In [1]:

```
import pandas as pd
import sqlite3
import numpy as np
conn = sqlite3.connect('Db-IMDB.db')
```

Data Preprocessing

1.Removing duplicates from person table

In [19]:

```
actor = pd.read_sql_query("select name,count(PID) as num_count from PERSON group by name order by
num_count desc ",conn)
actor.head()
```

Out[19]:

	Name	num_count
0	Rajesh	12
1	Rahul	11
2	Imran Khan	10
3	Deepak	9
4	Raju	8

In [27]:

```
person = pd.read_sql_query("select * from PERSON", conn)
person.head()
```

Out[27]:

	index	PID	Name	Gender
0	0	nm0000288	Christian Bale	Male
1	1	nm0000949	Cate Blanchett	Female
2	2	nm1212722	Benedict Cumberbatch	Male
3	3	nm0365140	Naomie Harris	Female
4	4	nm0785227	Andy Serkis	Male

In [22]:

```
# we can clearly see that we have a lot of duplicate rows.
# we will remove all the duplicates and create another table and insert rows in that table.
new_person = person.drop_duplicates(subset={'PID'},keep='first')
print("Rows before removing duplicates {}".format(person.shape[0]))
print("Rows after removing duplicates {}".format(new_person.shape[0]))
```

```
Rows before removing duplicates 38285
Rows after removing duplicates 37566
```

In [35]:

```
cursor = conn.cursor()
cursor.execute("DROP TABLE Persons")
conn.commit()
```

In [36]:

```
# We will create a new table with name persons
cursor = conn.cursor()
cursor.execute('CREATE TABLE Persons(PID VARCHAR(50) PRIMARY KEY,Name VARCHAR(100),Gender VARCHAR(
10));')
conn.commit()
```

In [37]:

```
from tqdm import tqdm
cursor = conn.cursor()
for i in tqdm(range(new_person.shape[0])):
    row = list(new_person.iloc[i].values)
    cursor.execute('INSERT INTO Persons VALUES(?,?,?)',row[1:])
conn.commit()
```

```
100%|██████████████████████████████████████████████████████████████████████████| 37566/37566
[00:26<00:00, 1439.51it/s]
```

2.Trimmimg PID from M_Cast table

In [42]:

```
cast = pd.read_sql_query("select * from M_Cast", conn)
```

In [49]:

```
cursor = conn.cursor()
cursor.execute("UPDATE M_Cast SET PID = REPLACE(PID, ' ', '')")
conn.commit()
```

Removing Roman Numerals from year column

In [52]:

```
rom_year = pd.read_sql_query("select year from Movie where year LIKE '%I%'", conn)
rom_year.head(5)
```

Out[52]:

	year
0	I 2009
1	I 2018
2	XVII 2016
3	I 2017
4	II 2018

In [54]:

```
year = pd.read_sql_query("select * from Movie", conn)
```

In [56]:

```
cursor = conn.cursor()
cursor.execute("UPDATE Movie SET year = REPLACE(year,'I','')")
cursor.execute("UPDATE Movie SET year = REPLACE(year,'V','')")
cursor.execute("UPDATE Movie SET year = REPLACE(year,'X','')")
conn.commit()
```

In [57]:

```
rom_year = pd.read_sql_query("select year from Movie where year LIKE '%I%'", conn)
rom_year.head(5)
```

Out[57]:

	year

1.List all the directors who directed a 'Comedy' movie in a leap year. (You need to check that the genre is 'Comedy' and year is a leap year) Your query should return director name, the movie name, and the year.

In [58]:

```
rows = pd.read_sql_query('''select Name,title,year from Persons p join M_Director md on p.PID = md
.PID join Movie m on m.MID = md.MID where m.MID in
                         (select MID from Movie where (year%4=0 and year%100!=0) and (year%4=0 o
r year%100=0 and year%400=0) and MID in
                         (select MID from M_Genre where GID in(select GID from Genre where Name
LIKE '%comedy%')))''', conn)
print(rows.head(10))
```

	Name	title	year
0	Milap Zaveri	Mastizaade	2016
1	Danny Leiner	Harold & Kumar Go to White Castle	2004
2	Anurag Kashyap	Gangs of Wasseypur	2012
3	Frank Coraci	Around the World in 80 Days	2004
4	Griffin Dunne	The Accidental Husband	2008
5	Anurag Basu	Barfi!	2012
6	Gurinder Chadha	Bride & Prejudice	2004
7	Mike Judge	Beavis and Butt-Head Do America	1996
8	Tarun Mansukhani	Dostana	2008
9	Shakun Batra	Kapoor & Sons	2016

2.List the names of all the actors who played in the movie 'Anand' (1970)

In [60]:

```
rows = pd.read_sql_query('''SELECT Name from Persons where TRIM(PID) in
                         (SELECT TRIM(PID) from M_Cast where MID in
                          (SELECT MID from Movie where title ='Anand'))''', conn)
print(rows)
```

	Name
0	Amitabh Bachchan
1	Rajesh Khanna
2	Sumita Sanyal
3	Ramesh Deo
4	Seema Deo
5	Asit Kumar Sen
6	Dev Kishan
7	Atam Prakash
8	Lalita Kumari
9	Savita
10	Brahm Bhardwaj
11	Gurnam Singh
12	Lalita Pawar
13	Durga Khote
14	Dara Singh
15	Johnny Walker
16	Moolchand

3.List all the actors who acted in a film before 1970 and in a film after 1990. (That is: < 1970 and > 1990.)

In [62]:

```
rows = pd.read_sql_query('''select Name from Persons where TRIM(PID) in
                         (select TRIM(PID) from M_Cast where MID in
                         (select TRIM(MID) from Movie where TRIM(year)<'1970') INTERSECT
                         select TRIM(PID) from M_Cast where TRIM(MID) in
                         (select TRIM(MID) from Movie where TRIM(year)>'1990'))''', conn)
print(rows.head(10))
```

```
                Name
0       Rishi Kapoor
1   Amitabh Bachchan
2             Asrani
3       Zohra Sehgal
4    Parikshat Sahni
5      Rakesh Sharma
6        Sanjay Dutt
7          Ric Young
8              Yusuf
9     Suhasini Mulay
```

4.List all directors who directed 10 movies or more, in descending order of the number of movies they directed. Return the directors' names and the number of movies each of them directed

In [63]:

```
rows = pd.read_sql_query('''select Name,count(Name) cn from
                         Persons p join M_Director md on p.PID = md.PID join Movie m on m.MID = md.
MID
                         group by Name having cn>10 order by cn desc''', conn)
print(rows.head())
```

```
                Name  cn
0      David Dhawan  39
1      Mahesh Bhatt  36
2   Ram Gopal Varma  30
3      Priyadarshan  30
4      Vikram Bhatt  29
```

5(a) For each year, count the number of movies in that year that had only female actors.

In [64]:

```
# 5(a)
rows = pd.read_sql_query('''select year,count(*) from Movie where MID in
                         (select MID from M_Cast where PID in
                         (select PID from Persons where Gender='Female') and MID NOT IN
                         (select MID from M_Cast where PID in(select PID from Persons where Gender=
'Male')))group by year''', conn)
rows.head()
```

Out[64]:

	year	count(*)
0	2018	1
1	1939	1
2	1999	1
3	2000	1

5(b) Now include a small change: report for each year the percentage of movies in that year with only female actors, and the total number of movies made that year. For example, one answer will be: 1990 31.81 13522 meaning that in 1990 there were 13,522 movies, and 31.81% had only

female actors. You do not need to round your answer.

In [65]:

```
# 5(b)
rows = pd.read_sql_query('''select year,(select count(*) from Movie where MID in(select MID from M
_Cast where PID in
                         (select PID from Persons where Gender='Female') and MID NOT IN
                         (select MID from M_Cast where PID in
                         (select PID from Persons where Gender='Male')))group by year) * 100 /count
(distinct MID) as percentage,count(distinct MID) as total_cnt from Movie where year in
                         (select year from Movie where MID in(select MID from M_Cast where PID in(s
elect PID from Persons where Gender='Female') and MID NOT IN
                         (select MID from M_Cast where PID in(select PID from Persons where Gender=
'Male')))group by year)group by year''', conn)
rows.head()
```

Out[65]:

	year	percentage	total_cnt
0	2018	9	11
1	1939	50	2
2	1999	1	66
3	2000	1	64

6. Find the film(s) with the largest cast. Return the movie title and the size of the cast. By "cast size" we mean the number of distinct actors that played in that movie: if an actor played multiple roles, or if it simply occurs multiple times in casts, we still count her/him only once.

In [66]:

```
a=pd.read_sql_query("SELECT title,COUNT(DISTINCT(PID)) AS cast_size  FROM M_Cast c join MOVIE m on
m.MID=c.MID group by title ORDER BY cast_size DESC",conn)
a.head()
```

Out[66]:

	title	cast_size
0	Ocean's Eight	238
1	Apaharan	233
2	Gold	215
3	My Name Is Khan	213
4	Captain America: Civil War	191

7. A decade is a sequence of 10 consecutive years. For example, say in your database you have movie information starting from 1965. Then the first decade is 1965, 1966, ..., 1974; the second one is 1967, 1968, ..., 1976 and so on. Find the decade D with the largest number of films and the total number of films in D.

In [67]:

```
a=pd.read_sql_query('''SELECT (year/10*10) || '-' || (year/10*10+9)  AS DECADE,count(MID) AS
movie_count FROM
                    MOVIE GROUP BY [Year]/10*10 ORDER BY movie_count DESC''',conn)
a.head()
```

Out[67]:

	DECADE	movie_count
0	2010-2019	1092

	DECADE	movie_count
1	2000-2009	988
2	1990-1999	556
3	1980-1989	350
4	1970-1979	254

8. Find the actors that were never unemployed for more than 3 years at a stretch. (Assume that the actors remain unemployed between two consecutive movies).

In [69]:

```
rows = pd.read_sql_query('''select Name from (M_cast mc join Movie m on mc.MID=m.MID join Persons
p on p.PID=mc.PID) as t1 where
                         Name in(select Name from (M_cast mc join Movie m on mc.MID=m.MID join Pers
ons p on p.PID=mc.PID)as t2
                         where t1.Name=t2.Name and (t1.year-t2.year)*(t1.year-t2.year)>9 order by y
ear)''',conn)
rows.head()
```

Out[69]:

	Name
0	Freida Pinto
1	Rohan Chand
2	Griffin Dunne
3	Damian Young
4	Waris Ahluwalia

9. Find all the actors that made more movies with Yash Chopra than any other director.

In [70]:

```
# Answer for question 9
rows = pd.read_sql_query("""select distinct actor from (select actor,director,max(c_dir) m from
                         (select act.Name as actor,dir.Name as director, count(*) as c_dir from
                          (M_Director md join Persons p on md.PID=p.PID) as dir
                          join(M_cast mc join Persons p on mc.PID=p.PID) as act
                          on dir.MID=act.MID group by actor,dir.Name order by c_dir desc)as result1
                         group by actor order by m desc)as result2 where TRIM(director) = 'Yash
Chopra'""", conn)
rows.head()
```

Out[70]:

	actor
0	Jagdish Raj
1	Manmohan Krishna
2	Iftekhar
3	Shashi Kapoor
4	Rakhee Gulzar

10. The Shahrukh number of an actor is the length of the shortest path between the actor and Shahrukh Khan in the "co-acting" graph. That is, Shahrukh Khan has Shahrukh number 0; all actors who acted in the same film as Shahrukh have Shahrukh number 1; all actors who acted in the same film as some actor with Shahrukh number 1 have Shahrukh number 2, etc. Return all actors whose Shahrukh number is 2.

In [71]:

```
# Answer of question 10
rows = pd.read_sql_query("""select Name from Person where PID in
                         (select distinct PID from M_Cast where MID in
                          (select MID from(M_Cast mc join Person p on mc.PID=p.PID)as t3 where t3.P
ID in
                           (select PID from M_Cast where MID in(select MID from(Person p join M_Cas
t mc on p.PID=mc.PID) as t1
                            where TRIM(t1.Name)='Shah Rukh Khan') and PID not in
                            (select PID from Person where TRIM(Name)='Shah Rukh Khan'))))""", conn)
rows.head()
```

Out[71]:

	Name
0	Freida Pinto
1	Rohan Chand
2	Damian Young
3	Waris Ahluwalia
4	Caroline Christl Long