MongoDB

Dataset: Movie

Create Database : **use mydb** // database name mydb

Collection name: mov

1. Retrieve all documents in the collection.
   db.mov.find()

2. Find all movies released in 2022.
   db.mov.find({year:2022})

3. Find movies with a `RATING` greater than 8.
   db.mov.find({rating:{$gt:8}})

4. Find movies with a `RATING` less than or equal to 5.
   db.mov.find({rating:{$lte:5}})

5. Find movies with a `DURATION` between 90 and 120 minutes.
   db.mov.find({duration: {$gte: 90, $lte:120}})

6. Retrieve movies with the name "Inception."
   db.mov.find({name:"Inception"})

7. Find movies released before the year 2000.
   db.mov.find({name:"Inception"})

8. Retrieve movies with an exact `RATING` of 7.5.
   db.mov.find({rating:7.5})

9. Retrieve only the `NAME` and `YEAR` fields of all documents.
   db.mov.find({},{name:1,year:1,_id:0})

10. Exclude the `ID` field while retrieving documents.
    db.mov.find({},{id:0,_id:0})

11.Find movies with `RATING` greater than 8 and released after 2015.
db.mov.find({rating:{$gt:8},year:{$gt:2015}})

12.Find movies with `RATING` greater than 9 or released before 2000.
db.mov.find({$or:[{rating:{$gt:8}},{year:{$lt:2000}}]})

13.Find movies with `RATING` not equal to 6.
db.mov.find({rating:{$ne:6}})

14.Retrieve movies with `YEAR` not in 2020 or 2021.
db.mov.find({year:{$nin:[2020,2021]}})

15.Find movies with either a `DURATION` of 120 or a `RATING` of 8.5.
db.mov.find({$or:[{duration:120},{rating:8.5}]})

16.Retrieve movies with specific `ID` values in a list.
db.movies.find({ ID: { $in: [1, 2, 3, 5, 8] }})

17. Find movies with `DURATION` not between 90 and 150 minutes.
db.mov.find({duration:{$not:{$gt:90,$lt:120}}})

18. Sort movies by `YEAR` in ascending order.
db.mov.find().sort({year:1})

19. Sort movies by `RATING` in descending order.
db.mov.find().sort({rating:-1})

20.Limit the result to the first 5 movies.
db.mov.find().limit(5)

21. Skip the first 10 documents and return the next 5.
db.mov.find().skip(10).limit(5)

22. Group movies by `YEAR` and calculate the average `RATING` for each year.
db.mov.aggregate([{$group:{_id:"$year",averageRating:{$avg:"$rating"}}}])

23. Count the number of movies released in each year.
db.mov.aggregate([{$group:{_id:"$year" , moviecount:{$sum:1}}}])

24. Find the highest-rated movie.
db.mov.find().sort({rating:-1}).limit(1)

25. Calculate the total duration of all movies combined.
db.mov.aggregate([{$group:{_id: null,totalDuration: { $sum: "$duration"}}}])

26. Find the average duration of movies with a `RATING` above 8.
db.mov.aggregate([{$match:{rating:{$gt:8}}},{$group:{_id: null, totalDuration: { $sum: "$duration"}}}])

27. Find movies whose names start with "A."
db.mov.find({name:{$regex:"^A",$option:"i"}})

28. Find movies whose names contain "Star."
db.mov.find({name:{$regex:"Star",$options:"i"}})

29. Find movies whose names end with "Man."
db.mov.find({name:{$regex:"Man$",$options:"i"}})

30. Count the number of movies with names containing "Avenger."
db.mov.aggregate([{$match:{name:{$regex:"Avenger",$options:"i"}}},{ $count:"moviecount"}])

31. Find movies where `DURATION` is greater than `RATING` multiplied by 10.
db.mov.find({$expr:{$gt:["$duration", {$multiply:["$rating",10]}]}})

32. Find movies where `YEAR` is an even number.
db.mov.find({year:{$mod:[2,0]}})

33. Increment the `RATING` of all movies released in 2022 by 1.
db.mov.updateMany({year:2022},{$inc:{rating:1}})

34. Find movies with a missing `RATING` field.
db.mov.find({rating:{$exists:false}})

35. Retrieve movies sorted by both `YEAR` (ascending) and `RATING` (descending).
    db.mov.find().sort({year:1,rating:-1})

36. Retrieve all unique years in which movies were released.
    db.mov.distinct("year")

37. Find movies that do not contain "The" in their name.
    db.mov.find({name:{$not:{$regex:"The",$options:"i"}}})