# SQL Assignment

In [1]:
```
import pandas as pd
import sqlite3
```

In [2]:
```
conn = sqlite3.connect("Db-IMDB-Assignment.db")
cursor=conn.cursor()

```

## Pre-Process the table

In [3]:
```
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 ", "");')
cursor.execute('UPDATE Movie SET title = LTRIM(title);')
cursor.execute('UPDATE Movie SET year = RTRIM(LTRIM(year));')
cursor.execute('UPDATE Movie SET rating = RTRIM(LTRIM(rating));')
cursor.execute('UPDATE Movie SET num_votes = RTRIM(LTRIM(num_votes));')

cursor.execute('UPDATE M_Producer SET pid = RTRIM(LTRIM(pid));')
cursor.execute('UPDATE M_Producer SET mid = RTRIM(LTRIM(mid));')

cursor.execute('UPDATE M_Director SET pid = RTRIM(LTRIM(pid));')
cursor.execute('UPDATE M_Director SET mid = RTRIM(LTRIM(mid));')

cursor.execute('UPDATE M_Cast SET pid = RTRIM(LTRIM(pid));')
cursor.execute('UPDATE M_Cast SET mid = RTRIM(LTRIM(mid));')

cursor.execute('UPDATE M_Genre SET gid = RTRIM(LTRIM(gid));')
cursor.execute('UPDATE M_Genre SET mid = RTRIM(LTRIM(mid));')

cursor.execute('UPDATE Genre SET gid = RTRIM(LTRIM(gid));')
cursor.execute('UPDATE Genre SET name = RTRIM(LTRIM(name));')

cursor.execute('UPDATE Person SET name = RTRIM(LTRIM(name));')
cursor.execute('UPDATE Person SET pid = RTRIM(LTRIM(pid));')
cursor.execute('UPDATE Person SET gender = RTRIM(LTRIM(gender));')

### conn.commit()

pd.read_sql_query("""  SELECT * from movie ORDER BY year DESC limit 5""", conn)
```

Out[3]:

| | index | MID | title | year | rating | num_votes |
|---|---|---|---|---|---|---|
| **0** | 0 | tt2388771 | Mowgli | 2018 | 6.6 | 21967 |
| **1** | 1 | tt5164214 | Ocean's Eight | 2018 | 6.2 | 110861 |
| **2** | 2 | tt1365519 | Tomb Raider | 2018 | 6.4 | 142585 |
| **3** | 4 | tt8239946 | Tumbbad | 2018 | 8.5 | 7483 |
| **4** | 5 | tt7027278 | Kedarnath | 2018 | 5.5 | 1970 |

## Q1 --- 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 [4]:

```
%%time
# Write your sql query below d.name,m.name,m.year
# select MID,case when ((year % 4 = 0 and  year % 100 <> 0) or (year % 400 = 0) ) then 'l'
#                     else 'nl' end as 'leap' from Movie
query = """ select  p.name as Director_name,m.title as Movie_name ,m.year as Year from
                Movie m
            join
                M_Director d using(MID)
            join
                Person p on d.PID=p.PID
            join
                M_Genre mg on mg.MID=m.MID
            join
                Genre g on g.GID=mg.GID
                where LOWER(g.name) like '%comedy%' and m.year in
                (select distinct year from
                    (select year,case
                                    when ((year % 4 = 0 and  year % 100 <> 0) or (year % 400 = 0) ) then 'leap'
                                    else 'non_leap'
                                end as 'year_type' from Movie)leap
                     where year_type = 'leap')
        """

q1 = pd.read_sql_query(query, conn)
print(q1.shape)

```

```
(232, 3)
Wall time: 107 ms
```

In [5]:

```
print(q1.head(10))
print('*'*50)
print('Number of rows ',len(q1))

```

```
     Director_name                       Movie_name  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
**************************************************
Number of rows  232
```

## Q2 --- List the names of all the actors who played in the movie 'Anand' (1971)

In [6]:

```
%%time
# Write your sql query below

query = """select  p.name as Actor_name from Person p join M_cast mc using(pid) join Movie m on m.MID = mc.mid where
        lower(m.title) = 'anand' """

q2 = pd.read_sql_query(query, conn)
print(q2.shape)
q2.head()
```

```
(17, 1)
Wall time: 265 ms
```

In [7]:

```
print(q2.head(10))
print('*'*50)
print('Number of rows ',len(q2))
```

```
        Actor_name
0  Amitabh Bachchan
1     Rajesh Khanna
2    Brahm Bhardwaj
3        Ramesh Deo
4         Seema Deo
5        Dev Kishan
6       Durga Khote
7     Lalita Kumari
8      Lalita Pawar
9      Atam Prakash
**************************************************
Number of rows  17
```

## Q3 --- List all the actors who acted in a film before 1970 and in a film after 1990. (That is: < 1970 and > 1990.)

In [8]:

```
%%time
# Write your sql query below

query = """select distinct A.name from

        (select p.PID,p.name from
            Person p
        join
            M_Cast mc on mc.PID = p.PID
        join
            Movie m on m.MID=mc.MID where year < 1970 )A

        join

        (select p.PID,p.name from
            Person p
        join
            M_Cast mc on mc.PID = p.PID
        join
            Movie m on m.MID=mc.MID where  year > 1990 )B

        on A.PID = B.PID

        """

q3 = pd.read_sql_query(query, conn)
print(q3.shape)
q3.head()
```

```
(300, 1)
Wall time: 692 ms
```

In [9]:

```
print(q3.head(10))
print('*'*50)
print('Number of rows ',len(q3))
```

```
            name
0        Mehmood
1          Ratna
2  Waheeda Rehman
3   Johnny Walker
4  Rajendra Kumar
5       Iftekhar
6      Raj Mehra
7    Lalita Pawar
8  Achala Sachdev
9      Sunil Dutt
**************************************************
Number of rows  300
```

## Q4 --- 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 [87]:

```
%%time
# Write your sql query below

query = """
        select p.name,count(distinct m.MID) as numberOfMovies  from M_Director md join Movie m using(MID) join Person p
        on p.PID=md.PID group by 1 having count(m.MID) >= 10 order by 2 desc

        """

q4 = pd.read_sql_query(query, conn)
print(q4.shape)
q4.head()
```

```
(58, 2)
Wall time: 178 ms
```

In [88]:

```
print(q4.head(5))
print('*'*50)
print('Number of rows ',len(q4))
```

```
              Name  numberOfMovies
0     David Dhawan              39
1     Mahesh Bhatt              36
2     Priyadarshan              30
3  Ram Gopal Varma              30
4     Vikram Bhatt              29
**************************************************
Number of rows  58
```

## Q5.a --- For each year, count the number of movies in that year that had only female actors.

In [41]:

```
%%time
# Write your sql query below
## added : UNION
##                     select distinct mc.MID from
##                         M_cast mc  where mc.PID is null
## to consider null PID from M_cast as non female movies too.
query = """ select m.year,count(A.MID) from Movie m left join (select m.year,m.MID
                    from Movie m where m.MID not in
                (select distinct m.MID from Movie m
                 join
                    M_cast mc using(MID)
                 join
                    Person p on p.PID=mc.PID where lower(TRIM(p.gender)) != 'female'
                 UNION
                    select distinct mc.MID from
                        M_cast mc  where mc.PID is null))A using(MID) group by 1

        """

q5a = pd.read_sql_query(query, conn)
print(q5a.shape)
q5a.head()
```

```
(78, 2)
Wall time: 373 ms
```

In [45]:

```
print(q5a.head(6))
print('*'*50)
print('Number of rows ',len(q5a))
```

```
   year  count(A.MID)
0  1931             0
1  1936             0
2  1939             1
3  1941             0
4  1943             0
5  1946             0
**************************************************
Number of rows  78
```

## Q5.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 [43]:

```
%%time
# Write your sql query below

query = """select m.year,count(distinct A.MID)*100 / count(distinct m.MID) as '%', count(distinct m.MID) as total_count
           from Movie m
           left join
                (select year,m.MID from Movie m where m.MID not in
                    (select distinct m.MID from Movie m
                 join
                    M_cast mc using(MID)
                 join
                    Person p on p.PID=mc.PID where lower(TRIM(p.gender)) != 'female'
                 UNION
                    select distinct mc.MID from
                        M_cast mc  where mc.PID is null))A
            on m.MId = A.MID group by 1
        """

q5b = pd.read_sql_query(query, conn)
print(q5b.shape)
q5b.head()
```

```
(78, 3)
Wall time: 390 ms
```

In [44]:

```
print(q5b.head(5))
print('*'*50)
print('Number of rows ',len(q5b))
```

```
   year   %  total_count
0  1931   0            1
1  1936   0            3
2  1939  50            2
3  1941   0            1
4  1943   0            1
**************************************************
Number of rows  78
```

## Q6 --- 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 [22]:

```
%%time
# Write your sql query below

query = """
        select m.title,count(distinct p.PID) from
            Movie m
        join
            M_Cast mc using(MID)
        join
            Person p using(PID)
        group by 1 order by 2 desc limit 1

        """

q6 = pd.read_sql_query(query, conn)
print(q6.shape)
q6.head()
```

```
(1, 2)
Wall time: 441 ms
```

In [23]:

```
print(q6.head(5))
print('*'*50)
print('Number of rows ',len(q6))
```

```
           title  count(distinct p.PID)
0  Ocean's Eight                    238
**************************************************
Number of rows  1
```

## Q7 --- 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 [24]:

```
%%time
# Write your sql query below

query = """
        select dy.year as begin, dy.year+9 as end,count(distinct m.MID) as count_movies
        from
            (select distinct year from Movie) dy
        join
            Movie m  on m.year>=begin and m.year<= end
        group by 1 order by 3 desc limit 1

        """

q7 = pd.read_sql_query(query, conn)
print(q7.shape)
q7.head()
```

```
(1, 3)
Wall time: 146 ms
```

In [25]:

```
print(q7.head(5))
print('*'*50)
print('Number of rows ',len(q7))
```

```
  begin   end  count_movies
0  2008  2017          1205
**************************************************
Number of rows  1
```

## Q8 --- Find all the actors that made more movies with Yash Chopra than any other director.

In [83]:

```
%%time
# Write your sql query below
## Approach ----
## 1. find all the  director name and actor name with count of distinct movies common to the actor and director
##    and sort by desc order of count of movies
## 1. query :
####   (select actor.name as act,dir.name as direc,count(distinct actor.MID) as cnt
#                 from
#                     (select m.MID,p.name from Movie m join M_director md using(MID) join Person p using(PID))dir
#                     join
#                     (select m.MID,p.name from Movie m join M_cast mc using(MID) join Person p using(PID))actor using (MID)
#              group by 1,2 order by 3 desc)
## 2. now group  them by the actors,so that we have a separate row for each actor .As we sorted in desc before in (1)
##    the directors with whom actors did most films will appear as the top row on each actors group.
## 3. now fiter out those rows from 2. which has director name as 'yash chopra'

query = """
        select B.act,B.direc,B.cnt from
            (select A.act,A.direc,A.cnt from
                (select actor.name as act,dir.name as direc,count(distinct actor.MID) as cnt
                from
                    (select m.MID,p.name from Movie m join M_director md using(MID) join Person p using(PID))dir
                    join
                    (select m.MID,p.name from Movie m join M_cast mc using(MID) join Person p using(PID))actor using (MID)
             group by 1,2 order by 3 desc)A group by 1)B
         where lower(B.direc) like '%yash chopra%' order by 3 desc

        """

q8 = pd.read_sql_query(query, conn)
print(q8.shape)
q8.head()
```

```
(98, 3)
Wall time: 1.85 s
```

In [84]:

```
print(q8.head(6))
print('*'*50)
print('Number of rows ',len(q8))
```

```
                act        direc  cnt
0       Jagdish Raj  Yash Chopra   11
1  Manmohan Krishna  Yash Chopra   10
2          Iftekhar  Yash Chopra    9
3     Shashi Kapoor  Yash Chopra    7
4     Rakhee Gulzar  Yash Chopra    5
5    Waheeda Rehman  Yash Chopra    5
**************************************************
Number of rows  98
```

**Q9 --- 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 [6]:

```
%%time
# Write your sql query below
## Approach :
##  (1) -- extracts all the movies by sharukh khan
# (1) --select distinct mc.MID from M_Cast mc join Person p using(PID) where lower(p.name) = 'sharukh khan'
## (2) ---extracts all the co-stars(aka sharukh 1) of shahrukh khan (aka sharukh 0)
# (2) --select distinct p.PID from M_Cast mc join Person p where mc.MID in (1) and  lower(p.name) != 'sharukh khan'
## (3)---extracts all the movies by co-stars(aka sharukh 1) of shahrukh khan (aka sharukh 0)
#(3) --select distinct mc.MID from M_Cast mc join Person p where p.PID in (2)
# (4)--## extracts all the co-stars(aka sharukh 2) of sharukh 1
#(4) --select distinct p.PID from M_Cast mc join Person p using(PID) where mc.MID in (3) and p.PID not in (2)

query = """ select distinct p.name from M_Cast mc join Person p using(PID)

            where mc.MID in

            (select distinct mc.MID
                from
                    M_Cast mc
                join
                    Person p using(PID)
                where p.PID in
                    (select distinct p.PID
                        from
                            M_Cast mc
                        join Person p using(PID)
                    where mc.MID in
                        (select distinct mc.MID
                            from
                                M_Cast mc
                            join Person p using(PID)
                        where lower(p.name) = 'shah rukh khan') and lower(p.name) != 'shah rukh khan'))

            and p.PID not in

            (select distinct p.PID
                from
                    M_Cast mc
                join Person p using(PID)
             where mc.MID in
                (select distinct mc.MID
                     from
                         M_Cast mc
                     join Person p using(PID)
                 where lower(p.name) = 'shah rukh khan') and  lower(p.name) != 'shah rukh khan')

            and lower(p.name) != 'shah rukh khan'


        """

q9 = pd.read_sql_query(query, conn)
print(q9.shape)
q9.head()
```

```
(24308, 1)
Wall time: 1.41 s
```

In [8]:

```
print(q9.head(5))
print('*'*50)
print('Number of rows ',len(q9))
```

```
                   Name
0          Freida Pinto
1           Rohan Chand
2          Damian Young
3       Waris Ahluwalia
4  Caroline Christl Long
**************************************************
Number of rows  24308
```