

SAAVN MapReduce Project

Submitted By
Venkatesh Jagannathan

Table of Contents

Overview.....	3
Mapper.....	3
Combiner.....	3
Structure & Flow.....	5
Reducer.....	6
Algorithm Description.....	6
Sorting with Efficient Memory Usage & Single Reducer.....	7
Trending Song.....	7
Window.....	7
Spike.....	7
Slope – Main Weight Factor.....	7
Citations.....	9
Justifications.....	10
Why no Partitioner?.....	10
Won't PriorityQueue cause premature poll out of a song?.....	10
%Overlap with with Saavn's List.....	10
Execution Details.....	10
Steps to Generate Trend List.....	11
Console Output.....	13
Appendix.....	16

Overview

The solution identifies 100 trending songs from Dec 26th to 31st by analyzing data trend in the window of last 24 hours window by aggregating quarter day wise stream counts using regression analysis, variance measures, efficient memory usage, single map reduce (no chaining) & minimal processing.

The solution involves basic mapper, combiner & reducer to process large input data provided. Only single map reduce phase is used to obtain desired result. Combiner indeed aggregates map output to increase efficiency. Processing is highly efficient by means of

- (a) **Priority Queue:** Sort order of data is maintained alongside minimum memory usage with the help of priority queue. New songs are added to queue end and song with least weight makes it to top of priority. Upon crossing required chart length, the top priority song gets polled out to give way for higher trending songs.
- (b) **Minimum Data Transfer:** The x-coordinate of data point is basic integer called daytime with format DDHH DD=Day HH=Integer (0,25,50,75 corresponding to each day quarter) instead of date time which makes comparison & processing faster, aggregation simpler, time scale linear & intuitive. For e.g. **0950** = time range between 2017/12/09 12PM and 2017/12/09 6PM

Only a single reducer is used to process result. The processing time was 32 minutes & no errors were reported during processing. Failure conditions were validated and additional exception handled to keep code fail safe with logging.

Mapper

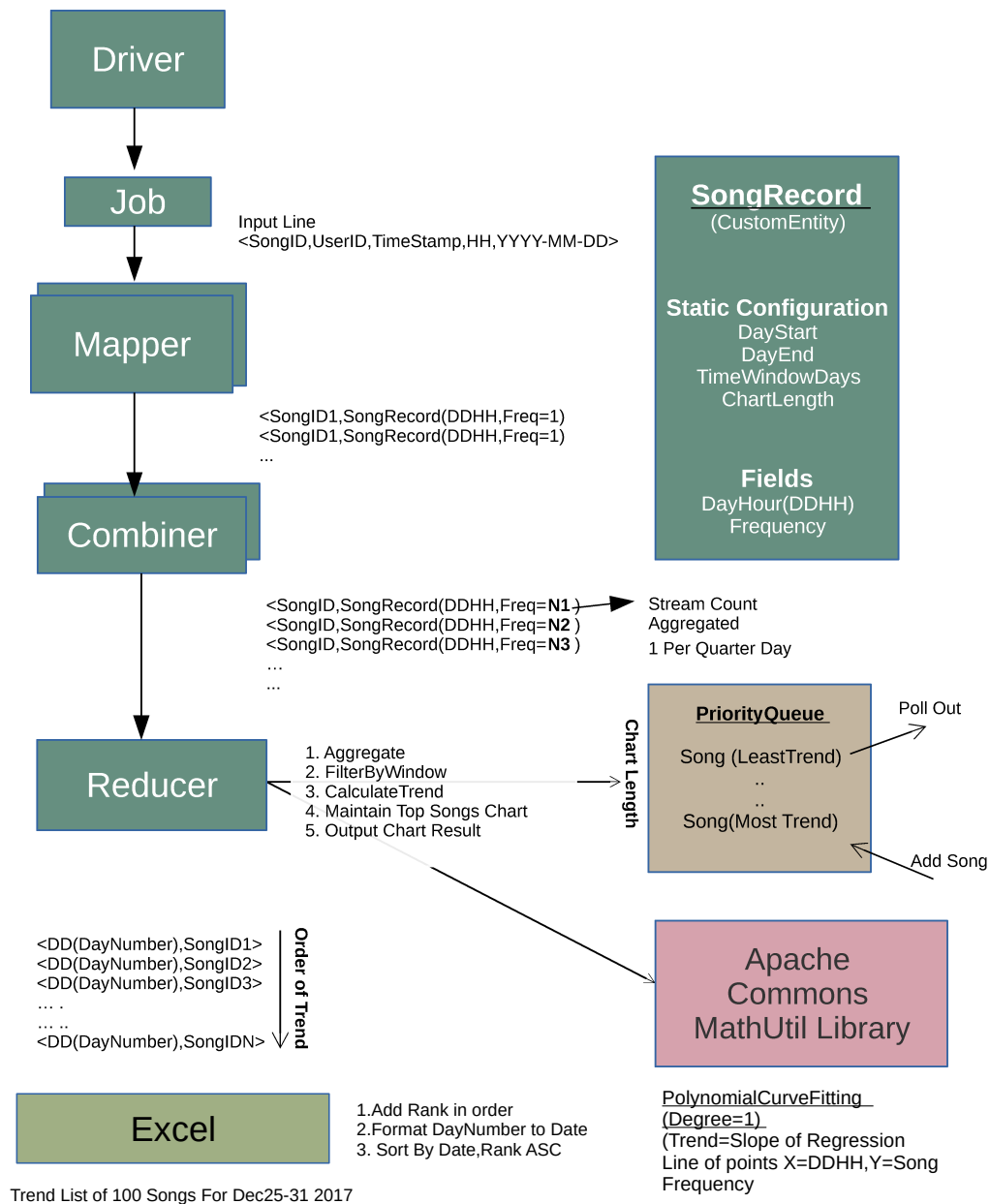
Takes each input stream line csv record and outputs

<Key=SongID,Value=SongRecord(F=1)>. SongRecord is custom WritableComparable entity to encapsulate transferred data and logic and contains (a) Stream Time (DDHH) and (b) Frequency. Mapper sets fixed Frequency = 1 for each input stream to represent a stream instance for that song.

Combiner

At this point we have many SongRecord objects with Frequency = 1 which needs to be aggregated before transferring. Combiner aggregates all objects of each song so that there is only one SongRecord object per day. Frequency = count of number of streams in that given day. <Key=SongID,Value=SongRecord(F=N)>. Aggregation is done using a hash map where key is a unique day and value is the count.

Structure & Flow



Reducer

The shuffle sort phase provides all daytime wise songrecords and reducer is invoked repeatedly once per song passing all songrecord data. There is only one object per daytime & its frequency giving us the count. Reducer aggregates the song records, filters by window span (24 hours), calculates its weight age (trend) & adds it to a day wise priority queue on the fly while removing the least priority records to keep queue size limited to required range. The order of priority is determined by value of daytime, its trend & absolute deviation from mean. Output is written in cleanup method once all songs for all dates are processed.

This processing continues till all songs are processed and top 100 songs are added to respective day wise queue. Once done, cleanup method of reducer is invoked only once where all day queues are taken and output in reverse order<Key=Day, Value=SongID> making highest trending song on top of day chart.

Algorithm Description

The implementation of finding top trending songs aims at weighing each song as trending: -

1. Increasing amount of streams over time
2. Recent streaming
3. Not all time trending
4. Not sporadic but consistent streaming pattern over time

The statistical measures used to measure these are listed below: -

Factor	Statistic	Usage
Spikes	Z-factor of data point $(y-\bar{y})/\sigma$	Z Factor indicates distance of point from mean Frequency of Stream in each aggregate count weighted by inverse of z-factor to reduce effect of spike during regression
New over Old	Coefficient of Variance σ/\bar{y}	Frequency multiplied with standard deviation. An old song will have consistent streaming frequency while a new song will deviate a lot from mean
Recency	No of days since window start/TotalWindow Span	Weights 0-20% = 1

		21-40% = 1 41-60% = 4 61-80% = 8 81-100% = 16
Increasing Stream	Least Square Regression Slope (each data point weighted by $1/z$ factor)	Higher slope of regression line indicates a higher rising trend of the song

Sorting with Efficient Memory Usage & Single Reducer

When working with big data it is not wise to hold more than necessary data. Extreme accuracy is of lesser priority compared to practical need. Hence considering this and average uniform nature of song streaming data, a **PriorityQueue** is used to keep adding songs kept sorted by its weight and lower weight ones polled out on the fly to accommodate for a higher trending song.

Song P Weight 40	Song Q Weight 35	Song R Weight 30	Song S Weight 25	Song T Weight 20	Song U Weight 15
---------------------	---------------------	---------------------	---------------------	---------------------	---------------------

A priority queue where Song P with higher weight is added and least weight Song U (thereby highest priority) is polled out to keep priority length fixed.

Trending Song

Window

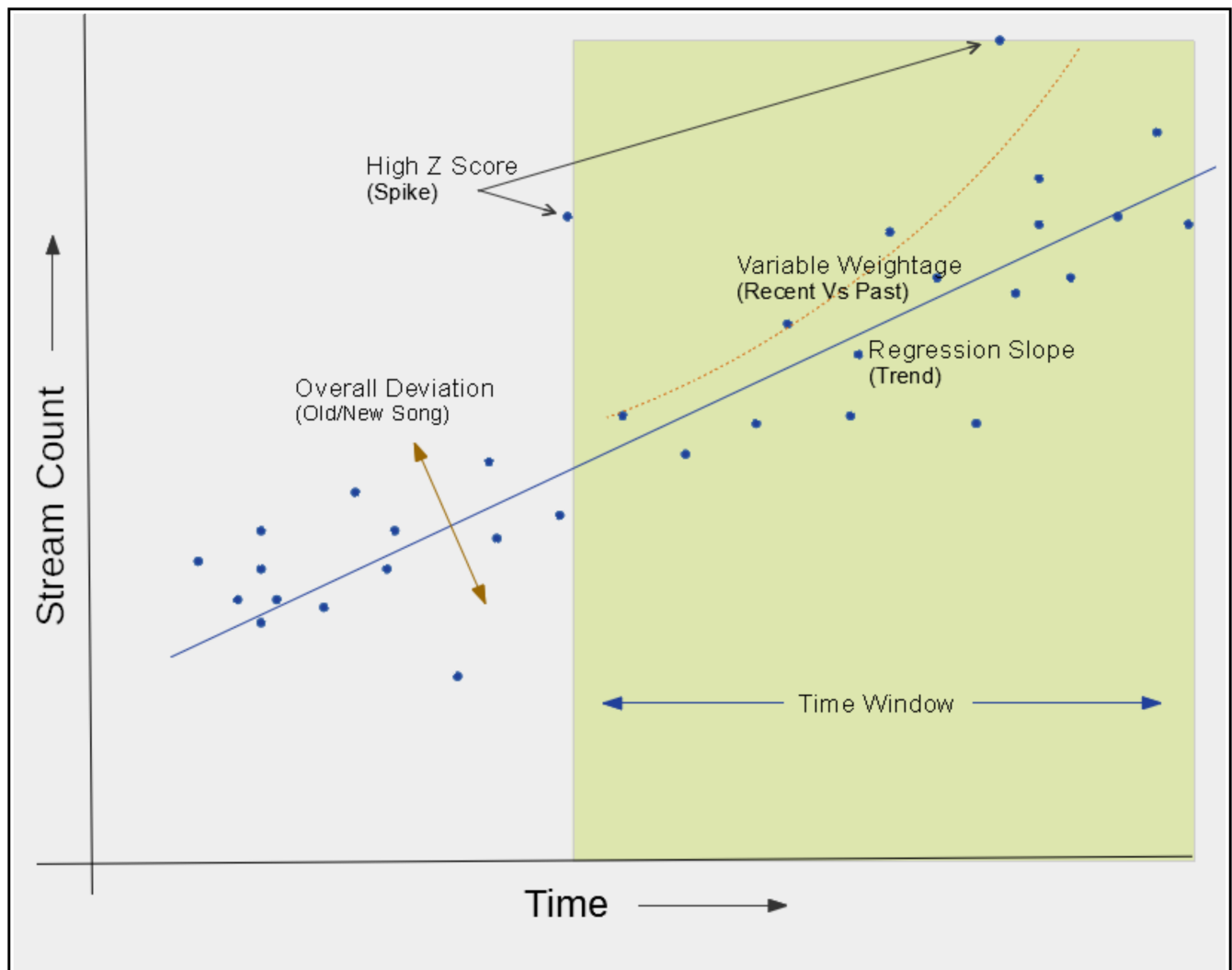
The time range in which song aggregate data point (day, count of streams) is picked

Spike

A data point that has much higher z-score than other points. Hence each data point is weighted inverse to this statistic.

Slope – Main Weight Factor

A single degree polynomial curve (regression line) fitting is done to identify data trend. The slope of this line is used as a factor to rank song & indicates how much streams have increased over time



Citations

Curve Fitting using Apache Commons Math Util Library

<http://commons.apache.org/proper/commons-math/userguide/fitting.html>

Coefficient of Variance

https://en.wikipedia.org/wiki/Coefficient_of_variation

Regression Analysis

https://en.wikipedia.org/wiki/Regression_analysis

Justifications

Why no Partitioner?

Partitioner would be useful if we could reduce parallel set of data for different key buckets. However in our case, for each song, we need to analyze historical data correlating with other song performance to rank them. In such a case, there will be duplicate data across reducers which could lead to lot of unnecessary network data transfer.

Won't PriorityQueue cause premature poll out of a song?

The data has average distribution & priority queue serves the practical need identifying top 100 trending songs well without being being overtly precise. This suits big data needs and stream processing algorithms. Sample testing for a single date resulting in no effect to result.

%Overlap with with Saavn's List

25th – 66%

26th – 65%

27th – 66%

28th – 64%

29th – 64%

30th – 67%

31st – 60%

Execution Details

The overall execution took <32mins to finish execution on m4.16 x-large ec2 instance to process ~44GB data. No exceptions were thrown during execution. Apache Commons Math library jar was added a library reference.

Command

```
hadoop jar saavn.jar -libjars commons-math3-3.6.1.jar  
s3a://mapreduce-project-bde/part-00000 s3a://mybucket/output
```

Output is in following format: -

```
25 X4q99UL7  
25 xoZWY8yV  
25 V3KN74T9
```

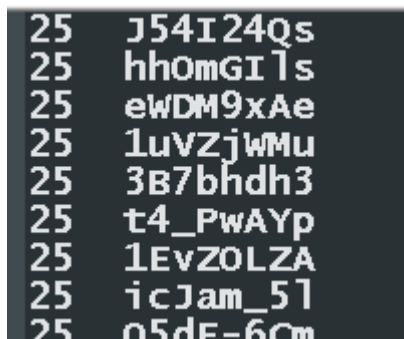


```
....
26 X4q99UL7
26 xoZWY8yV
...
...
31 2zXpXjMp
31 2zXpXjMp
31 3B7bhdh3
...
```

Tab separate 2 column data. First column is day and Second Column is Song ID. Data is already sorted in order of trend.

Steps to Generate Trend List

1. Create an s3 bucket to store output
2. Upload saavnhourly.jar and commons-math3-3.6.1.jar to EC2
3. Execute command `hadoop jar saavn.jar -libjars commons-math3-3.6.1.jar s3a://mapreduce-project-bde/part-00000 s3a://<bucket>/output`
4. Download the part-r-00000 file and rename to output.txt. Following is how output appears: -



```
25 J54I24Qs
25 hhOmGI1s
25 ewDM9xAe
25 1uVZjWMu
25 3B7bhdh3
25 t4_PwAYp
25 1EvZOLZA
25 icJam_51
25 05dE-6Cm
```

5. Copy and paste into Excel/LibreOffice Calc (select tab as delimiter). Assume columns A & B.
6. Insert a new row on top to provide header names. Name columns as “day” & “pid”
7. Insert a third column named “date” and insert following formula `=“2017-12-”&A2` below header & drag to apply whole column. Copy the entire column and paste special as values/unformatted text in place.
8. Insert a fourth column name “rank” & insert number series from 1 to 100 for each day against it.

	A	B	C	D	E
88	25	injzG32Z	2017-12-25	87	
89	25	Wa2ECpqQ	2017-12-25	88	
90	25	cWjC2ttw	2017-12-25	89	
91	25	og3XtU1p	2017-12-25	90	
92	25	0-5hUtBO	2017-12-25	91	
93	25	jjZ2jba_	2017-12-25	92	
94	25	nAl7nw4v	2017-12-25	93	
95	25	etrVmxnc	2017-12-25	94	
96	25	YJ6Vy4Nr	2017-12-25	95	
97	25	mVRHX07V	2017-12-25	96	
98	25	iKLVQyXn	2017-12-25	97	
99	25	9VwKDBO4	2017-12-25	98	
100	25	twNjx7HC	2017-12-25	99	
101	25	7nBCKJel	2017-12-25	100	
102	27	J54I24Qs	2017-12-27	1	
103	27	hhOmGlls	2017-12-27	2	
104	27	1uVZjWMu	2017-12-27		
105	27	eWDM9xAe	2017-12-27		
106	27	Wfngg_NC	2017-12-27		
107	27	1EvZOLZA	2017-12-27		
108	27	t4_PwAYp	2017-12-27		
109	27	3B7bhdh3	2017-12-27		
110	27	L_eY69Bv	2017-12-27		
111	27	fMag1Tga	2017-12-27		
112	27	Q5dE-6Cm	2017-12-27		
113	27	rM6SpcAX	2017-12-27		11
114	27	tgmmUOx0	2017-12-27		
115	27	icJam_5l	2017-12-27		
116	27	2zXpXjMp	2017-12-27		
117	27	wpe4NbSF	2017-12-27		

9. Sort entire data range by (a) day & then by (b) rank

10. Delete day column & Reorder columns as pid, rank, date.

	A	B	C
1	pid	rank	date
2	J54I24Qs	1	2017-12-25
3	hhOmGlls	2	2017-12-25
4	eWDM9xAe	3	2017-12-25
5	1uVZjWMu	4	2017-12-25
6	3B7bhdh3	5	2017-12-25
7	t4_PwAYp	6	2017-12-25

11. Save the file in as csv format.

Console Output

```
[ec2-user@ip-172-31-93-76 ~]$ hadoop jar saavnhourly.jar -libjars commons-math3-3.6.1.jar
s3a://mapreduce-project-bde/part-00000 s3a://monsoonfire/output
18/11/17 12:10:33 WARN impl.MetricsConfig: Cannot locate configuration: tried hadoop-
metrics2-s3a-file-system.properties,hadoop-metrics2.properties
18/11/17 12:10:33 INFO impl.MetricsSystemImpl: Scheduled snapshot period at 10 second(s).
18/11/17 12:10:33 INFO impl.MetricsSystemImpl: s3a-file-system metrics system started
18/11/17 12:10:35 INFO Configuration.deprecation: fs.s3a.server-side-encryption-key is
deprecated. Instead, use fs.s3a.server-side-encryption.key
18/11/17 12:10:35 INFO client.RMProxy: Connecting to ResourceManager at ip-172-31-93-
76.ec2.internal/172.31.93.76:8032
18/11/17 12:10:36 INFO input.FileInputFormat: Total input paths to process : 1
18/11/17 12:10:36 INFO mapreduce.JobSubmitter: number of splits:176
18/11/17 12:10:36 INFO mapreduce.JobSubmitter: Submitting tokens for job:
job_1542425199835_0006
18/11/17 12:10:36 INFO impl.YarnClientImpl: Submitted application
application_1542425199835_0006
18/11/17 12:10:36 INFO mapreduce.Job: The url to track the job: http://ip-172-31-93-
76.ec2.internal:8088/proxy/application_1542425199835_0006/
18/11/17 12:10:36 INFO mapreduce.Job: Running job: job_1542425199835_0006
18/11/17 12:10:45 INFO mapreduce.Job: Job job_1542425199835_0006 running in uber mode :
false
18/11/17 12:10:45 INFO mapreduce.Job: map 0% reduce 0%
18/11/17 12:11:05 INFO mapreduce.Job: map 1% reduce 0%
18/11/17 12:11:18 INFO mapreduce.Job: map 2% reduce 0%
18/11/17 12:11:42 INFO mapreduce.Job: map 3% reduce 0%
18/11/17 12:12:05 INFO mapreduce.Job: map 4% reduce 0%
18/11/17 12:12:13 INFO mapreduce.Job: map 5% reduce 0%
18/11/17 12:12:36 INFO mapreduce.Job: map 6% reduce 0%
18/11/17 12:12:49 INFO mapreduce.Job: map 7% reduce 0%
18/11/17 12:13:08 INFO mapreduce.Job: map 8% reduce 0%
18/11/17 12:13:20 INFO mapreduce.Job: map 9% reduce 0%
18/11/17 12:13:42 INFO mapreduce.Job: map 10% reduce 0%
18/11/17 12:14:04 INFO mapreduce.Job: map 11% reduce 0%
18/11/17 12:14:14 INFO mapreduce.Job: map 12% reduce 0%
18/11/17 12:14:36 INFO mapreduce.Job: map 13% reduce 0%
18/11/17 12:14:49 INFO mapreduce.Job: map 14% reduce 0%
18/11/17 12:15:07 INFO mapreduce.Job: map 15% reduce 0%
18/11/17 12:15:30 INFO mapreduce.Job: map 16% reduce 0%
18/11/17 12:15:43 INFO mapreduce.Job: map 17% reduce 0%
18/11/17 12:16:03 INFO mapreduce.Job: map 18% reduce 0%
18/11/17 12:16:16 INFO mapreduce.Job: map 19% reduce 0%
18/11/17 12:16:35 INFO mapreduce.Job: map 20% reduce 0%
18/11/17 12:16:58 INFO mapreduce.Job: map 21% reduce 0%
18/11/17 12:17:10 INFO mapreduce.Job: map 22% reduce 0%
18/11/17 12:17:33 INFO mapreduce.Job: map 23% reduce 0%
18/11/17 12:17:46 INFO mapreduce.Job: map 24% reduce 0%
18/11/17 12:18:06 INFO mapreduce.Job: map 25% reduce 0%
18/11/17 12:18:17 INFO mapreduce.Job: map 26% reduce 0%
18/11/17 12:18:40 INFO mapreduce.Job: map 27% reduce 0%
18/11/17 12:18:57 INFO mapreduce.Job: map 28% reduce 0%
18/11/17 12:19:12 INFO mapreduce.Job: map 29% reduce 0%
18/11/17 12:19:32 INFO mapreduce.Job: map 30% reduce 0%
18/11/17 12:19:47 INFO mapreduce.Job: map 31% reduce 0%
18/11/17 12:20:02 INFO mapreduce.Job: map 32% reduce 0%
18/11/17 12:20:19 INFO mapreduce.Job: map 33% reduce 0%
18/11/17 12:20:38 INFO mapreduce.Job: map 34% reduce 0%
18/11/17 12:21:01 INFO mapreduce.Job: map 35% reduce 0%
18/11/17 12:21:16 INFO mapreduce.Job: map 36% reduce 0%
18/11/17 12:21:33 INFO mapreduce.Job: map 37% reduce 0%
18/11/17 12:21:47 INFO mapreduce.Job: map 38% reduce 0%
18/11/17 12:22:05 INFO mapreduce.Job: map 39% reduce 0%
18/11/17 12:22:21 INFO mapreduce.Job: map 40% reduce 0%
18/11/17 12:22:40 INFO mapreduce.Job: map 41% reduce 0%
18/11/17 12:23:02 INFO mapreduce.Job: map 42% reduce 0%
```

18/11/17	12:23:15	INFO	mapreduce.Job:	map	43%	reduce	0%
18/11/17	12:23:34	INFO	mapreduce.Job:	map	44%	reduce	0%
18/11/17	12:23:47	INFO	mapreduce.Job:	map	45%	reduce	0%
18/11/17	12:24:10	INFO	mapreduce.Job:	map	46%	reduce	0%
18/11/17	12:24:26	INFO	mapreduce.Job:	map	47%	reduce	0%
18/11/17	12:24:43	INFO	mapreduce.Job:	map	48%	reduce	0%
18/11/17	12:25:02	INFO	mapreduce.Job:	map	49%	reduce	0%
18/11/17	12:25:16	INFO	mapreduce.Job:	map	50%	reduce	0%
18/11/17	12:25:36	INFO	mapreduce.Job:	map	51%	reduce	0%
18/11/17	12:25:49	INFO	mapreduce.Job:	map	52%	reduce	0%
18/11/17	12:26:13	INFO	mapreduce.Job:	map	53%	reduce	0%
18/11/17	12:26:34	INFO	mapreduce.Job:	map	54%	reduce	0%
18/11/17	12:26:45	INFO	mapreduce.Job:	map	55%	reduce	0%
18/11/17	12:27:05	INFO	mapreduce.Job:	map	56%	reduce	0%
18/11/17	12:27:17	INFO	mapreduce.Job:	map	57%	reduce	0%
18/11/17	12:27:41	INFO	mapreduce.Job:	map	58%	reduce	0%
18/11/17	12:27:57	INFO	mapreduce.Job:	map	59%	reduce	0%
18/11/17	12:28:14	INFO	mapreduce.Job:	map	60%	reduce	0%
18/11/17	12:28:30	INFO	mapreduce.Job:	map	61%	reduce	0%
18/11/17	12:28:46	INFO	mapreduce.Job:	map	62%	reduce	0%
18/11/17	12:29:05	INFO	mapreduce.Job:	map	63%	reduce	0%
18/11/17	12:29:29	INFO	mapreduce.Job:	map	64%	reduce	0%
18/11/17	12:29:42	INFO	mapreduce.Job:	map	65%	reduce	0%
18/11/17	12:29:59	INFO	mapreduce.Job:	map	66%	reduce	0%
18/11/17	12:30:16	INFO	mapreduce.Job:	map	67%	reduce	0%
18/11/17	12:30:32	INFO	mapreduce.Job:	map	68%	reduce	0%
18/11/17	12:30:44	INFO	mapreduce.Job:	map	69%	reduce	0%
18/11/17	12:31:11	INFO	mapreduce.Job:	map	70%	reduce	0%
18/11/17	12:31:30	INFO	mapreduce.Job:	map	71%	reduce	0%
18/11/17	12:31:43	INFO	mapreduce.Job:	map	72%	reduce	0%
18/11/17	12:32:06	INFO	mapreduce.Job:	map	73%	reduce	0%
18/11/17	12:32:22	INFO	mapreduce.Job:	map	74%	reduce	0%
18/11/17	12:32:39	INFO	mapreduce.Job:	map	75%	reduce	0%
18/11/17	12:32:52	INFO	mapreduce.Job:	map	76%	reduce	0%
18/11/17	12:33:11	INFO	mapreduce.Job:	map	77%	reduce	0%
18/11/17	12:33:28	INFO	mapreduce.Job:	map	78%	reduce	0%
18/11/17	12:33:45	INFO	mapreduce.Job:	map	79%	reduce	0%
18/11/17	12:34:09	INFO	mapreduce.Job:	map	80%	reduce	0%
18/11/17	12:34:21	INFO	mapreduce.Job:	map	81%	reduce	0%
18/11/17	12:34:39	INFO	mapreduce.Job:	map	82%	reduce	8%
18/11/17	12:34:46	INFO	mapreduce.Job:	map	82%	reduce	14%
18/11/17	12:34:53	INFO	mapreduce.Job:	map	82%	reduce	19%
18/11/17	12:34:59	INFO	mapreduce.Job:	map	82%	reduce	21%
18/11/17	12:35:05	INFO	mapreduce.Job:	map	82%	reduce	24%
18/11/17	12:35:08	INFO	mapreduce.Job:	map	83%	reduce	24%
18/11/17	12:35:11	INFO	mapreduce.Job:	map	83%	reduce	27%
18/11/17	12:35:23	INFO	mapreduce.Job:	map	83%	reduce	28%
18/11/17	12:35:32	INFO	mapreduce.Job:	map	84%	reduce	28%
18/11/17	12:35:56	INFO	mapreduce.Job:	map	85%	reduce	28%
18/11/17	12:36:12	INFO	mapreduce.Job:	map	86%	reduce	28%
18/11/17	12:36:23	INFO	mapreduce.Job:	map	86%	reduce	29%
18/11/17	12:36:35	INFO	mapreduce.Job:	map	87%	reduce	29%
18/11/17	12:36:52	INFO	mapreduce.Job:	map	88%	reduce	29%
18/11/17	12:37:16	INFO	mapreduce.Job:	map	89%	reduce	29%
18/11/17	12:37:18	INFO	mapreduce.Job:	map	89%	reduce	30%
18/11/17	12:37:32	INFO	mapreduce.Job:	map	90%	reduce	30%
18/11/17	12:37:55	INFO	mapreduce.Job:	map	91%	reduce	30%
18/11/17	12:38:17	INFO	mapreduce.Job:	map	92%	reduce	30%
18/11/17	12:38:29	INFO	mapreduce.Job:	map	92%	reduce	31%
18/11/17	12:38:40	INFO	mapreduce.Job:	map	93%	reduce	31%
18/11/17	12:38:56	INFO	mapreduce.Job:	map	94%	reduce	31%
18/11/17	12:39:21	INFO	mapreduce.Job:	map	95%	reduce	31%
18/11/17	12:39:30	INFO	mapreduce.Job:	map	95%	reduce	32%
18/11/17	12:39:45	INFO	mapreduce.Job:	map	96%	reduce	32%
18/11/17	12:39:57	INFO	mapreduce.Job:	map	97%	reduce	32%
18/11/17	12:40:15	INFO	mapreduce.Job:	map	98%	reduce	32%

18/11/17 12:40:24 INFO mapreduce.Job: map 98% reduce 33%
18/11/17 12:40:39 INFO mapreduce.Job: map 99% reduce 33%
18/11/17 12:40:59 INFO mapreduce.Job: map 100% reduce 33%
18/11/17 12:41:06 INFO mapreduce.Job: map 100% reduce 67%
18/11/17 12:41:12 INFO mapreduce.Job: map 100% reduce 71%
18/11/17 12:41:18 INFO mapreduce.Job: map 100% reduce 75%
18/11/17 12:41:24 INFO mapreduce.Job: map 100% reduce 80%
18/11/17 12:41:30 INFO mapreduce.Job: map 100% reduce 84%
18/11/17 12:41:36 INFO mapreduce.Job: map 100% reduce 88%
18/11/17 12:41:42 INFO mapreduce.Job: map 100% reduce 92%
18/11/17 12:41:48 INFO mapreduce.Job: map 100% reduce 97%
18/11/17 12:41:54 INFO mapreduce.Job: map 100% reduce 100%
18/11/17 12:41:56 INFO mapreduce.Job: Job job_1542425199835_0006 completed successfully
18/11/17 12:41:56 INFO mapreduce.Job: Counters: 54

File System Counters

FILE: Number of bytes read=496507780
FILE: Number of bytes written=1207592515
FILE: Number of read operations=0
FILE: Number of large read operations=0
FILE: Number of write operations=0
HDFS: Number of bytes read=18128
HDFS: Number of bytes written=0
HDFS: Number of read operations=176
HDFS: Number of large read operations=0
HDFS: Number of write operations=0
S3A: Number of bytes read=47088085523
S3A: Number of bytes written=8400
S3A: Number of read operations=363
S3A: Number of large read operations=0
S3A: Number of write operations=21

Job Counters

Launched map tasks=176
Launched reduce tasks=1
Rack-local map tasks=176
Total time spent by all maps in occupied slots (ms)=4857211
Total time spent by all reduces in occupied slots (ms)=451327
Total time spent by all map tasks (ms)=4857211
Total time spent by all reduce tasks (ms)=451327
Total vcore-milliseconds taken by all map tasks=4857211
Total vcore-milliseconds taken by all reduce tasks=451327
Total megabyte-milliseconds taken by all map tasks=4973784064
Total megabyte-milliseconds taken by all reduce tasks=462158848

Map-Reduce Framework

Map input records=702782657
Map output records=679963680
Map output bytes=16998921726
Map output materialized bytes=684193469
Input split bytes=18128
Combine input records=679963680
Combine output records=81425251
Reduce input groups=1806546
Reduce shuffle bytes=684193469
Reduce input records=81425251
Reduce output records=700
Spilled Records=162850502
Shuffled Maps =176
Failed Shuffles=0
Merged Map outputs=176
GC time elapsed (ms)=69071
CPU time spent (ms)=4140080
Physical memory (bytes) snapshot=119761162240
Virtual memory (bytes) snapshot=281466753024
Total committed heap usage (bytes)=126674272256

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=47088085523
File Output Format Counters
  Bytes Written=8400
18/11/17 12:41:56 INFO impl.MetricsSystemImpl: Stopping s3a-file-system metrics system...
18/11/17 12:41:56 INFO impl.MetricsSystemImpl: s3a-file-system metrics system stopped.
18/11/17 12:41:56 INFO impl.MetricsSystemImpl: s3a-file-system metrics system shutdown
complete.
[ec2-user@ip-172-31-93-76 ~]$
```

Appendix

Following files are placed in package to refer: -

1. saavn.zip – Project Source
2. solution_trending_data_daily.csv – Output
3. saavn.jar – Solution binary package (includes Main class, do not provide in args)
4. commons-math3-3.6.1.jar – Math library jar – Place alongside saavn.jar
5. command.txt – Command to execute
6. map reduce output.txt – Output result of map reduce program (song already sorted per date but dates out of order)