Assignment 1: Storage and processing of data

We have provided a data set which consists of 10 million ratings and 100,000 tag applications applied to 10,000 movies by 72,000 users.

Design data structures to represent the data (you might have one document that stores all information aggregated under user object or one document that stores information about aggregate under move object, or split into two or tree aggregations).

Your models should be able to answer the following queries below.

Create a data store. Extract, transform and load the provided data set into your data storage.

Answer the following questions.

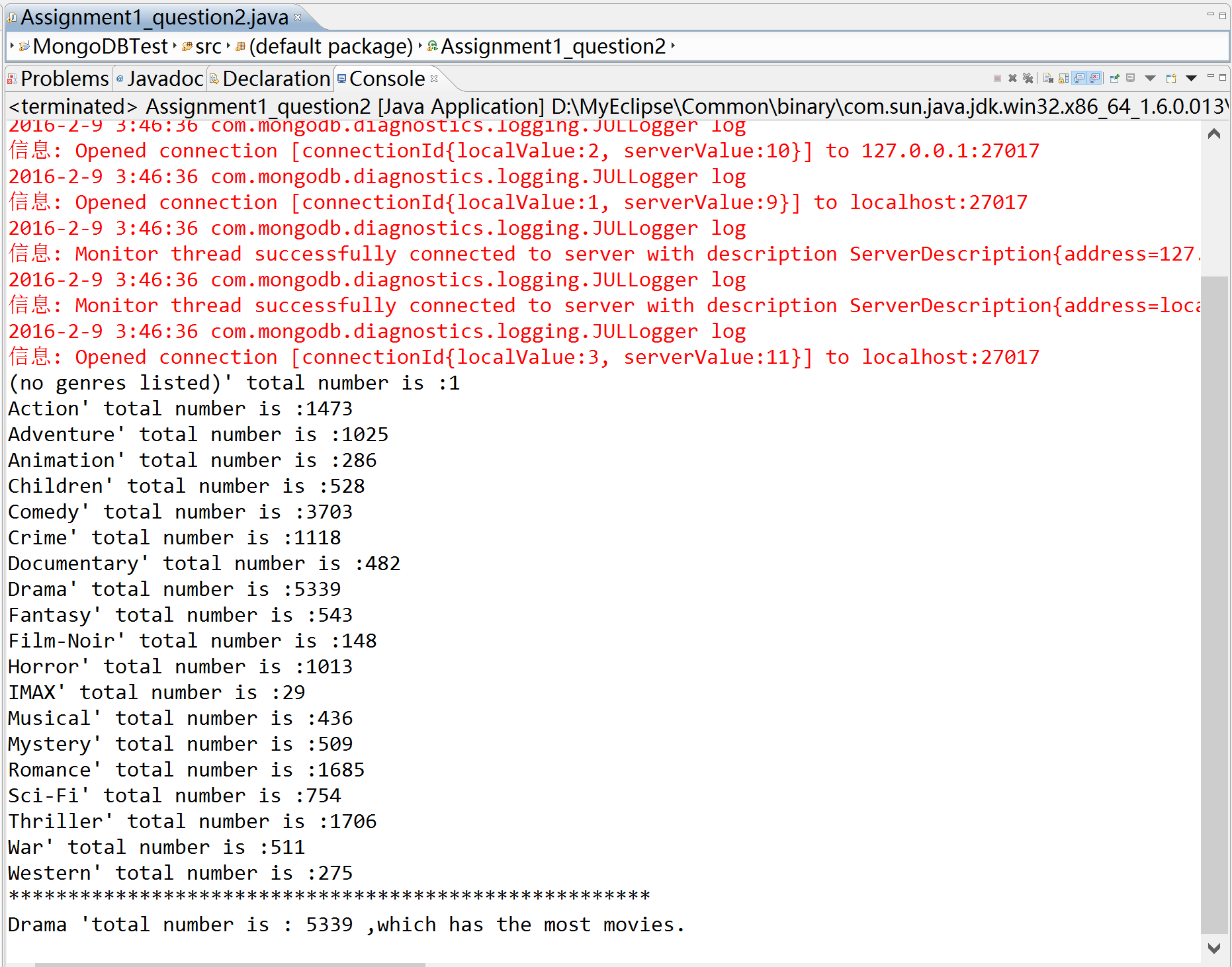
**Preparation:**

|  |  |
| --- | --- |
|  | Transfer the format of the given file. |
|  | Because MongoDB could only store files with suffix .cvs or .json , and all three given files ‘s format is .dat , |
|  | which couldn’t be read by the MongoDB . Therefore, we need do some necessary preparation--transferring the given |
|  | files to .json file. In order to complete the whole process, java has been used. |
|  |  |
|  | package Assignment1; |
|  |  |
|  | import java.io.BufferedReader; |
|  | import java.io.FileNotFoundException; |
|  | import java.io.FileReader; |
|  | import java.io.IOException; |
|  | import java.io.PrintWriter; |
|  | import java.io.UnsupportedEncodingException; |
|  |  |
|  | public class CsvToJason { |
|  |  |
|  | public static void main(String[] args) throws FileNotFoundException, UnsupportedEncodingException { |
|  | // tags first |
|  | System.out.println("Start with transfering tags.dat to tags.json"); |
|  | String csvFileToRead = "/Users/weiyituo/Desktop/Data analytic/project1/Mongo\_assignment\_1\_25\_16/tags.dat"; |
|  | BufferedReader br = null; |
|  | String line = ""; |
|  | String splitBy = "::"; |
|  | PrintWriter writer = new PrintWriter("/Users/weiyituo/Desktop/Data analytic/project1/tags.json", "UTF-8"); |
|  | try { |
|  | br = new BufferedReader(new FileReader(csvFileToRead)); |
|  | while ((line = br.readLine()) != null) { |
|  | String[] row = line.split(splitBy); |
|  | writer.println("{ \"UserID\": " + row[0] + ", \"MovieID\": " + row[1] + ", \"Tag\": \"" + row[2] |
|  | + "\", \"Timestamp\": " + row[3] + " },"); |
|  | } |
|  | } catch (FileNotFoundException e) { |
|  | e.printStackTrace(); |
|  | } catch (IOException e) { |
|  | e.printStackTrace(); |
|  | } finally { |
|  | if (br != null) { |
|  | try { |
|  | br.close(); |
|  | } catch (IOException e) { |
|  | e.printStackTrace(); |
|  | } |
|  | } |
|  | } |
|  | writer.close(); |
|  | System.out.println("Done with transfering tags.dat to tags.json"); |
|  | // Rating second |
|  | System.out.println("Start with transfering ratings.dat to rating.json"); |
|  | csvFileToRead = "/Users/weiyituo/Desktop/Data analytic/project1/Mongo\_assignment\_1\_25\_16/ratings.dat"; |
|  | br = null; |
|  | line = ""; |
|  | splitBy = "::"; |
|  | writer = new PrintWriter("/Users/weiyituo/Desktop/Data analytic/project1/rating.json", "UTF-8"); |
|  | try { |
|  | br = new BufferedReader(new FileReader(csvFileToRead)); |
|  | while ((line = br.readLine()) != null) { |
|  | String[] row = line.split(splitBy); |
|  | writer.println("{ \"UserID\": " + row[0] + ", \"MovieID\": " + row[1] + ", \"Rating\": " + row[2] |
|  | + ", \"Timestamp\": " + row[3] + " },"); |
|  | } |
|  | } catch (FileNotFoundException e) { |
|  | e.printStackTrace(); |
|  | } catch (IOException e) { |
|  | e.printStackTrace(); |
|  | } finally { |
|  | if (br != null) { |
|  | try { |
|  | br.close(); |
|  | } catch (IOException e) { |
|  | e.printStackTrace(); |
|  | } |
|  | } |
|  | } |
|  | writer.close(); |
|  | System.out.println("Done with transfering ratings.dat to ratings.json"); |
|  | // Movie at the end |
|  | System.out.println("Start with transfering movies.dat to movies.json"); |
|  | csvFileToRead = "/Users/weiyituo/Desktop/Data analytic/project1/Mongo\_assignment\_1\_25\_16/movies.dat"; |
|  | br = null; |
|  | line = ""; |
|  | splitBy = "::"; |
|  | writer = new PrintWriter("/Users/weiyituo/Desktop/Data analytic/project1/movies.json", "UTF-8"); |
|  | try { |
|  | br = new BufferedReader(new FileReader(csvFileToRead)); |
|  | while ((line = br.readLine()) != null) { |
|  | String[] row = line.split(splitBy); |
|  | writer.print("{ \"MovieID\": " + row[0] + ", \"Title\": \"" + row[1] + "\", \"Genres\": ["); |
|  | int length = row.length; |
|  | for (int i = 2; i < length; i++) { |
|  | writer.print("\"" + row[i] + "\""); |
|  | if (i < length - 1) |
|  | writer.print(", "); |
|  | } |
|  | writer.println("] },"); |
|  | } |
|  | } catch (FileNotFoundException e) { |
|  | e.printStackTrace(); |
|  | } catch (IOException e) { |
|  | e.printStackTrace(); |
|  | } finally { |
|  | if (br != null) { |
|  | try { |
|  | br.close(); |
|  | } catch (IOException e) { |
|  | e.printStackTrace(); |
|  | } |
|  | } |
|  | } |
|  | writer.close(); |
|  | System.out.println("Done with transfering movies.dat to movies.json"); |
|  | } |
|  |  |
|  | } |

|  |  |
| --- | --- |
|  | **Q1: What genre is the movie CopyCat in?** |
|  | db.movies.find({"Title":"Copycat (1995)"}).pretty() |

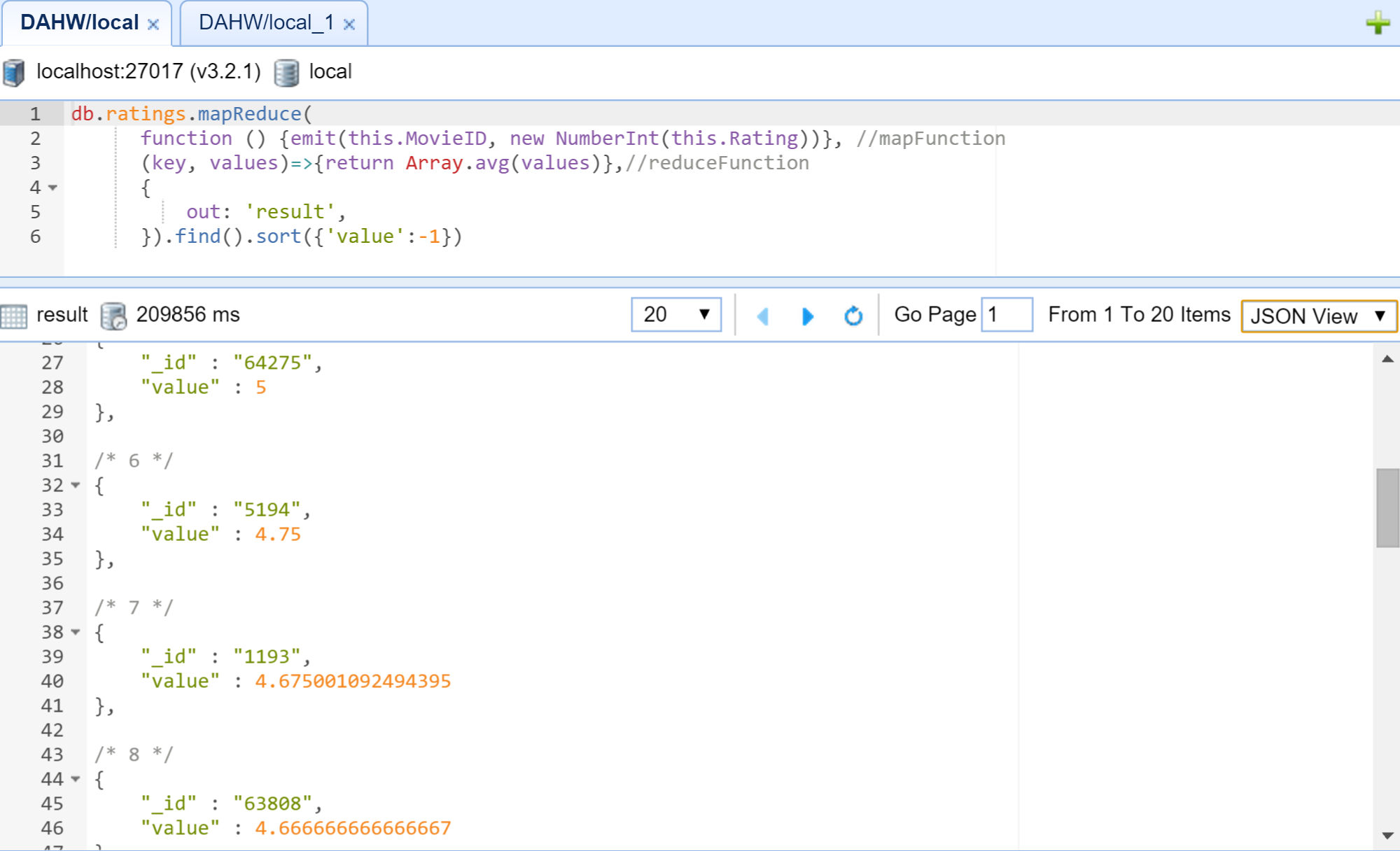


|  |
| --- |
|  |
|  | **Q2:what genre has the most movies?** |
|  | In this question, we need to find all genre of movies, but as we can see that all records have the following format: |
|  | {"MovieID":"2","Title":"Jumanji (1995)","Genres":["Adventure","Children","Fantasy"]} |
|  | So, at first we need to implement java programing to count the number |
|  |  |
|  |  |
|  | import com.mongodb.MongoClient; |
|  | import com.mongodb.client.MongoDatabase; |
|  | import com.mongodb.BasicDBObject; |
|  | import com.mongodb.DB; |
|  | import com.mongodb.DBCollection; |
|  | import com.mongodb.DBCursor; |
|  | import com.mongodb.Mongo; |
|  |  |
|  | import java.util.ArrayList; |
|  | import java.util.LinkedList; |
|  | import java.util.TreeSet; |
|  |  |
|  | public class Assignment1\_question2 { |
|  | public static void main(String args[]) { |
|  |  |
|  | Mongo mongo = new Mongo("localhost", 27017); |
|  | DB db = mongo.getDB("local"); |
|  | MongoClient mongoClient = new MongoClient(); |
|  | MongoDatabase db\_1 = mongoClient.getDatabase("local"); |
|  | DBCollection collection = db.getCollection("movies"); |
|  | CountGenres(collection, db\_1); |
|  | } |
|  |  |
|  | public static void CountGenres(DBCollection collection, MongoDatabase db\_1) { |
|  | DBCursor cursor = collection.find(); |
|  | TreeSet<String> ts = new TreeSet<String>(); |
|  | String[] genres; |
|  | while (cursor.hasNext()) { |
|  | genres = ((BasicDBObject) cursor.next()).get("Genres").toString().split("\\|"); |
|  | for (int i = 0; i < genres.length; i++) { |
|  | String temp=genres[i]; |
|  | if (ts.contains(temp)) { |
|  | continue; |
|  | } else { |
|  | ts.add(temp); |
|  | } |
|  | } |
|  | } |
|  | int count = 0; |
|  | String temp = null; |
|  | LinkedList<String> fList=new LinkedList<String>(); |
|  | ArrayList<Integer> countList=new ArrayList<Integer>(); |
|  | for (String f : ts) { |
|  | DBCursor cursor\_2 = collection.find(); |
|  | while (cursor\_2.hasNext()) { |
|  | temp = ((BasicDBObject) cursor\_2.next()).getString("Genres"); |
|  | if (temp.contains(f)) |
|  | count++; |
|  | } |
|  | fList.add(f); |
|  | countList.add(count); |
|  | System.out.println(f + "' total number is :" + count); |
|  | count = 0; |
|  | } |
|  | System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"); |
|  | int max=Integer.MIN\_VALUE; |
|  | int resultIndex=0; |
|  | for (int i = 0; i < countList.size(); i++) { |
|  | max=Math.max(max, countList.get(i)); |
|  | } |
|  | for (int i = 0; i < countList.size(); i++) { |
|  | if (countList.get(i)==max) { |
|  | resultIndex=i; |
|  | } |
|  | } |
|  | System.out.println(fList.get(resultIndex)+" 'total number is : "+countList.get(resultIndex)+" ,which has the most movies."); |
|  | } |
|  | } |



|  |  |
| --- | --- |
|  | 1. **what tags did user 146 use to describe the movie "2001: A Space Odyssey”**   >>>import sys  >>> import pymongo |
|  | >>> from pymongo import MongoClient |
|  | >>> client=MongoClient() |
|  | >>> db=client.test |
|  | >>> movieCollection=db.movies |
|  | >>> S='2001: A Space Odyssey' |
|  | >>> for item in db.movies.find({'Title':{'$regex':S}}): |
|  | print item["MovieID"] |
|  | 924 |
|  | >>> for item in db.tags.find({"MovieID":"924","UserID":"146"}): |
|  | print item["Tag"]  D:\QQ聊天记录\506964640\Image\C2C\578E1312D7D88C6C29492C8CDC13ADB2.png |

1. **What are the top 5 movies with the highest avg rating?**   
   db.ratings.mapReduce(  
         function () {emit(this.MovieID, new NumberInt(this.Rating))}, //mapFunction  
         (key, values)=>{return Array.avg(values)},//reduceFunction  
         {  
             out: 'result',  
         }).find().sort({'value':-1})



|  |
| --- |
|  |
|  | **5) What is the highest avg rating possible?**  db.ratings.mapReduce(  function () {emit(this.MovieID, new NumberInt(this.Rating))}, //mapFunction  (key, values)=>{return Array.avg(values)},//reduceFunction  {  out: 'result',  }).find({value:5}).count()/db.movies.count() |

**6) Write 3 different queries of your choice to demonstrate that your data storage is working.**

|  |
| --- |
|  |

db.ratings.insert({

"UserID" : "36",

"MovieID" : "1020",

"Rating" : "4",

"Timestamp" : "1049772929"

})

db.ratings.find({UserID: '36', MovieID:'1020'})

db.ratings.remove({UserID: '36', MovieID:'1020'})

