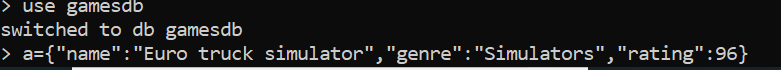
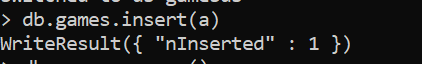
**MongoDB hands on part 1**

1. Create a collection called ‘games’. We’re going to put some games in it. Add 5 games to the database. Give each document the following properties: name, genre, rating (out of 100) If you make some mistakes and want to clean it out, use remove() on your collection.

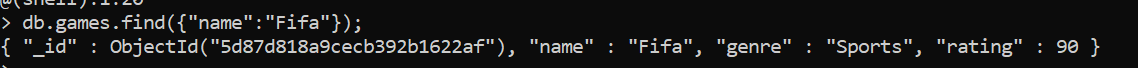




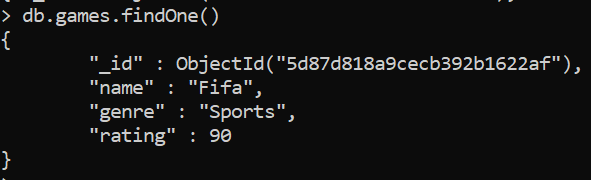
1. Write a query that returns all the games.



1. Write a query to find one of your games by name without using limit().



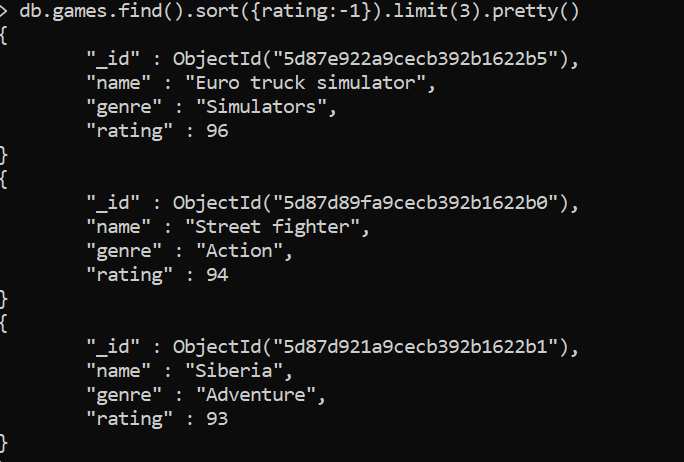
1. Use the findOne method.



1. Look how much nicer it’s formatted!

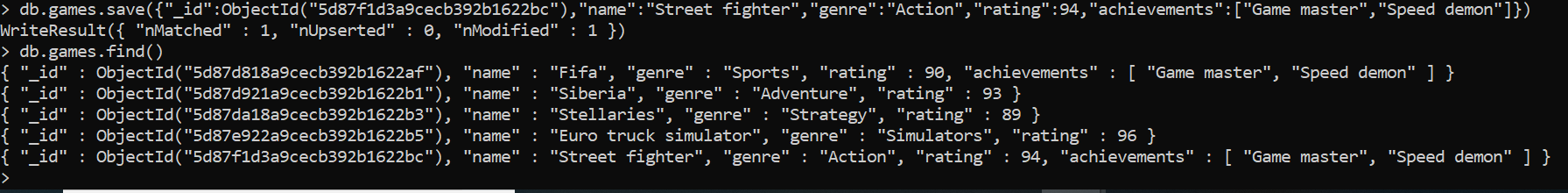


1. Write a query that returns the 3 highest rated games.

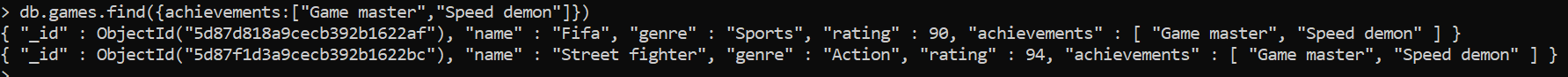


1. Update your two favorite games to have two achievements called ‘Game Master’ and ‘Speed Demon’, each under a single key. Show two ways to do this. Do the first using update() and do the second using save(). Hint: for save, you might want to query the object and store it in a variable first.

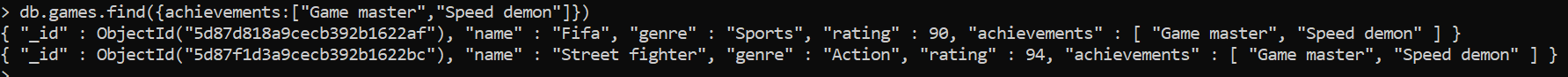




1. Write a query that returns all the games that have both the ‘Game Master’ and the ‘Speed Demon’ achievements.



1. Write a query that returns only games that have achievements. Not all of your games should have achievements, obviously. You could take the screenshots by press



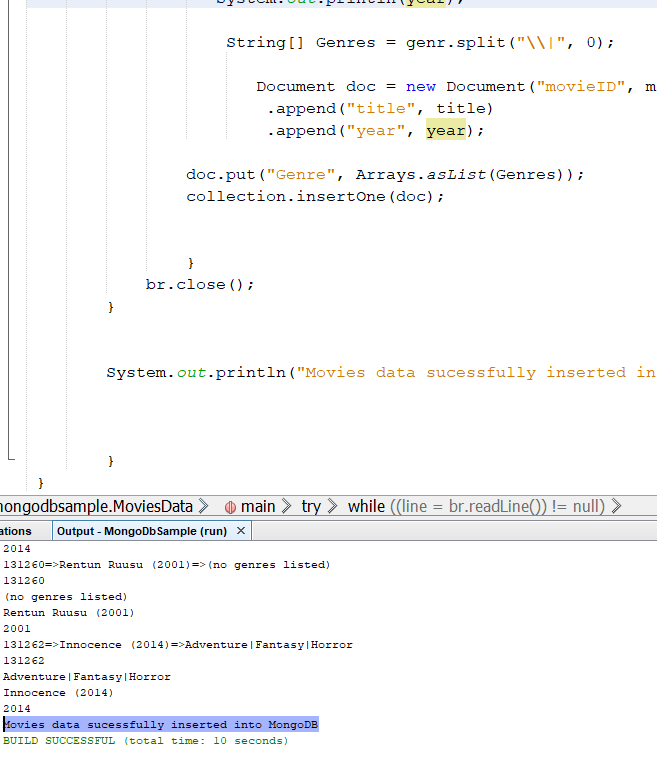
**MongoDB hands on part 2**

Write a Java (could be a console app - will only run once to import the data into MongoDB) program to read the following file, and insert into 3 different collections (movies, ratings, tags). - MovieLens 10M Stable benchmark dataset. 10 million ratings and 100,000 tag applications applied to 10,000 movies by 72,000 users. <http://grouplens.org/datasets/movielens/>

Once the data are inserted into MongoDB, do the followings using MapReduce: Write a MapReduce to do the followings: -

**Number of Movies released per year (Movies Collection)**

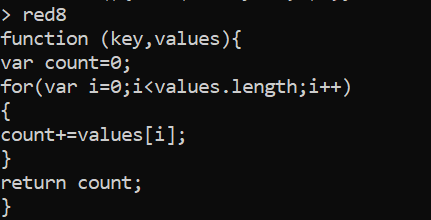
**JAVA console to dump data in mongoDB**



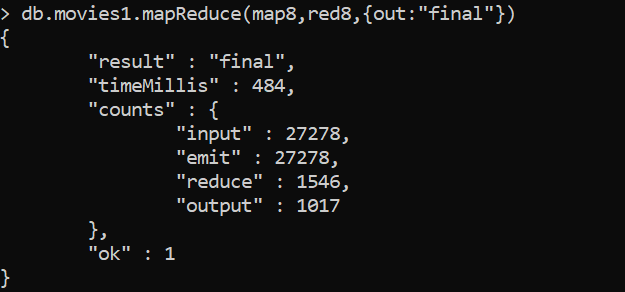
**Mapper function=map8**



**Reducer function=red8**



**Output of mapReducer**



**Number of Movies per genre (Movies Collection)**

**Mapper function= map7**

function ()

{

for(var idx=0;idx<this.Genre.length;idx++)

{

var key = this.Genre[idx];

var value = 1;

emit(key,value);

}

}

**Reduce Function=red7**

function(key,values){

var count=0;

for(var i=0;i<values.length;i++)

{

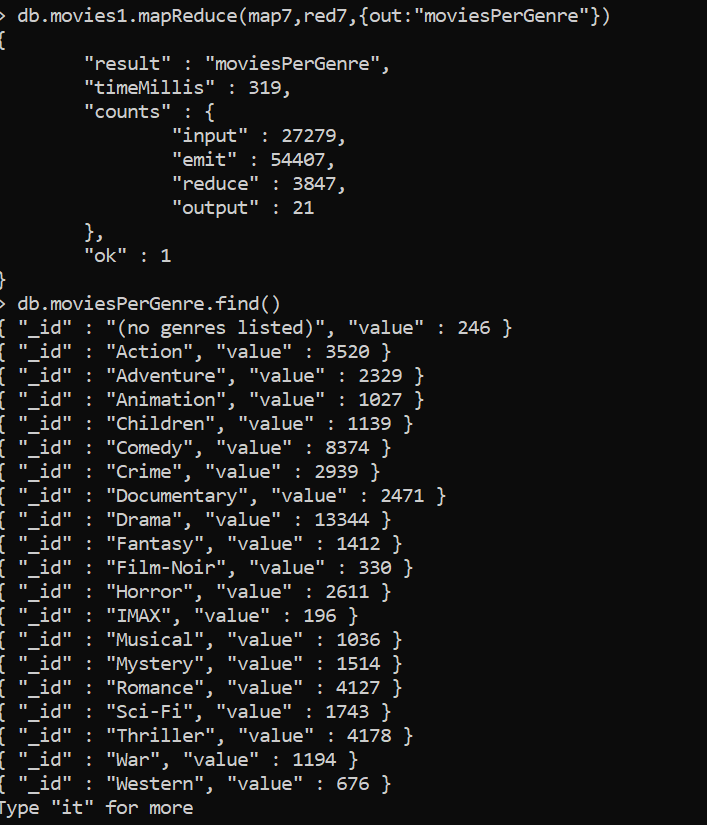
count+=values[i];

}

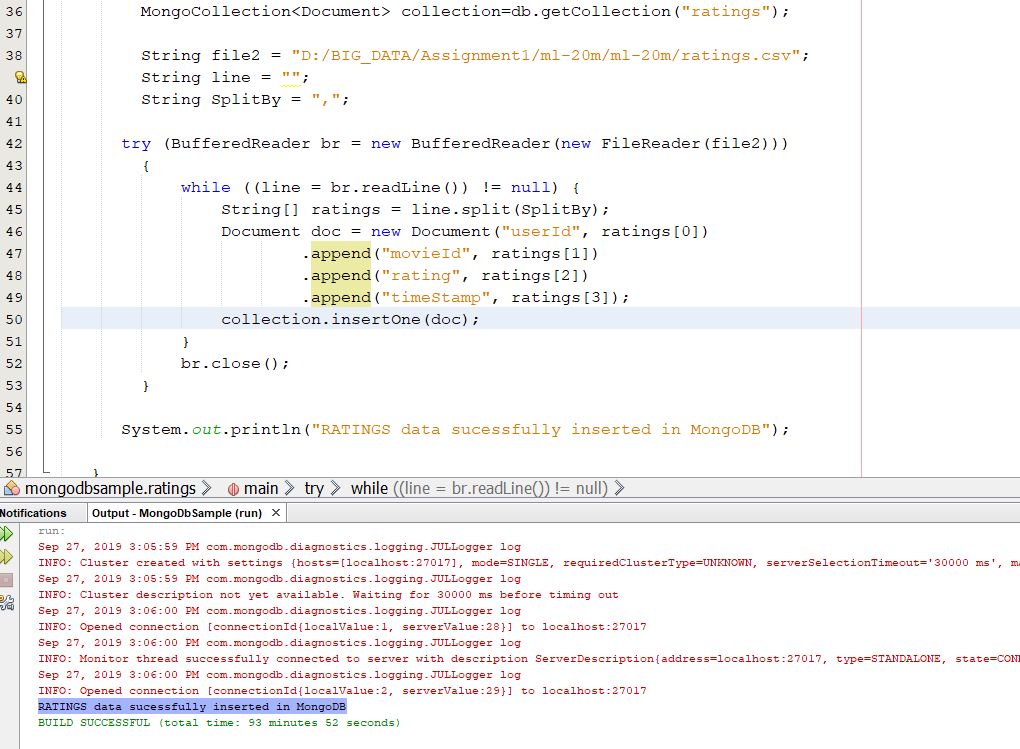
return count;

}

**Output of mapReducer**



**4.3. Number of Movies per rating (Ratings Collection)**



**Mapper function=m1**

function(){

Emit(this.rating,1);

}

**Reducer function=r1**

Function(key,values)

{

var count=0;

for(var I=0;i<values.length;i++)

{

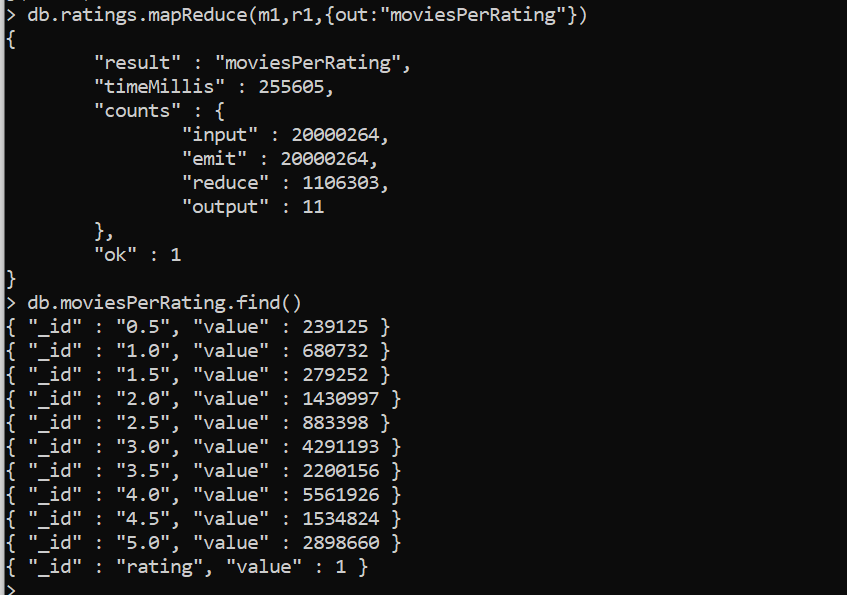
count+=values[i];

}

return count; }

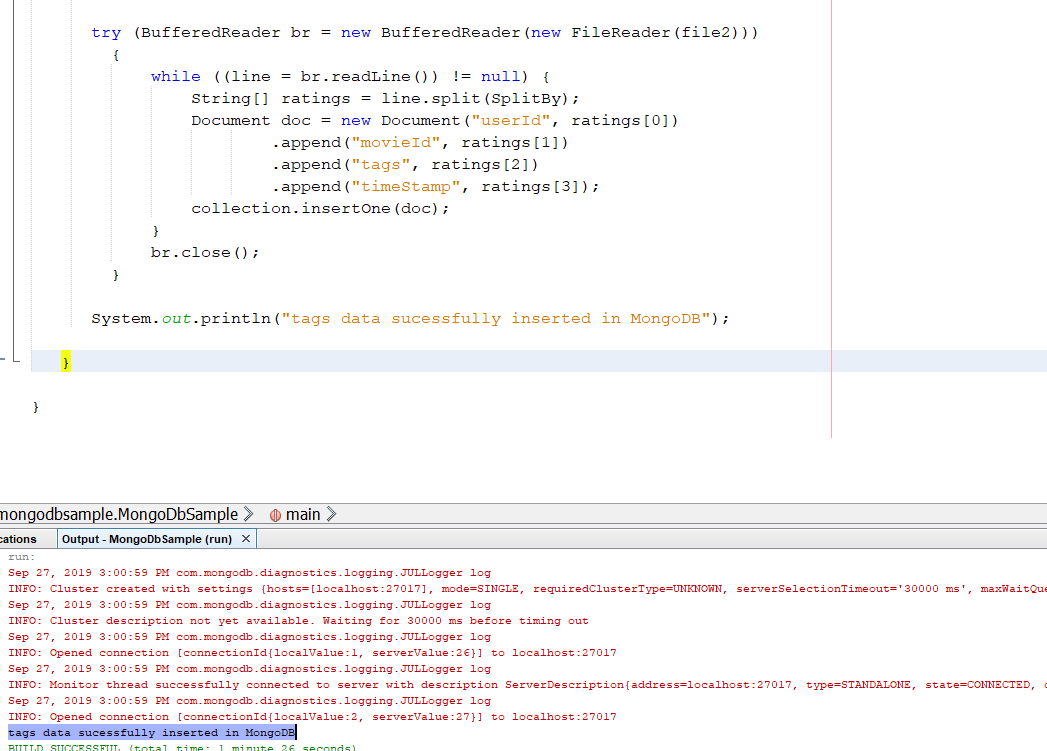
}

**Output of mapReduce**



**4.4. Number of times each movie was tagged (Tags Collection)**

JAVA console to dump data in mongoDB:



**Mapper function= map1**

function()

{

emit(this.movieId,1);

}

**Reducer function=red1**

function(key,values)

{

var count=0;

for(var i=0;i<values.length;i++)

{

count+=values[i];

}

return count;

}

}

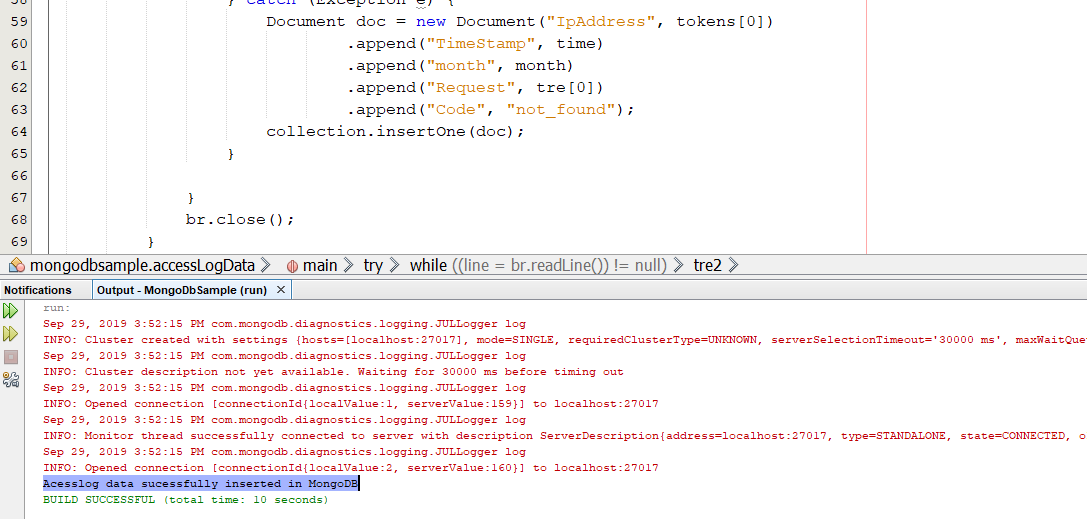
Output of mapReduce

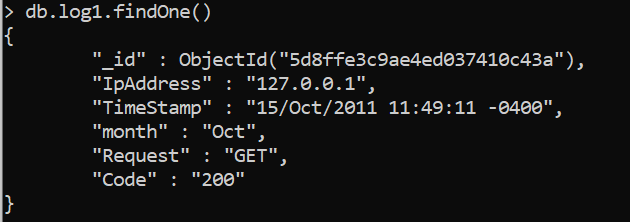


**MongoDB hands on part 3**

Write a Java (could be a console app - will only run once to import the data into MongoDB) program to read the access.log file (attached) and insert into access collection. Once the data are inserted into MongoDB, do the followings using MapReduce:

**JAVA console to dump data into mongoDB**





**5.1. Number of times any webpage was visited by the same IP address.**

**Mapper function name=m7**

function() {

emit(this.IpAddress,1) ;

}

**Reducer function name=r7**

function(key,values)

{

var count =0;

for(var i=0;i<values.length;i++)

{

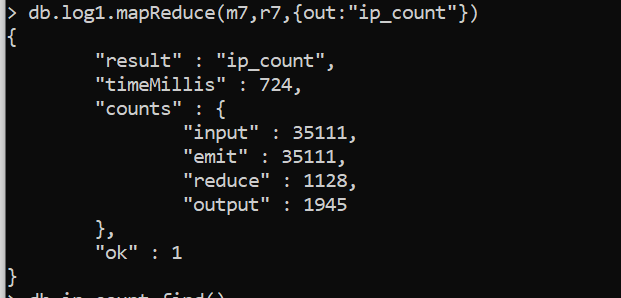
Count+=values[i];

}

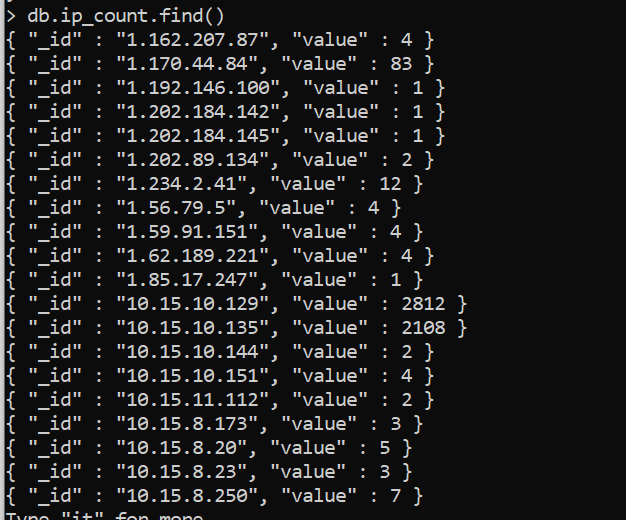
return count;

}

**Mapper Reducer output**

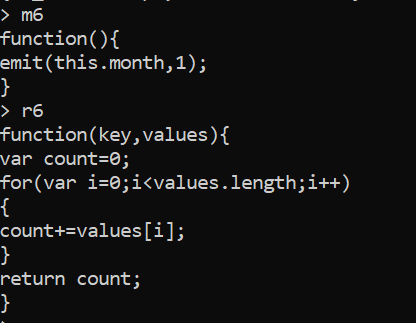


**Final output**

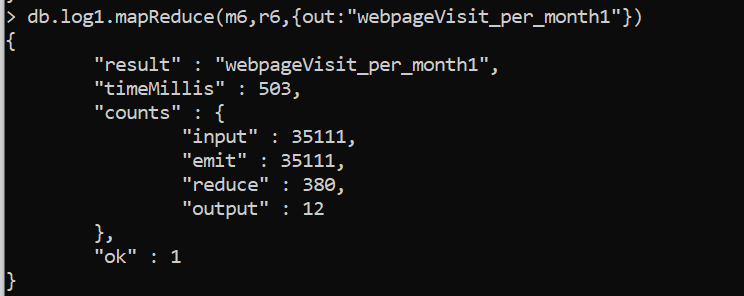


**5.2. Number of times any webpage was visited each month**

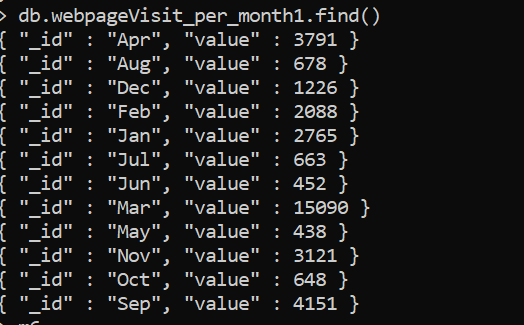
**Mapper function name=m6 and Reducer function name=r6**



**Output after running mapReduce**



**Final output**

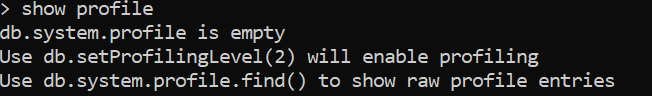


**MongoDB hands on part 4**

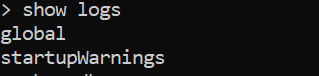
Execute 5 commands of your choice from each of the following groups and paste the screenshots in a word document.

**mongo> help [5 commands]**

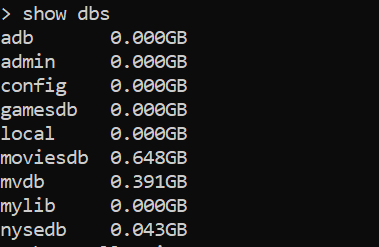
1



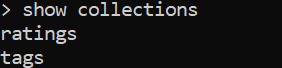
2



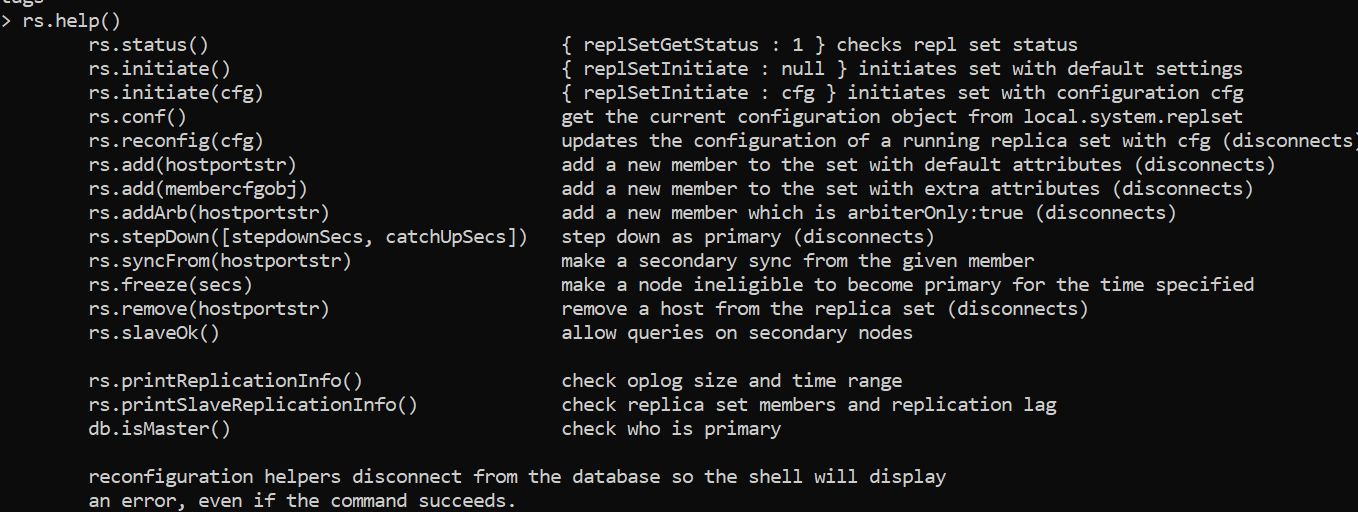
3



4



5

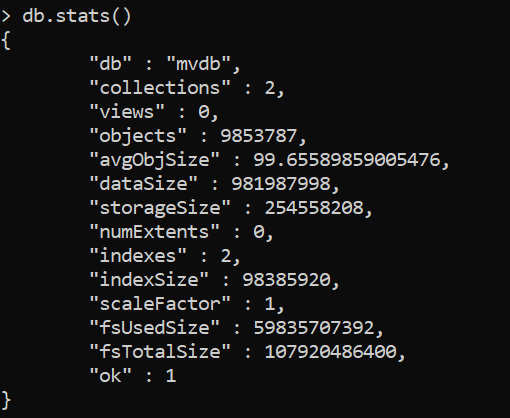


**mongo> db.help() [5 commands]**

1



2



3



4



5.

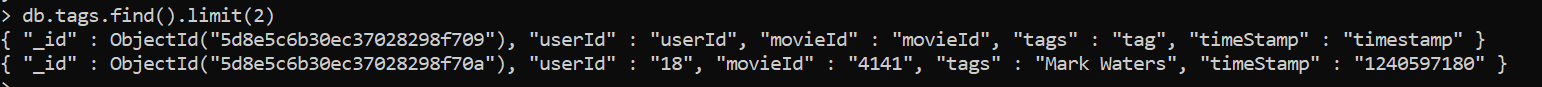


**mongo> db.mycoll.help() [10 commands]**

1



2



3



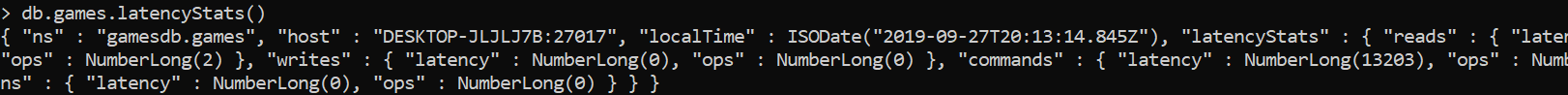
4



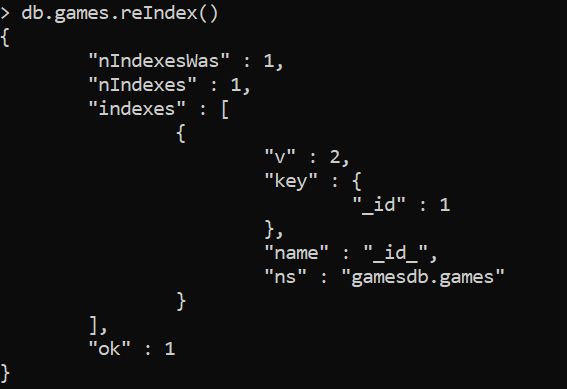
5



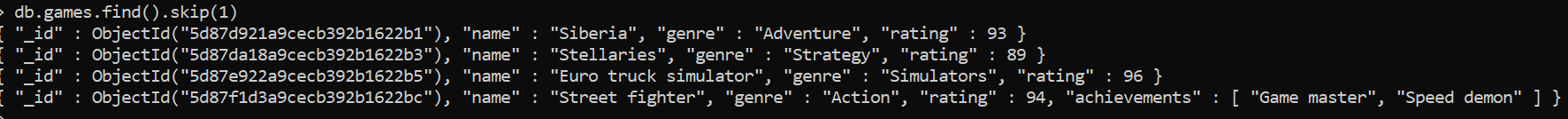
6



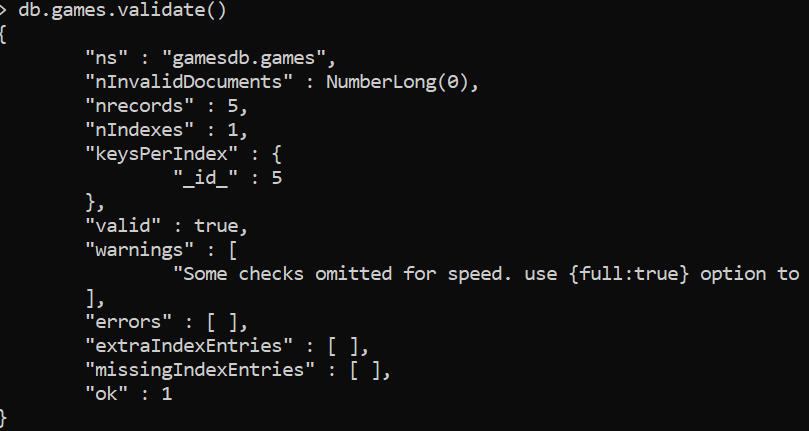
7



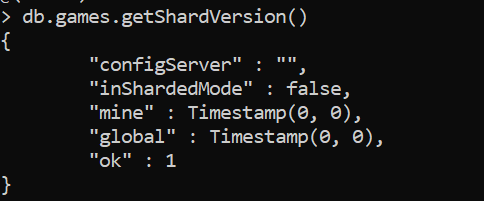
8



9



10



**MongoDB hands on part 5**

Write a .bat (for Windows) or .sh (for MacOS) to import the entire NYSE dataset (stocks A to Z) into MongoDB.

NYSE Dataset Link: <http://msis.neu.edu/nyse/nyse.zip>

**nysedb\_mongo.bat**

@echo off

For %%G in (A B C D E F G H I J K L M N O P Q R S T U V W X Y Z) do mongoimport -d nysedb -c stocks type csv --headerline --file D:\BIG\_DATA\Assignment1\nyse\NYSE\NYSE\_daily\_prices\_%%G.csv

pause

