In [86]:

```
!pip3 install db-sqlite3
```

```
Collecting db-sqlite3
Collecting db (from db-sqlite3)
Collecting antiorm (from db->db-sqlite3)
Installing collected packages: antiorm, db, db-sqlite3
Successfully installed antiorm-1.2.1 db-0.1.1 db-sqlite3-0.0.1
```

In [1]:

```
import pandas as pd
import sqlite3
```

In [2]:

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

In [3]:

```
data = pd.read_sql_query('''SELECT name from sqlite_master where type= "table";''',conn)
```

In [4]:

```
data
```

Out[4]:

	name
0	Movie
1	Genre
2	Language
3	Country
4	Location
5	M_Location
6	M_Country
7	M_Language
8	M_Genre
9	Person
10	M_Producer
11	M_Director
12	M_Cast

In [4]:

```
def execute_sql(q, c = conn):
    df = pd.read_sql_query(q,c )
    return df


def give_table(q,c=conn):
    print(pd.read_sql_query(q,c))


for i in data.name:
    print('_____')
```

```
    print('#'*10, i,'table')
    give_table(q = "select * from " + i )
```

```
_____________________________
########## Movie table
      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
...     ...        ...            ...   ...     ...        ...
3470   3470  tt0090611    Allah-Rakha  1986     6.2         96
3471   3471  tt0106270          Anari  1993     4.7        301
3472   3472  tt0852989  Come December  2006     5.7         57
3473   3473  tt0375882     Kala Jigar  1939     3.3        174
3474   3474  tt0375890         Kanoon  1994     3.2        103

[3475 rows x 6 columns]
_____________________________
########## Genre table
     index                                    Name  GID
0        0      Adventure, Drama, Fantasy           0
1        1          Action, Comedy, Crime           1
2        2     Action, Adventure, Fantasy           2
3        3      Action, Adventure, Sci-Fi           3
4        4        Drama, Horror, Thriller           4
..     ...                                     ...  ...
323    323  Animation, Adventure, Fantasy         323
324    324          Biography, Drama, War         324
325    325    Animation, Drama, Adventure         325
326    326                  Drama, Action         326
327    327         Drama, Mystery, Sci-Fi         327

[328 rows x 3 columns]
_____________________________
########## Language table
    index          Name  LAID
0       0       English     0
1       1       Marathi     1
2       2         Hindi     2
3       3     Cantonese     3
4       4        Telugu     4
5       5      Mandarin     5
6       6         Tamil     6
7       7       Punjabi     7
8       8        Danish     8
9       9        Korean     9
10     10       Spanish    10
11     11        Arabic    11
12     12          Urdu    12
13     13       Tibetan    13
14     14         Dutch    14
15     15       Kannada    15
16     16      Japanese    16
17     17       Bengali    17
18     18     Sinhalese    18
19     19     Malayalam    19
20     20        French    20
21     21       Russian    21
22     22      Gujarati    22
23     23      Sanskrit    23
24     24     Himachali    24
25     25          Zulu    25
26     26        German    26
27     27       Persian    27
28     28       Italian    28
29     29      Bhojpuri    29
30     30  Swiss German    30
31     31      Georgian    31

_____________________________
########## Country table
    index                  Name  CID
0       0                    UK    0
1       1                   USA    1
2       2                 India    2
3       3             Australia    3
```

```
4        4              Hong Kong    4
5        5                Germany    5
6        6                 Canada    6
7        7                Denmark    7
8        8                Belgium    8
9        9            South Korea    9
10      10                  China   10
11      11              Argentina   11
12      12                  Libya   12
13      13            Switzerland   13
14      14            Netherlands   14
15      15                 France   15
16      16               Pakistan   16
17      17                  Japan   17
18      18              Sri Lanka   18
19      19  United Arab Emirates   19
20      20                 Bhutan   20
21      21           Soviet Union   21
22      22                Iceland   22
23      23            Afghanistan   23
24      24           South Africa   24
25      25                  Egypt   25
26      26                Hungary   26
27      27                   Iran   27
28      28               Suriname   28
29      29                 Sweden   29
30      30                  Spain   30
31      31                  Italy   31
32      32            New Zealand   32
33      33                Georgia   33
```

```
########## Location table
     index                                                Name  LID
0        0                             Durban, South Africa    0
1        1                      New York City, New York, USA    1
2        2  Cape Town Film Studios, Cape Town, Western Cap...    2
3        3                     Pittsburgh, Pennsylvania, USA    3
4        4                             Atlanta, Georgia, USA    4
..     ...                                               ...  ...
554    554                       Perumbavoor, Kerala, India  554
555    555                     Jersey City, New Jersey, USA  555
556    556          Berner Oberland, Kanton Bern, Switzerland  556
557    557                       Aeroporti, Tbilisi, Georgia  557
558    558                              Aftab Studio, India  558

[559 rows x 3 columns]
```

```
########## M_Location table
      index        MID    LID    ID
0         0  tt2388771    0.0     0
1         1  tt5164214    1.0     1
2         2  tt1365519    2.0     2
3         3  tt0848228    3.0     3
4         4  tt8239946    NaN     4
...     ...        ...    ...   ...
3470   3470  tt0090611  219.0  3470
3471   3471  tt0106270    NaN  3471
3472   3472  tt0852989  164.0  3472
3473   3473  tt0375882    NaN  3473
3474   3474  tt0375890    7.0  3474

[3475 rows x 4 columns]
```

```
########## M_Country table
      index        MID  CID    ID
0         0  tt2388771  0.0     0
1         1  tt5164214  1.0     1
2         2  tt1365519  0.0     2
3         3  tt0848228  1.0     3
4         4  tt8239946  2.0     4
...     ...        ...  ...   ...
3470   3470  tt0090611  2.0  3470
3471   3471  tt0106270  2.0  3471
3472   3472  tt0852989  2.0  3472
3473   3473  tt0375882  2.0  3473
3474   3474  tt0375890  2.0  3474
```

```
[3475 rows x 4 columns]
_____________________________
########## M_Language table
      index        MID  LAID    ID
0         0  tt2388771     0     0
1         1  tt5164214     0     1
2         2  tt1365519     0     2
3         3  tt0848228     0     3
4         4  tt8239946     1     4
...     ...        ...   ...   ...
3470   3470  tt0090611     2  3470
3471   3471  tt0106270     2  3471
3472   3472  tt0852989     2  3472
3473   3473  tt0375882     2  3473
3474   3474  tt0375890     2  3474

[3475 rows x 4 columns]
_____________________________
########## M_Genre table
      index        MID  GID    ID
0         0  tt2388771    0     0
1         1  tt5164214    1     1
2         2  tt1365519    2     2
3         3  tt0848228    3     3
4         4  tt8239946    4     4
...     ...        ...  ...   ...
3470   3470  tt0090611   46  3470
3471   3471  tt0106270   20  3471
3472   3472  tt0852989   19  3472
3473   3473  tt0375882  309  3473
3474   3474  tt0375890   46  3474

[3475 rows x 4 columns]
_____________________________
########## Person table
       index        PID                   Name  Gender
0          0  nm0000288         Christian Bale    Male
1          1  nm0000949         Cate Blanchett  Female
2          2  nm1212722   Benedict Cumberbatch    Male
3          3  nm0365140          Naomie Harris  Female
4          4  nm0785227            Andy Serkis    Male
...      ...        ...                    ...     ...
38280  38280  nm1470989                 Kannan    None
38281  38281  nm0298158           Adrian Fulle    None
38282  38282  nm0474806          Gulshan Kumar    None
38283  38283  nm0066829                  Iqbal  Female
38284  38284  nm1421793       Sushma Shiromani  Female

[38285 rows x 4 columns]
_____________________________
########## M_Producer table
       index        MID         PID     ID
0          0  tt2388771   nm0057655      0
1          1  tt2388771   nm0147080      1
2          2  tt2388771   nm0389414      2
3          3  tt2388771   nm0460141      3
4          4  tt2388771   nm0672248      4
...      ...        ...         ...    ...
11746  11746  tt0852989   nm2371237  11746
11747  11747  tt0852989   nm2371184  11747
11748  11748  tt0852989   nm1246080  11748
11749  11749  tt0375882        None  11749
11750  11750  tt0375890   nm1421793  11750

[11751 rows x 4 columns]
_____________________________
########## M_Director table
      index        MID        PID    ID
0         0  tt2388771  nm0785227     0
1         1  tt5164214  nm0002657     1
2         2  tt1365519  nm1012385     2
3         3  tt0848228  nm0923736     3
4         4  tt8239946  nm9751348     4
...     ...        ...        ...   ...
3470   3470  tt0090611  nm0220823  3470
3471   3471  tt0106270  nm0613517  3471
3472   3472  tt0852989  nm2312263  3472
```

```
3473   3473  tt0375882  nm0066829  3473
3474   3474  tt0375890  nm1421793  3474

[3475 rows x 4 columns]
_____________________________
########## M_Cast table
       index        MID         PID     ID
0          0  tt2388771   nm0000288      0
1          1  tt2388771   nm0000949      1
2          2  tt2388771   nm1212722      2
3          3  tt2388771   nm0365140      3
4          4  tt2388771   nm0785227      4
...      ...        ...         ...    ...
82832  82832  tt0375890   nm0664109  82832
82833  82833  tt0375890   nm0505323  82833
82834  82834  tt0375890   nm0019427  82834
82835  82835  tt0375890   nm0197582  82835
82836  82836  tt0375890   nm0438467  82836

[82837 rows x 4 columns]
```

In [42]:

```
print(execute_sql('select * from movie limit 5'))
print('**************************************************')
print(execute_sql('select * from movie limit 5 offset 40'))
```

```
   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
**************************************************
   index        MID               title  year  rating  num_votes
0     40  tt0086034           Octopussy  1983     6.6      84600
1     41  tt0109424  Chung Hing sam lam  1994     8.1      50603
2     42  tt6452574               Sanju  2018     8.1      35436
3     43  tt5816682    Victoria & Abdul  2017     6.8      23051
4     44  tt7919680             Karwaan  2018     7.6       6333
```

In [50]:

```
execute_sql('select * from movie order by year desc limit 10')
```

Out[50]:

	index	MID	title	year	rating	num_votes
0	81	tt6206564	Trapped	XVII 2016	7.6	6814
1	412	tt4271730	Alone	VI 2015	3.8	1403
2	1549	tt4467202	Hero	V 2015	3.7	1950
3	2975	tt7399620	Game Over	IV 2017	6.9	65
4	2840	tt1702543	Lucky	IV 2011	7.1	233
5	2907	tt1948640	The Waiting Room	IV 2010	6.0	85
6	483	tt6926486	Daddy	III 2017	6.4	1239
7	130	tt5571734	Pink	III 2016	8.2	30231
8	726	tt4818930	Waiting	III 2015	7.2	1382
9	838	tt4603640	The Silence	III 2015	7.2	179

In [63]:

```
execute_sql('select DISTINCT RATING from movie LIMIT 10')
```

Out[63]:

	rating
0	6.6
1	6.2
2	6.4
3	8.1
4	8.5
5	5.5
6	7.8
7	9.0
8	5.7
9	5.3

Assignment

Q1:-

`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 [38]:

```
#Correct

execute_sql('select p.name, m.title,m.year from person p join m_director md on md.pid = p.pid join
m_genre mg on mg.mid = md.mid join genre g on g.gid = mg.gid join movie m on m.mid = mg.mid where
(((m.year % 4) == 0) and trim(g.name) like "Comedy")')
```

Out[38]:

	Name	title	year
0	Bhagyaraj	Mr. Bechara	1996
1	Bhagyaraj	Mr. Bechara	1996
2	Pankaj Parashar	Ab Ayega Mazaa	1984
3	Mahesh Bhatt	Papa Kahte Hain	1996
4	Mahesh Bhatt	Papa Kahte Hain	1996
5	Jabbar Patel	Ek Hota Vidushak	1992
6	Jabbar Patel	Ek Hota Vidushak	1992
7	Kawal Sharma	Maalamaal	1988
8	Srinivas Bhashyam	Paisa Vasool	2004
9	Raj Kaushal	Shaadi Ka Laddoo	2004
10	Raj Kaushal	Shaadi Ka Laddoo	2004
11	Siddharth Anand Kumar	Let's Enjoy	2004
12	Siddharth Anand Kumar	Let's Enjoy	2004
13	Govind Menon	Kis Kis Ki Kismat	2004
14	Govind Menon	Kis Kis Ki Kismat	2004
15	Sachin	Navra Mazha Navsacha	2004
16	Sachin	Navra Mazha Navsacha	2004
17	Karan Razdan	Mr Bhatti on Chutti	2012
18	Karan Razdan	Mr Bhatti on Chutti	2012
19	Anees Bazmee	Thank You I	2011
20	Anees Bazmee	Thank You I	2011

	Name	title	year
21	Sachin Yardi	Kyaa Super Kool Hain Hum	2012
22	Sameer Sharma	Luv Shuv Tey Chicken Khurana	2012
23	Anand Balraj	Daal Mein Kuch Kaala Hai	2012
24	Anand Balraj	Daal Mein Kuch Kaala Hai	2012
25	Rajnish Thakur	Mere Dost Picture Abhi Baaki Hai	2012
26	Rajnish Thakur	Mere Dost Picture Abhi Baaki Hai	2012
27	Vickrant Mahajan	Challo Driver	2012
28	Vickrant Mahajan	Challo Driver	2012
29	Jagdish Rajpurohit	Bumboo	2012
30	Jagdish Rajpurohit	Bumboo	2012
31	Rakesh Mehta	Life Ki Toh Lag Gayi	2012
32	Rakesh Mehta	Life Ki Toh Lag Gayi	2012
33	Nitin Kakkar	Filmistaan	2012
34	Aditya Datt	Will You Marry Me	2012
35	Milap Zaveri	Mastizaade	2016
36	Milap Zaveri	Mastizaade	2016
37	Umesh Ghadge	Kyaa Kool Hain Hum 3	2016
38	Umesh Ghadge	Kyaa Kool Hain Hum 3	2016
39	Abhishek Sharma	Tere Bin Laden: Dead Or Alive	2016
40	Abhishek Sharma	Tere Bin Laden: Dead Or Alive	2016
41	Sanjeev Sharma	Saat Uchakkey	2016
42	Sanjeev Sharma	Saat Uchakkey	2016
43	Krishnadev Yagnik	Days of Tafree	2016
44	Suhas Kadav	Motu Patlu: King of Kings	2016

Q2:-

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

In [50]:

```
#Correct
execute_sql('select p.name from person p join m_cast mc on trim(mc.pid) = p.pid join movie m on m.
mid = mc.mid where (m.title == "Anand")')
```

Out[50]:

	Name
0	Amitabh Bachchan
1	Rajesh Khanna
2	Brahm Bhardwaj
3	Ramesh Deo
4	Seema Deo
5	Dev Kishan
6	Durga Khote
7	Lalita Kumari
8	Lalita Pawar
9	Atam Prakash
10	Sumita Sanyal
11	Asit Kumar Sen
12	Dara Singh

13	Johnny Walker
14	Moolchand
15	Gurnam Singh
16	Savita

Q3:-

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

In [66]:

```
#Correct
execute_sql('select p.name, m.year from person p join m_cast mc on p.pid = trim(mc.pid) join movie
m on mc.mid = m.mid where (m.year  not between 1970 and 1990)')


# execute_sql('select m.mid from movie m where ( (m.year between 1970 and 1990))')
```

Out[66]:

	Name	year
0	Christian Bale	2018
1	Cate Blanchett	2018
2	John Benfield	2018
3	Lorna Brown	2018
4	Patrick Godfrey	2018
5	Naomie Harris	2018
6	Tom Hollander	2018
7	Eddie Marsan	2018
8	Peter Mullan	2018
9	Matthew Rhys	2018
10	Andy Serkis	2018
11	Andy Serkis	2018
12	Keveshan Pillay	2018
13	Moonsamy Narasigadu	2018
14	Soobrie Govender	2018
15	Gopal Singh	2018
16	Kista Munsami	2018
17	Mahomed Araf Cassim	2018
18	Roshan Jayesh Patel	2018
19	T'khai Phillips	2018
20	Sachin Soni	2018
21	Hridhay Somera	2018
22	Ethaniel Jaden Moonsamy	2018
23	Gareth Ryan Benjamin	2018
24	Nirvayesh Chakravorty Thanendra	2018
25	Adiyan Ahmed Choudhury	2018
26	Amara Motala	2018
27	Diyara Prakash	2018
28	Diyajal Prakash	2018
29	Kassius Carey-Johnson	2018
...	...	...
70743	Arun Bakshi	2006

	Name	year
70744	Milind Gunaji	2006
70745	Aushim Khetarpal	2006
70746	Bobby Darling	2006
70747	Mohini Manik	2006
70748	Sunil Pal	2006
70749	Sunil Pal	2006
70750	Kamal Maharshi	2006
70751	Hayley Cleghorn	2006
70752	Nirvasha Jithoo	2006
70753	Jaipreet Nagra	2006
70754	Ajay Kumar Verma	2006
70755	Iqbal	1939
70756	Iqbal	1939
70757	Gulshan Grover	1994
70758	Urmila Matondkar	1994
70759	Ishrat Ali	1994
70760	Prem Chopra	1994
70761	Sudhir Dalvi	1994
70762	Ajay Devgn	1994
70763	Ajay Devgn	1994
70764	Bharat Kapoor	1994
70765	Kiran Kumar	1994
70766	Reema Lagoo	1994
70767	Johnny Lever	1994
70768	Alok Nath	1994
70769	Yunus Parvez	1994
70770	Asha Sharma	1994
70771	Ajay Nagrath	1994
70772	Arun Govil	1994

70773 rows × 2 columns

Q4:-

`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 [58]:

```
#Correct
execute_sql('select p.name, count(m.title) as number_of_movies_directed from person p join m_direc
tor md on md.pid = p.pid join movie m on m.mid = md.mid  group by p.name having
(number_of_movies_directed >10) order by number_of_movies_directed desc ')
```

Out[58]:

	Name	number_of_movies_directed
0	David Dhawan	39
1	David Dhawan	39
2	Mahesh Bhatt	36
3	Mahesh Bhatt	36
4	Ram Gopal Varma	30
5	Priyadarshan	30

	Name	number_of_movies_directed
6	Ram Gopal Varma	30
7	Vikram Bhatt	29
8	Vikram Bhatt	29
9	Hrishikesh Mukherjee	27
10	Hrishikesh Mukherjee	27
11	Yash Chopra	21
12	Yash Chopra	21
13	Basu Chatterjee	19
14	Shakti Samanta	19
15	Basu Chatterjee	19
16	Shakti Samanta	19
17	Subhash Ghai	18
18	Subhash Ghai	18
19	Abbas Alibhai Burmawalla	17
20	Shyam Benegal	17
21	Abbas Alibhai Burmawalla	17
22	Rama Rao Tatineni	17
23	Shyam Benegal	17
24	Gulzar	16
25	Manmohan Desai	16
26	Raj N. Sippy	16
27	Gulzar	16
28	Manmohan Desai	16
29	Raj N. Sippy	16
...	...	...
58	Guddu Dhanoa	12
59	Madhur Bhandarkar	12
60	Nagesh Kukunoor	12
61	Prakash Jha	12
62	Prakash Mehra	12
63	Rohit Shetty	12
64	Satish Kaushik	12
65	Umesh Mehra	12
66	Anees Bazmee	12
67	Anil Sharma	12
68	Guddu Dhanoa	12
69	Madhur Bhandarkar	12
70	Nagesh Kukunoor	12
71	Prakash Jha	12
72	Prakash Mehra	12
73	Rohit Shetty	12
74	Satish Kaushik	12
75	Umesh Mehra	12
76	Govind Nihalani	11
77	Ketan Mehta	11
78	Mohit Suri	11
79	Nasir Hussain	11
80	Pramod Chakravorty	11
81	Sanjay Gupta	11
82	Govind Nihalani	11
83	Ketan Mehta	11

	Name	number_of_movies_directed
84	Mohit Suri	11
85	Nasir Hussain	11
86	Pramod Chakravorty	11
87	Sanjay Gupta	11

88 rows × 2 columns

Q5.a:-

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

In [48]:

```
execute_sql('select max(trim(m.year)) as year, mc.mid,count(distinct m.mid) as
Movie_count,count(mc.pid) F_count,count(case when p.gender="Male" then 1 end) as male_cnt from m_c
ast mc join person p on p.pid = trim(mc.pid) join movie m on m.mid = mc.mid group by mc.mid, m.yea
r having male_cnt = 0')
```

Out[48]:

	year	MID	Movie_count	F_count	male_cnt
0	1999	tt0272001	1	11	0
1	2000	tt0354922	1	11	0
2	1939	tt0375882	1	2	0
3	2018	tt8338754	1	1	0
4	I 2018	tt8458202	1	2	0

In [16]:

```
d = execute_sql('select max(trim(m.year)) as year, mc.mid,count(distinct m.mid) as
Movie_count,count(mc.pid) F_count,count(case when p.gender="Male" then 1 end) as male_cnt from m_c
ast mc join person p on p.pid = trim(mc.pid) join movie m on m.mid = mc.mid group by mc.mid, m.yea
r having male_cnt = 0')
```

In [21]:

```
e = execute_sql('select year, count(mid)  from movie  group by year')
```

In [34]:

```
tab = pd.merge(d,e ,on = 'year')
tab['per'] = (tab['Movie_count']/tab['count(mid)'])*100
tab
```

Out[34]:

	year	MID	Movie_count	F_count	male_cnt	count(mid)	per
0	1999	tt0272001	1	11	0	66	1.515152
1	2000	tt0354922	1	11	0	64	1.562500
2	1939	tt0375882	1	2	0	2	50.000000
3	2018	tt8338754	1	1	0	93	1.075269
4	I 2018	tt8458202	1	2	0	10	10.000000

In [59]:

```
execute_sql('select mm.year,a.Movie_count,a.male_cnt, count(mm.mid) as t_movies  from movie mm inn
er join (select max(trim(m.year)) as year, mc.mid,count(distinct m.mid) as
Movie_count,count(mc.pid) F_count,count(case when p.gender="Male" then 1 end) as male_cnt from m_c
```

```
Movie_count,count(mc.pid) F_count,count(case when p.gender= 'Male' then 1 end) as male_cnt from m_c
ast mc join person p on p.pid = trim(mc.pid) join movie m on m.mid = mc.mid group by mc.mid, m.yea
r having male_cnt = 0) as a group by mm.year')
```

Out[59]:

	year	Movie_count	male_cnt	t_movies
0	1931	1	0	5
1	1936	1	0	15
2	1939	1	0	10
3	1941	1	0	5
4	1943	1	0	5
5	1946	1	0	10
6	1947	1	0	10
7	1948	1	0	15
8	1949	1	0	15
9	1950	1	0	10
10	1951	1	0	30
11	1952	1	0	30
12	1953	1	0	40
13	1954	1	0	30
14	1955	1	0	45
15	1956	1	0	30
16	1957	1	0	65
17	1958	1	0	45
18	1959	1	0	30
19	1960	1	0	70
20	1961	1	0	35
21	1962	1	0	60
22	1963	1	0	50
23	1964	1	0	70
24	1965	1	0	70
25	1966	1	0	90
26	1967	1	0	95
27	1968	1	0	100
28	1969	1	0	85
29	1970	1	0	120
...	...	...	...	...
95	I 2009	1	0	45
96	I 2010	1	0	30
97	I 2011	1	0	25
98	I 2012	1	0	5
99	I 2013	1	0	35
100	I 2014	1	0	40
101	I 2015	1	0	30
102	I 2016	1	0	45
103	I 2017	1	0	25
104	I 2018	1	0	50
105	II 1983	1	0	5
106	II 1998	1	0	5
107	II 2008	1	0	5
108	II 2009	1	0	5

	year	Movie_count	male_cnt	t_movies
109	II 2010	1	0	5
110	II 2011	1	0	5
111	II 2012	1	0	5
112	II 2013	1	0	10
113	II 2017	1	0	5
114	II 2018	1	0	5
115	III 2007	1	0	10
116	III 2015	1	0	10
117	III 2016	1	0	5
118	III 2017	1	0	5
119	IV 2010	1	0	5
120	IV 2011	1	0	5
121	IV 2017	1	0	5
122	V 2015	1	0	5
123	VI 2015	1	0	5
124	XVII 2016	1	0	5

125 rows × 4 columns

In [ ]:

```
```

In [ ]:

```
```

Q5.b:-

`Report for each year the percentage of movies in that year with only female actors, and the total number of movies made that year.`

In [77]:

```
# Correct Only percentage printing problem
execute_sql('select m1.year, count(m1.mid) as t_Movie_count, m.Movie_count as f_Movie_count , ROUN
D(m.Movie_count*100/count(m1.mid),3) as Percentage from movie m1 join (select max(trim(m.year)) as
year, mc.mid,count(distinct m.mid) as Movie_count,count(mc.pid) F_count,count(case when
p.gender="Male" then 1 end) as male_cnt from m_cast mc join person p on p.pid = trim(mc.pid) join
movie m on m.mid = mc.mid group by mc.mid, m.year having male_cnt = 0) as m on m1.year = m.year gr
oup by  m1.year')
```

Out[77]:

	year	t_Movie_count	f_Movie_count	Percentage
0	1939	2	1	50.0
1	1999	66	1	1.0
2	2000	64	1	1.0
3	2018	93	1	1.0
4	I 2018	10	1	10.0

In [80]:

```
# execute_sql('select round(100/66)')
1/66
```

Out[80]:

0.015151515151515152

Q6:-

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 [97]:

```
#Correct
execute_sql('select m.title,count(distinct mc.pid) as num_actors from movie m join m_cast mc on mc
.mid = m.mid group by m.mid order by num_actors desc')
```

Out[97]:

	title	num_actors
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
10	The Avengers	138
11	Housefull 3	129
12	Fan	127
13	Split Wide Open	126
14	Bajrangi Bhaijaan	124
15	Train Station	122
16	Daddy	121
17	Million Dollar Arm	117
18	Octopussy	116
19	Dhoom:3	115
20	Miss Lovely	113
21	Jab Tak Hai Jaan	110
22	Love Aaj Kal	108
23	Mubarakan	108
24	Hey Ram	107
25	Midnight's Children	106
26	Judwaa 2	106
27	The Day the Earth Stood Still	105
28	Corporate	104
29	Oye Lucky! Lucky Oye!	104
...	...	...
3445	Silvat	4
3446	Phullu	4
3447	Haseena	4
3448	Hey Ram Hamne Gandhi Ko maar Diya	4
3449	Halkaa	4

	title	num_actors
3450	Yaadein	3
3451	Kaun?	3
3452	Rui Ka Bojh	3
3453	Mahakali Ka Insaaf	3
3454	Goopi Gawaiya Bagha Bajaiya	3
3455	Chhota Bheem and the Throne of Bali	3
3456	Man on Mission Fauladi	3
3457	Uyirile Kalanthathu	3
3458	Gauru: Journey of Courage	3
3459	Raja Aur Rangeeli	2
3460	Anjaam	2
3461	Ram Raaj	2
3462	Mumbai Delhi Mumbai	2
3463	Chaar Sahibzaade	2
3464	Man On Mission Jaanbaaz	2
3465	Motu Patlu: King of Kings	2
3466	Leera the Soulmate	2
3467	Pihu	2
3468	Kala Jigar	1
3469	Return of Hanuman	1
3470	Subah Subah	1
3471	Chaar Sahibzaade 2: Rise of Banda Singh Bahadur	1
3472	Vaibhav Sethia: Don't	1
3473	Yeh Hai Malegaon Ka Superman	0
3474	The Wish Fish	0

3475 rows × 2 columns

Q7:-

A decade is a sequence of 10 consecutive years. For example, say in your database you have movie information tarting 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 [70]:

```
#correct
execute_sql('SELECT  (CAST( (year/10) as int) *10) as decade, count(year) as Total_number_of_movie
FROM MOVIE WHERE decade!=0 group by decade order by Total_number_of_movie desc limit 1')
```

Out[70]:

	decade	Total_number_of_movie
0	2010	1018

In [126]:

```
execute_sql('select trim(year), count(mid) as number_of_movies, trim(substr(cast([year] as
varchar(4)), 1,3)) + "0s" as decade from movie group by trim(substr(cast([year] as varchar(4)), 1,
3)) order by number_of_movies desc')
```

Out[126]:

	trim(year)	number_of_movies	decade

	trim(year)	number_of_movies	decade
0	2010	1018	201
1	2006	959	200
2	1994	551	199
3	1986	342	198
4	1975	254	197
5	1968	145	196
6	I 2011	80	0
7	1958	71	195
8	I 1992	14	0
9	1946	12	194
10	II 2009	11	0
11	1939	6	193
12	III 2007	6	0
13	IV 2017	3	0
14	V 2015	1	0
15	VI 2015	1	0
16	XVII 2016	1	0

Q8:-

`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 [12]:

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

In [74]:

```
execute_sql('select pid, name from person where pid not in (select PID from M_Cast as C1 natural join Movie as \
            M1 where exists\
            (select MID from M_Cast as C2 natural join Movie as M2 \
            where C1.PID = C2.PID and (M2.year - 3) > M1.year and not exists\
            (select MID from M_Cast as C3 natural join Movie as M3 \
            where C1.PID = C3.PID and M1.year < M3.year and M3.year < M2.year)))')
```

Out[74]:

	PID	Name
0	nm0000288	Christian Bale
1	nm0000949	Cate Blanchett
2	nm1212722	Benedict Cumberbatch
3	nm0365140	Naomie Harris
4	nm0785227	Andy Serkis
5	nm0611932	Peter Mullan
6	nm2930503	Jack Reynor
7	nm0550371	Eddie Marsan
8	nm0390903	Tom Hollander
9	nm0722629	Matthew Rhys
10	nm2951768	Freida Pinto
11	nm4575116	Rohan Chand
12	nm10302065	Keveshan Pillay

	PID	Name
13	nm6162831	Louis Ashbourne Serkis
14	nm10302066	Moonsamy Narasigadu
15	nm10302067	Soobrie Govender
16	nm10302068	Gopal Singh
17	nm10302069	Kista Munsami
18	nm10302070	Mahomed Araf Cassim
19	nm5151595	Riaz Mansoor
20	nm10302071	Roshan Jayesh Patel
21	nm10302072	T'khai Phillips
22	nm10302073	Sachin Soni
23	nm10302074	Hridhay Somera
24	nm10302075	Ethaniel Jaden Moonsamy
25	nm10302076	Gareth Ryan Benjamin
26	nm10302077	Nirvayesh Chakravorty Thanendra
27	nm10302078	Adiyan Ahmed Choudhury
28	nm10302079	Amara Motala
29	nm10302080	Diyara Prakash
...	...	...
38255	nm0712994	Sandip Ray
38256	nm0007161	S.V. Krishna Reddy
38257	nm0783585	R.K. Selvamani
38258	nm2490004	Amma Rajasekhar
38259	nm1687944	Rahat Kazmi
38260	nm3262424	Rohit Gupta
38261	nm1948254	Bela Negi
38262	nm2833607	Sanjay Talreja
38263	nm1574172	Rajatesh Nayyar
38264	nm0619763	Murali Nair
38265	nm3359416	Pryas Gupta
38266	nm5676626	Shivamani
38267	nm0667393	Oliver Paulus
38268	nm2737088	Vishal Inamdar
38269	nm0787511	Kumar Shahani
38270	nm0906413	Ka-Fai Wai
38271	nm2333111	Avtandil Varsimashvili
38272	nm1421451	G. Ram Prasad
38273	nm4110102	Raja Chanda
38274	nm8558638	Deepak Ramteke
38275	nm2606304	Srinivas Sunderrajan
38276	nm2182643	Kamika Verma
38277	nm1029114	Dhorairaj Bhagavan
38278	nm3769883	Nasir Shaikh
38279	nm0007806	Abbas
38280	nm1470989	Kannan
38281	nm0298158	Adrian Fulle
38282	nm0474806	Gulshan Kumar
38283	nm0066829	Iqbal
38284	nm1421793	Sushma Shiromani

38285 rows × 2 columns

In [15]:

```
execute_sql('select * from movie natural join m_cast')
```

Out[15]:

	index	MID	title	year	rating	num_votes	PID	ID
0	0	tt2388771	Mowgli	2018	6.6	21967	nm0000288	0

In [8]:

```
execute_sql('select PID, Name from Person where PID not in\
            (select distinct(PID)from M_Cast as C1 natural join Movie as \
            M1 where exists\
            (select MID from M_Cast as C2 natural join Movie as M2 \
            where C1.PID = C2.PID and (M2.year - 3) > M1.year and not exists\
            (select MID from M_Cast as C3 natural join Movie as M3 \
            where C1.PID = C3.PID and M1.year < M3.year and M3.year < M2.year)));')
```

Out[8]:

	PID	Name
0	nm0000288	Christian Bale
1	nm0000949	Cate Blanchett
2	nm1212722	Benedict Cumberbatch
3	nm0365140	Naomie Harris
4	nm0785227	Andy Serkis
5	nm0611932	Peter Mullan
6	nm2930503	Jack Reynor
7	nm0550371	Eddie Marsan
8	nm0390903	Tom Hollander
9	nm0722629	Matthew Rhys
10	nm2951768	Freida Pinto
11	nm4575116	Rohan Chand
12	nm10302065	Keveshan Pillay
13	nm6162831	Louis Ashbourne Serkis
14	nm10302066	Moonsamy Narasigadu
15	nm10302067	Soobrie Govender
16	nm10302068	Gopal Singh
17	nm10302069	Kista Munsami
18	nm10302070	Mahomed Araf Cassim
19	nm5151595	Riaz Mansoor
20	nm10302071	Roshan Jayesh Patel
21	nm10302072	T'khai Phillips
22	nm10302073	Sachin Soni
23	nm10302074	Hridhay Somera
24	nm10302075	Ethaniel Jaden Moonsamy
25	nm10302076	Gareth Ryan Benjamin
26	nm10302077	Nirvayesh Chakravorty Thanendra
27	nm10302078	Adiyan Ahmed Choudhury
28	nm10302079	Amara Motala
29	nm10302080	Diyara Prakash
...	...	...
38255	nm0712994	Sandip Ray
38256	nm0007161	S.V. Krishna Reddy

	PID	Name
38257	nm0783583	R.K. Selvamani
38258	nm2490004	Amma Rajasekhar
38259	nm1687944	Rahat Kazmi
38260	nm3262424	Rohit Gupta
38261	nm1948254	Bela Negi
38262	nm2833607	Sanjay Talreja
38263	nm1574172	Rajatesh Nayyar
38264	nm0619763	Murali Nair
38265	nm3359416	Pryas Gupta
38266	nm5676626	Shivamani
38267	nm0667393	Oliver Paulus
38268	nm2737088	Vishal Inamdar
38269	nm0787511	Kumar Shahani
38270	nm0906413	Ka-Fai Wai
38271	nm2333111	Avtandil Varsimashvili
38272	nm1421451	G. Ram Prasad
38273	nm4110102	Raja Chanda
38274	nm8558638	Deepak Ramteke
38275	nm2606304	Srinivas Sunderrajan
38276	nm2182643	Kamika Verma
38277	nm1029114	Dhorairaj Bhagavan
38278	nm3769883	Nasir Shaikh
38279	nm0007806	Abbas
38280	nm1470989	Kannan
38281	nm0298158	Adrian Fulle
38282	nm0474806	Gulshan Kumar
38283	nm0066829	Iqbal
38284	nm1421793	Sushma Shiromani

38285 rows × 2 columns

Q9:-

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

In [11]:

```
execute_sql('select p2.pid, p2.name, count(m1.mid) as No_Movies_with_Yash from person p2  \
                join m_cast mc1 on trim(mc1.pid) = p2.pid \
                join movie m1 on m1.mid = mc1.mid \
                join m_director md1 on md1.mid = m1.mid\
                join person p3 on p3.pid = md1.pid where p3.name = "Yash Chopra" group by p2.pid ha
ving count(m1.mid) > \
                (select count(m.mid) from person  p \
                    join m_cast mc on trim(mc.pid) = p.pid \
                    join movie m on m.mid = mc.mid \
                    join m_director md on m.mid = md.mid \
                    join person p1 on trim(md.pid) = p1.pid where p2.pid = p.pid and p1.name != "Ya
sh Chopra" group by p1.pid)')
```

Out[11]:

	PID	Name	No_Movies_with_Yash
0	nm0000821	Amitabh Bachchan	6
1	nm0004434	Shashi Kapoor	14
2	nm0004435	Rajesh Khanna	3

	PID	Name	No_Movies_with_Yash
3	nm0004487	Juhi Chawla	2
4	nm0004564	Hema Malini	8
5	nm0006762	Saeed Jaffrey	2
6	nm0025630	Vikas Anand	8
7	nm0159159	Prem Chopra	3
8	nm0159165	Sudha Chopra	3
9	nm0347901	Rakhee Gulzar	5
10	nm0407002	Iftekhar	9
11	nm0438463	Anil Kapoor	3
12	nm0438465	Annu Kapoor	3
13	nm0451321	Shah Rukh Khan	4
14	nm0451600	Anupam Kher	14
15	nm0471443	Manmohan Krishna	20
16	nm0474876	Sanjeev Kumar	3
17	nm0534501	Madan Puri	8
18	nm0664109	Yunus Parvez	3
19	nm0707271	Jagdish Raj	11
20	nm0716851	Waheeda Rehman	5
21	nm0756378	Parikshat Sahni	3
22	nm0894340	Deven Verma	8
23	nm7760187	Saul George	3
24	nm9036653	Surendra Rahi	3

Q10:-

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 [7]:

```
execute_sql('select p.PID from person p join m_cast where name = " Shah Rukh Khan"')
```

Out[7]:

	PID
0	nm0451321
1	nm0451321
2	nm0451321
3	nm0451321
4	nm0451321
5	nm0451321
6	nm0451321
7	nm0451321
8	nm0451321
9	nm0451321
10	nm0451321
11	nm0451321
12	nm0451321
13	nm0451321

	PID
14	nm0451321
15	nm0451321
16	nm0451321
17	nm0451321
18	nm0451321
19	nm0451321
20	nm0451321
21	nm0451321
22	nm0451321
23	nm0451321
24	nm0451321
25	nm0451321
26	nm0451321
27	nm0451321
28	nm0451321
29	nm0451321
...	...
82807	nm0451321
82808	nm0451321
82809	nm0451321
82810	nm0451321
82811	nm0451321
82812	nm0451321
82813	nm0451321
82814	nm0451321
82815	nm0451321
82816	nm0451321
82817	nm0451321
82818	nm0451321
82819	nm0451321
82820	nm0451321
82821	nm0451321
82822	nm0451321
82823	nm0451321
82824	nm0451321
82825	nm0451321
82826	nm0451321
82827	nm0451321
82828	nm0451321
82829	nm0451321
82830	nm0451321
82831	nm0451321
82832	nm0451321
82833	nm0451321
82834	nm0451321
82835	nm0451321
82836	nm0451321

82837 rows × 1 columns