```
# Set up your connection to Mongo DB
import pymongo
from bson.json_util import dumps
import pprint

uri = "mongodb://[username]:[password]@localhost:27017"
client = pymongo.MongoClient(uri)
mflixdb = client.mflix
```

Directions:

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

Question 1:

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

[ ]:

```
match = {"$match": {"location.address.state": "MA" }}
project = {"$project": {"_id": 0,
                "Street": "$location.address.street1",
                "City": "$location.address.city",
                "Zip": "$location.address.zipcode"}}



agg = mflixdb.theaters.aggregate([match, project])
print(dumps(agg, indent=2))
```

Question 2:

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

[ ]:

```
group = {"$group": {"_id": "$location.address.state", "TotalTheaters": {"$sum": 1}}}
project = {"$project": {"_id": 0, "State": "$_id", "TotalTheaters": 1}}
sort = {"$sort": {"State": 1}}
```

```
agg = mflixdb.theaters.aggregate([group, project, sort])
print(dumps(agg, indent=2))
```

Question 3:

How many movies are in the Comedy genre?

[ ]:

```
match = {"$match": {"genres": "Comedy"}}
count = {"$count": "ComedyMovies"}

agg = mflixdb.movies.aggregate([match, count])
print(dumps(agg, indent=2))
```

Question 4:

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

[ ]:

```
sort = {"$sort": {"runtime": -1}}
project = {"$project": {"_id": 0, "title": 1, "genres": 1}}
limit = {"$limit": 1}

agg = mflixdb.movies.aggregate([sort, project, limit])
print(dumps(agg, indent=2))
```

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.

```
match = {"$match": {"year": {"$gt": 2010}, "tomatoes.viewer.rating": {"$gte": 3}}}
project = {"$project": {"_id": 0,"title": 1, "RottenTomatoesViewerRating": "$tomatoes.viewer.rating"}}
sort = {"$sort": {"RottenTomatoesViewerRating": -1}}

agg = mflixdb.movies.aggregate([match, project, sort])
print(dumps(agg, 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.

```
match = {"$match": {"plot": {"$regex": "police", "$options": "i"} }}
group = {"$group": {"_id": "$year", "TotalPoliceMovies": {"$sum": 1}}}
project = {"$project": {"_id": 0, "Year": "$_id", "TotalPoliceMovies": 1}}
sort = {"$sort": {"Year": 1}}

agg = mflixdb.movies.aggregate([match, group, project, sort])
print(dumps(agg, 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.

```
match = {"$match": {"year": {"$gte": 1970, "$lte": 2000} }}
group = {"$group": {"_id": "$year", "AvgVotes": {"$avg": "$imdb.votes"}}}
project = {"$project": {"_id": 0, "Year": "$_id", "AvgVotes": 1}}
sort = {"$sort": {"Year": 1}}

agg = mflixdb.movies.aggregate([match, group, project, sort])
print(dumps(agg, indent=2))
```

Question 8:

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

```
languages = mflixdb.movies.distinct("languages")
languages
```