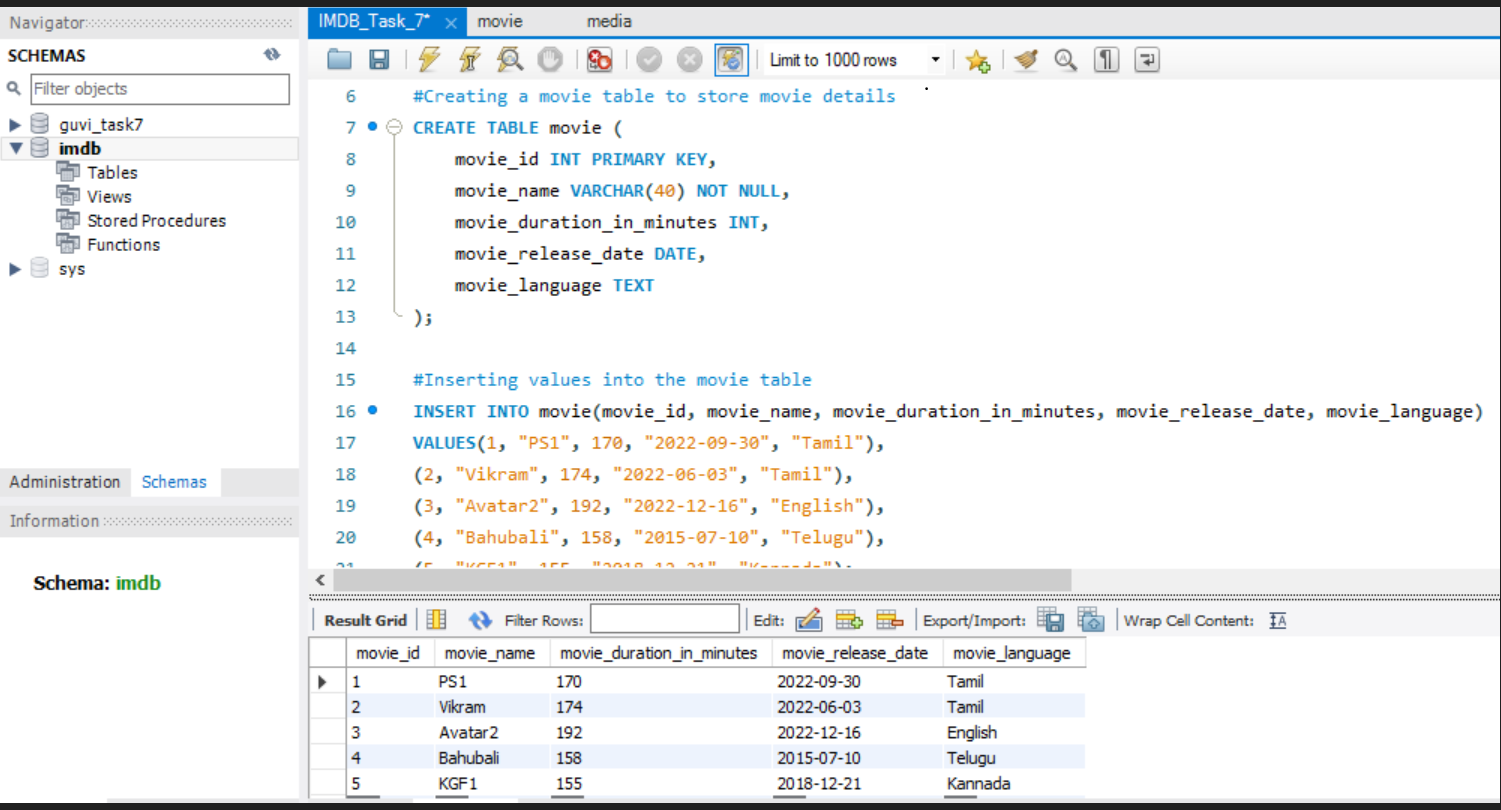
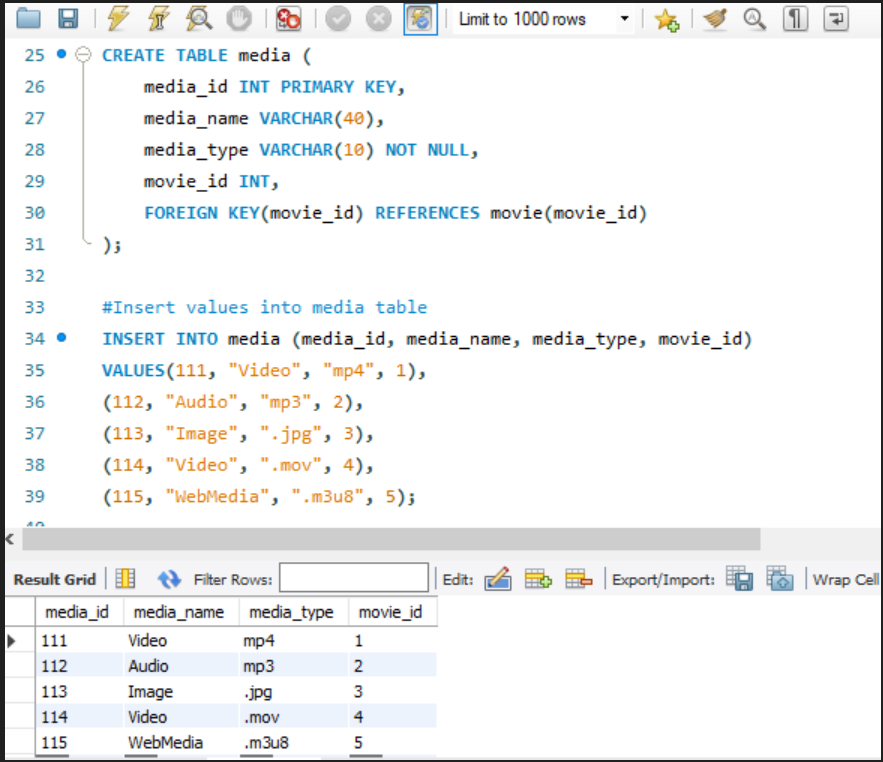
# TASK 7 – MySQL

## Created IMDB database with following tables using Create and SELECT queries from CRUD operations

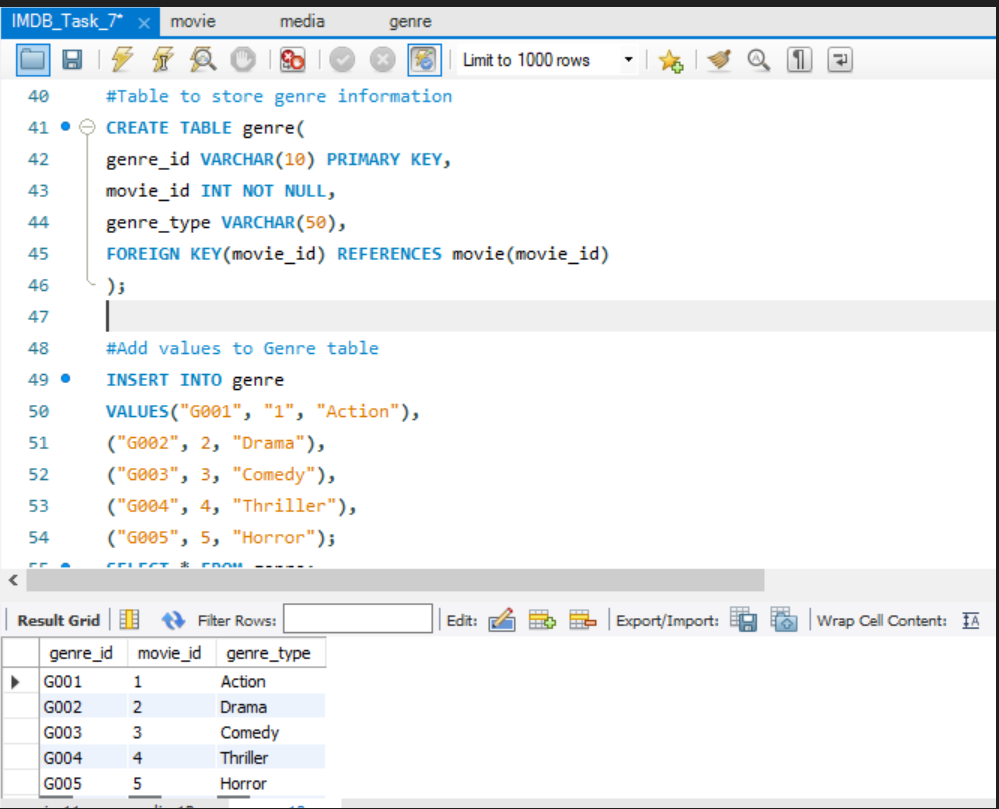
1. CREATE movie table with primary key



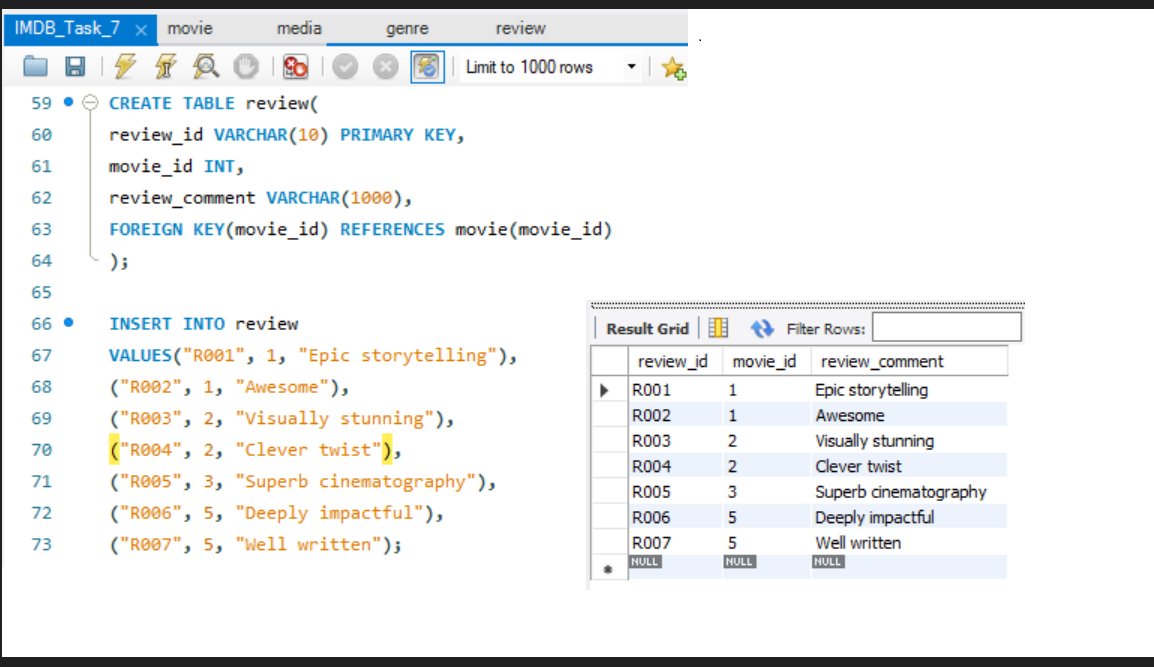
1. Media table with primary and foreign keys to relate with movie table



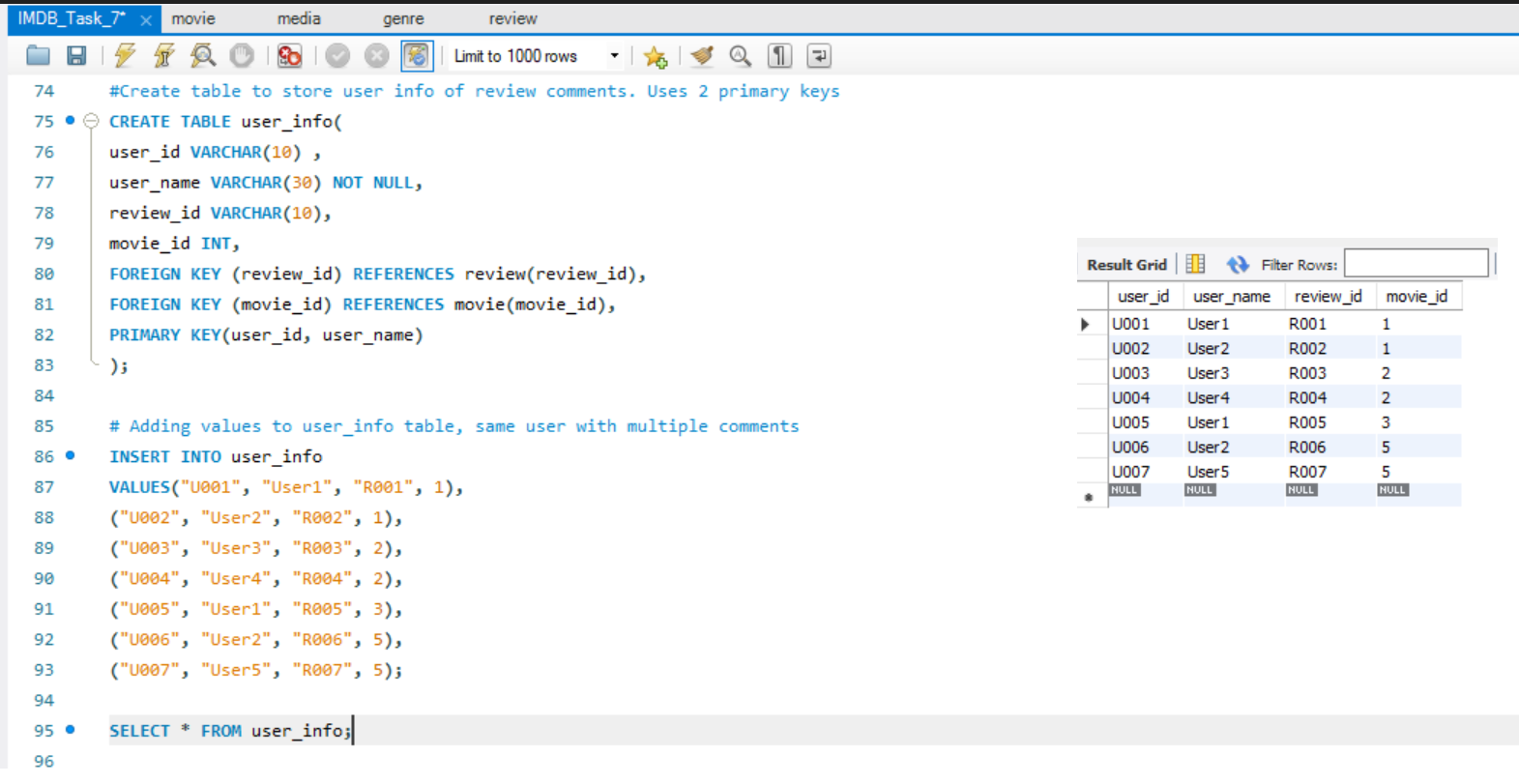
1. Genre table to store the genre of the movies using primary and foreign keys

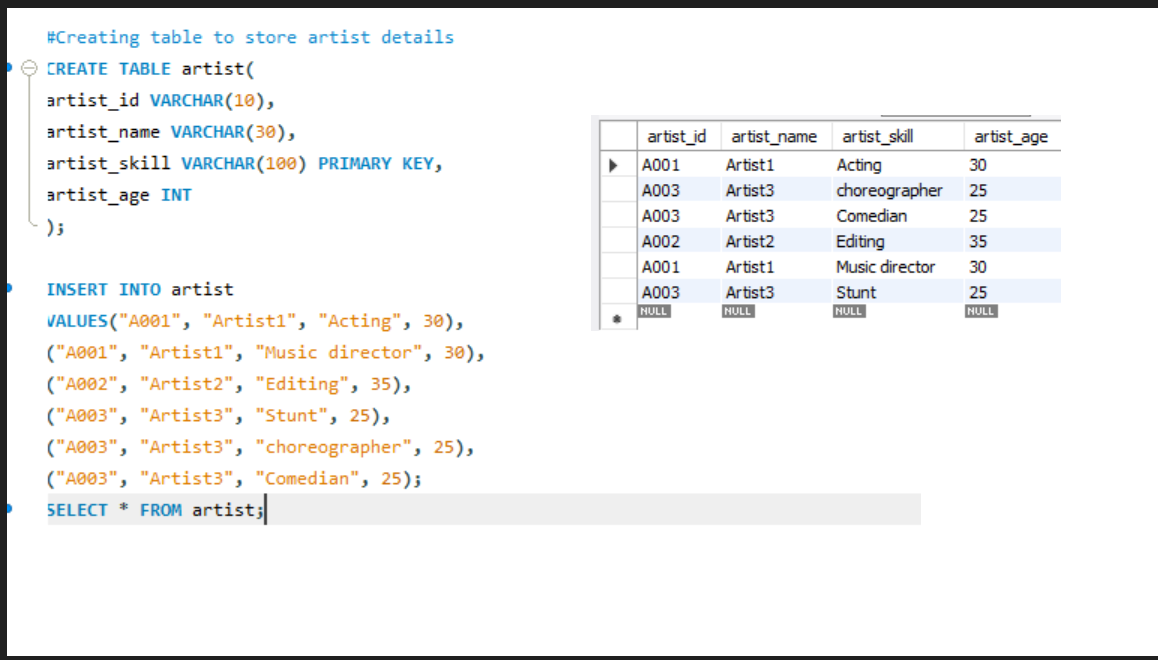


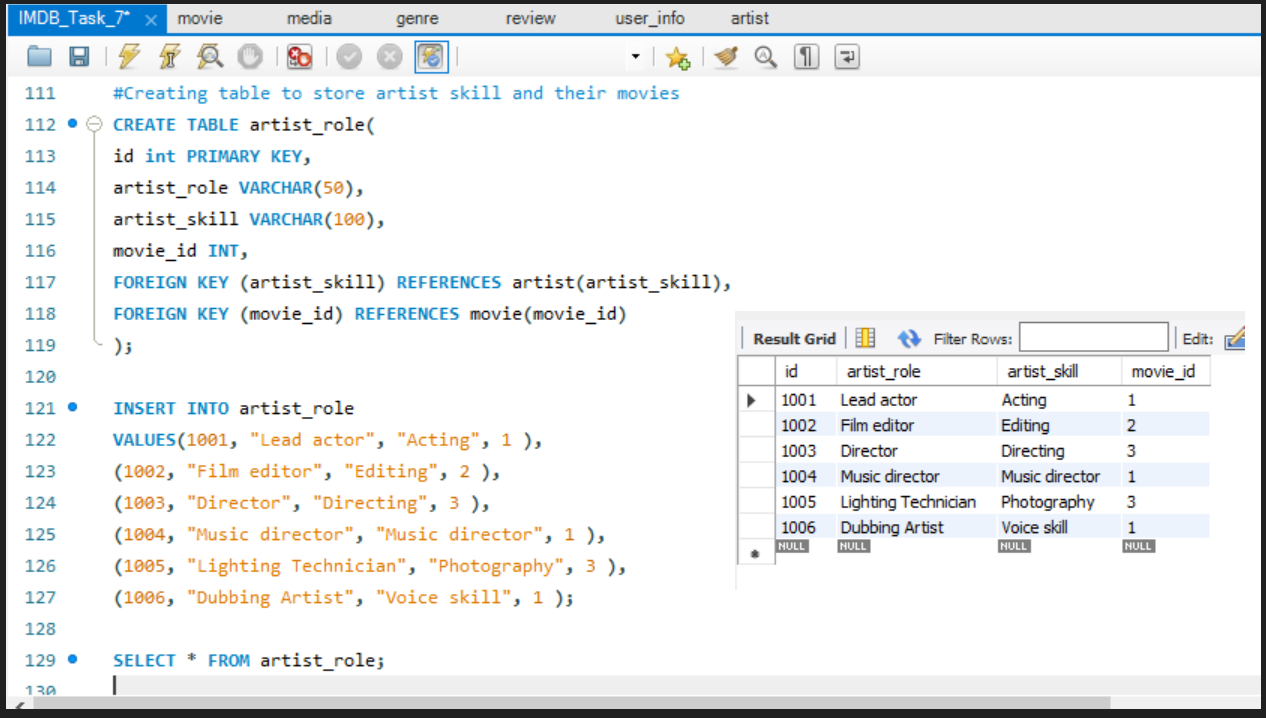
1. Review table to store the reviews of movies. Single movie with multiple reviews



1. Table to store the user info of the reviews. Single user with multiple reviews. Used 2 FOREIGN KEYS

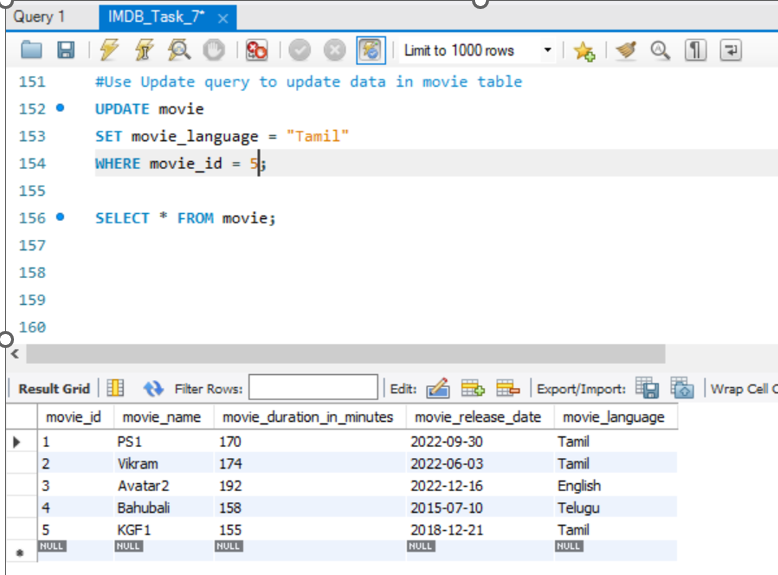


1. Table to store the artist info
2. Table to store the artist and their role in movie

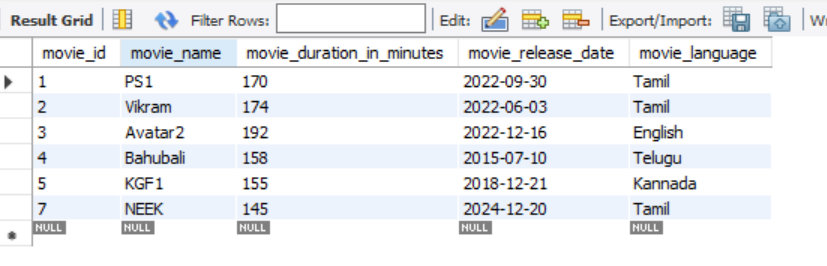
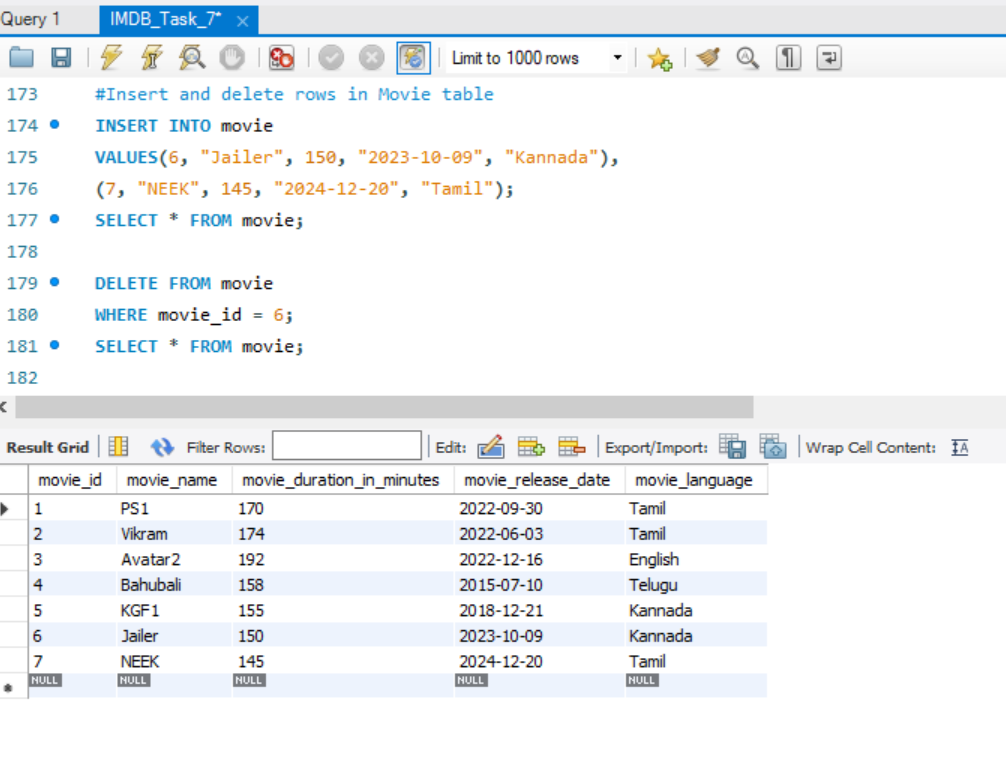


## CRUD : Update and delete operations in rows.

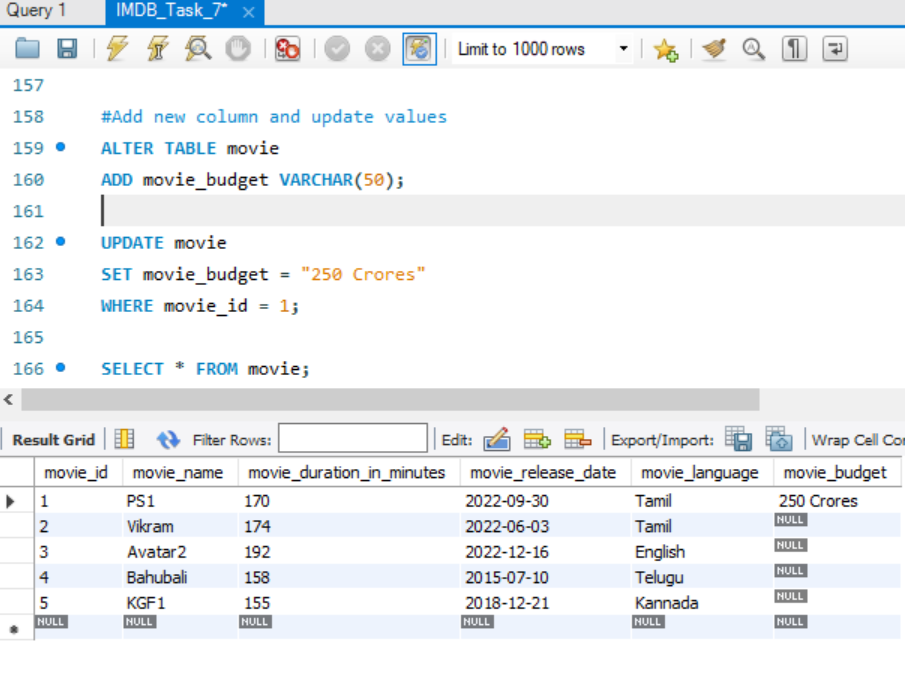
1. UPDATE KGF1 movie language from Kannada to tamil, using UPDATE query and primary key movie\_id

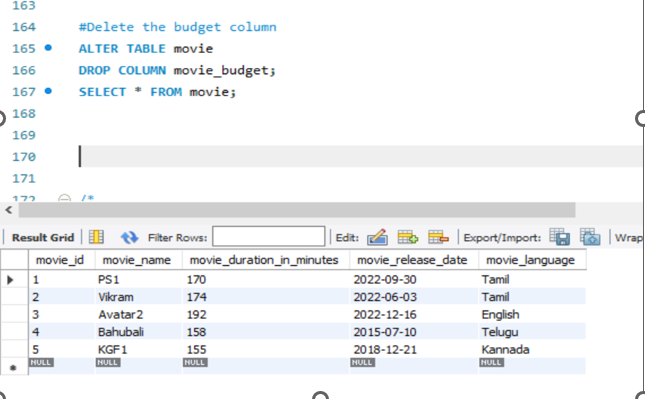


1. DELETE rows from movie table

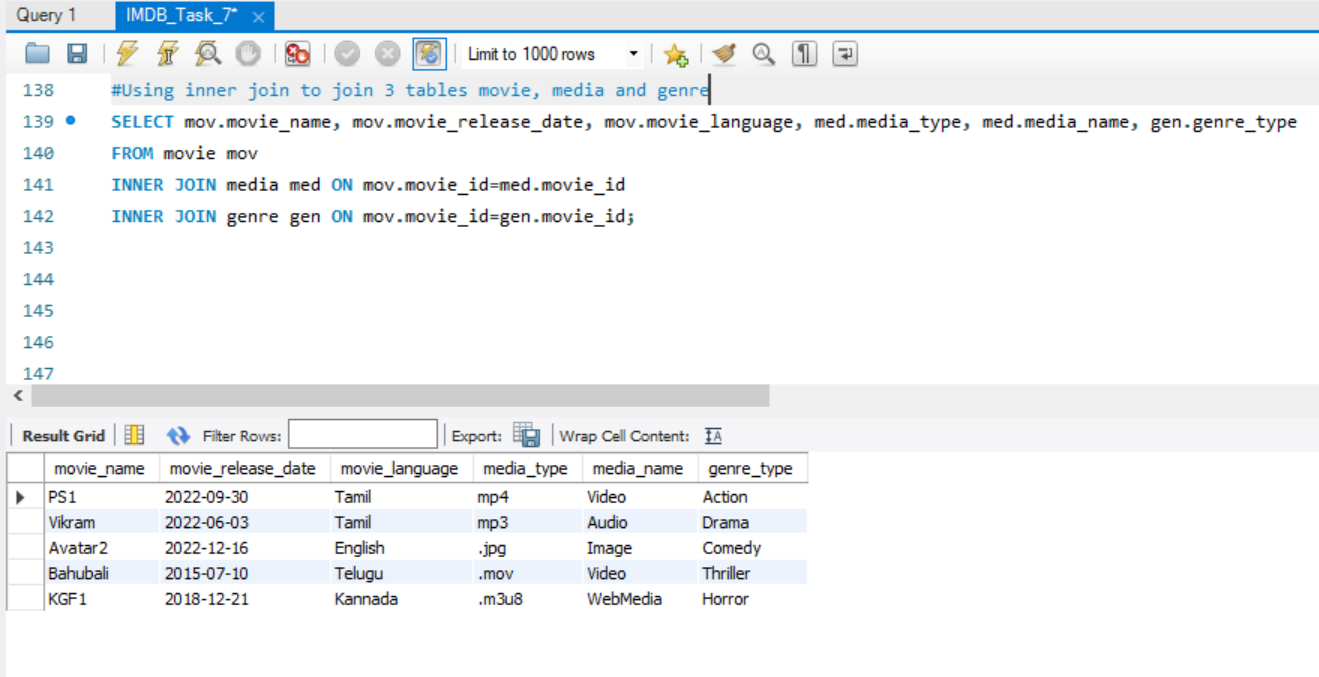


## ALTER : ADD and DROP columns

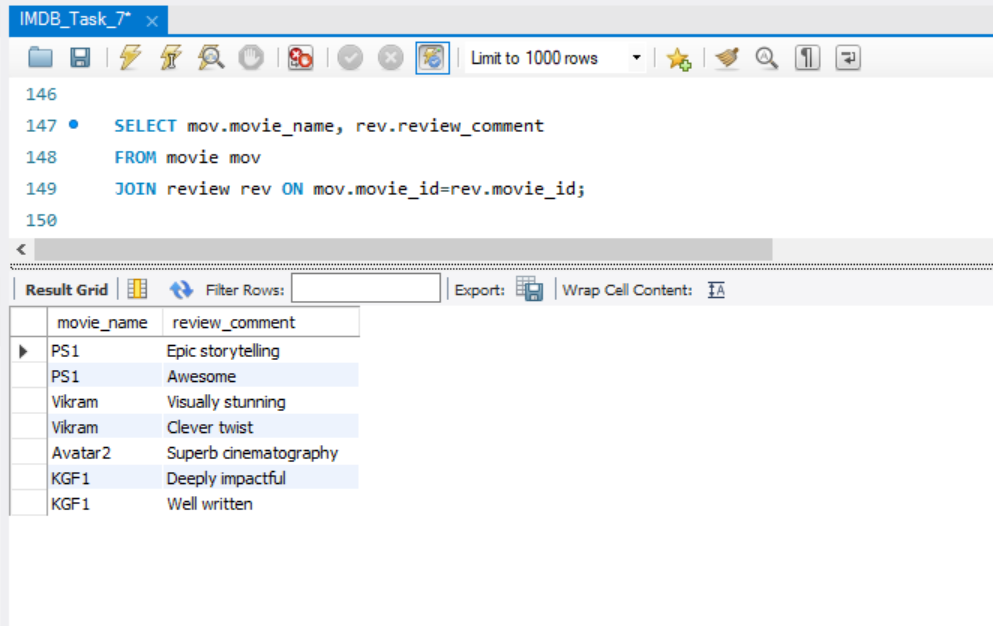
1. ADD budget column to movie and update budget of PS1
2. DROP the budget column using ALTER-DROP



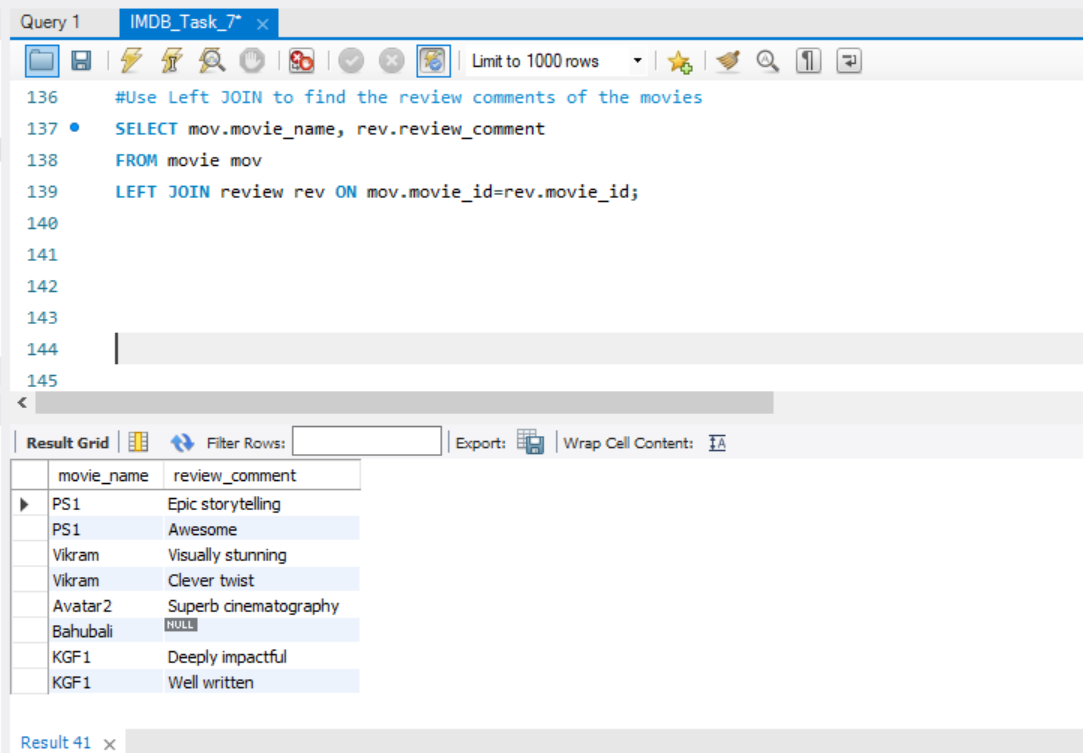
## JOIN

1. Select movie info along with media and genre details. Used INNER JOIN to combine 3 tables

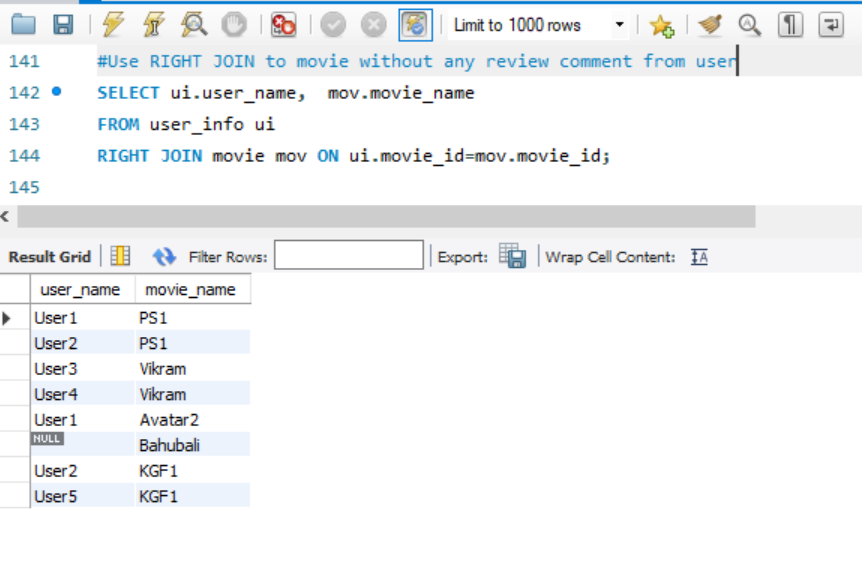
One more example INNER join between movie and review, where movie Bahubali is missed because it doesn’t have review comments. So, Inner join returns **only** the rows that are available in both tables.



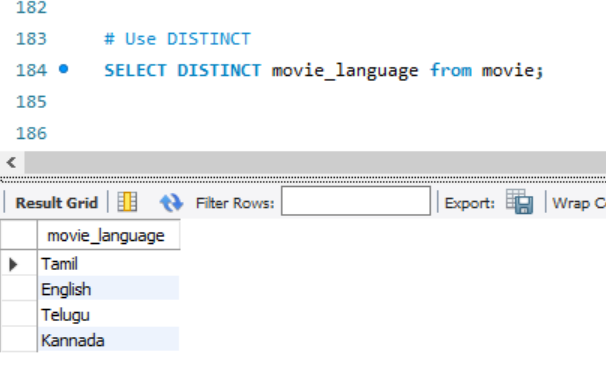
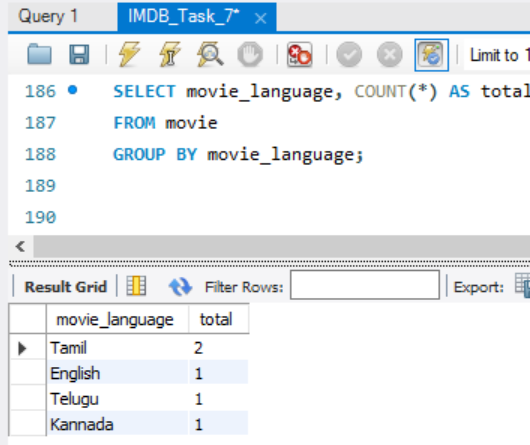
1. Show the review comments of the movies using LEFT JOIN. All the rows from left table is shown, Null values for missing values in the right table

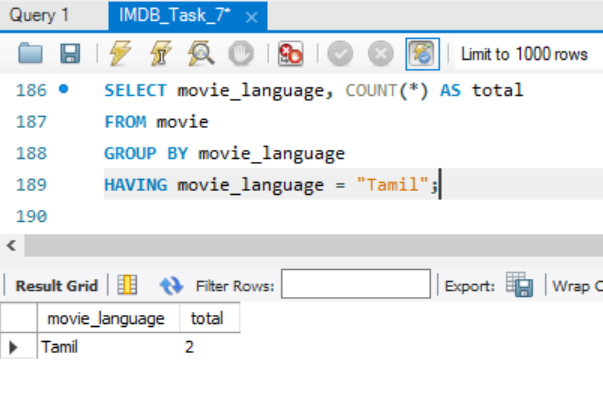


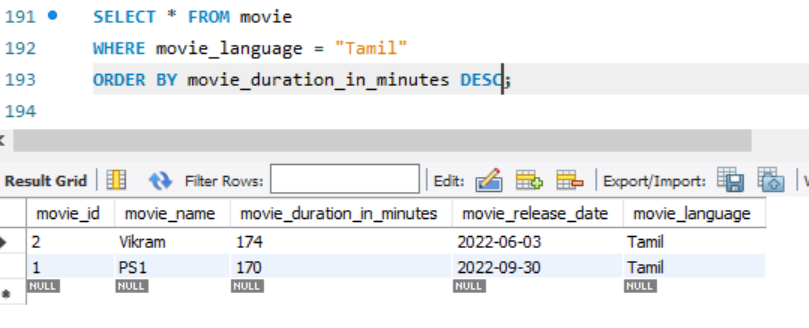
1. Find movies without review comments from user using RIGHT JOIN. All the rows from right table are shown, Null values for missing values in the left table



## Built-in Clauses

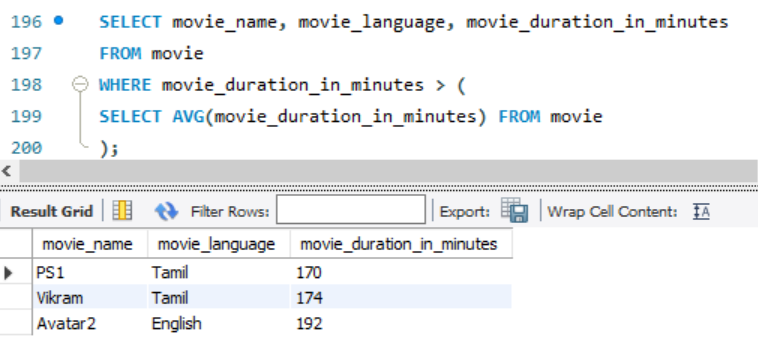
1. DISTINCT to get the distinct values in a selected column
2. Use of GROUP BY , COUNT and AS to find the count of each value in a given column
3. Combined GROUP BY with HAVING condition



1. ORDER BY
2. 

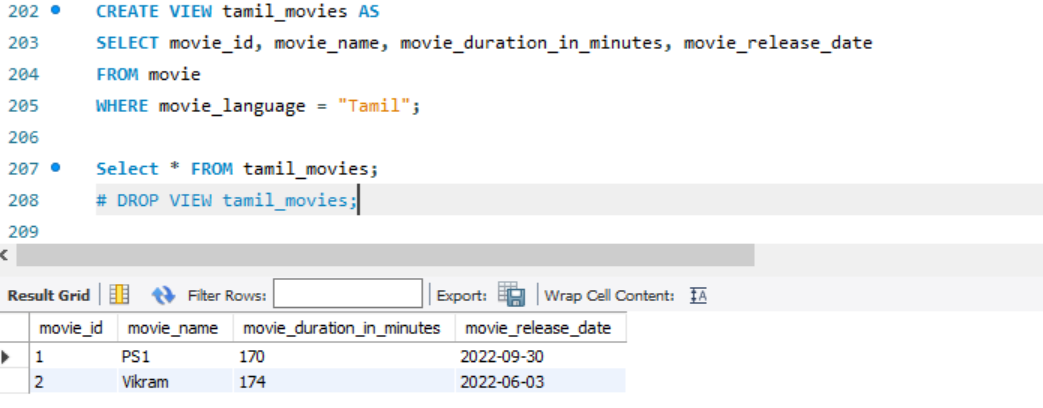
## SUB-QUERY:

Using Sub query at WHERE condition to find out the movies with duration more than the average movie duration.



## VIEW:

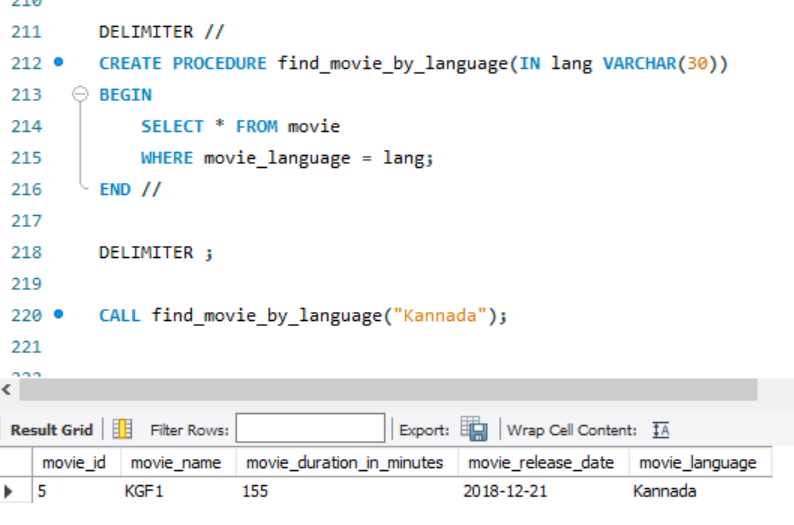
Using view to create virtual table for Tamil movies.

1. 

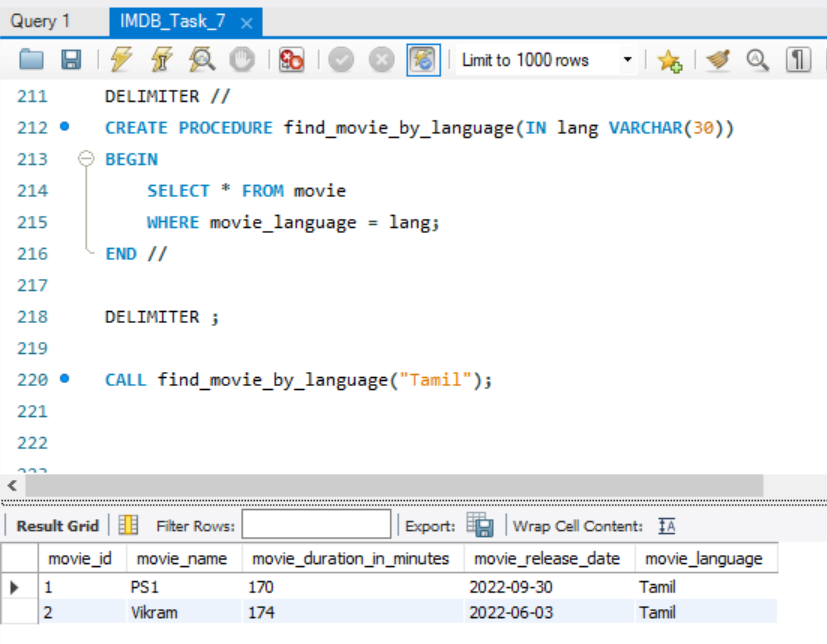
## STORED PROCEDURE

Used DELIMETER, BEGIN and END to create a procedure to filter the movies based on the language.

Example 1:



Example 2:



## TRANSACTIONS

Step 1: Added ticket price column to movie table

Step 2: Updated the ticket price to the movies using transaction

Step 3: Committed the changes.

