

Nama : steven Susanto

Npm : 1306412035

1. Upload file

Pertama, membuat dir dan diupload file ke direktori hdfs dengan perintah

```
Hdfs fs -mkdir 1306412035
```

```
Hdfs fs -put access_log 1306412035/
```

2. Gunakan map-reduce untuk memecah file tersebut menjadi 2 file terpisah yang dikelompokkan berdasarkan status code. File ke-1 berisi hanya log yang memiliki status code 200 dan file ke-2 berisi log yang memiliki status code selain 200.

(file : split200.zip      Driver class : Split200Driver)

Pada Mapper dilakukan pemrosesan menjadi key (yang merupakan status http) dan value yang merupakan seluruh line value.

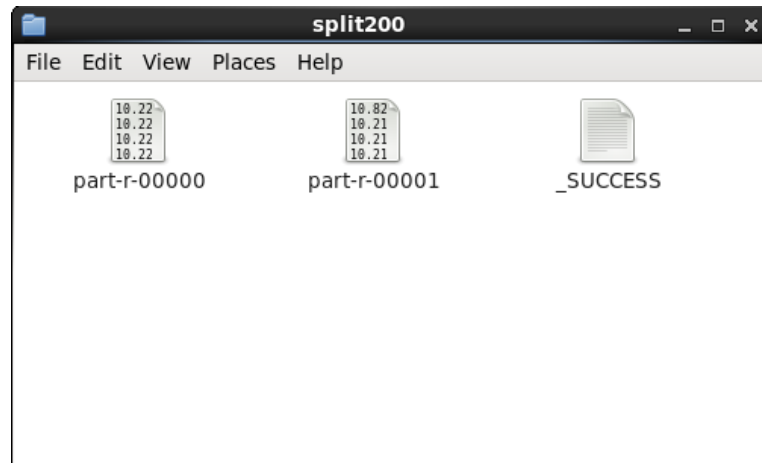
Cara pembagian file menjadi 2 part adalah dengan memanfaatkan partitioner, seperti pada cuplikan di bawah

```
public class Split200Partitioner extends Partitioner< IntWritable , Text> {  
    private int partitionNumber(IntWritable val){  
        if(val.get() == 200 ){  
            return 0;  
        }  
        else{  
            return 1;  
        }  
    }  
  
    public int getPartition( IntWritable key,Text value, int numReduceTasks) {  
        return partitionNumber(key);  
    }  
}
```

Pada Reducer hanya menyatukan Value untuk Key. Hasil Output hanya berupa value saja.

```
for(Text value : values){  
    context.write( value, NullWritable.get() );  
}
```

Hasil output dari 2 Reducer (job.setNumTaskReducer = 2) adalah 2 file dimana part-0000 berisi semua http status 200, dan file part-0001 berisi status selain 200.



3. Kembangkan map-reduce untuk menghitung total byte yang diakses oleh setiap alamat IP remote host. Pastikan hasilnya terurut berdasarkan total byte yang diakses dari besar ke kecil.

(file : bytecountbyhost.zip, Driver class : ByteCountDriver)

Penyelesaian dengan pertama kali dengan MapRed dari file ke pasangan key value {iphost , total byte}

Pada gambar di bawah potongan kode Mapper Stage 1

```
String[] fields = value.toString().split(" ");

if(fields.length > 3){
    //get the last index from the log line
    String s = fields[fields.length - 1];
    String s2 = fields[0];
    try{
        Long count = Long.parseLong(s);
        LongWritable bytecount = new LongWritable( count );
        Text hostip = new Text(s2);
        //write context
        context.write( hostip , bytecount );
    }catch (NumberFormatException nfe){
    }
}
```

Setelah di emit, pada Reducer dilakukan pengumpulan total count, dengan perintah seperti pada gambar di bawah

```
Long total = 0L;

for (LongWritable value : values){
    total += value.get();
}
context.write( key, new LongWritable(total) );
```

Langkah terakhir adalah melakukan inverse Map key value pada Mapper Stage 2 (SwitchMapper.class)

```
try{
    Long count = Long.parseLong(fields[1]);

    context.write(new LongWritable(count), new Text(fields[0]) );
}catch(NumberFormatException nfe){
}
}
```

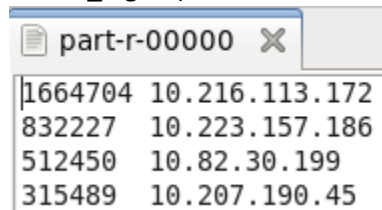
Agar terurut menurun (Descending), pada ByteCountDriver dilakukan penyesuaian seperti pada gambar di bawah

```
job.waitForCompletion(true);

Job job2 = new Job();
job2.setJarByClass(ByteCountDriver.class);
job2.setMapperClass(SwitchMapper.class);
job2.setSortComparatorClass(LongWritable.DecreasingComparator.class);
FileInputFormat.addInputPath(job2, new Path(args[0]+"*tmp"));
FileOutputFormat.setOutputPath(job2, new Path(args[1]));
```

Untuk Job2, dilakukan pengurutan dengan memanfaatkan Comparator Decreasing yang sudah diimplementasikan di LongWritable.

Contoh hasil output (file input mini\_log.txt)



```
part-r-00000 X
1664704 10.216.113.172
832227 10.223.157.186
512450 10.82.30.199
315489 10.207.190.45
```

4. Kembangkan map-reduce untuk menghasilkan daftar untuk setiap tanggal akses (per hari), alamat IP remote host yang paling banyak (dari jumlah akses, bukan ukuran total byte) mengakses server pada hari itu. Diurutkan per tanggal dari kecil ke besar.

(file : bydaycount.zip                      Driver class : ByDayCountDriver)

Penyelesaian digunakan CompositeKey dan MapRed 2 Stage. Pada Tahap pertama, terdapat CompositeKey seperti pada pseudocode berikut

```
CompositeKey implements WritableComparable{
    Text date;
    Text ip
}
}
```

Pada Mapper Stage 1 dilakukan emit key berupa key komposit dan hit count

```
context.write(new CompositeKey(date,ip), new IntWritable(1));
```

Pada Reducer Stage 1 dilakukan penjumlahan hit count

```
int total = 0;

for(IntWritable value : values){
    total += value.get();
}

context.write( key , new IntWritable(total) );
```

Untuk Stage 2, dilakukan pemisahan dengan Mapper ke CompositeKey2. Berikut pseudocode CompositeKey2

```
CompositeKey2 implements WritableComparable{

    Text ip
    IntWritable count;

    public int compareTo(CompositeKey2 pop)
    {
        return -1 * count.compareTo(pop.count);
    }
}
```

Metode compareTo dikalikan -1 agar terjadi pengurutan dari besar ke kecil (Descending).

Pada Mapper Stage 2 dilakukan pemindahan composite key

```
try{

    int count = Integer.parseInt(fields[2]);
    context.write(new Text(date), new CompositeKey2(ip,count) );

}
```

Pada Reducer Stage 2 dilakukan pengambilan value pertama untuk key, sehingga otomatis nilai yang diperoleh adalah terbesar, yaitu alamat ip dengan hit count terbesar untuk tanggal/ hari itu.

```
CompositeKey2 value;
for(CompositeKey2 val: values){
    value = val;
    context.write( key , value );
    break;
}
```

5. Kembangkan map-reduce untuk mencari file/URI yang memiliki status 200 apa yang paling banyak diakses di server tersebut? Apakah file tersebut merupakan file gambar (jpg)? Bila tidak, carilah file gambar (jpg) yang paling banyak diakses di server tersebut pula!

(file : filter200.zip

Driver class : Filter200Driver)

Penyelesaian menggunakan pendekatan Hive. Pertama file log di MapRed ke format

URL \t extension \t http\_status \t hit\_count

URL tanpa ekstensi menghasilkan ekstensi “-”.

URL index.php?id= menghasilkan ekstensi “-”.

Contoh hasil output

/	-	200	20	
/	-	403	1	
/about-us	-	301	4	
/about-us/	-	200	4	
/assets/css/960.css	css	200	8	
/assets/css/960.css	css	304	5	
/assets/css/reset.css	css	200	8	
/assets/css/reset.css	css	304	5	
/assets/css/the-associates.css	css	200	8	
/assets/css/the-associates.css	css	304	5	
/assets/img/about-us-logo.png	png	200	3	
/assets/img/closetlabel.gif	gif	200	8	
/assets/img/closetlabel.gif	gif	304	4	
/assets/img/contact-us-logo.png	png	200	2	

File dimap menggunakan filter200.zip, Driver class : Filter200Driver

Kemudian membuat table pada HIVE yang terhubung dengan lokasi file output pada HDFS dengan perintah seperti pada gambar di bawah

```
1 CREATE TABLE IF NOT EXISTS visitdaily
2 ( url STRING, ext STRING, status INT, hit INT)
3 ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'
4 LOCATION 'user/cloudera/b'
5
6 select a.url, a.ext, a.hit from visitdaily as a where a.hit in
7 (select max(b.hit) from visitdaily as b where b.status=200 );
8
9 select a.url, a.ext, a.hit from visitdaily as a where a.hit in
10 (select max(b.hit) from visitdaily as b where b.ext="jpg" );
11
```

Query untuk mencari URI dengan akses terbanyak dilakukan pada line ke 6 pada gambar di atas.

Pencarian JPG dengan hit terbanyak dilakukan dengan perintah pada line ke 9.

```
cloudera@quickstart:~/Desktop
File Edit View Search Terminal Help
> ;
OK
Time taken: 0.096 seconds
hive> CREATE EXTERNAL TABLE IF NOT EXISTS visitdaily(
>     url STRING, ext STRING, status INT, hit int)
>     ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'
>     LOCATION '/user/cloudera/b/';
OK
Time taken: 0.083 seconds
hive> select * from visitdaily;
OK
/      -      200      20
/      -      403      1
/about-us      -      301      4
/about-us/      -      200      4
/assets/css/960.css      css      200      8
/assets/css/960.css      css      304      5
/assets/css/reset.css      css      200      8
/assets/css/reset.css      css      304      5
/assets/css/the-associates.css      css      200      8
/assets/css/the-associates.css      css      304      5
/assets/img/about-us-logo.png      png      200      3
/assets/img/closetlabel.gif      gif      200      8
/assets/img/closetlabel.gif      gif      304      4
```

Pada gambar di atas contoh hasil import ke HIVE dan menampilkan isi data.

Pada gambar di bawah adalah contoh dari query

```
select a.url, a.ext, a.hit from visitdaily as a where a.hit in
(select max(b.hit) from visitdaily as b where b.ext="jpg" );
```

File yang menjadi input adalah mini\_log.txt (383 line pertama dari access\_log).

```
Stage: Stage 01 Map: 1 Cumulative CPU: 2.150 sec HDFS
Total MapReduce CPU Time Spent: 7 seconds 60 msec
OK
/assets/img/dummy/primary-news-1.jpg      200      8
/assets/img/dummy/primary-news-2.jpg      200      8
/assets/img/dummy/secondary-news-1.jpg      200      8
/assets/img/dummy/secondary-news-2.jpg      200      8
/assets/img/dummy/secondary-news-3.jpg      200      8
/assets/img/dummy/secondary-news-4.jpg      200      8
/assets/img/home-media-block-placeholder.jpg      200
Time taken: 52.492 seconds, Fetched: 7 row(s)
hive> █
```