

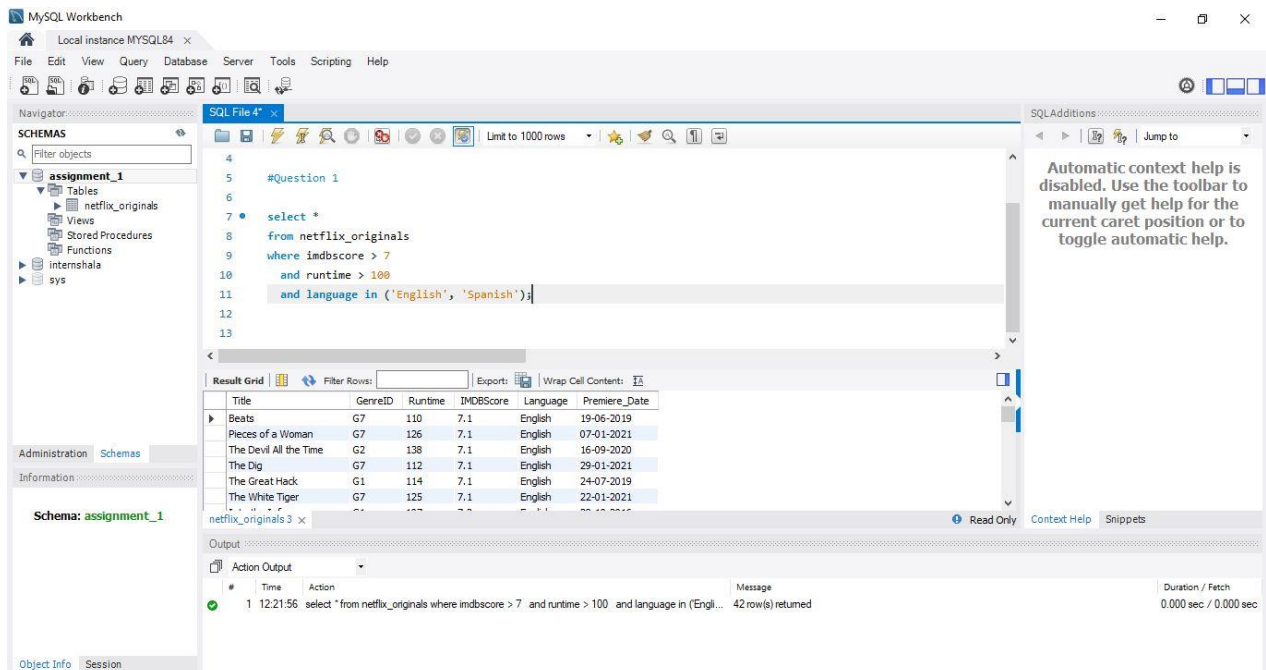
MySQL

ASSIGNMENT 1ST

Problem statement : The task involves analyzing the Netflix Originals dataset to derive insights about movie genres, runtime, IMDb scores, and premiere dates. The dataset holds valuable information regarding the content Netflix produces, and understanding these attributes can help identify trends and patterns. By using SQL queries, the aim is to perform complex filtering, aggregation, and sorting operations, providing meaningful insights for business decisions.

Task 1st: To retrieve all Netflix Originals with an IMDb score greater than 7, runtime greater than 100 minutes, and the language being either English or Spanish, I have used the following SQL query:

```
select *  
from netflix_originals  
where imdbscore > 7  
and runtime > 100  
and language in ('English', 'Spanish');
```



The screenshot displays the MySQL Workbench interface. The SQL Editor window shows the following query:

```
#Question 1  
  
select *  
from netflix_originals  
where imdbscore > 7  
and runtime > 100  
and language in ('English', 'Spanish');
```

The Results window shows the output of the query, which is a table with the following columns: Title, GenreID, Runtime, IMDbScore, Language, and Premiere_Date. The table contains 42 rows of data.

Title	GenreID	Runtime	IMDbScore	Language	Premiere_Date
Beats	G7	110	7.1	English	19-06-2019
Pieces of a Woman	G7	126	7.1	English	07-01-2021
The Devil All the Time	G2	138	7.1	English	16-09-2020
The Dig	G7	112	7.1	English	29-01-2021
The Great Hack	G1	114	7.1	English	24-07-2019
The White Tiger	G7	125	7.1	English	22-01-2021

The Output window shows the following message:

```
1 12:21:56 select * from netflix_originals where imdbscore > 7 and runtime > 100 and language in ('English', 'Spanish') 42 row(s) returned
```

Task 2nd: Finding the total number of Netflix Originals in each language, but only show those languages that have more than 5 titles.

select language, count(*) as totaltitles

from netflix_originals

group by language

having count(*) > 5;

The screenshot shows the MySQL Workbench interface. The SQL Editor contains the following query:

```
#Question 2
select language, count(*) as totaltitles
from netflix_originals
group by language
having count(*) > 5;
```

The Results window displays the following data:

language	totaltitles
English	420
Spanish	34
Italian	14
Hindi	33
Korean	6
Indonesian	9

The Output window shows the execution log:

#	Time	Action	Message	Duration / Fetch
1	12:21:56	select * from netflix_originals where imdbscore > 7 and runtime > 100 and language in (Engl...	42 row(s) returned	0.000 sec / 0.000 sec
2	12:26:52	select language, count(*) as totaltitles from netflix_originals group by language having count(*) > 5	9 row(s) returned	0.016 sec / 0.000 sec

Task 3rd: To get the top 3 longest-running movies in Hindi language sorted by IMDb score in descending order

select title, runtime, imdbscore

from netflix_originals

where language = 'hindi'

order by runtime desc, imdbscore desc

limit 3;

The screenshot shows the MySQL Workbench interface. The SQL Editor contains the following query:

```
#Question 3
select title, runtime, imdbscore
from netflix_originals
where language = 'hindi'
order by runtime desc, imdbscore desc
limit 3;
```

The Results Grid shows the following data:

title	runtime	imdbscore
Ludo	149	7.6
Raat Akeli Hai	149	7.3
Drive	147	3.5

The Output tab shows the following messages:

#	Time	Action	Message	Duration / Fetch
1	12:21:56	select * from netflix_originals where imdbscore > 7 and runtime > 100 and language in (Engl...	42 row(s) returned	0.000 sec / 0.000 sec
2	12:26:52	select language, count(*) as totaltitles from netflix_originals group by language having count(*) ...	9 row(s) returned	0.016 sec / 0.000 sec
3	12:29:49	select title, runtime, imdbscore from netflix_originals where language = 'hindi' order by runtime d...	3 row(s) returned	0.000 sec / 0.000 sec

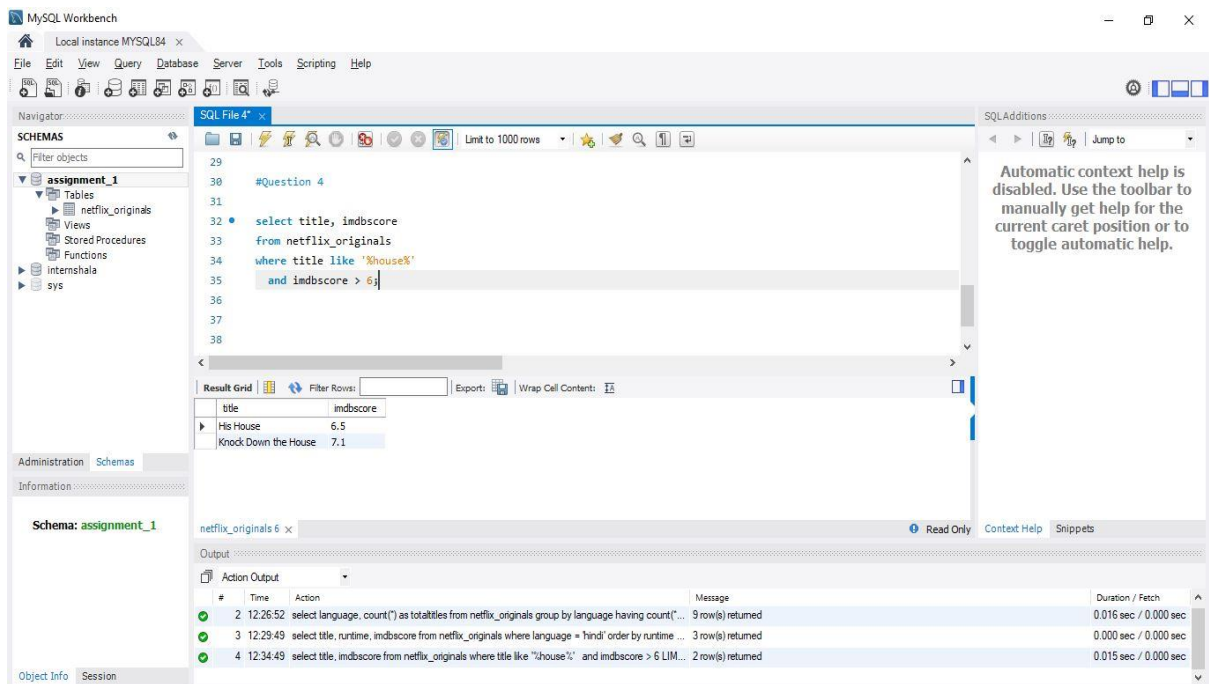
Task 4th : To retrieve all titles that contain the word "House" in their name and have an IMDb score greater than 6.

select title, imdbscore

from netflix_originals

where title like '%house%'

and imdbscore > 6;



The screenshot shows the MySQL Workbench interface. The SQL Editor contains the following query:

```
#Question 4
select title, imdbscore
from netflix_originals
where title like '%house%'
and imdbscore > 6;
```

The Results Grid shows the following data:

title	imdbscore
His House	6.5
Knock Down the House	7.1

The Output tab at the bottom shows the execution log:

#	Time	Action	Message	Duration / Fetch
2	12:26:52	select language, count(*) as totaltitles from netflix_originals group by language having count(*) > 9	9 row(s) returned	0.016 sec / 0.000 sec
3	12:29:49	select title, runtime, imdbscore from netflix_originals where language = 'hindi' order by runtime	3 row(s) returned	0.000 sec / 0.000 sec
4	12:34:49	select title, imdbscore from netflix_originals where title like "%house%" and imdbscore > 6	2 row(s) returned	0.015 sec / 0.000 sec

Task 5th: Finding all Netflix Originals released between the years 2018 and 2020 that are in either English, Spanish, or Hindi.

select title, premiere_date, language

from netflix_originals

where year(premiere_date) between 2018 and 2020

and language in ('english', 'spanish', 'hindi');

The screenshot shows the MySQL Workbench interface. The SQL Editor contains the following query:

```
#Question 5
select title, premiere_date, language
from netflix_originals
where year(premiere_date) between 2018 and 2020
and language in ('english', 'spanish', 'hindi');
```

The query is executed, and the results are displayed in the Result Grid. The columns are title, premiere_date, and language. The output shows 3 rows of data.

The Output pane shows the execution log:

#	Time	Action	Message	Duration / Fetch
3	12:29:49	select title, runtime, imdbscore from netflix_originals where language = 'hindi' order by runtime ...	3 row(s) returned	0.000 sec / 0.000 sec
4	12:34:49	select title, imdbscore from netflix_originals where title like "%house%" and imdbscore > 6 LIM...	2 row(s) returned	0.015 sec / 0.000 sec
5	12:40:28	select title, premiere_date, language from netflix_originals where year(premiere_date) between...	0 row(s) returned	0.000 sec / 0.000 sec

Task 6th : Finding all movies that either have a runtime less than 60 minutes or an IMDb score less than 5, sorted by Premiere Date.

select title, runtime, imdbscore, premiere_date

from netflix_originals

where runtime < 60 or imdbscore < 5

order by premiere_date;

The screenshot shows the MySQL Workbench interface. The SQL Editor contains the following query:

```
41 where year(premiere_date) between 2018 and 2020
42 and language in ('english', 'spanish', 'hindi');
43
44 #Question 6
45
46 • select title, runtime, imdbscore, premiere_date
47 from netflix_originals
48 where runtime < 60 or imdbscore < 5
49 order by premiere_date;
50
```

The Result Grid shows the following data:

	title	runtime	imdbscore	premiere_date
▶	Ghost Stories	144	4.3	01-01-2020
	What Happened to Mr. Cha?	102	4.3	01-01-2021
	The Minimalists: Less Is Now	53	5.9	01-01-2021
	Mrs. Serial Killer	106	4.8	01-05-2020
	Resurface	27	7	01-09-2017
	All Because of You	101	4.2	01-10-2020

The Output pane shows the following messages:

#	Time	Action	Message	Duration / Fetch
4	12:34:49	select title, imdbscore from netflix_originals where title like "house%" and imdbscore > 6 LIM...	2 row(s) returned	0.015 sec / 0.000 sec
5	12:40:28	select title, premiere_date, language from netflix_originals where year(premiere_date) between...	0 row(s) returned	0.000 sec / 0.000 sec
6	12:41:47	select title, runtime, imdbscore, premiere_date from netflix_originals where runtime < 60 or ind...	117 row(s) returned	0.000 sec / 0.000 sec

Task 7th: To get the average IMDb score for each genre where the genre has at least 10 movies.

```
select genreid, avg(imdbscore) as averageimdbscore  
from netflix_originals  
group by genreid  
having count(*) >= 10;
```

The screenshot shows the MySQL Workbench interface. The central editor displays a SQL query for 'Question 7' which is the same as the one in the task description. The 'Result Grid' below the query shows the results of the query, with columns 'genreid' and 'averageimdbscore'. The results are as follows:

genreid	averageimdbscore
G1	6.936477987421385
G2	5.697916666666668
G3	5.6117647058823525
G9	5.3578947368421055
G5	5.745
G6	5.743023255813954

The 'Action Output' pane at the bottom shows the execution log for three queries. The third query, which is the one shown in the editor, returned 11 rows in 0.000 seconds.

#	Time	Action	Message	Duration / Fetch
5	12:40:28	select title, premiere_date, language from netflix_originals where year(premiere_date) between...	0 row(s) returned	0.000 sec / 0.000 sec
6	12:41:47	select title, runtime, imdbscore, premiere_date from netflix_originals where runtime < 60 or ind...	117 row(s) returned	0.000 sec / 0.000 sec
7	12:43:45	select genreid, avg(imdbscore) as averageimdbscore from netflix_originals group by genreid h...	11 row(s) returned	0.000 sec / 0.000 sec

Task 8th: To retrieve the top 5 most common runtimes for Netflix Originals.

select runtime, count(*) as count

from netflix_originals

group by runtime

order by count desc

limit 5;

The screenshot shows the MySQL Workbench interface. The SQL editor contains the following query:

```
55 group by genreid
56 having count(*) >= 10;
57
58 #Question 8
59
60 • select runtime, count(*) as count
61   from netflix_originals
62  group by runtime
63  order by count desc
64  limit 5;
```

The Results window displays the following data:

runtime	count
97	24
94	19
98	19
95	18
100	17

The Action Output window shows the following messages:

#	Time	Action	Message	Duration / Fetch
6	12:41:47	select title, runtime, imdbscore, premiere_date from netflix_originals where runtime < 60 or imd...	117 row(s) returned	0.000 sec / 0.000 sec
7	12:43:45	select genreid, avg(imdbscore) as averageimdbscore from netflix_originals group by genreid h...	11 row(s) returned	0.000 sec / 0.000 sec
8	13:02:27	select runtime, count(*) as count from netflix_originals group by runtime order by count desc li...	5 row(s) returned	0.000 sec / 0.000 sec

Task 9th : To listing all Netflix Originals that were released in 2020, grouped by language, and show the total count of titles for each language.

select language, count(*) as totaltitles

from netflix_originals

where year(premiere_date) = 2020

group by language;

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'Schemas' panel with a tree view containing 'assignment_1', 'netflix_originals', 'Views', 'Stored Procedures', 'Functions', 'Internshala', and 'sys'. The main editor window, titled 'Assignment 1a1', contains the following SQL query:

```
63 order by count desc
64 limit 5;
65
66 #Question 9
67
68 • select language, count(*) as totaltitles
69   from netflix_originals
70   where year(premiere_date) = 2020
71   group by language;
72
```

Below the query editor, the 'Result Grid' is visible, showing a table with two columns: 'language' and 'totaltitles'. The 'Output' panel at the bottom displays the execution results:

#	Time	Action	Message	Duration / Fetch
7	12:43:45	select genreid, avg(imdbscore) as averageimdbscore from netflix_originals group by genreid h...	11 row(s) returned	0.000 sec / 0.000 sec
8	13:02:27	select runtime, count(*) as count from netflix_originals group by runtime order by count desc li...	5 row(s) returned	0.000 sec / 0.000 sec
9	13:05:37	select language, count(*) as totaltitles from netflix_originals where year(premiere_date) = 2020...	0 row(s) returned	0.000 sec / 0.000 sec

On the right side of the interface, a 'SQL Additions' panel is visible, displaying a message: 'Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.'

Task 10th: To creating a new table that enforces a constraint on the IMDb score to be between 0 and 10 and the runtime to be greater than 30 minutes.

```
create table netflix_originals_with_constraints (  
    title varchar(255),  
    genreid int,  
    runtime int check (runtime > 30),  
    imdbscore decimal(3,2) check (imdbscore between 0 and 10),  
    language varchar(50),  
    premiere_date date  
);
```

