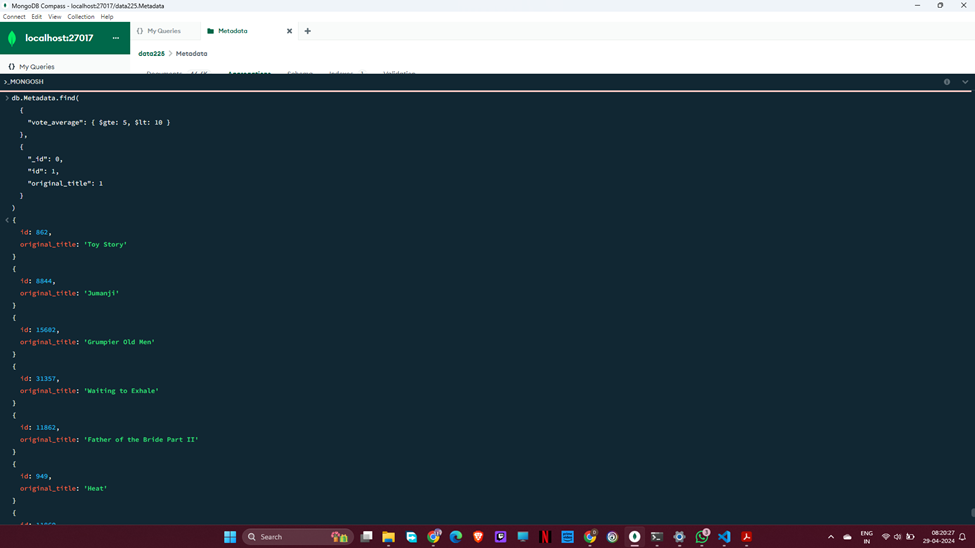
**Lab 2 Queries**

NoSQL Queries:

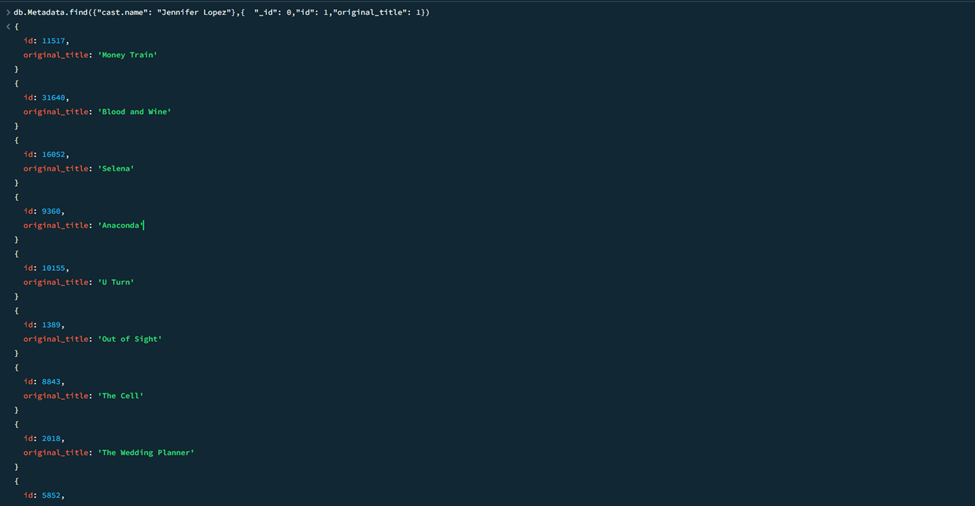
1. **Finding Vote Average between 5 and 10**

| db.Metadata.find(  {"vote\_average": { $gte: 5, $lt: 10 }},  {"\_id": 0,"id": 1,"original\_title": 1}) |
| --- |



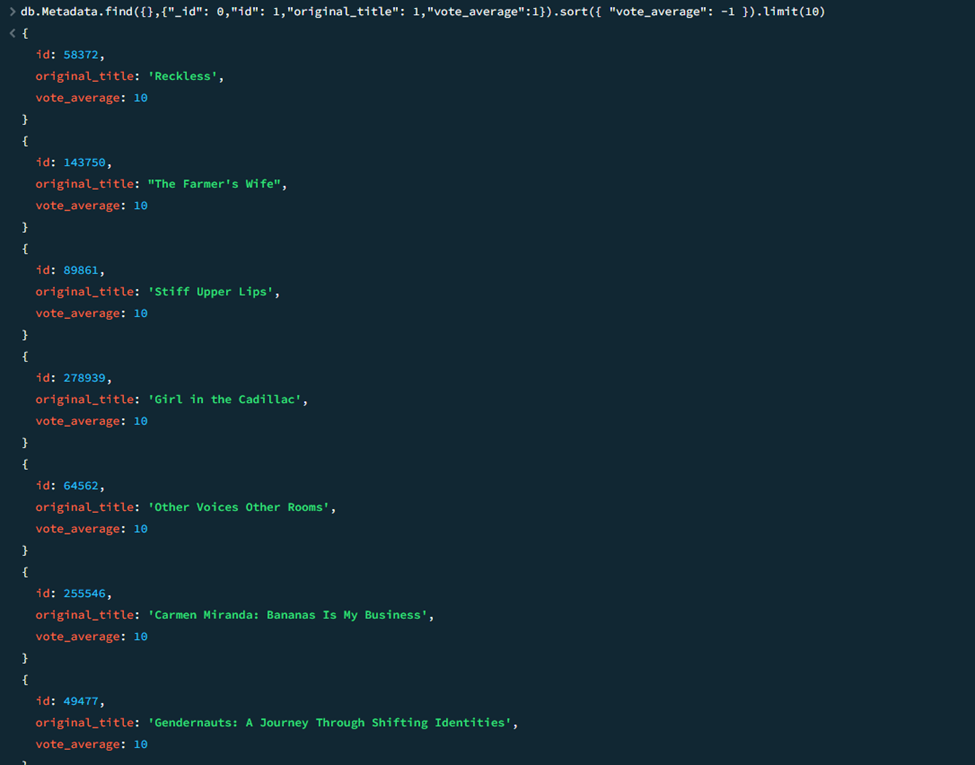
**2. Querying for movies made by Jennifer Lopez :**

| db.Metadata.find({"cast.name": "Jennifer Lopez"},{"\_id": 0,"id": 1,"original\_title": 1}) |
| --- |



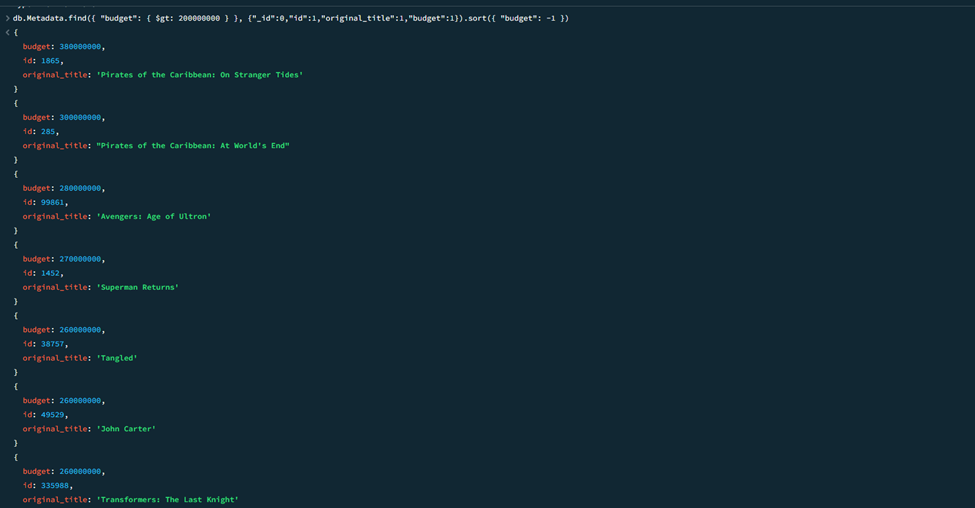
**3. Finding the Movies which has highest Vote Average :**

| db.Metadata.find({},{"\_id": 0,"id": 1,"original\_title": 1,"vote\_average":1}).sort({ "vote\_average": -1 }).limit(10) |
| --- |



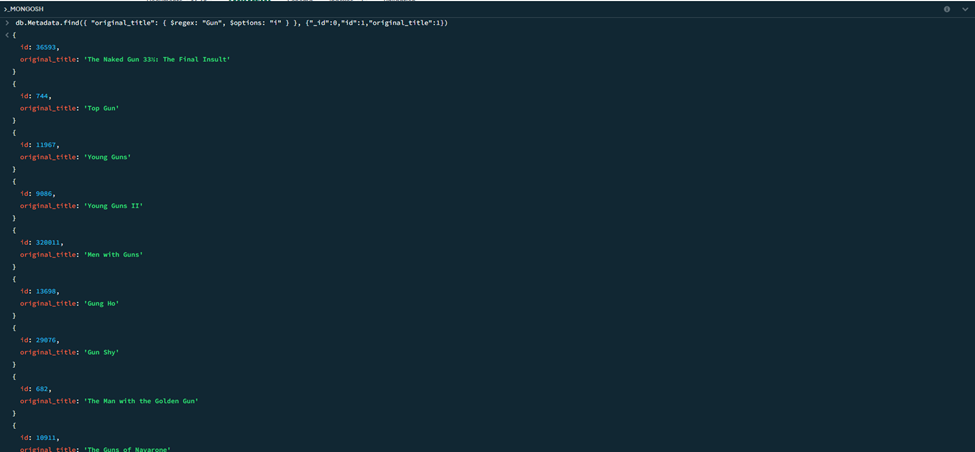
**4. Displaying movies with a budget greater than 2,000,000,000.**

| db.Metadata.find({ "budget": { $gt: 200000000 } }, {"\_id":0,"id":1,"original\_title":1,"budget":1}).sort({ "budget": -1 }) |
| --- |



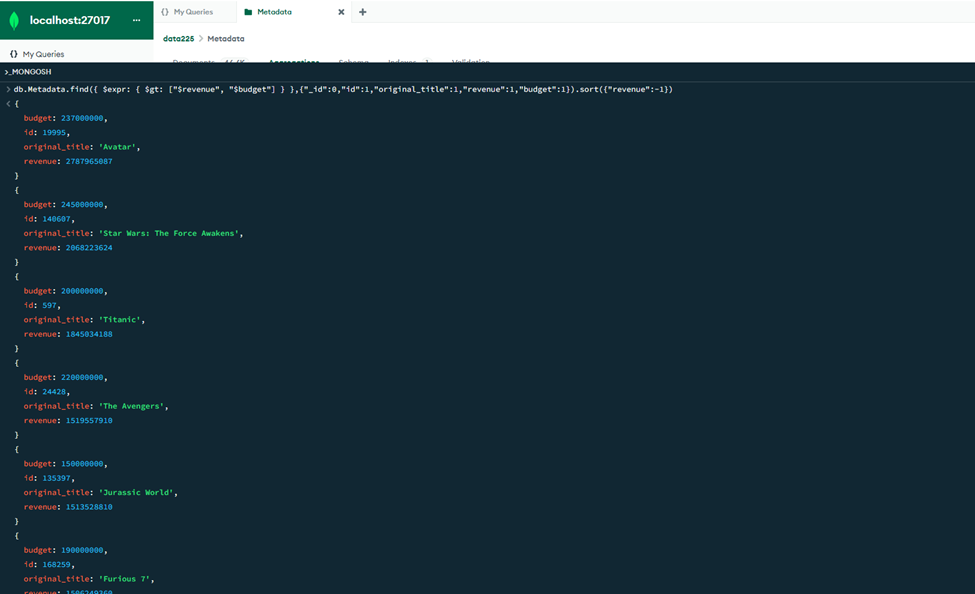
**5. Movie names with Gun in its title :**

| db.Metadata.find({ "original\_title": { $reg ex: "Gun", $options: "i" } }, {"\_id":0,"id":1,"original\_title":1}) |
| --- |



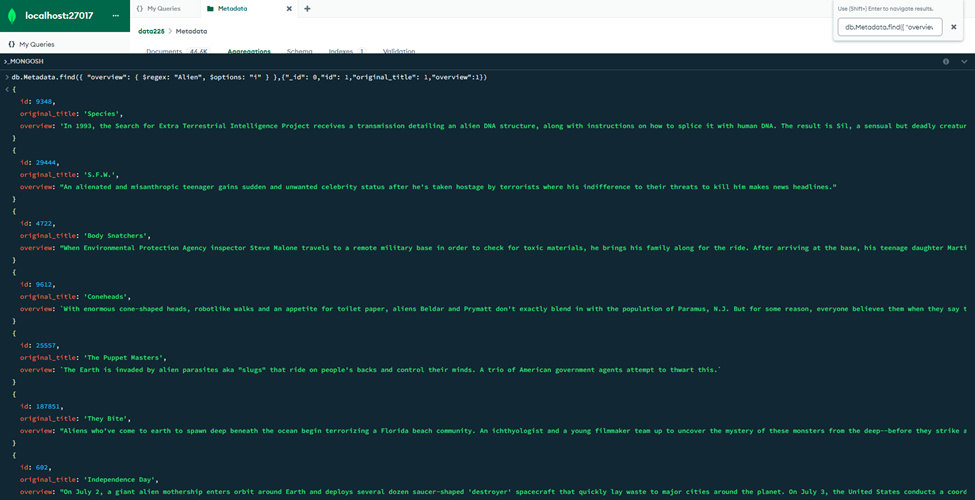
**6. Displaying movies which has the highest revenue**

| db.Metadata.find({ $expr: { $gt: ["$revenue", "$budget"] } },{"\_id":0,"id":1,"original\_title":1,"revenue":1,"budget":1}).sort({"revenue":-1} |
| --- |



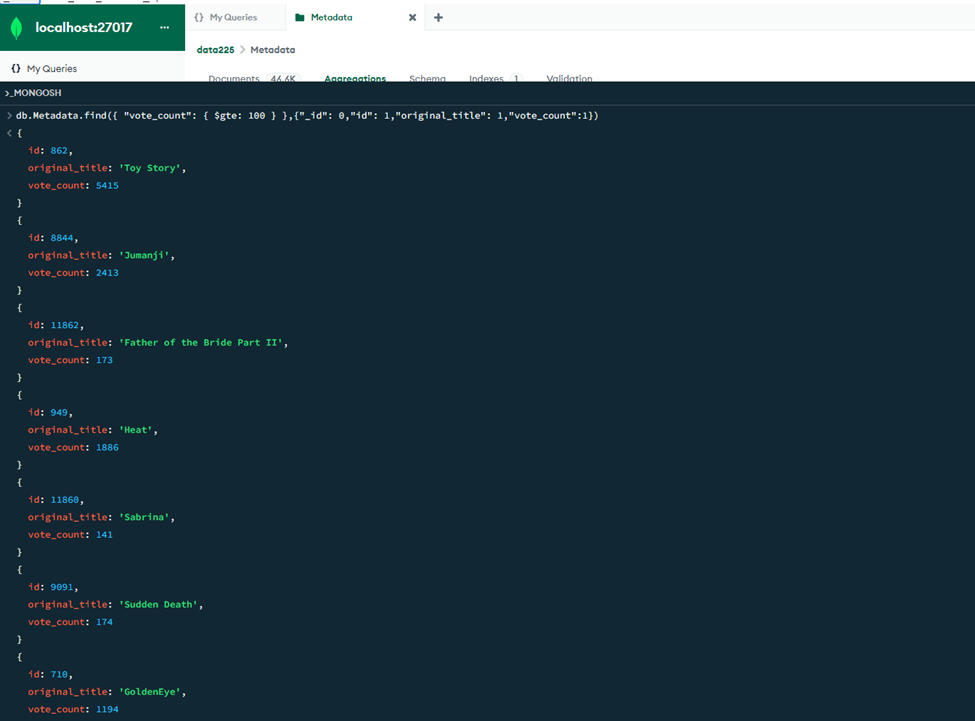
**7. Movies with Alien as the Keyword in the Overview :**

| **db.Metadata.find({ "overview": { $regex: "Alien", $options: "i" } },{"\_id": 0,"id": 1,"original\_title": 1})** |
| --- |

****

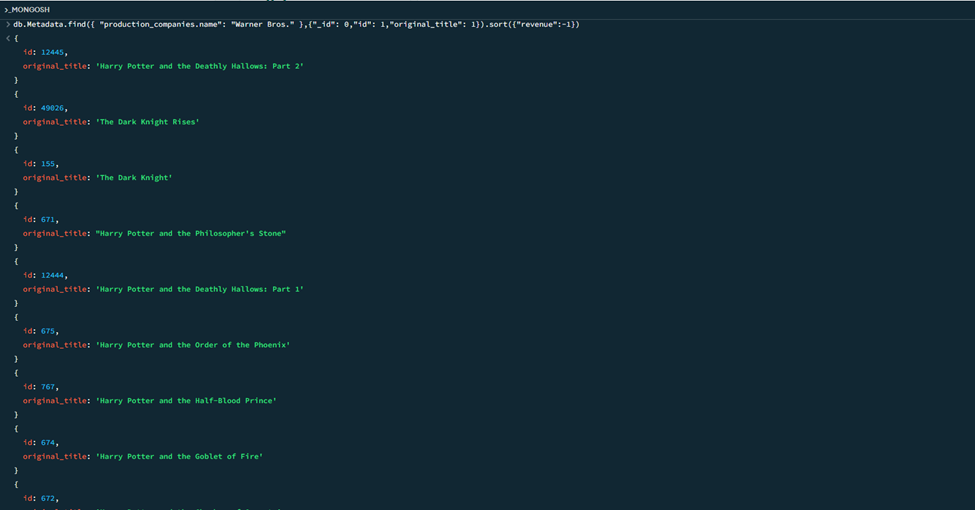
**8. Movies with Highest vote count in descending order :**

| db.Metadata.find({ "vote\_count": { $gte: 100 } },{"\_id": 0,"id": 1,"original\_titl=e": 1,"vote\_count":1}) |
| --- |



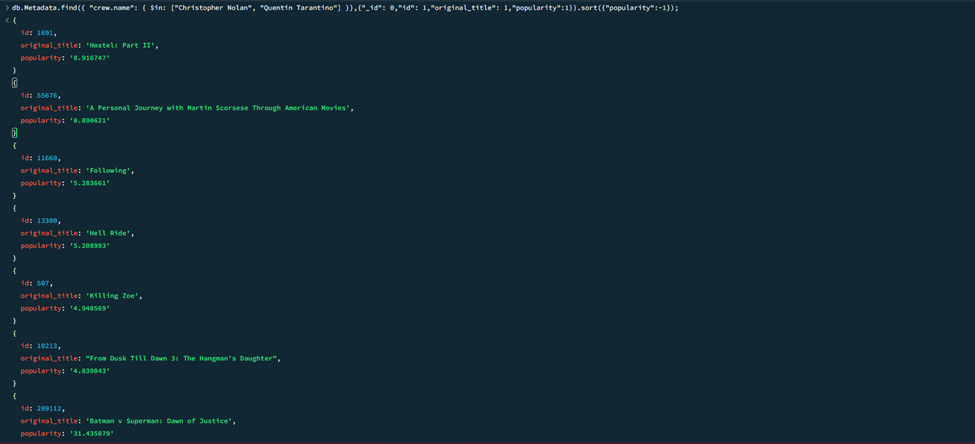
**9. Movies which were a hit produced by Warner Brothers with revenue :**

| db.Metadata.find({ "production\_companies.name": "Warner Bros." },{"\_id": 0,"id": 1,"original\_title": 1}).sort({"revenue":-1}) |
| --- |



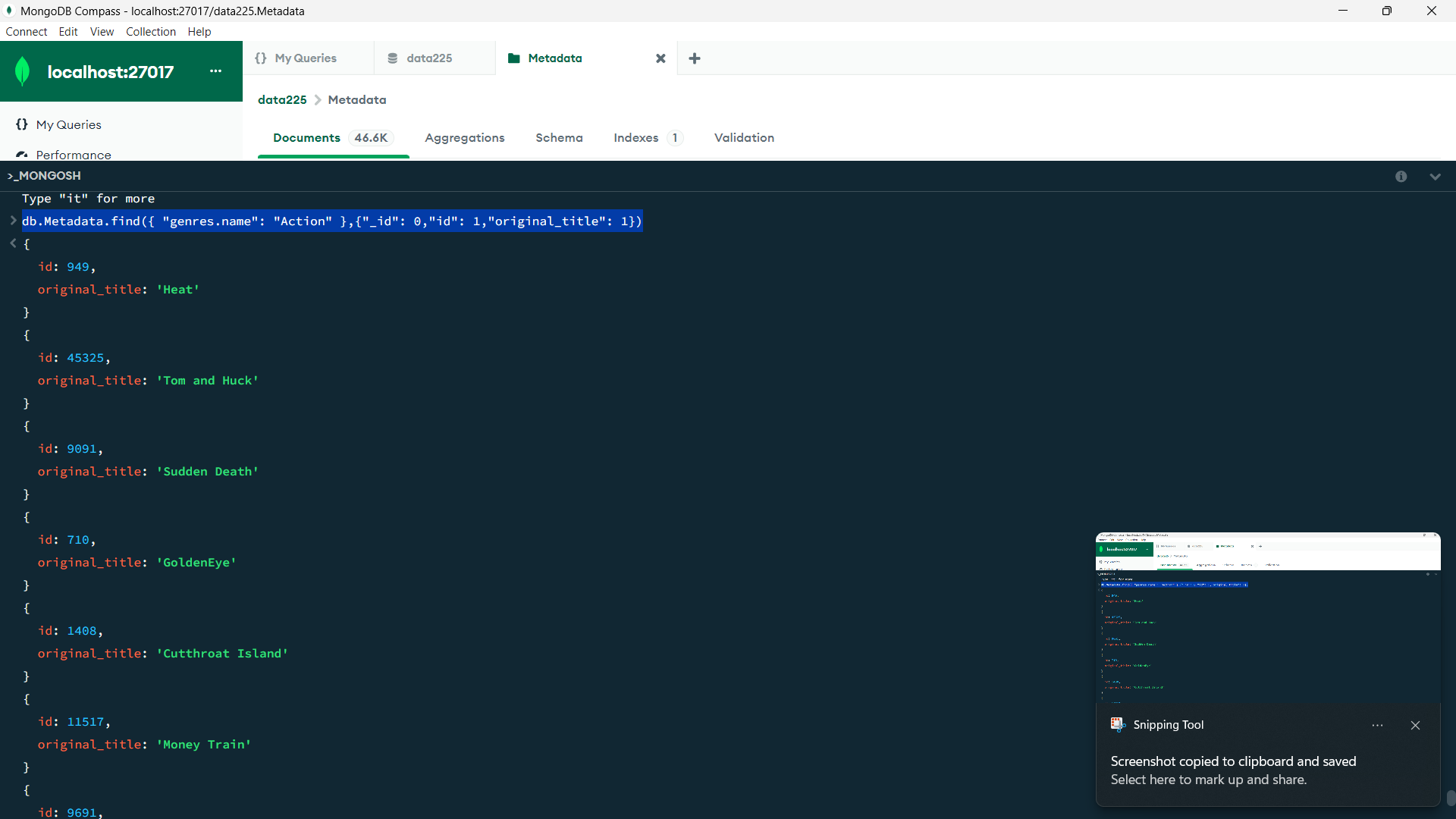
**10. Movies directed by Christopher Nolan and Quentin Tarantino with highest Popularity :**

| db.Metadata.find({ "crew.name": { $in: ["Christopher Nolan", "Quentin Tarantino"] }},{"\_id": 0,"id": 1,"original\_title": 1,"popularity":1}).sort({"popularity":-1}); |
| --- |



**11. Finding action Movies in the Movies Dataset :**

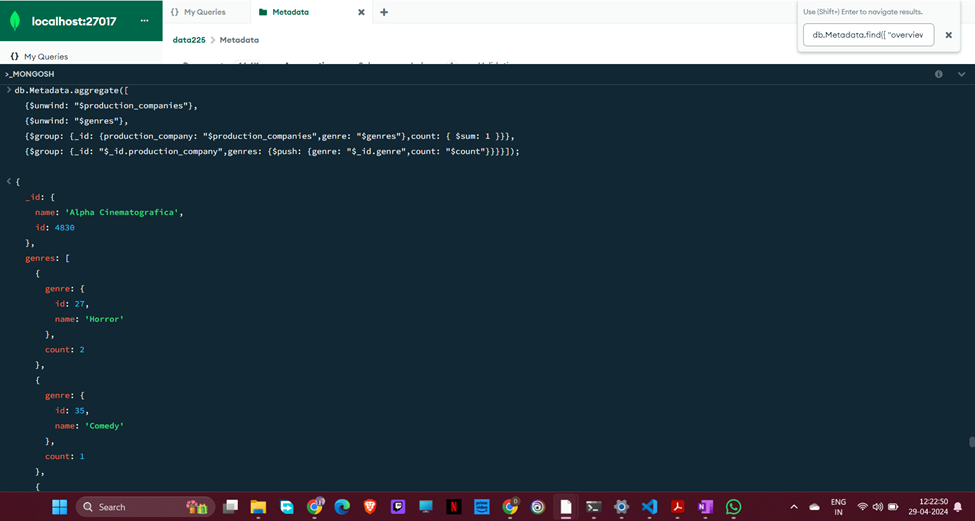
| db.Metadata.find({ "genres.name": "Action" },{"\_id": 0,"id": 1,"original\_title": 1}) |
| --- |



Aggregations :

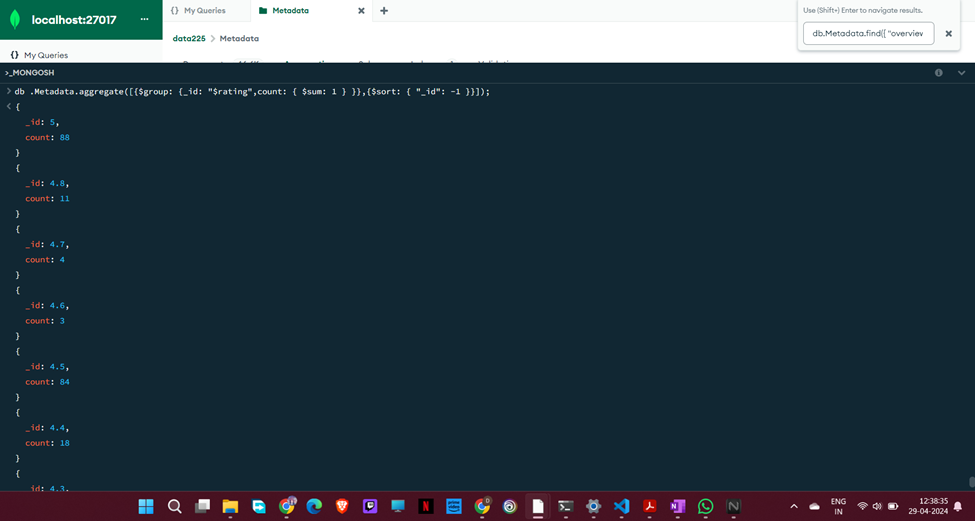
**1. Calculate the number of genres represented in movies produced by each production company:**

| db.Metadata.aggregate([  {$unwind: "$production\_companies"},  {$unwind: "$genres"},  {$group: {\_id: {production\_company: "$production\_companies",genre: "$genres"},count: { $sum: 1 }}},  {$group: {\_id: "$\_id.production\_company",genres: {$push: {genre: "$\_id.genre",count: "$count"}}}}]); |
| --- |



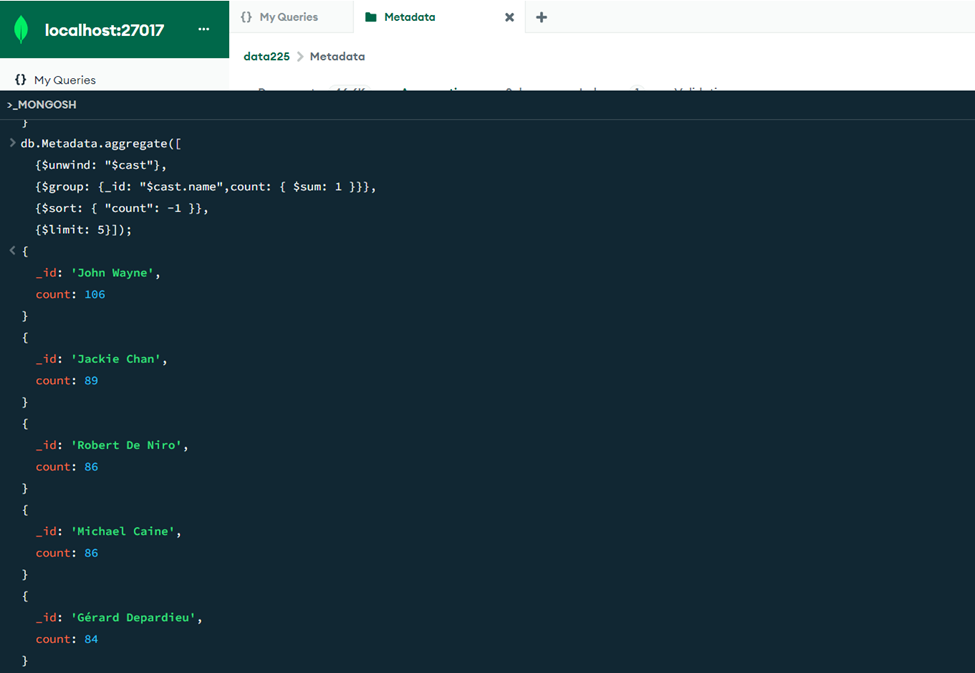
1. **Aggregation of Movie Ratings and Counting by Rating:**

| db .Metadata.aggregate([{$group: {\_id: "$rating",count: { $sum: 1 } }},{$sort: { "\_id": -1 }}]); |
| --- |



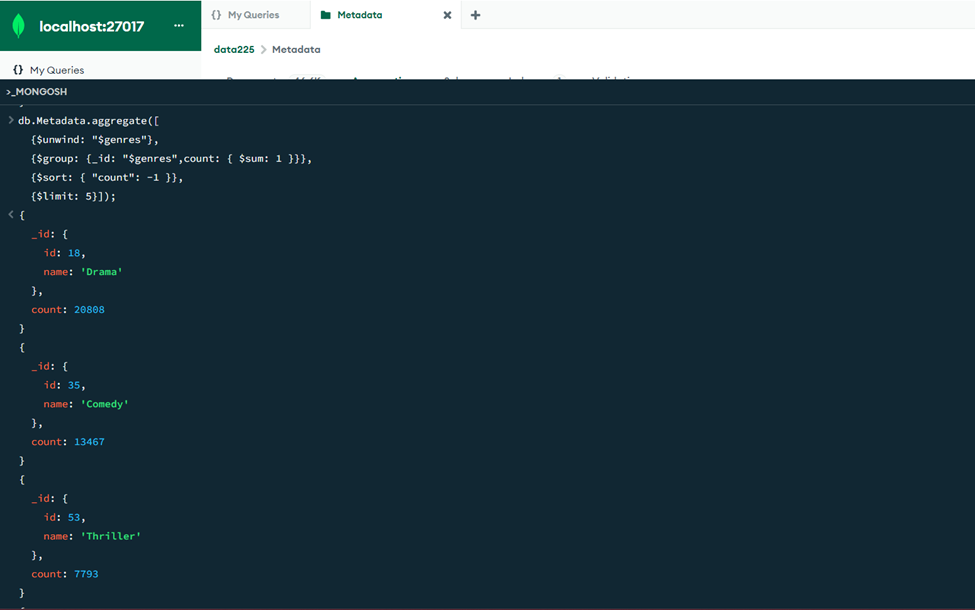
**3. Top 5 Most Prolific Actors in the Movie Metadata Collection:**

| db.Metadata.aggregate([  {$unwind: "$cast"},  {$group: {\_id: "$cast.name",count: { $sum: 1 }}},  {$sort: { "count": -1 }},  {$limit: 5}]); |
| --- |



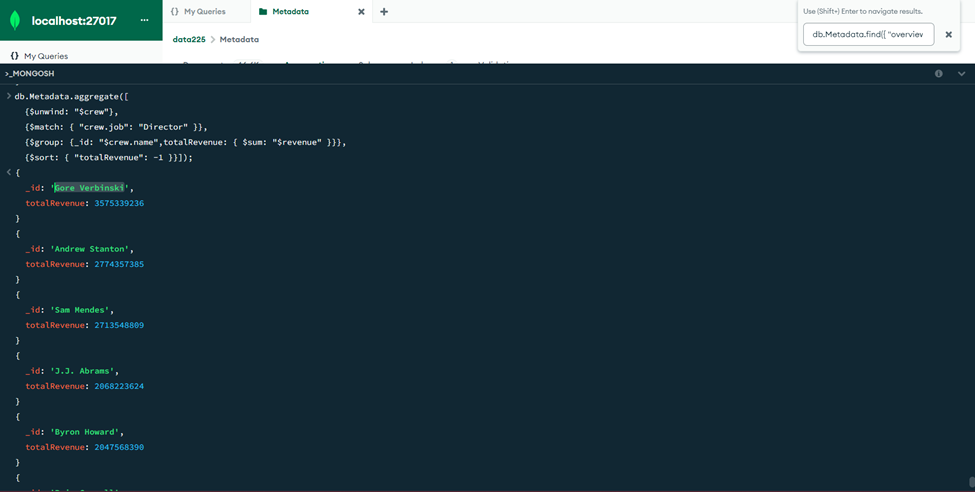
**4. Top 5 Most Common Movie Genres in the Dataset**

| db.Metadata.aggregate([  {$unwind: "$genres"},  {$group: {\_id: "$genres",count: { $sum: 1 }}},  {$sort: { "count": -1 }},  {$limit: 5}]); |
| --- |



**5. Top Revenue by Director**

| db.Metadata.aggregate([  {$unwind: "$crew"},  {$match: { "crew.job": "Director" }},  {$group: {\_id: "$crew.name",totalRevenue: { $sum: "$revenue" }}},  {$sort: { "totalRevenue": -1 }}]); |
| --- |



**6. Top 5 most versatile actors based on the number of unique genres they have appeared in.**

| db.Metadata.aggregate([{ $unwind: "$cast" }, { $group: { \_id: "$cast.name", uniqueGenres: { $addToSet: "$genres" } } }, { $project: { \_id: 1, versatility: { $size: "$uniqueGenres" } } }, { $sort: { versatility: -1 } }, { $limit: 5 }]); |
| --- |



Screenshot of logs generated after while performing queries:

