SQL Assignment

In [1]:

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

In [2]:

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

In [31]:

```
# http://www.appliedaicourse.com/lecture/11/applied-machine-learning-online-course/4142/assignment-22-sql-assignment-on-imdb-data/1/module-1-fundamentals-of-programming#comment91709
# Preprocessed the DB

cursor = conn.cursor()
# Movie table
cursor.execute('UPDATE Movie SET year = REPLACE(year, "I", "");')
cursor.execute('UPDATE Movie SET year = REPLACE(year, "V", "");')
cursor.execute('UPDATE Movie SET year = REPLACE(year, "X ", "");')
cursor.execute('UPDATE Movie SET title = TRIM(title);')
cursor.execute('UPDATE Movie SET year = TRIM(year);')
cursor.execute('UPDATE Movie SET rating = TRIM(rating);')
cursor.execute('UPDATE Movie SET num_votes = TRIM(num_votes);')

# M_Producer
cursor.execute('UPDATE M_Producer SET PID = TRIM(PID);')
cursor.execute('UPDATE M_Producer SET MID = TRIM(MID);')

# M_Director
cursor.execute('UPDATE M_Director SET PID = TRIM(PID);')
cursor.execute('UPDATE M_Director SET MID = TRIM(MID);')

# M_Cast
cursor.execute('UPDATE M_Cast SET PID = TRIM(PID);')
cursor.execute('UPDATE M_Cast SET MID = TRIM(MID);')

# M_Genre
cursor.execute('UPDATE M_Genre SET GID = TRIM(GID);')
cursor.execute('UPDATE M_Genre SET MID = TRIM(MID);')

# Genre
cursor.execute('UPDATE Genre SET GID = TRIM(GID);')
cursor.execute('UPDATE Genre SET Name = TRIM(Name);')

# Person
cursor.execute('UPDATE Person SET Name = TRIM(Name);')
cursor.execute('UPDATE Person SET PID = TRIM(PID);')
cursor.execute('UPDATE Person SET Gender = TRIM(Gender);')

#conn.commit() temporary ( un-comment it to make permanent)
```

Out[31]:

```
<sqlite3.Cursor at 0x253be8478f0>
```

In [ ]:

```

```

Sample Code

In [47]:

In [47]:

```
%%time
# Write your sql query below

query = """
        SELECT *
        FROM Movie
        WHERE Movie.rating < 3

        """

q = pd.read_sql_query(query, conn)
print(q.shape)
```

```
(85, 6)
Wall time: 4.98 ms
```

In [48]:

```
q.head()
```

Out[48]:

	index	MID	title	year	rating	num_votes
0	13	tt3726012	Mastizaade	2016	2.4	2205
1	29	tt1098327	Dragonball Evolution	2009	2.6	64493
2	60	tt7820846	Loveyatri	2018	2.9	1345
3	75	tt7431594	Race 3	2018	2.1	27282
4	111	tt2574698	Gunday	2014	2.1	56040

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 [6]:

```
%%time
# Write your sql query below

query = """
        SELECT p.Name,m.title,m.year as year
        FROM Movie AS m, Person AS p, Genre AS g, M_Director AS md, M_Genre AS mg
        WHERE md.MID = m.MID AND md.PID = p.PID AND g.GID = mg.GID AND mg.MID = m.MID
        GROUP BY g.Name
        HAVING (((m.year % 4 == 0) AND (m.year % 100 <> 0)) OR m.year % 400 == 0) AND g.Name like '%Com
edy%'

        """

q1 = pd.read_sql_query(query, conn)
print(q1.shape)
q1.head()
```

```
(21, 3)
Wall time: 49.9 ms
```

Out[6]:

	Name	title	year
0	Remo D'Souza	A Flying Jatt	2016
1	S.S. Rajamouli	Eega	2012
2	Rohit Shetty	Sunday	2008

	Name	title	year
3	Jugal Hansraj	Roadside Romeo	2008
4	Brij	Bombay 405 Miles	1980

Q2 --- List the names of all the actors who played in the movie 'Anand' (1971)

In [7]:

```
%%time
# Write your sql query below

query = """
        SELECT distinct(p.Name)
        FROM Person AS p, M_Cast AS mc, Movie AS m
        WHERE mc.PID = p.PID AND mc.MID = m.MID AND m.year = 1971 AND m.title='Anand'

        """

q2 = pd.read_sql_query(query, conn)
print(q2.shape)
q2.head()
```

```
(17, 1)
Wall time: 119 ms
```

Out[7]:

	Name
0	Amitabh Bachchan
1	Rajesh Khanna
2	Brahm Bhardwaj
3	Ramesh Deo
4	Seema Deo

Q3 --- List all the actors who acted in a film before 1970 and in a film after 1990. (That is: < 1970 and > 1990.)

In [8]:

```
%%time
# Write your sql query below

query = """
        SELECT distinct(p.Name)
        FROM Movie AS m, Person AS p, M_Producer AS mp
        WHERE mp.MID = m.MID AND mp.PID = p.PID AND m.year > 1990
        INTERSECT
        SELECT distinct(p.Name)
        FROM Movie AS m, Person AS p, M_Producer AS mp
        WHERE mp.MID = m.MID AND mp.PID = p.PID AND m.year < 1970

        """

q3 = pd.read_sql_query(query, conn)
print(q3.shape)
q3.head()
```

```
(15, 1)
Wall time: 114 ms
```

Out[8]:

	Name
0	A.A. Nadiadwala
1	B.R. Chopra
2	Dev Anand
3	G.P. Sippy
4	Hrishikesh Mukherjee

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 [20]:

```
%%time
# Write your sql query below

query = """
        SELECT distinct(p.Name), count(m.MID) as Number_of_Movies
        FROM M_Director AS md, Person AS p, Movie AS m
        WHERE md.PID = p.PID AND md.MID = m.MID
        GROUP BY md.PID
        HAVING count(m.MID) >= 10
        ORDER BY count(m.MID) DESC

        """

q4 = pd.read_sql_query(query, conn)
print(q4.shape)
```

```
(58, 2)
Wall time: 54.9 ms
```

In [21]:

```
q4.head()
```

Out[21]:

	Name	Number_of_Movies
0	David Dhawan	39
1	Mahesh Bhatt	35
2	Priyadarshan	30
3	Ram Gopal Varma	30
4	Vikram Bhatt	29

Q5.a --- For each year, count the number of movies in that year that had only female actors.

In [91]:

```
%%time
# Write your sql query below

query = """
        SELECT m.year, count(*) FROM Movie AS m, M_Cast AS mc, Person AS p
        WHERE m.MID = TRIM(mc.MID) AND p.PID = TRIM(mc.PID) AND p.Gender="Female"
        GROUP BY m.year ORDER BY m.year ASC

        """
```

```
q5a = pd.read_sql_query(query, conn)
print(q5a.shape)
```

```
(125, 2)
Wall time: 105 ms
```

In [92]:

```
q5a.head()
```

Out[92]:

	year	count(*)
0	1931	3
1	1936	19
2	1939	18
3	1941	7
4	1943	3

Q5.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 [112]:

```
%%time
# Write your sql query below

query = """
        SELECT X.year, (X.Mov*1.0/Y.Mov)*100 AS FemalePercentage, Y.Mov AS TotalMovie FROM
        (SELECT m.year, count(*) AS Mov FROM Movie AS m, M_Cast AS mc, Person AS p
        WHERE m.MID = trim(mc.MID) AND p.PID = trim(mc.PID) AND p.Gender="Female"
        GROUP BY m.year ORDER BY m.year ASC) AS X,
        (SELECT m.year, count(*) AS Mov FROM Movie AS m, M_Cast AS mc, Person AS p
        WHERE m.MID = TRIM(mc.MID) AND p.PID = TRIM(mc.PID)
        GROUP BY m.year ORDER BY m.year ASC) AS Y
        WHERE Y.year = X.year
        """

q5b = pd.read_sql_query(query, conn)
print(q5b.shape)
```

```
(125, 3)
Wall time: 253 ms
```

In [114]:

```
q5b.head()
```

Out[114]:

	year	FemalePercentage	TotalMovie
0	1931	33.333333	9
1	1936	40.425532	47
2	1939	40.000000	45
3	1941	12.962963	54
4	1943	21.428571	14

	year	FemalePercentage	TotalMovie
4	1940	21.428571	14

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 [160]:

```
%%time
# Write your sql query below

query = """
        SELECT X.* FROM
        (Select m.title AS Movie_Name, count(*) AS count_distinct_actor
        FROM Movie AS m, Person AS p, M_Cast AS mc
        WHERE m.MID = TRIM(mc.MID) AND TRIM(mc.PID) = p.PID
        GROUP BY m.title) AS X
        ORDER BY X.count_distinct_actor DESC
        LIMIT 1
        """

q6 = pd.read_sql_query(query, conn)
print(q6.shape)
```

```
(1, 2)
Wall time: 174 ms
```

In [32]:

```
q6.head()
```

Out[32]:

	Movie_Name	count_distinct_actor
0	Ocean's Eight	238

Q7 --- 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 [99]:

```
%%time
# Write your sql query below

query = """
         SELECT * FROM
         (SELECT
         CAST((m.year/10)*10 AS VARCHAR) || "-" || CAST((m.year/10)*10+9 AS VARCHAR)
         AS decade, COUNT(*) AS films
         FROM Movie as m
         GROUP BY m.year/10
         ORDER BY 1) AS X
         ORDER BY X.films DESC
         LIMIT 1
        """

q7 = pd.read_sql_query(query, conn)
print(q7.shape)
```

```
(1, 2)
Wall time: 6.98 ms
```

In [100]:

```
q7
```

Out[100]:

	decade	films
0	2010-2019	1092

Q8 --- Find all the actors that made more movies with Yash Chopra than any other director.

In [41]:

```
%%time
# Write your sql query below

query = """
         Select p.Name, Q1.pcount as moviewith_yashchopra from Person as p,
         (
             Select p1.PID as personID, count(*) as pcount from movie as m1, person as p1, m_cast as mc
1,
             (
                 SELECT distinct(m.MID) as MoviesID from movie as m, person as p, m_director as md
                 where m.MID = md.MID and p.PID = md.PID AND p.Name = "Yash Chopra"
             ) as X
             where X.MoviesID = m1.MID and p1.PID = mc1.PID and m1.MID = mc1.MID
             group by mc1.PID
         ) as Q1,
         (
             Select p2.PID as personID, count(*) as pcount from movie as m2, person as p2, m_cast as mc
2,
             (
                 SELECT distinct(m.MID) as MoviesID from movie as m, person as p, m_director as md
                 where m.MID = md.MID and p.PID = md.PID AND p.Name != "Yash Chopra"
             ) as Y
             where Y.MoviesID = m2.MID and p2.PID = mc2.PID and m2.MID = mc2.MID
             group by mc2.PID
         ) as Q2 on
         Q1.personID = Q2.personID and Q1.personID = p.PID and Q1.pcount > Q2.pcount
        """

q8 = pd.read_sql_query(query, conn)
print(q8.shape)
```

```
(3, 2)
Wall time: 599 ms
```

In [42]:

```
q8.head()
```

Out[42]:

	Name	moviewith_yashchopra
0	Yash Chopra	2
1	Ashok Verma	2
2	Nazir	2

Q9 --- 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

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]:

```
%%time
# Write your sql query below

query = """


        """

q9 = pd.read_sql_query(query, conn)
print(q9.shape)
q9.head()
```

In [0]:

```

```