Step 1: Unzip the Required Files

Before proceeding, extract the contents of question5.zip, which contains the necessary files for this task. Run:

```
unzip question5.zip
```

This will extract the dataset and required resources into the current directory.

Step 2: Start Hadoop Services

Make sure Hadoop is running before proceeding. Start the Distributed File System (HDFS) and the Resource Manager (YARN):

```
start-dfs.sh
start-yarn.sh
```

This ensures that file storage and resource management are active.

Step 3: Prepare HDFS Directories

Create necessary directories in HDFS:

1. **Create a directory for storing the 10K text files**:

```
hadoop fs -mkdir /10000
```

2. **Create a directory for storing stopwords** (used to filter out common words like "the", "is", etc.):

```
hdfs dfs -mkdir -p /user/abhay/assignment2/stopword/
```

Step 4: Upload Files to HDFS

Move the required files from your local system to HDFS:

1. **Upload all 10,000 text files (dataset for analysis)**:

```
hadoop fs -put *.txt /10000/
```

2. **Upload the stopwords file (used for filtering out common words)**:

```
hdfs dfs -put stopwords.txt /user/abhay/assignment2/stopword/stopwords.txt
```

Step 5: Build the Project

Before running the job, compile and package the Java code into a JAR file:

```
mvn clean package assembly:single
```

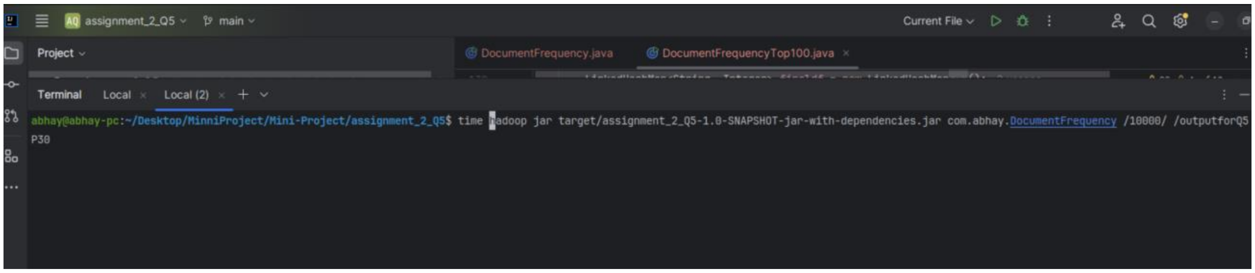
This ensures all dependencies are included in a single JAR, making execution smoother.

Step 6: Run the Hadoop Jobs

Now, execute the MapReduce jobs one by one.

Job 1: Compute Document Frequency

```
hadoop jar target/assignment_2_Q5-1.0-SNAPSHOT-jar-with-dependencies.jar \
com.abhay.DocumentFrequency /10000/ /outputforQ5P30
```



```
abhay@abhay-pc:~/Desktop/MinniProject/Mini-Project/assignment_2_Q5$ time hadoop jar target/assignment_2_Q5-1.0-SNAPSHOT-jar-with-dependencies.jar com.abhay.DocumentFrequency /10000/ /outputforQ5
P30
```

What it does:

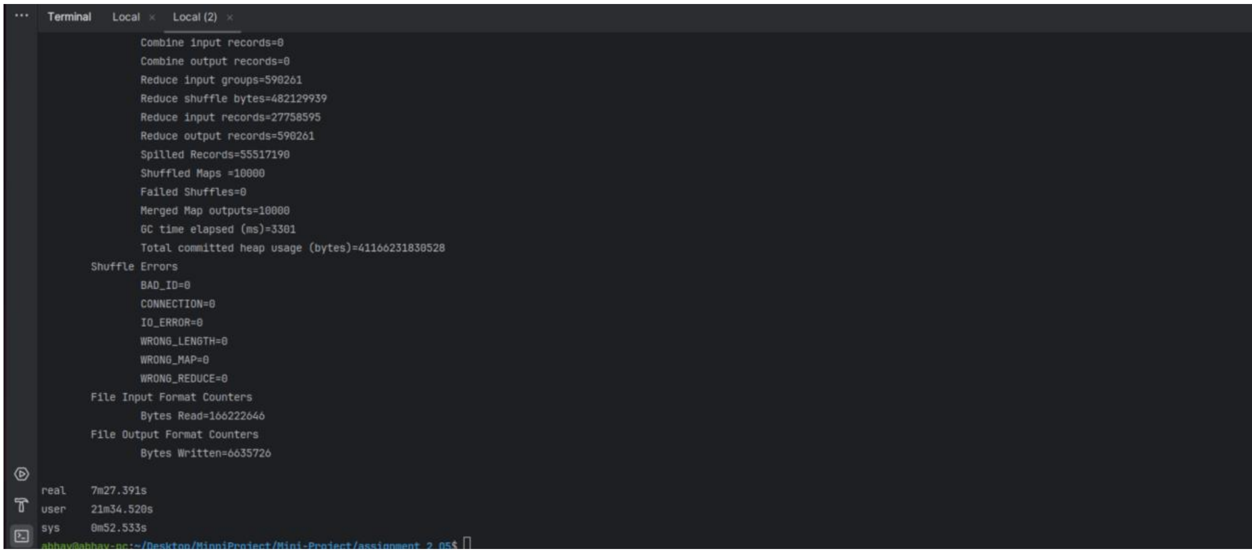
- Processes all 10K text files.
- Computes how many documents contain each word (Document Frequency).
- Stores the output in HDFS at /outputforQ5P30.

1. **Mapper Function:**
   a. First, we **tokenize** each sentence into individual words.

   b. Next, we **eliminate stopwords** to remove common words that do not contribute much meaning.
   c. The remaining words are **converted to lowercase** and then passed through the **Porter-Stemmer algorithm** to get their root form.
   d. For each stemmed word, we generate a **key-value pair** in the format: **(word, document ID)**.

2. **Reducer Function:**
   a. The reducer receives each **unique word** as the key and a list of document IDs where it appears.
   b. We create a **set** of document IDs to ensure **each document is counted only once** for that word.
   c. The total count of unique document IDs in the set represents the **document frequency** of the word.
   d. The final output is formatted as **(word, document frequency)**.

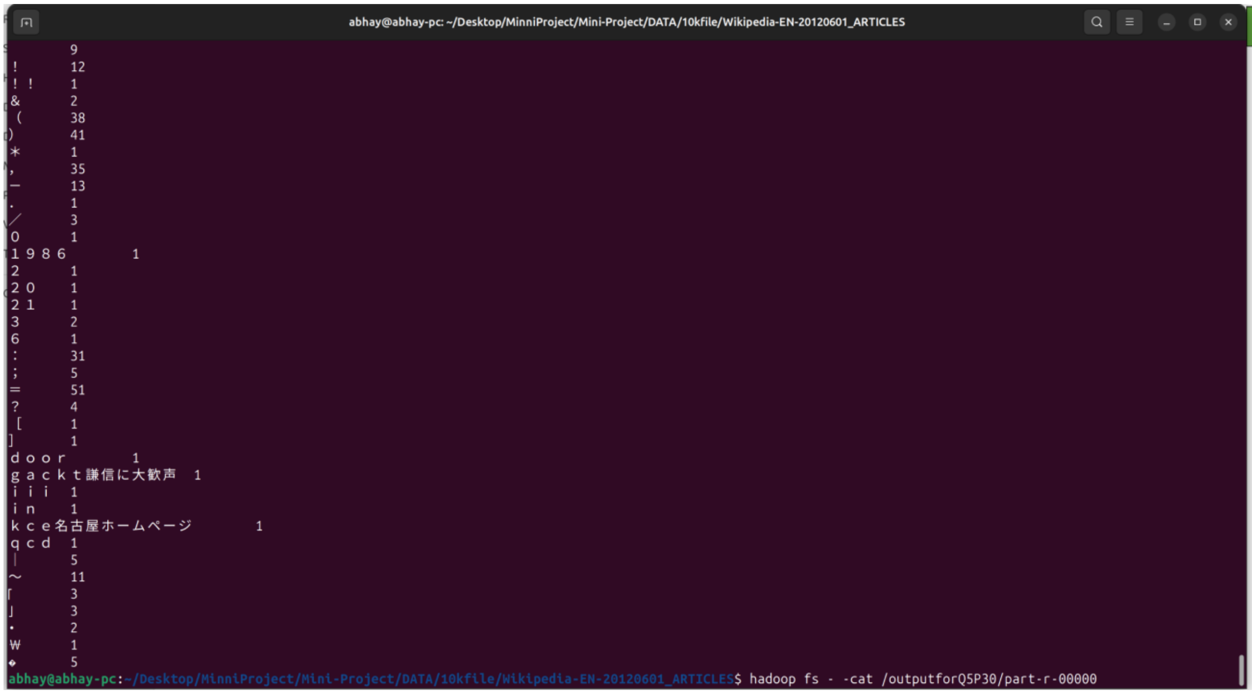
3. **Preprocessing Considerations:**
   a. **Punctuation removal** was **not** applied during preprocessing.
   b. Due to this, variations like **"work"** and **"work."** appeared separately in the results.



```
                Combine input records=0
                Combine output records=0
                Reduce input groups=590261
                Reduce shuffle bytes=482129939
                Reduce input records=27758595
                Reduce output records=590261
                Spilled Records=55517190
                Shuffled Maps =10000
                Failed Shuffles=0
                Merged Map outputs=10000
                GC time elapsed (ms)=3301
                Total committed heap usage (bytes)=41166231830528
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=166222646
        File Output Format Counters
                Bytes Written=6635726

real    7m27.391s
user    21m34.520s
sys     0m52.533s
abhay@abhay-pc:~/Desktop/MinniProject/Mini-Project/assignment_2_Q5$
```

Completed time

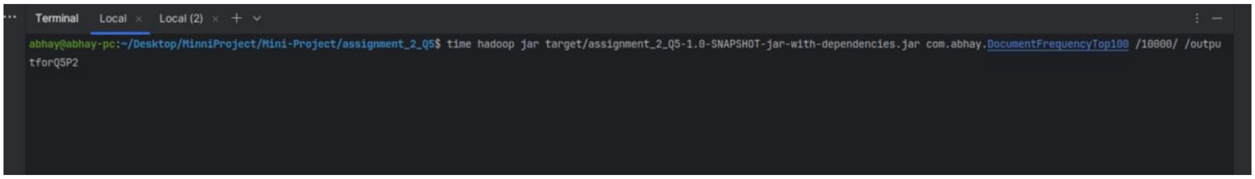

abhay@abhay-pc: ~/Desktop/MinniProject/Mini-Project/DATA/10kfile/Wikipedia-EN-20120601_ARTICLES

```
　      9
！      12
！！    1
＆      2
（      38
）      41
＊      1
，      35
－      13
．      1
／      3
０      1
１９８６        1
２      1
２０    1
２１    1
３      2
６      1
：      31
；      5
＝      51
？      4
［      1
］      1
ｄｏｏｒ        1
ｇａｃｋｔ謙信に大歓声  1
ｉｉｉ  1
ｉｎ    1
ｋｃｅ名古屋ホームページ        1
ｑｃｄ  1
｜      5
～      11
「      3
」      3
・      2
￦      1
�       5
abhay@abhay-pc:~/Desktop/MinniProject/Mini-Project/DATA/10kfile/Wikipedia-EN-20120601_ARTICLES$ hadoop fs - -cat /outputforQ5P30/part-r-00000
```

Output

Job 2: Extract the Top 100 Frequent Terms

hadoop jar target/assignment_2_Q5-1.0-SNAPSHOT-jar-with-dependencies.jar \
com.abhay.DocumentFrequencyTop100 /50/ /outputforQ5P2



```
abhay@abhay-pc:~/Desktop/MinniProject/Mini-Project/assignment_2_Q5$ time hadoop jar target/assignment_2_Q5-1.0-SNAPSHOT-jar-with-dependencies.jar com.abhay.DocumentFrequencyTop100 /10000/ /outpu
tforQ5P2
```

What it does:

- Reads a filtered dataset (possibly from previous output).
- Extracts the **top 100 most frequently occurring words**.
- Saves the output to HDFS at /outputforQ5P2.

1. **Mapper Function:**
    a. The same **stemming** and **preprocessing** steps used in the previous task are applied here.
    b. We utilize the **stripes approach**, where we generate key-value pairs in the format:
    (Document Name + Term, Term Frequency).
2. **Reducer Function:**

a. Before processing, the **setup function** is executed to read the TSV file produced in the earlier step.

b. This file contains **terms and their document frequencies**, which are stored in a **hashmap** for quick access.

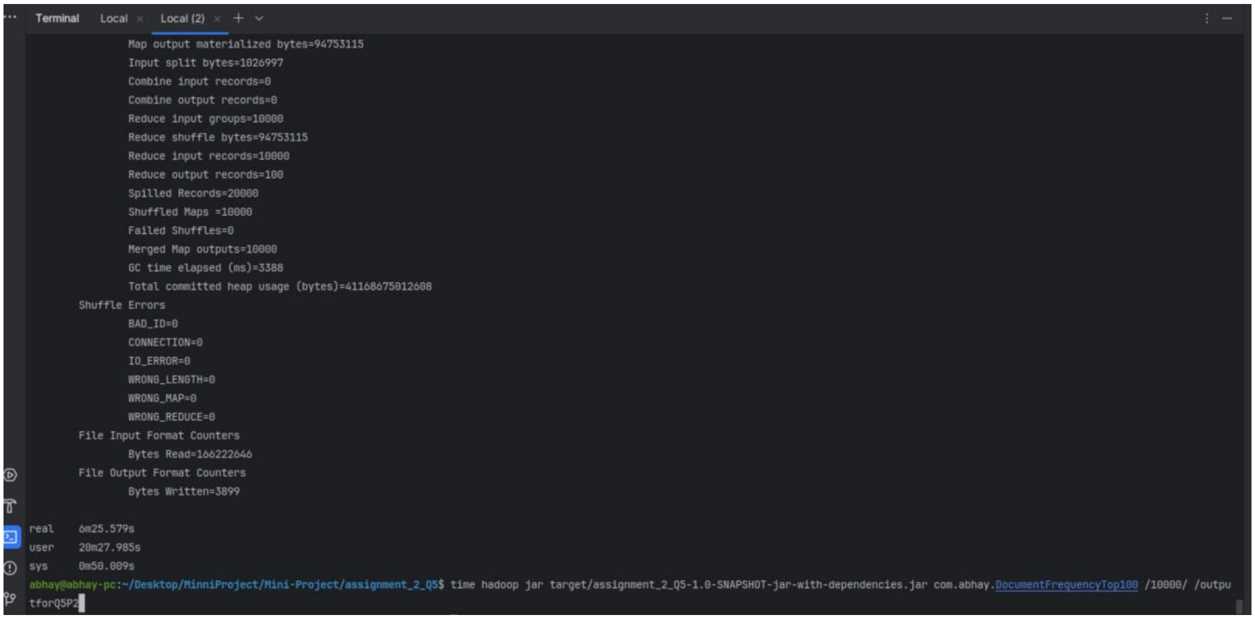
c. In the **reduce function**, we aggregate the total **term frequency** for each term-document pair.

d. Using this data, we compute a **score** for each term-document combination.

e. The final output is formatted as: **(Document Name + Term, Score)**.

3. **Output Format:**

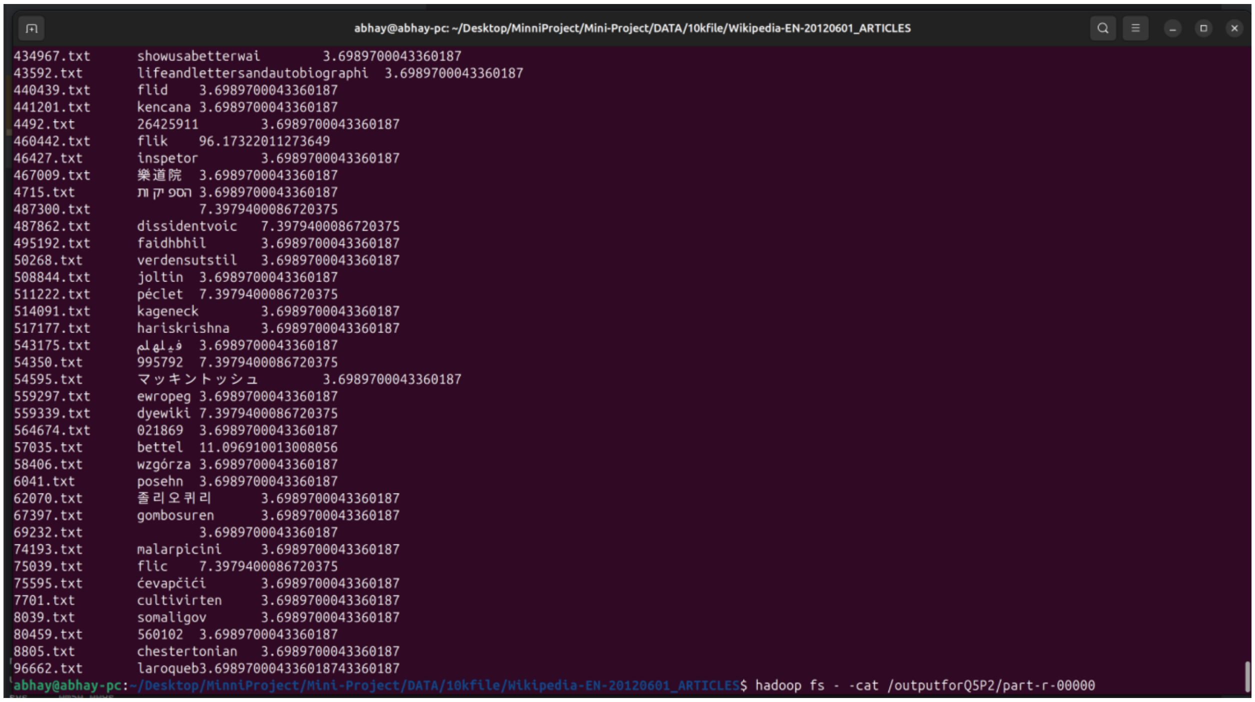
a. We noticed that the **default output format is tab-separated**, so there is **no need for additional formatting**.



```
		Map output materialized bytes=94753115
		Input split bytes=1026997
		Combine input records=0
		Combine output records=0
		Reduce input groups=10000
		Reduce shuffle bytes=94753115
		Reduce input records=10000
		Reduce output records=100
		Spilled Records=20000
		Shuffled Maps =10000
		Failed Shuffles=0
		Merged Map outputs=10000
		GC time elapsed (ms)=3388
		Total committed heap usage (bytes)=41168675012608
	Shuffle Errors
		BAD_ID=0
		CONNECTION=0
		IO_ERROR=0
		WRONG_LENGTH=0
		WRONG_MAP=0
		WRONG_REDUCE=0
	File Input Format Counters
		Bytes Read=166222646
	File Output Format Counters
		Bytes Written=3899

real	6m25.579s
user	20m27.985s
sys	0m50.009s
abhay@abhay-pc:~/Desktop/MinniProject/Mini-Project/assignment_2_Q5$ time hadoop jar target/assignment_2_Q5-1.0-SNAPSHOT-jar-with-dependencies.jar com.abhay.DocumentFrequencyTop100 /10000/ /outpu
tforQ5P2
```

Completion Time



abhay@abhay-pc: ~/Desktop/MinniProject/Mini-Project/DATA/10kfile/Wikipedia-EN-20120601_ARTICLES

```
434967.txt      showusabetterwai        3.6989700043360187
43592.txt       lifeandlettersandautobiographi  3.6989700043360187
440439.txt      flid    3.6989700043360187
441201.txt      kencana 3.6989700043360187
4492.txt        26425911        3.6989700043360187
460442.txt      flik    96.17322011273649
46427.txt       inspetor        3.6989700043360187
467009.txt      樂道院  3.6989700043360187
4715.txt        הספ יק ות 3.6989700043360187
487300.txt              7.3979400086720375
487862.txt      dissidentvoic   7.3979400086720375
495192.txt      faidhbhil       3.6989700043360187
50268.txt       verdensutstil   3.6989700043360187
508844.txt      joltin  3.6989700043360187
511222.txt      péclet  7.3979400086720375
514091.txt      kageneck        3.6989700043360187
517177.txt      hariskrishna    3.6989700043360187
543175.txt      فيلهلم  3.6989700043360187
54350.txt       995792  7.3979400086720375
54595.txt       マッキントッシュ        3.6989700043360187
559297.txt      ewropeg 3.6989700043360187
559339.txt      dyewiki 7.3979400086720375
564674.txt      021869  3.6989700043360187
57035.txt       bettel  11.096910013008056
58406.txt       wzgórza 3.6989700043360187
6041.txt        posehn  3.6989700043360187
62070.txt       졸리오퀴리      3.6989700043360187
67397.txt       gombosuren      3.6989700043360187
69232.txt               3.6989700043360187
74193.txt       malarpicini     3.6989700043360187
75039.txt       flic    7.3979400086720375
75595.txt       ćevapčići       3.6989700043360187
7701.txt        cultivirten     3.6989700043360187
8039.txt        somaligov       3.6989700043360187
80459.txt       560102  3.6989700043360187
8805.txt        chestertonian   3.6989700043360187
96662.txt       laroqueb3.698970004336018743360187
abhay@abhay-pc:~/Desktop/MinniProject/Mini-Project/DATA/10kfile/Wikipedia-EN-20120601_ARTICLES$ hadoop fs - -cat /outputforQ5P2/part-r-00000
```

OutPut