Homework Assignment 03

DS 4300 - Spring 2025

- **EC Due Date**: Feb 16, 2025 @ 11:59pm
- **Regular Due Date**: Feb 18, 2025 @ 11:59pm
- Upload to GradeScope (no question/solutions to Match)

In [87]:
```
# Set up your connection to Mongo DB here.

import pymongo
from bson.json_util import dumps

uri = "mongodb://localhost:27017/"
client = pymongo.MongoClient(uri)
mflixdb = client.mflix

# questions
# - 2: is it ok that it is named _id
# - 6: do we have to return 0 for years when police is not found
# - 7: is it average number of votes per movie for each year, is it ok that it is named _id
```

Directions:

- Use the mflix sample database to prepare a pymongo query each of the following prompts.
- Be sure to print the results of your query using the `dumps` function.

Question 1:

Give the street, city, and zipcode of all theaters in Massachusetts.

In [5]:
```
data = mflixdb.theaters.find({"location.address.state": "MA"}, {"_id": 0,
                                                                "location.address.street1": 1,
                                                                "location.address.city": 1,
                                                                "location.address.zipcode": 1,})

print(dumps(data, indent=2))
```

...

Question 2:

How many theaters are there in each state? Order the output in alphabetical order by 2-character state code.

In [73]:
```
data = mflixdb.theaters.aggregate([{"$group": {"_id": "$location.address.state", "count": {"$sum":1}}}
                                   {"$sort": {"_id":1}}])

print(dumps(data, indent=2))
```

...

Question 3:

How many movies are in the Comedy genre?

In [24]:
```
numComedies = mflixdb.movies.count_documents({ "genres": "Comedy"})
print(f'There are {numComedies} movies in the Comedy genre')
```

```
There are 6532 movies in the Comedy genre
```

Question 4:

What movie has the longest run time? Give the movie's title and genre(s).

In [31]:
```
data = mflixdb.movies.find({ }, {"_id": 0, "title": 1, "genres": 1}).sort("runtime", -1).limit(1)
print(dumps(data, indent=2))
```

```
[
  {
    "genres": [
      "Action",
      "Adventure",
      "Drama"
    ],
    "title": "Centennial"
  }
]
```

Question 5:

Which movies released after 2010 have a Rotten Tomatoes viewer rating of 3 or higher? Give the title of the movies along with their Rotten Tomatoes viewer rating score. The viewer rating score should become a top-level attribute of the returned documents. Return the matching movies in descending order by viewer rating.

In [41]:
```
data = mflixdb.movies.find({"$and": [{"year": {"$gt":2010}}, {"tomatoes.viewer.rating": {"$gte":3}}]}
                           {"_id": 0, "title": 1, "rating": "$tomatoes.viewer.rating"}).sort("tomatoe
print(dumps(data, indent=2))
```

...

Question 6:

How many movies released each year have a plot that contains some type of police activity (i.e., plot contains the word "police")? The returned data should be in ascending order by year.

In [81]:
```
data = mflixdb.movies.aggregate([{"$match": {"plot": {"$regex": r"police", "$options": "i"}}},
                                 {"$group": {"_id": "$year", "count": {"$sum":1}}},
                                 {"$sort": {"_id": 1}}])

print(dumps(data, indent=2))
```

...

Question 7:

What is the average number of imdb votes per year for movies released between 1970 and 2000 (inclusive)? Make sure the results are order by year.

In [89]:
```
data = mflixdb.movies.aggregate([{"$match": {"$and": [{"year": {"$gte":1970}}, {"year": {"$lte":2000}
                                 {"$group": {"_id": "$year", "average_votes": {"$avg": "$imdb.votes"}}
                                 {"$sort": {"_id": 1}}])

print(dumps(data, indent=2))
```


```
[
  {
    "_id": 1970,
    "average_votes": 4786.925
  },
  {
    "_id": 1971,
    "average_votes": 8528.462264150943
  },
  {
    "_id": 1972,
    "average_votes": 13582.685950413223
  },
  {
    "_id": 1973,
    "average_votes": 14478.785714285714
  },
  {
    "_id": 1974,
```


Question 8:

What distinct movie languages are represented in the database? You only need to provide the list of languages.

In [51]:
```
data = mflixdb.movies.distinct("languages")
print(dumps(data, indent=2))
```

```
[
  " Ancient (to 1453)",
  " Old",
  "Abkhazian",
  "Aboriginal",
  "Acholi",
  "Afrikaans",
  "Aidoukrou",
  "Albanian",
  "Algonquin",
  "American Sign Language",
  "Amharic",
  "Apache languages",
  "Arabic",
  "Aramaic",
  "Arapaho",
  "Armenian",
  "Assamese",
  "Assyrian Neo-Aramaic",
```