

Unstructured Data Management

CIS8045: In-Class Exercise 02

```
# Query-1:
var pipeline = [
{ $match:{
  released: {$lte: new ISODate("2001-01-01T00:00:00Z")},
  runtime: {$lte: 218},
  "imdb.rating": {$gte: 7.0}
}},
{$sort : {"imdb.rating": -1}},
{$group: {
  _id: {"$arrayElemAt": ["$genres", 0]},
  "recommended_title": {$first: "$title"},
  "recommended_rating": {$first: "$imdb.rating"},
  "recommended_runtime": {$first: "$runtime"},
  "popularity": {$avg: "$imdb.rating"},
  "top_movie": {$max: "$imdb.rating"},
  "longest_runtime": {$max: "$runtime"}}},
{$sort: {popularity:-1}},
{$project: {_id: 1,
popularity: 1,
top_movie: 1,
recommended_title:1,
recommended_rating: 1,
recommended_runtime: 1,
recommended_tot_runtime: {$add: ["$recommended_runtime", 12]}
}}
];
```

```
db.movies.aggregate(pipeline)
```

The screenshot shows the Robo 3T - 1.4 interface. On the left, a sidebar lists various sample databases. The main window displays a MongoDB aggregation pipeline in a code editor. Below the code editor, the results of the aggregation are shown in a table format.

Aggregation Pipeline Code:

```
var pipeline = [
{ $match:{
  released: {$lte: new ISODate("2001-01-01T00:00:00Z")},
  runtime: {$lte: 218},
  "imdb.rating": {$gte: 7.0}
}},
{$sort : {"imdb.rating": -1}},
{$group: {
  _id: {"$arrayElemAt": ["$genres", 0]},
  "recommended_title": {$first: "$title"},
  "recommended_rating": {$first: "$imdb.rating"},
  "recommended_runtime": {$first: "$runtime"},
  "popularity": {$avg: "$imdb.rating"},
  "top_movie": {$max: "$imdb.rating"},
  "longest_runtime": {$max: "$runtime"}}},
{$sort: {popularity:-1}},
{$project: {_id: 1,
popularity: 1,
top_movie: 1,
recommended_title:1,
recommended_rating: 1,
recommended_runtime: 1,
recommended_tot_runtime: {$add: ["$recommended_runtime", 12]}
}}
];

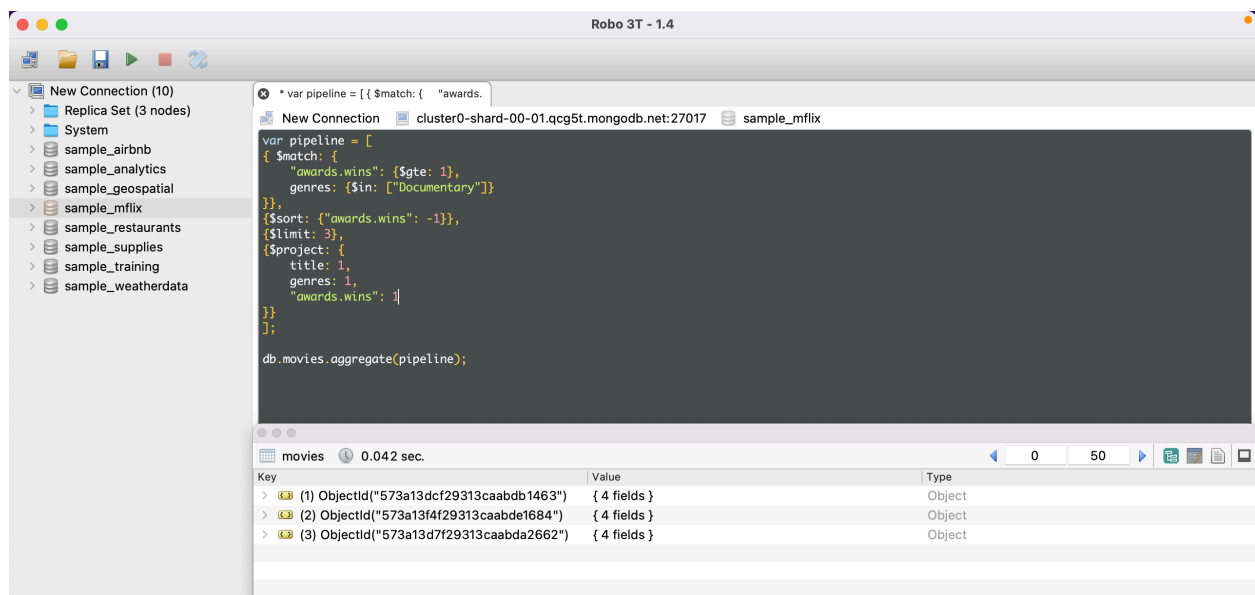
db.movies.aggregate(pipeline)
```

Results Table:

Key	Value	Type
(1) Film-Noir	{ 7 fields }	Object
(2) Documentary	{ 7 fields }	Object
(3) History	{ 7 fields }	Object
(4) Animation	{ 7 fields }	Object
(5) Crime	{ 7 fields }	Object
(6) Short	{ 7 fields }	Object
(7) Drama	{ 7 fields }	Object
(8) Western	{ 7 fields }	Object

```
# Query-2:
var pipeline = [
  { $match: {
    "awards.wins": { $gte: 1 },
    genres: { $in: ["Documentary"] }
  } },
  { $sort: { "awards.wins": -1 } },
  { $limit: 3 },
  { $project: {
    title: 1,
    genres: 1,
    "awards.wins": 1
  } }
];

db.movies.aggregate(pipeline);
```



The screenshot shows the Robo 3T - 1.4 application window. On the left, a sidebar lists various sample databases, with 'sample_movies' selected. The main area displays a MongoDB aggregation pipeline query in a dark-themed editor. Below the editor, a results pane shows the output of the query, which is a table with three rows of document objects.

```
* var pipeline = [ { $match: { "awards.wins": { $gte: 1 }, genres: { $in: ["Documentary"] } } }, { $sort: { "awards.wins": -1 } }, { $limit: 3 }, { $project: { title: 1, genres: 1, "awards.wins": 1 } } ];
```

```
db.movies.aggregate(pipeline);
```

Key	Value	Type
> (1) ObjectId("573a13dcf29313caabdb1463")	{ 4 fields }	Object
> (2) ObjectId("573a13f4f29313caabde1684")	{ 4 fields }	Object
> (3) ObjectId("573a13d7f29313caabda2662")	{ 4 fields }	Object