In [1]:

```
import pandas as pd
import sqlite3

from IPython.display import display, HTML
```

In [2]:

```
conn = sqlite3.connect("Db-IMDB-Assignment.db")
```

1

In [3]:

```
%%time
def grader_1(q1):
    q1_results  = pd.read_sql_query(q1,conn)
    print(q1_results)
    print(q1_results.shape)
    assert (q1_results.shape == (232,3))


query1="""SELECT M.title,P.Name,CAST(SUBSTR(TRIM(M.year),-4) AS INTEGER) Year FROM Person
P INNER JOIN M_Director MD ON P.PID=MD.PID INNER JOIN Movie M ON M.MID=MD.MID INNER JOIN
            M_GENRE MG ON MG.MID=M.MID INNER JOIN Genre G
            ON MG.GID=G.GID
            WHERE (lower(G.Name) LIKE '%comedy%') AND ((CAST(SUBSTR(TRIM(M.year),-4) AS
INTEGER)%4 = 0 AND CAST(SUBSTR(TRIM(M.year),-4) AS INTEGER)%100 != 0) OR CAST(SUBSTR(TRIM
(M.year),-4) AS INTEGER)%400 = 0)"""
grader_1(query1)
```

```
                                title                  Name  Year
0                           Mastizaade          Milap Zaveri  2016
1    Harold & Kumar Go to White Castle         Danny Leiner  2004
2                    Gangs of Wasseypur       Anurag Kashyap  2012
3          Around the World in 80 Days          Frank Coraci  2004
4                The Accidental Husband         Griffin Dunne  2008
..                                  ...                   ...   ...
227                        Let's Enjoy  Siddharth Anand Kumar  2004
228                            Sathyam      Amma Rajasekhar  2008
229                     Tandoori Love         Oliver Paulus  2008
230                      Le Halua Le           Raja Chanda  2012
231                Raja Aur Rangeeli      K.S. Prakash Rao  1996

[232 rows x 3 columns]
(232, 3)
Wall time: 187 ms
```

2

In [4]:

```
%%time
def grader_2(q2):
    q2_results  = pd.read_sql_query(q2,conn)
    print(q2_results)
    print(q2_results.shape)
    assert (q2_results.shape == (17,1))


query2 = "SELECT Name FROM Person WHERE PID IN (SELECT LTRIM(PID) FROM M_Cast WHERE MID I
N(SELECT MID FROM Movie WHERE title LIKE 'Anand' AND year=1971))"
grader_2(query2)
```

```
                 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
(17, 1)
Wall time: 96.5 ms
```

3

In [5]:

```
%%time

def grader_3a(query_less_1970, query_more_1990):
    q3_a = pd.read_sql_query(query_less_1970,conn)
    print(q3_a.shape)
    q3_b = pd.read_sql_query(query_more_1990,conn)
    print(q3_b.shape)
    return (q3_a.shape == (4942,1)) and (q3_b.shape == (62570,1))

query_less_1970 ="""
Select p.PID from Person p
inner join
(
    select trim(mc.PID) PD, mc.MID from M_cast mc
where mc.MID
in
(
    select mv.MID from Movie mv where CAST(SUBSTR(mv.year,-4) AS Integer)<1970
)
) r1
on r1.PD=p.PID
"""
query_more_1990 ="""
Select p.PID from Person p
inner join
(
    select trim(mc.PID) PD, mc.MID from M_cast mc
where mc.MID
in
(
    select mv.MID from Movie mv where CAST(SUBSTR(mv.year,-4) AS Integer)>1990
)
) r1
on r1.PD=p.PID """
print(grader_3a(query_less_1970, query_more_1990))

# using the above two queries, you can find the answer to the given question
```

```
(4942, 1)
(62570, 1)
True
Wall time: 449 ms
```

In [6]:

```
%%time
def grader_3(q3):
    q3_results  = pd.read_sql_query(q3,conn)
    print(q3_results.head(10))
    print(q3_results.shape)
    assert (q3_results.shape == (300,1))

query3 = """
Select DISTINCT p.Name from Person p
inner join
(
    select trim(mc.PID) PD, mc.MID from M_cast mc
where mc.MID
in
(
    select mv.MID from Movie mv where CAST(SUBSTR(mv.year,-4) AS Integer)<1970
)
) r1
on r1.PD=p.PID
WHERE p.PID IN (Select p.PID from Person p
inner join
(
    select trim(mc.PID) PD, mc.MID from M_cast mc
where mc.MID
in
(
    select mv.MID from Movie mv where CAST(SUBSTR(mv.year,-4) AS Integer)>1990
)
) r1
on r1.PD=p.PID)"""
grader_3(query3)
```

```
              Name
0   Waheeda Rehman
1    Johnny Walker
2          Mehmood
3            Ratna
4   Rajendra Kumar
5         Iftekhar
6        Raj Mehra
7     Lalita Pawar
8   Achala Sachdev
9       Sunil Dutt
(300, 1)
Wall time: 451 ms
```

4

In [7]:

```
%%time

def grader_4a(query_4a):
    query_4a = pd.read_sql_query(query_4a,conn)
    print(query_4a.head(10))
    return (query_4a.shape == (1462,2))


#query_4a ="""SELECT PID Director_Id,COUNT(MID) Movie_Count FROM M_Director GROUP BY PID
ORDER BY Movie_Count """

query_4a ="""SELECT PID Director_Id,COUNT(MID) Movie_Count FROM M_Director GROUP BY PID "
""
print(grader_4a(query_4a))
```

```
  Director_Id  Movie_Count
0   nm0000180            1
1   nm0000187            1
2   nm0000229            1
3   nm0000269            1
4   nm0000386            1
```

```
5    nm0000487             2
6    nm0000965             1
7    nm0001060             1
8    nm0001162             1
9    nm0001241             1
True
Wall time: 17.6 ms
```

In [8]:

```
%%time
def grader_4(q4):
    q4_results  = pd.read_sql_query(q4,conn)
    print(q4_results.head(10))
    assert (q4_results.shape == (58,2))

query4 = """SELECT P.Name,COUNT(MID) C FROM Person P INNER JOIN M_Director MD
            ON P.PID=MD.PID GROUP BY MD.PID HAVING C>=10 ORDER BY C DESC"""
grader_4(query4)
```

```
                    Name   C
0            David Dhawan  39
1            Mahesh Bhatt  35
2         Ram Gopal Varma  30
3            Priyadarshan  30
4            Vikram Bhatt  29
5   Hrishikesh Mukherjee  27
6             Yash Chopra  21
7          Shakti Samanta  19
8         Basu Chatterjee  19
9            Subhash Ghai  18
Wall time: 66.1 ms
```

5 a

In [9]:

```
def grader_5aa(query_5aa):
    query_5aa = pd.read_sql_query(query_5aa,conn)
    print(query_5aa.head(10))
    return (query_5aa.shape == (8846,3))

query_5aa ="""SELECT MC.MID,P.Gender,COUNT(*) FROM Person P INNER JOIN M_Cast MC ON TRIM(
MC.PID)=P.PID GROUP BY MC.MID,P.Gender"""

print(grader_5aa(query_5aa))
```

```
         MID  Gender  COUNT(*)
0  tt0021594    None         1
1  tt0021594  Female         3
2  tt0021594    Male         5
3  tt0026274    None         2
4  tt0026274  Female        11
5  tt0026274    Male         9
6  tt0027256    None         2
7  tt0027256  Female         5
8  tt0027256    Male         8
9  tt0028217  Female         3
True
```

In [10]:

```
def grader_5ab(query_5ab):
    query_5ab = pd.read_sql_query(query_5ab,conn)
    print(query_5ab.head(10))
    return (query_5ab.shape == (3469, 3))

query_5ab ="""SELECT MC.MID,P.Gender,COUNT(*) C FROM Person P INNER JOIN M_Cast MC ON TRI
M(MC.PID)=P.PID GROUP BY MC.MID,P.Gender HAVING P.Gender='Male' """
```

```
print(grader_5ab(query_5ab))
```

```
         MID Gender   C
0  tt0021594   Male   5
1  tt0026274   Male   9
2  tt0027256   Male   8
3  tt0028217   Male   7
4  tt0031580   Male  27
5  tt0033616   Male  46
6  tt0036077   Male  11
7  tt0038491   Male   7
8  tt0039654   Male   6
9  tt0040067   Male  10
True
```

In [11]:

```
%%time
def grader_5a(q5a):
    q5a_results  = pd.read_sql_query(q5a,conn)
    print(q5a_results.head(10))
    assert (q5a_results.shape == (4,2))

query5a = """  WITH T1 AS(
 SELECT MC.MID MOVIE_ID,COUNT(*) C FROM Person P INNER JOIN M_Cast MC ON TRIM(MC.PID)=P.P
ID GROUP BY MC.MID,P.GENDER HAVING P.GENDER='Female'),
 T2 AS(
 SELECT MC.MID MOVIE_ID,COUNT(*) C FROM Person P INNER JOIN M_Cast MC ON TRIM(MC.PID)=P.P
ID GROUP BY MC.MID)
 SELECT SUBSTR(TRIM(year),-4) YEAR ,COUNT(*) FEMALE_CAST FROM Movie M WHERE MID IN (SELEC
T T1.MOVIE_ID FROM T1 INNER JOIN T2 ON T1.MOVIE_ID=T2.MOVIE_ID WHERE T1.C=T2.C) GROUP BY
YEAR
"""
grader_5a(query5a)
```

```
   YEAR  FEMALE_CAST
0  1939            1
1  1999            1
2  2000            1
3  2018            1
Wall time: 453 ms
```

5b

In [12]:

```
%%time
def grader_5b(q5b):
    q5b_results  = pd.read_sql_query(q5b,conn)
    print(q5b_results.head(10))
    assert (q5b_results.shape == (4,3))

query5b = """ WITH T1 AS(
SELECT MC.MID MOVIE_ID,COUNT(*) C FROM Person P INNER JOIN M_Cast MC ON TRIM(MC.PID)=P.PI
D GROUP BY MC.MID,P.GENDER HAVING P.GENDER='Female'),
T2 AS(
SELECT MC.MID MOVIE_ID,COUNT(*) C FROM Person P INNER JOIN M_Cast MC ON TRIM(MC.PID)=P.PI
D GROUP BY MC.MID),
T3 AS(
SELECT SUBSTR(TRIM(year),-4) YEAR ,COUNT(*) FEMALE_CAST FROM Movie M WHERE MID IN (SELECT
T1.MOVIE_ID FROM T1 INNER JOIN T2 ON T1.MOVIE_ID=T2.MOVIE_ID WHERE T1.C=T2.C) GROUP BY SU
BSTR(TRIM(year),-4)),
T4 AS(
SELECT SUBSTR(TRIM(M.YEAR),-4) YEAR,T3.FEMALE_CAST FEMALE_CAST,COUNT(*) TOTAL_MOVIE FROM
T3 INNER JOIN MOVIE M ON T3.YEAR=SUBSTR(TRIM(M.YEAR),-4) GROUP BY SUBSTR(TRIM(M.YEAR),-4)
,T3.FEMALE_CAST)
SELECT YEAR,CAST(FEMALE_CAST AS FLOAT) / CAST(TOTAL_MOVIE AS FLOAT)AS PERCENT,TOTAL_MOVIE
FROM T4"""
grader_5b(query5b)
```

```
   YEAR   PERCENT  TOTAL_MOVIE
0  1939  0.500000            2
1  1999  0.015152           66
2  2000  0.015625           64
3  2018  0.009615          104
Wall time: 485 ms
```

6

In [13]:

```
%%time
def grader_6(q6):
    q6_results  = pd.read_sql_query(q6,conn)
    print(q6_results.head(10))
    assert (q6_results.shape == (3473, 2))

query6 = """SELECT M.title,COUNT(DISTINCT(MC.PID)) C FROM Movie M INNER JOIN M_Cast MC ON
M.MID=MC.MID GROUP BY MC.MID  ORDER  BY C DESC"""
grader_6(query6)
```

```
                        title    C
0                Ocean's Eight  238
1                     Apaharan  233
2                         Gold  215
3              My Name Is Khan  213
4  Captain America: Civil War  191
5                     Geostorm  170
6                      Striker  165
7                         2012  154
8                       Pixels  144
9       Yamla Pagla Deewana 2  140
Wall time: 351 ms
```

7

In [14]:

```
%%time
def grader_7a(q7a):
    q7a_results  = pd.read_sql_query(q7a,conn)
    print(q7a_results)
    print(q7a_results.shape)
    assert (q7a_results.shape == (78, 2))

query7a = """SELECT SUBSTR(Year,-4) Movie_Year,COUNT(Year) Total_Movies FROM Movie GROUP
BY SUBSTR(Year,-4) ORDER BY Year"""
grader_7a(query7a)
```

```
   Movie_Year  Total_Movies
0        1931             1
1        1936             3
2        1939             2
3        1941             1
4        1943             1
..        ...           ...
73       2016           129
74       2017           126
75       2018           104
76       1964            15
77       2009           109

[78 rows x 2 columns]
(78, 2)
Wall time: 18.4 ms
```

In [15]:

```
%%time
def grader_7b(q7b):
    q7b_results  = pd.read_sql_query(q7b,conn)
    print(q7b_results.head(10))
    assert (q7b_results.shape == (713, 4))

query7b = """WITH T1 AS(
SELECT CAST(SUBSTR(YEAR,-4) AS DECIMAL) Movie_Year_1,COUNT(*) Total_Movies_1 FROM Movie G
ROUP BY Movie_Year_1 ORDER BY Year)
SELECT * FROM T1 T11,T1 T12 WHERE T12.Movie_Year_1<=T11.Movie_Year_1+ 9 AND T12.Movie_Yea
r_1>=T11.Movie_Year_1 """
grader_7b(query7b)
# if you see the below results the first movie year is less than 2nd movie year and
# 2nd movie year is less or equal to the first movie year+9

# using the above query, you can write the answer to the given question
```

```
   Movie_Year_1  Total_Movies_1  Movie_Year_1  Total_Movies_1
0          1931               1          1931               1
1          1931               1          1936               3
2          1931               1          1939               2
3          1936               3          1936               3
4          1936               3          1939               2
5          1936               3          1941               1
6          1936               3          1943               1
7          1939               2          1939               2
8          1939               2          1941               1
9          1939               2          1943               1
Wall time: 20.5 ms
```

In [16]:

```
%%time
def grader_7(q7):
    q7_results  = pd.read_sql_query(q7,conn)
    print(q7_results.head(10))
    assert (q7_results.shape == (1, 2))

query7 = """ select year_A || " to "|| year_B Decade, sum(count_B) dec_t from
(select A.year_n year_A ,A.c_m count_A, B.year_n+9 year_B,B.c_m count_B from
(select CAST(SUBSTR(TRIM(year),-4) AS INTEGER) year_n, count(*) c_m
from movie
group by year_n) A Join
(select CAST(SUBSTR(TRIM(year),-4) AS INTEGER) year_n, count(*) c_m
from movie
group by year_n) B
on A.year_n = A.year_n
where
A.year_n+9 >= B.year_n and B.year_n>= A.year_n
order by A.year_n)
group by year_A
order by dec_t
Desc
LIMIT 1"""
grader_7(query7)
# if you check the output we are printinng all the year in that decade, its fine you can
print 2008 or 2008-2017
```

```
         Decade  dec_t
0  2008 to 2017   1203
Wall time: 29.5 ms
```

8

In [17]:

```
%%time
def grader_8a(q8a):
    q8a_results  = pd.read_sql_query(q8a,conn)
```

```
    print(q8a_results.head(10))
    print(q8a_results.shape)
    assert (q8a_results.shape == (73408, 3))

query8a = """ SELECT MD.PID,MC.PID,COUNT(*) FROM  M_DIRECTOR MD INNER JOIN M_CAST MC ON M
D.MID=MC.MID GROUP BY MD.PID,MC.PID"""
grader_8a(query8a)

# using the above query, you can write the answer to the given question
```

```
         PID         PID  COUNT(*)
0  nm0000180   nm0000027         1
1  nm0000180   nm0001114         1
2  nm0000180   nm0001919         1
3  nm0000180   nm0006762         1
4  nm0000180   nm0030062         1
5  nm0000180   nm0038970         1
6  nm0000180   nm0051856         1
7  nm0000180   nm0085966         1
8  nm0000180   nm0097889         1
9  nm0000180   nm0125497         1
(73408, 3)
Wall time: 625 ms
```

In [18]:

```
%%time

def grader_8(q8):
    q8_results  = pd.read_sql_query(q8,conn)
    print(q8_results.head(10))
    print(q8_results.shape)
    assert (q8_results.shape == (245, 2))

total_num_movies = "SELECT PERSON.NAME as dir_name, M_CAST.PID as actor_id ,COUNT(*) as c
ount FROM MOVIE JOIN M_DIRECTOR ON MOVIE.MID = M_DIRECTOR.MID JOIN M_CAST ON M_CAST.MID =
MOVIE.MID JOIN PERSON ON PERSON.PID = M_DIRECTOR.PID GROUP BY TRIM(M_CAST.PID), M_DIRECTO
R.PID"

num_yash_movies = total_num_movies + " HAVING TRIM(PERSON.NAME) IS 'Yash Chopra'"

num_yash_movies_actor_id = "SELECT x.actor_id FROM (" + num_yash_movies + ") as x"

non_yash_movies = "SELECT actor_id, dir_name, MAX(count) as count FROM (" + total_num_mov
ies + ") GROUP BY actor_id HAVING TRIM(dir_name) <> 'Yash Chopra' AND actor_id IN(" + num
_yash_movies_actor_id + ")"

common_actor_yash_and_non_yash = "SELECT PERSON.NAME, x.actor_id as Yash_actors, y.actor_
id as Non_Yash_actors, x.count as yash_count, y.count as no_yash_count FROM (" + num_yash
_movies + ") as x LEFT OUTER JOIN (" + non_yash_movies + ") as y ON x.actor_id = y.actor
_id JOIN PERSON ON PERSON.PID = TRIM(x.actor_id) GROUP BY x.actor_id"

query8 = "SELECT Name, yash_count FROM (" + common_actor_yash_and_non_yash + ") WHERE ya
sh_count >= no_yash_count OR no_yash_count IS NULL ORDER BY yash_count DESC"
grader_8(query8)
```

```
                NAME  yash_count
0         Jagdish Raj          11
1    Manmohan Krishna          10
2            Iftekhar           9
3       Shashi Kapoor           7
4       Rakhee Gulzar           5
5      Waheeda Rehman           5
6            Ravikant           4
7      Achala Sachdev           4
8         Neetu Singh           4
9       Leela Chitnis           3
(245, 2)
Wall time: 1.96 s
```

a

In [19]:

```
%%time
def grader_9a(q9a):
    q9a_results  = pd.read_sql_query(q9a,conn)
    print(q9a_results.head(10))
    print(q9a_results.shape)
    assert (q9a_results.shape == (2382, 1))

query9a = """ SELECT DISTINCT(PID) FROM M_CAST WHERE TRIM(MID) IN (SELECT TRIM(MID) FROM
M_CAST WHERE TRIM(PID) IN(SELECT TRIM(PID) FROM PERSON P WHERE TRIM(NAME)='Shah Rukh Khan
')) AND TRIM(PID) NOT IN(SELECT TRIM(PID) FROM PERSON P WHERE TRIM(NAME)='Shah Rukh Khan'
);"""
grader_9a(query9a)
# using the above query, you can write the answer to the given question

# selecting actors who acted with srk (S1)
# selecting all movies where S1 actors acted, this forms S2 movies list
# selecting all actors who acted in S2 movies, this gives us S2 actors along with S1 acto
rs
# removing S1 actors from the combined list of S1 & S2 actors, so that we get only S2 act
ors
```

```
          PID
0   nm0004418
1   nm1995953
2   nm2778261
3   nm0631373
4   nm0241935
5   nm0792116
6   nm1300111
7   nm0196375
8   nm1464837
9   nm2868019
(2382, 1)
Wall time: 106 ms
```

In [20]:

```
%%time
def grader_9(q9):
    q9_results  = pd.read_sql_query(q9,conn)
    print(q9_results.head(10))
    print(q9_results.shape)
    assert (q9_results.shape == (25698, 1))

query9 = """ WITH S1 AS(
SELECT DISTINCT(PID) PID FROM M_CAST WHERE MID IN (SELECT MID FROM M_CAST WHERE TRIM(PID)
IN(SELECT PID FROM PERSON P WHERE TRIM(NAME)='Shah Rukh Khan')) AND PID NOT IN(SELECT PID
FROM PERSON P WHERE TRIM(NAME)='Shah Rukh Khan')),
S2 AS(
SELECT DISTINCT(MID) MID FROM M_CAST MC INNER JOIN S1 ON MC.PID=S1.PID),
T1 AS(
SELECT DISTINCT(PID) PID FROM M_CAST MC INNER JOIN S2 ON MC.MID=S2.MID)
SELECT NAME FROM PERSON WHERE TRIM(PID) IN(SELECT TRIM(PID) FROM T1 WHERE PID NOT IN (SEL
ECT PID FROM S1))  """
grader_9(query9)
```

```
                   Name
0           Freida Pinto
1            Rohan Chand
2           Damian Young
3         Waris Ahluwalia
4   Caroline Christl Long
5           Rajeev Pahuja
6       Michelle Santiago
7         Alicia Vikander
8            Dominic West
9          Walton Goggins
(25698, 1)
```

```
Wall time: 471 ms
```

In [ ]:

```
```