Below is the approach I followed.

1. Created java project “SaavnCaseStudy” and added relevant Hadoop libraries.
2. Created a driver class “SaavnDriver” which creates a job and configure all the required settings for the job.
3. Created a Mapper “SaavnMapper” which acts as a mapper.
   a. Input to the map function is one stream or tuple with comma separated values.
   b. Split value by comma to get SongId , date, hour. Get datepart from date which have values in range (1-31).
   c. I have chosen to consider a window size of 1 day i.e., using data of 24th Dec to get trending songs for 25th Dec. Hence ignoring data from 1st Dec to 23rd Dec and 31st Dec and also ignoring junk data i.e., empty/null song Ids.
   d. Output of a mapper is a custom SaavnKeyWritable object which holds datepart and SongId as a key and weight of the song (hour*1) as value, which means that latest song streamed will be having more weightage than the one which streamed few hours back.
4. Created a combiner “SaavnCombiner” which takes output from the mapper, aggregates values based on key and writes weights for each song played on date as an output.
5. Created a partitioner “SaavnPartitioner” which partitions data based on datepart such that each reducer will have data which are streamed on specific date. For example, songs played on date 24 will be sent to reducer 0, songs played on date 25 will be sent to reducer 1 and so on.
6. Created a reducer “SaavnReducer” which aggregates values specific to the key i.e., calculates weight for each song played on specific date.
   a. Used HashMap in SaavnReducer to store intermediate output from the reduce method. reduce method inserts record with songId as key and weight as value. After all the reduce methods executed for all the keys, hashMap contains songIds and corresponding weights, but unsorted by weights.
   b. Used cleanup method of Reducer since it gets executed only once per reducer after all reduce methods are completed. Here performed logic to sort hashMap based on weight, and only writing top 100 songs as an output (songId as key and weight as value).
7. Output of this Map Reduce program will have 7 files containing top 100 songs for each date specified as follows.
   (part-r-00000, corresponds to trending songs for 25th Dec)
   (part-r-00001, corresponds to trending songs for 26th Dec)
   (part-r-00002, corresponds to trending songs for 27th Dec)
   (part-r-00003, corresponds to trending songs for 28th Dec)
   (part-r-00004, corresponds to trending songs for 29th Dec)
   (part-r-00005, corresponds to trending songs for 30th Dec)
   (part-r-00006, corresponds to trending songs for 31st Dec)
8. Download these intermediate output files from Map reduce, open then in excel and delete the second column containing weights, and save them to contain only song Ids in the output. This gives us trending songs for each date.