

**Aishwarya.M**  
**1BM20CS401**  
**CSE-4A**

### **PROGRAM 9: MOVIE DATABASE**

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

- i. List the titles of all movies directed by 'Hitchcock'.
- ii. Find the movie names where one or more actors acted in two or more movies.
- iii. List all actors who acted in a movie before 2000 and also in a movie after 2015 (use JOIN operation).
- iv. 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.
- v. Update rating of all movies directed by 'Steven Spielberg' to 5.

CREATE DATABASE MOVIE;

USE MOVIE;

```
CREATE TABLE ACTOR (  
    ACT_ID INT,  
    ACT_NAME VARCHAR (20),  
    ACT_GENDER CHAR (1),  
    PRIMARY KEY (ACT_ID));
```

```
CREATE TABLE DIRECTOR (  
    DIR_ID INT,  
    DIR_NAME VARCHAR (20),  
    DIR_PHONE LONG,
```

PRIMARY KEY (DIR\_ID));

```
CREATE TABLE MOVIES (  
  MOV_ID INT,  
  MOV_TITLE VARCHAR (25),  
  MOV_YEAR INT,  
  MOV_LANG VARCHAR (12),  
  DIR_ID INT,  
  PRIMARY KEY (MOV_ID),  
  FOREIGN KEY (DIR_ID) REFERENCES DIRECTOR (DIR_ID));
```

```
CREATE TABLE MOVIE_CAST (  
  ACT_ID INT,  
  MOV_ID INT,  
  AROLE VARCHAR(10),  
  PRIMARY KEY (ACT_ID, MOV_ID),  
  FOREIGN KEY(ACT_ID) REFERENCES ACTOR(ACT_ID) ON DELETE CASCADE,  
  FOREIGN KEY(MOV_ID) REFERENCES MOVIES(MOV_ID) ON DELETE  
  CASCADE);
```

```
CREATE TABLE RATING (  
  MOV_ID INT,  
  REV_STARS VARCHAR (25),  
  PRIMARY KEY (MOV_ID),  
  FOREIGN KEY (MOV_ID) REFERENCES MOVIES (MOV_ID));
```

```
INSERT INTO ACTOR VALUES (301,'ANUSHKA','F');  
INSERT INTO ACTOR VALUES (302,'PRABHAS','M');  
INSERT INTO ACTOR VALUES (303,'PUNITH','M');  
INSERT INTO ACTOR VALUES (304,'JERMY','M');
```

```

INSERT INTO DIRECTOR VALUES (60,'RAJAMOULI', 8751611001);
INSERT INTO DIRECTOR VALUES (61,'HITCHCOCK', 7766138911);
INSERT INTO DIRECTOR VALUES (62,'FARAN', 9986776531);
INSERT INTO DIRECTOR VALUES (63,'STEVEN SPIELBERG', 8989776530);

```

```

INSERT INTO MOVIES VALUES (1001,'BAHUBALI-2', 2017,'TELAGU', 60);
INSERT INTO MOVIES VALUES (1002,'BAHUBALI-1', 2015, 'TELAGU', 60);
INSERT INTO MOVIES VALUES (1003,'AKASH', 2008, 'KANNADA', 61);
INSERT INTO MOVIES VALUES (1004,'WAR HORSE', 2011, 'ENGLISH', 63);

```

```

INSERT INTO MOVIE_CAST VALUES (301, 1002, 'HEROINE');
INSERT INTO MOVIE_CAST VALUES (301, 1001, 'HEROINE');
INSERT INTO MOVIE_CAST VALUES (303, 1003, 'HERO');
INSERT INTO MOVIE_CAST VALUES (303, 1002, 'GUEST');
INSERT INTO MOVIE_CAST VALUES (304, 1004, 'HERO');

```

```

INSERT INTO RATING VALUES (1001, 4);
INSERT INTO RATING VALUES (1002, 2);
INSERT INTO RATING VALUES (1003, 5);
INSERT INTO RATING VALUES (1004, 4);

```

```

select * from rating;

```

Result Grid		
Filter Rows:		
	MOV_ID	REV_STARS
▶	1001	4
	1002	2
	1003	5
	1004	4
*	NULL	NULL

rating 1 x

select \* from actor;

Result Grid			
Filter Rows: <input type="text"/>			
Edit: <input type="text"/>			
	ACT_ID	ACT_NAME	ACT_GENDER
▶	301	ANUSHKA	F
	302	PRABHAS	M
	303	PUNITH	M
	304	JERMY	M
*	NULL	NULL	NULL

actor 2 x

select \* from director;

Result Grid			
Filter Rows: <input type="text"/>			
Edit: <input type="text"/>			
	DIR_ID	DIR_NAME	DIR_PHONE
▶	60	RAJAMOULI	8751611001
	61	HITCHCOCK	7766138911
	62	FARAN	9986776531
	63	STEVEN SPIELBERG	8989776530
*	NULL	NULL	NULL

director 3 x

select \* from movies;

Result Grid					
Filter Rows: <input type="text"/>					
Edit: <input type="text"/>					
	MOV_ID	MOV_TITLE	MOV_YEAR	MOV_LANG	DIR_ID
▶	1001	BAHUBALI-2	2017	TELAGU	60
	1002	BAHUBALI-1	2015	TELAGU	60
	1003	AKASH	2008	KANNADA	61
	1004	WAR HORSE	2011	ENGLISH	63
*	NULL	NULL	NULL	NULL	NULL

movies 4 x

select \* from movie\_cast;

Result Grid			
Filter Rows:			
	ACT_ID	MOV_ID	AROLE
▶	301	1001	HEROINE
	301	1002	HEROINE
	303	1002	GUEST
	303	1003	HERO
	304	1004	HERO
✱	NULL	NULL	NULL

movie\_cast 5 ×

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

```
SELECT MOV_TITLE
FROM MOVIES
WHERE DIR_ID IN (SELECT DIR_ID
FROM DIRECTOR
WHERE DIR_NAME = 'HITCHCOCK');
```

Result Grid	
Filter Rows:	
	MOV_TITLE
▶	AKASH

MOVIES 6 ×

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

```
SELECT MOV_TITLE
FROM MOVIES M, MOVIE_CAST MV
WHERE M.MOV_ID=MV.MOV_ID AND ACT_ID IN (SELECT ACT_ID
FROM MOVIE_CAST GROUP BY ACT_ID
HAVING COUNT(ACT_ID)>1)
GROUP BY MOV_TITLE
HAVING COUNT(*)>1;
```

Result Grid		Filter Rows:
	MOV_TITLE	
▶	BAHUBALI-1	

Result 7 ×

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

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

Result Grid		Filter Rows:	Export:
	ACT_NAME	MOV_TITLE	MOV_YEAR
▶	ANUSHKA	BAHUBALI-2	2017

Result 8 ×

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.

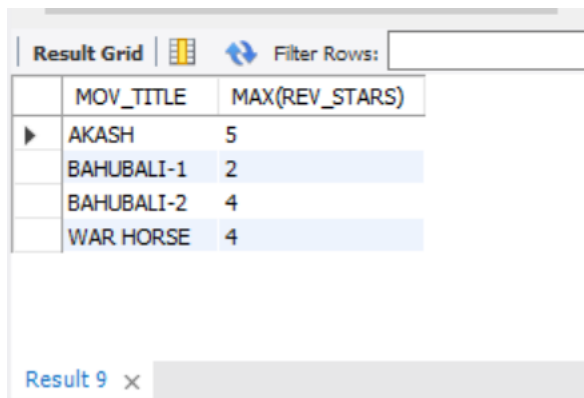
Sort the result by movie title.

```
SELECT MOV_TITLE, MAX(REV_STARS)
FROM MOVIES
INNER JOIN RATING USING (MOV_ID)
```

GROUP BY MOV\_TITLE

HAVING MAX(REV\_STARS)>0

ORDER BY MOV\_TITLE;



Result Grid | Filter Rows:

	MOV_TITLE	MAX(REV_STARS)
▶	AKASH	5
	BAHUBALI-1	2
	BAHUBALI-2	4
	WAR HORSE	4

Result 9 x

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

UPDATE RATING

SET REV\_STARS=5

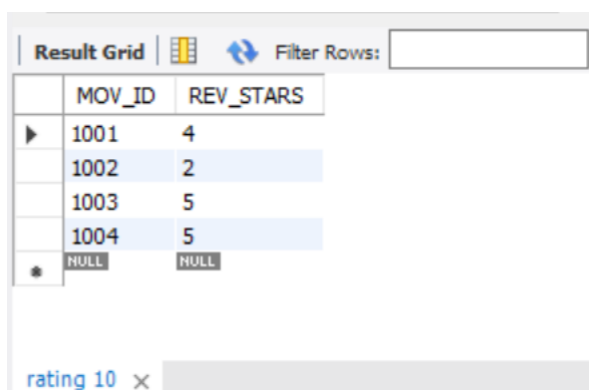
WHERE MOV\_ID IN(SELECT MOV\_ID FROM MOVIES

WHERE DIR\_ID IN(SELECT DIR\_ID

FROM DIRECTOR

WHERE DIR\_NAME = 'STEVEN SPIELBERG'));

select \* from rating;



Result Grid | Filter Rows:

	MOV_ID	REV_STARS
▶	1001	4
	1002	2
	1003	5
	1004	5
*	NULL	NULL

rating 10 x