SQL Assignment

In [2]:
```
import pandas as pd
import sqlite3
```

In [3]:
```
conn = sqlite3.connect("Db-IMDB-Assignment.db")
```

Sample Code

In [4]:
```
%%time
# Write your sql query below

query = '''
        SELECT TRIM(Movie.title) AS 'Movie_Name'
        FROM Movie
        WHERE Movie.rating < 3
        '''



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

```
(85, 1)
Wall time: 402 ms
```

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 [20]:
```
%%time
# Write your sql query below

query = """

       SELECT p.name AS director_name,m.title AS movie_name,(CAST(SUBSTR(TRIM(m.year),-4)AS INT)) AS _year
       FROM
       movie m JOIN m_director d ON m.mid=d.mid
       JOIN person p ON d.pid=p.pid
       WHERE((_year%4=0 AND _year%100<>0)
       OR _year%400=0)
       AND
       m.mid IN (SELECT mg.mid from m_genre mg where mg.gid
       IN(SELECT g.gid FROM genre g WHERE TRIM(g.name) LIKE'%Comedy%'))









        """

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

```
(232, 3)
     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
Wall time: 79 ms
```

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

In [21]:
```
%%time
# Write your sql query below

query = """
       SELECT p.name FROM person p
       WHERE TRIM(p.pid)
       IN
       (SELECT  TRIM(c.pid) FROM m_cast c WHERE TRIM(c.mid)
       IN
       (SELECT TRIM(m.mid) FROM movie m WHERE TRIM(m.title)='Anand'))


        """

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

```
(17, 1)
                Name
0   Amitabh Bachchan
1      Rajesh Khanna
2      Sumita Sanyal
3         Ramesh Deo
4          Seema Deo
Wall time: 24 ms
```

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

In [7]:
```
%%time
# Write your sql query below

query = """
    SELECT p.name FROM person p  WHERE TRIM(p.pid)
      IN
      ( SELECT trim(t1.pid) from
    (SELECT DISTINCT(c1.pid)  from m_cast c1 where TRIM(c1.mid) IN (SELECT  TRIM(m.mid) FROM movie m
      WHERE
      cast(substr(trim(m.year),-4)as int) <1970 )) as t1

      join

      (SELECT DISTINCT(c2.pid) from m_cast c2 where TRIM(c2.mid) IN (SELECT  TRIM(m.mid) FROM movie m
      WHERE
      cast(substr(trim(m.year),-4)as int) >1990 )) as  t2

      ON TRIM(t1.pid)=TRIM(t2.pid))




        """

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

```
(300, 1)
                Name
0       Rishi Kapoor
1   Amitabh Bachchan
2             Asrani
3       Zohra Sehgal
4    Parikshat Sahni
Wall time: 23.9 s
```

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 [8]:
```
%%time
# Write your sql query below

query = """
        SELECT p.name,t.movie_count
        FROM
        person p
        JOIN
        (SELECT COUNT(d.mid) movie_count,d.pid FROM m_director d GROUP BY d.pid
        Having movie_count>=10 ) AS t
        ON TRIM(p.pid)=TRIM(t.pid)
        ORDER BY movie_count DESC


        """

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

```
(58, 2)
               Name  movie_count
0      David Dhawan           39
1      Mahesh Bhatt           35
2   Ram Gopal Varma           30
3      Priyadarshan           30
4      Vikram Bhatt           29
Wall time: 786 ms
```

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

In [22]:
```
%%time
# Write your sql query below

query = """
        SELECT m.year,COUNT(m.mid) FROM movie m
        WHERE TRIM(m.mid) NOT IN
        (SELECT Distinct(TRIM(c.mid)) FROM m_cast c
        WHERE (c.pid IS NULL
        OR
        TRIM(c.pid) IN
        (SELECT TRIM(p.pid) FROM person p
        WHERE TRIM(p.gender) <>'Female'  or TRIM(p.gender) IS NULL)))
        GROUP BY cast(substr(trim(m.year),-4)as int)


        """

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

```
(4, 2)
     year  COUNT(m.mid)
0    1939             1
1    1999             1
2    2000             1
3  I 2018             1
Wall time: 93 ms
```

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 [23]:
```
%%time
# Write your sql query below

query = """
      SELECT m1.year,count(m1.mid) total_movies,(((y.q*1.0)/(count(m1.mid)*1.0))*100) perc_onlyFemalemovie FR
OM
       movie m1  join
       (SELECT m.year,COUNT(m.mid) q FROM movie m
        WHERE TRIM(m.mid) NOT IN
        (SELECT Distinct(TRIM(c.mid)) FROM m_cast c
        WHERE (c.pid IS NULL
        OR
        TRIM(c.pid) IN
        (SELECT TRIM(p.pid) FROM person p
        WHERE TRIM(p.gender) <>'Female'  or TRIM(p.gender) IS NULL)))
        GROUP BY cast(substr(trim(m.year),-4)as int)) y
        on cast(substr(trim(m1.year),-4)as int)=cast(substr(trim(y.year),-4)as int)

        GROUP BY cast(substr(trim(m1.year),-4)as int)



        """


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

```
(4, 3)
   year  total_movies  perc_onlyFemalemovie
0  1939             2             50.000000
1  1999            66              1.515152
2  2000            64              1.562500
3  2018           104              0.961538
Wall time: 100 ms
```

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 [24]:
```
%%time
# Write your sql query below

query = """
    SELECT m.title,t.cast_size FROM movie m
       JOIN

       (SELECT COUNT(DISTINCT(c.pid)) cast_size,c.mid FROM m_cast c GROUP BY TRIM(c.mid)
      HAVING cast_size=(select max(j.mp) from
       (SELECT COUNT(DISTINCT(pid)) mp FROM m_cast GROUP BY mid)j))t

       ON TRIM(m.mid)=TRIM(t.mid)
        """

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

```
(1, 2)
           title  cast_size
0  Ocean's Eight        238
Wall time: 161 ms
```

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 [25]:
```
%%time
# Write your sql query below

query = """

        Select count(x.yr) total_movies,x.yr  decade_start,x.yr+9 decade_end

        FROM movie m

        JOIN



        (select DISTINCT(cast(substr(trim(year),-4)as int)) yr from movie )x

        ON cast(substr(trim(m.year),-4)as int)>=x.yr AND m.year<=x.yr+9

        GROUP BY x.yr

        order by total_movies DESC

        LIMIT 1

        """
#select DISTINCT(cast(substr(trim(year),-4)as int)) yr from movie order by yr desc
q7 = pd.read_sql_query(query, conn)
print(q7.shape)
print(q7)
```

```
(1, 3)
   total_movies  decade_start  decade_end
0          1128          2008        2017
Wall time: 97 ms
```

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

In [26]:
```
%%time
# Write your sql query below

query = """

    SELECT name FROM person p
    WHERE TRIM(pid) IN
    (
   SELECT TRIM(YRF.yactors) FROM
    (
     SELECT  count(m.mid)ymovies,TRIM(m.pid) yactors FROM m_cast m
       WHERE TRIM(m.mid) IN
       (
       SELECT TRIM(d.mid) FROM m_director d
       WHERE TRIM(d.pid) =
       (
       SELECT TRIM(p.pid) FROM person p
       WHERE TRIM(p.name)='Yash Chopra'
       )
       )
       group by trim(m.pid)
    )YRF



     LEFT JOIN

    (
    SELECT max(other.otherdirector)nonyashmoviecount,other.otherperson nonyashactor
    FROM
    (
       SELECT count(x.nymovies)otherdirector,x.nyperson otherperson FROM
         (
         SELECT DISTINCT
     TRIM(m1.mid) nymovies,TRIM(m1.pid) nyperson FROM m_cast m1
     WHERE TRIM(m1.PID) IN
     (

     SELECT  DISTINCT(TRIM(m.pid)) yactors FROM m_cast m
       WHERE TRIM(m.mid) IN
       (
       SELECT DISTINCT(TRIM(d.mid)) FROM m_director d
```

```
       WHERE TRIM(d.pid) =
       (
       SELECT TRIM(p.pid) FROM person p
       WHERE TRIM(p.name)='Yash Chopra'
       )
       )
       )
       AND m1.PID IS NOT NULL
       ) x

       JOIN m_director d1

       ON x.nymovies=TRIM(d1.mid)

       where TRIM(d1.pid)!=(
       SELECT TRIM(p.pid) FROM person p
       WHERE TRIM(p.name)='Yash Chopra'
       )

       GROUP BY x.nyperson,TRIM(d1.pid)

      )other
      GROUP BY other.otherperson
      )NYRF

      ON  YRF.yactors=NYRF.nonyashactor

     WHERE YRF.ymovies>=NYRF.nonyashmoviecount OR NYRF.nonyashmoviecount IS NULL

     )




        """
#GROUP BY TRIM(m.pid)
#       order by ymovies desc
q8 = pd.read_sql_query(query, conn)
print(q8.shape)
print(q8.head())
```

```
(245, 1)
                Name
0     Sharib Hashmi
1   Kulbir Badesron
2       Gurdas Maan
3   Parikshat Sahni
4     Claire Ashton
Wall time: 107 ms
```

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 [27]:
```
%%time
# Write your sql query below

query = """


     SELECT p.name from person p
     WHERE TRIM(p.pid) IN
     (SELECT DISTINCT(TRIM(c4.pid)) s2 FROM m_cast c4
     WHERE TRIM(c4.mid) IN

     (SELECT DISTINCT(TRIM(c3.mid)) FROM m_cast c3
     WHERE TRIM(c3.pid) IN


    (SELECT DISTINCT(TRIM(c2.pid)) s1 FROM m_cast c2
             WHERE c2.mid  IN


     (SELECT DISTINCT(TRIM(c1.mid)) FROM m_cast c1 WHERE
             TRIM(c1.pid)=

     (SELECT TRIM(pid)t FROM  person where TRIM(name)='Shah Rukh Khan') AND c1.mid IS NOT NULL)

     AND TRIM(c2.pid)!=(SELECT TRIM(pid)t FROM  person where TRIM(name)='Shah Rukh Khan') AND c2.pid IS NOT N
ULL)



     AND TRIM(c3.mid) NOT IN (SELECT DISTINCT(TRIM(c1.mid)) FROM m_cast c1 WHERE
             TRIM(c1.pid)=

     (SELECT TRIM(pid)t FROM  person where TRIM(name)='Shah Rukh Khan')) AND c3.MID IS NOT NULL)



     AND TRIM(c4.pid) NOT IN

     (SELECT DISTINCT(TRIM(c2.pid)) s1 FROM m_cast c2
             WHERE c2.mid  IN
```

```
    (SELECT DISTINCT(TRIM(c1.mid)) FROM m_cast c1 WHERE
            TRIM(c1.pid)=

    (SELECT TRIM(pid)t FROM  person where TRIM(name)='Shah Rukh Khan'))

    AND TRIM(c2.pid)!=(SELECT TRIM(pid)t FROM  person where TRIM(name)='Shah Rukh Khan')))









       """



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

```
(25698, 1)
                    Name
0            Freida Pinto
1             Rohan Chand
2            Damian Young
3         Waris Ahluwalia
4   Caroline Christl Long
Wall time: 268 ms
```