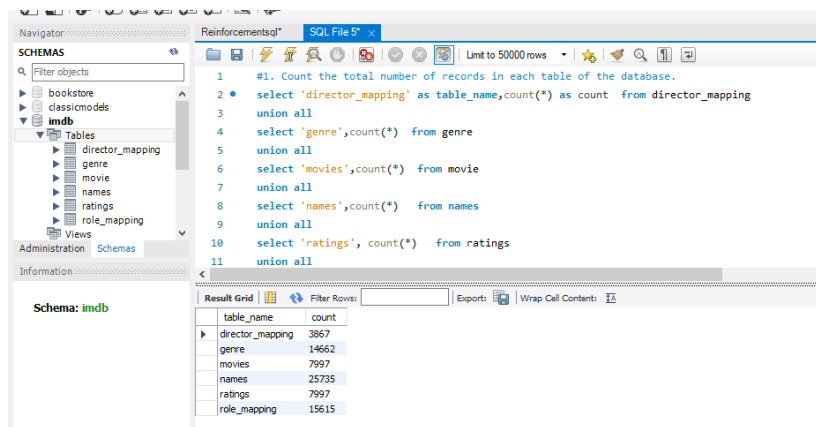


1. Count the total number of records in each table of the database.

QUERY

```
select 'director_mapping' as table_name,count(*) as count from director_mapping
union all
select 'genre',count(*) from genre
union all
select 'movies',count(*) from movie
union all
select 'names',count(*) from names
union all
select 'ratings', count(*) from ratings
union all
select 'role_mapping',count(*) from role_mapping;
```



The screenshot shows a SQL IDE window titled 'Reinforcementai*' with a tab for 'SQL File 5'. The left sidebar shows a 'SCHEMAS' tree with 'imdb' selected. The main editor displays the SQL query from the previous block. The bottom pane shows the 'Result Grid' with the following data:

table_name	count
director_mapping	3867
genre	14662
movies	7997
names	25735
ratings	7997
role_mapping	15615

2. Identify which columns in the movie table contain null values.

QUERY

```
select 'country' as table_name, count(*) as count from movie where country is null
union all
select 'worldwide_gross_income',count(*) from movie where worldwide_gross_income
is null
union all
select 'languages',count(*) from movie where languages is null
union all
select 'production_company',count(*) from movie where production_company is null;
```

Reinforcementsql SQL File 5

```

1 #2. Identify which columns in the movie table contain null values.
2 select 'country' as table_name, count(*) as count from movie where country is null
3 union all
4 select 'worldwide_gross_income', count(*) from movie where worldwide_gross_income is null
5 union all
6 select 'languages', count(*) from movie where languages is null
7 union all
8 select 'production_company', count(*) from movie where production_company is null;

```

Result Grid

table_name	count
country	20
worldwide_gross_income	3724
languages	194
production_company	528

- Determine the total number of movies released each year, and analyze how the trend changes month-wise.

QUERY

```

select year, count(*) as movie_count from movie group by year order by year;
select year, month(date_published) as release_month, count(*) as movie_count from movie group by year, release_month order by year, release_month;

```

Reinforcementsql SQL File 3

```

1 #3. Determine the total number of movies released each year, and analyze how the trend changes month-wise.
2 select year, count(*) as movie_count from movie group by year order by year;
3 select year, month(date_published) as release_month, count(*) as movie_count from movie group by year, release_month order by year, release_month;
4

```

Result Grid

year	movie_count
2017	3052
2018	2944
2019	2001

- How many movies were produced in either the USA or India in the year 2019?

QUERY

```

select count(*) as movie_count from movie where (country LIKE '%USA%' OR country LIKE '%India%') and year = '2019';

```

Reinforcementsql SQL File 3*	
Limit to 50000 rows	
1	#4. How many movies were produced in either the USA or India in the year 2019?
2	select count(*) as movie_count from movie where (country LIKE '%USA%' OR country LIKE '%India%') and year = '2019';
3	

Result Grid	Filter Rows:	Export:	Wrap Cell Contents:
movie_count			
1059			

5. List the unique genres in the dataset, and count how many movies belong exclusively to one genre.

QUERY

```
select distinct genre from genre;
select count(*) as exclusive_count from (select movie_id from genre group by
movie_id having count(*) = 1) as exclusive_genre ;
```

Reinforcementsql SQL File 3*	
Limit to 50000 rows	
1	#5. List the unique genres in the dataset, and count how many movies belong exclusively to one genre.
2	select distinct genre from genre;
3	select count(*) as exclusive_count from (select movie_id from genre group by movie_id having count(*) = 1) as exclusive_genre ;
4	

Result Grid	Filter Rows:	Export:	Wrap Cell Contents:
genre			
Drama			
Fantasy			
Thriller			
Comedy			
Horror			
Family			
Romance			
Adventure			
Action			
Sci-Fi			

6. Which genre has the highest total number of movies produced?

QUERY

```
select genre, count(*)as movie_count from genre group by genre order by
movie_count desc limit 1;
```

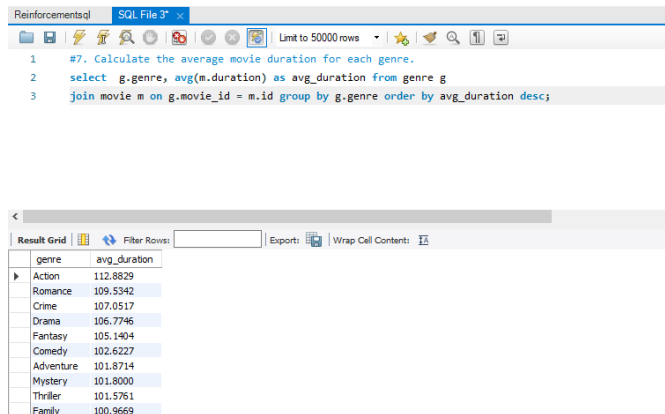
Reinforcementsql SQL File 3*	
Limit to 50000 rows	
1	#6. Which genre has the highest total number of movies produced?
2	select genre, count(*)as movie_count from genre group by genre order by movie_count desc limit 1;
3	

Result Grid	Filter Rows:	Export:	Wrap Cell Contents:	Fetch rows:
genre movie_count				
Drama 4285				

7. Calculate the average movie duration for each genre.

QUERY

```
select g.genre, avg(m.duration) as avg_duration from genre g
join movie m on g.movie_id = m.id group by g.genre order by avg_duration desc;
```



The screenshot shows a SQL editor window titled "Reinforcementsql" with a file named "SQL File 3*". The query is as follows:

```
1 #7. Calculate the average movie duration for each genre.
2 select g.genre, avg(m.duration) as avg_duration from genre g
3 join movie m on g.movie_id = m.id group by g.genre order by avg_duration desc;
```

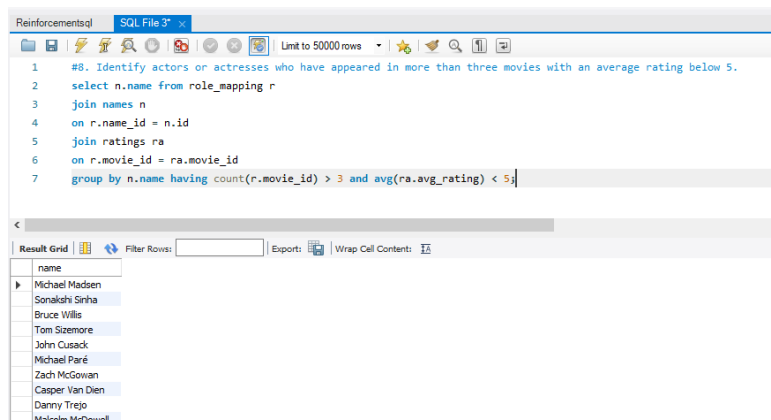
The results are displayed in a table with the following data:

genre	avg_duration
Action	112.8829
Romance	109.5342
Crime	107.0517
Drama	106.7746
Fantasy	105.1404
Comedy	102.6227
Adventure	101.8714
Mystery	101.8000
Thriller	101.5761
Familv	100.9669

8. Identify actors or actresses who have appeared in more than three movies with an average rating below 5.

QUERY

```
select n.name from role_mapping r
join names n
on r.name_id = n.id
join ratings ra
on r.movie_id = ra.movie_id
group by n.name having count(r.movie_id) > 3 and avg(ra.avg_rating) < 5;
```



The screenshot shows a SQL editor window titled "Reinforcementsql" with a file named "SQL File 3*". The query is as follows:

```
1 #8. Identify actors or actresses who have appeared in more than three movies with an average rating below 5.
2 select n.name from role_mapping r
3 join names n
4 on r.name_id = n.id
5 join ratings ra
6 on r.movie_id = ra.movie_id
7 group by n.name having count(r.movie_id) > 3 and avg(ra.avg_rating) < 5;
```

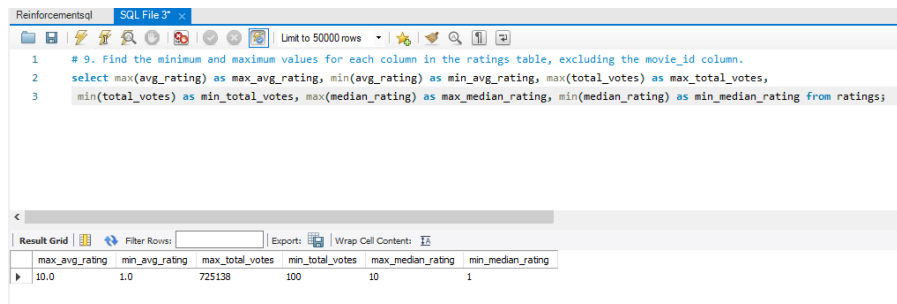
The results are displayed in a table with the following data:

name
Michael Madsen
Sonakshi Sinha
Bruce Willis
Tom Sizemore
John Cusack
Michael Pare
Zach McGowan
Casper Van Dien
Danny Trejo
Malcolm McDowell

9. Find the minimum and maximum values for each column in the ratings table, excluding the movie_id column.

QUERY

```
select max(avg_rating) as max_avg_rating, min(avg_rating) as min_avg_rating,  
max(total_votes) as max_total_votes,  
min(total_votes) as min_total_votes, max(median_rating) as max_median_rating,  
min(median_rating) as min_median_rating from ratings;
```



The screenshot shows a SQL IDE window titled 'Reinforcementsql' with a file named 'SQL File 3'. The query editor contains the following SQL code:

```
# 9. Find the minimum and maximum values for each column in the ratings table, excluding the movie_id column.  
select max(avg_rating) as max_avg_rating, min(avg_rating) as min_avg_rating, max(total_votes) as max_total_votes,  
min(total_votes) as min_total_votes, max(median_rating) as max_median_rating, min(median_rating) as min_median_rating from ratings;
```

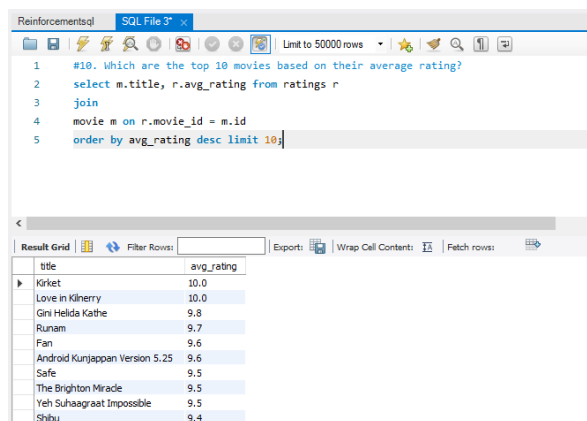
The results grid below the query editor displays the following data:

max_avg_rating	min_avg_rating	max_total_votes	min_total_votes	max_median_rating	min_median_rating
10.0	1.0	725138	100	10	1

10. Which are the top 10 movies based on their average rating?

QUERY

```
select m.title, r.avg_rating from ratings r  
join  
movie m on r.movie_id = m.id  
order by avg_rating desc limit 10;
```



The screenshot shows a SQL IDE window titled 'Reinforcementsql' with a file named 'SQL File 3'. The query editor contains the following SQL code:

```
#10. Which are the top 10 movies based on their average rating?  
select m.title, r.avg_rating from ratings r  
join  
movie m on r.movie_id = m.id  
order by avg_rating desc limit 10;
```

The results grid below the query editor displays the following data:

title	avg_rating
Kirkit	10.0
Love in Kineri	10.0
Gini Helda Kathe	9.8
Runam	9.7
Fan	9.6
Android Kunjappa Version 5.25	9.6
Safe	9.5
The Brighton Miracle	9.5
Yeh Suhaagraat Impossible	9.5
Shibu	9.4

11. Summarize the ratings table by grouping movies based on their median ratings.

QUERY

```
select median_rating, count(*) as movie_count from ratings group by median_rating  
order by movie_count desc ;
```

ReinforcementSQL SQL File 3

```

1 #11. Summarize the ratings table by grouping movies based on their median ratings.
2 select median_rating, count(*) as movie_count from ratings group by median_rating order by movie_count desc ;
3

```

Result Grid

median_rating	movie_count
7	2257
6	1975
8	1030
5	985
4	479
9	429
10	346
3	283
2	119
1	94

12. How many movies, released in March 2017 in the USA within a specific genre, had more than 1,000 votes?

QUERY

```

select m.title,g.genre from movie m
join genre g
on m.id = g.movie_id
join ratings r
on m.id = r.movie_id
where m.country like '%USA%' and m.date_published like '2017-03-%' and
r.total_votes >1000;

```

ReinforcementSQL SQL File 3

```

1 #12. How many movies, released in March 2017 in the USA within a specific genre, had more than 1,000 votes?
2 select m.title,g.genre from movie m
3 join genre g
4 on m.id = g.movie_id
5 join ratings r
6 on m.id = r.movie_id
7 where m.country like 'USA%' and m.date_published like '2017-03-%' and r.total_votes >1000;
8

```

Result Grid

title	genre
CHIPS	Action
CHIPS	Comedy
CHIPS	Crime
The Lost City of Z	Drama
Ghost in the Shell	Action
Ghost in the Shell	Drama
Ghost in the Shell	Sci-Fi
Before I Fall	Drama
Before I Fall	Fantasy
Before I Fall	Mystery

13. Find movies from each genre that begin with the word “The” and have an average rating greater than 8.

QUERY

```

SELECT m.title, g.genre, r.avg_rating from movie m
join genre g on m.id = g.movie_id
join ratings r on m.id = r.movie_id
where m.title LIKE 'The%'
and r.avg_rating > 8;

```

Reinforcementsql SQL File 3"

```

1 #13. Find movies from each genre that begin with the word "The" and have an average rating greater than 8.
2 SELECT m.title, g.genre, r.avg_rating from movie m
3 join genre g on m.id = g.movie_id
4 join ratings r on m.id = r.movie_id
5 where m.title LIKE 'The%'
6 and r.avg_rating > 8;
7

```

Result Grid

title	genre	avg_rating
The Blue Elephant 2	Drama	8.8
The Blue Elephant 2	Horror	8.8
The Blue Elephant 2	Mystery	8.8
The Brighton Mirade	Drama	9.5
The Irishman	Crime	8.7
The Irishman	Drama	8.7
The Colour of Darkness	Drama	9.1
Theeran Adhigaaram Ondru	Action	8.3
Theeran Adhigaaram Ondru	Crime	8.3
Theeran Adhigaaram Ondru	Thriller	8.3

Result 30 x

14. Of the movies released between April 1, 2018, and April 1, 2019, how many received a median rating of 8?

QUERY

```

select count(*) as 'movie count = 8' from movie m
join ratings r
on m.id = r.movie_id
where m.date_published between '2018-04-01' and '2019-04-01' and r.median_rating
= 8;

```

Reinforcementsql SQL File 3"

```

1 #14. Of the movies released between April 1, 2018, and April 1, 2019, how many received a median rating of 8?
2 select count(*) as 'movie count = 8' from movie m
3 join ratings r
4 on m.id = r.movie_id
5 where m.date_published between '2018-04-01' and '2019-04-01' and r.median_rating = 8;
6

```

Result Grid

movie count = 8
361

15. Do German movies receive more votes on average than Italian movies?

QUERY

```

select m.country, avg(r.total_votes) from movie m
join ratings r
on m.id = r.movie_id
where m.country in ("Germany", "Italy")
group by m.country;

```

ReinforcementSQL SQL File 3*							
<pre> 1 #15. Do German movies receive more votes on average than Italian movies? 2 select m.country, avg(r.total_votes) from movie m 3 join ratings r 4 on m.id = r.movie_id 5 where m.country in ("Germany", "Italy") 6 group by m.country; 7 </pre>							
<table border="1"> <thead> <tr> <th>country</th><th>avg(r.total_votes)</th></tr> </thead> <tbody> <tr> <td>Germany</td><td>730.8904</td></tr> <tr> <td>Italy</td><td>633.8618</td></tr> </tbody> </table>		country	avg(r.total_votes)	Germany	730.8904	Italy	633.8618
country	avg(r.total_votes)						
Germany	730.8904						
Italy	633.8618						

16. Identify the columns in the names table that contain null values.

QUERY

```

select 'height' as table_name, count(*) as null_count from names where height is null
union all
select 'date_of_birth', count(*) from names where date_of_birth is null
union all
select 'known_for_movies', count(*) from names where known_for_movies is null;

```

ReinforcementSQL SQL File 5*									
<pre> 1 #16. Identify the columns in the names table that contain null values. 2 select 'height' as table_name, count(*) as null_count from names where height is null 3 union all 4 select 'date_of_birth', count(*) from names where date_of_birth is null 5 union all 6 select 'known_for_movies', count(*) from names where known_for_movies is null; </pre>									
<table border="1"> <thead> <tr> <th>table_name</th><th>null_count</th></tr> </thead> <tbody> <tr> <td>height</td><td>17335</td></tr> <tr> <td>date_of_birth</td><td>13431</td></tr> <tr> <td>known_for_movies</td><td>15226</td></tr> </tbody> </table>		table_name	null_count	height	17335	date_of_birth	13431	known_for_movies	15226
table_name	null_count								
height	17335								
date_of_birth	13431								
known_for_movies	15226								

17. Who are the top two actors whose movies have a median rating of 8 or higher?

QUERY

```

select n.name, rm.category, r.median_rating from names n
join role_mapping rm
on id = name_id
join ratings r
on rm.movie_id = r.movie_id
where rm.category = 'actor' and r.median_rating >= 8
order by median_rating desc limit 2;

```


Reinforcementsql

SQL File 3

Limit to 50000 rows

18. Which are the top three production companies based on the total number of votes their movies received?

QUERY

```

select m.production_company, sum(r.total_votes) as total_votes from movie m
join ratings r
on m.id = r.movie_id group by production_company order by total_votes desc limit 3;

```

Reinforcementsql									
SQL File 3*									
Limit to 50000 rows									
<pre> 1 #18. Which are the top three production companies based on the total number of votes their movies received? 2 select m.production_company, sum(r.total_votes) as total_votes from movie m 3 join ratings r 4 on m.id = r.movie_id group by production_company order by total_votes desc limit 3; </pre>									
<table border="1"> <thead> <tr> <th>production_company</th><th>total_votes</th></tr> </thead> <tbody> <tr> <td>Marvel Studios</td><td>2656967</td></tr> <tr> <td>Twentieth Century Fox</td><td>2411163</td></tr> <tr> <td>Warner Bros.</td><td>2396057</td></tr> </tbody> </table>		production_company	total_votes	Marvel Studios	2656967	Twentieth Century Fox	2411163	Warner Bros.	2396057
production_company	total_votes								
Marvel Studios	2656967								
Twentieth Century Fox	2411163								
Warner Bros.	2396057								

19. How many directors have worked on more than three movies?

QUERY

```

select count(*) as director_count from
(select name_id from director_mapping
group by name_id having count(movie_id)>3) as director ;

```

ReinforcementSQL

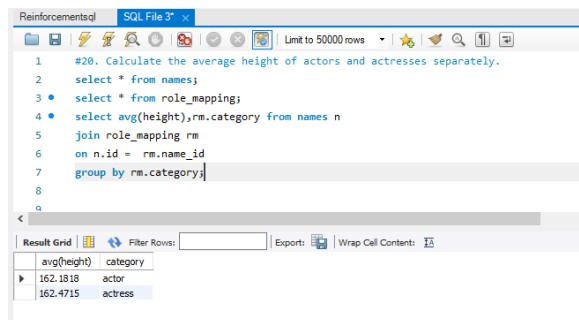
SQL File 3*

</

20. Calculate the average height of actors and actresses separately.

QUERY

```
select * from names;
select * from role_mapping;
select avg(height),rm.category from names n
join role_mapping rm
on n.id = rm.name_id
group by rm.category;
```



The screenshot shows a SQL editor window titled 'SQL File 3*' with the following query:

```
1 #20. Calculate the average height of actors and actresses separately.
2 select * from names;
3 select * from role_mapping;
4 select avg(height),rm.category from names n
5 join role_mapping rm
6 on n.id = rm.name_id
7 group by rm.category;
```

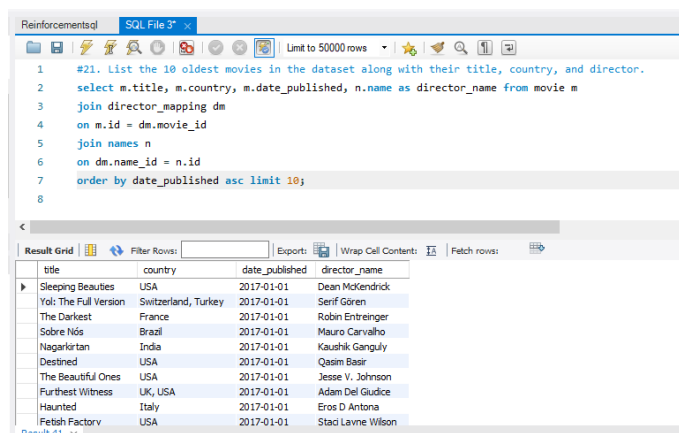
Below the query is the 'Result Grid' showing the results of the query:

avg(height)	category
162.1818	actor
162.4715	actress

21. List the 10 oldest movies in the dataset along with their title, country, and director.

QUERY

```
select m.title, m.country, m.date_published, n.name as director_name from movie m
join director_mapping dm
on m.id = dm.movie_id
join names n
on dm.name_id = n.id
order by date_published asc limit 10;
```



The screenshot shows a SQL editor window titled 'SQL File 3*' with the following query:

```
1 #21. List the 10 oldest movies in the dataset along with their title, country, and director.
2 select m.title, m.country, m.date_published, n.name as director_name from movie m
3 join director_mapping dm
4 on m.id = dm.movie_id
5 join names n
6 on dm.name_id = n.id
7 order by date_published asc limit 10;
```

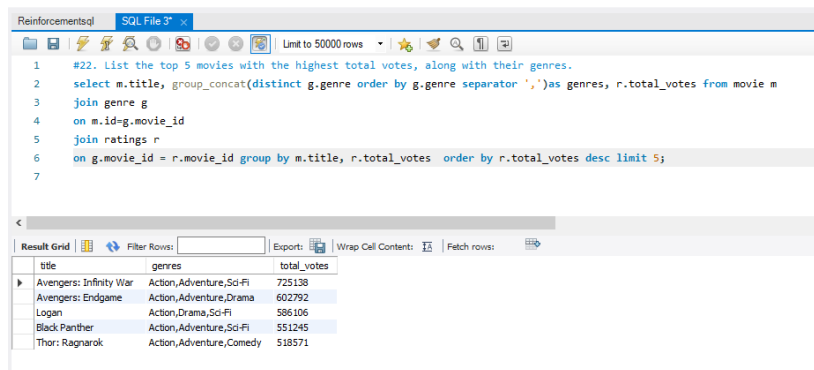
Below the query is the 'Result Grid' showing the results of the query:

title	country	date_published	director_name
Sleeping Beauties	USA	2017-01-01	Dean McKendrick
Yol: The Full Version	Switzerland, Turkey	2017-01-01	Serif Gören
The Darkest	France	2017-01-01	Robin Entreinger
Sobre Nós	Brazil	2017-01-01	Mauro Carvalho
Nagarikartan	India	2017-01-01	Kaushik Ganguly
Destined	USA	2017-01-01	Qasim Basir
The Beautiful Ones	USA	2017-01-01	Jesse V. Johnson
Furthest Witness	UK, USA	2017-01-01	Adam Del Giudice
Haunted	Italy	2017-01-01	Eros D'Antona
Fetish Factory	USA	2017-01-01	Staci Lavine Wilson

22. List the top 5 movies with the highest total votes, along with their genres.

QUERY

```
select m.title, group_concat(distinct g.genre order by g.genre separator ',')as genres,  
r.total_votes from movie m  
join genre g  
on m.id=g.movie_id  
join ratings r  
on g.movie_id = r.movie_id group by m.title, r.total_votes order by r.total_votes desc  
limit 5;
```



The screenshot shows a SQL IDE window titled 'Reinforcementsql' with a file named 'SQL File 3*'. The query is as follows:

```
1 #22. List the top 5 movies with the highest total votes, along with their genres.  
2 select m.title, group_concat(distinct g.genre order by g.genre separator ',')as genres, r.total_votes from movie m  
3 join genre g  
4 on m.id=g.movie_id  
5 join ratings r  
6 on g.movie_id = r.movie_id group by m.title, r.total_votes order by r.total_votes desc limit 5;  
7
```

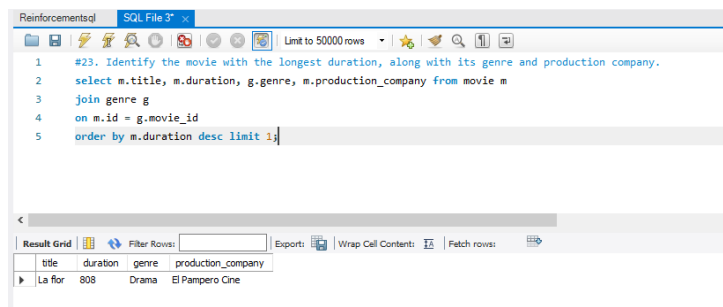
The result grid below the query shows the following data:

title	genres	total_votes
Avengers: Infinity War	Action,Adventure,Sci-Fi	725138
Avengers: Endgame	Action,Adventure,Drama	602792
Logan	Action,Drama,Sci-Fi	586106
Black Panther	Action,Adventure,Sci-Fi	551245
Thor: Ragnarok	Action,Adventure,Comedy	518571

23. Identify the movie with the longest duration, along with its genre and production company.

QUERY

```
select m.title, m.duration, g.genre, m.production_company from movie m  
join genre g  
on m.id = g.movie_id  
order by m.duration desc limit 1;
```



The screenshot shows a SQL IDE window titled 'Reinforcementsql' with a file named 'SQL File 3*'. The query is as follows:

```
1 #23. Identify the movie with the longest duration, along with its genre and production company.  
2 select m.title, m.duration, g.genre, m.production_company from movie m  
3 join genre g  
4 on m.id = g.movie_id  
5 order by m.duration desc limit 1;
```

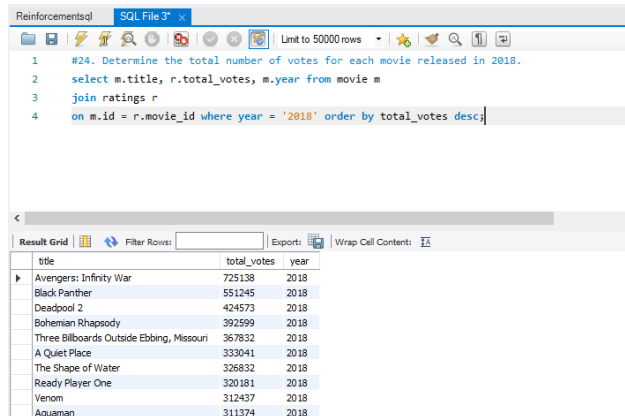
The result grid below the query shows the following data:

title	duration	genre	production_company
La flor	808	Drama	El Pampero Cine

24. Determine the total number of votes for each movie released in 2018.

QUERY

```
select m.title, r.total_votes, m.year from movie m
join ratings r
on m.id = r.movie_id where year = '2018' order by total_votes desc;
```



The screenshot shows a SQL IDE window titled 'Reinforcementsql' with a tab 'SQL File 3'. The query editor contains the following SQL code:

```
1 #24. Determine the total number of votes for each movie released in 2018.
2 select m.title, r.total_votes, m.year from movie m
3 join ratings r
4 on m.id = r.movie_id where year = '2018' order by total_votes desc;
```

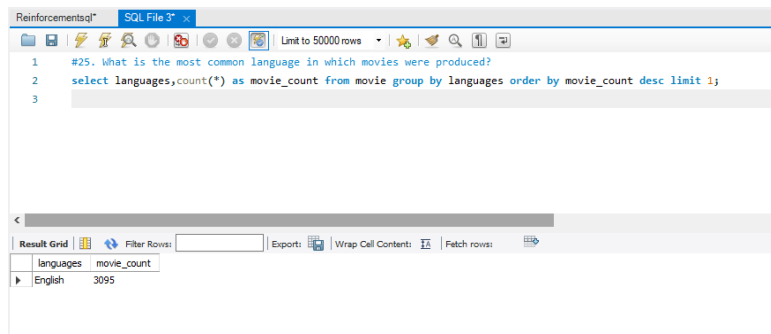
Below the query editor, the 'Result Grid' is displayed with the following data:

title	total_votes	year
Avengers: Infinity War	725138	2018
Black Panther	551245	2018
Deadpool 2	424573	2018
Bohemian Rhapsody	392599	2018
Three Billboards Outside Ebbing, Missouri	367832	2018
A Quiet Place	333041	2018
The Shape of Water	326832	2018
Ready Player One	320181	2018
Venom	312437	2018
Aquaman	311374	2018

25. What is the most common language in which movies were produced?

QUERY

```
select languages,count(*) as movie_count from movie group by languages order by
movie_count desc limit 1;
```



The screenshot shows a SQL IDE window titled 'Reinforcementsql' with a tab 'SQL File 3'. The query editor contains the following SQL code:

```
1 #25. What is the most common language in which movies were produced?
2 select languages,count(*) as movie_count from movie group by languages order by movie_count desc limit 1;
3
```

Below the query editor, the 'Result Grid' is displayed with the following data:

languages	movie_count
English	3095