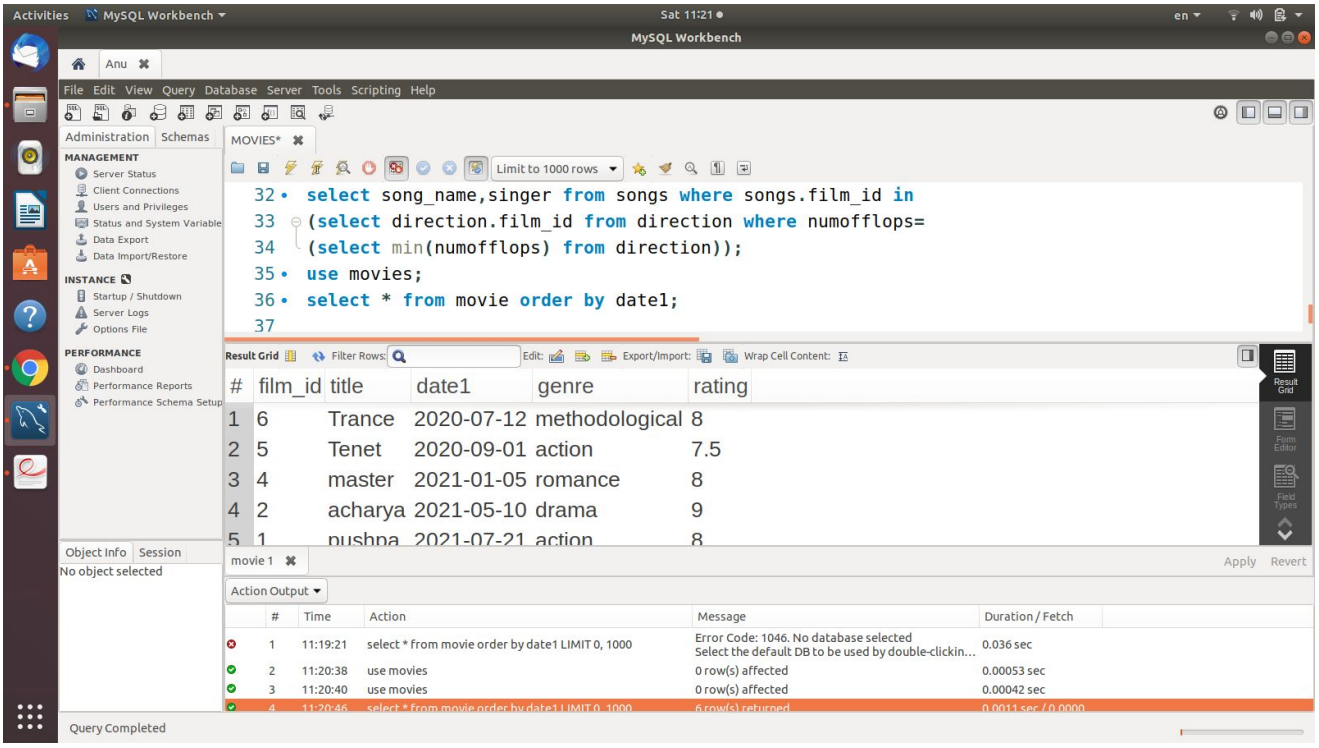
ASSIGNMENT - 4

1. Order by Clause

QUERY : select * from movie order by date1;

OUTPUT:

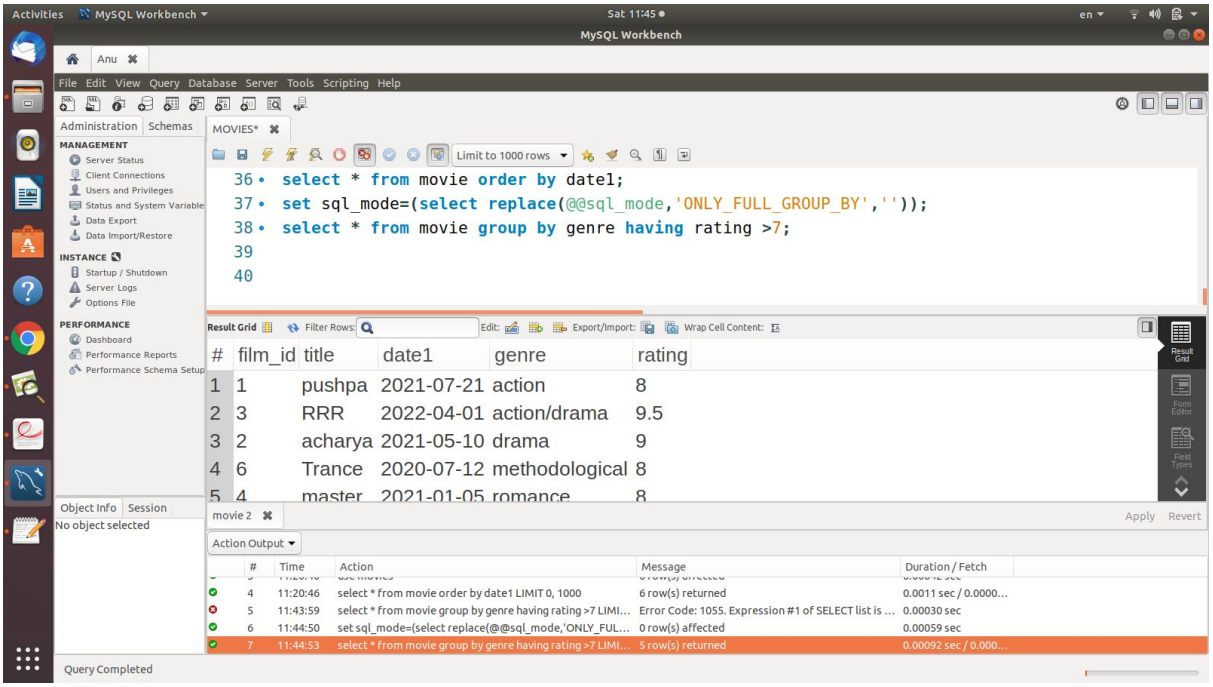


2. Group by and having

QUERY:

```
set sql_mode=(select replace(@@sql_mode,'ONLY_FULL_GROUP_BY',''));
  select * from movie group by genre having rating >7;
```

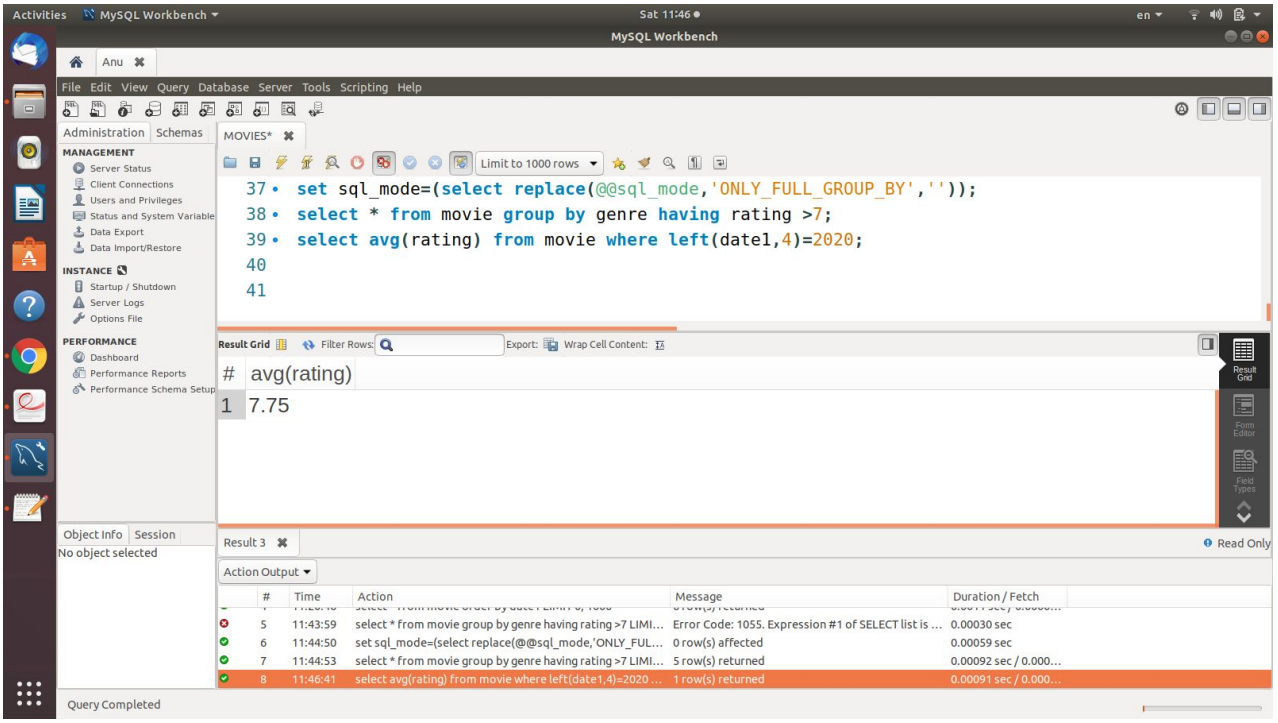
OUTPUT:



3. Aggregate functions

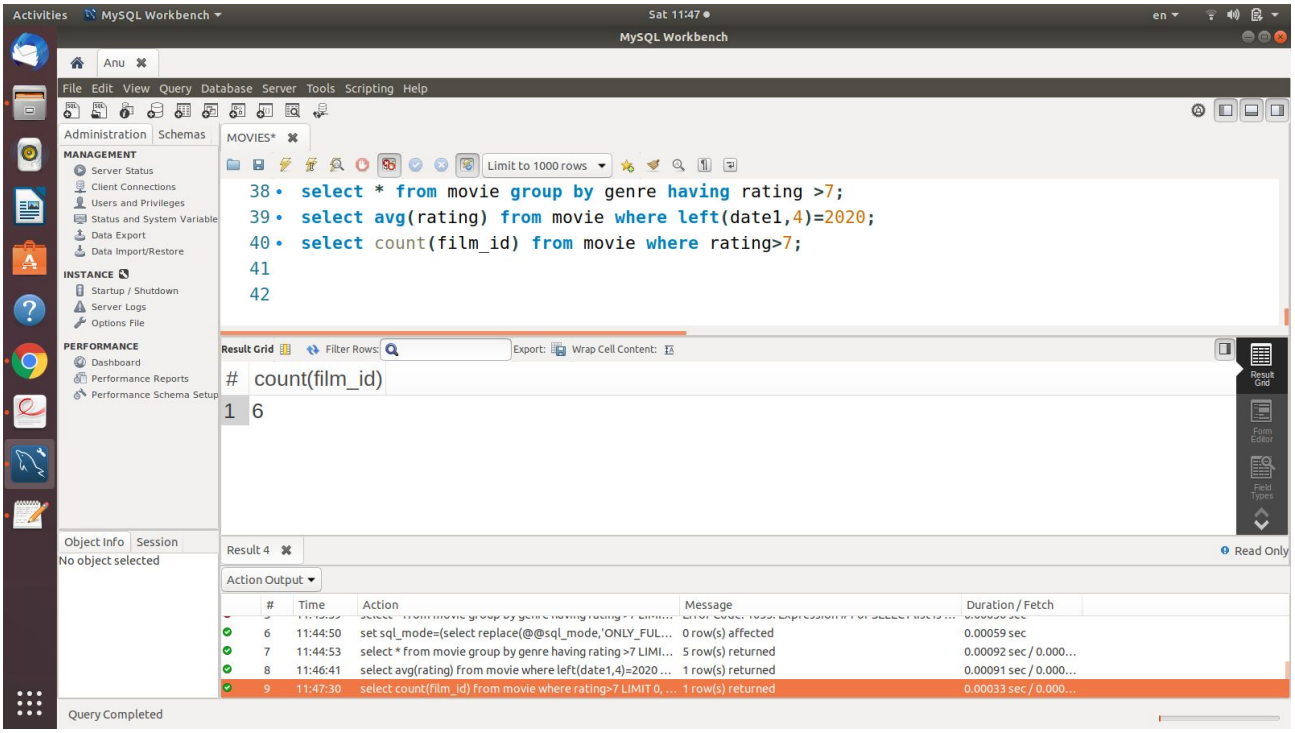
QUERY: select avg(rating) from movie where left(date1,4)=2020;

OUTPUT:

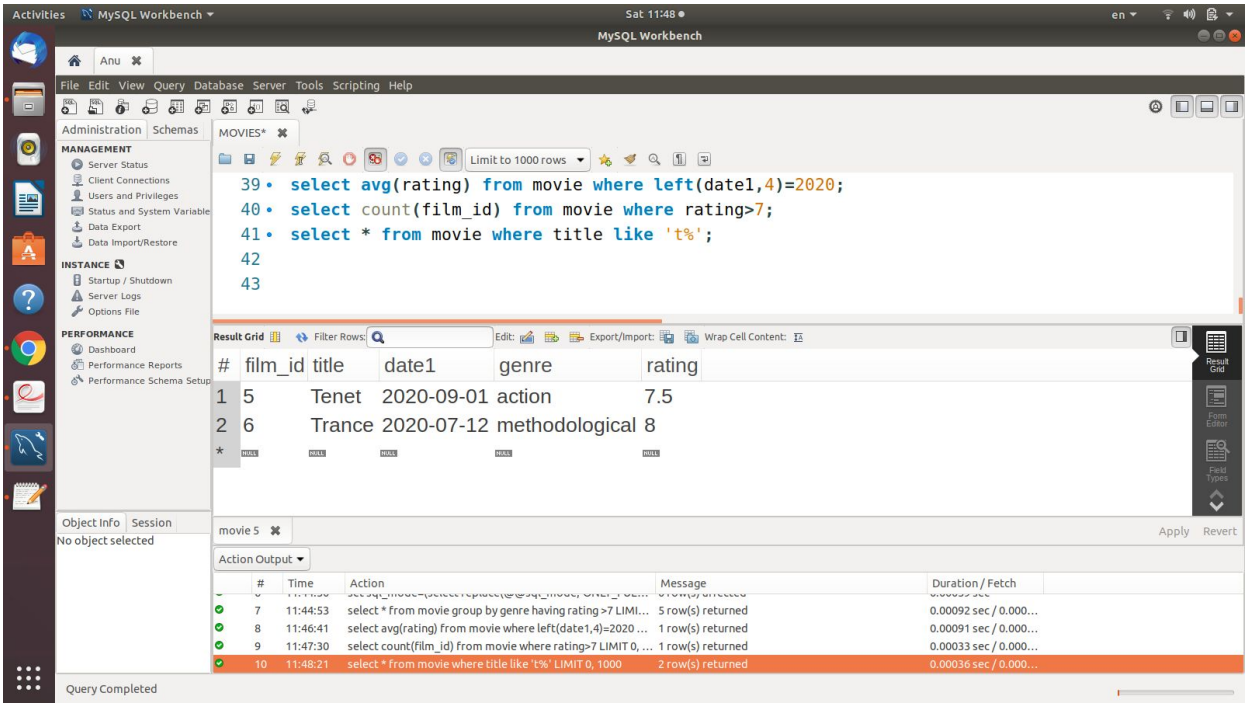


QUERY: select count(film_id) from movie where rating>7;

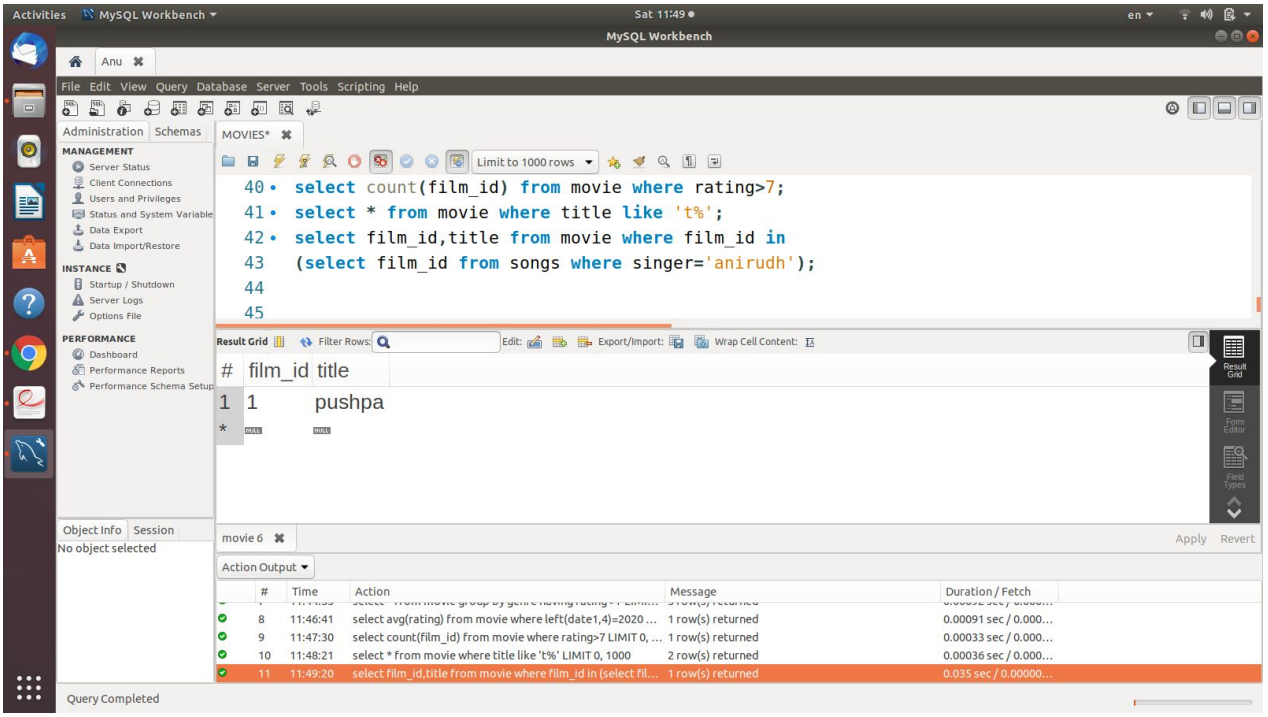
OUTPUT:



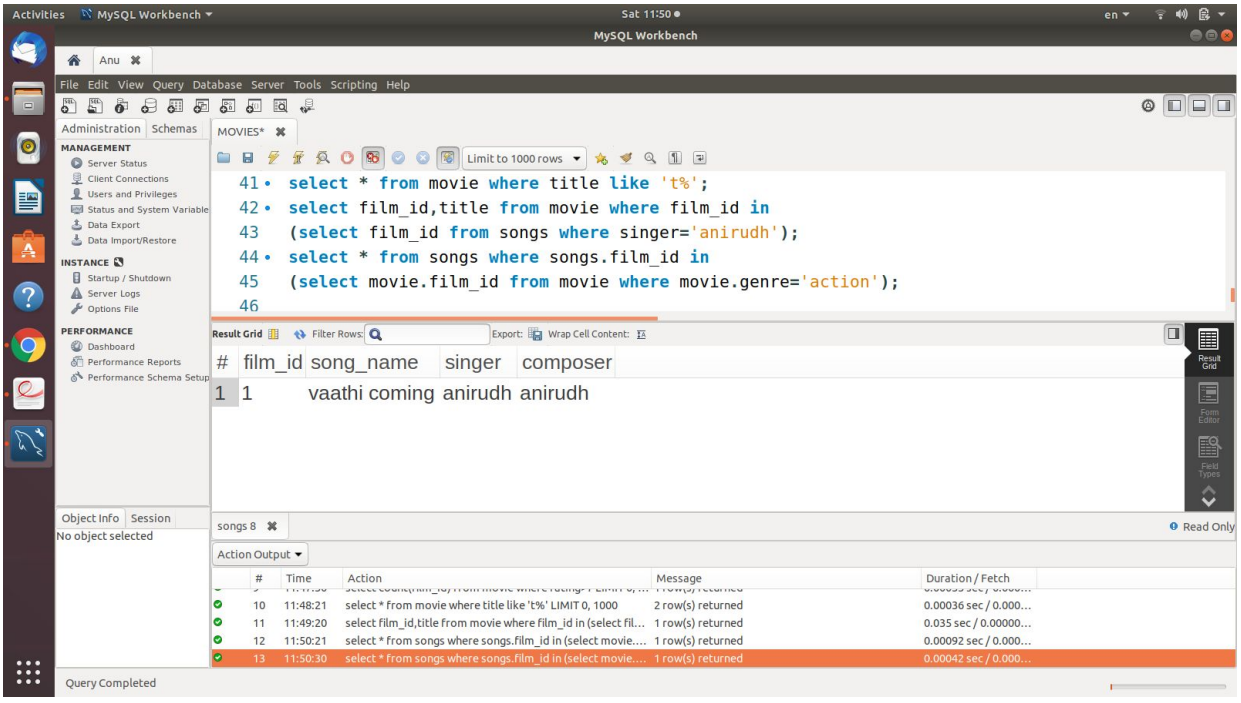
4. Logical operators especially with LIKE
QUERY:select * from movie where title like 't%';
OUTPUT:



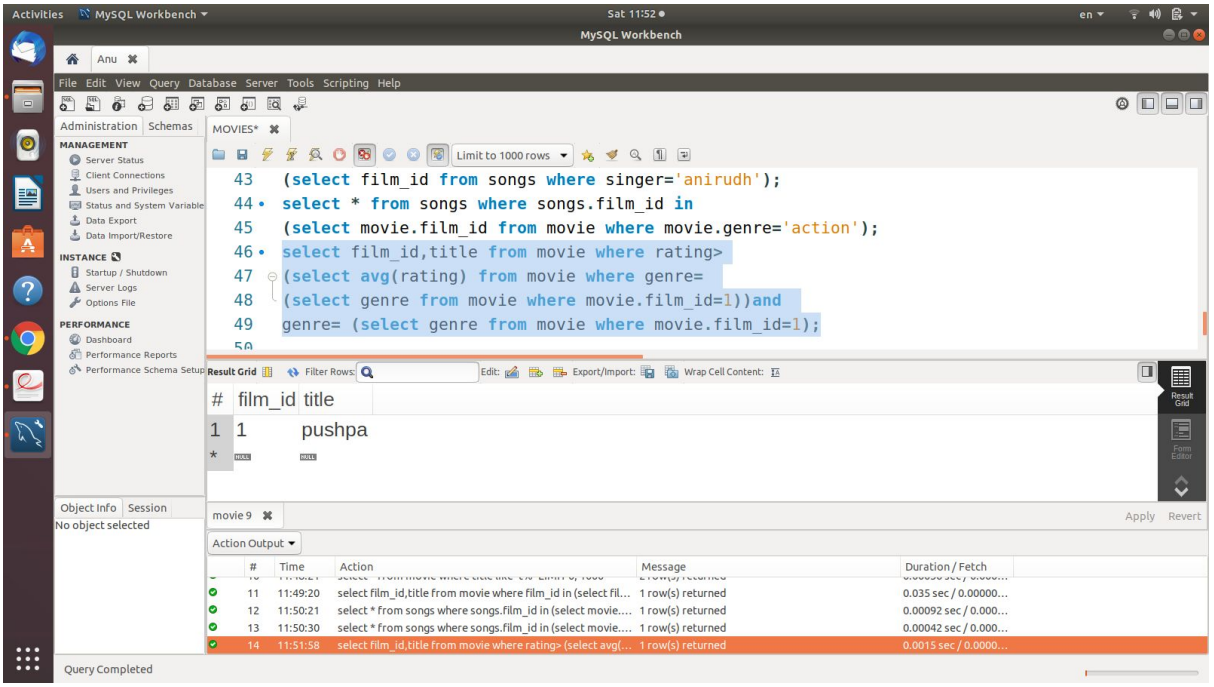
5. At least 4 Nested queries specific to your Database, out of which at least 2 should have multiple subquery.
QUERY 1 : select film_id,title from movie where film_id in
(select film_id from songs where singer='anirudh');
OUTPUT 1:



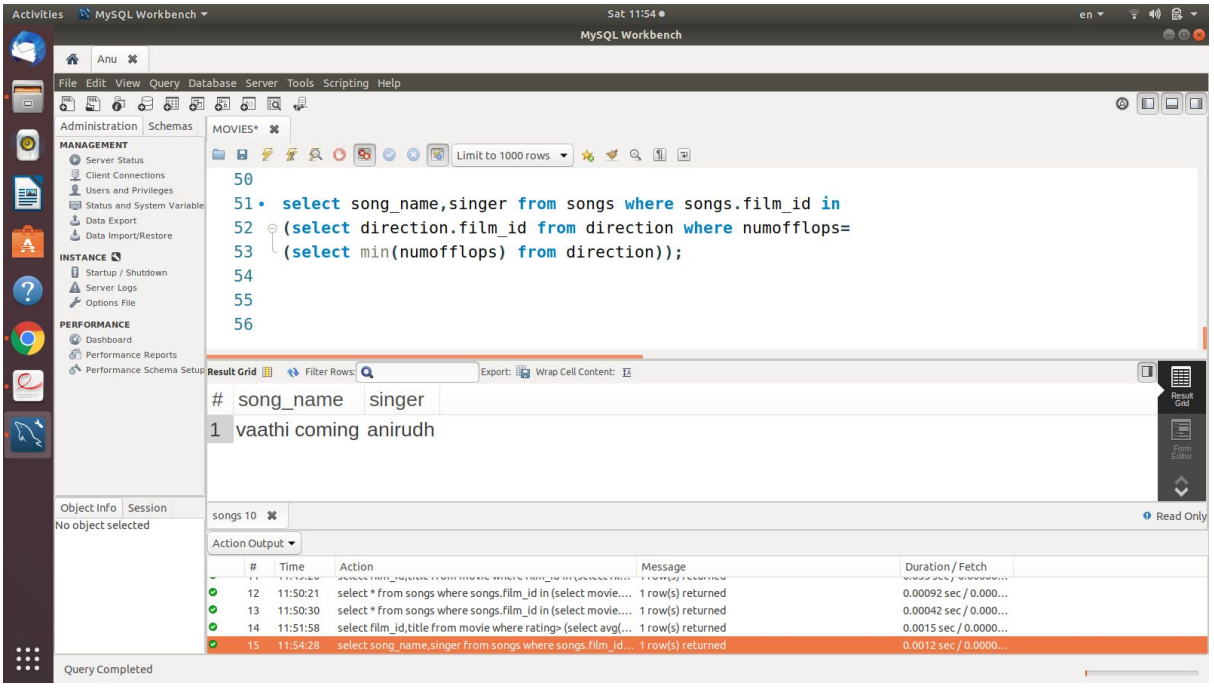
QUERY 2:|select * from songs where songs.film_id in
(select movie.film_id from movie where movie.genre='action');
OUTPUT 2 :



QUERY 3 :select film_id,title from movie where rating>
(select avg(rating) from movie where genre=
(select genre from movie where movie.film_id=1))and
genre= (select genre from movie where movie.film_id=1);
OUTPUT 3:



QUERY 4 :select song_name,singer from songs where songs.film_id in
(select direction.film_id from direction where numofflops=
(select min(numofflops) from direction));
OUTPUT 4:



QUERY3 EXPLANATION: /The inner query returns the average of the ratings of those movies with genre same as the genre of movies with movie_id=134. The outer query displays the movie details with rating > the value returned by the inner query and with genre same as that with movie_id=1./

QUERY 4 EXPLANATION: /*Displays the song name and singer from the songs of movies directed by directors with no_of_flops greater than the minimum number of flops of any director from the director table*/