

Course	MSc in Data Science (CW_KCDAR_M Y5)	Lecturer	Michael Gleeson
Module	Infrastructure for BigData (DATAC5201)	Student	Gokul Thillainathan
Assignment	: Big Data Solution – CA3		

HADOOP MULTINODE CLUSTER WITH SPARK

1. PROBLEM STATEMENT:

The Motto of this is to develop and implement a scalable and fault-tolerant infrastructure for data processing and analysis. As a result, it will improve

- The performance by supporting parallel and distributed computing for higher performance while managing larger datasets.
- Ensure the availability to continuously access the data even during node failures through multiple nodes.
- Improved response time for both batch processing and real-time data analytics.

The solution addresses the limitation of old single server design, providing improved scalability, reliability and processing efficiently the datasets.

2. SOLUTION ARCHITECTURE:

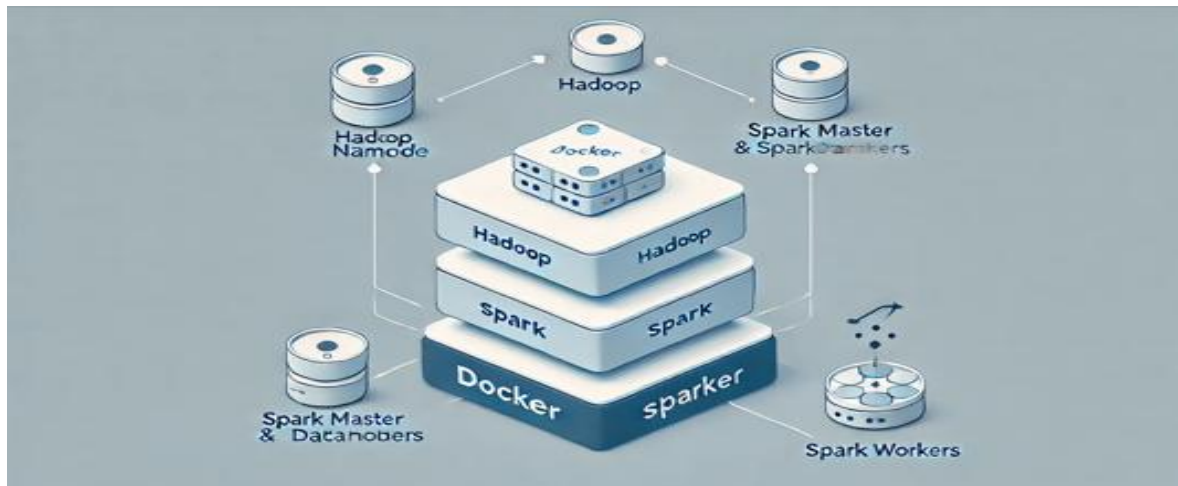
To address the issue, I have used Hadoop Ecosystem (HDFS, SPARK, ZEPPELIN) on a Docker-based setup with following components:

Hadoop Cluster: A multinode HDFS cluster to handle data.

Spark: For processing the data.

Zeppelin: for data visualization and executing query.

Docker: Containers to deploy Hadoop, Spark, and Zeppelin services.



3. INFRASTRUCTURE DETAILS:

Docker Configuration:

The Hadoop cluster was deployed using docker compose with the following services:

- Namenode: Master node managing metadata
- Datanode: Two nodes for data storage
- Spark Master: Controller for spark jobs
- Spark Workers: for processing (2 nodes)
- Zeppelin: for running spark queries and visualizations

4. PREREQUISITES:

--> Docker and Docker compose installed on the host machine.

--> Dataset: quakes-cleaned.csv (Earthquake data) stored locally before ingestion.

5. STEP-BY-STEP IMPLEMENTATION:

Step 1:

- Deploy the Hadoop, spark, and zeppelin services using Docker compose:
docker-compose up -d
- Verify that all the containers are running:
docker ps

Step 2:

- Copy dataset to Namenode container:
`docker cp /path/quakes-cleaned.csv namenode:/quakes-cleaned.csv`
- Exec the namenode container:
`docker exec -it namenode bash`
- Upload the dataset to HDFS:
`hdfs dfs -put /quakes-cleaned.csv /iot_data/`
- Verify the upload:
`hdfs dfs -ls /iot_data/`

Step 3:

- Open the Zeppelin web interface (<http://localhost:8085>) and create a new notebook.
- Load the dataset into spark
- Perform aggregation (average magnitude by location)

Step 4:

- Save the aggregated data.
- Verify the data

6. RESULTS:

Starting container in the docker:

```
PS C:\Users\Administrator\Desktop\iot_bigdata_infrastructure> docker-compose start
[+] Running 7/7
✓Container spark-master   Started      3.0s
✓Container namenode       Started      3.2s
✓Container zeppelin       Started      6.8s
✓Container spark-worker2  Started      4.1s
✓Container spark-worker1  Started      6.5s
✓Container datanode1      Started      6.0s
✓Container datanode2      Started      6.1s
PS C:\Users\Administrator\Desktop\iot_bigdata_infrastructure>
```

Container status:

```
PS C:\Users\Administrator\Desktop\iot_bigdata_infrastructure> docker ps
```

CONTAINER ID	IMAGE	NAMES	COMMAND	CREATED	STATUS	PORTS
eda69365c700	apache/zeppelin:0.9.0	zeppelin	"/usr/bin/tini -- bi..."	26 hours ago	Up 2 minutes	0.0.0.0:8085->8080/tcp
178dc3c41a12	bde2020/spark-worker:2.4.0-hadoop2.7	spark-worker1	"/bin/bash /worker.sh"	26 hours ago	Up 2 minutes	0.0.0.0:8081->8081/tcp
fcef26726af6	bde2020/spark-worker:2.4.0-hadoop2.7	spark-worker2	"/bin/bash /worker.sh"	26 hours ago	Up 2 minutes	0.0.0.0:8082->8081/tcp
73b9766c050b	bde2020/hadoop-datanode:2.0.0-hadoop2.7.4-java8	datanode1	"/entrypoint.sh /run..."	26 hours ago	Up 2 minutes (healthy)	0.0.0.0:9001->50075/tcp
b4d2073957e6	bde2020/hadoop-datanode:2.0.0-hadoop2.7.4-java8	datanode2	"/entrypoint.sh /run..."	26 hours ago	Up 2 minutes (healthy)	0.0.0.0:9002->50075/tcp
5439699dc055	bde2020/spark-master:2.4.0-hadoop2.7	spark-master	"/bin/bash /master.sh"	26 hours ago	Up 2 minutes	0.0.0.0:7077->7077/tcp, 6066/tcp, 0.0.0.0:8080->8080/tcp
aeb389f6997b	bde2020/hadoop-namenode:2.0.0-hadoop2.7.4-java8	namenode	"/entrypoint.sh /run..."	26 hours ago	Up 2 minutes (healthy)	0.0.0.0:8020->8020/tcp, 0.0.0.0:9000->50070/tcp

Accessing the webpage:

Hadoop Namenode: <http://localhost:9000/>

The screenshot shows a web browser window with the address bar displaying 'localhost:9000/dfshealth.html#tab-overview'. The page has a green header with navigation tabs: 'Hadoop', 'Overview' (selected), 'Datanodes', 'Datanode Volume Failures', 'Snapshot', 'Startup Progress', and 'Utilities'. The main content area is titled 'Overview' and shows details for 'namenode:8020' (active). Below this is a table with key information:

Started:	Wed Dec 11 22:38:42 UTC 2024
Version:	2.7.4, rcd915e1e8d9d0131462a0b7301586c175728a282
Compiled:	2017-08-01T00:29Z by kshvachk from branch-2.7.4
Cluster ID:	CID-346af108-c118-4839-a7e0-58820ae944aa
Block Pool ID:	BP-354424951-172.18.0.3-1733855726891

Below the table is a 'Summary' section with the following text:

Security is off.
Safemode is off.
407 files and directories, 401 blocks = 808 total filesystem object(s).
Heap Memory used 77.73 MB of 218 MB Heap Memory. Max Heap Memory is 889 MB.
Non Heap Memory used 41.08 MB of 41.75 MB Committed Non Heap Memory. Max Non Heap Memory is -1 B.

At the bottom, there is a table showing capacity and usage:

Configured Capacity:	1.97 TB
DFS Used:	8.02 MB (0%)

Spark: <http://localhost:8088/>

The screenshot shows the Spark Master web interface. At the top, there's a header with the Spark logo and version 2.4.0, followed by the title 'Spark Master at spark://5439699dc055:7077'. Below this, a URL is provided: 'URL: spark://5439699dc055:7077'. A summary of cluster status is shown: 'Alive Workers: 2', 'Cores in use: 16 Total, 0 Used', 'Memory in use: 3.8 GB Total, 0.0 B Used', 'Applications: 0 Running, 0 Completed', 'Drivers: 0 Running, 0 Completed', and 'Status: ALIVE'. A section titled 'Workers (2)' contains a table with two rows of worker information. Below that, 'Running Applications (0)' and 'Completed Applications (0)' are shown, each with an empty table header.

Worker Id	Address	State	Cores	Memory
worker-20241211223843-172.18.0.4-44091	172.18.0.4:44091	ALIVE	8 (0 Used)	1920.0 MB (0.0 B Used)
worker-20241211223843-172.18.0.7-35715	172.18.0.7:35715	ALIVE	8 (0 Used)	1920.0 MB (0.0 B Used)

Application ID	Name	Cores	Memory per Executor	Submitted Time	User	State	Duration
----------------	------	-------	---------------------	----------------	------	-------	----------

Application ID	Name	Cores	Memory per Executor	Submitted Time	User	State	Duration
----------------	------	-------	---------------------	----------------	------	-------	----------

Zeppelin: <http://localhost:8085/>

The screenshot shows the Zeppelin web interface. The header includes the Zeppelin logo, 'Notebook' and 'Job' tabs, a search bar, and a user indicator 'anonymous'. The main content area has a 'Welcome to Zeppelin!' message, followed by a brief description of Zeppelin as a web-based notebook. On the left, there's a 'Notebook' section with links to 'Import note' and 'Create new note', and a list of notebooks including 'Flink Tutorial', 'Miscellaneous Tutorial', 'Python Tutorial', 'R Tutorial', 'Spark Tutorial', and 'Big_data'. On the right, there's a 'Help' section with links to 'Get started with Zeppelin documentation' and 'Community', and a 'Mailing list' link. A large blue whale illustration is on the right side of the page.

Verify Dataset in the container:

```
PS C:\Users\Administrator\Desktop\iot_bigdata_infrastructure> docker exec -it namenode bash
root@aeb389f6997b:/# hdfs dfs -ls /iot_data/
Found 3 items
drwxr-xr-x - zeppelin zeppelin      0 2024-12-10 19:12 /iot_data/aggregated_quakes_data
drwxr-xr-x - zeppelin zeppelin      0 2024-12-10 21:15 /iot_data/filtered_quakes_data
-rwxrwxr-x  3 zeppelin zeppelin 845738 2024-12-10 18:56 /iot_data/quakes-cleaned.csv
root@aeb389f6997b:/# |
```

Code to load dataset to Spark:

```
%spark
```

```
val csvDF = spark.read.option("header", "true")
                        .option("inferSchema", "true")
                        .csv("hdfs://namenode:8020/iot_data/quakes-cleaned.csv")
```

```
csvDF.show(truncate = false)
```

Code to perform filter:

```
%spark
```

```
import org.apache.spark.sql.functions.avg
```

```
val aggDF = csvDF.groupBy("place")
                  .agg(avg("mag").alias("avg_mag"))
aggDF.show(truncate = false)
```

Save the processed data back to HDFS:

```
%spark
```

```
aggDF.write.option("header",  
"true").csv("hdfs://namenode:8020/iot_data/aggregated_quakes_data")
```

Verify the dataset:

```
hdfs dfs -ls /iot_data/aggregated_quakes_data
```

```
root@aeb389f6997b:/# hdfs dfs -ls /iot_data/aggregated_quakes_data  
Found 201 items  
-rw-r--r-- 3 zeppelin zeppelin      0 2024-12-10 19:12 /iot_data/aggregated_quakes_data/_SUCCESS  
-rw-r--r-- 3 zeppelin zeppelin 1813 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00000-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv  
-rw-r--r-- 3 zeppelin zeppelin 1142 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00001-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv  
-rw-r--r-- 3 zeppelin zeppelin 654 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00002-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv  
-rw-r--r-- 3 zeppelin zeppelin 768 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00003-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv  
-rw-r--r-- 3 zeppelin zeppelin 1100 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00004-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv  
-rw-r--r-- 3 zeppelin zeppelin 1083 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00005-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv  
-rw-r--r-- 3 zeppelin zeppelin 782 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00006-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv  
-rw-r--r-- 3 zeppelin zeppelin 1115 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00007-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv  
-rw-r--r-- 3 zeppelin zeppelin 415 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00008-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv  
-rw-r--r-- 3 zeppelin zeppelin 788 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00009-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv  
-rw-r--r-- 3 zeppelin zeppelin 782 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00010-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv  
-rw-r--r-- 3 zeppelin zeppelin 938 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00011-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv  
-rw-r--r-- 3 zeppelin zeppelin 1139 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00012-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv  
-rw-r--r-- 3 zeppelin zeppelin 768 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00013-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv  
-rw-r--r-- 3 zeppelin zeppelin 902 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00014-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv  
-rw-r--r-- 3 zeppelin zeppelin 591 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00015-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv  
-rw-r--r-- 3 zeppelin zeppelin 1024 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00016-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv  
-rw-r--r-- 3 zeppelin zeppelin 1298 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00017-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv  
-rw-r--r-- 3 zeppelin zeppelin 803 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00018-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv  
-rw-r--r-- 3 zeppelin zeppelin 890 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00019-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv  
-rw-r--r-- 3 zeppelin zeppelin 1197 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00020-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv  
-rw-r--r-- 3 zeppelin zeppelin 874 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00021-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv  
-rw-r--r-- 3 zeppelin zeppelin 479 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00022-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv  
-rw-r--r-- 3 zeppelin zeppelin 1171 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00023-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv  
-rw-r--r-- 3 zeppelin zeppelin 919 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00024-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv  
-rw-r--r-- 3 zeppelin zeppelin 701 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00025-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv  
-rw-r--r-- 3 zeppelin zeppelin 521 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00026-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv  
-rw-r--r-- 3 zeppelin zeppelin 750 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00027-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv  
-rw-r--r-- 3 zeppelin zeppelin 903 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00028-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv  
-rw-r--r-- 3 zeppelin zeppelin 1197 2024-12-10 19:12 /iot_data/aggregated_quakes_data/part-00029-6d77f4ac-0834-4123-aac7-2491cdecdc07-c000.csv
```

7. CHALLENGES AND SOLUTIONS:

--> Permission issues: Resolved by setting appropriate permissions on HDFS directories.

--> Replication Factor: Ensured data replication across both datanodes.

8. FUTURE ENHANCEMENTS:

- * Automating the data Ingestion.
- * Real-time processing using spark streaming.
- * Integrating jupyter Notebook for advanced workflows.

9. CONCLUSION:

In this project successfully implemented a scalable and fault-tolerant Big Data infrastructure using Hadoop, Spark, and Zeppelin. The infrastructure also lays the base for the future enhancements which includes real-time data processing and advanced data analytics.

10. REFERENCES:

https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html

<https://www.tutorialkart.com/apache-spark/how-to-setup-an-apache-spark-cluster/>

<https://princetonits.com/blog/technology/how-to-configure-replication-factor-and-block-size-for-hdfs/>

<https://spark.apache.org/docs/latest/spark-standalone.html>

<https://medium.com/@ARishi/optimizing-apache-spark-executors-for-improved-performance-067eea2349e2>