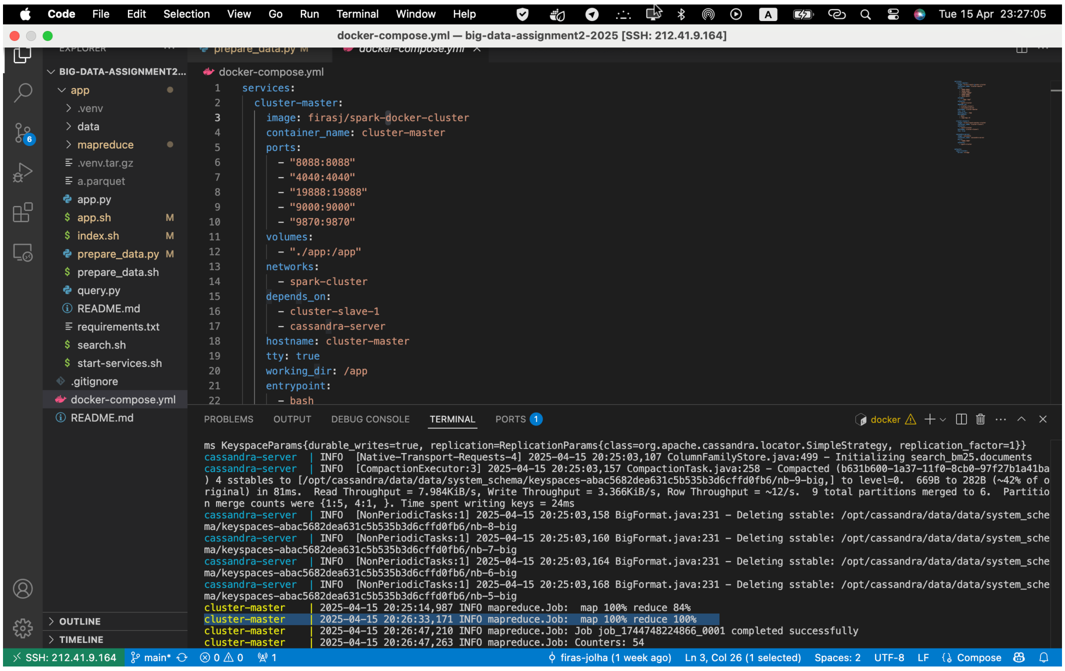
Demo



```
services:
  cluster-master:
    image: firasj/spark-docker-cluster
    container_name: cluster-master
    ports:
      - "8088:8088"
      - "4040:4040"
      - "19888:19888"
      - "9000:9000"
      - "9870:9870"
    volumes:
      - "./app:/app"
    networks:
      - spark-cluster
    depends_on:
      - cluster-slave-1
      - cassandra-server
    hostname: cluster-master
    tty: true
    working_dir: /app
    entrypoint:
      - bash
```

```
cluster-master    | -rw-r--r--   1 root supergroup       1310 2025-04-15 20:24 /data/9925262_Ansel_Sterling.txt
cluster-master    | -rw-r--r--   1 root supergroup        616 2025-04-15 20:22 /data/9947241_A_Day_of_Renew.txt
cluster-master    | -rw-r--r--   1 root supergroup       3038 2025-04-15 20:24 /data/9959435_Alpha_Kappa_Nu.txt
cluster-master    | -rw-r--r--   1 root supergroup        896 2025-04-15 20:23 /data/9965276_A_Book_of_Human_Language.txt
cluster-master    | -rw-r--r--   1 root supergroup        412 2025-04-15 20:23 /data/9983283_A_Good_Enough_Day.txt
cluster-master    | Found 2 items
cluster-master    | -rw-r--r--   1 root supergroup          0 2025-04-15 20:19 /index/data/_SUCCESS
cluster-master    | -rw-r--r--   1 root supergroup    3565323 2025-04-15 20:19 /index/data/part-00000-549646b4-cd9d-4dca-91da-cf50c7fbf761-c000.cs
v
cluster-master    | done data preparation!
cluster-master    | packageJobJar: [/tmp/hadoop-unjar6220824868419696688/] [] /tmp/streamjob8977028262644343906.jar tmpDir=null
cluster-master    | 2025-04-15 20:24:42,329 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at cluster-master/172.18
.0.4:8032
cluster-master    | 2025-04-15 20:24:42,443 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at cluster-master/172.18
.0.4:8032
cluster-master    | 2025-04-15 20:24:42,615 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/root/.
staging/job_1744748224866_0001
cluster-master    | 2025-04-15 20:24:43,332 INFO mapred.FileInputFormat: Total input files to process : 1
```

Figure 1: Data preparation



```
services:
  cluster-master:
    image: firasj/spark-docker-cluster
    container_name: cluster-master
    ports:
      - "8088:8088"
      - "4040:4040"
      - "19888:19888"
      - "9000:9000"
      - "9870:9870"
    volumes:
      - "./app:/app"
    networks:
      - spark-cluster
    depends_on:
      - cluster-slave-1
      - cassandra-server
    hostname: cluster-master
    tty: true
    working_dir: /app
    entrypoint:
      - bash
```

```
ms KeyspaceParams{durable_writes=true, replication=ReplicationParams{class=org.apache.cassandra.locator.SimpleStrategy, replication_factor=1}}
cassandra-server  | INFO  [Native-Transport-Requests-4] 2025-04-15 20:25:03,107 ColumnFamilyStore.java:499 - Initializing search_bm25.documents
cassandra-server  | INFO  [CompactionExecutor:3] 2025-04-15 20:25:03,157 CompactionTask.java:258 - Compacted (b631b600-1a37-11f0-8cb0-97f27b1a41ba
) 4 sstables to [/opt/cassandra/data/data/system_schema/keyspaces-abac5682dea631c5b535b3d6cffd0fb6/nb-9-big,] to level=0.  669B to 282B (~42% of o
riginal) in 81ms.  Read Throughput = 7.984KiB/s, Write Throughput = 3.366KiB/s, Row Throughput = ~12/s.  9 total partitions merged to 6.  Partitio
n merge counts were {1:5, 4:1, }. Time spent writing keys = 24ms
cassandra-server  | INFO  [NonPeriodicTasks:1] 2025-04-15 20:25:03,158 BigFormat.java:231 - Deleting sstable: /opt/cassandra/data/data/system_sche
ma/keyspaces-abac5682dea631c5b535b3d6cffd0fb6/nb-8-big
cassandra-server  | INFO  [NonPeriodicTasks:1] 2025-04-15 20:25:03,160 BigFormat.java:231 - Deleting sstable: /opt/cassandra/data/data/system_sche
ma/keyspaces-abac5682dea631c5b535b3d6cffd0fb6/nb-7-big
cassandra-server  | INFO  [NonPeriodicTasks:1] 2025-04-15 20:25:03,164 BigFormat.java:231 - Deleting sstable: /opt/cassandra/data/data/system_sche
ma/keyspaces-abac5682dea631c5b535b3d6cffd0fb6/nb-6-big
cassandra-server  | INFO  [NonPeriodicTasks:1] 2025-04-15 20:25:03,168 BigFormat.java:231 - Deleting sstable: /opt/cassandra/data/data/system_sche
ma/keyspaces-abac5682dea631c5b535b3d6cffd0fb6/nb-5-big
cluster-master    | 2025-04-15 20:25:14,987 INFO mapreduce.Job:  map 100% reduce 84%
cluster-master    | 2025-04-15 20:26:33,171 INFO mapreduce.Job:  map 100% reduce 100%
cluster-master    | 2025-04-15 20:26:47,210 INFO mapreduce.Job: Job job_1744748224866_0001 completed successfully
cluster-master    | 2025-04-15 20:26:47,263 INFO mapreduce.Job: Counters: 54
```

Figure 2: Indexing 1000 documents using MapReduce

MapReduce

Mapper (`mapper1.py`) accepts input in the format `<doc_id> <doc_title> <doc_text>` separated by tabs. It tokenizes the text, counts the total number of tokens, and uses a counter to compute term frequencies. It gives three outputs: document info (`doc_id`, `title`, `doc_len`), document frequency (`term`, `doc_id`), and term frequency (`term`, `doc_id`, `tf`).

Reducer (`reducer1.py`) reads this output, does the final aggregation, and saves the results into Cassandra tables.

Cassandra Tables

I saved the tables in this format so that it would be convenient to calculate bm25 for query later.

term_frequency

No.	Column Name	Type
1	term	TEXT
2	doc_id	TEXT
3	tf	INT
Primary Key: (term, doc_id)		

Table 1: Schema of `term_frequency` table

document_frequency

No.	Column Name	Type
1	term	TEXT
2	df	INT
Primary Key: (term)		

Table 2: Schema of `document_frequency` table

documents

No.	Column Name	Type
1	doc_id	TEXT
2	title	TEXT
3	doc_len	INT
Primary Key: (doc_id)		

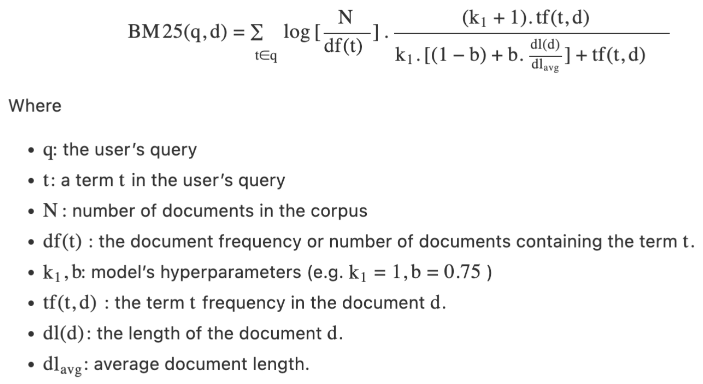
Table 3: Schema of `documents` table

Query and Ranking

I wrote a script to find top 10 for a query. It connects to Cassandra and first reads all the documents to get their names and length. This is needed to calculate N and the average length of the document.

It then looks up each query term and gets how many documents it occurs in (df) and how often it occurs in each document (tf). After that, it calculates BM25 for each document considering all the query terms.

At the end it shows the top 10 documents with the highest scores. The formula I used is shown below.



$$\mathrm{BM25}(q,d) = \sum_{t \in q} \log\left[\frac{N}{df(t)}\right] . \frac{(k_1 + 1).\, tf(t,d)}{k_1 . [(1-b) + b.\frac{dl(d)}{dl_{avg}}] + tf(t,d)}$$

Where

- q: the user's query
- t: a term t in the user's query
- N : number of documents in the corpus
- $df(t)$: the document frequency or number of documents containing the term t.
- k_1, b: model's hyperparameters (e.g. $k_1 = 1, b = 0.75$)
- $tf(t,d)$: the term t frequency in the document d.
- $dl(d)$: the length of the document d.
- dl_{avg}: average document length.

Figure 3: BM25 scoring formula used for ranking

Conclusion

This was an interesting task because it combined many tools like Hadoop, Cassandra, and Spark. I got a chance to work with all of them together and see how they connect in a real pipeline.

Since it's a real Big Data task, I faced problems related to memory. My local machine ran out of memory, so I had to run everything on a remote server. It wasn't easy, but I learned a lot during the process.