HASAN ALI

CMSC 462 A-5

December 1, 2023

hali4@umbc.edu

UM81934

Libraries

In [1]:
```
#The following command resets the environment
from IPython.core.display import HTML
HTML("<script>Jupyter.notebook.kernel.restart()</script>")

# If you get error no module named pymogo
# !python -m pip install pymongo

from pymongo import MongoClient
import pymongo
import pandas as pd
import re
```

Connecting to MongoDB

In [2]:
```
try:
    client = MongoClient("mongodb://localhost:27017/")
    client.drop_database("HA_CMSC462_A5_DB")
    print("Successfully connected and deleted the Database if it already existed")
except Exception as e:
    print(e)
```

```
Successfully connected and deleted the Database if it already existed
```

In [3]:
```
try:
    #Create database
    db=client.get_database("HA_CMSC462_A5_DB")
    print("Database created successfully")
except Exception as e:
    print(e)
```

```
Database created successfully
```

In [4]:
```
#this would list an empty list, since no data in the collection
db.list_collection_names()
```

Out[4]:
```
[]
```

Reading movies.csv file

In [5]:
```
# Read movies.csv into a DataFrame
movies_df = pd.read_csv('movies.csv')

# Insert movies into MongoDB
movies_collection = db['Movies']

for index, row in movies_df.iterrows():
    movie_data = {
        'Movie ID': row['movieId'],
        'Title': row['title'],
        'Genres': row['genres']
    }
    movies_collection.insert_one(movie_data)
```

Reading ratings.csv file

In [6]:
```
# Read ratings.csv into a DataFrame
ratings_df = pd.read_csv('ratings.csv')

# Insert ratings into MongoDB
ratings_collection = db['Ratings']

for index, row in ratings_df.iterrows():
    rating_data = {
        'User ID': row['userId'],
        'Movie ID': row['movieId'],
        'Rating': row['rating'],
        'Time Stamp': row['timestamp']
    }
    ratings_collection.insert_one(rating_data)
```

Reading tags.csv (for some reason, reading and inserting tag.csv takes time on mongodb. It will eventually show up after 10 -20 seconds)

In [7]:
```
# Read tags.csv into a DataFrame
tags_df = pd.read_csv('tags.csv')

# Insert ratings into MongoDB
tags_collection = db['Tags']

for index, row in tags_df.iterrows():
    tags_data = {
        'User ID': row['userId'],
        'Movie ID': row['movieId'],
```

```
        'Tag': row['tag'],
        'Time Stamp': row['timestamp']
    }
    tags_collection.insert_one(tags_data)
```

Inserting my own movies into Movies collection

In [8]:
```
# I checked in the movies.csv and found no mention of the movies I am add in this sect
# So these movies are unique to the .csv file
my_movie_1 = {'Movie ID': '193700', 'Title': 'Kabhi Khushi Kabhie Gham (2001)',
              'Genres': 'Romance|Drama|Comedy|Family Drama'}
my_movie_2 = { "Movie ID": "193800", "Title": "Kal Ho Naa Ho (2003)", "Genres": "Romar
my_movie_3 = {"Movie ID": "193900", "Title": "Fanaa (2006)", "Genres": "Thriller|Romar
my_movie_4 = {"Movie ID": "194000", "Title": "Taare Zameen Par (2007)", "Genres": "Com
my_movie_5 = {"Movie ID": "194100", "Title": "Phir Hera Pheri (2006)","Genres": "Comed

movie_1 = movies_collection.find_one({"Movie ID": "193700"})
movie_2 = movies_collection.find_one({"Movie ID": "193800"})
movie_3 = movies_collection.find_one({"Movie ID": "193900"})
movie_4 = movies_collection.find_one({"Movie ID": "194000"})
movie_5 = movies_collection.find_one({"Movie ID": "194100"})

# Edge cases so I dont insert the same movie more than once
if movie_1:
    print("Movie already exists")
else:
    print("Movie does not exist. Adding to the Movies collection")
    movies_collection.insert_one(my_movie_1)

if movie_2:
    print("Movie already exists")
else:
    print("Movie does not exist. Adding to the Movies collection")
    movies_collection.insert_one(my_movie_2)

if movie_3:
    print("Movie already exists")
else:
    print("Movie does not exist. Adding to the Movies collection")
    movies_collection.insert_one(my_movie_3)

if movie_4:
    print("Movie already exists")
else:
    print("Movie does not exist. Adding to the Movies collection")
    movies_collection.insert_one(my_movie_4)

if movie_5:
    print("Movie already exists")
else:
    print("Movie does not exist. Adding to the Movies collection")
    movies_collection.insert_one(my_movie_5)
```

```
Movie does not exist. Adding to the Movies collection
Movie does not exist. Adding to the Movies collection
Movie does not exist. Adding to the Movies collection
Movie does not exist. Adding to the Movies collection
Movie does not exist. Adding to the Movies collection
```

In [9]:
```
# Printing the first movie I added in the movie collection to show its in the database
movie_1 = movies_collection.find_one({"Movie ID": "193700"})

print("Movie ID:", movie_1["Movie ID"])
print("Title:", movie_1["Title"])
print("Genre:", movie_1["Genres"])
```

```
Movie ID: 193700
Title: Kabhi Khushi Kabhie Gham (2001)
Genre: Romance|Drama|Comedy|Family Drama
```

Inserting my own ratings into Ratings collection

In [10]:
```
# Creating my ratings for the five movies
my_rating_1 = {'User ID': '690', 'Movie ID': '193700', 'Rating': '5.0', 'Time Stamp':
my_rating_2 = {'User ID': '690', 'Movie ID': '193800', 'Rating': '5.0', 'Time Stamp':
my_rating_3 = {'User ID': '690', 'Movie ID': '193900', 'Rating': '5.0', 'Time Stamp':
my_rating_4 = {'User ID': '690', 'Movie ID': '194000', 'Rating': '5.0', 'Time Stamp':
my_rating_5 = {'User ID': '690', 'Movie ID': '194100', 'Rating': '5.0', 'Time Stamp':

ratings_list = [my_rating_1, my_rating_2, my_rating_3, my_rating_4, my_rating_5]

for item in ratings_list:
    ratings_collection.insert_one(item)
```

In [11]:
```
# Printing the rating for the first movie I added in the rating collection to show its
rating_1 = ratings_collection.find_one({"Movie ID": "193700"})

print("UserID:", rating_1["User ID"])
print("Movie ID:", rating_1["Movie ID"])
print("Rating:", rating_1["Rating"])
print("Time Stamp:", rating_1["Time Stamp"])
```

```
UserID: 690
Movie ID: 193700
Rating: 5.0
Time Stamp: 1493855270
```

Inserting my own tags into Tags collection

In [12]:
```
# Creating my own tags for each movie. Not checking if the tag already exist in this c
my_tag_1 = {'User ID': '690', 'Movie ID': '193700', 'Tag': 'Love Story', 'Time Stamp':
my_tag_2 = {'User ID': '690', 'Movie ID': '193700', 'Tag': 'Shah Rukh Khan', 'Time Sta
my_tag_3 = {'User ID': '690', 'Movie ID': '193700', 'Tag': 'Funny', 'Time Stamp': '149

my_tag_4 = {'User ID': '690', 'Movie ID': '193800', 'Tag': 'Shah Rukh Khan', 'Time Sta
my_tag_5 = {'User ID': '690', 'Movie ID': '193800', 'Tag': 'Funny', 'Time Stamp': '149
my_tag_6 = {'User ID': '690', 'Movie ID': '193800', 'Tag': 'Love Triangle', 'Time Stam
my_tag_7 = {'User ID': '690', 'Movie ID': '193800', 'Tag': 'Crying', 'Time Stamp': '16
my_tag_8 = {'User ID': '690', 'Movie ID': '193800', 'Tag': 'Feel Good', 'Time Stamp':
my_tag_9 = {'User ID': '690', 'Movie ID': '193800', 'Tag': 'Best Movie', 'Time Stamp':

my_tag_10 = {'User ID': '690', 'Movie ID': '193900', 'Tag': 'Funny', 'Time Stamp': '16
my_tag_11 = {'User ID': '690', 'Movie ID': '193900', 'Tag': 'Sad', 'Time Stamp': '1493
my_tag_12 = {'User ID': '690', 'Movie ID': '193900', 'Tag': 'Aamir Khan', 'Time Stamp'
my_tag_13 = {'User ID': '690', 'Movie ID': '193900', 'Tag': 'Kajol', 'Time Stamp': '14
```

```
my_tag_14 = {'User ID': '690', 'Movie ID': '193900', 'Tag': 'Blind Love', 'Time Stamp'
my_tag_15 = {'User ID': '690', 'Movie ID': '193900', 'Tag': 'Crying', 'Time Stamp': '1

my_tag_16 = {'User ID': '690', 'Movie ID': '194000', 'Tag': 'Sad', 'Time Stamp': '1473
my_tag_17 = {'User ID': '690', 'Movie ID': '194000', 'Tag': 'Feel Good', 'Time Stamp':

my_tag_18 = {'User ID': '690', 'Movie ID': '194100', 'Tag': 'Hilarious', 'Time Stamp':
my_tag_19 = {'User ID': '690', 'Movie ID': '194100', 'Tag': 'Sequel', 'Time Stamp': '1
my_tag_20 = {'User ID': '690', 'Movie ID': '194100', 'Tag': 'Funny', 'Time Stamp': '14

tags_list = [my_tag_1, my_tag_2, my_tag_3, my_tag_4, my_tag_5, my_tag_6, my_tag_7, my_
             my_tag_11, my_tag_12, my_tag_13, my_tag_14, my_tag_15, my_tag_16, my_tag_

for item in tags_list:
    tags_collection.insert_one(item)
```

In [13]:
```
# Printing the tags for the first movie I added in the tags collection to show its in
tags_1 = tags_collection.find_one({"Movie ID": "193700"})

print("UserID:", tags_1["User ID"])
print("Movie ID:", tags_1["Movie ID"])
print("Tag:", tags_1["Tag"])
print("Time Stamp:", tags_1["Time Stamp"])
```

```
UserID: 690
Movie ID: 193700
Tag: Love Story
Time Stamp: 1493855270
```

Using Aggregation Pipeline

Code to find the number of movies released per year

In [14]:
```
# Aggregation pipeline to group by year and count the number of movies
pipeline = [
    {'$addFields': {'Year': {'$regexFind': {'input': "$Title", 'regex': r'\((\d{4})\)'
    {'$group': {'_id': '$Year.match', 'count': {'$sum': 1}}},
    {'$sort': {'count': -1}}
]

# Execute the aggregation pipeline
result = list(movies_collection.aggregate(pipeline))

# Display the result
for entry in result:
    print(f"Year: {entry['_id']}, Number of Movies: {entry['count']}")
```

```
Year: (2002), Number of Movies: 311
Year: (2006), Number of Movies: 297
Year: (2001), Number of Movies: 295
Year: (2007), Number of Movies: 285
Year: (2000), Number of Movies: 283
Year: (2009), Number of Movies: 282
Year: (2003), Number of Movies: 280
Year: (2004), Number of Movies: 279
Year: (2014), Number of Movies: 278
Year: (1996), Number of Movies: 276
Year: (2015), Number of Movies: 274
Year: (2005), Number of Movies: 273
Year: (2008), Number of Movies: 269
Year: (1999), Number of Movies: 263
Year: (1997), Number of Movies: 260
Year: (1995), Number of Movies: 259
Year: (1998), Number of Movies: 258
Year: (2011), Number of Movies: 254
Year: (2010), Number of Movies: 247
Year: (2013), Number of Movies: 239
Year: (1994), Number of Movies: 237
Year: (2012), Number of Movies: 233
Year: (2016), Number of Movies: 218
Year: (1993), Number of Movies: 198
Year: (1992), Number of Movies: 167
Year: (1988), Number of Movies: 165
Year: (1987), Number of Movies: 153
Year: (1991), Number of Movies: 147
Year: (1990), Number of Movies: 147
Year: (2017), Number of Movies: 147
Year: (1989), Number of Movies: 142
Year: (1986), Number of Movies: 139
Year: (1985), Number of Movies: 126
Year: (1984), Number of Movies: 101
Year: (1981), Number of Movies: 92
Year: (1980), Number of Movies: 89
Year: (1982), Number of Movies: 87
Year: (1983), Number of Movies: 83
Year: (1979), Number of Movies: 69
Year: (1977), Number of Movies: 63
Year: (1978), Number of Movies: 59
Year: (1973), Number of Movies: 59
Year: (1971), Number of Movies: 47
Year: (1965), Number of Movies: 47
Year: (1974), Number of Movies: 45
Year: (1976), Number of Movies: 44
Year: (1964), Number of Movies: 43
Year: (1968), Number of Movies: 42
Year: (1967), Number of Movies: 42
Year: (1966), Number of Movies: 42
Year: (1975), Number of Movies: 42
Year: (2018), Number of Movies: 41
Year: (1962), Number of Movies: 40
Year: (1972), Number of Movies: 39
Year: (1963), Number of Movies: 39
Year: (1960), Number of Movies: 37
Year: (1959), Number of Movies: 37
Year: (1955), Number of Movies: 36
Year: (1969), Number of Movies: 35
Year: (1961), Number of Movies: 34
```

```
Year: (1970), Number of Movies: 33
Year: (1957), Number of Movies: 33
Year: (1958), Number of Movies: 31
Year: (1953), Number of Movies: 30
Year: (1956), Number of Movies: 30
Year: (1949), Number of Movies: 25
Year: (1940), Number of Movies: 25
Year: (1946), Number of Movies: 23
Year: (1954), Number of Movies: 23
Year: (1939), Number of Movies: 23
Year: (1942), Number of Movies: 23
Year: (1951), Number of Movies: 22
Year: (1950), Number of Movies: 21
Year: (1948), Number of Movies: 20
Year: (1947), Number of Movies: 20
Year: (1936), Number of Movies: 18
Year: (1941), Number of Movies: 18
Year: (1945), Number of Movies: 17
Year: (1944), Number of Movies: 16
Year: (1937), Number of Movies: 16
Year: (1952), Number of Movies: 16
Year: (1938), Number of Movies: 15
Year: (1931), Number of Movies: 14
Year: (1935), Number of Movies: 13
Year: None, Number of Movies: 13
Year: (1933), Number of Movies: 12
Year: (1934), Number of Movies: 11
Year: (1943), Number of Movies: 10
Year: (1932), Number of Movies: 9
Year: (1927), Number of Movies: 7
Year: (1924), Number of Movies: 5
Year: (1930), Number of Movies: 5
Year: (1926), Number of Movies: 5
Year: (1925), Number of Movies: 4
Year: (1916), Number of Movies: 4
Year: (1923), Number of Movies: 4
Year: (1929), Number of Movies: 4
Year: (1928), Number of Movies: 4
Year: (1920), Number of Movies: 2
Year: (1908), Number of Movies: 1
Year: (1917), Number of Movies: 1
Year: (1903), Number of Movies: 1
Year: (1922), Number of Movies: 1
Year: (1921), Number of Movies: 1
Year: (1902), Number of Movies: 1
Year: (1919), Number of Movies: 1
Year: (1915), Number of Movies: 1
```

Code to find the number of movies per genre

In [15]:

```
# Aggregation pipeline to split genres, unwind, and then group by genre
pipeline = [
    {'$project': {'Genres': {'$split': ["$Genres", "|"]}}},
    {'$unwind': '$Genres'},
    {'$group': {'_id': '$Genres', 'count': {'$sum': 1}}},
    {'$sort': {'count': -1}}
]
```

```
# Execute the aggregation pipeline
result = list(movies_collection.aggregate(pipeline))

# Display the result
for entry in result:
    print(f"Genre: {entry['_id']}, Number of Movies: {entry['count']}")
```

```
Genre: Drama, Number of Movies: 4363
Genre: Comedy, Number of Movies: 3760
Genre: Thriller, Number of Movies: 1896
Genre: Action, Number of Movies: 1828
Genre: Romance, Number of Movies: 1599
Genre: Adventure, Number of Movies: 1263
Genre: Crime, Number of Movies: 1199
Genre: Sci-Fi, Number of Movies: 980
Genre: Horror, Number of Movies: 978
Genre: Fantasy, Number of Movies: 779
Genre: Children, Number of Movies: 664
Genre: Animation, Number of Movies: 611
Genre: Mystery, Number of Movies: 573
Genre: Documentary, Number of Movies: 440
Genre: War, Number of Movies: 382
Genre: Musical, Number of Movies: 334
Genre: Western, Number of Movies: 167
Genre: IMAX, Number of Movies: 158
Genre: Film-Noir, Number of Movies: 87
Genre: (no genres listed), Number of Movies: 34
Genre: Crime Fiction, Number of Movies: 2
Genre: Family Drama, Number of Movies: 1
Genre: Tragedy, Number of Movies: 1
Genre: Melodrama, Number of Movies: 1
Genre: Children's Film, Number of Movies: 1
```

Code to find the number of movies per rating

In [16]:

```
# Aggregation pipeline to group by rating and count the number of unique movies
pipeline = [
    {'$group': {'_id': '$Rating', 'uniqueMovies': {'$addToSet': '$Movie ID'}}},
    {'$project': {'rating': '$_id', 'numberOfMovies': {'$size': '$uniqueMovies'}}},
    {'$sort': {'rating': -1}}
]

# Execute the aggregation pipeline
result = list(ratings_collection.aggregate(pipeline))

# Display the result
for entry in result:
    print(f"Rating: {entry['rating']}, Number of Movies: {entry['numberOfMovies']}")
```

```
Rating: 5.0, Number of Movies: 5
Rating: 5.0, Number of Movies: 2954
Rating: 4.5, Number of Movies: 2710
Rating: 4.0, Number of Movies: 5109
Rating: 3.5, Number of Movies: 4216
Rating: 3.0, Number of Movies: 4986
Rating: 2.5, Number of Movies: 2925
Rating: 2.0, Number of Movies: 3339
Rating: 1.5, Number of Movies: 1386
Rating: 1.0, Number of Movies: 1726
Rating: 0.5, Number of Movies: 1066
```

Code to find the number of movies tagged

In [17]:
```python
# Aggregation pipeline to count the number of unique movies with tags
pipeline = [
    {'$group': {'_id': '$Movie ID'}},
    {'$count': 'numberOfMovies'}
]

# Execute the aggregation pipeline
result = list(tags_collection.aggregate(pipeline))

# The result will be a list with a single document
if result:
    print(f"Number of Movies Tagged: {result[0]['numberOfMovies']}")
else:
    print("No tagged movies found.")
```

```
Number of Movies Tagged: 1577
```

Code to find the most popular tag

In [19]:
```python
# Aggregation pipeline to find the most popular tag
pipeline = [
    {'$group': {'_id': '$Tag', 'count': {'$sum': 1}}},
    {'$sort': {'count': -1}},
    {'$limit': 1}
]

# Execute the aggregation pipeline
result = list(tags_collection.aggregate(pipeline))

# Display the most popular tag
if result:
    print(f"Most Popular Tag: {result[0]['_id']}, Count: {result[0]['count']}")
else:
    print("No tags found.")
```

```
Most Popular Tag: In Netflix queue, Count: 131
```