

EE4202 - Database Systems

Mini Project – Semester 4: 09 April 2024

GROUP NUMBER – 87

Table of Contents

<u>1.</u>	<u>CHAPTER 1 - REQUIREMENT ANALYSIS</u>	<u>3</u>
<u>I.</u>	<u>FUNCTIONAL REQUIREMENTS</u>	<u>3</u>
<u>II.</u>	<u>DATA REQUIREMENTS.....</u>	<u>4</u>
<u>2.</u>	<u>CHAPTER 2 - CONCEPTUAL DESIGN.....</u>	<u>5</u>
<u>3.</u>	<u>CHAPTER 3 – IMPLEMENTATION</u>	<u>8</u>
<u>4.</u>	<u>CHAPTER 4 - TRANSACTIONS.....</u>	<u>31</u>
<u>I.</u>	<u>SIMPLE QUERIES</u>	<u>31</u>
<u>II.</u>	<u>COMPLEX QUERIES.....</u>	<u>38</u>
<u>III.</u>	<u>COMPARISON OF COMPLEXITY OF THE QUERIES BEFORE AND AFTER INSERTING THE INDEX 51</u>	
<u>IV.</u>	<u>ERROR AND SOLUTION.....</u>	<u>61</u>

1. Chapter 1 - Requirement Analysis

i. Functional Requirements

i. Manage Movies and TV Series

The system should allow users to add, update, and delete information about movies and TV series.

ii. Track Director Information

Users should be able to associate directors with movies and TV series and track their details such as name, age, nationality, and awards.

iii. Studio and Website Association

The system should facilitate linking studios and websites with TV series for streaming.

iv. Episode Management

Users should be able to manage episodes of TV series, including their titles, ratings, length, and cost per episode.

v. Actor Details

The system should store information about main actors, including their names, nationality, and awards.

vi. Song Information

Users should be able to track songs featured in movies, including details about singers, writers, and awards.

vii. Review Management

The system should allow users to add reviews for both movies and TV series, including comments, ratings, and review dates.

ii. Data Requirements

i. Movie Data

Information about movies such as ID, name, release year, genre, duration, language, and budget.

ii. Director Data

Details about directors including license number, name, age, active period, country, nationality, awards, and associations with movies/TV series.

iii. TV Series Data

Data related to TV series such as ID, name, release year, genre, language, number of seasons, and country of origin.

iv. Country Data

Information about countries including name, rank, and region.

v. Episode Data

Details about episodes including title, ratings, length, cost per episode, and association with TV series.

vi. Studio Data

Information about studios such as license number, established date, revenue, country, and association with TV series.

vii. Website Data

Data related to websites for streaming including name, license number, and association with TV series.

viii. Main Actor Data

Information about main actors including license number, name, nationality, date of birth, active period, awards, and associations with movies.

ix. Song Data

Details about songs featured in movies including license ID, name, singer, writer, language, awards, and associations with movies.

x. Review Data

Data related to reviews including comments, ratings, review date, user experience, and associations with movies/TV series.

2. Chapter 2 - Conceptual Design

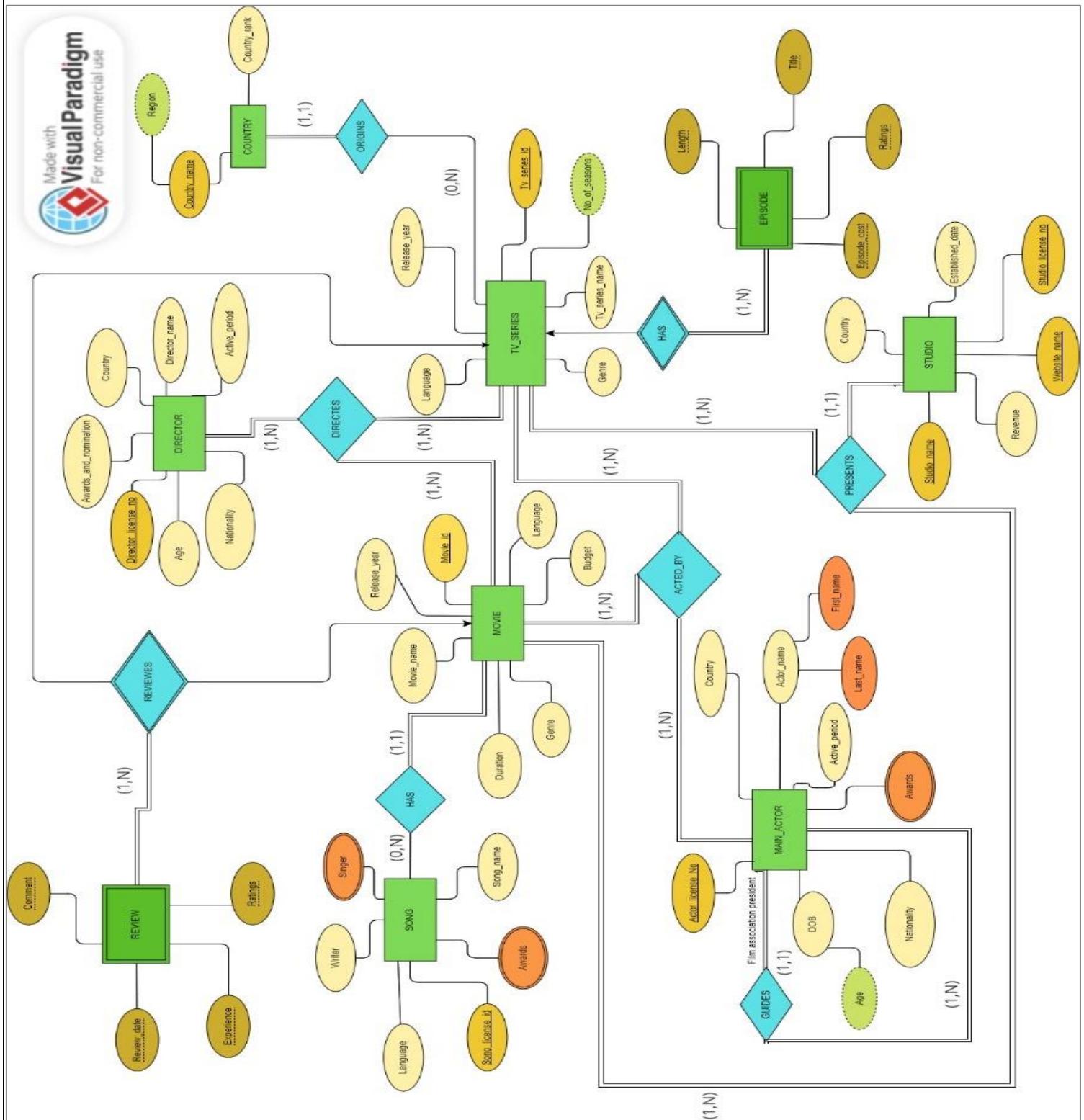


FIGURE 1 : ER DIAGRAM

1). Recursive relationship

- GUIDES

2). Multivalued attribute

- Awards in MAIN ACTOR relation
- Awards in SONG relation
- Singer in SONG relation

3). Composite attribute

- Actor name in MAIN ACTOR relation

4). Derived attribute

- Age in the MAIN ACTOR

5). Multiple candidate key

- In the relation STUDIO

6). Strong entities

- DIRECTOR
- COUNTRY
- TV_SERIES
- MOVIE
- SONG
- STUDIO

7). Derived attribute

- Age in the MAIN ACTOR

Class Diagram1

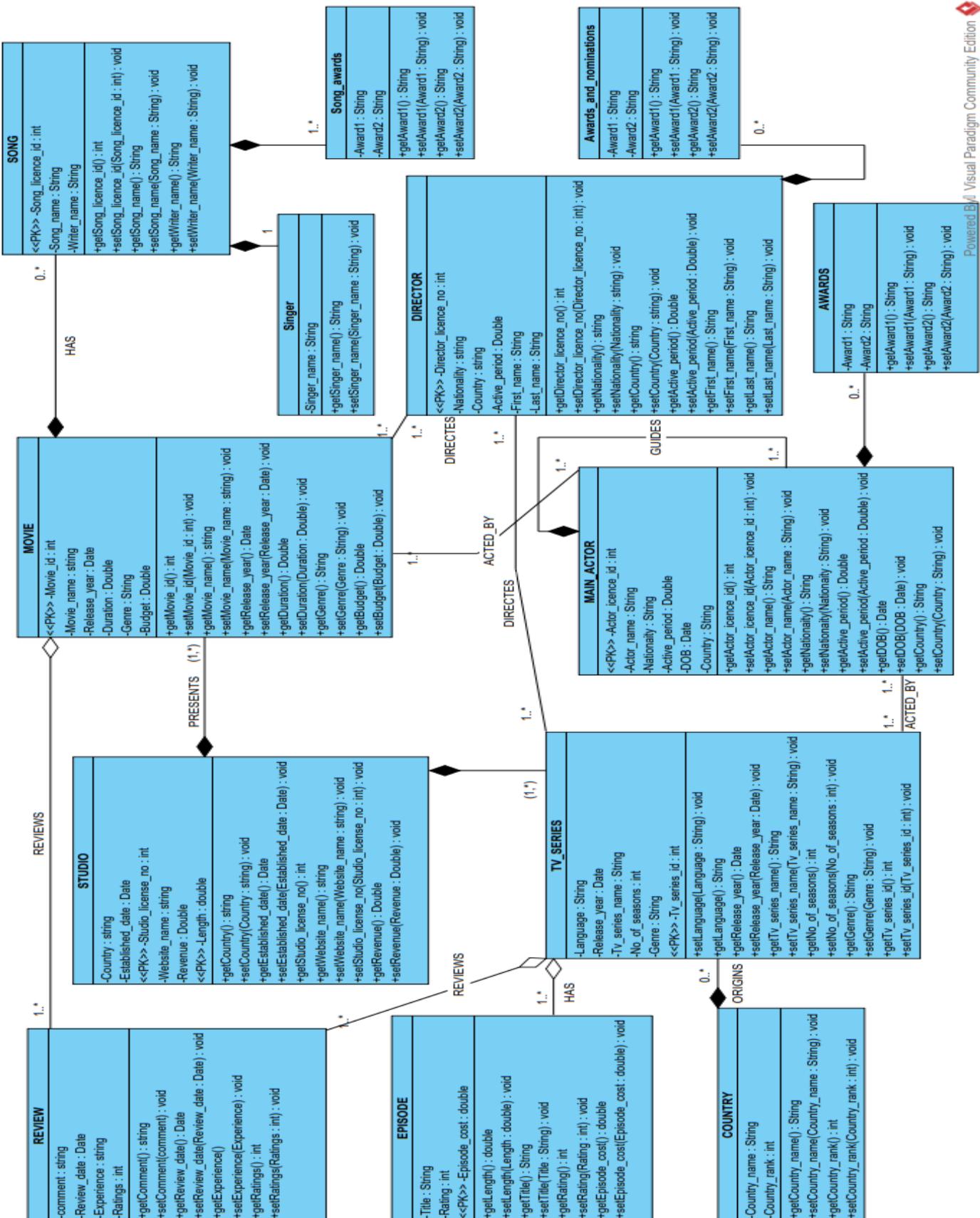


FIGURE 2 : UML DIAGRAM

3. Chapter 3 – Implementation

3.1 Creation of the schema

The screenshot shows the MySQL Workbench interface. In the top navigation bar, there are tabs for 'unconnected' and 'Local instance MySQL80 (foe_23)'. The main area displays a SQL editor window titled 'EE4202_Final' containing the following SQL code:

```
1 • CREATE DATABASE TVSEREIS_MOVIES_COLLECTION1 ;
2 • use TVSEREIS_MOVIES_COLLECTION1 ;
3
4
5 • CREATE TABLE MOVIE(
6     MOVIE_ID varchar (10) NOT NULL,
7     MOVIE_NAME varchar (50) NOT NULL,
8     RELEASED_YEAR int (4) NOT NULL,
9     GENRE varchar (20) NOT NULL,
10    DURATION time not null, M_LANGUAGE varchar (20), BUDGET double (30,5),
11    constraint pk1 primary key(MOVIE_ID)
12 );
13 • CREATE TABLE DIRECTOR (
14     LICENSE_NO varchar (10) NOT NULL,
15     DIRECTOR_NAME varchar (50) NOT NULL,
16     AGE INT (3),
17     ACTIVE_PERIOD_IN_YEARS INT (5),
18     COUNTRY varchar (20) NOT NULL,
19     NATIONALITY varchar (15) not null, AWARDS_AND_NOMINATIONS varchar (50),
20     movie_id varchar(10), tv_series_id varchar (10) ,
21     constraint pk1 primary key(LICENSE_NO)
22 );
```

The 'Output' pane shows the results of the execution:

#	Time	Action	Message	Duration / Fetch
1	00:27:04	CREATE DATABASE TVSEREIS_MOVIES_COLLECTION1	1 row(s) affected	0.016 sec
2	00:27:04	use TVSEREIS_MOVIES_COLLECTION1	0 row(s) affected	0.000 sec

The status bar at the bottom right indicates the date and time: 4/3/2024 12:40 AM.

FIGURE 3 : CREATION OF SCHEMA

3.2 Table Definitions

The screenshot shows the MySQL Workbench interface. In the top navigation bar, there are tabs for 'unconnected' and 'Local instance MySQL80 (foe_23)'. The main area displays a SQL editor window titled 'EE4202_Final' containing the same SQL code as Figure 3, but with additional table definitions:

```
1 • CREATE DATABASE TVSEREIS_MOVIES_COLLECTION1 ;
2 • use TVSEREIS_MOVIES_COLLECTION1 ;
3
4
5 • CREATE TABLE MOVIE(
6     MOVIE_ID varchar (10) NOT NULL,
7     MOVIE_NAME varchar (50) NOT NULL,
8     RELEASED_YEAR int (4) NOT NULL,
9     GENRE varchar (20) NOT NULL,
10    DURATION time not null, M_LANGUAGE varchar (20), BUDGET double (30,5),
11    constraint pk1 primary key(MOVIE_ID)
12 );
13 • CREATE TABLE DIRECTOR (
14     LICENSE_NO varchar (10) NOT NULL,
15     DIRECTOR_NAME varchar (50) NOT NULL,
16     AGE INT (3),
17     ACTIVE_PERIOD_IN_YEARS INT (5),
18     COUNTRY varchar (20) NOT NULL,
19     NATIONALITY varchar (15) not null, AWARDS_AND_NOMINATIONS varchar (50),
20     movie_id varchar(10), tv_series_id varchar (10) ,
21     constraint pk1 primary key(LICENSE_NO)
22 );
```

The status bar at the bottom right indicates the date and time: 4/3/2024 12:40 AM.

FIGURE 4 : CREATING MOVIE AND DIRECTOR TABLES

The screenshot shows the MySQL Workbench interface with a query editor window titled "EE4202_Final". The code being run creates two tables: TV_SERIES and COUNTRY. The TV_SERIES table has columns for NATIONALITY (VARCHAR(15) NOT NULL), AWARDS_AND_NOMINATIONS (VARCHAR(50)), movie_id (VARCHAR(10)), tv_series_id (VARCHAR(10)), and a primary key constraint pk1 on the LICENSE_NO column. The COUNTRY table has columns for COUNTRY_NAME (VARCHAR(20) NOT NULL), COUNTRY_RANK (INT(6)), REGION (VARCHAR(20)), and a primary key constraint pk1 on the COUNTRY_NAME column.

```

19 NATIONALITY varchar (15) not null, AWARDS_AND_NOMINATIONS varchar (50),
20 movie_id varchar(10), tv_series_id varchar (10) ,
21 constraint pk1 primary key(LICENSE_NO)
22 );
23
24 • CREATE TABLE TV_SERIES(
25   TV_SERIES_ID varchar (10) NOT NULL,
26   TV_SERIES_NAME varchar (50) NOT NULL,
27   RELEASED_YEAR int (4) NOT NULL,
28   GENRE varchar (20) NOT NULL,
29   TS_LANGUAGE varchar (20),
30   NO_OF_SEASONS INT (6),
31   country_name varchar (20),
32   constraint pk1 primary key (TV_SERIES_ID)
33 );
34
35 • CREATE TABLE COUNTRY(
36   COUNTRY_NAME varchar (20) Not null,
37   COUNTRY_RANK int(6),
38   REGION varchar (20),
39   constraint pk1 Primary key (COUNTRY_NAME)
40 );

```

FIGURE 6 : CREATING TV SERIES AND COUNTRY TABLES

The screenshot shows the MySQL Workbench interface with a query editor window titled "EE4202_Final". The code being run creates two tables: EPISODE and STUDIO. The EPISODE table has columns for TITLE (VARCHAR(150) NOT NULL), RATINGS (DOUBLE(3,2)), EPISODE_LENGTH (TIME), tv_series_id (VARCHAR(50) NOT NULL), and COST_PER_EPISODE (DOUBLE(30,5)), with a primary key constraint pk1 on the combination of tv_series_id and title. The STUDIO table has columns for LICENSE_NO (VARCHAR(10) NOT NULL), ESTABLISHED_DATE (DATE), REVENUE (DOUBLE(60,3)), COUNTRY (VARCHAR(15)), tv_series_id (VARCHAR(20)), and a primary key constraint pk1 on the LICENSE_NO column.

```

39 constraint pk1 Primary key (COUNTRY_NAME)
40 );
41
42 • CREATE TABLE EPISODE(
43   TITLE varchar (150) Not null,
44   RATINGS double(3,2),
45   EPISODE_LENGTH time,
46   tv_series_id varchar (50) Not null,
47   COST_PER_EPISODE double(30,5),
48   constraint pk1 Primary key (tv_series_id, title)
49 );
50
51 • CREATE TABLE STUDIO(
52   LICENSE_NO varchar (10) NOT NULL,
53   ESTABLISHED_DATE date,
54   REVENUE double (60,3),
55   COUNTRY varchar(15),
56   tv_series_id varchar(20),
57   constraint pk1 Primary key (LICENSE_NO)
58 );
59
60

```

FIGURE 5 : CREATING EPISODE AND STUDIO TABLES

The screenshot shows the MySQL Workbench interface with the following details:

- File Bar:** File, Edit, View, Query, Database, Server, Tools, Scripting, Help.
- Navigator:** Schemas (foe_23 selected), Tables, Views, Stored Procedures, Functions, sys, tv_sereis_movies_collection.
- Editor:** EE4202_Final tab, SQL pane. The code is as follows:

```

59
60 • CREATE TABLE MAIN_ACTOR(
61     ACTOR_LICENSE_NO INT(10) NOT NULL,
62     FIRST_NAME varchar(15),
63     LAST_NAME varchar(15),
64     NATIONALITY varchar(15),
65     COUNTRY varchar(15),
66     DATE_OF_BIRTH date,
67     ACTIVE_PERIOD int(10),
68     AWARDS varchar(40),
69     movie_id varchar(10),
70     president INT(10),
71     constraint pk1 primary key(ACTOR_LICENSE_NO)
72 );
73
74 • CREATE TABLE NAME_OF_THE_STUDIO(
75     studio_name varchar(30),
76     license_no varchar(10) Not null,
77     tv_series_id varchar(20),
78     constraint pk1 Primary key (license_no)
79 );
80

```

- SQLAdditions:** Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.
- Bottom:** Object Info, Session, Context Help, Snippets, Query Completed, Taskbar, System tray.

FIGURE 8 : CREATING MAIN ACTOR AND NAME OF THE STUDIO TABLES

The screenshot shows the MySQL Workbench interface with the following details:

- File Bar:** File, Edit, View, Query, Database, Server, Tools, Scripting, Help.
- Navigator:** Schemas (foe_23 selected), Tables, Views, Stored Procedures, Functions, sys, tv_sereis_movies_collection.
- Editor:** EE4202_Final tab, SQL pane. The code is as follows:

```

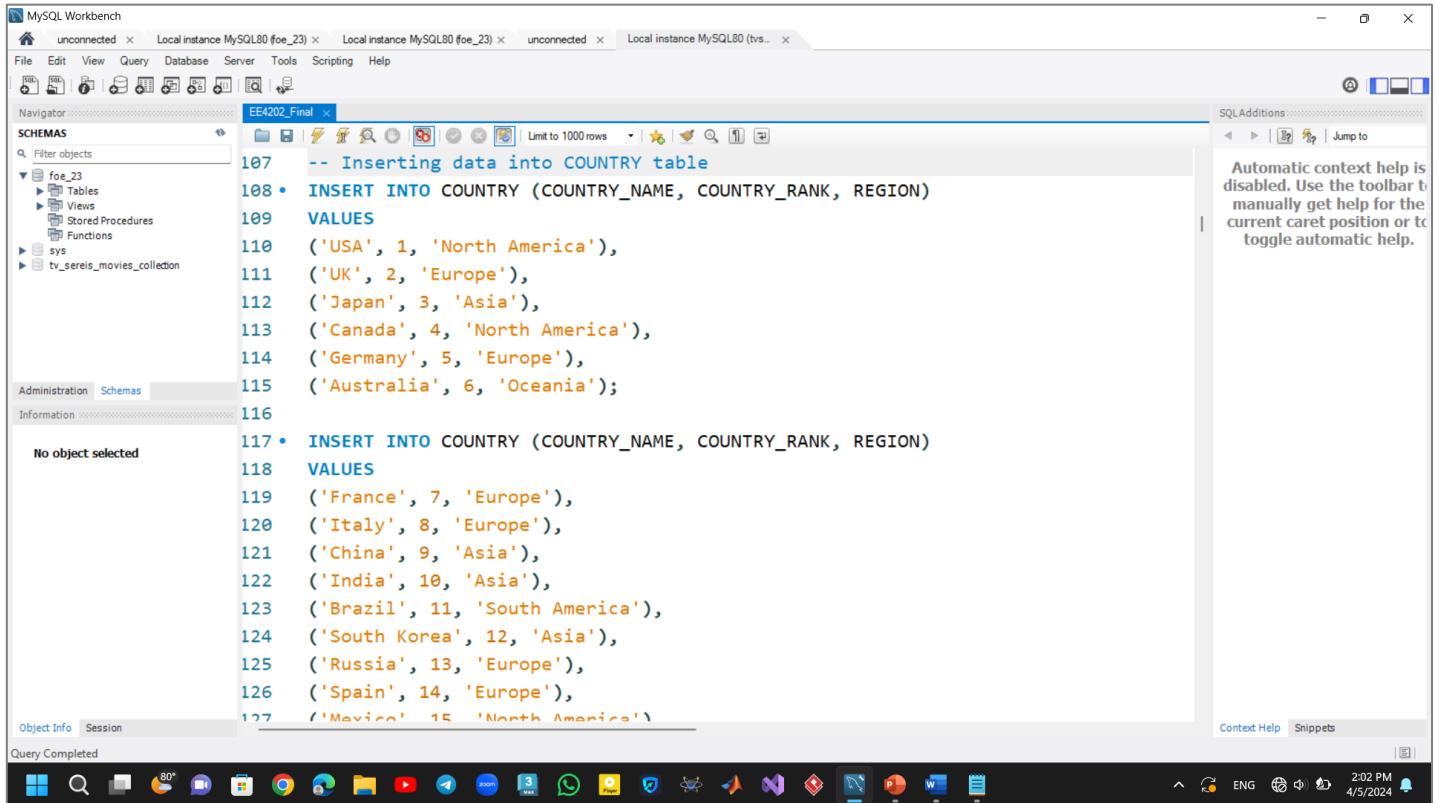
81 • CREATE TABLE NAME_OF_THE_WEBSITE(
82     website_name varchar(30),
83     license_no varchar(10) Not null,
84     tv_series_id varchar(20),
85     constraint pk1 Primary key (license_no)
86 );
87
88 • CREATE TABLE SONG(
89     LICENSE_ID varchar(10) NOT NULL,
90     SONG_NAME varchar(200),
91     SINGER varchar(200),
92     WRITER varchar(200),
93     S_LANGUAGE varchar(200), AWARDS varchar(200),
94     movie_id varchar(10),
95     constraint pk1 primary key(LICENSE_ID)
96 );
97
98 • CREATE TABLE REVIEW(
99     COMMENTS varchar(200),
100    RATING double(10,3) Not null,
101    REVIEW_DATE date, EXPERIENCE varchar(50),
102    movie_id varchar(10),
103    tv_series_id varchar(10) Not null,
104    constraint pk1 primary key(tv_series_id,RATING)
105 );
106

```

- SQLAdditions:** Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.
- Bottom:** Object Info, Session, Context Help, Snippets, Query Completed, Taskbar, System tray.

FIGURE 7 : CREATING NAME OF THE WEBSITE, SONGS, REVIEW TABLES

3.3 Data Insertion



The screenshot shows the MySQL Workbench interface with a query editor window titled 'EE4202_Final'. The code is for inserting data into the 'COUNTRY' table:

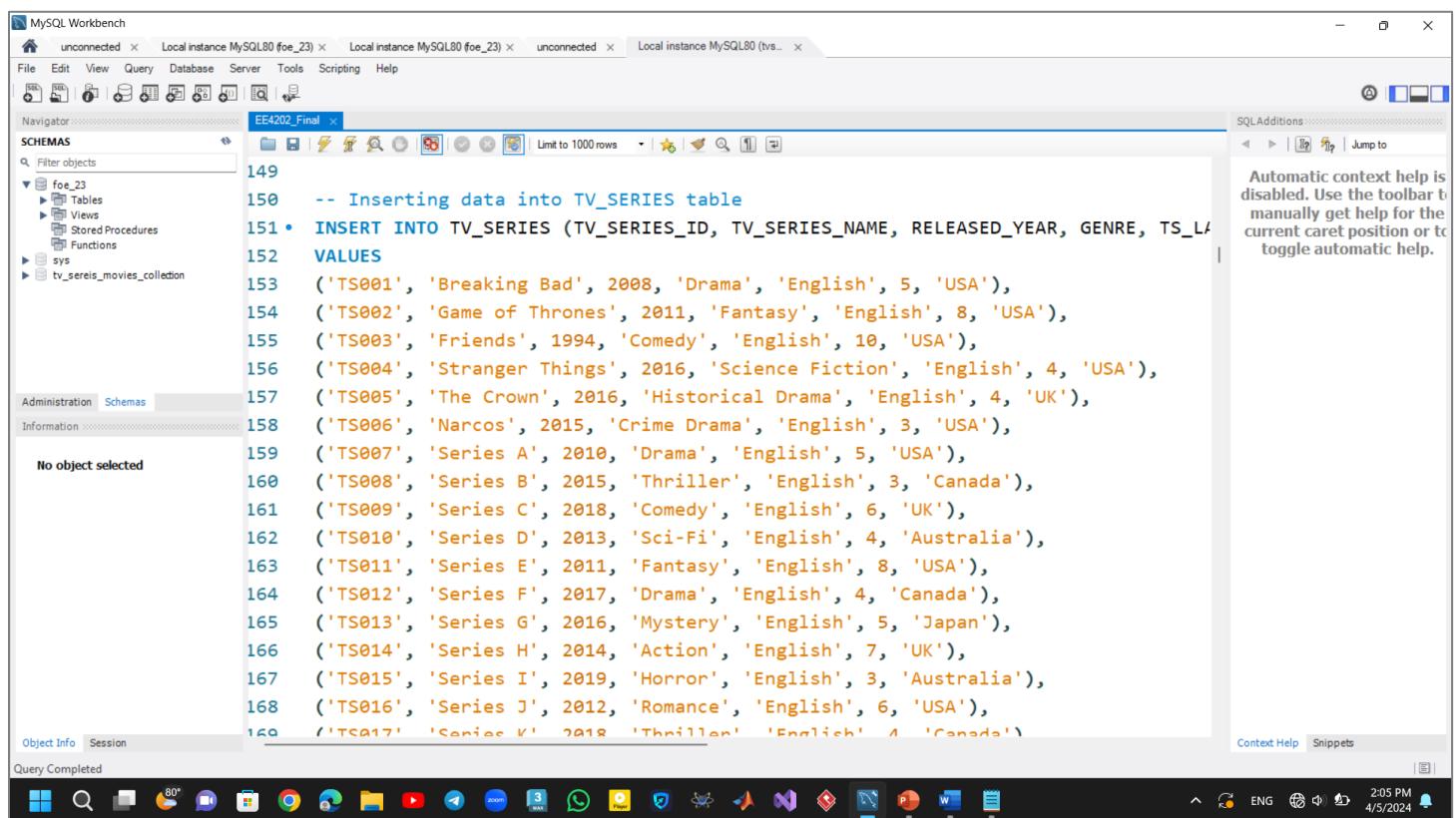
```

107 -- Inserting data into COUNTRY table
108 • INSERT INTO COUNTRY (COUNTRY_NAME, COUNTRY_RANK, REGION)
109   VALUES
110   ('USA', 1, 'North America'),
111   ('UK', 2, 'Europe'),
112   ('Japan', 3, 'Asia'),
113   ('Canada', 4, 'North America'),
114   ('Germany', 5, 'Europe'),
115   ('Australia', 6, 'Oceania');
116
117 • INSERT INTO COUNTRY (COUNTRY_NAME, COUNTRY_RANK, REGION)
118   VALUES
119   ('France', 7, 'Europe'),
120   ('Italy', 8, 'Europe'),
121   ('China', 9, 'Asia'),
122   ('India', 10, 'Asia'),
123   ('Brazil', 11, 'South America'),
124   ('South Korea', 12, 'Asia'),
125   ('Russia', 13, 'Europe'),
126   ('Spain', 14, 'Europe'),
127   ('Mexico', 15, 'North America')

```

The code consists of two parts, each starting with a comment line and followed by an 'INSERT INTO' statement. The first part inserts 6 rows, and the second part inserts 15 more rows, totaling 21 rows inserted into the 'COUNTRY' table.

FIGURE 10 : INSERTING TUPLES INTO COUNTRY TABLE



The screenshot shows the MySQL Workbench interface with a query editor window titled 'EE4202_Final'. The code is for inserting data into the 'TV_SERIES' table:

```

149
150 -- Inserting data into TV_SERIES table
151 • INSERT INTO TV_SERIES (TV_SERIES_ID, TV_SERIES_NAME, RELEASED_YEAR, GENRE, TS_LA)
152   VALUES
153   ('TS001', 'Breaking Bad', 2008, 'Drama', 'English', 5, 'USA'),
154   ('TS002', 'Game of Thrones', 2011, 'Fantasy', 'English', 8, 'USA'),
155   ('TS003', 'Friends', 1994, 'Comedy', 'English', 10, 'USA'),
156   ('TS004', 'Stranger Things', 2016, 'Science Fiction', 'English', 4, 'USA'),
157   ('TS005', 'The Crown', 2016, 'Historical Drama', 'English', 4, 'UK'),
158   ('TS006', 'Narcos', 2015, 'Crime Drama', 'English', 3, 'USA'),
159   ('TS007', 'Series A', 2010, 'Drama', 'English', 5, 'USA'),
160   ('TS008', 'Series B', 2015, 'Thriller', 'English', 3, 'Canada'),
161   ('TS009', 'Series C', 2018, 'Comedy', 'English', 6, 'UK'),
162   ('TS010', 'Series D', 2013, 'Sci-Fi', 'English', 4, 'Australia'),
163   ('TS011', 'Series E', 2011, 'Fantasy', 'English', 8, 'USA'),
164   ('TS012', 'Series F', 2017, 'Drama', 'English', 4, 'Canada'),
165   ('TS013', 'Series G', 2016, 'Mystery', 'English', 5, 'Japan'),
166   ('TS014', 'Series H', 2014, 'Action', 'English', 7, 'UK'),
167   ('TS015', 'Series I', 2019, 'Horror', 'English', 3, 'Australia'),
168   ('TS016', 'Series J', 2012, 'Romance', 'English', 6, 'USA'),
169   ('TS017', 'Series K', 2018, 'Thriller', 'English', 4, 'Canada')

```

The code consists of a single 'INSERT INTO' statement with 17 values, inserting data into the 'TV_SERIES' table.

FIGURE 9 : INSERT TUPLES INTO TV SERIES TABLE

```

179
180
181 -- Inserting data into MOVIE table
182 • INSERT INTO MOVIE (MOVIE_ID, MOVIE_NAME, RELEASED_YEAR, GENRE, DURATION, M_LANG)
183     VALUES
184     ('MV001', 'The Shawshank Redemption', 1994, 'Drama', '02:22:00', 'English', 250)
185     ('MV002', 'The Godfather', 1972, 'Crime', '02:55:00', 'English', 6000000),
186     ('MV003', 'The Dark Knight', 2008, 'Action', '02:32:00', 'English', 185000000),
187     ('MV004', 'Pulp Fiction', 1994, 'Crime', '02:34:00', 'English', 8000000),
188     ('MV005', 'Forrest Gump', 1994, 'Drama', '02:22:00', 'English', 5500000),
189     ('MV006', 'Inception', 2010, 'Science Fiction', '02:28:00', 'English', 160000000),
190     ('MV007', 'Schindler''s List', 1993, 'Biography', '03:15:00', 'English', 2200000),
191     ('MV008', 'The Lord of the Rings: The Return of the King', 2003, 'Adventure', '03:25:00', 'English', 300000000),
192     ('MV009', 'The Matrix', 1999, 'Action', '02:16:00', 'English', 63000000),
193     ('MV010', 'The Silence of the Lambs', 1991, 'Thriller', '01:58:00', 'English', 12000000),
194     ('MV011', 'The Green Mile', 1999, 'Crime', '03:09:00', 'English', 60000000),
195     ('MV012', 'Gladiator', 2000, 'Action', '02:35:00', 'English', 103000000),
196     ('MV013', 'Goodfellas', 1990, 'Biography', '02:26:00', 'English', 25000000),
197     ('MV014', 'The Departed', 2006, 'Crime', '02:31:00', 'English', 90000000),
198     ('MV015', 'The Pianist', 2002, 'Biography', '02:30:00', 'English', 35000000),
199     ('MV016', 'The Shawshank Redemption', 1994, 'Drama', '02:22:00', 'English', 250)

```

FIGURE 12 : INSERTING TUPLES INTO MOVIE TABLE

```

221
222 -- Inserting data into DIRECTOR table
223 • INSERT INTO DIRECTOR (LICENSE_NO, DIRECTOR_NAME, AGE, ACTIVE_PERIOD_IN_YEARS, COUNTRY)
224     VALUES
225     ('LIC001', 'Frank Darabont', 62, 25, 'USA', 'American', '2 Oscars, 5 Golden Globes'),
226     ('LIC002', 'Francis Ford Coppola', 83, 50, 'USA', 'American', '5 Oscars, 3 Golden Globes'),
227     ('LIC003', 'Christopher Nolan', 52, 25, 'UK', 'British', '1 Oscar, 4 Golden Globes'),
228     ('LIC004', 'Quentin Tarantino', 58, 30, 'USA', 'American', '2 Oscars, 3 Golden Globes'),
229     ('LIC005', 'Robert Zemeckis', 70, 40, 'USA', 'American', '1 Oscar, 3 Golden Globes'),
230     ('LIC006', 'Christopher Nolan', 52, 25, 'UK', 'British', '1 Oscar, 4 Golden Globes'),
231     ('LIC007', 'Steven Spielberg', 75, 50, 'USA', 'American', '3 Oscars, 7 Golden Globes'),
232     ('LIC008', 'Martin Scorsese', 79, 55, 'USA', 'American', '1 Oscar, 3 Golden Globes'),
233     ('LIC009', 'James Cameron', 67, 40, 'Canada', 'Canadian', '3 Oscars, 2 Golden Globes'),
234     ('LIC010', 'Alfred Hitchcock', 82, 50, 'UK', 'British', '0 Oscars, 0 Golden Globes'),
235     ('LIC012', 'David Fincher', 59, 30, 'USA', 'American', '0 Oscars, 2 Golden Globes'),
236     ('LIC013', 'Tim Burton', 63, 35, 'USA', 'American', '0 Oscars, 0 Golden Globes'),
237     ('LIC014', 'Ridley Scott', 84, 50, 'UK', 'British', '0 Oscars, 2 Golden Globes'),
238     ('LIC015', 'Clint Eastwood', 92, 60, 'USA', 'American', '4 Oscars, 4 Golden Globes'),
239     ('LIC016', 'Steven Soderbergh', 59, 30, 'USA', 'American', '2 Oscars, 2 Golden Globes'),
240     ('LIC017', 'Woody Allen', 86, 55, 'USA', 'American', '4 Oscars, 6 Golden Globes'),
241     ('LIC018', 'George Lucas', 78, 45, 'USA', 'American', '0 Oscars, 0 Golden Globes')

```

FIGURE 11: INSERTING TUPLES INTO DIRECTOR TABLE

The screenshot shows the MySQL Workbench interface with two tabs open in the query editor: 'EE4202_Final' and 'EE4202_Final'. The 'EE4202_Final' tab contains the following SQL code:

```
260
261 -- Inserting data into EPISODE table
262 • INSERT INTO EPISODE (TITLE, RATINGS, EPISODE_LENGTH, tv_series_id, COST_PER_EPISODE)
263 VALUES
264 ('Pilot', 8.7, '00:58:00', 'TS001', 3000000),
265 ('Winter Is Coming', 9.1, '00:58:00', 'TS002', 8000000),
266 ('The One Roommate', 8.3, '00:22:00', 'TS003', 500000),
267 ('Chapter of Byers', 8.8, '00:48:00', 'TS004', 6000000),
268 ('Wolferton Splash', 8.7, '00:58:00', 'TS005', 10000000),
269 ('Descenso', 8.8, '00:57:00', 'TS006', 5000000);
270
271 -- Inserting data into STUDIO table
272 • INSERT INTO STUDIO (LICENSE_NO, ESTABLISHED_DATE, REVENUE, COUNTRY, tv_series_id)
273 VALUES
274 ('ST001', '1986-01-15', 25000000, 'USA', 'TS001'),
275 ('ST002', '1977-05-25', 35000000, 'USA', 'TS002'),
276 ('ST003', '1939-06-20', 18000000, 'USA', 'TS003'),
277 ('ST004', '2006-07-18', 40000000, 'USA', 'TS004'),
278 ('ST005', '2010-01-01', 20000000, 'UK', 'TS005'),
279 ('ST006', '1998-01-01', 30000000, 'USA', 'TS006');
280
```

The 'Navigator' pane on the left shows the database schema with tables like 'foe_23', 'sys', and 'tv_sereis_movies_collection'. The 'Information' pane below the Navigator shows 'No object selected'. The status bar at the bottom indicates 'Query Completed'.

FIGURE 14 : INSERTING TUPLES INTO EPISODE AND STUDIO TABLES

The screenshot shows the MySQL Workbench interface with the following details:

- MySQL Workbench** window title.
- File Edit View Query Database Server Tools Scripting Help** menu bar.
- Navigator** pane on the left showing **SCHEMAS** (foe_23, sys, tv_sereis_movies_collection) and **Administration Schemas** tabs. The **No object selected** message is displayed.
- EE4202_Final** query editor tab.
- SQLAdditions** toolbar on the right with icons for help, search, and jump.
- Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.** status message.
- Query Editor Content:**

```
280 |
281 -- Inserting data into MAIN_ACTOR table
282 • INSERT INTO MAIN_ACTOR (ACTOR_LICENSE_NO, FIRST_NAME, LAST_NAME, NATIONALITTY, COUNTRY,
283 VALUES
284 ('1001', 'Morgan', 'Freeman', 'American', 'USA', '1937-06-01', 50, '1 Oscar, 5 Golden Gl
285 ('1002', 'Marlon', 'Brando', 'American', 'USA', '1924-04-03', 30, '2 Oscars, 2 Golden Gl
286 ('1003', 'Christian', 'Bale', 'British', 'UK', '1974-01-30', 25, '1 Oscar, 1 Golden Globe
287 ('1004', 'John', 'Travolta', 'American', 'USA', '1954-02-18', 40, '1 Golden Globe', 'MV00
288 ('1005', 'Tom', 'Hanks', 'American', 'USA', '1956-07-09', 40, '2 Oscars, 4 Golden Globes'
289 ('1006', 'Leonardo', 'DiCaprio', 'American', 'USA', '1974-11-11', 30, '1 Oscar, 3 Golden
290
291 -- Inserting data into the NAME_OF_THE_STUDIO table
292 • INSERT INTO NAME_OF_THE_STUDIO (studio_name, license_no, tv_series_id)
293 VALUES
294 ('Studiopolis', 'STU001', 'TS001'),
295 ('Entertainment One', 'STU002', 'TS002'),
296 ('Warner Bros. Television', 'STU003', 'TS003'),
297 ('Netflix Studios', 'STU004', 'TS004'),
298 ('Sony Pictures Television', 'STU005', 'TS005'),
299 ('Universal Television', 'STU006', 'TS006');
300
```
- Object Info Session** tabs at the bottom.
- Query Completed** message at the bottom.
- System tray icons** at the bottom right.
- Bottom status bar** showing 80%, 212 PM, ENG, 4/5/2024.

FIGURE 13 : INSERTING TUPLES INTO MAIN ACTOR AND NAME OF THE STUDIO TABLES

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

foe_23

- Tables
- Views
- Stored Procedures
- Functions

sys

tv_series_movies_collection

EE4202_Final

```

301 -- Inserting data into the NAME_OF_THE_WEBSITE table
302 • INSERT INTO NAME_OF_THE_WEBSITE (website_name, license_no, tv_series_id)
303     VALUES
304     ('Netflix', 'WEB001', 'TS001'),
305     ('Hulu', 'WEB002', 'TS002'),
306     ('Amazon Prime Video', 'WEB003', 'TS003'),
307     ('Disney+', 'WEB004', 'TS004'),
308     ('Apple TV+', 'WEB005', 'TS005'),
309     ('HBO Max', 'WEB006', 'TS006');
310
311 -- Inserting data into the SONG table
312 • INSERT INTO SONG (LICENSE_ID, SONG_NAME, SINGER, WRITER, S_LANGUAGE, AWARDS, movie_id)
313     VALUES
314     ('SONG001', 'My Heart Will Go On', 'Celine Dion', 'James Horner', 'English', '4 Grammys', 'MV001'),
315     ('SONG002', 'Let It Go', 'Idina Menzel', 'Robert Lopez', 'English', '2 Grammys', 'MV002'),
316     ('SONG003', 'Skyfall', 'Adele', 'Adele Adkins', 'English', '1 Grammy', 'MV003'),
317     ('SONG004', 'Jai Ho', 'A.R. Rahman', 'Gulzar', 'Hindi', '2 Oscars', 'MV004'),
318     ('SONG005', 'Can You Feel the Love Tonight', 'Elton John', 'Tim Rice', 'English', '1 Oscar', 'MV005'),
319     ('SONG006', 'Shallow', 'Lady Gaga & Bradley Cooper', 'Lady Gaga', 'English', '1 Oscar', 'MV006');
320
321 -- Inserting data into the REVIEW table
322 • INSERT INTO REVIEW (COMMENTS, RATING, REVIEW_DATE, EXPERIENCE, movie_id, tv_series_id)
323     VALUES
324     ('Great movie!', 4.5, '2023-01-15', 'Captivating storyline.', 'MV001', 'TS001'),
325     ('Fantastic performance!', 4.8, '2023-03-22', 'Amazing acting.', 'MV002', 'TS002'),
326     ('Brilliant cinematography!', 4.2, '2023-05-10', 'Beautifully shot.', 'MV003', 'TS003').

```

No object selected

Object Info Session

Query Completed

Context Help Snippets

FIGURE 15 : INSERTING TUPLES INTO NAME_OF_THE_WEBSITE, SONG AND REVIEW TABLES

3.4 Creating foreign key constraints

This screenshot shows the MySQL Workbench interface with the 'Workshop_2and3' database selected. The 'tv_seris_movies_collection1' schema is currently expanded in the Navigator pane. The SQL editor tab displays the following SQL code:

```
338 • alter table DIRECTOR add constraint fk1
339   foreign key(movie_id) references MOVIE(MOVIE_ID)
340   on delete set null
341   on update cascade;
342
343 • alter table DIRECTOR add constraint fk2
344   foreign key(tv_series_id) references TV_SERIES(TV_SERIES_ID)
345   on delete set null
346   on update cascade;
347
348 • alter table TV_SERIES add constraint fk3
349   foreign key(country_name) references COUNTRY(COUNTRY_NAME)
350   on delete set null
351   on update cascade;
352
353 • alter table EPISODE add constraint fk4
354   foreign key(tv_series_id) references TV_SERIES(TV_SERIES_ID)
355   on update cascade;
356
357 • alter table STUDIO add constraint fk5
358   foreign key(tv_series_id) references TV_SERIES(TV_SERIES_ID)
359   on delete set null
360   on update cascade;
361
362 • alter table NAME_OF_THE_STUDIO add constraint fk6
363   foreign key(tv_series_id) references TV_SERIES(TV_SERIES_ID)
364   on delete set null
```

The status bar at the bottom right shows the date and time as 11:21 PM 4/8/2024.

This screenshot continues the MySQL Workbench session, showing the same environment and schema structure. The SQL editor tab now contains the following additional code:

```
367 • alter table NAME_OF_THE_WEBSITE add constraint fk7
368   foreign key(tv_series_id) references TV_SERIES(TV_SERIES_ID)
369   on delete set null
370   on update cascade;
371
372 • alter table MAIN_ACTOR add constraint fk8
373   foreign key(movie_id) references MOVIE(MOVIE_ID)
374   on delete set null
375   on update cascade;
376
377 • alter table MAIN_ACTOR add constraint fk9
378   foreign key(president) references MAIN_ACTOR(ACTOR_LICENSE_NO)
379   on delete set null
380   on update cascade;
381
382 • alter table SONG add constraint fk10
383   foreign key(movie_id) references MOVIE(MOVIE_ID)
384   on delete set null
385   on update cascade;
386
387 • alter table REVIEW add constraint fk11
388   foreign key(tv_series_id) references TV_SERIES(TV_SERIES_ID)
389   on delete set null
390   on update cascade;
391
392 • alter table REVIEW add constraint fk12 |
393   foreign key(movie_id) references MOVIE(MOVIE_ID)
394   on delete set null
395   on update cascade;
```

The status bar at the bottom right shows the date and time as 11:22 PM 4/8/2024.

```
224
225 • alter table DIRECTOR add constraint fk1
226   foreign key(movie_id) references MOVIE(MOVIE_ID)
227   on delete set null
228   on update cascade;
```

FIGURE 17 : INSERTION OF 1ST FOREIGN KEY IN DIRECTOR RELATION

```
229
230 • alter table DIRECTOR add constraint fk2
231   foreign key(tv_series_id) references TV_SERIES(TV_SERIES_ID)
232   on delete set null
233   on update cascade;
```

FIGURE 16 : INSERTION OF 2ND FOREIGN KEY IN DIRECTOR RELATION

```
235 • alter table TV_SERIES add constraint fk3
236   foreign key(country_name) references COUNTRY(COUNTRY_NAME)
237   on delete set null
238   on update cascade;
239
```

```
239
240 • alter table EPISODE add constraint fk4
241   foreign key(tv_series_id) references TV_SERIES(TV_SERIES_ID)
242   on update cascade;
243
```

```
248
249 • alter table NAME_OF_THE_STUDIO add constraint fk6
250   foreign key(tv_series_id) references TV_SERIES(TV_SERIES_ID)
251   on delete set null
252   on update cascade;
```

FIGURE 18 : INSERTION OF 1ST FOREIGN KEY IN EPISODE RELATION

```
254 • alter table NAME_OF_THE_WEBSITE add constraint fk7  
255     foreign key(tv_series_id) references TV_SERIES(TV_SERIES_ID)  
256     on delete set null  
257     on update cascade;
```

FIGURE 19 : INSERTION OF 1ST FOREIGN KEY IN NAME_OF_THE_WEBSITE RELATION

```
243  
244 • alter table STUDIO add constraint fk5  
245     foreign key(tv_series_id) references TV_SERIES(TV_SERIES_ID)  
246     on delete set null  
247     on update cascade;
```

FIGURE 20 : INSERTION OF 1ST FOREIGN KEY IN NAME_OF_THE_STUDIO RELATION

```
259 • alter table MAIN_ACTOR add constraint fk8  
260   foreign key(movie_id) references MOVIE(MOVIE_ID)  
261   on delete set null  
262   on update cascade;  
263
```

FIGURE 21 : INSERTION OF 1ST FOREIGN KEY IN MAIN_ACTOR RELATION

```
264 • alter table MAIN_ACTOR add constraint fk9  
265   foreign key(president) references MAIN_ACTOR(ACTOR_LICENSE_NO)  
266   on delete set null  
267   on update cascade;  
268
```

FIGURE 22: INSERTION OF 2ND FOREIGN KEY IN MAIN_ACTOR RELATION

```
269 • alter table SONG add constraint fk10  
270   foreign key(movie_id) references MOVIE(MOVIE_ID)  
271   on delete set null  
272   on update cascade;
```

FIGURE 25: INSERTION OF 1ST FOREIGN KEY IN SONG RELATION

```
274 • alter table REVIEW add constraint fk11  
275   foreign key(tv_series_id) references TV_SERIES(TV_SERIES_ID)  
276   on delete set null  
277   on update cascade;
```

FIGURE 24: INSERTION OF 1ST FOREIGN KEY IN REVIEW RELATION

```
279 • alter table REVIEW add constraint fk12  
280   foreign key(movie_id) references MOVIE(MOVIE_ID)  
281   on delete set null  
282   on update cascade;
```

FIGURE 23: INSERTION OF 2ND FOREIGN KEY IN REVIEW RELATION

3.5 Updating and Deleting

	COUNTRY_NAME	COUNTRY_RANK	REGION
▶	Australia	6	Oceania
	Canada	4	North America
	Germany	5	Europe
	Japan	3	Asia
	UK	2	Europe
	USA	1	North America
*	HULL	NULL	NULL

FIGURE 29 : TABLE OF COUNTRY RELATION BEFORE UPDATE

```
298    -- Update function 1
299 •  UPDATE COUNTRY
300   SET COUNTRY_RANK = 10
301   WHERE COUNTRY_NAME = 'Japan';
302
303    -- Update function 2
304 •  UPDATE COUNTRY
305   SET REGION = 'America-1'
306   WHERE COUNTRY_NAME = 'Canada';
...  
...
```

FIGURE 28 : QUERY FOR THE ABOVE UPDATING COUNTRY RELATION

	COUNTRY_NAME	COUNTRY_RANK	REGION
▶	Australia	6	Oceania
	Canada	4	America-1
	Germany	5	Europe
	Japan	10	Asia
	UK	2	Europe
	USA	1	North America
*	HULL	NULL	NULL

FIGURE 27 : TABLE OF COUNTRY RELATION AFTER THE 1ST UPDATE

```
307
308    -- Delete function
309 •  DELETE FROM COUNTRY
310   WHERE COUNTRY_NAME = 'Germany';  
...
```

FIGURE 26 : QUERY FOR THE ABOVE DELETING IN COUNTRY RELATION

	COUNTRY_NAME	COUNTRY_RANK	REGION
▶	Australia	6	Oceania
	Canada	4	America-1
	Japan	10	Asia
	UK	2	Europe
	USA	1	North America
●	NULL	NULL	NULL

FIGURE 31 : TABLE OF COUNTRY RELATION AFTER THE DELETE

```

312      -- Update function 1
313 •   UPDATE DIRECTOR
314     SET ACTIVE_PERIOD_IN_YEARS = 35
315     WHERE LICENSE_NO = 'LIC001';
316
317      -- Delete function
318 •   DELETE FROM DIRECTOR
319     WHERE LICENSE_NO = 'LIC006';

```

FIGURE 30 : QUERY FOR THE ABOVE UPDATING AND DELETING IN DIRECTOR RELATION

LICENSE_NO	DIRECTOR_NAME	AGE	ACTIVE_PERIOD_IN_YEARS	COUNTRY	NATIONALITY	AWARDS_AND_NOMINATIONS	movie_id	tv_series_id
LIC001	Frank Darabont	62	25	USA	American	2 Oscars, 5 Golden Globes	MV001	NULL
LIC002	Francis Ford Coppola	83	50	USA	American	5 Oscars, 3 Golden Globes	MV002	NULL
LIC003	Christopher Nolan	52	25	UK	British	1 Oscar, 4 Golden Globes	MV003	NULL
LIC004	Quentin Tarantino	58	30	USA	American	2 Oscars, 3 Golden Globes	MV004	NULL
LIC005	Robert Zemeckis	70	40	USA	American	1 Oscar, 3 Golden Globes	MV005	NULL
LIC006	Christopher Nolan	52	25	UK	British	1 Oscar, 4 Golden Globes	NULL	TS003
●	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

FIGURE 32 : TABLE OF DIRECTOR RELATION BEFORE UPDATE

LICENSE_NO	DIRECTOR_NAME	AGE	ACTIVE_PERIOD_IN_YEARS	COUNTRY	NATIONALITY	AWARDS_AND_NOMINATIONS	movie_id	tv_series_id
LIC001	Frank Darabont	62	35	USA	American	2 Oscars, 5 Golden Globes	MV001	NULL
LIC002	Francis Ford Coppola	83	50	USA	American	5 Oscars, 3 Golden Globes	MV002	NULL
LIC003	Christopher Nolan	52	25	UK	British	1 Oscar, 4 Golden Globes	MV003	NULL
LIC004	Quentin Tarantino	58	30	USA	American	2 Oscars, 3 Golden Globes	MV004	NULL
LIC005	Robert Zemeckis	70	40	USA	American	1 Oscar, 3 Golden Globes	MV005	NULL
LIC006	Christopher Nolan	52	25	UK	British	1 Oscar, 4 Golden Globes	NULL	TS003
●	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

FIGURE 33 : TABLE OF DIRECTOR RELATION AFTER 1ST UPDATE

	LICENSE_NO	DIRECTOR_NAME	AGE	ACTIVE_PERIOD_IN_YEARS	COUNTRY	NATIONALITY	AWARDS_AND_NOMINATIONS	movie_id	tv_series_id
▶	LIC001	Frank Darabont	62	35	USA	American	2 Oscars, 5 Golden Globes	MV001	NULL
	LIC002	Francis Ford Coppola	83	50	USA	American	5 Oscars, 3 Golden Globes	MV002	NULL
	LIC003	Christopher Nolan	52	25	UK	British	1 Oscar, 4 Golden Globes	MV003	NULL
	LIC004	Quentin Tarantino	58	30	USA	American	2 Oscars, 3 Golden Globes	MV004	NULL
*	LIC005	Robert Zemedikis	70	40	USA	American	1 Oscar, 3 Golden Globes	MV005	NULL
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

FIGURE 34 : TABLE OF DIRECTOR RELATION AFTER DELETING

	TITLE	RATINGS	EPISODE_LENGTH	tv_series_id	COST_PER_EPISODE
▶	Pilot	8.70	00:58:00	TS001	3000000.00000
	Winter Is Coming	9.00	00:58:00	TS002	8000000.00000
	Chapter of Byers	8.80	00:48:00	TS004	6000000.00000
	Wolferton Splash	8.70	00:58:00	TS005	10000000.00000
*	NULL	NULL	NULL	NULL	NULL

FIGURE 36 : TABLE OF EPISODE RELATION BEFORE UPDATING

```

435      -- Update function 1
436 •   UPDATE EPISODE
437     SET COST_PER_EPISODE = 7000000
438     WHERE TITLE = 'Pilot';
439
440      -- Update function 2
441 •   UPDATE EPISODE
442     SET RATINGS = 8.0
443     WHERE TITLE = 'Winter Is Coming';
444
445      -- Delete function
446 •   DELETE FROM EPISODE
447     WHERE TITLE = 'Pilot';
```

FIGURE 35 : QUERY FOR THE ABOVE UPDATING AND DELETING IN EPISODE RELATION

	TITLE	RATINGS	EPISODE_LENGTH	tv_series_id	COST_PER_EPISODE
▶	Pilot	8.70	00:58:00	TS001	7000000.00000
	Winter Is Coming	9.00	00:58:00	TS002	8000000.00000
	Chapter of Byers	8.80	00:48:00	TS004	6000000.00000
	Wolferton Splash	8.70	00:58:00	TS005	10000000.00000
*	NULL	NULL	NULL	NULL	NULL

FIGURE 37 : TABLE OF EPISODE RELATION AFTER 1ST AND 2ND UPDATE

	TITLE	RATINGS	EPISODE_LENGTH	tv_series_id	COST_PER_EPISODE
▶	Winter Is Coming	8.00	00:58:00	TS002	8000000.00000
	Chapter of Byers	8.80	00:48:00	TS004	6000000.00000
*	Wolferton Splash	8.70	00:58:00	TS005	10000000.00000
	NULL	NULL	NULL	NULL	NULL

FIGURE 38 : TABLE OF EPISODE RELATION AFTER DELETING

	ACTOR_LICENSE_NO	FIRST_NAME	LAST_NAME	NATIONALITY	COUNTRY	DATE_OF_BIRTH	ACTIVE_PERIOD	AWARDS	movie_id	president
▶	1001	Reset all sorted columns	freeman	American	USA	1937-06-01	50	1 Oscar, 5 Golden Globes	MV001	NULL
	1002	Marlon	Brando	American	USA	1924-04-03	30	2 Oscars, 2 Golden Globes	MV002	NULL
	1003	Christian	Bale	British	UK	1974-01-30	25	1 Oscar, 1 Golden Globe	MV003	1003
	1004	John	Travolta	American	USA	1954-02-18	40	1 Golden Globe	MV004	1004
	1005	Tom	Hanks	American	USA	1956-07-09	40	2 Oscars, 4 Golden Globes	MV005	1005
*	1006	Leonardo	DiCaprio	American	USA	1974-11-11	30	1 Oscar, 3 Golden Globes	MV006	1006
	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

FIGURE 40 : TABLE OF MAIN_ACTOR RELATION BEFORE UPDATING

```

335      -- Update function 1
336 • UPDATE MAIN_ACTOR
337     SET AWARDS = '3 Oscars, 5 Golden Glo
338     WHERE ACTOR_LICENSE_NO = 1001;
339
340      -- Update function 2
341 • UPDATE MAIN_ACTOR
342     SET DATE_OF_BIRTH = '1976-11-11'
343     WHERE ACTOR_LICENSE_NO = 1006;
344
345      -- Delete function
346 • DELETE FROM MAIN_ACTOR
347     WHERE ACTOR_LICENSE_NO = 1004;

```

FIGURE 39 : QUERY FOR THE ABOVE UPDATING AND DELETING IN MAIN_ACTOR RELATION

	ACTOR LICENSE NO	FIRST NAME	LAST NAME	NATIONALITY	COUNTRY	DATE OF BIRTH	ACTIVE PERIOD	AWARDS	movie_id	president
▶	1001	Morgan	Freeman	American	USA	1937-06-01	50	3 Oscars, 5 Golden Globes	MV001	NULL
	1002	Marlon	Brando	American	USA	1924-04-03	30	2 Oscars, 2 Golden Globes	MV002	NULL
	1003	Christian	Bale	British	UK	1974-01-30	25	1 Oscar, 1 Golden Globe	MV003	1003
	1004	John	Travolta	American	USA	1954-02-18	40	1 Golden Globe	MV004	1004
	1005	Tom	Hanks	American	USA	1956-07-09	40	2 Oscars, 4 Golden Globes	MV005	1005
*	1006	Leonardo	DiCaprio	American	USA	1976-11-11	30	1 Oscar, 3 Golden Globes	MV006	1006
	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

FIGURE 41 : TABLE OF MAIN_ACTOR RELATION AFTER 1ST AND 2ND UPDATE

	ACTOR_LICENSE_NO	FIRST_NAME	LAST_NAME	NATIONALITY	COUNTRY	DATE_OF_BIRTH	ACTIVE_PERIOD	AWARDS	movie_id	president
▶	1001	Morgan	Freeman	American	USA	1937-06-01	50	3 Oscars, 5 Golden Globes	MV001	NULL
	1002	Marlon	Brando	American	USA	1924-04-03	30	2 Oscars, 2 Golden Globes	MV002	NULL
	1003	Christian	Bale	British	UK	1974-01-30	25	1 Oscar, 1 Golden Globe	MV003	1003
	1005	Tom	Hanks	American	USA	1956-07-09	40	2 Oscars, 4 Golden Globes	MV005	1005
*	1006	Leonardo	DiCaprio	American	USA	1976-11-11	30	1 Oscar, 3 Golden Globes	MV006	1006
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

FIGURE 42 : TABLE OF MAIN_ACTOR RELATION AFTER DELETING

	studio_name	license_no	tv_series_id
▶	Studiopolis	STU001	TS001
	Entertainment One	STU002	TS002
	Warner Bros. Television	STU003	TS003
	Netflix Studios	STU004	TS004
	Sony Pictures Television	STU005	TS005
*	Universal Television	STU006	TS006
*	NULL	NULL	NULL

FIGURE 43 : TABLE OF NAME_OF_THE_STUDIO RELATION BEFORE UPDATING

```

378      -- Update function 1
379 • UPDATE NAME_OF_THE_STUDIO
380     SET studio_name = 'Studio X'
381     WHERE license_no = 'STU001';
382
383      -- Update function 2
384 • UPDATE NAME_OF_THE_STUDIO
385     SET tv_series_id = 'TS002'
386     WHERE license_no = 'STU004';
387
388      -- Delete function
389 • DELETE FROM NAME_OF_THE_STUDIO
390     WHERE license_no = 'STU006';

```

FIGURE 44 : QUERY FOR THE ABOVE UPDATING AND DELETING IN NAME_OF_THE_STUDIO RELATION

	studio_name	license_no	tv_series_id
▶	Studio X	STU001	TS001
	Entertainment One	STU002	TS002
	Warner Bros. Television	STU003	TS003
▶	Netflix Studios	STU004	TS002
	Sony Pictures Television	STU005	TS005
▶	Universal Television	STU006	TS006
*	NULL	NULL	NULL

FIGURE 46 : TABLE OF NAME_OF_THE_STUDIO RELATION AFTER 1ST AND 2ND UPDATING

	studio_name	license_no	tv_series_id
▶	Studio X	STU001	TS001
	Entertainment One	STU002	TS002
	Warner Bros. Television	STU003	TS003
▶	Netflix Studios	STU004	TS002
	Sony Pictures Television	STU005	TS005
*	NULL	NULL	NULL

FIGURE 45 : TABLE OF NAME_OF_THE_STUDIO RELATION AFTER DELETING

	website_name	license_no	tv_series_id
▶	Netflix	WEB001	TS001
	Hulu	WEB002	TS002
	Amazon Prime Video	WEB003	TS003
	Disney+	WEB004	TS004
	Apple TV+	WEB005	TS005
	HBO Max	WEB006	TS006
*	NULL	NULL	NULL

FIGURE 48 : TABLE OF NAME_OF_THE_WEBSITE RELATION BEFORE UPDATING

```

392 -- Update function 1
393 • UPDATE NAME_OF_THE_WEBSITE
394 SET website_name = 'Netflix Prime'
395 WHERE license_no = 'WEB001';
396
397 -- Update function 2
398 • UPDATE NAME_OF_THE_WEBSITE
399 SET tv_series_id = 'TS005'
400 WHERE license_no = 'WEB003';
401
402 -- Delete function
403 • DELETE FROM NAME_OF_THE_WEBSITE
404 WHERE license_no = 'WEB006';

```

FIGURE 47: QUERY FOR THE ABOVE UPDATING AND DELETING IN
NAME_OF_THE_WEBSITE RELATION

	website_name	license_no	tv_series_id
▶	Netflix Prime	Netflix Prime	TS001
	Hulu	WEB002	TS002
	Amazon Prime Video	WEB003	TS005
	Disney+	WEB004	TS004
	Apple TV+	WEB005	TS005
	HBO Max	WEB006	TS006
*	NULL	NULL	NULL

FIGURE 49 : : TABLE OF NAME_OF THE WEBSITE RELATION AFTER 1ST AND 2ND UPDATING

	website_name	license_no	tv_series_id
▶	Netflix Prim	Netflix Prime 001	TS001
	Hulu	WEB002	TS002
	Amazon Prime Video	WEB003	TS005
	Disney+	WEB004	TS004
	Apple TV+	WEB005	TS005
*	NULL	NULL	NULL

FIGURE 50 : : TABLE OF NAME_OF THE WEBSITE RELATION AFTER DELETING

LICENSE_ID	SONG_NAME	SINGER	WRITER	S_LANGUAGE	AWARDS	movie_id
▶ SONG001	My Heart Will Go On	Celine Dion	James Horner	English	4 Grammys	MV001
SONG002	Let It Go	Idina Menzel	Robert Lopez	English	2 Grammys	MV002
SONG003	Skyfall	Adele	Adele Adkins	English	1 Grammy	MV003
SONG004	Jai Ho	A.R. Rahman	Gulzar	Hindi	2 Oscars	MV004
SONG005	Can You Feel the Love Tonight	Elton John	Tim Rice	English	1 Oscar	MV005
SONG006	Shallow	Lady Gaga & Bradley Cooper	Lady Gaga	English	1 Oscar	MV006
*	NULL	NULL	NULL	NULL	NULL	NULL

FIGURE 52: TABLE OF SONG RELATION BEFORE UPDATING

```

406    -- Update function 1
407 • UPDATE SONG
408    SET AWARDS = '5 Grammys'
409    WHERE LICENSE_ID = 'SONG001';
410
411    -- Update function 2
412 • UPDATE SONG
413    SET SINGER = 'Arijit Singh'
414    WHERE LICENSE_ID = 'SONG004';
415
416    -- Delete function
417 • DELETE FROM SONG
418    WHERE LICENSE_ID = 'SONG006';

```

FIGURE 51 : QUERY FOR THE ABOVE UPDATING AND DELETING IN SONG RELATION

LICENSE_ID	SONG_NAME	SINGER	WRITER	S_LANGUAGE	AWARDS	movie_id
SONG001	My Heart Will Go On	Celine Dion	James Horner	English	5 Grammys	MV001
SONG002	Let It Go	Idina Menzel	Robert Lopez	English	2 Grammys	MV002
SONG003	Skyfall	Adele	Adele Adkins	English	1 Grammy	MV003
SONG004	Jai Ho	Arijit Singh	Gulzar	Hindi	2 Oscars	MV004
SONG005	Can You Feel the Love Tonight	Elton John	Tim Rice	English	1 Oscar	MV005
SONG006	Shallow	Lady Gaga & Bradley Cooper	Lady Gaga	English	1 Oscar	MV006
NULL	NULL	NULL	NULL	NULL	NULL	NULL

FIGURE 53 : TABLE OF SONG RELATION AFTER 1ST AND 2ND UPDATING

LICENSE_ID	SONG_NAME	SINGER	WRITER	S_LANGUAGE	AWARDS	movie_id
SONG001	My Heart Will Go On	Celine Dion	James Horner	English	5 Grammys	MV001
SONG002	Let It Go	Idina Menzel	Robert Lopez	English	2 Grammys	MV002
SONG003	Skyfall	Adele	Adele Adkins	English	1 Grammy	MV003
SONG004	Jai Ho	Arijit Singh	Gulzar	Hindi	2 Oscars	MV004
SONG005	Can You Feel the Love Tonight	Elton John	Tim Rice	English	1 Oscar	MV005
NULL	NULL	NULL	NULL	NULL	NULL	NULL

FIGURE 54: TABLE OF SONG RELATION AFTER DELETING

	COMMENTS	RATING	REVIEW_DATE	EXPERIENCE	movie_id	tv_series_id
▶	Great movie!	4.500	2023-01-15	Captivating storyline.	MV001	TS001
	Fantastic performance!	4.800	2023-03-22	Amazing acting.	MV002	TS002
	Brilliant cinematography!	4.200	2023-05-10	Beautifully shot.	MV003	TS003
	Outstanding music!	4.700	2023-07-05	Great soundtrack.	MV004	TS004
	Classic Disney!	4.600	2023-09-18	Nostalgic.	MV005	TS005
	Incredible chemistry!	4.900	2023-11-30	Compelling performances.	MV006	TS006
●	NULL	NULL	NULL	NULL	NULL	NULL

FIGURE 56 : TABLE OF REVIEW RELATION BEFORE UPDATING

```

420    -- Update function 1
421 • UPDATE REVIEW
422   SET RATING = 4.9
423   WHERE movie_id = 'MV001';
424
425    -- Update function 2
426 • UPDATE REVIEW
427   SET COMMENTS = 'The best series ever'
428   WHERE tv_series_id = 'TS002';
429
430    -- Delete function
431 • DELETE FROM REVIEW
432   WHERE movie_id = 'MV006';

```

FIGURE 55 : QUERY FOR THE ABOVE UPDATING AND DELETING IN REVIEW RELATION

	COMMENTS	RATING	REVIEW_DATE	EXPERIENCE	movie_id	tv_series_id
▶	Great movie!	4.900	2023-01-15	Captivating storyline.	MV001	TS001
	The best series ever!	4.800	2023-03-22	Amazing acting.	MV002	TS002
	Brilliant cinematography!	4.200	2023-05-10	Beautifully shot.	MV003	TS003
	Outstanding music!	4.700	2023-07-05	Great soundtrack.	MV004	TS004
	Classic Disney!	4.600	2023-09-18	Nostalgic.	MV005	TS005
	Incredible chemistry!	4.900	2023-11-30	Compelling performances.	MV006	TS006
*	HULL	HULL	HULL	HULL	HULL	HULL

FIGURE 57: TABLE OF REVIEW RELATION AFTER 1ST AND 2ND UPDATING

	COMMENTS	RATING	REVIEW_DATE	EXPERIENCE	movie_id	tv_series_id
▶	Great movie!	4.900	2023-01-15	Captivating storyline.	MV001	TS001
	The best series ever!	4.800	2023-03-22	Amazing acting.	MV002	TS002
	Brilliant cinematography!	4.200	2023-05-10	Beautifully shot.	MV003	TS003
	Outstanding music!	4.700	2023-07-05	Great soundtrack.	MV004	TS004
	Classic Disney!	4.600	2023-09-18	Nostalgic.	MV005	TS005
*	HULL	HULL	HULL	HULL	HULL	HULL

FIGURE 58: TABLE OF REVIEW RELATION AFTER DELETING

	LICENSE_NO	ESTABLISHED_DATE	REVENUE	COUNTRY	tv_series_id
▶	ST001	1986-01-15	25000000.000	USA	TS001
	ST002	1977-05-25	35000000.000	USA	TS002
	ST003	1939-06-20	18000000.000	USA	TS003
	ST004	2006-07-18	40000000.000	USA	TS004
	ST005	2010-01-01	20000000.000	UK	TS005
	ST006	1998-01-01	30000000.000	USA	TS006
*	HULL	HULL	HULL	HULL	HULL

FIGURE 59 :: TABLE OF STUDIO RELATION BEFORE UPDATING

```

351      -- Update function 1
352  •   UPDATE STUDIO
353      SET REVENUE = 30000000
354      WHERE LICENSE_NO = 'ST001';
355
356      -- Update function 2
357  •   UPDATE STUDIO
358      SET COUNTRY = 'UK'
359      WHERE LICENSE_NO = 'ST004';
360
361      -- Delete function
362  •   DELETE FROM STUDIO
363      WHERE LICENSE_NO = "STU006";

```

Wrong
LICENSE_NO. So
that row was not
removed

FIGURE 60: QUERY FOR THE ABOVE UPDATING AND DELETING IN STUDIO RELATION (WITH SOME ERROR)

LICENSE_NO	ESTABLISHED_DATE	REVENUE	COUNTRY	tv_series_id
ST001	1986-01-15	30000000.000	USA	TS001
ST002	1977-05-25	35000000.000	USA	TS002
ST003	1939-06-20	18000000.000	USA	TS003
ST004	2006-07-18	40000000.000	UK	TS004
ST005	2010-01-01	20000000.000	UK	TS005
ST006	1998-01-01	30000000.000	USA	NULL
*	NULL	NULL	NULL	NULL

FIGURE 61 : TABLE OF STUDIO RELATION AFTER 1ST AND 2ND UPDATING

LICENSE_NO	ESTABLISHED_DATE	REVENUE	COUNTRY	tv_series_id
ST001	1986-01-15	30000000.000	USA	TS001
ST002	1977-05-25	35000000.000	USA	TS002
ST003	1939-06-20	18000000.000	USA	TS003
ST004	2006-07-18	40000000.000	UK	TS004
ST005	2010-01-01	20000000.000	UK	TS005
ST006	1998-01-01	30000000.000	USA	NULL
*	NULL	NULL	NULL	NULL

FIGURE 63 : TABLE OF STUDIO RELATION AFTER TRYING TO DELETE

```

351      -- Update function 1
352  ●   UPDATE STUDIO
353      SET REVENUE = 30000000
354      WHERE LICENSE_NO = 'ST001';
355
356      -- Update function 2
357  ●   UPDATE STUDIO
358      SET COUNTRY = 'UK'
359      WHERE LICENSE_NO = 'ST004';
360
361      -- Delete function
362  ●   DELETE FROM STUDIO
363      WHERE LICENSE_NO = 'ST006';
```

Now it has removed

LICENSE_NO	ESTABLISHED_DATE	REVENUE	COUNTRY	tv_series_id
ST001	1986-01-15	30000000.000	USA	TS001
ST002	1977-05-25	35000000.000	USA	TS002
ST003	1939-06-20	18000000.000	USA	TS003
ST004	2006-07-18	40000000.000	UK	TS004
ST005	2010-01-01	20000000.000	UK	TS005
*	NULL	NULL	NULL	NULL

FIGURE 62 : TABLE OF STUDIO RELATION AFTER DELETING

	TV_SERIES_ID	TV_SERIES_NAME	RELEASED_YEAR	GENRE	TS_LANGUAGE	NO_OF_SEASONS	country_name
▶	TS001	Breaking Bad	2008	Drama	English	5	USA
	TS002	Game of Thrones	2011	Fantasy	English	8	USA
	TS003	Friends	1994	Comedy	English	10	USA
	TS004	Stranger Things	2016	Science Fiction	English	4	USA
	TS005	The Crown	2016	Historical Drama	English	4	UK
*	TS006	Narcos	2015	Crime Drama	English	3	USA
	NULL	NULL	NULL	NULL	NULL	NULL	NULL

FIGURE 64: TABLE OF TV_SERIES RELATION BEFORE UPDATING

	TV_SERIES_ID	TV_SERIES_NAME	RELEASED_YEAR	GENRE	TS_LANGUAGE	NO_OF_SEASONS	country_name
▶	TS001	Breaking Bad	2008	Drama	English	6	USA
	TS002	Game of Thrones	2011	Fantasy	English	8	USA
	TS003	Friends	1994	Comedy	English	10	USA
	TS004	Stranger Things	2016	Science Fiction	English	4	USA
	TS005	The Crown	2016	Historical Drama	English	4	UK
*	TS006	Narcos	2015	Crime Drama	English	3	USA
	NULL	NULL	NULL	NULL	NULL	NULL	NULL

FIGURE 65 : TABLE OF TV_SERIES RELATION AFTER 1ST UPDATING

	TV_SERIES_ID	TV_SERIES_NAME	RELEASED_YEAR	GENRE	TS_LANGUAGE	NO_OF_SEASONS	country_name
▶	TS001	Breaking Bad	2008	Drama	English	6	USA
	TS002	Game of Thrones	2013	Fantasy	English	8	USA
	TS003	Friends	1994	Comedy	English	10	USA
	TS004	Stranger Things	2016	Science Fiction	English	4	USA
	TS005	The Crown	2016	Historical Drama	English	4	UK
*	TS006	Narcos	2015	Crime Drama	English	3	USA
	NULL	NULL	NULL	NULL	NULL	NULL	NULL

FIGURE 66 : TABLE OF TV_SERIES RELATION AFTER 2ND UPDATING

ERROR MESSAGE! WHEN TRYING TO DELETE A TUPLUE FROM THE TV_SERIES RELATION

alter table EPISODE add constraint fk4 foreign key(tv_series_id) references 0 227 23:07:47 TV_SERIES(TV_SERIES_ID) on update cascade ON DELETE CASCADE	Error Code: 1826. Duplicate foreign key constraint name sec 0.000 'fk4'
---	--

BECAUSE OF THIS CONSTRAINT

```
240 • alter table EPISODE add constraint fk4
241     foreign key(tv_series_id) references TV_SERIES(TV_SERIES_ID)
242     on update cascade;
243
```

```

240 • alter table EPISODE add constraint fk4
241   foreign key(tv_series_id) references TV_SERIES(TV_SERIES_ID)
242   on update cascade
243   on delete cascade;

```

FIGURE 68 : UPDATED CONSTRAINT

```

364      -- Update function 1
365 • UPDATE TV_SERIES
366   SET NO_OF_SEASONS = 6
367   WHERE TV_SERIES_ID = 'TS001';
368
369      -- Update function 2
370 • UPDATE TV_SERIES
371   SET RELEASED_YEAR = 2013
372   WHERE TV_SERIES_NAME = 'Game of Thrones';
373
374      -- Delete function
375 • DELETE FROM TV_SERIES
376   WHERE TV_SERIES_ID = 'TS006';

```

FIGURE 67: QUERY FOR THE ABOVE UPDATING AND DELETING IN TV_SERIES

	TV_SERIES_ID	TV_SERIES_NAME	RELEASED_YEAR	GENRE	TS_LANGUAGE	NO_OF_SEASONS	country_name
▶	TS001	Breaking Bad	2008	Drama	English	6	USA
	TS002	Game of Thrones	2013	Fantasy	English	8	USA
	TS003	Friends	1994	Comedy	English	10	USA
	TS004	Stranger Things	2016	Science Fiction	English	4	USA
*	TS005	The Crown	2016	Historical Drama	English	4	UK
	NULL	NULL	NULL	NULL	NULL	NULL	NULL

FIGURE 69 : TABLE OF TV_SERIES RELATION AFTER DELETING

4. Chapter 4 - Transactions

i. Simple queries

This screenshot shows the MySQL Workbench interface with a query editor window titled "EE4202_Final". The code in the editor is as follows:

```
562 -- retrieve all the data from the relation movie
563 • SELECT *
564   FROM MOVIE;
565
566 -- retrieve distinct genre values from the movie relation
567 • SELECT DISTINCT GENRE
568   FROM MOVIE;
569
570 -- retrieving data from 2 tables by combining
571 • SELECT MOVIE_NAME FROM MOVIE WHERE RELEASED_YEAR = 1994
572 UNION
573 SELECT TV_SERIES_NAME FROM TV_SERIES WHERE RELEASED_YEAR = 2016;
574
575 -- retrieving data from the movie relation in the descending order of released year
576 • SELECT * FROM MOVIE
577 ORDER BY RELEASED_YEAR
578 DESC;
579
580 -- selecting the tuples of movie whose released year is between 2000 and 2010
581 • SELECT *
582   FROM MOVIE
583   WHERE RELEASED_YEAR
584     BETWEEN 2000 AND 2010;
585
```

The "Information" pane on the left shows details for the "episode" table, including its columns: TITLE (PK), RATINGS, EPISODE_LENGTH (PK), tv_series_id (PK), and COST_PER_EPISODE (PK).

This screenshot shows the MySQL Workbench interface with a query editor window titled "EE4202_Final". The code in the editor is as follows:

```
586 -- retrieving only particular 2 columns
587 • SELECT MOVIE_NAME, GENRE
588   FROM MOVIE;
589
590 • SELECT *
591   FROM MOVIE, DIRECTOR;
592
593 -- creating the view
594 • CREATE VIEW MovieDetails
595 AS
596   SELECT MOVIE_NAME, GENRE, RELEASED_YEAR
597   FROM MOVIE;
598
599 -- display the view
600 • SHOW FULL TABLES
601   IN TVSERIES_MOVIES_COLLECTION1
602   WHERE TABLE_TYPE LIKE 'VIEW';
603
604 -- retrieve the director name who has the age of 20
605 • SELECT
606   DIRECTOR_NAME AS Director,
607   AGE AS Director_Age
608   FROM DIRECTOR
609   where AGE = 20;
```

The "Information" pane on the left shows details for the "episode" table, including its columns: TITLE (PK), RATINGS, EPISODE_LENGTH (PK), tv_series_id (PK), and COST_PER_EPISODE (PK).

MySQL Workbench

Local instance MySQL80 ×

File Edit View Query Database Server Tools Scripting Help

Navigator: EE4202_Final* | director movie

SCHEMAS: tvseries_movies_collection

Tables: country, director, episode, main_actor, movie, name_of_the_studio, name_of_the_website, review, song, studio, tv_series

Views, Stored Procedures, Functions

Information: Administration Schemas

Table: episode

Columns:

- TITLE** va PK
- RATINGS do
- EPISODE_LENGTH tin
- tv_series_id** va PK
- COST_PER_EPISODE do

SQLAdditions: Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

```

610
611 -- rename and selecting average rating
612 • SELECT
613 AVG(RATING) AS Avg_Rating
614 FROM REVIEW;
615
616 -- selecting the minimum released year by renaming as latest_release
617 • SELECT
618 min(RELEASED_YEAR)
619 AS Latest_Release
620 FROM MOVIE;
621
622 -- retrieve the data from the relation movie by using the like operator
623 • SELECT *
624 FROM MOVIE
625 WHERE MOVIE_NAME LIKE '%Ava%';
626
627
628 -- Create a view for MOVIE and DIRECTOR tables
629 • CREATE VIEW MOVIE_DIRECTOR_VIEW AS
630 SELECT m.MOVIE_ID, m.MOVIE_NAME, m.RELEASED_YEAR, m.GENRE,m.DURATION, m.M_LANGUAGE,m.BUDGET, d.DIRECTOR_NAME,d.
631 FROM MOVIE m LEFT JOIN DIRECTOR d
632 ON m.MOVIE_ID = d.movie_id;
633

```

Object Info Session Output

82°F Light rain Search

7:23 PM 4/8/2024

448

449 • `SELECT *`

450 `FROM MOVIE;`

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: |

MOVIE_ID	MOVIE_NAME	RELEASED_YEAR	GENRE	DURATION	M_LANGUAGE	BUDGET
MV001	The Shawshank Redemption	1994	Drama	02:22:00	English	25000000.00000
MV002	The Godfather	1972	Drama, Crime	02:55:00	English	6000000.00000
MV003	The Dark Knight	2008	Action	02:32:00	English	20000000.00000
MV004	Pulp Fiction	1994	Crime	02:34:00	English	8000000.00000
MV005	Forrest Gump	1994	Drama	02:22:00	English	55000000.00000
MV006	Inception	2010	Science Fiction	02:28:00	English	160000000.00000
NULL	NULL	NULL	NULL	NULL	NULL	NULL

FIGURE 70 : QUERY AND OUTPUT FOR THE SELECT OPERATION

451

452 • `SELECT DISTINCT`

453 `GENRE FROM MOVIE;`

Result Grid | Filter Rows: | Export/Import: | Wrap Cell Content: |

GENRE
Drama
Drama, Crime
Action
Crime
Science Fiction

FIGURE 71 : QUERY AND OUTPUT FOR THE DISTINCT SELECT OPERATION

```

454
455 •   SELECT MOVIE_NAME FROM MOVIE WHERE RELEASED_YEAR = 1994
456 UNION
457 SELECT TV_SERIES_NAME FROM TV_SERIES WHERE RELEASED_YEAR = 2016;
458
459

```

Result Grid | Filter Rows: Export: Wrap Cell Content:

MOVIE_NAME
The Shawshank Redemption
Pulp Fiction
Forrest Gump
Stranger Things
The Crown

FIGURE 72 : QUERY AND OUTPUT FOR THE SET OPERATION UNION

```

458
459 •   SELECT * FROM MOVIE ORDER BY RELEASED_YEAR DESC;
460
461

```

Result Grid | Filter Rows: Edit: Export/Import: Wrap Cell Content:

MOVIE_ID	MOVIE_NAME	RELEASED_YEAR	GENRE	DURATION	M_LANGUAGE	BUDGET
MV006	Inception	2010	Science Fiction	02:28:00	English	160000000.00000
MV003	The Dark Knight	2008	Action	02:32:00	English	200000000.00000
MV001	The Shawshank Redemption	1994	Drama	02:22:00	English	25000000.00000
MV004	Pulp Fiction	1994	Crime	02:34:00	English	8000000.00000
MV005	Forrest Gump	1994	Drama	02:22:00	English	55000000.00000
MV002	The Godfather	1972	Drama, Crime	02:55:00	English	6000000.00000
*	NULL	NULL	NULL	NULL	NULL	NULL

FIGURE 73: QUERY AND OUTPUT FOR THE CREATING ORDERED VIEW

```

460
461 •   SELECT * FROM MOVIE WHERE RELEASED_YEAR BETWEEN 2000 AND 2010;
462
463

```

Result Grid | Filter Rows: Edit: Export/Import: Wrap Cell Content:

MOVIE_ID	MOVIE_NAME	RELEASED_YEAR	GENRE	DURATION	M_LANGUAGE	BUDGET
MV003	The Dark Knight	2008	Action	02:32:00	English	200000000.00000
MV006	Inception	2010	Science Fiction	02:28:00	English	160000000.00000
*	NULL	NULL	NULL	NULL	NULL	NULL

FIGURE 74 : QUERY AND OUTPUT FOR THE SET OPERATION BETWEEN

462

463 • `SELECT MOVIE_NAME, GENRE FROM MOVIE;`

464

465

MOVIE_NAME	GENRE
The Shawshank Redemption	Drama
The Godfather	Drama, Crime
The Dark Knight	Action
Pulp Fiction	Crime
Forrest Gump	Drama
Inception	Science Fiction

FIGURE 75 : QUERY AND OUTPUT FOR THE PROJECT OPERATION

464

465 • `SELECT * FROM MOVIE, DIRECTOR;`

466

467

MOVIE_ID	MOVIE_NAME	RELEASED_YEAR	GENRE	DURATION	M_LANGUAGE	BUDGET	LICENSE_NO	DIRECTOR_NAME	AGE	ACTIVE_PERIO
MV001	The Shawshank Redemption	1994	Drama	02:22:00	English	25000000.00000	LIC005	Robert Zemeckis	70	40
MV001	The Shawshank Redemption	1994	Drama	02:22:00	English	25000000.00000	LIC004	Quentin Tarantino	58	30
MV001	The Shawshank Redemption	1994	Drama	02:22:00	English	25000000.00000	LIC003	Christopher Nolan	52	25
MV001	The Shawshank Redemption	1994	Drama	02:22:00	English	25000000.00000	LIC002	Francis Ford Coppola	83	50
MV001	The Shawshank Redemption	1994	Drama	02:22:00	English	25000000.00000	LIC001	Frank Darabont	62	35
MV002	The Godfather	1972	Drama, Crime	02:55:00	English	6000000.00000	LIC005	Robert Zemeckis	70	40
MV002	The Godfather	1972	Drama, Crime	02:55:00	English	6000000.00000	LIC004	Quentin Tarantino	58	30
MV002	The Godfather	1972	Drama, Crime	02:55:00	English	6000000.00000	LIC003	Christopher Nolan	52	25
MV002	The Godfather	1972	Drama, Crime	02:55:00	English	6000000.00000	LIC002	Francis Ford Coppola	83	50
MV002	The Godfather	1972	Drama, Crime	02:55:00	English	6000000.00000	LIC001	Frank Darabont	62	35
MV003	The Dark Knight	2008	Action	02:32:00	English	200000000.00000	LIC005	Robert Zemeckis	70	40
MV003	The Dark Knight	2008	Action	02:32:00	English	200000000.00000	LIC004	Quentin Tarantino	58	30
MV003	The Dark Knight	2008	Action	02:32:00	English	200000000.00000	LIC003	Christopher Nolan	52	25
MV003	The Dark Knight	2008	Action	02:32:00	English	200000000.00000	LIC002	Francis Ford Coppola	83	50
MV003	The Dark Knight	2008	Action	02:32:00	English	200000000.00000	LIC001	Frank Darabont	62	35
MV004	Pulp Fiction	1994	Crime	02:34:00	English	8000000.00000	LIC005	Robert Zemeckis	70	40
MV004	Pulp Fiction	1994	Crime	02:34:00	English	8000000.00000	LIC004	Quentin Tarantino	58	30
MV004	Pulp Fiction	1994	Crime	02:34:00	English	8000000.00000	LIC003	Christopher Nolan	52	25
MV004	Pulp Fiction	1994	Crime	02:34:00	English	8000000.00000	LIC002	Francis Ford Coppola	83	50
MV004	Pulp Fiction	1994	Crime	02:34:00	English	8000000.00000	LIC001	Frank Darabont	62	35
MV005	Forrest Gump	1994	Drama	02:22:00	English	55000000.00000	LIC005	Robert Zemeckis	70	40
MV005	Forrest Gump	1994	Drama	02:22:00	English	55000000.00000	LIC004	Quentin Tarantino	58	30
MV005	Forrest Gump	1994	Drama	02:22:00	English	55000000.00000	LIC003	Christopher Nolan	52	25
MV005	Forrest Gump	1994	Drama	02:22:00	English	55000000.00000	LIC002	Francis Ford Coppola	83	50
MV006	Inception	2010	Science Fiction	02:28:00	English	160000000.00000	LIC005	Robert Zemeckis	70	40
MV006	Inception	2010	Science Fiction	02:28:00	English	160000000.00000	LIC004	Quentin Tarantino	58	30
MV006	Inception	2010	Science Fiction	02:28:00	English	160000000.00000	LIC003	Christopher Nolan	52	25
MV006	Inception	2010	Science Fiction	02:28:00	English	160000000.00000	LIC002	Francis Ford Coppola	83	50
MV006	Inception	2010	Science Fiction	02:28:00	English	160000000.00000	LIC001	Frank Darabont	62	35

FIGURE 76 : QUERY AND OUTPUT FOR THE CARTESIAN PRODUCT OPERATION

```

466
467 • CREATE VIEW MovieDetails AS
468     SELECT MOVIE_NAME, GENRE, RELEASED_YEAR FROM MOVIE;
469
470 • SHOW FULL TABLES IN TVSEREIS_MOVIES_COLLECTION1 WHERE TABLE_TYPE LIKE 'VIEW';
471
472

```

Result Grid		Filter Rows:	Export:	Wrap Cell Content:
Tables_in_tvseries_movies_collection1	Table_type			
moviedetails	VIEW			

FIGURE 77 : QUERY, OUTPUT AND CHECK THE VIEW CREATION FOR THE CREATE VIEW OPERATION

Result Grid		Filter Rows:
Director	Director_Age	
Frank Darabont	62	
Francis Ford Coppola	83	
Christopher Nolan	52	
Quentin Tarantino	58	
Robert Zemeckis	70	

FIGURE 78 : QUERY AND OUTPUT FOR THE RENAMING OPERATION

Result Grid		Filter Rows:	Export
Avg_Rating			
4.6400000			

FIGURE 79 : QUERY AND OUTPUT FOR THE AVERAGE AGGREGATION FUNCTION

```

481 •   SELECT
482     MAX(RELEASED_YEAR)
483     AS Latest_Release
484     FROM MOVIE;
485

```

Result Grid | Filter Rows:

	Latest_Release
▶	2010

FIGURE 80 : QUERY AND OUTPUT FOR THE MAXIMUM AGGREGATION FUNCTION

```

485
486 •   SELECT *
487     FROM MOVIE
488     WHERE MOVIE_NAME LIKE '%Godfather%';
489

```

Result Grid | Filter Rows: Edit: Export/Import: Wrap Cell Content

MOVIE_ID	MOVIE_NAME	RELEASED_YEAR	GENRE	DURATION	M_LANGUAGE	BUDGET
MV002	The Godfather	1972	Drama, Crime	02:55:00	English	6000000.00000
NULL	NULL	NULL	NULL	NULL	NULL	NULL

FIGURE 81 : QUERY AND OUTPUT FOR THE LIKE KEYWORD

ii. Complex queries

MySQL Workbench

Local instance MySQL80 ×

File Edit View Query Database Server Tools Scripting Help

Navigator EE4202_Final* × director movie

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

```

626
627
628 -- Create a view for MOVIE and DIRECTOR tables
629 • CREATE VIEW MOVIE_DIRECTOR_VIEW AS
630 SELECT m.MOVIE_ID, m.MOVIE_NAME, m.RELEASED_YEAR, m.GENRE, m.DURATION, m.M_LANGUAGE, m.BUDGET, d.DIRECTOR_NAME, d.AGE, d.ACTIVE_PERIOD_IN_YEARS
631 FROM MOVIE m LEFT JOIN DIRECTOR d
632 ON m.MOVIE_ID = d.movie_id;
633
634
635

```

Result Grid | Form Editor | Field Types | Query Stats | Execution Plan

MOVIE_ID	MOVIE_NAME	RELEASED_YEAR	GENRE	DURATION	M_LANGUAGE	BUDGET	DIRECTOR_NAME	AGE	ACTIVE_PERIOD_IN_YEARS
MV002	The Godfather	1972	Drama, Crime	02:55:00	English	60000000.0000	Francis Ford Coppola	83	50
MV003	The Dark Knight	2008	Action	02:32:00	English	200000000.0000	Christopher Nolan	52	25
MV006	Inception	2010	Science Fiction	02:28:00	English	160000000.0000	HULL	HULL	HULL
MV012	Gladiator	2000	Action	02:35:00	English	103000000.0000	HULL	HULL	HULL
MV013	Goodfellas	1990	Biography	02:26:00	English	25000000.0000	HULL	HULL	HULL
MV014	The Departed	2006	Crime	02:31:00	English	90000000.0000	HULL	HULL	HULL
MV015	The Pianist	2002	Biography	02:30:00	English	35000000.0000	HULL	HULL	HULL
MV017	Saving Private Ryan	1998	Drama	02:49:00	English	70000000.0000	HULL	HULL	HULL
MV018	The Godfather: Part II	1974	Crime	03:22:00	English	13000000.0000	HULL	HULL	HULL
MV019	The Dark Knight Rises	2012	Action	02:44:00	English	250000000.0000	HULL	HULL	HULL
MV024	The Terminator	1984	Action	01:47:00	English	64000000.0000	HULL	HULL	HULL
MV026	Avatar	2009	Action	02:42:00	English	237000000.0000	HULL	HULL	HULL
MV027	Titanic	1997	Romance	03:15:00	English	200000000.0000	HULL	HULL	HULL
MV029	The Shining	1980	Horror	02:26:00	English	19000000.0000	HULL	HULL	HULL
MV030	Jaws	1975	Thriller	02:04:00	English	70000000.0000	HULL	HULL	HULL
MV031	Casablanca	1942	Drama	01:42:00	English	9500000.0000	HULL	HULL	HULL
MV034	Citizen Kane	1941	Drama	01:59:00	English	839727.0000	HULL	HULL	HULL
MV035	Psycho	1960	Horror	01:49:00	English	806947.0000	HULL	HULL	HULL
MV036	The Graduate	1967	Drama	01:46:00	English	3800000.0000	HULL	HULL	HULL

MOVIE_DIRECTOR_VIEW 40 ×

Result Grid | Form Editor | Field Types | Query Stats | Execution Plan

Object Info Session Output

85°F Light rain

Search

File Edit View Query Database Server Tools Scripting Help

MySQL Workbench

Local instance MySQL80 ×

File Edit View Query Database Server Tools Scripting Help

Navigator EE4202_Final* × director movie

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

```

636 -- Create a view for TV_SERIES and EPISODE tables
637 • CREATE VIEW TV_SERIES_EPISODE_VIEW AS
638 SELECT ts.TV_SERIES_ID, ts.TV_SERIES_NAME, ts.RELEASED_YEAR, ts.GENRE, ts.TS_LANGUAGE, ts.NO_OF_SEASONS, e.TITLE, e.RATINGS, e.EPISODE_LENGTH
639 FROM TV_SERIES ts LEFT JOIN EPISODE e
640 ON ts.TV_SERIES_ID = e.tv_series_id;
641
642 • SELECT * FROM TV_SERIES_EPISODE_VIEW;

```

Result Grid | Form Editor | Field Types | Query Stats | Execution Plan

TV_SERIES_ID	TV_SERIES_NAME	RELEASED_YEAR	GENRE	TS_LANGUAGE	NO_OF_SEASONS	TITLE	RATINGS	EPISODE_LENGTH
TS001	Breaking Bad	2008	Drama	English	6	HULL	HULL	HULL
TS002	Game of Thrones	2013	Fantasy	English	8	Winter Is Coming	8.00	00:58:00
TS003	Friends	1994	Comedy	English	10	HULL	HULL	HULL
TS004	Stranger Things	2016	Science Fiction	English	4	Chapter of Byers	8.80	00:48:00
TS005	The Crown	2016	Historical Drama	English	4	Wolferton Splash	8.70	00:58:00
TS007	Series A	2010	Drama	English	5	HULL	HULL	HULL
TS008	Series B	2015	Thriller	English	3	HULL	HULL	HULL
TS009	Series C	2018	Comedy	English	6	HULL	HULL	HULL
TS010	Series D	2013	Sci-Fi	English	4	HULL	HULL	HULL
TS011	Series E	2011	Fantasy	English	8	HULL	HULL	HULL
TS012	Series F	2017	Drama	English	4	HULL	HULL	HULL
TS013	Series G	2016	Mystery	English	5	HULL	HULL	HULL
TS014	Series H	2014	Action	English	7	HULL	HULL	HULL
TS015	Series I	2019	Horror	English	3	HULL	HULL	HULL
TS016	Series J	2012	Romance	English	6	HULL	HULL	HULL
TS017	Series K	2018	Thriller	English	4	HULL	HULL	HULL
TS018	Series L	2015	Sci-Fi	English	5	HULL	HULL	HULL
TS019	Series M	2016	Comedy	English	8	HULL	HULL	HULL
TS020	Series N	2013	Drama	English	6	HULL	HULL	HULL
TS021	Series O	2017	Fantasy	English	5	HULL	HULL	HULL
TS022	Series P	2014	Mystery	English	7	HULL	HULL	HULL
TS023	Series Q	2011	Action	English	4	HULL	HULL	HULL
TS024	Series R	2019	Horror	English	3	HULL	HULL	HULL

TV_SERIES_EPISODE_VIEW 41 ×

Result Grid | Form Editor | Field Types | Query Stats | Execution Plan

Object Info Session Output

Breaking news Trump says abor...

Search

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator: EE4202_Final* director movie

SCHEMAS: tvseries_movies_collection

Tables: country, director, episode, main_actor, movie, name_of_the_studio, name_of_the_website, review, song, studio, tv_series

Views, Stored Procedures, Functions

SQL Editor:

```

48 -- Set difference of TV series genres between australia and drama
49 • SELECT TV_SERIES_NAME, GENRE
50 FROM TV_SERIES
51 WHERE country_name = 'Australia'
52 □ EXCEPT
53 SELECT TV_SERIES_NAME, GENRE
54 FROM TV_SERIES
55 WHERE GENRE = 'Drama';
56
57
58

```

Information:

Table: episode

Columns: TITLE, RATING, EPISODE_LENGTH, tv_series_id, COST_PER_EPISODE

Result Grid:

TV_SERIES_NAME	GENRE
Series D	Sci-Fi
Series I	Horror
Series S	Romance

Object Info Session

Result 42 x Output

Result Grid Form Editor Field Types Query Stats

Context Help Snippets

Read Only

7:15 PM 4/8/2024

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator: EE4202_Final* director movie

SCHEMAS: tvseries_movies_collection

Tables: country, director, episode, main_actor, movie, name_of_the_studio, name_of_the_website, review, song, studio, tv_series

Views, Stored Procedures, Functions

SQL Editor:

```

652
653 -- Division operation: Find TV series with all episodes
654 • SELECT TV_SERIES_ID, TV_SERIES_NAME
655 FROM TV_SERIES
656 WHERE TV_SERIES_ID
657 NOT IN
658 (
659   SELECT tv_series_id
660   FROM EPISODE
661   WHERE tv_series_id IS NULL
662

```

Information:

Table: episode

Columns: TITLE, RATING, EPISODE_LENGTH, tv_series_id, COST_PER_EPISODE

Result Grid:

TV_SERIES_ID	TV_SERIES_NAME
TS001	Breaking Bad
TS002	Game of Thrones
TS003	Friends
TS004	Stranger Things
TS005	The Crown
TS007	Series A
TS008	Series B
TS009	Series C
TS010	Series D
TS011	Series E
TS012	Series F
TS013	Series G
TS014	Series H
TS015	Series I
TS016	Series J
TS017	Series K

Object Info Session

Result 43 x Output

Result Grid Form Editor Field Types Query Stats

Context Help Snippets

Apply Revert

7:19 PM 4/8/2024

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator: EE4202_Final* director movie

SCHEMAS: tvseries_movies_collection

Tables: country, director, episode, main_actor, movie, name_of_the_studio, name_of_the_website, review, song, studio, tv_series

Views, Stored Procedures, Functions

Information: Table: episode

Columns:

- TITLE: va PK
- RATINGS: do
- EPISODE_LENGTH: tn
- tv_series_id**: va PK
- COST_PER_EPISODE: do

```

635 -- Create a view for TV_SERIES and EPISODE tables
636
637 • CREATE VIEW TV_SERIES_EPISODE_VIEW AS
638   SELECT ts.TV_SERIES_ID,ts.TV_SERIES_NAME, ts.RELEASED_YEAR,ts.GENRE, ts.TS_LANGUAGE, ts.NO_OF_SEASONS, e.TITLE, e.RATINGS, e.EPISODE_LENGTH
639   FROM TV_SERIES ts LEFT JOIN EPISODE e
640   ON ts.TV_SERIES_ID = e.tv_series_id;
641
642 •   SELECT * FROM TV_SERIES_EPISODE_VIEW;
643
644 -- Set difference of TV series genres between australia and drama
645 •   SELECT TV_SERIES_NAME, GENRE
646   FROM TV_SERIES
647   WHERE country_name = 'Australia'
648 □ EXCEPT
649   SELECT TV_SERIES_NAME, GENRE
650   FROM TV_SERIES
651   WHERE GENRE = 'Drama';
652
653 -- Division operation: Find TV series with all episodes
654 •   SELECT TV_SERIES_ID, TV_SERIES_NAME
655   FROM TV_SERIES
656   WHERE TV_SERIES_ID
657     NOT IN
658   (
659     SELECT tv_series_id
660     FROM EPISODE
661     WHERE tv_series_id IS NULL
662   );
663
664

```

Object Info Session Output

82°F Light rain Search

7:23 PM 4/8/2024

SQLAdditions: Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator: EE4202_Final* director movie tv_series

SCHEMAS: tvseries_movies_collection

Tables: country, director, episode, main_actor, movie, name_of_the_studio, name_of_the_website, review, song, studio, tv_series

Views, Stored Procedures, Functions

Information: Table: episode

Columns:

- TITLE: va PK
- RATINGS: do
- EPISODE_LENGTH: tn
- tv_series_id**: va PK
- COST_PER_EPISODE: do

```

563
564
565 -- Inner join between MOVIE_DIRECTOR_VIEW and TV_SERIES_EPISODE_VIEW
566 •   SELECT mv.MOVIE_NAME, mv.GENRE, mv.RELEASED_YEAR, te.TITLE, te.RATINGS
567   FROM MOVIE_DIRECTOR_VIEW mv INNER JOIN TV_SERIES_EPISODE_VIEW te
568   ON mv.RELEASED_YEAR = te.RELEASED_YEAR;
569
570
571
572
573
574

```

Result Grid | Filter Rows: Export: Wrap Cell Content: □

MOVIE_NAME	GENRE	RELEASED_YEAR	TITLE	RATINGS
The Dark Knight	Action	2008	NULL	NULL
Inception	Science Fiction	2010	NULL	NULL
The Dark Knight Rises	Action	2012	NULL	NULL

Result 45 x

Object Info Session Output

82°F Light rain Search

7:29 PM 4/8/2024

SQLAdditions: Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator: EE4202_Final* | director movie tv_series

SQLAdditions: Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Result Grid | Filter Rows: Export: Wrap Cell Content: Result Grid | Form Editor | Field Types | Query Stats | Execution Plan | Read Only | Context Help | Snippets

Table: episode

Columns:

TITLE	PK
RATINGS	PK
EPISODE_LENGTH	DO
tv_series_id	VA
COST_PER_EPISODE	PK

Object Info Session

Output:

Action Output | # Time Action | Message | Duration / Fetch

92 19:36:11 explain SELECT mv.MOVIE_NAME, mv.GENRE, mv.RELEASED_YEAR, te.TITLE, te.RATINGS FROM MOVIE mv LEFT OUTER JOIN TV_SERIES_EPISODE_VIEW te ON mv.MOVIE_ID = te.movie_id; 4 row(s) returned 0.000 sec / 0.000 sec

93 19:37:38 SELECT TV_SERIES_NAME, RELEASED_YEAR, GENRE, NO_OF_SEASONS, TITLE, RATINGS FROM TV_SERIES_EPISODE_VIEW NATURAL JOIN TV_SERIES; 25 row(s) returned 0.047 sec / 0.000 sec

82°F Light rain 7:37 PM 4/8/2024

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator: EE4202_Final* | director movie tv_series

SQLAdditions: Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Result Grid | Filter Rows: Export: Wrap Cell Content: Result Grid | Form Editor | Field Types | Query Stats | Execution Plan | Read Only | Context Help | Snippets

81

82 -- Right outer join between MOVIE_DIRECTOR_VIEW and DIRECTOR table

83 • SELECT mv.MOVIE_NAME, mv.GENRE, mv.RELEASED_YEAR, md.DIRECTOR_NAME

84 FROM MOVIE_DIRECTOR_VIEW mv

85 RIGHT OUTER JOIN DIRECTOR md ON mv.MOVIE_ID = md.movie_id;

86

87

88

Result Grid | Filter Rows: Export: Wrap Cell Content: Result Grid | Form Editor | Field Types | Query Stats | Execution Plan | Read Only | Context Help | Snippets

MOVIE_NAME | GENRE | RELEASED_YEAR | DIRECTOR_NAME

MOVIE_NAME	GENRE	RELEASED_YEAR	DIRECTOR_NAME
HULL	HULL	HULL	Frank Darabont
The Godfather	Drama, Crime	1972	Francis Ford Coppola
The Dark Knight	Action	2008	Christopher Nolan
			Quentin Tarantino
			Robert Zemeckis

Output:

Action Output | # Time Action | Message | Duration / Fetch

113 19:59:25 explain SELECT ma.FIRST_NAME, ma.LAST_NAME, m.MOVIE_NAME FROM MAIN_ACTOR_MOVIE_VIEW ma LEFT OUTER JOIN MOVIE m ON ma.movie_id = m.movie_id; 3 row(s) returned 0.000 sec / 0.000 sec

114 20:06:00 SELECT mv.MOVIE_NAME, mv.GENRE, mv.RELEASED_YEAR, md.DIRECTOR_NAME FROM MOVIE_DIRECTOR_VIEW mv RIGHT OUTER JOIN DIRECTOR md ON mv.MOVIE_ID = md.movie_id; 5 row(s) returned 0.015 sec / 0.000 sec

83°F Mostly cloudy 8:06 PM 4/8/2024

MySQL Workbench

Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

EE4202_Final* | director movie tv_series

SQLAdditions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

```

693
694 • SELECT
695     COALESCE(te.TV_SERIES_NAME, ts.TV_SERIES_NAME) AS TV_SERIES_NAME,
696     COALESCE(te.RELEASED_YEAR, ts.RELEASED_YEAR) AS RELEASED_YEAR,
697     COALESCE(ts.GENRE, te.GENRE) AS GENRE,
698     ts.NO_OF_SEASONS,
699     te.TITLE,
700     te.RATINGS
701
702     FROM
703     TV_SERIES_EPISODE_VIEW te
704     LEFT JOIN
705     TV_SERIES ts ON te.TV_SERIES_ID = ts.TV_SERIES_ID
706     UNION
707     SELECT
708         COALESCE(te.TV_SERIES_NAME, ts.TV_SERIES_NAME) AS TV_SERIES_NAME,
709         COALESCE(te.RELEASED_YEAR, ts.RELEASED_YEAR) AS RELEASED_YEAR,
710         COALESCE(ts.GENRE, te.GENRE) AS GENRE,
711         ts.NO_OF_SEASONS,
712         NULL AS TITLE,
713         NULL AS RATINGS
714
715     FROM
716     TV_SERIES ts
717     RIGHT JOIN
718     TV_SERIES_EPISODE_VIEW te ON te.TV_SERIES_ID = ts.TV_SERIES_ID;

```

Result Grid | Filter Rows: Export: Wrap Cell Content:

TV_SERIES_NAME	RELEASED_YEAR	GENRE	NO_OF_SEASONS	TITLE	RATINGS
Breaking Bad	2008	Drama	6	NULL	NULL
Game of Thrones	2013	Fantasy	8	Winter Is Coming	8.00
Friends	1994	Comedy	10	NULL	NULL
Stranger Things	2016	Science Fiction	4	Chapter of Byers	8.80
The Crown	2016	Historical Drama	4	Wolferton Splash	8.70
Series A	2010	Drama	5	NULL	NULL
Series B	2015	Thriller	3	NULL	NULL
Series C	2018	Comedy	6	NULL	NULL
Series D	2013	Sci-Fi	4	NULL	NULL

Result 66 x

Output

83°F Mostly cloudy

8:13 PM 4/8/2024

```

490 -- Create a view for MOVIE and DIRECTOR tables
491 • CREATE VIEW MOVIE_DIRECTOR_VIEW AS
492     SELECT m.MOVIE_ID, m.MOVIE_NAME, m.RELEASED_YEAR, m.GENRE, m.DURATION, m.M_LANGUAGE, m.BUDGET, d.DIRECTOR_NAME, d.AGE, d.ACTIVE_PERIOD_IN_YEARS
493     FROM MOVIE m
494     LEFT JOIN DIRECTOR d ON m.MOVIE_ID = d.movie_id;
495
496 -- Create a view for TV_SERIES and EPISODE tables
497 • CREATE VIEW TV_SERIES_EPISODE_VIEW AS
498     SELECT ts.TV_SERIES_ID, ts.TV_SERIES_NAME, ts.RELEASED_YEAR, ts.GENRE, ts.TS_LANGUAGE, ts.NO_OF_SEASONS, e.TITLE, e.RATINGS, e.EPISODE_LENGTH
499     FROM TV_SERIES ts
500     LEFT JOIN EPISODE e ON ts.TV_SERIES_ID = e.tv_series_id;
501
502 -- Create a view for MAIN_ACTOR and MOVIE tables
503 • CREATE VIEW MAIN_ACTOR_MOVIE_VIEW AS
504     SELECT ma.ACTOR_LICENSE_NO, ma.FIRST_NAME, ma.LAST_NAME, ma.NATIONALITY, ma.COUNTRY, ma.DATE_OF_BIRTH, ma.ACTIVE_PERIOD, ma.AWARDS, m.MOVIE_NAME
505     FROM MAIN_ACTOR ma
506     LEFT JOIN MOVIE m ON ma.movie_id = m.MOVIE_ID;
507
508 • SHOW FULL TABLES IN TVSEREIS_MOVIES_COLLECTION1 WHERE TABLE_TYPE LIKE 'VIEW';
509
510

```

Result Grid | Filter Rows: Export: Wrap Cell Content:

Tables_in_tvseries_movies_collection1	Table_type
main_actor_movie_view	VIEW
movie_director_view	VIEW
moviedetails	VIEW
tv_series_episode_view	VIEW

FIGURE 82 : COMPLEX QUERY AND OUTPUT FOR THE CREATING VIEW AND CHECK THE VIEWS

```

509
510      -- Union of two sets of TV series from different countries
511 •   SELECT TV_SERIES_NAME, RELEASED_YEAR, GENRE, TS_LANGUAGE
512     FROM TV_SERIES
513     WHERE country_name = 'USA'
514     UNION
515     SELECT TV_SERIES_NAME, RELEASED_YEAR, GENRE, TS_LANGUAGE
516     FROM TV_SERIES
517     WHERE country_name = 'UK';
518

```

Result Grid | Filter Rows: Export: Wrap Cell Content

	TV_SERIES_NAME	RELEASED_YEAR	GENRE	TS_LANGUAGE
▶	Breaking Bad	2008	Drama	English
	Game of Thrones	2013	Fantasy	English
	Friends	1994	Comedy	English
	Stranger Things	2016	Science Fiction	English
	The Crown	2016	Historical Drama	English

FIGURE 83 : COMPLEX QUERY AND OUTPUT FOR SET OPERATION UNION

```

519      -- Intersection of TV series genres between two different countries
520 •   SELECT TV_SERIES_NAME, GENRE
521     FROM TV_SERIES
522     WHERE country_name = 'USA'
523 ✘ INTERSECT
524     SELECT TV_SERIES_NAME, GENRE
525     FROM TV_SERIES
526     WHERE country_name = 'UK';
527

```

Result Grid | Filter Rows: Export: Wrap Cell Content:

	TV_SERIES_NAME	GENRE

**It is showing that
INTERSECT is not
valid for this point**

FIGURE 84 : COMPLEX QUERY AND OUTPUT FOR SET OPERATION INTERSECT

```

528      -- Set difference of TV series genres between USA and UK
529 •   SELECT TV_SERIES_NAME, GENRE
530     FROM TV_SERIES
531     WHERE country_name = 'USA'
532 ✘   EXCEPT
533     SELECT TV_SERIES_NAME, GENRE
534     FROM TV_SERIES
535     WHERE country_name = 'UK';
536

```

**It is showing that
EXCEPT is not valid
for this point**

Result Grid		Filter Rows:	Export:	Wrap Cell
	TV_SERIES_NAME	GENRE		
▶	Breaking Bad	Drama		
	Game of Thrones	Fantasy		
	Friends	Comedy		
	Stranger Things	Science Fiction		

FIGURE 85 : COMPLEX QUERY AND OUTPUT FOR SET OPERATION EXCEPT

```

536
537      -- Division operation: Find TV series with all episodes
538 •   SELECT TV_SERIES_ID, TV_SERIES_NAME
539     FROM TV_SERIES
540     WHERE TV_SERIES_ID NOT IN (
541       SELECT tv_series_id
542         FROM EPISODE
543        WHERE tv_series_id IS NULL
544     );
545
546
547

```

Result Grid		Filter Rows:	Edit:
	TV_SERIES_ID	TV_SERIES_NAME	
▶	TS001	Breaking Bad	
	TS002	Game of Thrones	
	TS003	Friends	
	TS004	Stranger Things	
	TS005	The Crown	
*	NULL	NULL	

FIGURE 86: COMPLEX QUERY AND OUTPUT FOR DIVISION OPERATION

```

546      -- Nested query with set difference
547  ●   SELECT TV_SERIES_NAME
548    FROM TV_SERIES
549    WHERE TV_SERIES_ID IN (
550        SELECT TV_SERIES_ID
551        FROM EPISODE
552        WHERE RATINGS > 9.0
553    );
554

```

| Result Grid | Filter Rows: _____ | Export: | V
| TV_SERIES_NAME |

FIGURE 87: COMPLEX QUERY AND OUTPUT FOR NESTED QUERY WITH SET DIFFERENT

```

555      -- Multiset comparison with correlated nested query
556  ●   SELECT MOVIE_NAME
557    FROM MOVIE m
558    WHERE EXISTS (
559        SELECT 1
560        FROM DIRECTOR d
561        WHERE d.movie_id = m.MOVIE_ID AND d.ACTIVE_PERIOD_IN_YEARS > 30
562    );
563

```

| Result Grid | Filter Rows: _____ | Export: | Wrap Cell Content: | V
| MOVIE_NAME |
▶ The Shawshank Redemption
The Godfather
Forrest Gump

FIGURE 88 : COMPLEX QUERY AND OUTPUT FOR MULTISET COMPARISON WITH CORRELATED NESTED QUERY

```
564      -- Correlated nested query with set difference
565 •   SELECT TV_SERIES_NAME
566     FROM TV_SERIES ts
567     WHERE ts.TV_SERIES_ID NOT IN (
568       SELECT tv_series_id
569         FROM EPISODE e
570        WHERE e.RATINGS > (
571          SELECT AVG(RATINGS)
572            FROM EPISODE
573        )
574    );
575
```

Result Grid | Filter Rows: Export:

TV_SERIES_NAME
▶ Breaking Bad
Game of Thrones
Friends

FIGURE 89 : COMPLEX QUERY AND OUTPUT FOR CORRELATED NESTED QUERY WITH SET DIFFERENCE

```

576      -- Recursive query to find all TV series and their episodes
577  • WITH RECURSIVE TVSeries_CTE AS (
578      SELECT TV_SERIES_ID, TV_SERIES_NAME
579      FROM TV_SERIES
580      UNION ALL
581      SELECT e.tv_series_id, e.TITLE
582      FROM TVSeries_CTE t
583      JOIN EPISODE e ON t.TV_SERIES_ID = e.tv_series_id
584  )
585      SELECT *
586      FROM TVSeries_CTE;

```

FIGURE 90 : COMPLEX QUERY FOR RECURSION

```

546      -- Inner join between MOVIE_DIRECTOR_VIEW and TV_SERIES_EPISODE_VIEW
547  •   SELECT mv.MOVIE_NAME, mv.GENRE, mv.RELEASED_YEAR, te.TITLE, te.RATINGS
548      FROM MOVIE_DIRECTOR_VIEW mv
549      INNER JOIN TV_SERIES_EPISODE_VIEW te ON mv.MOVIE_ID = te.TV_SERIES_ID;
550

```

MOVIE_NAME	GENRE	RELEASED_YEAR	TITLE	RATINGS

FIGURE 91 : COMPLEX QUERY FOR INNER JOIN

```

550
551      -- Natural join between TV_SERIES_EPISODE_VIEW and TV_SERIES table
552  •   SELECT TV_SERIES_NAME, RELEASED_YEAR, GENRE, NO_OF_SEASONS, TITLE, RATINGS
553      FROM TV_SERIES_EPISODE_VIEW
554      NATURAL JOIN TV_SERIES;

```

TV_SERIES_NAME	RELEASED_YEAR	GENRE	NO_OF_SEASONS	TITLE	RATINGS
Breaking Bad	2008	Drama	6	NULL	NULL
Game of Thrones	2013	Fantasy	8	Winter Is Coming	8.00
Friends	1994	Comedy	10	NULL	NULL
Stranger Things	2016	Science Fiction	4	Chapter of Byers	8.80
The Crown	2016	Historical Drama	4	Wolferton Splash	8.70

FIGURE 92 : COMPLEX QUERY FOR NATURAL JOIN

```

555
556      -- Left outer join between MAIN_ACTOR_MOVIE_VIEW and MOVIE table
557 •   SELECT ma.FIRST_NAME, ma.LAST_NAME, m.MOVIE_NAME
558     FROM MAIN_ACTOR_MOVIE_VIEW ma
559     LEFT OUTER JOIN MOVIE m ON ma.MOVIE_NAME = m.MOVIE_NAME;
560

```

FIRST_NAME	LAST_NAME	MOVIE_NAME
Morgan	Freeman	The Shawshank Redemption
Marlon	Brando	The Godfather
Christian	Bale	The Dark Knight
Tom	Hanks	Forrest Gump
Leonardo	DiCaprio	Inception

FIGURE 93 : COMPLEX QUERY FOR LEFT OUTER JOIN

```

560
561      -- Right outer join between MOVIE_DIRECTOR_VIEW and DIRECTOR table
562 •   SELECT mv.MOVIE_NAME, mv.GENRE, mv.RELEASED_YEAR, md.DIRECTOR_NAME
563     FROM MOVIE_DIRECTOR_VIEW mv
564     RIGHT OUTER JOIN DIRECTOR md ON mv.MOVIE_ID = md.movie_id;
565

```

MOVIE_NAME	GENRE	RELEASED_YEAR	DIRECTOR_NAME
The Shawshank Redemption	Drama	1994	Frank Darabont
The Godfather	Drama, Crime	1972	Francis Ford Coppola
The Dark Knight	Action	2008	Christopher Nolan
Pulp Fiction	Crime	1994	Quentin Tarantino
Forrest Gump	Drama	1994	Robert Zemeckis

FIGURE 94 : COMPLEX QUERY FOR RIGHT OUTER JOIN

```

566      -- Full outer join between TV_SERIES_EPISODE_VIEW and TV_SERIES table
567 •   SELECT COALESCE(te.TV_SERIES_NAME, ts.TV_SERIES_NAME) AS TV_SERIES_NAME, te.RELEASED_YEAR, ts.GENRE, te.NO_OF_SEASONS, te.TITLE, te.RATINGS
568     FROM TV_SERIES_EPISODE_VIEW te
569     FULL OUTER JOIN TV_SERIES ts ON te.TV_SERIES_ID = ts.TV_SERIES_ID;

```

FIGURE 96 : COMPLEX QUERY FOR FULL OUTER JOIN

But it's not supporting

```

571 •      SELECT
572         COALESCE(te.TV_SERIES_NAME, ts.TV_SERIES_NAME) AS TV_SERIES_NAME,
573         COALESCE(te.RELEASED_YEAR, ts.RELEASED_YEAR) AS RELEASED_YEAR,
574         COALESCE(ts.GENRE, te.GENRE) AS GENRE,
575         te.NO_OF_SEASONS,
576         te.TITLE,
577         te.RATINGS
578     FROM
579         TV_SERIES_EPISODE_VIEW te
580     LEFT JOIN
581         TV_SERIES ts ON te.TV_SERIES_ID = ts.TV_SERIES_ID
582     UNION
583     SELECT
584         COALESCE(te.TV_SERIES_NAME, ts.TV_SERIES_NAME) AS TV_SERIES_NAME,
585         COALESCE(te.RELEASED_YEAR, ts.RELEASED_YEAR) AS RELEASED_YEAR,
586         COALESCE(ts.GENRE, te.GENRE) AS GENRE,
587         ts.NO_OF_SEASONS,
588         NULL AS TITLE,
589         NULL AS RATINGS
590     FROM
591         TV_SERIES ts
592     RIGHT JOIN
593         TV_SERIES_EPISODE_VIEW te ON te.TV_SERIES_ID = ts.TV_SERIES_ID;
594

```

Result Grid		Filter Rows:	Export:		Wrap Cell Content:	
	TV_SERIES_NAME	RELEASED_YEAR	GENRE	NO_OF_SEASONS	TITLE	RATINGS
▶	Breaking Bad	2008	Drama	6	NULL	NULL
	Game of Thrones	2013	Fantasy	8	Winter Is Coming	8.00
	Friends	1994	Comedy	10	NULL	NULL
	Stranger Things	2016	Science Fiction	4	Chapter of Byers	8.80
	The Crown	2016	Historical Drama	4	Wolferton Splash	8.70
	Game of Thrones	2013	Fantasy	8	NULL	NULL
	Stranger Things	2016	Science Fiction	4	NULL	NULL
	The Crown	2016	Historical Drama	4	NULL	NULL

FIGURE 97 : COMPLEX QUERY FOR UPDATED FULL OUTER JOIN

```

596      -- Outer union between MOVIE_DIRECTOR_VIEW and DIRECTOR table
597  ●   SELECT mv.MOVIE_NAME, mv.GENRE, mv.RELEASED_YEAR, md.DIRECTOR_NAME
598      FROM MOVIE_DIRECTOR_VIEW mv
599  ✘   OUTER UNION
600      SELECT NULL, NULL, NULL, DIRECTOR_NAME
601      FROM DIRECTOR;
602
603  ●   SELECT mv.MOVIE_NAME, mv.GENRE, mv.RELEASED_YEAR, mv.DIRECTOR_NAME
604      FROM MOVIE_DIRECTOR_VIEW mv
605      UNION
606      SELECT NULL, NULL, NULL, d.DIRECTOR_NAME
607      FROM DIRECTOR d
608      LEFT JOIN MOVIE_DIRECTOR_VIEW mv ON d.DIRECTOR_NAME = mv.DIRECTOR_NAME;
609

```

**THIS IS NOT
SUPPORTING**

UPDATED ONE

	MOVIE_NAME	GENRE	RELEASED_YEAR	DIRECTOR_NAME
▶	The Shawshank Redemption	Drama	1994	Frank Darabont
	The Godfather	Drama, Crime	1972	Francis Ford Coppola
	The Dark Knight	Action	2008	Christopher Nolan
	Pulp Fiction	Crime	1994	Quentin Tarantino
	Forrest Gump	Drama	1994	Robert Zemeckis
	Inception	Science Fiction	2010	NULL
	NULL	NULL	NULL	Frank Darabont
	NULL	NULL	NULL	Francis Ford Coppola
	NULL	NULL	NULL	Christopher Nolan
	NULL	NULL	NULL	Quentin Tarantino
	NULL	NULL	NULL	Robert Zemeckis

FIGURE 98 : COMPLEX QUERY FOR OUTER UNION (UPDATED AND NON-SUPPORTED FORMAT)

iii. Comparison of complexity of the queries before and after inserting the index

The screenshot shows the MySQL Workbench interface with the following details:

- File Tab:** Local instance MySQL80
- Schemas:** tvseries_movies_collection
- Tables:** country, director, episode, main_actor, movie, name_of_the_studio, name_of_the_website, review
- Indexes:** PRIMARY, fk12
- Triggers:** song, studio, tv_series
- Views:** None
- Stored Procedures:** None
- Functions:** None
- Information:** Table: episode, Columns: TITLE, RATINGS, EPISODE_LENGTH, tv_series_id, COST_PER_EPISODE
- Object Info:** Session
- System Tray:** 90°F Partly sunny
- Toolbar:** Standard MySQL icons
- SQL Editor:**

```

91 • EXPLAIN
92 SELECT
93 DIRECTOR_NAME AS Director,
94 AGE AS Director_Age
95 FROM DIRECTOR
96 where AGE = 20;
97
  
```
- Result Grid:**

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	DIRECTOR	NULL	ALL	NULL	NULL	NULL	NULL	5	20.00	Using where
- Output:**

#	Time	Action	Message	Duration / Fetch
23	14:35:44	drop index RATINGS_INDEX on review	Error Code: 1091. Can't DROP 'RATINGS_INDEX'; check that column/key exists	0.000 sec
24	15:11:29	SELECT DIRECTOR_NAME AS Director, AGE AS Director_Age FROM DIRECTOR where AGE > 20 LIMIT ...	1 row(s) returned	0.000 sec / 0.000 sec
25	15:11:51	SELECT *FROM tvseries_movies_collection1director LIMIT 0, 1000	5 row(s) returned	0.000 sec / 0.000 sec
26	15:12:16	drop index Director_Age_index on DIRECTOR	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.047 sec
27	15:12:28	EXPLAIN SELECT DIRECTOR_NAME AS Director, AGE AS Director_Age FROM DIRECTOR where AGE = ...	1 row(s) returned	0.000 sec / 0.000 sec
- System Bar:** 3:12 PM 4/8/2024

FIGURE 100 : COMPLEXITY OF THE QUERY BEFORE ADDING THE DIERCTOR_AGE INDEX

The screenshot shows the MySQL Workbench interface with the following details:

- File Tab:** Local instance MySQL80
- Schemas:** tvseries_movies_collection
- Tables:** country, director, episode, main_actor, movie, name_of_the_studio, name_of_the_website, review
- Indexes:** PRIMARY, fk12
- Triggers:** song, studio, tv_series
- Views:** None
- Stored Procedures:** None
- Functions:** None
- Information:** Table: episode, Columns: TITLE, RATINGS, EPISODE_LENGTH, tv_series_id, COST_PER_EPISODE
- Object Info:** Session
- System Tray:** 90°F Partly sunny
- Toolbar:** Standard MySQL icons
- SQL Editor:**

```

597
598 • create unique index Director_Age_index using BTREE on DIRECTOR(AGE);
599 • EXPLAIN
600 SELECT
601 DIRECTOR_NAME AS Director,
602 AGE AS Director_Age
603 FROM DIRECTOR
604 where AGE = 20;
  
```
- Result Grid:**

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	DIRECTOR	NULL	const	Director_Age_index	Director_Age_index	5	const	1	100.00	NULL
- Output:**

#	Time	Action	Message	Duration / Fetch
28	15:14:32	create unique index Director_Age_index using BTREE on DIRECTOR(AGE)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.076 sec
29	15:14:43	EXPLAIN SELECT DIRECTOR_NAME AS Director, AGE AS Director_Age FROM DIRECTOR	1 row(s) returned	0.000 sec / 0.000 sec
30	15:15:26	create unique index Director_Age_index using BTREE on DIRECTOR(AGE)	Error Code: 1061. Duplicate key name 'Director_Age_index'	0.000 sec
31	15:15:36	EXPLAIN SELECT DIRECTOR_NAME AS Director, AGE AS Director_Age FROM DIRECTOR	1 row(s) returned	0.000 sec / 0.000 sec
32	15:16:10	EXPLAIN SELECT DIRECTOR_NAME AS Director, AGE AS Director_Age FROM DIRECTOR where AGE = ...	1 row(s) returned	0.000 sec / 0.000 sec
- System Bar:** 3:16 PM 4/8/2024

FIGURE 99 : COMPLEXITY OF THE QUERY AFTER ADDING THE DIERCTOR_AGE INDEX

The screenshot shows the MySQL Workbench interface with the following details:

- Schemas:** tvseries_movies_collection
- Tables:** country, director, episode, main_actor, movie, name_of_the_studio, name_of_the_website, review, song, studio, tv_series.
- SQL Editor:** EE4202_Final* - SQL File 11


```

615 -- selecting the minimum released year by renaming as latest_release
616 • SELECT
617 min(Released_Year)
618 AS Latest_Release
619 FROM MOVIE;
620
621 • explain
622 SELECT
623 min(Released_Year)
624 AS Latest_Release
625 FROM MOVIE;
      
```
- Result Grid:** Shows 19 rows returned.
- Output Window:**

#	Time	Action	Message	Duration / Fetch
35	15:34:18	SELECT min(Released_Year) AS Latest_Release FROM MOVIE LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec
36	15:34:25	SELECT * FROM tvseries_movies_collection1.movie LIMIT 0, 1000	19 row(s) returned	0.000 sec / 0.000 sec
37	15:35:36	explain SELECT min(Released_Year) AS Latest_Release FROM MOVIE	1 row(s) returned	0.000 sec / 0.000 sec
38	15:35:58	drop index Latest_Release_index on MOVIE	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.063 sec
39	15:36:11	explain SELECT min(Released_Year) AS Latest_Release FROM MOVIE	1 row(s) returned	0.000 sec / 0.000 sec

FIGURE 101 : COMPLEXITY OF THE QUERY BEFORE ADDING THE LATEST_RELEASE INDEX

The screenshot shows the MySQL Workbench interface with the following details:

- Schemas:** tvseries_movies_collection
- Tables:** country, director, episode, main_actor, movie, name_of_the_studio, name_of_the_website, review, song, studio, tv_series.
- SQL Editor:** EE4202_Final* - SQL File 11


```

526
527 • create unique index Latest_Release_index using BTREE on MOVIE(Released_Year);
528
529 • explain
530 SELECT
531 min(Released_Year)
532 AS Latest_Release
533 FROM MOVIE;
      
```
- Result Grid:** Shows 1 row(s) returned.
- Output Window:**

#	Time	Action	Message	Duration / Fetch
37	15:35:36	explain SELECT min(Released_Year) AS Latest_Release FROM MOVIE	1 row(s) returned	0.000 sec / 0.000 sec
38	15:35:58	drop index Latest_Release_index on MOVIE	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.063 sec
39	15:36:11	explain SELECT min(Released_Year) AS Latest_Release FROM MOVIE	1 row(s) returned	0.000 sec / 0.000 sec
40	15:38:15	create unique index Latest_Release_index using BTREE on MOVIE(Released_Year)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.062 sec
41	15:38:48	explain SELECT min(Released_Year) AS Latest_Release FROM MOVIE	1 row(s) returned	0.000 sec / 0.000 sec

FIGURE 102 : COMPLEXITY OF THE QUERY AFTER ADDING THE LATEST_RELEASE INDEX

```

123 • SELECT *
124   FROM MOVIE
125 WHERE MOVIE_NAME LIKE '%Ava%';
126
127 • explain
128   SELECT *
129   FROM MOVIE
130 WHERE MOVIE_NAME LIKE '%Ava%';

Result Grid | Filter Rows: | Exports: | Wrap Cell Content: |
id select_type table partitions type possible_keys key key_len ref rows filtered Extra
1 SIMPLE MOVIE ALL NULL NULL NULL NULL NULL 19 11.11 Using where

```

Result 20 x

Action	Time	Action	Message	Duration / Fetch
40	15:38:15	create unique index Latest_Release_index using BTREE on MOVIE(RELEASED_YEAR)	0 rows affected Records: 0 Duplicates: 0 Warnings: 0	0.062 sec
41	15:38:48	explain SELECT min(Released_Year) AS Latest_Release FROM MOVIE	1 row(s) returned	0.000 sec / 0.000 sec
42	17:30:42	SELECT * FROM tvseries_movies_collection1.movie LIMIT 0, 1000	19 row(s) returned	0.000 sec / 0.000 sec
43	17:32:10	SELECT * FROM MOVIE WHERE MOVIE_NAME LIKE '%Ava%' LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec
44	17:32:37	explain SELECT * FROM MOVIE WHERE MOVIE_NAME LIKE '%Ava%'	1 row(s) returned	0.000 sec / 0.000 sec

FIGURE 103 : COMPLEXITY OF THE QUERY BEFORE ADDING THE MOVIE_NAME INDEX

```

527 • explain
528   SELECT *
529   FROM MOVIE
530 WHERE MOVIE_NAME LIKE '%Ava%';
531
532 • create index movie_name_index on movie(MOVIE_NAME);
533
534 • explain
535   SELECT *
536   FROM MOVIE
537 WHERE MOVIE_NAME LIKE '%Ava%';
538

Result Grid | Filter Rows: | Exports: | Wrap Cell Content: |
id select_type table partitions type possible_keys key key_len ref rows filtered Extra
1 SIMPLE MOVIE ALL NULL NULL NULL NULL NULL 19 11.11 Using where

Result 21 x
Action Output
# Time Action Message Duration / Fetch
42 17:30:42 SELECT * FROM tvseries_movies_collection1.movie LIMIT 0, 1000 19 row(s) returned 0.000 sec / 0.000 sec

```

FIGURE 104 : COMPLEXITY OF THE QUERY AFTER ADDING THE MOVIE_NAME INDEX

According to the rule we cannot use indexing for the like operator because it will search all the rows even there is an index. That's why here there is no change in the complexity of the queries.

```

MySQL Workbench
File Edit View Query Database Server Tools Scripting Help
Navigator: EE4202_Final SQL File 11 country movie director review episode director movie movie tv_series
Schemas: Filter objects
tvseries_movies_collection
tvseries_movies_collection
Tables: country director episode main_actor movie name_of_the_studio name_of_the_website review song studio tv_series
Views: Stored Procedures Functions
Administration Schemas
Information: Table: episode
Columns: TITLE RATINGS EPISODE_LENGTH tv_series_id COST_PER_EPISODE
Object Info Session Output: 86°F Partly sunny
Result Grid | Filter Rows: Export: Wrap Cell Contents: 
1 PRIMARY TV_SERIES ALL ref #3 #3 83 const 100.00 Using temporary
2 EXCEPT TV_SERIES ALL ALL ALL ALL ALL ALL 25 Using where
3 EXCEPT RESULT <except1,2> ALL ALL ALL ALL ALL ALL 0 Using temporary
Result 25 x
Read Only Context Help Snippets

```

FIGURE 105 : COMPLEXITY OF THE QUERY BEFORE ADDING THE COUNTRY_NAME_INDEX

```

MySQL Workbench
File Edit View Query Database Server Tools Scripting Help
Navigator: EE4202_Final SQL File 11 country movie director review episode director movie movie tv_series
Schemas: Filter objects
tvseries_movies_collection
tvseries_movies_collection
Tables: country director episode main_actor movie name_of_the_studio name_of_the_website review song studio tv_series
Views: Stored Procedures Functions
Administration Schemas
Information: Table: episode
Columns: TITLE RATINGS EPISODE_LENGTH tv_series_id COST_PER_EPISODE
Object Info Session Output: 86°F Partly sunny
Result Grid | Filter Rows: Export: Wrap Cell Contents: 
1 PRIMARY TV_SERIES ALL ref #3 #3 83 const 100.00 Using temporary
2 EXCEPT TV_SERIES ALL ref GENRE_index 82 const 4 100.00 Using temporary
3 EXCEPT RESULT <except1,2> ALL ALL ALL ALL ALL ALL 0 Using temporary
Result 26 x
Action Output: 54 17:51:01 create index country_name_index on country(country_name)
Message: 0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0
Duration / Fetch: 0.047 sec
Read Only Context Help Snippets

```

FIGURE 106 : COMPLEXITY OF THE QUERY AFTER ADDING THE COUNTRY_NAME_INDEX

The screenshot shows the MySQL Workbench interface with a query editor window titled "EE4202_Final". The code is as follows:

```

853 •    SELECT TV_SERIES_NAME
854     FROM TV_SERIES
855     WHERE TV_SERIES_ID IN (
856         SELECT TV_SERIES_ID
857         FROM REVIEW
858         WHERE COMMENTS = 'Classic Disney!'
859     );
860
861 •    explain
862     SELECT TV_SERIES_NAME
863     FROM TV_SERIES
864     WHERE TV_SERIES_ID IN (
865         SELECT TV_SERIES_ID
866         FROM REVIEW
867         WHERE COMMENTS = 'Classic Disney!'
868     );

```

The "Result Grid" pane shows the execution plan for the EXPLAIN command. The output is:

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	<subquery2>	ALL	ALL					100.00	100.00	
1	SIMPLE	TV_SERIES	ALL	eq_ref	PRIMARY,`idx_tv_series_id`	PRIMARY	42	<subquery2>.TV_SERIES_ID	100.00	100.00	
2	MATERIALIZED	REVIEW	ALL	PRIMARY					5	20.00	Using where

The "Result 30" pane shows the results of the query, which is empty.

FIGURE 107 : COMPLEXITY OF THE QUERY BEFORE ADDING THE COMMENTS_INDEX

The screenshot shows the MySQL Workbench interface with the same query as Figure 107, but after creating an index on the "COMMENTS" column of the "REVIEW" table. The code is identical to Figure 107.

The "Result Grid" pane shows the execution plan for the EXPLAIN command. The output is:

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	REVIEW	ALL	ref	PRIMARY,comments_index	comments_index	803	const	1	100.00	Using index
1	SIMPLE	TV_SERIES	ALL	eq_ref	PRIMARY,`idx_tv_series_id`	PRIMARY	42	tseries_movies_collection.REVIEW.tv_series_id	100.00	100.00	

The "Result 31" pane shows the results of the query, which is empty.

FIGURE 108 : COMPLEXITY OF THE QUERY AFTER ADDING THE COMMENTS_INDEX

The screenshot shows the MySQL Workbench interface with a query editor containing a complex multi-table query. The code includes multiple subqueries and joins between tables like director, movie, and country. The 'Result Grid' pane shows the execution plan with many rows and columns of data. The status bar at the bottom indicates the query was executed at 18:24:11.

```

SELECT MOVIENAME
FROM MOVIE m
WHERE EXISTS (
    SELECT 1
    FROM DIRECTOR d
    WHERE d.movie_id = m.movie_id AND d.country = 'UK'
);

EXPLAIN
SELECT MOVIENAME
FROM MOVIE m
WHERE EXISTS (
    SELECT 1
    FROM DIRECTOR d
    WHERE d.movie_id = m.movie_id AND d.country = 'UK'
);

```

FIGURE 109 : COMPLEXITY OF THE QUERY BEFORE ADDING THE COUNTRY_NAME

The screenshot shows the MySQL Workbench interface with the same query as Figure 109, but now it includes a 'create index' statement for the 'country_name' column on the 'DIRECTOR' table. The 'Result Grid' pane shows the execution plan with fewer rows and columns compared to Figure 109, indicating improved performance. The status bar at the bottom indicates the index creation was completed at 18:25:36.

```

CREATE INDEX country_name ON DIRECTOR(COUNTRY);

EXPLAIN
SELECT MOVIENAME
FROM MOVIE m
WHERE EXISTS (
    SELECT 1
    FROM DIRECTOR d
    WHERE d.movie_id = m.movie_id AND d.country = 'UK'
);

```

FIGURE 110 : COMPLEXITY OF THE QUERY AFTER ADDING THE COUNTRY_NAME

```

571 -- Natural join between TV_SERIES_EPISODE_VIEW and TV_SERIES table
572 • SELECT TV_SERIES_NAME, RELEASED_YEAR, GENRE, NO_OF_SEASONS, TITLE, RATINGS
573   FROM TV_SERIES_EPISODE_VIEW NATURAL JOIN TV_SERIES
574   where GENRE = 'Action' ;
575
576 • explain
577   SELECT TV_SERIES_NAME, RELEASED_YEAR, GENRE, NO_OF_SEASONS, TITLE, RATINGS
578   FROM TV_SERIES_EPISODE_VIEW NATURAL JOIN TV_SERIES
579   where GENRE = 'Action' ;
580

```

Result Grid

In	id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	1	SIMPLE	TV_SERIES	NULL	ALL	PRIMARY,IDX_ts_TV_SERIES_ID,released_year...		NULL	NULL	25	10.00	Using where
1	1	SIMPLE	ts	NULL	eq_ref	PRIMARY,IDX_ts_TV_SERIES_ID,released_year...	PRIMARY	42	tvsereis_movies_collection1.TV_SERIES.TV_SE...	1	5.00	Using where
1	1	SIMPLE	e	NULL	ref	PRIMARY	202		tvsereis_movies_collection1.TV_SERIES.TV_SE...	1	100.00	Using where

Action Output

#	Time	Action	Message	Duration / Fetch
104	19:48:49	drop index GENRE_index on TV_SERIES	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.031 sec
105	19:49:14	explain SELECT TV_SERIES_NAME, RELEASED_YEAR, GENRE, NO_OF_SEASONS, TITLE, RATINGS FROM TV_SERIES... 3 row(s) returned		0.000 sec / 0.000 sec

FIGURE 111 : COMPLEXITY OF THE QUERY BEFORE ADDING THE GENRE_INDEX

```

575
576
577 |
578 • create index GENRE_index on TV_SERIES(GENRE);
579
580 • explain
581   SELECT TV_SERIES_NAME, RELEASED_YEAR, GENRE, NO_OF_SEASONS, TITLE, RATINGS
582   FROM TV_SERIES_EPISODE_VIEW NATURAL JOIN TV_SERIES
583   where GENRE = 'Action' ;
584
585
586

```

Result Grid

In	id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	1	SIMPLE	ts	NULL	ref	PRIMARY,IDX_ts_TV_SERIES_ID,released_year...	GENRE_index	82	const	2	100.00	NULL
1	1	SIMPLE	TV_SERIES	NULL	eq_ref	PRIMARY,IDX_ts_TV_SERIES_ID,released_year...	PRIMARY	42	tvsereis_movies_collection1.ts.TV_SERIES.ID	1	5.00	Using where
1	1	SIMPLE	e	NULL	ref	PRIMARY	202		tvsereis_movies_collection1.ts.TV_SERIES.ID	1	100.00	Using where

Action Output

#	Time	Action	Message	Duration / Fetch
106	19:49:52	create index GENRE_index on TV_SERIES(GENRE)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.046 sec
107	19:50:11	explain SELECT TV_SERIES_NAME, RELEASED_YEAR, GENRE, NO_OF_SEASONS, TITLE, RATINGS FROM TV_SERIES... 3 row(s) returned		0.000 sec / 0.000 sec

FIGURE 112 : COMPLEXITY OF THE QUERY AFTER ADDING THE GENRE_INDEX

```

MySQL Workbench
File Edit View Query Database Server Tools Scripting Help
EE4202_Final* director movie tv_series
585
586
587 -- Left outer join between MAIN_ACTOR_MOVIE_VIEW and MOVIE table
588 • SELECT ma.FIRST_NAME, ma.LAST_NAME, m.MOVIE_NAME
589 FROM MAIN_ACTOR_MOVIE_VIEW ma LEFT OUTER JOIN MOVIE m
590 ON ma.MOVIE_NAME = m.MOVIE_NAME;
591
592 • explain
593 SELECT ma.FIRST_NAME, ma.LAST_NAME, m.MOVIE_NAME
594 FROM MAIN_ACTOR_MOVIE_VIEW ma
595 LEFT OUTER JOIN MOVIE m ON ma.MOVIE_NAME = m.MOVIE_NAME;
596

```

Result Grid | Filter Rows: Export: Wrap Cell Content:

In	id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	1	SIMPLE	ma	NULL	ALL	NULL	NULL	NULL	NULL	6	100.00	Using where
1	1	SIMPLE	m	NULL	eq_ref	PRIMARY	PRIMARY	42	tvseries_movies_collection1.ma.movie_id	1	100.00	Using where; Using join buffer (hash join)
1	1	SIMPLE	m	NULL	ALL	NULL	NULL	NULL	NULL	19	100.00	Using where; Using join buffer (hash join)

Result 63 x

Action Output

#	Time	Action	Message	Duration / Fetch
110	19:57:49	drop index movie_name_index on movie	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.031 sec
111	19:58:17	explain SELECT ma.FIRST_NAME, ma.LAST_NAME, m.MOVIE_NAME FROM MAIN_ACTOR_MOVIE_VIEW ma LEFT OUTER...	3 row(s) returned	0.000 sec / 0.000 sec

Output

83°F Mostly cloudy

FIGURE 113 : COMPLEXITY OF THE QUERY BEFORE ADDING MOVIE_NAME_INDEX

```

MySQL Workbench
File Edit View Query Database Server Tools Scripting Help
EE4202_Final* director movie tv_series
96
97 • create index movie_name_index on movie(MOVIE_NAME);
98
99 • explain
00 SELECT ma.FIRST_NAME, ma.LAST_NAME, m.MOVIE_NAME
01 FROM MAIN_ACTOR_MOVIE_VIEW ma
02 LEFT OUTER JOIN MOVIE m ON ma.MOVIE_NAME = m.MOVIE_NAME;
03
04

```

Result Grid | Filter Rows: Export: Wrap Cell Content:

In	In	id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	1	1	SIMPLE	ma	NULL	ALL	NULL	NULL	NULL	6	100.00	NULL	
1	1	1	SIMPLE	m	NULL	eq_ref	PRIMARY	PRIMARY	42	tvseries_movies_collection1.ma.movie_id	1	100.00	NULL
1	1	1	SIMPLE	m	NULL	ref	movie_name_index	movie_name_index	202	tvseries_movies_collection1.m.MOVIE_NAME	1	100.00	Using index

Result 64 x

Action Output

#	Time	Action	Message	Duration / Fetch
112	19:59:10	create index movie_name_index on movie(MOVIE_NAME)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.047 sec
113	19:59:25	explain SELECT ma.FIRST_NAME, ma.LAST_NAME, m.MOVIE_NAME FROM MAIN_ACTOR_MOVIE_VIEW ma LEFT OUTER...	3 row(s) returned	0.000 sec / 0.000 sec

Output

83°F Mostly cloudy

FIGURE 114 : COMPLEXITY OF THE QUERY AFTER ADDING MOVIE_NAME_INDEX

```

514 •    SELECT
515      min(RELEASED_YEAR)
516      AS Latest_Release
517
518
519
520 •    explain
521      SELECT
522      min(RELEASED_YEAR)
523      AS Latest_Release
524      FROM MOVIE;
525

```

Result Grid Filter Rows: <input type="text"/> Export: Wrap Cell Content:												
	id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	1	SIMPLE	MOVIE	NULL	ALL	NULL	NULL	NULL	NULL	19	100.00	NULL

FIGURE 115 : COMPLEXITY OF THE QUERY BEFORE ADDS THE INDEX

```

526 •    create unique index Latest_Release_index using BTREE on MOVIE(RELEASED_YEAR);
527
528 •    explain
529      SELECT
530      min(RELEASED_YEAR)
531      AS Latest_Release
532      FROM MOVIE;
533

```

Result Grid Filter Rows: <input type="text"/> Export: Wrap Cell Content:												
	id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	1	SIMPLE	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	Select tables optimized away

FIGURE 116: COMPLEXITY OF THE QUERY AFTER ADDS THE INDEX

- As the index was added according to the query tuning guide lines.
- It reduces the searching time from 19 rows to 0 rows.
- So, it reduces the complexity for the retrieval.

```

505 •   SELECT
506     DIRECTOR_NAME AS Director,
507     AGE AS Director_Age
508   FROM DIRECTOR
509   where AGE = 20;
510
511 •   EXPLAIN
512   SELECT
513     DIRECTOR_NAME AS Director,
514     AGE AS Director_Age
515   FROM DIRECTOR
516   where AGE = 20;

```

Result Grid Filter Rows: <input type="text"/> Export: Wrap Cell Content:												
	id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
▶	1	SIMPLE	DIRECTOR	NULL	ALL	NULL	NULL	NULL	NULL	5	20.00	Using where

FIGURE 117 : COMPLEXITY OF THE QUERY BEFORE ADDS THE INDEX

```

505 •   SELECT
506     DIRECTOR_NAME AS Director,
507     AGE AS Director_Age
508   FROM DIRECTOR
509   where AGE = 20;
510
511 •   EXPLAIN
512   SELECT
513     DIRECTOR_NAME AS Director,
514     AGE AS Director_Age
515   FROM DIRECTOR
516   where AGE = 20;

```

Result Grid Filter Rows: <input type="text"/> Export: Wrap Cell Content:												
	id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
▶	1	SIMPLE	DIRECTOR	NULL	const	Director_Age_index	Director_Age_index	5	const	1	100.00	NULL

FIGURE 118 : COMPLEXITY OF THE QUERY AFTER ADDS THE INDEX

iv. ERROR AND SOLUTION

ERROR 01

As we tried to create unique index for the non-unique attribute. So, we got this error.

0 80 20:37:15 create unique index GENRE_INDEX Error Code: 1062. Duplicate entry 'Drama' 0.031
using BTREE on MOVIE(GENRE) for key 'movie.GENRE_INDEX' sec

3

create unique index
0 81 20:41:38 RELEASED_YEAR_INDEX using
BTREE on
MOVIE(RELEASED_YEAR)

Error Code: 1062. Duplicate entry '1994'
for key
'movie.RELEASED_YEAR_INDEX'
0.032 sec

	MOVIE_ID	MOVIE_NAME	RELEASED_YEAR	GENRE	DURATION	M_LANGUAGE	BUDGET
▶	MV001	The Shawshank Redemption	1994	Drama	02:22:00	English	25000000.00000
	MV002	The Godfather	1972	Drama, Crime	02:55:00	English	6000000.00000
	MV003	The Dark Knight	2008	Action	02:32:00	English	200000000.00000
	MV004	Pulp Fiction	1994	Crime	02:34:00	English	8000000.00000
	MV005	Forrest Gump	1994	Drama	02:22:00	English	55000000.00000
	MV006	Inception	2010	Science Fiction	02:28:00	English	160000000.00000
*	NUL	NUL	NUL	NUL	NUL	NUL	NUL

691

692 • show index from MOVIE ;

Result Grid														
Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment	Visible	Expression
▶ movie	0	PRIMARY	1	MOVIE_ID	A	6	NUL	NUL		BTREE			YES	NUL
movie	0	MOVIE_NAME_INDEX	1	MOVIE_NAME	A	6	NUL	NUL		BTREE			YES	NUL

- Because those are not unique keys. so, we cannot create the unique index. We have to create the secondary index (clustering index).

ERROR - 02

```

736 •   explain
737     SELECT TV_SERIES_NAME
738     FROM TV_SERIES ts
739     WHERE ts.TV_SERIES_ID NOT IN (
740         SELECT tv_series_id
741         FROM EPISODE e
742         WHERE e.RATINGS > (
743             SELECT AVG(RATINGS)
744             FROM EPISODE
745         )
746     );
    
```

Result Grid									rows	filtered	Extra	
	id	select_type	table	partitions	type	possible_keys	key	key_len	ref			
▶	1	PRIMARY	ts	NULL	ALL	NULL	NULL	NULL	NULL	5	100.00	NULL
	1	PRIMARY	e	NULL	ref	PRIMARY	PRIMARY	202	tvsereis_movies_collection1.ts.TV_SERIES_ID	1	100.00	Using where; Not exists
	3	SUBQUERY	EPISODE	NULL	ALL	NULL	NULL	NULL	NULL	4	100.00	NULL

FIGURE 119 : COMPLEXITY OF THE QUERY BEFORE ADDS THE INDEX

The error came as we try to implement the index again for the particular column.

create unique index 0 74 20:25:03 TV_SERIES_NAME_INDEX using BTREE on TV_SERIES(TV_SERIES_NAME)	Error Code: 1061. Duplicate key name 'TV_SERIES_NAME_INDEX' 0.000 sec
--	--

```

822 •   create unique index TV_SERIES_NAME_INDEX
823     using BTREE
824     on TV_SERIES(TV_SERIES_NAME);
825
826 •   explain
827     SELECT TV_SERIES_NAME
828     FROM TV_SERIES ts
829     WHERE ts.TV_SERIES_ID NOT IN (
830         SELECT tv_series_id
831         FROM EPISODE e
832         WHERE e.RATINGS > (
833             SELECT AVG(RATINGS)
834             FROM EPISODE
835         )
836     );
    
```