SQL Assignment on IMDB data

1.0 Connecting to database using sqlite3 ¶

In [1]:

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

2.0 Cleaning Movie table

In [0]:

```
cursor = conn.cursor()
#https://www.w3schools.com/sql/func_sqlserver_cast.asp
cursor.execute("SELECT MID,TRIM(title),CAST(SUBSTR(year,-4,4) AS int),CAST(rating AS fl
oat),CAST(TRIM(num_votes) AS int) FROM MOVIE;")
formatted_result = [i for i in cursor.fetchall()]
print(formatted_result[0])
```

```
('tt2388771', 'Mowgli', 2018, 6.6, 21967)
```

In [0]:

```
df= pd.DataFrame(formatted_result, columns=['MID', 'title', 'year', 'rating', 'num_vote
s'])
df.head()
```

Out[0]:

	MID	title	year	rating	num_votes
0	tt2388771	Mowgli	2018	6.6	21967
1	tt5164214	Ocean's Eight	2018	6.2	110861
2	tt1365519	Tomb Raider	2018	6.4	142585
3	tt0848228	The Avengers	2012	8.1	1137529
4	tt8239946	Tumbbad	2018	8.5	7483

In [0]:

```
df.year.unique()
```

Out[0]:

```
array([2018, 2012, 2016, 2017, 2008, 2009, 1977, 2013, 2015, 2007, 2002,
       1951, 2014, 2004, 1997, 1983, 1994, 2011, 1996, 2001, 2010, 2006,
       1971, 1958, 1984, 1987, 2005, 2003, 1995, 1998, 1975, 1939, 2000,
       1988, 1991, 1993, 1981, 1989, 1976, 1959, 1970, 1979, 1964, 1999,
       1990, 1992, 1957, 1980, 1966, 1967, 1973, 1968, 1969, 1982, 1978,
       1965, 1972, 1956, 1974, 1960, 1985, 1986, 1949, 1955, 1961, 1962,
       1954, 1941, 1963, 1931, 1953, 1948, 1952, 1947, 1936, 1946, 1943,
       1950], dtype=int64)
```

In [0]:

```
df.title.unique()
```

Out[0]:

```
array(['Mowgli', "Ocean's Eight", 'Tomb Raider', ..., 'Allah-Rakha',
       'Come December', 'Kala Jigar'], dtype=object)
```

In [0]:

```
df.rating.unique()
```

Out[0]:

```
array([6.6, 6.2, 6.4, 8.1, 8.5, 5.5, 7.8, 9. , 5.7, 5.3, 8. , 5.8, 2.4,
       7.7, 7.5, 7.9, 5.6, 3.6, 8.2, 7.1, 7.2, 5.9, 6.7, 7. , 2.6, 7.3,
       7.4, 8.4, 6.8, 7.6, 6. , 2.9, 8.8, 8.3, 2.1, 3.3, 4.4, 4.6, 5.2,
       6.9, 3.7, 6.5, 3.1, 4.5, 5.4, 4.2, 6.1, 6.3, 4. , 4.1, 2.3, 4.8,
       5.1, 1.3, 8.6, 4.3, 3.4, 3.2, 3.9, 4.9, 5. , 3. , 4.7, 8.9, 3.8,
       2.2, 1.9, 1.7, 1.8, 2.8, 2.7, 3.5, 2.5, 2. , 8.7, 9.3, 9.2, 1.6,
       9.4, 9.1, 9.6, 9.5])
```

In [0]:

```
df.num_votes.unique()
```

Out[0]:

```
array([ 21967, 110861, 142585, ...,    664,   1030,    509], dtype=int64)
```

In [0]:

```
df.to_sql('MOVIES', conn, if_exists="replace", index = False)
```

In [0]:

```
#cross-checking to ensure year columns does not have roman numbers
df2= pd.read_sql_query(" SELECT * FROM MOVIES;",conn)
df2.head()
```

Out[0]:

	MID	title	year	rating	num_votes
0	tt2388771	Mowgli	2018	6.6	21967
1	tt5164214	Ocean's Eight	2018	6.2	110861
2	tt1365519	Tomb Raider	2018	6.4	142585
3	tt0848228	The Avengers	2012	8.1	1137529
4	tt8239946	Tumbbad	2018	8.5	7483

In [0]:

```
df2.year.unique()
```

Out[0]:

```
array([2018, 2012, 2016, 2017, 2008, 2009, 1977, 2013, 2015, 2007, 2002,
       1951, 2014, 2004, 1997, 1983, 1994, 2011, 1996, 2001, 2010, 2006,
       1971, 1958, 1984, 1987, 2005, 2003, 1995, 1998, 1975, 1939, 2000,
       1988, 1991, 1993, 1981, 1989, 1976, 1959, 1970, 1979, 1964, 1999,
       1990, 1992, 1957, 1980, 1966, 1967, 1973, 1968, 1969, 1982, 1978,
       1965, 1972, 1956, 1974, 1960, 1985, 1986, 1949, 1955, 1961, 1962,
       1954, 1941, 1963, 1931, 1953, 1948, 1952, 1947, 1936, 1946, 1943,
       1950], dtype=int64)
```

In [0]:

```
df2.MID.unique()
```

Out[0]:

```
array(['tt2388771', 'tt5164214', 'tt1365519', ..., 'tt0852989',
       'tt0375882', 'tt0375890'], dtype=object)
```

3.0 Queries

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 [0]:

```
query= "SELECT p.Name, m.title, m.year FROM Person p JOIN M_Director md on p.PID=md.PID
JOIN Movies m on m.MID=md.MID JOIN M_Genre mg on m.MID=mg.MID JOIN Genre g on mg.GID=g.
GID WHERE trim(g.Name) LIKE 'Comedy' AND m.year%4==0;"
df1= pd.read_sql_query(query,conn)
df1
```

Out[0]:

	Name	title	year
0	Milap Zaveri	Mastizaade	2016
1	Milap Zaveri	Mastizaade	2016
2	Umesh Ghadge	Kyaa Kool Hain Hum 3	2016
3	Umesh Ghadge	Kyaa Kool Hain Hum 3	2016
4	Sameer Sharma	Luv Shuv Tey Chicken Khurana	2012
5	Sanjeev Sharma	Saat Uchakkey	2016
6	Sanjeev Sharma	Saat Uchakkey	2016
7	Nitin Kakkar	Filmistaan	2012
8	Krishnadev Yagnik	Days of Tafree	2016
9	Abhishek Sharma	Tere Bin Laden: Dead Or Alive	2016
10	Abhishek Sharma	Tere Bin Laden: Dead Or Alive	2016
11	Mahesh Bhatt	Papa Kahte Hain	1996
12	Mahesh Bhatt	Papa Kahte Hain	1996
13	Sachin Yardi	Kyaa Super Kool Hain Hum	2012
14	Sachin	Navra Mazha Navsacha	2004
15	Sachin	Navra Mazha Navsacha	2004
16	Kawal Sharma	Maalamaal	1988
17	Aditya Datt	Will You Marry Me	2012
18	Rajnish Thakur	Mere Dost Picture Abhi Baaki Hai	2012
19	Rajnish Thakur	Mere Dost Picture Abhi Baaki Hai	2012
20	Karan Razdan	Mr Bhatti on Chutti	2012
21	Karan Razdan	Mr Bhatti on Chutti	2012
22	Jagdish Rajpurohit	Bumboo	2012
23	Jagdish Rajpurohit	Bumboo	2012
24	Suhas Kadav	Motu Patlu: King of Kings	2016
25	Vickrant Mahajan	Challo Driver	2012
26	Vickrant Mahajan	Challo Driver	2012
27	Bhagyaraj	Mr. Bechara	1996
28	Bhagyaraj	Mr. Bechara	1996
29	Rakesh Mehta	Life Ki Toh Lag Gayi	2012
30	Rakesh Mehta	Life Ki Toh Lag Gayi	2012
31	Anand Balraj	Daal Mein Kuch Kaala Hai	2012
32	Anand Balraj	Daal Mein Kuch Kaala Hai	2012

	Name	title	year
33	Govind Menon	Kis Kis Ki Kismat	2004
34	Govind Menon	Kis Kis Ki Kismat	2004
35	Pankaj Parashar	Ab Ayega Mazaa	1984
36	Jabbar Patel	Ek Hota Vidushak	1992
37	Jabbar Patel	Ek Hota Vidushak	1992
38	Srinivas Bhashyam	Paisa Vasool	2004
39	Raj Kaushal	Shaadi Ka Laddoo	2004
40	Raj Kaushal	Shaadi Ka Laddoo	2004
41	Siddharth Anand Kumar	Let's Enjoy	2004
42	Siddharth Anand Kumar	Let's Enjoy	2004

Eliminating duplicates from the above table

In [0]:

```
df1.drop_duplicates(subset="title")
```

Out[0]:

	Name	title	year
0	Milap Zaveri	Mastizaade	2016
2	Umesh Ghadge	Kyaa Kool Hain Hum 3	2016
4	Sameer Sharma	Luv Shuv Tey Chicken Khurana	2012
5	Sanjeev Sharma	Saat Uchakkey	2016
7	Nitin Kakkar	Filmistaan	2012
8	Krishnadev Yagnik	Days of Tafree	2016
9	Abhishek Sharma	Tere Bin Laden: Dead Or Alive	2016
11	Mahesh Bhatt	Papa Kahte Hain	1996
13	Sachin Yardi	Kyaa Super Kool Hain Hum	2012
14	Sachin	Navra Mazha Navsacha	2004
16	Kawal Sharma	Maalamaal	1988
17	Aditya Datt	Will You Marry Me	2012
18	Rajnish Thakur	Mere Dost Picture Abhi Baaki Hai	2012
20	Karan Razdan	Mr Bhatti on Chutti	2012
22	Jagdish Rajpurohit	Bumboo	2012
24	Suhas Kadav	Motu Patlu: King of Kings	2016
25	Vickrant Mahajan	Challo Driver	2012
27	Bhagyaraj	Mr. Bechara	1996
29	Rakesh Mehta	Life Ki Toh Lag Gayi	2012
31	Anand Balraj	Daal Mein Kuch Kaala Hai	2012
33	Govind Menon	Kis Kis Ki Kismat	2004
35	Pankaj Parashar	Ab Ayega Mazaa	1984
36	Jabbar Patel	Ek Hota Vidushak	1992
38	Srinivas Bhashyam	Paisa Vasool	2004
39	Raj Kaushal	Shaadi Ka Laddoo	2004
41	Siddharth Anand Kumar	Let's Enjoy	2004

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

In [0]:

```
query2= "SELECT p.Name FROM Person p JOIN M_Cast mc ON trim(p.PID)=trim(mc.PID) JOIN Mo
vies m ON trim(mc.MID)=m.MID WHERE trim(m.title) LIKE 'Anand';"
df2= pd.read_sql_query(query2,conn)
df2
```

Out[0]:

	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 [2]:

```
query3= "SELECT p.Name from Person_ p WHERE p.PID in (SELECT trim(mc1.PID) from M_Cast
 mc1 where trim(mc1.MID) in (SELECT m1.MID from Movies m1 where m1.year<1970))\
AND p.PID in (SELECT mc2.PID from M_Cast mc2 where mc2.MID in (SELECT m2.MID from Movie
s m2 where m2.year>1990));"
df3= pd.read_sql_query(query3,conn)
df3
```

Out[2]:

	Name
0	Rishi Kapoor
1	Amitabh Bachchan
2	Asrani
3	Zohra Sehgal
4	Michael Rennie
5	Patricia Neal
6	Hugh Marlowe
7	Sam Jaffe
8	Billy Gray
9	Frances Bavier
10	Lock Martin
11	Patrick Aherne
12	Walter Bacon
13	Rama Bai
14	Oscar Blank
15	Marshall Bradford
16	Chet Brandenburg
17	John Brown
18	John Burton
19	Michael Capanna
20	Wheaton Chambers
21	Spencer Chan
22	Jean Charney
23	Beulah Christian
24	John Close
25	Louise Colombet
26	James Conaty
27	Frank Conroy
28	Eric Corrie
29	John Costello
...	...
1929	Ranjana
1930	Michael

	Name
1931	Sudesh
1932	Bahujbal Singh
1933	Goldstein
1934	Young Nurjehan
1935	Baby Nirmal
1936	Nand Kishore
1937	Shivaji Rathore
1938	Suraj Prakash
1939	Chandan Mukherji
1940	Tillu
1941	Sudha Rani
1942	Bhattacharya
1943	Rafia
1944	Nazira
1945	Nafis Khalili
1946	Shanta Kumari
1947	Baby Indira
1948	Subhashini
1949	Dalip Kumar
1950	Kumar Sahu
1951	Karan Singh
1952	Jayashree
1953	Prof. Hudlikar
1954	Salvi
1955	Dewan Sharar
1956	Master Vinayak
1957	Sabitha Perara
1958	Iqbal

1959 rows × 1 columns

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 [0]:

```
query4= "SELECT p.Name, COUNT(p.PID) dir_count FROM Person p JOIN M_Director md ON trim
(p.PID)=trim(md.PID) JOIN Movies m ON trim(md.MID)=m.MID GROUP BY p.PID HAVING dir_coun
t>=10 ORDER BY dir_count DESC;"
df4= pd.read_sql_query(query4,conn)
df4
```

Out[0]:

	Name	dir_count
0	David Dhawan	78
1	Mahesh Bhatt	70
2	Ram Gopal Varma	60
3	Vikram Bhatt	58
4	Hrishikesh Mukherjee	54
5	Yash Chopra	42
6	Basu Chatterjee	38
7	Shakti Samanta	38
8	Subhash Ghai	36
9	Shyam Benegal	34
10	Abbas Alibhai Burmawalla	34
11	Manmohan Desai	32
12	Gulzar	32
13	Raj N. Sippy	32
14	Raj Kanwar	30
15	Mahesh Manjrekar	30
16	Priyadarshan	30
17	Indra Kumar	28
18	Raj Khosla	28
19	Rahul Rawail	28
20	Rajkumar Santoshi	28
21	Rakesh Roshan	26
22	Dev Anand	26
23	Vijay Anand	26
24	Harry Baweja	26
25	Anurag Kashyap	26
26	Ananth Narayan Mahadevan	26
27	K. Raghavendra Rao	26
28	Anees Bazmee	24
29	Guddu Dhanoa	24
...	...	...
126	Aziz Mirza	10
127	Saeed Akhtar Mirza	10

	Name	dir_count
128	Mohan Babu	10
129	Shankar Mukherjee	10
130	Subodh Mukherji	10
131	K. Muralimohana Rao	10
132	Shashilal K. Nair	10
133	Amol Palekar	10
134	Pankaj Parashar	10
135	Bharat Rangachary	10
136	Ramanand Sagar	10
137	Mohan Segal	10
138	Bhappi Sonie	10
139	Lekh Tandon	10
140	Rakeysh Omprakash Mehra	10
141	Remo D'Souza	10
142	Sanjay Gadhvi	10
143	Prawaal Raman	10
144	Sujoy Ghosh	10
145	Pradeep Sarkar	10
146	Sajid Khan	10
147	Krishna D.K.	10
148	Subhash Kapoor	10
149	S.S. Rajamouli	10
150	Nishikant Kamat	10
151	Siddharth Anand	10
152	Dibakar Banerjee	10
153	Shoojit Sircar	10
154	R. Balki	10
155	Neeraj Pandey	10

156 rows × 2 columns

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

In [0]:

```
query5_a="SELECT m.year,  COUNT(m.MID) F_movie_count FROM Movies m WHERE m.MID NOT IN(S
ELECT trim(mc.MID) FROM M_Cast mc LEFT JOIN Person p ON trim(mc.PID)=trim(p.PID) WHERE
 Gender!='Female') GROUP by m.year ORDER BY m.year;"
df5_a= pd.read_sql_query(query5_a,conn)
df5_a
```

Out[0]:

	year	F_movie_count
0	1939	1
1	1999	1
2	2000	1
3	2009	1
4	2012	1
5	2018	2

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 [3]:

```
query5_b="SELECT m.year, COUNT(m.MID) as total_Movie_count, f.F_movie_count, (f.F_movie
_count*100.00/COUNT(m.MID)) as Percentage FROM MOVIES m JOIN (SELECT m.year, COUNT(m.MI
D) F_movie_count FROM Movies m JOIN (SELECT DISTINCT MID FROM M_Cast WHERE trim(MID) NO
T IN(SELECT trim(mc.MID) FROM M_Cast mc JOIN Person p ON trim(mc.PID)=trim(p.PID) WHERE
Gender!='Female')) a ON trim(a.MID)=m.MID GROUP by m.year) f ON m.year=f.year GROUP by
 m.year ORDER BY m.year;"
df5_b= pd.read_sql_query(query5_b,conn)
df5_b
```

Out[3]:

	year	total_Movie_count	F_movie_count	Percentage
0	1939	2	1	50.000000
1	1999	66	1	1.515152
2	2000	64	1	1.562500
3	2009	110	1	0.909091
4	2012	111	1	0.900901
5	2018	104	2	1.923077

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 [13]:

```
query_6= "SELECT title,MAX(num) AS cast_size FROM (SELECT title ,COUNT(PID) as num FROM
MOVIES m JOIN M_Cast mc ON m.MID=trim(mc.MID) GROUP BY title);"
df6= pd.read_sql_query(query_6,conn)
df6
```

Out[13]:

	title	cast_size
0	Ocean's Eight	238

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 [0]:

```
# Ref: https://stackoverflow.com/questions/51609285/sql-query-for-find-the-decade-with-the-largest-number-of-records

query_7= "SELECT y.year as decade_start, y.year + 9 as decade_end, count(MID) as num_movies FROM (SELECT distinct year from MOVIES) y JOIN Movies m ON m.year >= y.year and m.year < y.year + 10 group by y.year order by num_movies desc;"
df7= pd.read_sql_query(query_7,conn)
df7
```

Out[0]:

	decade_start	decade_end	num_movies
0	2008	2017	1205
1	2009	2018	1202
2	2007	2016	1188
3	2005	2014	1170
4	2006	2015	1160
5	2004	2013	1147
6	2003	2012	1114
7	2010	2019	1092
8	2002	2011	1090
9	2001	2010	1047
10	2000	2009	986
11	2011	2020	967
12	1999	2008	942
13	1998	2007	890
14	2012	2021	851
15	1997	2006	836
16	1996	2005	795
17	2013	2022	740
18	1995	2004	722
19	1994	2003	679
20	1993	2002	639
21	1992	2001	610
22	2014	2023	604
23	1991	2000	578
24	1990	1999	556
25	1989	1998	537
26	1988	1997	519
27	1987	1996	496
28	2015	2024	478
29	1986	1995	469
...	...	...	...
48	1968	1977	245
49	1967	1976	236

	decade_start	decade_end	num_movies
50	1966	1975	232
51	2017	2026	230
52	1965	1974	222
53	1964	1973	211
54	1963	1972	192
55	1962	1971	175
56	1961	1970	158
57	1960	1969	148
58	1959	1968	136
59	1958	1967	124
60	1957	1966	118
61	1956	1965	106
62	2018	2027	104
63	1955	1964	101
64	1954	1963	92
65	1953	1962	90
66	1952	1961	84
67	1951	1960	83
68	1950	1959	71
69	1949	1958	68
70	1948	1957	62
71	1947	1956	51
72	1946	1955	47
73	1943	1952	25
74	1941	1950	14
75	1939	1948	11
76	1936	1945	7
77	1931	1940	6

78 rows × 3 columns

- Decade 2008-2017 has the largest number of films

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 [0]:

```
query_8= "SELECT DISTINCT(PID), Name FROM Person \
WHERE trim(PID) NOT IN(SELECT DISTINCT(PID) FROM M_Cast C1 JOIN Movies as M1 \
WHERE EXISTS( SELECT trim(MID) FROM M_Cast C2 JOIN Movies as M2 \
WHERE TRIM(C1.PID) = TRIM(C2.PID) and (M2.year - 3) > M1.year AND \
NOT EXISTS( SELECT trim(MID) from M_Cast as C3 JOIN Movies as M3 WHERE TRIM(C1.PID) = T
RIM(C3.PID) and M1.year < M3.year and M3.year < M2.year )));"

df8= pd.read_sql_query(query_8,conn)
df8
```

Out[0]:

	PID	Name
0	nm0000288	Christian Bale
1	nm0000949	Cate Blanchett
2	nm1212722	Benedict Cumberbatch
3	nm0365140	Naomie Harris
4	nm0785227	Andy Serkis
...	...	...
37561	nm2182643	Kamika Verma
37562	nm1029114	Dhorairaj Bhagavan
37563	nm3769883	Nasir Shaikh
37564	nm1470989	Kannan
37565	nm0298158	Adrian Fulle

37566 rows × 2 columns

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

In [2]:

```
query_9= " SELECT t2.actor,t2.number FROM (SELECT MAX(t1.count) Number, t1.director, t
1.actor FROM (SELECT COUNT(trim(p.PID)) count, trim(p.Name) actor, trim(md.PID) directo
r FROM Person_ p JOIN M_Cast mc ON trim(p.PID)=trim(mc.PID) JOIN M_Director md ON trim
(mc.MID)=trim(md.MID) GROUP BY md.PID, p.PID) t1 GROUP BY t1.actor) t2 WHERE t2.directo
r=(SELECT PID FROM Person_ WHERE trim(Name)='Yash Chopra') order by Number desc;"
df9= pd.read_sql_query(query_9,conn)
df9
```

Out[2]:

	actor	Number
0	Jagdish Raj	11
1	Manmohan Krishna	10
2	Iftekhar	9
3	Shashi Kapoor	7
4	Rakhee Gulzar	5
5	Waheeda Rehman	5
6	Achala Sachdev	4
7	Neetu Singh	4
8	Ravikant	4
9	Leela Chitnis	3
10	Mohan Sherry	3
11	Parikshat Sahni	3
12	Saul George	3
13	Sudha Chopra	3
14	Surendra Rahi	3
15	Ashok Verma	2
16	C.L. Shah	2
17	Chandu Allahabadi	2
18	Gajanan Jagirdar	2
19	Manohar Singh	2
20	Nanda	2
21	Nazir	2
22	Nissar	2
23	Padma Khanna	2
24	Poonam Dhillon	2
25	Raj Hans	2
26	Rehman	2
27	Shyam Arora	2
28	Surendra Nath	2
29	Yash Chopra	2
...	...	...
189	Shaun Lucas	1
190	Shivani Vazir	1

	actor	Number
191	Shivaya Singh	1
192	Shruti Ultaf	1
193	Silvia Crastan	1
194	Stephanie Aslan	1
195	Steve Box	1
196	Steven Baker	1
197	Stuart Bailey	1
198	Sujata	1
199	Sumati Gupte	1
200	Sundeep	1
201	Surekha	1
202	Susan Fordham	1
203	Susana Rodrigues	1
204	Sushil Kumar	1
205	Syed Firdaus	1
206	Taj Gill	1
207	Tina Tancakova	1
208	Tom Baines Maher	1
209	Tom Swacha	1
210	Uma Khosla	1
211	Uttam Sodi	1
212	Varun Thajur	1
213	Varun Thakur	1
214	Vic Waghorn	1
215	Vicky K. Foster	1
216	Vinita Sharma	1
217	Vinod Negi	1
218	Yasin Khan	1

219 rows × 2 columns

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 [0]:

```
query_10 = "SELECT trim(p.Name) as Actor FROM Person p WHERE trim(p.PID) IN (SELECT tri
m(PID) FROM M_Cast  WHERE trim(MID) IN\
(SELECT trim(MID) FROM M_Cast  WHERE trim(PID) IN (SELECT trim(PID) FROM M_Cast WHERE t
rim(MID) IN\
(SELECT trim(MID) FROM M_Cast  WHERE trim(PID) IN (SELECT trim(p1.PID) FROM Person p1 W
HERE trim(p1.Name)='Shah Rukh Khan')))));"

df10= pd.read_sql_query(query_10,conn)
df10
```

Out[0]:

	Actor
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
10	Daniel Wu
11	Kristin Scott Thomas
12	Derek Jacobi
13	Alexandre Willaume
14	Tamer Burjaq
15	Adrian Collins
16	Keenan Arrison
17	Andrian Mazive
18	Milton Schorr
19	Hannah John-Kamen
20	Peter Waison
21	Samuel Mak
22	Sky Yang
23	Civic Chung
24	Josef Altin
25	Billy Postlethwaite
26	Roger Jean Nsengiyumva
27	Jaime Winstone
28	Michael Obiora
29	Shekhar Varma
...	...
28413	Ashoke Pandit
28414	Deepak Kumar Bandhu

	Actor
28415	Shivajee Chandrabhushan
28416	Anand Balraj
28417	Muazzam Beg
28418	Asad Shan
28419	A.M.R. Ramesh
28420	Joy Mukherjee
28421	Punarvasu Naik
28422	Ashok Kheny
28423	Nagarjuna Akkineni
28424	Sartaj Singh Pannu
28425	Ranjit Kapoor
28426	Sanjay Amar
28427	Anjan Dutt
28428	Manmohan Krishna
28429	V. Ravichandran
28430	Pawan Gill
28431	Bharat Dabholkar
28432	Jyothika
28433	Yograj Bhat
28434	Sadhu Kokila
28435	Parvin Dabas
28436	Vineet Khetrapal
28437	Vijaya Mehta
28438	Rahat Kazmi
28439	Srinivas Sunderrajan
28440	Abbas
28441	Gulshan Kumar
28442	Sushma Shiromani

28443 rows × 1 columns