In [1]:

```
print('databse')
```

```
databse
```

In [2]:

```
import pandas as pd
import sqlite3
conn = sqlite3.connect("C:/Users/HARRY/Desktop/ML/Applied ai/Assinments/22 assinment/Db-IMDB.db")
```

(SQL) ASSINMENT 22

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 [ ]:

```
#https://www.sqlbook.com/sql-string-functions/sql-trim-functions-purpose-syntax-and-common-uses/
#https://github.com/jasonMatney/SQLWork/blob/master/wash.sql
# left join because we want all the directors
sql_query = """


  SELECT  p.Name,m.title,m.year   FROM Person p
  LEFT JOIN M_Director c ON LTRIM(c.PID)=LTRIM(p.PID)
  LEFT JOIN Movie m ON c.MID=m.MID
  LEFT JOIN M_Genre mg ON m.MID=mg.MID
  LEFT JOIN Genre g ON mg.GID=g.GID
  GROUP BY p.PID having (((m.year % 4 = 0 and m.year % 100 <> 0) OR m.year % 400 = 0)) AND LTRIM(R
TRIM((replace(g.Name, ',', '') )))='Comedy'




"""
df= pd.read_sql_query(sql_query, conn)
df
```

2.

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

In [ ]:

```
#(((m.year % 4 = 0 and m.year % 100 <> 0) OR m.year % 400 = 0))
sql_query = """

  SELECT Name FROM Person WHERE PID IN
  (SELECT LTRIM(c.PID) FROM M_cast c
  JOIN Movie m on c.MID=m.MID WHERE m.title='Anand')




"""
df= pd.read_sql_query(sql_query, conn)
df
```

3.

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

In [ ]:

```
#For reference
#https://github.com/hafizusman/csep544_/blob/master/01Hw/hw1-queries.sql

sql_query = """


SELECT DISTINCT Name
FROM Person AS A, M_cast AS C1, Movie AS M1, M_cast AS C2, Movie AS M2
WHERE  LTRIM(A.PID)= LTRIM(C1.PID)           AND C1.MID=M1.MID
AND LTRIM(A.PID)= LTRIM(C2.PID)           AND C2.MID=M2.MID
AND M1.year < 1970 AND M2.year > 1990



"""
df= pd.read_sql_query(sql_query, conn)
df
```

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 [ ]:

```
sql_query = """

  SELECT DISTINCT p.Name,count(m.title) ct  FROM Person p
  LEFT JOIN M_Director c ON LTRIM(c.PID)=LTRIM(p.PID)
  LEFT JOIN Movie m ON c.MID=m.MID
  GROUP BY p.PID
  HAVING ct>=10
  ORDER BY ct DESC



"""
df= pd.read_sql_query(sql_query, conn)
df
```

5.

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

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 [ ]:

```
#a-----------------------------------------------


#https://stackoverflow.com/questions/1400078/is-it-possible-to-specify-condition-in-count
# we can use 2 group by i think
# problem of duplicates tomorow i'll solve
sql_query = """
 SELECT m.year, count(p.Name) as Total FROM Movie m

 LEFT JOIN M_Cast me ON  m.MID=me.MID
```

```
 LEFT JOIN Person p ON LTRIM(me.PID)=LTRIM(p.PID)
 GROUP BY m.year HAVING LTRIM(RTRIM(p.Gender))='Female'



"""
df= pd.read_sql_query(sql_query, conn)
df
```

In [ ]:

```
#b.-------------------------------------------------------------------------------



#https://stackoverflow.com/questions/1400078/is-it-possible-to-specify-condition-in-count
sql_query =
"""

 SELECT m.year, count(p2.Names)/count(p.Names) as Female_percentage,count(p.Names) as Total
 FROM Movie as m,Movie as m2

 LEFT JOIN M_Cast me ON  m.MID=me.MID
 LEFT JOIN Person p ON LTRIM(me.PID)=LTRIM(p.PID)

 LEFT JOIN M_Cast me2 ON  m2.MID=me2.MID
 LEFT JOIN Person p2 ON LTRIM(me2.PID)=LTRIM(p2.PID)

 GROUP BY m.year, GROUP BY m2.year  HAVING LTRIM(RTRIM(p2.Gender))='Female'

"""
df= pd.read_sql_query(sql_query, conn)
df
```

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 [ ]:

```
#https://stackoverflow.com/questions/1400078/is-it-possible-to-specify-condition-in-count
#https://stackoverflow.com/questions/2436820/can-i-do-a-maxcount-in-sql
#https://github.com/jasonMatney/SQLWork/blob/master/wash.sql



sql_query = """


 SELECT  m.title,COUNT( DISTINCT p.Name)  as Cast_size from Person p
 LEFT JOIN M_Cast mc ON LTRIM(mc.PID)=LTRIM(p.PID)
 LEFT JOIN Movie m ON Mc.MID=m.MID
 GROUP BY m.MID
 HAVING COUNT(p.Name) =  (SELECT COUNT(p.Name) FROM Person p
 LEFT JOIN M_Cast mc ON LTRIM(mc.PID)=LTRIM(p.PID)
 LEFT JOIN Movie m ON Mc.MID=m.MID
 GROUP BY m.MID ORDER BY COUNT( DISTINCT p.Name) DESC LIMIT 1)


"""
df= pd.read_sql_query(sql_query, conn)
df
```

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 [ ]:

```
#FOR REFERENCE
#https://stackoverflow.com/questions/51609285/sql-query-for-find-the-decade-with-the-largest-numbe
r-of-records



sql_query = """


SELECT y.year AS decade_start, y.year + 9 AS decade_end,
       count(*) AS num_movies
FROM (SELECT DISTINCT year from Movie) y JOIN
     Movie m
     ON m.year >= y.year AND m.year < y.year + 10
GROUP BY y.year
ORDER BY count(*) DESC
limit 1


"""
df= pd.read_sql_query(sql_query, conn)
df
```

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 [ ]:

```
#For reference:
#https://www.coursehero.com/file/p7mfaba/15-Find-the-actors-who-were-never-unemployed-for-more-tha
n-3-years-at-a-stretch/

sql_query = """



SELECT  PID, Name FROM Person WHERE PID NOT in
(
SELECT DISTINCT PID FROM M_Cast AS C1 NATURAL JOIN Movie as M1
WHERE EXISTS
(
SELECT MID FROM M_Cast as C2 NATURAL JOIN Movie as M2

WHERE  LTRIM(C1.PID) =  LTRIM(C2.PID)     and (M2.year - 3) > M1.year

AND NOT EXISTS(SELECT MID FROM M_Cast AS C3 NATURAL JOIN Movie AS

M3 WHERE LTRIM(C1.PID) =  LTRIM(C3.PID) AND M1.year < M3.year AND  M3.year < M2.year)))



"""
df= pd.read_sql_query(sql_query, conn)
df
```

9.

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

In [ ]:

```
#For reference:
```

```
#For reference:
#https://www.coursehero.com/file/p7mfaba/15-Find-the-actors-who-were-never-unemployed-for-more-tha
n-3-years-at-a-stretch/
#https://www.geeksforgeeks.org/sql-with-clause/
#https://www.dofactory.com/sql/subquery

sql_query = """

SELECT   P1.PID,   P1.Name, COUNT (Movie.MID)  AS movies
             WITH yc FROM Person as P1
            NATURAL JOIN M_Cast NATURAL JOIN Movie
JOIN M_Director ON (Movie.MID = M_Director.MID)

JOIN Person as P2 ON (M_Director.PID = P2.PID)
             WHERE P2.Name = 'Yash Chopra' GROUP BY P1.PID
             HAVING COUNT(Movie.MID) >  ALL
(SELECT COUNT(Movie.MID) FROM Person AS P3
             NATURAL JOIN M_Cast NATURAL JOIN Movie
JOIN M_Director ON (Movie.MID = M_Director.MID)
JOIN Person AS P4 ON  (M_Director.PID = P4.PID)
         WHERE P1.PID = P3.PID AND P4.Name != 'Yash Chopra'
        GROUP BY P4.PID)
        ORDER BY movies WITH  yc DESC

"""
df= pd.read_sql_query(sql_query, conn)
df
```

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 [ ]:

```
#For reference:
#https://www.coursehero.com/file/p7mfaba/15-Find-the-actors-who-were-never-unemployed-for-more-tha
n-3-years-at-a-stretch/
#https://www.geeksforgeeks.org/sql-with-clause/
#https://www.dofactory.com/sql/subquery


sql_query = """



SELECT DISTINCT  PID, Name FROM Person
             NATURAL JOIN M_Cast
             WHERE Name <> 'Shah Rukh Khan'
AND MID IN (SELECT MID FROM M_Cast WHERE PID IN
(SELECT PID FROM Person NATURAL JOIN
 M_Cast WHERE Name <> 'Shah Rukh Khan' AND MID IN
 (SELECT  MID FROM Person NATURAL
 JOIN M_Cast
 WHERE Name = 'Shah Rukh Khan')))
 AND PID NOT IN
 (SELECT PID FROM Person
 NATURAL JOIN M_Cast
 WHERE Name <> 'Shah Rukh Khan'
 AND MID IN (SELECT MID FROM Person
 NATURAL JOIN M_Cast WHERE Name = 'Shah Rukh Khan'))









"""
df= pd.read_sql_query(sql_query, conn)
```

```
df
```