In [1]:

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

In [2]:

```
cursor = conn.cursor()
cursor.execute('UPDATE Movie SET year = trim(substr(year,instr(year," ")));')
conn.commit()
```

In [3]:

```
cursor.execute('delete from MOVIE where MID in(select MID from MOVIE group by MID having count(*)
>1);')
conn.commit()
cursor.execute('delete from PERSON where PID in(select PID from PERSON group by PID having count(*
) >1);')
conn.commit()
cursor.execute('delete from GENRE where GID in(select GID from GENRE group by GID having count(*)
>1);')
conn.commit()
```

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 [31]:

```
query1 = pd.read_sql_query('''SELECT m.year,m.title,p.Name FROM movie m JOIN M_Director md
ON md.mid = m.mid JOIN Person p ON p.pid = md.pid where (m.year % 4 = 0) AND ((m.year % 100 != 0)
OR
(m.year % 400 = 0)) and m.title IN(SELECT m.title FROM GENRE g JOIN M_Genre mg ON mg.gid = g.gid J
OIN
Movie m on m.mid=mg.mid where g.name LIKE '%Comedy%');''', conn)
query1
```

Out[31]:

	year	title	Name
0	2016	Dishoom	Rohit Dhawan
1	2016	Sanam Teri Kasam	Radhika Rao
2	2000	Hera Pheri	Priyadarshan
3	2016	Hotel Salvation	Shubhashish Bhutiani
4	2008	The Other End of the Line	James Dodson
5	1996	Tere Mere Sapne	Joy Augustine
6	2016	Tu Hai Mera Sunday	Milind Dhaimade
7	2012	Luv Shuv Tey Chicken Khurana	Sameer Sharma
8	2004	Meri Biwi Ka Jawab Nahin	Pankaj Parashar
9	1996	Kadhal Desam	Kathir
10	1996	Raja Hindustani	Dharmesh Darshan
11	2012	Filmistaan	Nitin Kakkar
12	2008	Mere Baap Pehle Aap	Priyadarshan
13	2012	Son of Sardaar	Ashwani Dhir
14	2004	Hulchul	Priyadarshan
15	1964	Rajkumar	K. Shankar
16	2008	Bhoothnath	Vivek Sharma

	year	title	Name
17	2012	Aiyyaa	Sachin Kundalkar
18	2000	Hadh Kar Di Aapne	Manoj Agrawal
19	1956	Jagte Raho	Amit Mitra
20	2000	Mela	Dharmesh Darshan
21	2004	Tauba Tauba	T.L.V. Prasad
22	2016	Days of Tafree	Krishnadev Yagnik
23	2000	Raju Chacha	Anil Devgan
24	1956	Chori Chori	Anant Thakur
25	2016	Fuddu	Sunil Subramani
26	2012	Kyaa Super Kool Hain Hum	Sachin Yardi
27	2016	Thikka	Sunil K. Reddy
28	2012	Kamaal Dhamaal Malamaal	Priyadarshan
29	1980	Swayamvar	P. Sambasiva Rao
...	...	...	...
50	2012	Tere Naal Love Ho Gaya	Mandeep Kumar
51	2012	Will You Marry Me	Aditya Datt
52	2012	Chaalis Chauraasi	Hriday Shetty
53	2000	Hum To Mohabbat Karega	Kundan Shah
54	2016	Direct Ishq	Rajiv S. Ruia
55	2004	Ek Se Badhkar Ek	Kundan Shah
56	2008	Oh, My God!!	Sourabh Shrivastava
57	1972	Raaste Kaa Patthar	Mukul Dutt
58	2008	Hari Puttar: A Comedy of Terrors	Rajesh Bajaj
59	2016	Motu Patlu: King of Kings	Suhas Kadav
60	2008	Hulla	Jaideep Varma
61	1956	Jagriti	Satyen Bose
62	2000	Beti No. 1	Rama Rao Tatineni
63	1972	Garam Masala	Aspi Irani
64	2016	Hello Mumbai: Salaam Mumbai	Ghorban Mohammadpour
65	2008	Hastey Hastey Follow Your Heart	Ramanjit Juneja
66	1984	Ab Ayega Mazaa	Pankaj Parashar
67	1980	Dadar Kirti	Tarun Majumdar
68	2008	Bach ke Zara	Salim Raza
69	2008	Ugly Aur Pagli	Sachin Kamlakar Khot
70	1988	Ghar Ghar Ki Kahani	Kalpataru
71	2004	Paisa Vasool	Srinivas Bhashyam
72	1968	Do Dooni Char	Debu Sen
73	2000	Azaad	Tirupati Swamy
74	2012	Khokababu	Shankaraiya
75	2008	Meerabai Not Out	Chandrakant Kulkarni
76	2008	Sathyam	Amma Rajasekhar
77	2008	Tandoori Love	Oliver Paulus
78	2012	Le Halua Le	Raja Chanda
79	1996	Raja Aur Rangeeli	K.S. Prakash Rao

80 rows × 3 columns

80 rows × 3 columns

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

In [34]:

```
query2 = pd.read_sql_query('''SELECT p.Name from Person p WHERE p.PID IN
(SELECT LTRIM(mc.PID) from M_Cast mc JOIN Movie m on mc.MID=m.MID WHERE m.title ='Anand')''', conn
)
query2
```

Out[34]:

	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

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

In [59]:

```
query3 = pd.read_sql_query('''SELECT p.Name from Person p WHERE p.PID IN
(SELECT LTRIM(mc.PID) from M_Cast mc JOIN Movie m on mc.MID=m.MID WHERE (m.year < 1970))
INTERSECT SELECT p.Name from Person p WHERE p.PID IN (SELECT LTRIM(mc.PID)
from M_Cast mc JOIN Movie m on mc.MID=m.MID WHERE (m.year > 1990))''', conn)
query3
```

Out[59]:

	Name
0	A.K. Hangal
1	Aachi Manorama
2	Abbas
3	Abdul
4	Abhi Bhattacharya
5	Achala Sachdev
6	Adil
7	Ajay

	Name
8	Ajit
9	Akashdeep
10	Akbar Bakshi
11	Alka
12	Allu Ramalingaiah
13	Altaf
14	Amar
15	Amarnath
16	Ameer
17	Amitabh Bachchan
18	Amjad Khan
19	Amol Sen
20	Amrit
21	Anand
22	Anand Kumar
23	Anand Tiwari
24	Anil
25	Anil Kumar
26	Anil Nagrath
27	Anjali Kadam
28	Anju Mahendru
29	Anoop Kumar
...	...
372	Tanuja
373	Tej Sapru
374	Thapa
375	Tulsi
376	Uma
377	Umesh Sharma
378	Unni Mary
379	Urmila Bhatt
380	Usha Kiran
381	Utpal Dutt
382	Veena
383	Veera
384	Vijay
385	Vijayalalitha
386	Vijayalaxmi
387	Viju Khote
388	Vikram Makandar
389	Vineet Kumar
390	Vinod Sharma
391	Vishnu
392	Vishwa Mehra
393	Waheeda Rehman

	Name
394	Wasi Khan
395	Yash Kumar
396	Yasmin
397	Yunus Parvez
398	Yusuf
399	Zia
400	Zohra Sehgal
401	Zul Vellani

402 rows × 1 columns

1. 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 [34]:

```
query4 = pd.read_sql_query('''SELECT count(m.title) as 'No of Movies',p.Name FROM movie m
JOIN M_Director md ON md.mid = m.mid JOIN Person p ON p.pid = md.pid group by p.name having
count(m.title) >= 10 order by count(m.title) desc;''', conn)
query4
```

Out[34]:

	No of Movies	Name
0	30	Priyadarshan
1	17	Rama Rao Tatineni
2	10	K. Bapaiah
3	10	K. Muralimohana Rao
4	10	Pankaj Parashar

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

In [79]:

```
#https://stackoverflow.com/questions/17975229/using-sql-count-in-a-case-statement
query5a = pd.read_sql_query('''SELECT count(case when p.gender='Female' then 1 end) as 'No of Movi
es',m.year
from Person p JOIN M_Cast mc on LTRIM(mc.pid)=p.pid JOIN Movie m on LTRIM(mc.MID)=m.MID group by m
.year;''', conn)
query5a
```

Out[79]:

	No of Movies	year
0	3	1931
1	19	1936
2	17	1939
3	7	1941
4	3	1943
5	6	1946
6	11	1947
7	10	1948
8	15	1949
9	13	1950

	No of Movies	year
10	37	1951
11	27	1952
12	69	1953
13	22	1954
14	58	1955
15	29	1956
16	78	1957
17	62	1958
18	34	1959
19	96	1960
20	51	1961
21	85	1962
22	64	1963
23	77	1964
24	93	1965
25	119	1966
26	118	1967
27	154	1968
28	127	1969
29	167	1970
...	...	...
48	273	1989
49	240	1990
50	209	1991
51	313	1992
52	302	1993
53	332	1994
54	278	1995
55	283	1996
56	294	1997
57	284	1998
58	453	1999
59	425	2000
60	499	2001
61	646	2002
62	662	2003
63	527	2004
64	855	2005
65	632	2006
66	762	2007
67	893	2008
68	817	2009
69	965	2010
70	823	2011
71	796	2012

	No. of Movies	year
72	814	2013
73	764	2014
74	817	2015
75	960	2016
76	965	2017
77	838	2018

78 rows × 2 columns

5b. 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 [85]:

```
#https://stackoverflow.com/questions/8257106/divide-the-value-of-a-column-by-another-column
query5b = pd.read_sql_query('''SELECT count(case when p.gender='Female' then 1 end) as
'FemaleMovies',
count(*) as 'TotalMovies',m.year,(CAST(count(case when p.gender='Female' then 1 end)as
FLOAT)/count(*))*100 as 'Result'
from Person p JOIN M_Cast mc on LTRIM(mc.pid)=p.pid JOIN Movie m on LTRIM(mc.MID)=m.MID group by m
.year;''', conn)
query5b
```

Out[85]:

	FemaleMovies	TotalMovies	year	Result
0	3	8	1931	37.500000
1	19	46	1936	41.304348
2	17	44	1939	38.636364
3	7	54	1941	12.962963
4	3	13	1943	23.076923
5	6	20	1946	30.000000
6	11	22	1947	50.000000
7	10	38	1948	26.315789
8	15	41	1949	36.585366
9	13	43	1950	30.232558
10	37	174	1951	21.264368
11	27	75	1952	36.000000
12	69	217	1953	31.797235
13	22	75	1954	29.333333
14	58	148	1955	39.189189
15	29	117	1956	24.786325
16	78	250	1957	31.200000
17	62	203	1958	30.541872
18	34	102	1959	33.333333
19	96	274	1960	35.036496
20	51	155	1961	32.903226
21	85	251	1962	33.864542
22	64	185	1963	34.594595
23	77	236	1964	32.627119
24	93	282	1965	32.978723

	FemaleMovies	TotalMovies	year	Result
25	119	364	1966	32.692308
26	118	392	1967	30.102041
27	154	419	1968	36.754177
28	127	442	1969	28.733032
29	167	540	1970	30.925926
...	...	...	...	...
48	273	962	1989	28.378378
49	240	901	1990	26.637070
50	209	776	1991	26.932990
51	313	1153	1992	27.146574
52	302	1193	1993	25.314334
53	332	1272	1994	26.100629
54	278	1118	1995	24.865832
55	283	1122	1996	25.222816
56	294	1209	1997	24.317618
57	284	1217	1998	23.336072
58	453	1547	1999	29.282482
59	425	1382	2000	30.752533
60	499	1623	2001	30.745533
61	646	2020	2002	31.980198
62	662	2113	2003	31.329863
63	527	1868	2004	28.211991
64	855	2817	2005	30.351438
65	632	2067	2006	30.575714
66	762	2396	2007	31.803005
67	893	2820	2008	31.666667
68	817	3054	2009	26.751801
69	965	3441	2010	28.044173
70	823	2733	2011	30.113428
71	796	2733	2012	29.125503
72	814	2921	2013	27.867169
73	764	2685	2014	28.454376
74	817	2865	2015	28.516579
75	960	3256	2016	29.484029
76	965	3530	2017	27.337110
77	838	2875	2018	29.147826

78 rows × 4 columns

1. 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 [12]:

```
query6 = pd.read_sql_query('''SELECT m.title as 'Movie Name',count(DISTINCT p.name) as 'Size of cast'
FROM Movie m JOIN M_Cast mc ON LTRIM(mc.mid) = m.mid JOIN Person p ON p.pid = LTRIM(mc.pid)
group by m.title order by count(DISTINCT p.name) desc;''', conn)
group by m.title order by count(DISTINCT p.Name) desc''', conn)
query6
```

Out[12]:

	Movie Name	Size of cast
0	Ocean's Eight	237
1	Apaharan	232
2	Gold	213
3	My Name Is Khan	210
4	Captain America: Civil War	191
5	Geostorm	170
6	Striker	164
7	2012	154
8	Pixels	143
9	The Avengers	138
10	Yamla Pagla Deewana 2	137
11	Daddy	129
12	Housefull 3	129
13	Fan	127
14	Bajrangi Bhaijaan	124
15	Split Wide Open	124
16	Train Station	122
17	Million Dollar Arm	117
18	Octopussy	116
19	Dhoom:3	114
20	Miss Lovely	113
21	Mubarakan	108
22	Love Aaj Kal	107
23	Jab Tak Hai Jaan	106
24	The Day the Earth Stood Still	105
25	Judwaa 2	103
26	Midnight's Children	103
27	Oye Lucky! Lucky Oye!	103
28	Phantom	101
29	Hey Ram	100
...	...	...
3309	Uski Roti	4
3310	Cheluvi	3
3311	Chhota Bheem and the Throne of Bali	3
3312	Gauru: Journey of Courage	3
3313	Genesis	3
3314	Goopi Gawaiya Bagha Bajaiya	3
3315	Jogi the King	3
3316	Kaun?	3
3317	Kya Dilli Kya Lahore	3
3318	Main Hoon Khiladiyon Ka Khiladi	3

3319	Man on Mission Taqatwar Movie Name	Size of cast
3320	Military Officer	3
3321	Partha	3
3322	Sins	3
3323	Chaar Sahibzaade	2
3324	Dilwale:The Brave Heart	2
3325	Leera the Soulmate	2
3326	Mahakali Ka Insaaf	2
3327	Man on Mission Fauladi	2
3328	Motu Patlu: King of Kings	2
3329	Mumbai Delhi Mumbai	2
3330	Pihu	2
3331	Rui Ka Bojh	2
3332	Uyirile Kalanthathu	2
3333	Yaadein	2
3334	Chaar Sahibzaade 2: Rise of Banda Singh Bahadur	1
3335	Man On Mission Jaanbaaz	1
3336	Raja Aur Rangeeli	1
3337	Return of Hanuman	1
3338	Vaibhav Sethia: Don't	1

3339 rows × 2 columns

1. 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 [58]:

```
query7 = pd.read_sql_query('''SELECT count(title) as 'No of Movies',cast(MIN(year) as varchar) ||
'-' ||
cast(MIN(year)+10 as varchar) as 'Decade'
FROM MOVIE group by substr(year,0,4);''', conn)
query7
```

Out[58]:

	No of Movies	Decade
0	6	1931-1941
1	12	1941-1951
2	71	1950-1960
3	148	1960-1970
4	254	1970-1980
5	350	1980-1990
6	556	1990-2000
7	986	2000-2010
8	1092	2010-2020

1. 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 [35]:

```
query8 = pd.read_sql_query('''Select FirstName from (SELECT p.pid as Actor_ID ,m.year as Year,
p.name as FirstName FROM
movie m
JOIN M_cast mc ON LTRIM(mc.mid) = m.mid
JOIN Person p ON p.pid = LTRIM(mc.pid)
order by p.name,m.year asc) t1

JOIN

(SELECT p.pid as Actor_ID, m.year as Year, p.name as SecondName FROM
movie m
JOIN M_cast mc ON LTRIM(mc.mid) = m.mid
JOIN Person p ON p.pid = LTRIM(mc.pid)
order by p.name,m.year asc) t2 on t1.Actor_ID=t2.Actor_ID
where t2.Year-t1.Year < 3 and t1.FirstName=t2.SecondName group by FirstName;''', conn)
query8
```

Out[35]:

	FirstName
0	'Ganja' Karuppu
1	'Lee' George Quinones
2	'Musafir' Radio Performing
3	'Nandha' Saravanan
4	'Om' Rakesh Chaturvedi
5	'Snub' Pollard
6	A'Ali de Sousa
7	A. Abdul Hameed
8	A. Darpan
9	A. Deiva Sundari
10	A. Gabibi
11	A. Kapoor
12	A. Khan
13	A. Kukereja
14	A. Lakshmi
15	A. Narsimha
16	A. Prabhakar
17	A. Ravi Verma
18	A. Shalomayev
19	A. Sharma
20	A. Vithya
21	A.A. Deepak
22	A.A. Khan
23	A.A. Premawathie
24	A.C. Murali
25	A.C. Sarkar
26	A.D. Singh
27	A.G. Poddar
28	A.H. Shore
29	A.J. Rosen
...	...

	FirstName
30043	Zoya Shah
30044	Zoë Bright
30045	Zoë Castle
30046	Zsolt Nagy
30047	Zsolt Viczei
30048	Zubaida
30049	Zubair Khan
30050	Zubeda
30051	Zubeda Khan
30052	Zubeen Garg
30053	Zubeida
30054	Zuber k Khan
30055	Zuberi
30056	Zubin Jauhari
30057	Zubin Vicky Driver
30058	Zufin
30059	Zuha Sharma
30060	Zul Vellani
30061	Zul Vilani
30062	Zuleikha Robinson
30063	Zulfi Sayed
30064	Zulfikar Ali
30065	Zulica Fito Cia
30066	Zulkhumor Muminova
30067	Zurab Kapianidze
30068	Zuri Echea
30069	Zuzanna Zajac
30070	Àaron Brewster
30071	Éric Berger
30072	Ócsai Krisztián

30073 rows × 1 columns

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

In [134]:

```
query9 = pd.read_sql_query('''WITH
    YASH_CHOPRAS_PID AS
    (
        SELECT
            TRIM(P.PID) AS PID
        FROM
            Person P
        WHERE
            Trim(P.Name) = 'Yash Chopra'
    ),
    NUM_OF_MOV_BY_ACTOR_DIRECTOR AS
    (
        SELECT
            TRIM(MC.PID) ACTOR_PID,
            TRIM(MD.PID) DIRECTOR_PID,
            COUNT(DISTINCT TRIM(MD.MID)) AS NUM_OF_MOV
        FROM
```

```
            M_Cast MC,
            M_Director MD
        WHERE
            TRIM(MC.MID)= TRIM(MD.MID)
        GROUP BY
            ACTOR_PID,
            DIRECTOR_PID
    ),
    NUM_OF_MOVIES_BY_YC AS
    (
        SELECT
            NM.ACTOR_PID,
            NM.DIRECTOR_PID,
            NM.NUM_OF_MOV NUM_OF_MOV_BY_YC
        FROM
            NUM_OF_MOV_BY_ACTOR_DIRECTOR NM,
            YASH_CHOPRAS_PID YCP
        WHERE
            NM.DIRECTOR_PID = YCP.PID
    ),
    MAX_MOV_BY_OTHER_DIRECTORS AS
    (
        SELECT
            ACTOR_PID,
            MAX(NUM_OF_MOV) MAX_NUM_OF_MOV
        FROM
            NUM_OF_MOV_BY_ACTOR_DIRECTOR NM,
            YASH_CHOPRAS_PID YCP
        WHERE
            NM.DIRECTOR_PID <> YCP.PID
        GROUP BY
            ACTOR_PID
    ),
    ACTORS_MOV_COMPARISION AS
    (
    SELECT
        NMY.ACTOR_PID,
        CASE WHEN NMY.NUM_OF_MOV_BY_YC > IFNULL(NMO.MAX_NUM_OF_MOV,0) THEN 'Y' ELSE 'N' END
MORE_MOV_BY_YC
    FROM
        NUM_OF_MOVIES_BY_YC NMY
        LEFT OUTER JOIN
        MAX_MOV_BY_OTHER_DIRECTORS NMO
        ON
        NMY.ACTOR_PID = NMO.ACTOR_PID
    )
    SELECT
        DISTINCT
        TRIM(P.Name) ACTOR_NAME
    FROM
        Person P
    WHERE
        TRIM(P.PID) IN (
            SELECT
                DISTINCT
                ACTOR_PID
            FROM
                ACTORS_MOV_COMPARISION
            WHERE
                MORE_MOV_BY_YC = 'Y');''', conn)
query9
```

Out[134]:

	ACTOR_NAME

1. 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 [136]:

```
query10 = pd.read_sql_query('''WITH
    SHAHRUKH_0 AS
    (
        SELECT
            TRIM(P.PID) PID
        FROM
            Person P
        WHERE
            Trim(P.Name) like '%Shahrukh%'
    ),
    SHAHRUKH_1_MOVIES AS
    (
        SELECT
            DISTINCT
            TRIM(MC.MID) MID,
            S0.PID
        FROM
            M_Cast MC,
            SHAHRUKH_0 S0
        WHERE
            TRIM(MC.PID) = S0.PID
    ),
    SHAHRUKH_1_ACTORS AS
    (
        SELECT
            DISTINCT
            TRIM(MC.PID) PID
        FROM
            M_Cast MC,
            SHAHRUKH_1_MOVIES S1M
        WHERE
            TRIM(MC.MID) = S1M.MID AND
            TRIM(MC.PID) <> S1M.PID
    ),
    SHAHRUKH_2_MOVIES AS
    (
        SELECT
            DISTINCT
            TRIM(MC.MID) MID,
            S1A.PID
        FROM
            M_Cast MC,
            SHAHRUKH_1_ACTORS S1A
        WHERE
            TRIM(MC.PID) = S1A.PID
    )
    SELECT
        DISTINCT
        TRIM(MC.PID) PID,
        TRIM(P.Name) ACTOR_NAME
    FROM
        Person P,
        M_Cast MC,
        SHAHRUKH_2_MOVIES S2M
    WHERE
            TRIM(MC.PID) = TRIM(P.PID) AND
            TRIM(MC.MID) = S2M.MID AND
            TRIM(MC.PID) <> S2M.PID;''', conn)
query10
```

Out[136]:

	PID	ACTOR_NAME
0	nm2951768	Freida Pinto
1	nm6467532	Caroline Christl Long
2	nm6071249	Rajeev Pahuja
3	nm3491108	Michelle Santiago
4	nm7509518	Jandre le Roux
5	nm5951787	Raj Awasti
6	nm5525290	Michael Chapman

	PID	ACTOR_NAME
7	nm8232648	James Heron
8	nm7247557	Alex Jaep
9	nm6631007	Marian Lorencik
10	nm7255636	Celina Nessa
11	nm5721141	James Pimenta
12	nm4964257	M'laah Kaur Singh
13	nm0380073	Maximiliano Hernández
14	nm3630374	Sohum Shah
15	nm3708961	Deepak Damle
16	nm8334880	Piyush Kaushik
17	nm1390115	Harish Khanna
18	nm3818286	Sushant Singh Rajput
19	nm0080232	Nitish Bharadwaj
20	nm8644385	Lalu Makhija
21	nm6661769	Mir Sarwar
22	nm4731677	Ayushmann Khurrana
23	nm0007102	Tabu
24	nm2331000	Radhika Apte
25	nm0223521	Anil Dhawan
26	nm2435847	Manav Vij
27	nm1817397	Ashwini Kalsekar
28	nm3777127	Chhaya Kadam
29	nm1664541	Zakir Hussain
...	...	...
15614	nm2019990	Pradhan Manjari
15615	nm2021136	Poonam Jha
15616	nm1459053	Sunila Karambelkar
15617	nm1763457	Arup Ganguli
15618	nm1760405	Laxmi Patel
15619	nm1760399	Meena Pankaj
15620	nm1686154	Pratap
15621	nm2184586	Vidya Shenoy
15622	nm2182462	Jeetendra Khanna
15623	nm2177257	K.L. Sethi
15624	nm2084775	Malaika Shinoy
15625	nm2465676	Poonam Bajwa
15626	nm1688585	Kishin Punjabi
15627	nm1082198	Surjeet Redi
15628	nm0695939	Premji
15629	nm1693988	Kamu
15630	nm5578623	Monal
15631	nm1524755	Ushma Rathod
15632	nm1567918	Shilpi
15633	nm1946131	Zubeda Khan
15634	nm2519512	N. Sagar
[illegible]	[illegible]	[illegible]

	PID	ACTOR_NAME
15635	nm4042918	Habib Tanvar
15636	nm1881395	Mohd. Zahiruddin
15637	nm2522571	Muktha George
15638	nm0030135	Anjuman
15639	nm3099317	Dhruv Shetty
15640	nm2371614	Hayley Cleghorn
15641	nm2675737	Nirvasha Jithoo
15642	nm2370589	Kamal Maharshi
15643	nm1866356	Mohini Manik

15644 rows × 2 columns