0.1 Introduction

```
Assignment: SQL analysis on IMDB data
```

The IMDB dataset here refers to the relational database whose schema is given ahead. It has 13 different tables. The table Movie is the main table which binds all the others directly or indirectly with the primary key `MID` . Let's start by importing the libraries we need.

In [1]:

```
import pandas as pd
import numpy as np
import os
import sqlite3
```

executed in 4ms, finished 21:04:05 2019-06-26

In [694]:

```
os.listdir()
```

executed in 4ms, finished 06:28:34 2019-06-25

Out[694]:

```
['questions.pdf',
 'imdb analysis using SQL.ipynb',
 'db_schema.jpeg',
 'Db-IMDB.db',
 '.ipynb_checkpoints']
```

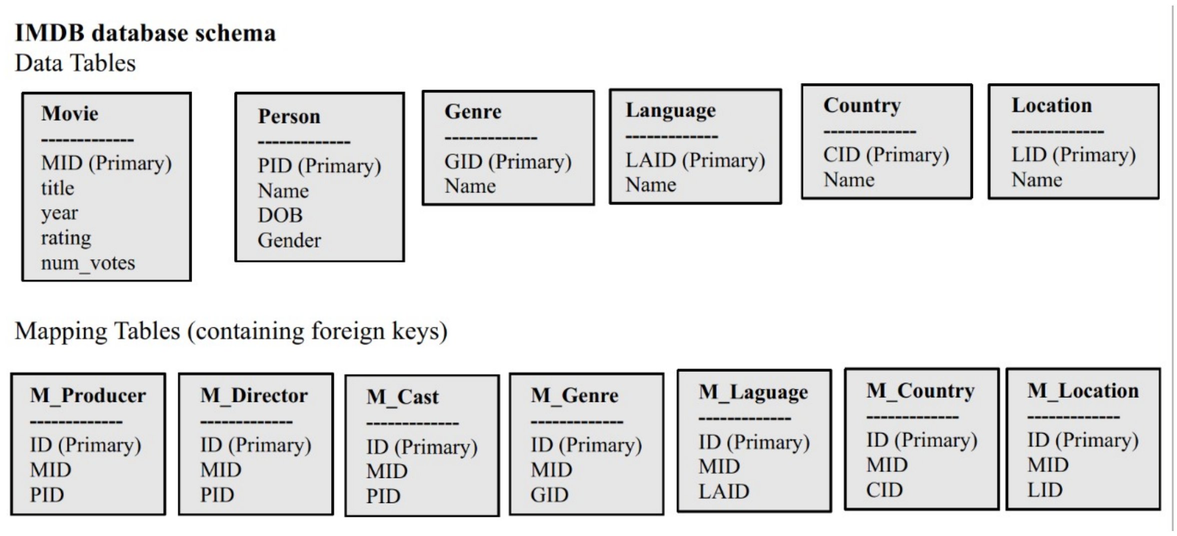
0.2 The schema

In [3]:

```
from IPython.display import Image
Image('db_schema.jpeg')
```

executed in 67ms, finished 17:57:06 2019-06-23

Out[3]:


IMDB database schema
Data Tables
Movie ------------- MID (Primary) title year rating num_votes
Person ------------- PID (Primary) Name DOB Gender
Genre ------------- GID (Primary) Name
Language ------------- LAID (Primary) Name
Country ------------- CID (Primary) Name
Location ------------- LID (Primary) Name
Mapping Tables (containing foreign keys)
M_Producer ------------- ID (Primary) MID PID
M_Director ------------- ID (Primary) MID PID
M_Cast ------------- ID (Primary) MID PID
M_Genre ------------- ID (Primary) MID GID
M_Laguage ------------- ID (Primary) MID LAID
M_Country ------------- ID (Primary) MID CID
M_Location ------------- ID (Primary) MID LID

In [3]:

```
os.path.abspath('questions.pdf')
```

executed in 12ms, finished 18:07:35 2019-06-23

Out[3]:

```
'/home/tanmay/work/github_projects/my_projects/imdb/questions.pdf'
```

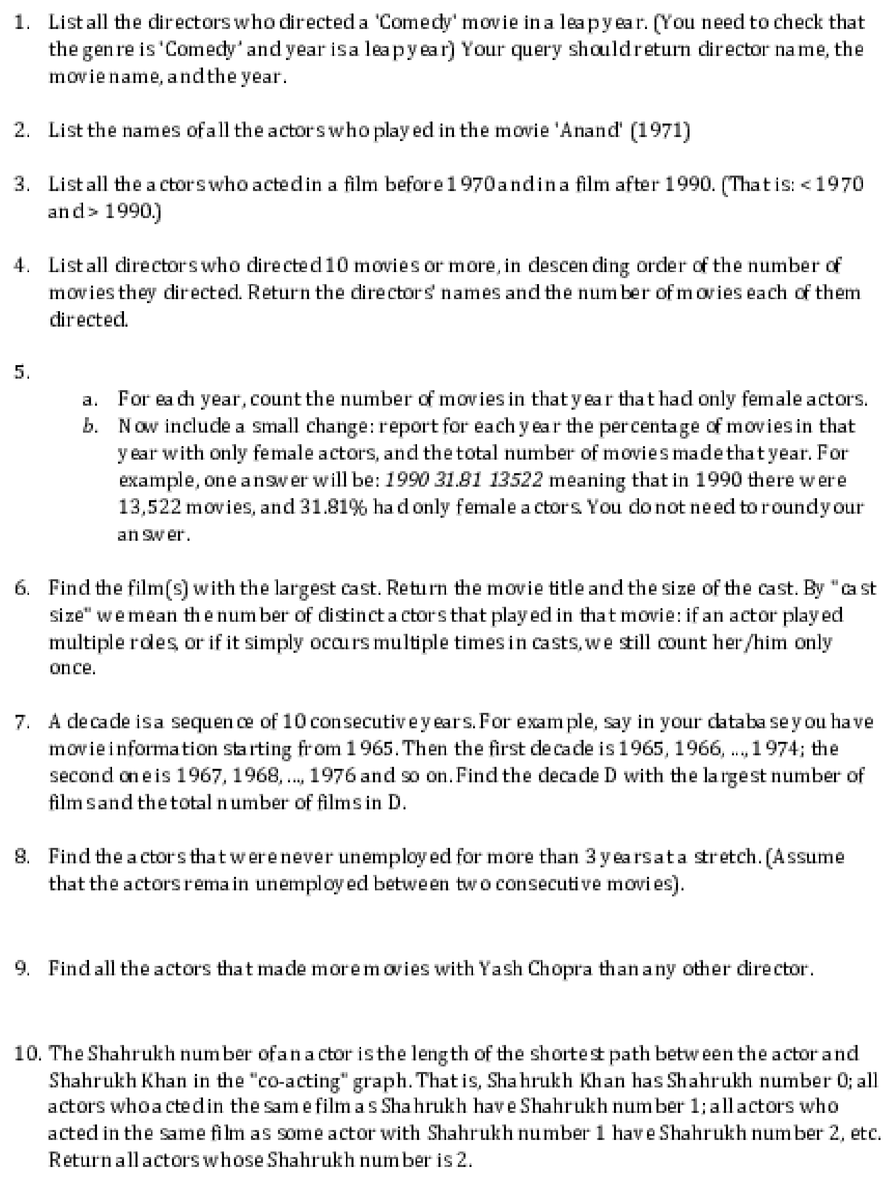
0.3 The questions

In [696]:

```python
from wand.image import Image as WImage
img = WImage(filename=r'questions.pdf', height=900, width=700)
img
```

executed in 3.38s, finished 06:29:47 2019-06-25

Out[696]:



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.

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

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

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.

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.

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.

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.

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).

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

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.

0.4 Connect to the sqlite database

In [151]:

```
con = sqlite3.connect('Db-IMDB.db')

df_movie = pd.read_sql_query('select * from movie', con)
df_genre_mapping = pd.read_sql_query('select * from M_genre', con)
df_director = pd.read_sql_query('select * from m_director', con)
df_genre = pd.read_sql_query('select * from Genre', con)
df_person = pd.read_sql_query('select * from person', con)
df_cast_mapping = pd.read_sql_query('select * from M_cast', con)
```

executed in 419ms, finished 21:49:52 2019-06-26

In [3]:

```
df_movie.head()
```

executed in 65ms, finished 21:05:06 2019-06-26

Out[3]:

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

0.5 Clean the data

There are certain quirks in this data, such as extra spaces at the beginning of `Name`. Here's an example. There are two separate records for the person `Andy Serkis`. Note the space at the beginning of the second name.

In [115]:

```
df_person[df_person.Name.isin(
    ['Andy Serkis', ' Andy Serkis', 'Ang Lee', ' Ang Lee'])]
```

executed in 32ms, finished 21:40:00 2019-06-26

Out[115]:

	PID	Name	Gender
4	nm0785227	Andy Serkis	Male
32253	nm0000487	Ang Lee	None
36821	nm0785227	Andy Serkis	Male
36837	nm0000487	Ang Lee	None

Extra characters at the beginning of `year`

In [5]:

```
df_movie.year.value_counts().tail(10)
```

executed in 20ms, finished 21:05:11 2019-06-26

Out[5]:

```
III 2017    1
I 2005      1
I 1997      1
II 2009     1
I 1983      1
III 2016    1
I 2012      1
I 1996      1
VI 2015     1
IV 2017     1
Name: year, dtype: int64
```

In [152]:

```
# Removing trailing spaces from people's names
df_person.Name = df_person.Name.str.strip()

# Strip invalid characters from the column "year"
df_movie.year = df_movie.year.str.strip('I ').str.strip('II ').str.strip(
    'III ').str.strip('IV ').str.strip('V ').str.strip('VI ').str.strip(
        'XVII ').str.strip()

# strip spaces from PID and MID
df_person.PID = df_person.PID.str.strip()
df_director.PID = df_director.PID.str.strip()
df_movie.MID = df_movie.MID.str.strip()
df_director.MID = df_director.MID.str.strip()
df_cast_mapping.PID = df_cast_mapping.PID.str.strip()
```

executed in 154ms, finished 21:49:56 2019-06-26

In [153]:

```
# create new column YEAR in movies which is a datetime64 object
df_movie['YEAR'] = df_movie.year.astype('datetime64')
```

executed in 16ms, finished 21:50:01 2019-06-26

In [154]:

```
df_movie.year = df_movie.year.astype('int')
```

executed in 13ms, finished 21:50:03 2019-06-26

0.6 Drop unnecessary index columns

In [ ]:

```
df_cast_mapping.drop('index', axis=1, inplace=True)
df_director.drop(['index', 'ID'], axis=1, inplace=True)
df_genre.drop('index', axis=1, inplace=True)
df_genre_mapping.drop(['index','ID'], axis=1, inplace=True)
df_movie.drop('index', axis=1, inplace=True)
df_person.drop('index', axis=1, inplace=True)
```

executed in 60ms, finished 21:51:14 2019-06-26

1 Q. Directors who directed a comedy movie in leap year

In [117]:

```
df_01 = pd.merge(pd.merge(pd.merge(pd.merge(df_movie,
                                             df_genre_mapping,
                                             on='MID'),
                                   df_genre,
                                   on='GID'),
                         df_director,
                         on='MID'),
               df_person,
               on='PID',
               suffixes=('_genre', '_director'))
```

executed in 122ms, finished 21:42:31 2019-06-26

In [119]:

```
df_01[(df_01.Name_genre.str.contains('Comedy'))
      & (df_01.YEAR.dt.is_leap_year)].drop_duplicates()[[
          'Name_director', 'title', 'year', 'Name_genre'
      ]]
```

executed in 98ms, finished 21:42:47 2019-06-26

Out[119]:

	Name_director	title	year	Name_genre
7	Anurag Kashyap	Gangs of Wasseypur	2012	Action, Comedy, Crime
41	Priyadarshan	Hera Pheri	2000	Action, Comedy, Crime
47	Priyadarshan	Kamaal Dhamaal Malamaal	2012	Comedy, Drama
48	Priyadarshan	Muskurahat	1992	Comedy, Drama
51	Priyadarshan	Mere Baap Pehle Aap	2008	Comedy, Drama, Romance
54	Priyadarshan	Hulchul	2004	Action, Comedy, Drama
151	David Dhawan	Saajan Chale Sasural	1996	Comedy, Drama
155	David Dhawan	Mujhse Shaadi Karogi	2004	Comedy, Drama, Romance
157	David Dhawan	Kunwara	2000	Comedy, Drama, Romance
167	David Dhawan	Dulhan Hum Le Jayenge	2000	Comedy, Drama, Romance
177	David Dhawan	Bol Radha Bol	1992	Action, Comedy, Drama
183	David Dhawan	Chal Mere Bhai	2000	Comedy, Romance
187	David Dhawan	Loafer	1996	Action, Comedy, Romance
207	Anees Bazmee	Singh Is Kinng	2008	Action, Comedy, Crime
239	Mahesh Bhatt	Papa Kahte Hain	1996	Comedy
365	Raj Kanwar	Deewana	1992	Action, Comedy, Crime
381	Raj Kanwar	Har Dil Jo Pyar Karega...	2000	Comedy, Drama, Musical
395	Prakash Mehra	Hera Pheri	1976	Action, Comedy, Crime
401	Prakash Mehra	Sharaabi	1984	Comedy, Drama, Romance
452	Rajeev Kumar	Raja Ko Rani Se Pyar Ho Gaya	2000	Comedy, Musical, Romance
454	Deepak S. Shivdasani	Mr. White Mr. Black	2008	Action, Comedy, Crime
479	Naresh Kumar	Gora Aur Kala	1972	Action, Comedy, Crime
485	Eeshwar Nivas	My Name Is Anthony Gonsalves	2008	Action, Comedy, Crime
489	Eeshwar Nivas	De Taali	2008	Comedy, Drama, Romance
516	Rajnish Thakur	Mere Dost Picture Abhi Baaki Hai	2012	Comedy
541	Indra Kumar	Great Grand Masti	2016	Comedy, Drama, Fantasy
553	Indra Kumar	Masti	2004	Comedy, Crime, Mystery
588	Chandrakant Singh	Rama Rama Kya Hai Dramaaa	2008	Comedy, Drama, Romance
608	Rakesh Roshan	Khel	1992	Comedy, Drama, Romance
701	Gauri Shinde	English Vinglish	2012	Comedy, Drama, Family
...	...	...	...	...
4743	Hriday Shetty	Chaalis Chauraasi	2012	Comedy, Crime

	Name_director	title	year	Name_genre
4780	Brij	Victoria No. 203	1972	Comedy, Drama, Musical
4788	Brij	Bombay 405 Miles	1980	Comedy, Action
4818	S.S. Rajamouli	Eega	2012	Action, Comedy, Fantasy
4848	Krishna Vamsi	Ninne Pelladatha	1996	Romance, Comedy, Drama
4967	Akashdeep	Santa Banta Pvt Ltd	2016	Action, Comedy
4995	Shirish Kunder	Joker	2012	Comedy, Family, Sci-Fi
5222	Mike Judge	Beavis and Butt-Head Do America	1996	Animation, Adventure, Comedy
5245	K.S. Ravi	Mr. Romeo	1996	Action, Comedy, Romance
5270	Shyam Ramsay	Purana Mandir	1984	Comedy, Drama, Horror
5333	Sajid	Housefull 3	2016	Action, Comedy
5337	Arbaaz Khan	Dabangg 2	2012	Action, Comedy
5341	Ashish R. Mohan	Khiladi 786	2012	Action, Comedy
5345	Sunil K. Reddy	Thikka	2016	Action, Comedy
5351	Bobby Kolli	Sardaar Gabbar Singh	2016	Action, Comedy
5357	Amma Rajasekhar	Sathyam	2008	Action, Comedy
5358	Madonna	Filth and Wisdom	2008	Comedy, Drama, Music
5378	Dibakar Banerjee	Oye Lucky! Lucky Oye!	2008	Comedy, Crime, Drama
5427	Jaideep Sen	Krazzy 4	2008	Comedy, Crime, Drama
5428	Saurabh Kabra	EMI: Liya Hai To Chukana Padega	2008	Comedy, Crime, Drama
5430	Shashi Ranjan	Dhoom Dadakka	2008	Comedy, Crime, Drama
5502	Jyoti Swaroop	Padosan	1968	Comedy, Musical, Romance
5572	Ganapathy Bharat	Hari Om	2004	Romance, Comedy
5577	Sanjay M. Khanduri	Kismet Love Paisa Dilli	2012	Adventure, Comedy, Crime
5594	K. Shankar	Rajkumar	1964	Musical, Drama, Comedy
5632	Rajpal Yadav	Ata Pata Lapatta	2012	Comedy, Musical
5724	Abhishek Jain	Kevi Rite Jaish	2012	Comedy, Family
5728	Ram Mukherjee	Leader	1964	Comedy, Romance, Thriller
5772	Aspi Irani	Garam Masala	1972	Comedy, Musical, Action
5778	Salim Raza	Bach ke Zara	2008	Comedy, Horror, Musical

232 rows × 4 columns

2 Q. Names of all actors in the movie 'Anand'

In [121]:

```
pd.merge(pd.merge(df_movie[df_movie.title == 'Anand'],
                  df_cast_mapping,
                  on='MID'),
         df_person,
         on='PID')[['Name', 'title', 'Gender', 'year']]
```

executed in 111ms, finished 21:43:58 2019-06-26

Out[121]:

	Name	title	Gender	year
0	Rajesh Khanna	Anand	Male	1971
1	Amitabh Bachchan	Anand	Male	1971
2	Sumita Sanyal	Anand	Female	1971
3	Ramesh Deo	Anand	Male	1971
4	Seema Deo	Anand	Female	1971
5	Asit Kumar Sen	Anand	Male	1971
6	Dev Kishan	Anand	Male	1971
7	Atam Prakash	Anand	Male	1971
8	Lalita Kumari	Anand	Female	1971
9	Savita	Anand	Female	1971
10	Brahm Bhardwaj	Anand	Male	1971
11	Gurnam Singh	Anand	Male	1971
12	Lalita Pawar	Anand	Female	1971
13	Durga Khote	Anand	Female	1971
14	Dara Singh	Anand	Male	1971
15	Johnny Walker	Anand	Male	1971
16	Moolchand	Anand	Male	1971

3 Q. All the actors who acted in a film before 1970 and after 1990

In [122]:

```
g_03 = pd.merge(pd.merge(df_movie, df_cast_mapping, on='MID'),
                df_person,
                on='PID').groupby('Name')
```

executed in 228ms, finished 21:44:19 2019-06-26

In [123]:

```
df_03 = g_03.filter(lambda x: ((x['year'] < 1970).any() & (x['year'] > 1990).
                               any()))
```

executed in 30.9s, finished 21:44:50 2019-06-26

In [126]:

```python
df_03.sort_values(['Name', 'year'])[['Name', 'title','year']].head(5)
```

executed in 46ms, finished 21:45:41 2019-06-26

Out[126]:

	Name	title	year
12107	A.K. Hangal	Teesri Kasam	1966
12109	A.K. Hangal	Shagird	1967
12120	A.K. Hangal	Saat Hindustani	1969
12094	A.K. Hangal	Guddi	1971
12134	A.K. Hangal	Mere Apne	1971

In [128]:

```
df_03.groupby(['Name'])['year'].agg(['min', 'max']).reset_index().rename(
    {
        'min': 'min year',
        'max': 'max year'
    }, axis=1).head(50)
```

executed in 54ms, finished 21:46:05 2019-06-26

Out[128]:

	Name	min year	max year
0	A.K. Hangal	1966	2012
1	Aachi Manorama	1966	2013
2	Abbas	1948	2006
3	Abdul	1949	2017
4	Abhi Bhattacharya	1956	1994
5	Achala Sachdev	1954	2008
6	Adil	1958	2016
7	Ajay	1955	2017
8	Ajit	1952	2006
9	Akashdeep	1960	2010
10	Akbar Bakshi	1961	1997
11	Alka	1969	2006
12	Allu Ramalingaiah	1957	2001
13	Altaf	1964	2013
14	Amar	1948	2000
15	Amarnath	1962	2016
16	Ameer	1966	2005
17	Amitabh Bachchan	1969	2018
18	Amjad Khan	1957	1996
19	Amol Sen	1962	1999
20	Amrit	1969	2003
21	Anand	1957	2016
22	Anand Kumar	1962	2014
23	Anand Tiwari	1969	2015
24	Anil	1958	2018
25	Anil Kumar	1962	2015
26	Anil Nagrath	1966	2013
27	Anjali Kadam	1969	1994
28	Anju Mahendru	1967	2012
29	Anoop Kumar	1958	1993
30	Arun	1960	2011

	Name	min year	max year
31	Aruna Irani	1961	2011
32	Asha	1953	2010
33	Asha Parekh	1959	1999
34	Ashok Kumar	1936	2015
35	Ashwani Kumar	1965	2018
36	Asit Kumar Sen	1953	1996
37	Asrani	1969	2018
38	Atul Kumar	1968	2018
39	Aziz	1964	1995
40	B.M. Vyas	1946	1995
41	Baba	1961	2008
42	Babbanlal Yadav	1967	2001
43	Babloo	1961	2005
44	Baby Deepali	1968	1992
45	Baby Sonu	1961	2002
46	Balbir	1958	2014
47	Begum Para	1947	2007
48	Bharat Bhushan	1952	2016
49	Bharati Devi	1963	2006

4 Q. Directors who directed 10 or more movies

In [129]:

```
df_04 = pd.merge(pd.merge(
    df_movie, df_director, on='MID'), df_person, on='PID').drop_duplicates()
```

executed in 110ms, finished 21:46:22 2019-06-26

In [130]:

```
df_04.head()
```

executed in 43ms, finished 21:46:24 2019-06-26

Out[130]:

	MID	title	year	rating	num_votes	YEAR	decade	PID	Name	Gender
0	tt2388771	Mowgli	2018	6.6	21967	2018-01-01	9	nm0785227	Andy Serkis	Male
2	tt5164214	Ocean's Eight	2018	6.2	110861	2018-01-01	9	nm0002657	Gary Ross	None
3	tt1365519	Tomb Raider	2018	6.4	142585	2018-01-01	9	nm1012385	Roar Uthaug	None
4	tt0848228	The Avengers	2012	8.1	1137529	2012-01-01	9	nm0923736	Joss Whedon	None
5	tt8239946	Tumbbad	2018	8.5	7483	2018-01-01	9	nm9751348	Rahi Anil Barve	None

In [132]:

```
g_04 = df_04.groupby('PID')

g_04.filter(lambda x: x['MID'].nunique() >= 10).groupby(
    'Name', as_index=False)['MID'].count().sort_values(
        'MID', ascending=False).rename({'MID': 'Number of movies directed'},
                                        axis=1).head(20)
```

executed in 920ms, finished 21:46:43 2019-06-26

Out[132]:

	Name	Number of movies directed
7	David Dhawan	39
23	Mahesh Bhatt	35
43	Ram Gopal Varma	30
35	Priyadarshan	30
55	Vikram Bhatt	29
14	Hrishikesh Mukherjee	27
57	Yash Chopra	21
5	Basu Chatterjee	19
48	Shakti Samanta	19
50	Subhash Ghai	18
49	Shyam Benegal	17
44	Rama Rao Tatineni	17
0	Abbas Alibhai Burmawalla	17
11	Gulzar	16
40	Raj N. Sippy	16
25	Manmohan Desai	16
24	Mahesh Manjrekar	15
37	Raj Kanwar	15
36	Rahul Rawail	14
41	Rajkumar Santoshi	14

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

In [133]:

```
df_05_1 = pd.merge(pd.merge(df_movie, df_cast_mapping, on='MID'),
                   df_person,
                   on='PID')
df_05_1.head(5)
```

executed in 179ms, finished 21:47:13 2019-06-26

Out[133]:

	MID	title	year	rating	num_votes	YEAR	decade	PID	ID	Name
0	tt2388771	Mowgli	2018	6.6	21967	2018-01-01	9	nm0000288	0	Christian Bale
1	tt2388771	Mowgli	2018	6.6	21967	2018-01-01	9	nm0000949	1	Cate Blanchett
2	tt5164214	Ocean's Eight	2018	6.2	110861	2018-01-01	9	nm0000949	47	Cate Blanchett
3	tt2388771	Mowgli	2018	6.6	21967	2018-01-01	9	nm1212722	2	Benedict Cumberbatch
4	tt2388771	Mowgli	2018	6.6	21967	2018-01-01	9	nm0365140	3	Naomie Harris

In [134]:

```
df_05_1.Gender.unique()
```

executed in 29ms, finished 21:47:20 2019-06-26

Out[134]:

```
array(['Male', 'Female', None], dtype=object)
```

In [135]:

```
_only_female_cast = df_05_1.groupby(
    'MID').filter(lambda x: (x['Gender'] == 'Female').all()).groupby(
        'year', as_index=False)['MID'].count().rename(
            {'MID': 'Number of movies with only Female actors'}, axis=1)
```

executed in 2.69s, finished 21:47:38 2019-06-26

In [136]:

```
_only_female_cast
```

executed in 27ms, finished 21:47:39 2019-06-26

Out[136]:

	year	Number of movies with only Female actors
0	1939	2
1	1999	11
2	2000	11
3	2018	2

5.1 Q. year, only female actors, total number of movies

In [137]:

```python
_movies_per_year = df_05_1.groupby(
    'year', as_index=True)['MID'].nunique().reset_index().rename(
        {'MID': 'Total Number of Movies'}, axis=1)
```

executed in 161ms, finished 21:47:47 2019-06-26

In [138]:

```
_comparison = pd.merge(_movies_per_year, _only_female_cast, how='left', on =
_comparison
```

executed in 74ms, finished 21:47:52 2019-06-26

Out[138]:

	year	Total Number of Movies	Number of movies with only Female actors
0	1931	1	0.0
1	1936	3	0.0
2	1939	2	2.0
3	1941	1	0.0
4	1943	1	0.0
5	1946	2	0.0
6	1947	2	0.0
7	1948	3	0.0
8	1949	3	0.0
9	1950	2	0.0
10	1951	6	0.0
11	1952	6	0.0
12	1953	8	0.0
13	1954	6	0.0
14	1955	9	0.0
15	1956	6	0.0
16	1957	13	0.0
17	1958	9	0.0
18	1959	6	0.0
19	1960	14	0.0
20	1961	7	0.0
21	1962	12	0.0
22	1963	10	0.0
23	1964	15	0.0
24	1965	14	0.0
25	1966	18	0.0
26	1967	19	0.0
27	1968	21	0.0
28	1969	18	0.0
29	1970	24	0.0
...	...	...	...
48	1989	47	0.0
49	1990	42	0.0

	year	Total Number of Movies	Number of movies with only Female actors
50	1991	41	0.0
51	1992	58	0.0
52	1993	63	0.0
53	1994	60	0.0
54	1995	56	0.0
55	1996	60	0.0
56	1997	55	0.0
57	1998	55	0.0
58	1999	66	11.0
59	2000	64	11.0
60	2001	73	0.0
61	2002	87	0.0
62	2003	103	0.0
63	2004	103	0.0
64	2005	129	0.0
65	2006	101	0.0
66	2007	109	0.0
67	2008	107	0.0
68	2009	110	0.0
69	2010	125	0.0
70	2011	116	0.0
71	2012	111	0.0
72	2013	136	0.0
73	2014	126	0.0
74	2015	119	0.0
75	2016	129	0.0
76	2017	126	0.0
77	2018	104	2.0

78 rows × 3 columns

In [139]:

```
_comparison[
    'percent_movies_with_only_female_actors'] = _comparison['Number of movies
        'Total Number of Movies'] * 100
```

executed in 11ms, finished 21:47:59 2019-06-26

In [140]:

```
_comparison
```

executed in 59ms, finished 21:48:15 2019-06-26

Out[140]:

	year	Total Number of Movies	Number of movies with only Female actors	percent_movies_with_only_female_actors
0	1931	1	0.0	0.000000
1	1936	3	0.0	0.000000
2	1939	2	2.0	100.000000
3	1941	1	0.0	0.000000
4	1943	1	0.0	0.000000
5	1946	2	0.0	0.000000
6	1947	2	0.0	0.000000
7	1948	3	0.0	0.000000
8	1949	3	0.0	0.000000
9	1950	2	0.0	0.000000
10	1951	6	0.0	0.000000
11	1952	6	0.0	0.000000
12	1953	8	0.0	0.000000
13	1954	6	0.0	0.000000
14	1955	9	0.0	0.000000
15	1956	6	0.0	0.000000
16	1957	13	0.0	0.000000
17	1958	9	0.0	0.000000
18	1959	6	0.0	0.000000
19	1960	14	0.0	0.000000
20	1961	7	0.0	0.000000
21	1962	12	0.0	0.000000
22	1963	10	0.0	0.000000
23	1964	15	0.0	0.000000
24	1965	14	0.0	0.000000
25	1966	18	0.0	0.000000
26	1967	19	0.0	0.000000
27	1968	21	0.0	0.000000
28	1969	18	0.0	0.000000
29	1970	24	0.0	0.000000
...	...	...	...	...
48	1989	47	0.0	0.000000
49	1990	42	0.0	0.000000

	year	Total Number of Movies	Number of movies with only Female actors	percent_movies_with_only_female_actors
50	1991	41	0.0	0.000000
51	1992	58	0.0	0.000000
52	1993	63	0.0	0.000000
53	1994	60	0.0	0.000000
54	1995	56	0.0	0.000000
55	1996	60	0.0	0.000000
56	1997	55	0.0	0.000000
57	1998	55	0.0	0.000000
58	1999	66	11.0	16.666667
59	2000	64	11.0	17.187500
60	2001	73	0.0	0.000000
61	2002	87	0.0	0.000000
62	2003	103	0.0	0.000000
63	2004	103	0.0	0.000000
64	2005	129	0.0	0.000000
65	2006	101	0.0	0.000000
66	2007	109	0.0	0.000000
67	2008	107	0.0	0.000000
68	2009	110	0.0	0.000000
69	2010	125	0.0	0.000000
70	2011	116	0.0	0.000000
71	2012	111	0.0	0.000000
72	2013	136	0.0	0.000000
73	2014	126	0.0	0.000000
74	2015	119	0.0	0.000000
75	2016	129	0.0	0.000000
76	2017	126	0.0	0.000000
77	2018	104	2.0	1.923077

78 rows × 4 columns

6 Q. Movies with the largest cast size

In [141]:

```
df_05 = pd.merge(df_movie, df_cast_mapping, on = 'MID')
```

executed in 107ms, finished 21:48:37 2019-06-26

In [142]:

```
df_05.groupby('title')[['PID']].nunique().rename({
    'PID': 'Cast Size'
}, axis=1).reset_index().sort_values('Cast Size', ascending=False).head(10)
```

executed in 212ms, finished 21:48:40 2019-06-26

Out[142]:

	title	Cast Size
2231	Ocean's Eight	238
297	Apaharan	233
1128	Gold	215
2138	My Name Is Khan	213
582	Captain America: Civil War	191
1091	Geostorm	170
2877	Striker	165
21	2012	154
2365	Pixels	144
3277	Yamla Pagla Deewana 2	140

7 Q. Decade D with the largest number of films

In [157]:

```
df_movie.year.min(), df_movie.year.max()
```

executed in 19ms, finished 21:52:10 2019-06-26

Out[157]:

```
(1931, 2018)
```

In [158]:

```
def decade(x):

    decades = [dec for dec in range(df_movie.year.min(), df_movie.year.max(),

    for ix, d in enumerate(decades):
        if x <= d:
            return ix + 1
    return ix + 1
```

executed in 19ms, finished 21:52:11 2019-06-26

In [159]:

```
df_movie = pd.concat(
    [df_movie, df_movie.year.map(decade).rename('decade')], axis=1)
```

executed in 871ms, finished 21:52:13 2019-06-26

In [160]:

```
df_movie.decade.value_counts(dropna=False).reset_index().rename(
    {
        'index': 'decade',
        'decade': 'Number of movies in decade'
    }, axis=1)
```

executed in 11ms, finished 21:52:13 2019-06-26

Out[160]:

	decade	Number of movies in decade
0	9	1941
1	8	610
2	7	369
3	6	270
4	5	175
5	4	84
6	3	19
7	2	6
8	1	1

8 Q. Actors who were never unemployed for more than 3 years

In [161]:

```
df_08 = pd.merge(pd.merge(df_movie, df_cast_mapping, on='MID'),
                 df_person,
                 on='PID')
```

executed in 187ms, finished 21:52:26 2019-06-26

In [162]:

```
df_08.drop_duplicates(subset=['MID', 'PID', 'year'], inplace=True)
```

executed in 151ms, finished 21:52:27 2019-06-26

In [163]:

```
df_081 = df_08.sort_values(['Name', 'year'])[['Name', 'PID', 'title', 'year']
```

executed in 106ms, finished 21:52:27 2019-06-26

In [164]:

```
# Added gap between consecutive movies
df_082 = pd.concat([
    df_081,
    df_081.groupby(df_081['PID'])['year'].diff().rename('gap in years').filln
],
                   axis=1)
```

executed in 7.94s, finished 21:52:36 2019-06-26

In [165]:

```
df_082.head()
```

executed in 10ms, finished 21:52:37 2019-06-26

Out[165]:

	Name	PID	title	year	gap in years
80632	'Ganja' Karuppu	nm2128968	Sandai Kozhi	2005	0.0
80633	'Ganja' Karuppu	nm2128968	Pazhani	2008	3.0
79480	'Lee' George Quinones	nm0704042	Bomb the System	2002	0.0
45616	'Musafir' Radio Performing	nm8644387	Rock On!!	2008	0.0
76882	'Nandha' Saravanan	nm5163714	Nandha	2001	0.0

In [166]:

```
g_082 = df_082.groupby('PID')
```

executed in 42ms, finished 21:52:42 2019-06-26

In [167]:

```
df_083 = g_082.filter( lambda x: (x['gap in years'] <= 3).all() )
```

executed in 20.2s, finished 21:53:03 2019-06-26

In [168]:

```
df_083['gap in years'].replace({0: np.nan}, inplace=True)
df_083.sample(20)
```

executed in 137ms, finished 21:53:04 2019-06-26

```
/home/tanmay/anaconda3/lib/python3.7/site-packages/pandas/core/generi
c.py:6586: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-
docs/stable/indexing.html#indexing-view-versus-copy (http://pandas.pyd
ata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy)
  self._update_inplace(new_data)
```

Out[168]:

	Name	PID	title	year	gap in years
86053	Janardhan Chowdary	nm2902521	50 Lakh	2007	NaN
23943	Vaunisha Kapoor	nm6870504	Guest iin London	2017	NaN
83374	Kinkar Sinha	nm1580148	Anubhav	1971	NaN
52919	Ronald Cardew	nm0136411	North West Frontier	1959	NaN
15254	Jane Hamer	nm5709457	Gold	2018	NaN
83372	Rainer F. Brusten	nm1158528	Anubhav	1971	NaN
83099	Chaitanya Sinh	nm1224983	Everybody Says I'm Fine!	2001	NaN
4032	Anurag Arora	nm3189957	Raees	2017	1.0
86476	Suneet Kochar	nm2897677	Life! Camera Action...	2012	NaN
73909	Frank Faylen	nm0269709	Footsteps in the Dark	1941	NaN
28278	Shreya Bhatt	nm3129191	Jab We Met	2007	NaN
86504	Jeetendra Bisht	nm3934882	Daayen Ya Baayen	2010	NaN
81141	Narendra Goswami	nm2191602	Chamatkar	1992	NaN
21738	Chuchok Ritnok	nm3307450	Street Fighter: The Legend of Chun-Li	2009	NaN
18941	Ali Zafar	nm3773554	Dear Zindagi	2016	2.0
61948	Dan Dhanoa	nm1058208	Karma	1986	1.0
51228	Fardeen Khan	nm0007228	Fida	2004	1.0
45442	Baby Shalu	nm1587092	Taqdeer	1983	2.0
8346	Om Puri	nm0700875	Fool N Final	2007	NaN
65459	Gautam Sharma	nm3083829	Bhindi Baazaar	2011	NaN

9 Q. More movies with Yash Chopra than any other directors

In [169]:

```
df_090 = pd.merge(pd.merge(pd.merge(df_movie, df_director, on='MID'),
                           df_person,
                           on='PID'),
                  df_cast_mapping,
                  on='MID',
                  suffixes=('_director', '_actor')).drop(
                      labels=['num_votes', 'decade', 'Gender'], axis=1)
```

executed in 241ms, finished 21:53:21 2019-06-26

In [170]:

```
df_090.rename({'Name':'director_name'}, axis=1, inplace=True)
```

executed in 61ms, finished 21:53:22 2019-06-26

In [171]:

```
df_090.head()
```

executed in 17ms, finished 21:53:22 2019-06-26

Out[171]:

	MID	title	year	rating	YEAR	PID_director	director_name	PID_actor	ID
0	tt2388771	Mowgli	2018	6.6	2018-01-01	nm0785227	Andy Serkis	nm0000288	0
1	tt2388771	Mowgli	2018	6.6	2018-01-01	nm0785227	Andy Serkis	nm0000949	1
2	tt2388771	Mowgli	2018	6.6	2018-01-01	nm0785227	Andy Serkis	nm1212722	2
3	tt2388771	Mowgli	2018	6.6	2018-01-01	nm0785227	Andy Serkis	nm0365140	3
4	tt2388771	Mowgli	2018	6.6	2018-01-01	nm0785227	Andy Serkis	nm0785227	4

In [172]:

```
g_090 = df_090.groupby('PID_actor')
```

executed in 23ms, finished 21:53:25 2019-06-26

In [173]:

```
df_091 = g_090.filter(lambda x: (x['director_name'] == 'Yash Chopra').any()
                      ).sort_values('PID_actor').drop_duplicates()
```

executed in 21.4s, finished 21:53:48 2019-06-26

In [174]:

```
df_091.sample(5)
```

executed in 41ms, finished 21:53:50 2019-06-26

Out[174]:

	MID	title	year	rating	YEAR	PID_director	director_name	PID_actor	I
63987	tt0222024	Hum Tumhare Hain Sanam	2002	5.5	2002-01-01	nm1147556	K.S. Adiyaman	nm0622186	1938
63157	tt0248012	Fiza	2000	6.2	2000-01-01	nm0006659	Khalid Mohamed	nm0006433	1872
110839	tt0313495	Prem Geet	1981	7.8	1981-01-01	nm0411541	Sudesh Issar	nm0044796	4102
56675	tt0101437	Beta	1992	6.3	1992-01-01	nm0409791	Indra Kumar	nm0438463	6558
42839	tt0119285	Hero No. 1	1997	6.1	1997-01-01	nm0223522	David Dhawan	nm0025630	3492

In [175]:

```
g_091 = df_091.groupby(['PID_actor', 'director_name'])
```

executed in 8ms, finished 21:53:52 2019-06-26

In [176]:

```
# total number of actors who've done at least one film with Yash Chopra than
df_091.PID_actor.nunique()
```

executed in 13ms, finished 21:53:55 2019-06-26

Out[176]:

```
430
```

In [177]:

```
df_092 = g_091['MID'].count().reset_index().rename({
    'MID': 'count'
}, axis=1).sort_values(['PID_actor', 'count'],
                       ascending=False).drop_duplicates('PID_actor',
                                                        keep='first')

df_092.sample(20)
```

executed in 83ms, finished 21:54:02 2019-06-26

Out[177]:

	PID_actor	director_name	count
1328	nm0044343	Rakeysh Omprakash Mehra	2
7440	nm5138567	Yash Chopra	1
7208	nm3157251	Yash Chopra	1
4422	nm0611552	Karan Johar	3
5799	nm0837126	Asit Sen	1
7413	nm4958954	Yash Chopra	1
2132	nm0201711	Shakti Samanta	4
1748	nm0080149	Hrishikesh Mukherjee	5
5552	nm0789374	B.R. Chopra	3
7523	nm7076286	Yash Chopra	1
7139	nm2117890	Dilip Shukla	1
6594	nm1261150	Bharathiraja	1
2820	nm0420092	Manmohan Desai	7
3321	nm0451321	Aziz Mirza	5
7394	nm4807680	Yash Chopra	1
5665	nm0796496	Babbar Subhash	3
6479	nm1210840	Feroz Khan	1
474	nm0004434	Yash Chopra	7
1347	nm0044796	David Dhawan	3
2227	nm0239270	Ketan Mehta	1

In [178]:

```
# Number of actors who did the highest number of films with Yash Chopra is
# simply the number records in the above dataframe where director == Yash Cho
# because this df is sorted on the number of movies and duplicates for PID_ac
# have been removed

df_093 = df_092[df_092.director_name == 'Yash Chopra'].sort_values(
    'count', ascending=False)
```

executed in 6ms, finished 21:54:23 2019-06-26

In [179]:

```
df_093.PID_actor = df_093.PID_actor.map(dict(df_person[['PID', 'Name']].value
```

executed in 180ms, finished 21:54:29 2019-06-26

In [180]:

```
df_093.rename({'PID_actor': 'Actor name'}, axis=1).head(30)
```

executed in 37ms, finished 21:54:30 2019-06-26

Out[180]:

	Actor name	director_name	count
4961	Jagdish Raj	Yash Chopra	11
3766	Manmohan Krishna	Yash Chopra	10
2650	Iftekhar	Yash Chopra	9
474	Shashi Kapoor	Yash Chopra	7
2479	Rakhee Gulzar	Yash Chopra	5
5030	Waheeda Rehman	Yash Chopra	5
5732	Neetu Singh	Yash Chopra	4
5167	Achala Sachdev	Yash Chopra	4
2055	Sudha Chopra	Yash Chopra	3
1942	Leela Chitnis	Yash Chopra	3
6984	Shyam Arora	Yash Chopra	2
4214	Nissar	Yash Chopra	2
7032	Ashok Verma	Yash Chopra	2
7001	Chandu Allahabadi	Yash Chopra	2
1007	Yash Chopra	Yash Chopra	2
5878	Surendra Nath	Yash Chopra	2
7210	Nazir	Yash Chopra	2
6620	Raj Hans	Yash Chopra	2
7107	Chandni Jas Keerat	Yash Chopra	1
7012	Sanjeev Kohli	Yash Chopra	1
7030	Ravi Dubey	Yash Chopra	1
7035	Pratima Puri	Yash Chopra	1
7055	Master Rizwan	Yash Chopra	1
7125	Vinod Negi	Yash Chopra	1
7108	Manish Arora	Yash Chopra	1
7109	Huzefa Gadiwala	Yash Chopra	1
7164	Nick Thomas-Webster	Yash Chopra	1
7159	Pankaj Raina	Yash Chopra	1
7191	Ramanand	Yash Chopra	1
7190	Kishan	Yash Chopra	1

10 Q. Shahrukh number

In [181]:

```python
df_10 = pd.merge(pd.merge(df_movie, df_cast_mapping, on='MID'),
                 df_person,
                 on='PID')  #.query("Name == 'Shahrukh Khan'")
```

executed in 173ms, finished 21:54:44 2019-06-26

Find out all movies SRK was in

In [182]:

```python
srk_movies = df_10.query("Name == 'Shah Rukh Khan'")['MID'].unique().tolist()
```

executed in 128ms, finished 21:54:46 2019-06-26

Get index of all records where movie is an srk movie and it's not SRK himself

In [183]:

```python
index_srk_co_actors = df_10[(df_10.MID.isin(srk_movies))
                            & (df_10.Name != 'Shah Rukh Khan')].index
```

executed in 54ms, finished 21:54:49 2019-06-26

Set the srk_number of all those records to 1

In [184]:

```python
df_10.loc[index_srk_co_actors, 'srk_number'] = 1
```

executed in 23ms, finished 21:54:51 2019-06-26

Get the actors whose srk_number is 1

In [185]:

```python
srk_number_1_actors = df_10[df_10.srk_number == 1]['PID'].unique().tolist()
```

executed in 17ms, finished 21:54:52 2019-06-26

Get the movies of the actors with srk_number 1

In [186]:

```python
srk_number_1_movies = df_10[df_10.PID.isin(srk_number_1_actors) & (
    df_10.Name != 'Shah Rukh Khan') & (df_10.srk_number != 1)].MID.unique()
```

executed in 74ms, finished 21:54:53 2019-06-26

Set the srk_number of all those records to 2

In [187]:

```python
df_10.loc[df_10[df_10.MID.isin(srk_number_1_movies)].index, 'srk_number'] = 2
```

executed in 63ms, finished 21:54:55 2019-06-26

In [188]:

```
df_10.srk_number.value_counts(dropna=False)
```

executed in 8ms, finished 21:54:56 2019-06-26

Out[188]:

```
2.0    77645
NaN     5805
1.0     3402
Name: srk_number, dtype: int64
```

In [114]:

```
df_10.sample(30)
```

executed in 77ms, finished 21:37:44 2019-06-26

Out[114]:

	MID	title	year	rating	num_votes	YEAR	decade	PID	ID	
28849	tt0473367	Jaane Tu... Ya Jaane Na	2008	7.5	22562	2008-01-01	9	nm1405359	12262	
41709	tt0449999	Kabhi Alvida Naa Kehna	2006	6.1	16374	2006-01-01	9	nm1999213	13250	
19242	tt3309662	Jackpot	2013	2.2	635	2013-01-01	9	nm0787462	35571	
59815	tt0187109	Gurudev	1993	4.9	105	1993-01-01	8	nm0151539	54405	Cha
43178	tt2556308	Holiday	2014	7.4	22160	2014-01-01	9	nm3083004	20431	Aa
67168	tt1773015	Phas Gaye Re Obama	2010	7.5	4463	2010-01-01	9	nm4357108	29012	
48247	tt0079221	Gol Maal	1979	8.6	14778	1979-01-01	6	nm1193461	16310	
52658	tt0230991	Zabardast	1985	5.5	67	1985-01-01	7	nm1090174	58707	
65912	tt0158587	Dhund	1973	7.3	397	1973-01-01	6	nm0451387	56042	Pa
23985	tt0410952	Charas: A Joint Effort	2004	5.3	278	2004-01-01	9	nm0159167	58069	
28259	tt1395054	Once Upon a Time in Mumbaai	2010	7.4	14114	2010-01-01	9	nm3124900	18604	S
77615	tt1182908	Krazzy 4	2008	4.2	1766	2008-01-01	9	nm3166027	50120	
11855	tt1828289	Shagird	2011	7.0	1561	2011-01-01	9	nm0904503	34884	
55565	tt0086230	Sadma	1983	8.5	2587	1983-01-01	7	nm0481362	21185	
84045	tt4121522	Shuruaat Ka Interval	2014	7.6	57	2014-01-01	9	nm6845067	73262	P
54939	tt0110449	Mammo	1994	7.7	324	1994-01-01	8	nm1276263	74318	
12042	tt0346457	Mangal Pandey: The Rising	2005	6.7	8592	2005-01-01	9	nm1004422	14857	
85276	tt1990976	7 Welcome to London	2012	5.0	102	2012-01-01	9	nm4637429	77943	Lis
20052	tt7363076	Raid	2018	7.4	9462	2018-01-01	9	nm9615361	5168	
284	tt5164214	Ocean's Eight	2018	6.2	110861	2018-01-01	9	nm1811793	267	Ja

	MID	title	year	rating	num_votes	YEAR	decade	PID	ID	
64085	tt0139525	Phool Aur Patthar	1966	6.6	185	1966-01-01	5	nm0474932	29762	M
63159	tt0173081	Pyaar To Hona Hi Tha	1998	6.7	2480	1998-01-01	8	nm2520787	63701	S
80833	tt0215196	Split Wide Open	1999	6.4	212	1999-01-01	8	nm2715042	61062	
19723	tt0422689	Madurey	2004	4.6	1180	2004-01-01	9	nm0712437	37776	N
47656	tt1266583	Mumbai Meri Jaan	2008	7.8	4986	2008-01-01	9	nm3257460	15978	G
41171	tt0268216	Charas	1976	6.6	136	1976-01-01	6	nm0045119	46958	
14288	tt0071811	Manoranjan	1974	6.8	175	1974-01-01	6	nm0783996	47609	As
74422	tt0246095	Mutamestri	1993	7.1	242	1993-01-01	8	nm0576169	50292	
33078	tt1841542	Chillar Party	2011	7.5	5517	2011-01-01	9	nm4496851	35851	Ch
17350	tt0348172	Tehzeeb	2003	6.1	459	2003-01-01	9	nm0794363	63981	