Steps of execution:

1. Created the maven project (shared as zip) and exported the SaavnFilter.jar.

2. Using winscp transferred the jar on ec2 server.

3. S3 connector is configured on ec2 server as per the module guidelines.

4. Login to ec2 server using putty and switch to hdfs user
   ```
   #> sudo su hdfs
   ```

5. Create the ec2-user directory on hdfs
   ```
   #> hadoop fs -mkdir /user/ec2-user
   ```

6. Change the permission and ownership recursively to ec2-user
   ```
   #> hadoop fs -chown -R ec2-user:ec2-user /user/ec2-user
   #> hadoop fs -chmod -R 777 /user/ec2-user/
   ```

7. Exit from the hdfs user
   ```
   #> exit
   ```

8. Verify jar exists
   ```
   #> ls SaavnFilter.jar
   ```

9. Execute the jar
   ```
   #> hadoop jar SaavnFilter.jar com.saavn.SaavnFilter s3a://mapreduce-project-bde/part-00000
   s3a://saavn-output-shubhra/output
   ```

10. When execution gets completed, download the following files from s3 bucket (s3a://saavn-output-shubhra/output)
    a. part-r-00000
    b. part-r-00001

11. Download the files from s3 web interface and copy to the ec2 server (any unix server) using winscp. Run the following commands to process the files and generate required output.
    ```
    #> cat part-r-0000* | sed 's/\t/,/g' | sort -t"," -n -k2 -r | head -100 | cut -d',' -f1 > 25.txt
    #> cat part-r-0000* | sed 's/\t/,/g' | sort -t"," -n -k3 -r | head -100 | cut -d',' -f1 > 26.txt
    #> cat part-r-0000* | sed 's/\t/,/g' | sort -t"," -n -k4 -r | head -100 | cut -d',' -f1 > 27.txt
    #> cat part-r-0000* | sed 's/\t/,/g' | sort -t"," -n -k5 -r | head -100 | cut -d',' -f1 > 28.txt
    #> cat part-r-0000* | sed 's/\t/,/g' | sort -t"," -n -k6 -r | head -100 | cut -d',' -f1 > 29.txt
    #> cat part-r-0000* | sed 's/\t/,/g' | sort -t"," -n -k7 -r | head -100 | cut -d',' -f1 > 30.txt
    #> cat part-r-0000* | sed 's/\t/,/g' | sort -t"," -n -k8 -r | head -100 | cut -d',' -f1 > 31.txt
    ```

12. Command Explanation:
    cat part-r-0000* : Browse all the files

sed 's/\t/,/g' : replace the tab with comma, in order to parse uniformly as CSV
sort -t"," -n -k2 -r : sort the ‘,’ separated columns on column (k2 here means column 2) by numerically in reverse order
head -100 : pick top 100 lines
cut -d’,’ -f1 : cut the columns by ‘,’ delimiter in order to pick songid
><filename> : to write in the file

Steps to check overlap:

1. Download the file https://s3.amazonaws.com/mapreduce-project-bde/trending_data_daily.csv and winscp to unix server.

2. Cut the songid column using following command:

   ```
   #> cat trending_data_daily.csv | cut -d',' -f1 > trending_songs_saavn.txt
   ```

3. Check the overlap using following commands. Result shows the number of songid existing in MR output files and saavn trending songs file.

   ```
   #> comm -12 <(sort trending_songs_saavn.txt | uniq) <(sort 25.txt | uniq) | wc -l

   98

   #> comm -12 <(sort trending_songs_saavn.txt | uniq) <(sort 26.txt | uniq) | wc -l

   98

   #> comm -12 <(sort trending_songs_saavn.txt | uniq) <(sort 27.txt | uniq) | wc -l

   98

   #> comm -12 <(sort trending_songs_saavn.txt | uniq) <(sort 28.txt | uniq) | wc -l

   98

   #> comm -12 <(sort trending_songs_saavn.txt | uniq) <(sort 29.txt | uniq) | wc -l

   96

   #> comm -12 <(sort trending_songs_saavn.txt | uniq) <(sort 30.txt | uniq) | wc -l

   95

   #> comm -12 <(sort trending_songs_saavn.txt | uniq) <(sort 31.txt | uniq) | wc -l

   96
   ```

   Note: comm command needs the input files to be sorted and with uniq entries