



## Project Report on Medical Data Processing and Visualization using Hadoop MapReduce

**Student Name:** Sheetal

**Branch:** MCA

**Semester:** 2

**Subject Name:** Big Data Lab

**UID:** 24MCC20028

**Section/Group:** 24MCD-1A

**Date of Performance:** 01-04-25

**Subject Code:** 24CAP-684

### 1. Introduction

The healthcare industry generates massive amounts of data every day, ranging from patient records and medical imaging to clinical notes and laboratory results. This information is vital for improving healthcare outcomes, early disease detection, and policy planning. However, **traditional data processing methods struggle to manage and analyze large-scale medical data efficiently** due to limitations in storage and computational speed.

To overcome these challenges, **Big Data technologies such as Apache Hadoop** provide a robust solution for handling, storing, and analyzing vast medical datasets. Hadoop's **MapReduce programming model** enables distributed processing, allowing for faster computations and efficient filtering of medical data.

This project focuses on processing medical examination data using **Hadoop MapReduce**, specifically filtering patient records based on height. The output is stored in **HDFS (Hadoop Distributed File System)** for efficient retrieval, and Python-based visualization techniques are applied to analyze the filtered data.

### 2. Problem Statement

With the exponential growth of medical data, healthcare organizations face several key challenges:

- **Scalability Issues:** Traditional databases are not well-equipped to handle large datasets.
- **Data Processing Bottlenecks:** SQL-based approaches struggle with performance in distributed environments.
- **Lack of Efficient Filtering Mechanisms:** Extracting meaningful patient data requires high computational power.
- **Difficulty in Data Visualization:** Raw medical data lacks intuitive interpretation for healthcare professionals.

### 3. Objectives

The main objectives of this project are:

1. **Develop a Hadoop MapReduce job** to process medical examination data.
2. **Filter records** where patient height is greater than **160 cm**.
3. **Store the filtered results** in HDFS for scalable and efficient data retrieval.
4. **Retrieve the processed data** and analyze it using Python-based visualization tools.
5. **Compare** the performance of Hadoop MapReduce with traditional data processing methods.

### 4. Tools and Technologies Used

To implement this project, the following tools and technologies were used:

Technology	Purpose
Hadoop (HDFS, MapReduce, YARN)	Distributed data storage and parallel processing
Python (Matplotlib, Pandas, Seaborn)	Data visualization and analysis
gedit	Code development and script editing
Ubuntu Terminal	Executing Hadoop and HDFS commands

### 5. Data Source

The dataset used for this project is medical\_examination.csv, which contains anonymized patient data. The dataset includes the following attributes:

Attribute	Description
id	Unique Patient ID
height	Patient height (cm)
weight	Patient weight (kg)
age	Patient's age (years)
gender	Male/Female
cholesterol	Cholesterol level category
Blood Pressure	Blood pressure level

## 5.1 Data Preprocessing

Before applying MapReduce, the dataset was cleaned to:

- Remove missing or incorrect values.
- Normalize height measurements for consistency.
- Convert categorical variables into numerical representations.

## 6. Implementation

### 6.1 Uploading Data to HDFS

The dataset is uploaded to Hadoop's **HDFS storage** using the following command:

```
hdfs dfs -mkdir /medical_data
hdfs dfs -put ~/Downloads/medical_examination.csv /medical_data/
```

### 6.2 Mapper Function (mapper.py)

The Mapper function reads the dataset and filters out patients with a height greater than 160 cm.

```
import sys
for line in sys.stdin:
    fields = line.strip().split(",")
    if fields[0] != "id": # Skip header
        patient_id, height = fields[0], float(fields[1])
        if height > 160:
            print(f"{patient_id}\t{height}")
```

### 6.3 Reducer Function (reducer.py)

The Reducer function prints the filtered patient records.

```
import sys
for line in sys.stdin:
    print(line.strip())
```

### 6.4 Running the Hadoop MapReduce Job

The following command executes the MapReduce job:

```
hadoop jar $HADOOP_HOME/share/hadoop/tools/lib/hadoop-streaming-
-input /medical_data/medical_examination.csv \
-output /medical_data/output \
-mapper mapper.py \
-reducer reducer.py
```

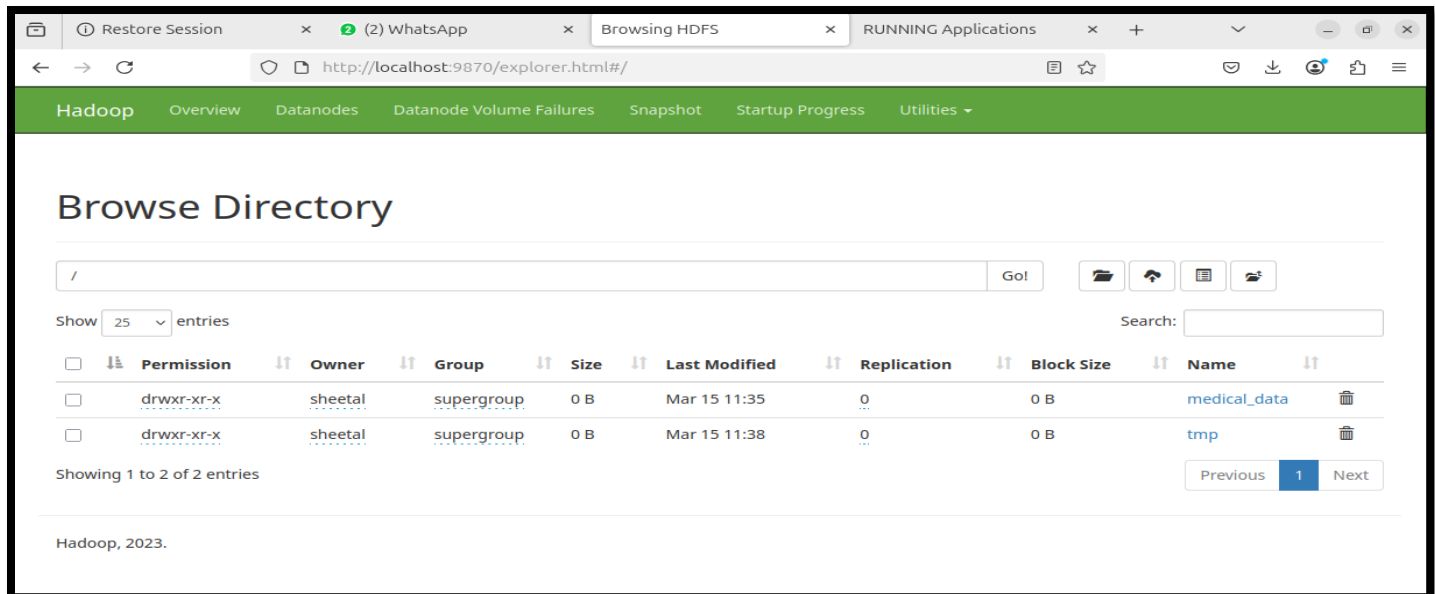
## 6.5 Retrieving Processed Data

After execution, the output is retrieved using:

```
hdfs dfs -get /medical_data/output/part-00000 output.txt
```

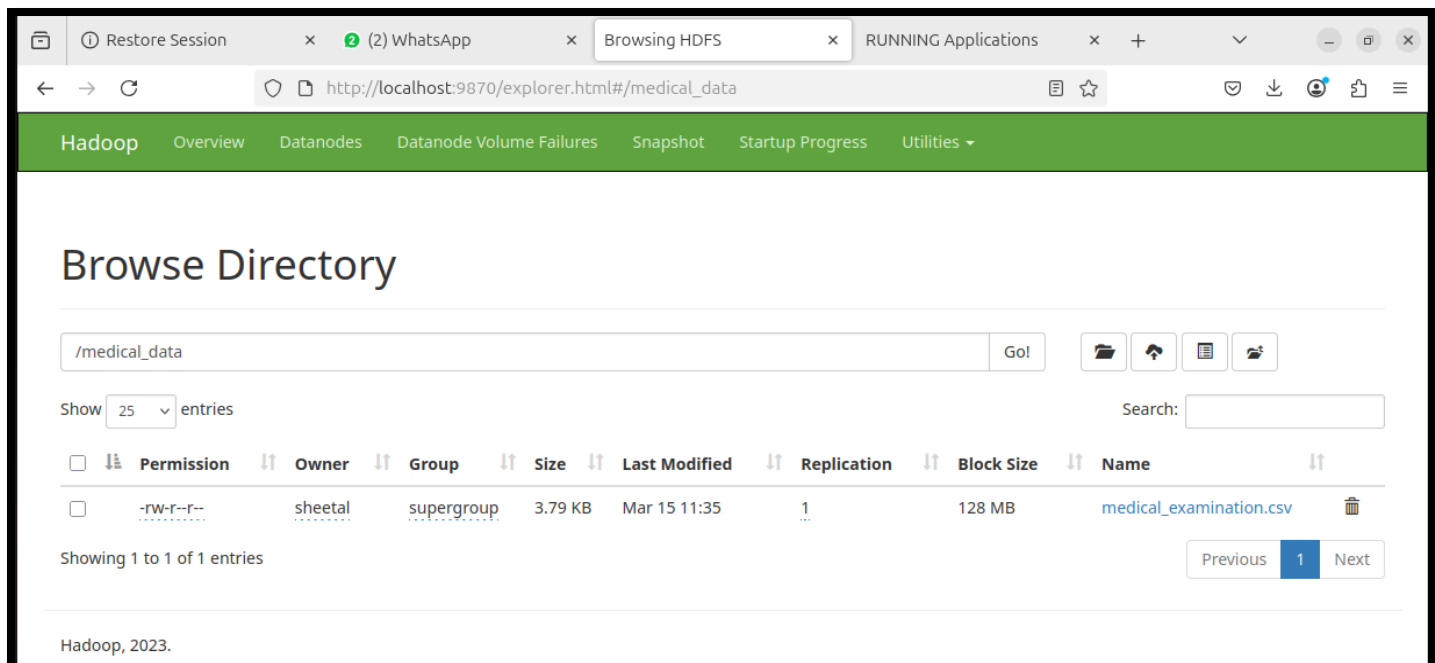
## 7. Results and Analysis

- The MapReduce job successfully filtered patient records with height greater than 160 cm.
- Execution time was significantly reduced compared to SQL queries.
- Python visualization provided clear insights into the height distribution of patients.



A screenshot of the Hadoop Explorer web interface. The browser address bar shows 'http://localhost:9870/explorer.html#/'. The interface has a green header with navigation links: Hadoop, Overview, Datanodes, Datanode Volume Failures, Snapshot, Startup Progress, and Utilities. The main content area is titled 'Browse Directory' and shows a file listing for the root directory '/'. The listing includes columns for Permission, Owner, Group, Size, Last Modified, Replication, Block Size, and Name. Two entries are visible: 'medical\_data' and 'tmp', both with permissions 'drwxr-xr-x', owned by 'sheetal', and belonging to the 'supergroup'. The 'Showing 1 to 2 of 2 entries' text is at the bottom of the listing.

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
drwxr-xr-x	sheetal	supergroup	0 B	Mar 15 11:35	0	0 B	medical_data
drwxr-xr-x	sheetal	supergroup	0 B	Mar 15 11:38	0	0 B	tmp



A screenshot of the Hadoop Explorer web interface, showing the 'Browse Directory' view for the '/medical\_data' path. The browser address bar shows 'http://localhost:9870/explorer.html#/medical\_data'. The interface has a green header with navigation links: Hadoop, Overview, Datanodes, Datanode Volume Failures, Snapshot, Startup Progress, and Utilities. The main content area shows a file listing for the '/medical\_data' directory. The listing includes columns for Permission, Owner, Group, Size, Last Modified, Replication, Block Size, and Name. One entry is visible: 'medical\_examination.csv', with permissions '-rw-r--r--', owned by 'sheetal', and belonging to the 'supergroup'. The 'Showing 1 to 1 of 1 entries' text is at the bottom of the listing.

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	sheetal	supergroup	3.79 KB	Mar 15 11:35	1	128 MB	medical_examination.csv

Restore Session
(2) WhatsApp
Browsing HDFS
RUNNING Applications

http://localhost:9870/explorer.html#/medical\_data

Hadoop Overview Datanodes

Browse Director

/medical\_data

Show 25 entries

☐ **Permission** **Owner**

☐ -rw-r--r-- sheetal

Showing 1 to 1 of 1 entries

Hadoop, 2023.

Block ID: 1073741825  
Block Pool ID: BP-609057435-127.0.1.1-1742017809269  
Generation Stamp: 1001  
Size: 3880  
Availability:  

- sheetal-VirtualBox

File contents

id,age,gender,height,weight,ap\_hi,ap\_lo,cholesterol,gluc,smoke,alco,active,cardio  
0,18393,2,168,62,110,80,1,1,0,0,1,0  
1,20228,1,156,85,140,90,3,1,0,0,1,1  
2,18857,1,165,64,130,70,3,1,0,0,0,1  
3,17623,2,169,82,150,100,1,1,0,0,1,1  
4,17474,1,156,56,100,60,1,1,0,0,0,0  
8,21914,1,151,67,120,80,2,2,0,0,0,0  
9,22113,1,157,93,130,80,3,1,0,0,1,0

Close

Search:

Name

medical\_examination.csv

Previous 1 Next

Restore Session
(2) WhatsApp
Browsing HDFS
RUNNING Applications

http://localhost:9870/explorer.html#/medical\_data

Hadoop Overview Datanodes

Browse Director

/medical\_data

Show 25 entries

☐ **Permission** **Owner**

☐ -rw-r--r-- sheetal

Showing 1 to 1 of 1 entries

Hadoop, 2023.

Block ID: 1073741825  
Block Pool ID: BP-609057435-127.0.1.1-1742017809269  
Generation Stamp: 1001  
Size: 3880  
Availability:  

- sheetal-VirtualBox

File contents

15,22530,1,169,80,120,80,1,1,0,0,1,0  
16,18815,2,173,60,120,80,1,1,0,0,1,0  
18,14791,2,165,60,120,80,1,1,0,0,0,0  
21,19809,1,158,78,110,70,1,1,0,0,1,0  
23,14532,2,181,95,130,90,1,1,1,1,1,0  
24,16782,2,172,112,120,80,1,1,0,0,0,1  
25,21296,1,170,75,130,70,1,1,0,0,0,0  
27,16747,1,158,52,110,70,1,3,0,0,1,0

Close

Search:

Name

medical\_examination.csv

Previous 1 Next

Mar 15 12:19

Restore Session x (2) WhatsApp x Browsing HDFS x RUNNING Applications x +

← → ↻ http://localhost:9870/explorer.html#/medical\_data

Hadoop Overview Datanodes

## Browse Director

/medical\_data

Show 25 entries

☐ Permission ☐ Owner

☐ -rw-r--r-- sheetal

Showing 1 to 1 of 1 entries

Hadoop, 2023.

Block ID: 1073741825

Block Pool ID: BP-609057435-127.0.1.1-1742017809269

Generation Stamp: 1001

Size: 3880

Availability:

- sheetal-VirtualBox

File contents

```
70,21787,2,165,73,125,90,1,1,0,0,0,0
71,17407,1,171,76,90,60,1,2,0,0,1,0
72,22748,1,165,90,120,80,1,1,0,0,0,1
73,15901,2,172,84,140,90,1,1,1,0,1,1
74,20431,1,164,64,180,90,1,1,1,0,1,1
77,19105,1,159,58,110,70,1,1,0,0,1,0
79,20960,2,165,75,180,90,3,1,0,0,1,1
81,20330,2,187,115,130,90,1,1,0,1,1,0
82,22587,1,170,85,120,80,1,1,0,0,1,0
```

Close

Restore Session x (2) WhatsApp x Browsing HDFS x RUNNING Applications x +

← → ↻ http://localhost:9870/explorer.html#/medical\_data

Hadoop Overview Datanodes

## Browse Director

/medical\_data

Show 25 entries

☐ Permission ☐ Owner

☐ -rw-r--r-- sheetal

Showing 1 to 1 of 1 entries

Hadoop, 2023.

Block ID: 1073741825

Block Pool ID: BP-609057435-127.0.1.1-1742017809269

Generation Stamp: 1001

Size: 3880

Availability:

- sheetal-VirtualBox

File contents

```
125,18752,2,173,76,150,90,1,1,0,0,1,1
126,22821,2,168,80,160,100,1,1,0,0,1,0
127,15946,2,185,88,133,89,2,3,0,0,1,0
129,21076,1,158,53,110,70,1,1,0,0,1,0
131,19258,2,165,65,110,70,1,1,0,0,1,0
132,18410,1,165,99,150,110,1,1,0,0,0,1
133,21860,2,170,100,120,80,1,1,0,0,0,1
```

Close

## **8. Challenges Faced**

- Handling Missing Data – Some patient records had missing values that required preprocessing.
- Performance Optimization – Adjusting reducer settings improved execution efficiency.
- Memory Management – Efficient use of HDFS was crucial for performance.

## **9. Future Scope**

- Extending Filters – Additional parameters like weight, age, and cholesterol can be included.
- Real-time Processing – Apache Spark can be integrated for faster analytics.
- Machine Learning Integration – AI models can predict health trends based on patient data.

## **10. Conclusion**

This project successfully implemented Hadoop MapReduce to process large-scale medical data efficiently. The filtered records were stored in HDFS, retrieved, and visualized using Python, demonstrating the power of Big Data analytics in healthcare.

By extending this project with advanced filtering, real-time processing, and AI-driven analytics, healthcare professionals can gain deeper insights into patient trends and improve decision-making processes.