

Ex.No: 11
Date: 20.10.25

Writing to Sequence Files in Hadoop

Aim:

To write key-value pairs into a Hadoop SequenceFile format and verify the content using Hadoop APIs.

Procedure:

1. Create a Java Program to Write SequenceFile:

2. Compile and Create JAR:

```
javac -classpath $(hadoop classpath) SequenceFileWriterExample.java
```

```
jar cf seqwriter.jar SequenceFileWriterExample*.class
```

3. Run the Program:

```
hadoop jar seqwriter.jar SequenceFileWriterExample
```

4. Verify Output:

Use the hadoop fs -text command to view the binary contents in readable format:

```
hadoop fs -put sequencefile_output.seq /user/hadoop/
```

```
hadoop fs -text /user/hadoop/sequencefile_output.seq
```

Program:

```
import java.io.IOException;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.io.SequenceFile;
import org.apache.hadoop.io.SequenceFile.Writer;

public class SequenceFileWriterExample {
    public static void main(String[] args) throws IOException {
        Configuration conf = new Configuration();
        Path path = new Path("sequencefile_output.seq");

        IntWritable key = new IntWritable();
```

```
Text value = new Text();
SequenceFile.Writer writer = SequenceFile.createWriter(conf,
    Writer.file(path),
    Writer.keyClass(IntWritable.class),
    Writer.valueClass(Text.class));
for (int i = 0; i < 5; i++) {
    key.set(i);
    value.set("Value_" + i);
    writer.append(key, value);
}
writer.close();
System.out.println("Sequence file written successfully.");
}
```

Output:

```
0 Value_0
1 Value_1
2 Value_2
3 Value_3
4 Value_4
```

Result:

Successfully wrote and verified data in Hadoop's SequenceFile format using Java API.