



B. M. S. COLLEGE OF ENGINEERING

Lab Internals 2

Subject: DataBase Management Systems (MySQL)

Name	Kiran M K
USN	1BM19CS073
Sem & Section	4B
Dept.	CSE
Date	July 7, 2021

Question:

Consider the schema for Movie Database:

ACTOR(Act_id, Act_Name, Act_Gender)

DIRECTOR(Dir_id, Dir_Name, Dir_Phone)

MOVIES(Mov_id, Mov_Title, Mov_Year, Mov_Lang, Dir_id)

MOVIE_CAST(Act_id, Mov_id, Role)

RATING(Mov_id, Rev_Stars)

Write SQL queries to

1. List the titles of all movies directed by 'Hitchcock'.
2. Find the movie names where one or more actors acted in two or more movies.
3. List all actors who acted in a movie before 2000 and also in a movie after 2015 (use JOIN operation).
4. Find the title of movies and number of stars for each movie that has at least one rating and find the highest number of stars that movie received. Sort the result by movie title.
5. Update rating of all movies directed by 'Steven Spielberg' to 5

Code:

```
CREATE DATABASE LABEXAM2;  
USE LABEXAM2;
```

```
CREATE TABLE ACTOR(ACT_ID INT,  
ACT_NAME VARCHAR(30),  
ACT_GENDER VARCHAR(6),  
PRIMARY KEY(ACT_ID));
```

```
CREATE TABLE DIRECTOR(  
DIR_ID INT,  
DIR_NAME VARCHAR(50),  
DIR_PHONE INT,  
PRIMARY KEY(DIR_ID));
```

```
CREATE TABLE MOVIES(MOV_ID INT,  
MOV_TITLE VARCHAR(100),  
MOV_YEAR INT,  
MOV_LANG VARCHAR(30),
```

```
DIR_ID INT,  
PRIMARY KEY(MOV_ID),  
FOREIGN KEY(DIR_ID) REFERENCES DIRECTOR(DIR_ID) ON DELETE  
CASCADE ON UPDATE CASCADE);
```

```
CREATE TABLE MOVIE_CAST(ACT_ID INT,  
MOV_ID INT,  
ROLEP VARCHAR(30),  
FOREIGN KEY(ACT_ID) REFERENCES ACTOR(ACT_ID) ON DELETE  
CASCADE ON UPDATE CASCADE,  
FOREIGN KEY(MOV_ID) REFERENCES MOVIES(MOV_ID) ON DELETE  
CASCADE ON UPDATE CASCADE);
```

```
CREATE TABLE RATING(MOV_ID INT,  
STARS FLOAT,  
FOREIGN KEY(MOV_ID) REFERENCES MOVIES(MOV_ID) ON  
DELETE CASCADE ON UPDATE CASCADE);
```

```
INSERT INTO ACTOR VALUES(001, "HERBERT MARSHALL", "MALE"),  
(002, "NORAH BURING", "FEMALE"),  
(003, "EDWARD CHAPMAN", "MALE"),  
(004, "SAM NEILL", "MALE"),  
(005, "LAURA DERN", "FEMALE"),  
(006, "RAJ KUMAR", "MALE"),  
(007, "ANANT NAG", "MALE"),  
(008, "KALPANA", "FEMALE");
```

```
INSERT INTO DIRECTOR VALUES(001, "HITCHCOCK", 1234567890),  
(002, "STEVEN SPIELBERG", 1345167890),  
(003, "DORAI BHAGWAN", 1876543210),  
(004, "RISHAB SHETTY", 1789012345);
```

```
INSERT INTO MOVIES VALUES(001, "MURDER!", 1930, "ENGLISH",  
001),  
(002, "JURASSIC PARK", 1993, "ENGLISH", 002),  
(003, "ERADU KANASU", 1974, "KANNADA", 003),  
(004, "BAYALU DAARI", 1976, "KANNADA", 003),  
(005, "SARKARI HIRIYA PRATHAMIKA SHAALE KASARGOD", 2018,  
"KANNADA", 004);
```

```
INSERT INTO MOVIE_CAST VALUES(001, 001, "ACTOR-MANAGER"),
(002, 001, "ACTRESS"),
(003, 001, "STAGE-MANAGER"),
(004, 002, "HERO"),
(005, 002, "HEROINE"),
(006, 003, "HERO"),
(008, 003, "HEROINE"),
(007, 004, "HERO"),
(008, 004, "HEROINE"),
(007, 005, "LAWYER");
```

```
INSERT INTO RATING VALUES(001, 4),
(001, 5),
(001, 5),
(002, 3),
(002, 4),
(002, 5),
(003, 4),
(003, 4),
(003, 4),
(004, 4),
(004, 3),
(005, 3),
(005, 4),
(005, 5);
```

```
/*1. List the titles of all movies directed by 'Hitchcock'.*/
SELECT M.MOV_TITLE FROM MOVIES M, DIRECTOR D
WHERE D.DIR_NAME = "HITCHCOCK"
AND D.DIR_ID = M.DIR_ID;
```

```
/*2. Find the movie names where one or more actors acted in two or more
movies.*/
SELECT M.MOV_TITLE FROM MOVIES M, MOVIE_CAST MC
WHERE M.MOV_ID = MC.MOV_ID
AND MC.ACT_ID IN
(SELECT ACT_ID FROM MOVIE_CAST, MOVIES
GROUP BY ACT_ID HAVING COUNT(ACT_ID)>1)
GROUP BY MOV_TITLE
HAVING COUNT(*)>=2;
```

/*3. List all actors who acted in a movie before 2000 and also in a movie after 2015 (use JOIN operation).*/

```
SELECT A.ACT_NAME, M.MOV_TITLE, M.MOV_YEAR
FROM ACTOR A JOIN
MOVIE_CAST MCT
ON A.ACT_ID = MCT.ACT_ID
JOIN MOVIES M
ON MCT.MOV_ID = M.MOV_ID
WHERE M.MOV_YEAR NOT BETWEEN 2000 AND 2015;
```

/*4. Find the title of movies and number of stars for each movie that has at least one rating and find the highest number of stars that movie received. Sort the result by movie title.*/

```
SELECT MOV_TITLE, MAX(STARS)
FROM MOVIES
INNER JOIN RATING USING (MOV_ID)
GROUP BY MOV_TITLE
HAVING MAX(STARS)>0
ORDER BY MOV_TITLE;
```

/*5. Update rating of all movies directed by 'Steven Spielberg' to 5*/

```
UPDATE RATING
SET STARS = 5
WHERE MOV_ID IN (SELECT MOV_ID FROM MOVIES
WHERE DIR_ID IN (SELECT DIR_ID FROM DIRECTOR WHERE
DIR_NAME = "STEVEN SPIELBERG"));
```

```
SELECT R.STARS FROM RATING R, MOVIES M, DIRECTOR D
WHERE
D.DIR_NAME = "STEVEN SPIELBERG" AND M.DIR_ID = D.DIR_ID
AND R.MOV_ID = M.MOV_ID;
```

Output:

/*1. List the titles of all movies directed by 'Hitchcock'.*/

The screenshot shows a database query editor with a toolbar at the top. The SQL code is as follows:

```
75 (002, 5),
76 (003, 4),
77 (003, 4),
78 (003, 4),
79 (004, 4),
80 (004, 3),
81 (005, 3),
82 (005, 4),
83 (005, 5);
84
85 /*1. List the titles of all movies directed by 'Hitchcock'.*/
86 • SELECT M.MOV_TITLE FROM MOVIES M, DIRECTOR D
87 WHERE D.DIR_NAME = "HITCHCOCK"
88 AND D.DIR_ID = M.DIR_ID;
89
90 /*2. Find the movie names where one or more actors acted in two or more movies.*/
91 • SELECT M.MOV_TITLE FROM MOVIES M, MOVIE_CAST MC
92 WHERE M.MOV_ID = MC.MOV_ID
93 AND MC.ACT_ID IN
94 (SELECT ACT_ID FROM MOVIE_CAST, MOVIES
95 GROUP BY ACT_ID HAVING COUNT(ACT_ID)>1)
```

The result grid at the bottom shows a table with the following data:

MOV_TITLE
MURDER!

2. Find the movie names where one or more actors acted in two or more movies.

The screenshot shows a database query editor with a toolbar at the top. The SQL code is as follows:

```
83 (005, 5);
84
85 /*1. List the titles of all movies directed by 'Hitchcock'.*/
86 • SELECT M.MOV_TITLE FROM MOVIES M, DIRECTOR D
87 WHERE D.DIR_NAME = "HITCHCOCK"
88 AND D.DIR_ID = M.DIR_ID;
89
90 /*2. Find the movie names where one or more actors acted in two or more movies.*/
91 • SELECT M.MOV_TITLE FROM MOVIES M, MOVIE_CAST MC
92 WHERE M.MOV_ID = MC.MOV_ID
93 AND MC.ACT_ID IN
94 (SELECT ACT_ID FROM MOVIE_CAST, MOVIES
95 GROUP BY ACT_ID HAVING COUNT(ACT_ID)>1)
96 GROUP BY MOV_TITLE
97 HAVING COUNT(*)>=2;
98
99 /*3. List all actors who acted in a movie before 2000 and also in a movie after 2015 (use JOIN
100 operation).*/
101 • SELECT A.ACT_NAME, M.MOV_TITLE, M.MOV_YEAR
102 FROM ACTOR A JOIN
103 MOVIE_CAST MCT
```

The result grid at the bottom shows a table with the following data:

MOV_TITLE
MURDER!
JURASSIC PARK
ERADU KANASU
BAYALU DAARI

3. List all actors who acted in a movie before 2000 and also in a movie after 2015 (use JOIN operation)

```

94 (SELECT ACT_ID FROM MOVIE_CAST, MOVIES
95 GROUP BY ACT_ID HAVING COUNT(ACT_ID)>1)
96 GROUP BY MOV_TITLE
97 HAVING COUNT(*)>=2;
98
99 /*3. List all actors who acted in a movie before 2000 and also in a movie after 2015 (use JOIN
100 operation).*/
101 • SELECT A.ACT_NAME, M.MOV_TITLE, M.MOV_YEAR
102 FROM ACTOR A JOIN
103 MOVIE_CAST MCT
104 ON A.ACT_ID = MCT.ACT_ID
105 JOIN MOVIES M
106 ON MCT.MOV_ID = M.MOV_ID
107 WHERE M.MOV_YEAR NOT BETWEEN 2000 AND 2015;
108
109 /*4. Find the title of movies and number of stars for each movie that has at least one rating and
110 find the highest

```

Result Grid

ACT_NAME	MOV_TITLE	MOV_YEAR
HERBERT MARSHALL	MURDER!	1930
NORAH BURING	MURDER!	1930
EDWARD CHAPMAN	MURDER!	1930
SAM NEILL	JURASSIC PARK	1993
LAURA DERN	JURASSIC PARK	1993
RAJ KUMAR	ERADU KANASU	1974
KALPANA	ERADU KANASU	1974
ANANT NAG	BAYALU DAARI	1976
KALPANA	BAYALU DAARI	1976
ANANT NAG	SARKARI HIRIYA PRATHAMIKA SHAALE KAS...	2018

4. Find the title of movies and number of stars for each movie that has at least one rating and find the highest number of stars that movie received. Sort the result by movie title.

```

101 • SELECT A.ACT_NAME, M.MOV_TITLE, M.MOV_YEAR
102 FROM ACTOR A JOIN
103 MOVIE_CAST MCT
104 ON A.ACT_ID = MCT.ACT_ID
105 JOIN MOVIES M
106 ON MCT.MOV_ID = M.MOV_ID
107 WHERE M.MOV_YEAR NOT BETWEEN 2000 AND 2015;
108
109 /*4. Find the title of movies and number of stars for each movie that has at least one rating and
110 find the highest
111 number of stars that movie received. Sort the result by movie title.*/
112 • SELECT MOV_TITLE, MAX(STARS)
113 FROM MOVIES
114 INNER JOIN RATING USING (MOV_ID)
115 GROUP BY MOV_TITLE
116 HAVING MAX(STARS)>0
117 ORDER BY MOV_TITLE;

```

Result Grid

MOV_TITLE	MAX(STARS)
BAYALU DAARI	4
ERADU KANASU	4
JURASSIC PARK	5
MURDER!	5
SARKARI HIRIYA PRATHAMIKA SHAALE KAS...	5

5. Update rating of all movies directed by 'Steven Spielberg' to 5

The screenshot shows a database management interface with a SQL editor and a result grid. The SQL editor contains the following queries:

```
111 -- number of stars that movie received. Sort the result by movie title.*/
112 • SELECT MOV_TITLE, MAX(STARS)
113 FROM MOVIES
114 INNER JOIN RATING USING (MOV_ID)
115 GROUP BY MOV_TITLE
116 HAVING MAX(STARS)>0
117 ORDER BY MOV_TITLE;
118
119 /*5. Update rating of all movies directed by 'Steven Spielberg' to 5*/
120 • UPDATE RATING
121 SET STARS = 5
122 WHERE MOV_ID IN (SELECT MOV_ID FROM MOVIES
123 WHERE DIR_ID IN (SELECT DIR_ID FROM DIRECTOR WHERE DIR_NAME = "STEVEN SPIELBERG"));
124
125 • SELECT R.STARS FROM RATING R, MOVIES M, DIRECTOR D
126 WHERE
127 D.DIR_NAME = "STEVEN SPIELBERG" AND M.DIR_ID = D.DIR_ID AND R.MOV_ID = M.MOV_ID;
```

The result grid shows the output of the third query, displaying the 'STARS' column for movies directed by Steven Spielberg. The first three rows show a rating of 5.

STARS
5
5
5

The interface includes a top toolbar with icons for file operations, a 'Limit to 1000 rows' dropdown, and a right sidebar with options for 'Result Grid', 'Form Editor', and 'Field Types'.