

CENTRALESUPÉLEC

2021-2022

Big Data, Techniques and Platforms

Document Databases

For this assignment you are required to use MongoDB. The only necessary file is:

* artistsnestedsongs.json

This file contains the same informations you used in the previous exercise of Spark/MapReduce. Download the files and store them in a directory of your computer. I suggest you to create again a directory called data.

1 IMPORT DATA

Import data in your MongoDB database, and take advantage of the Studio3T graphical inter- face.

2 QUERYING DATA

Before starting with the queries look at data and provide a short description of them: the most common structure of the documents (the most present attributes, nested documents, etc.). Then provide the queries that give you the following informations and the returned result.

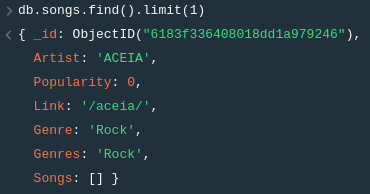
**Descriptive Details about the dataset:**

* 7 Features, dtype = String(6), objectid(1)
* Total count: 3242 objects

1. **(1 point)** - Retrieve one element of the collection.

Solution:

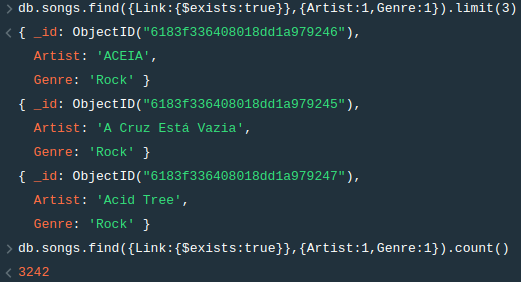
db.songs.find().limit(1)



1. **(1 point)** - Retrieve the artist name (Artist attribute) and the main genre of any artist for which the link to the webpage is provided.

Solution:

db.songs.find({Link:{$exists:true}},{Artist:1,Genre:1})

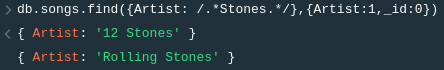


Total number of artists for which the link to the webpage is provided: 3242

1. **(1 point)** - Show the list of artist (just their name) that contain Stones in the name.

Solution:

db.songs.find({Artist: /.\*Stones.\*/},{Artist:1,\_id:0})



1. **(1 point)** - Show the number of artists that are returned by the previous query.

Solution:

db.songs.find({Artist: /.\*Stones.\*/},{Artist:1,\_id:1}).count()



1. **(1 point)** - Provide the name of the artists that have popularity greater than 4.

Solution:

db.songs.find({Popularity: {$gt:4}},{Artist:1,\_id:1})  
  
There are a total of 269 artists who have a popularity greater than 4

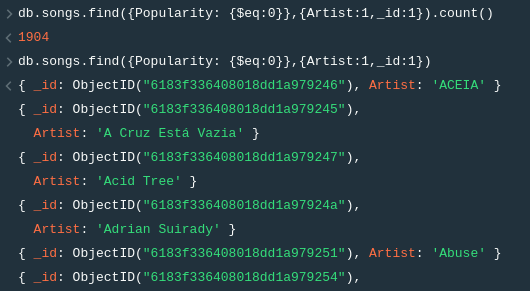


1. **(1 point)** - Show the artists with the lowest popularity.

Since the lowest popularity can be 0, we query all the artists with popularity equal to 0

Solution:

db.songs.find({Popularity: {$eq:0}},{Artist:1,\_id:1})

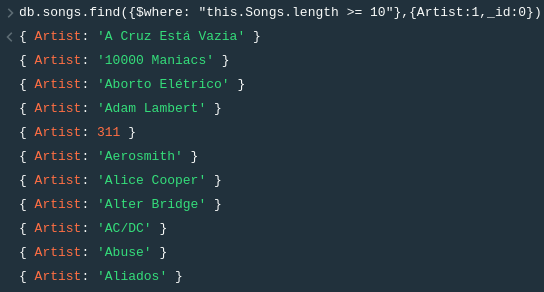


1. **(1 point)** - Give the list of artists that have at list 10 songs listed.

Solution:

db.songs.find({$where: "this.Songs.length >= 10"},{Artist:1,\_id:0})

There are a total of 2050 artists that have at least 10 songs listed

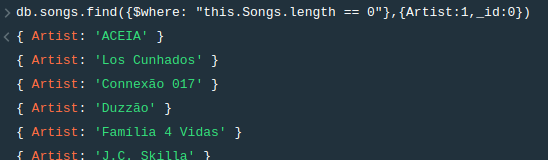


1. **(1 point)** - Give the artists (if any) that do not have songs listed.

Solution:

db.songs.find({$where: "this.Songs.length == 0"},{Artist:1,\_id:0}

There are a total of 51 artists that do not have any songs listed



1. **(3 point)** - Give the list of artists that have at list 10 distinct songs listed and the cri- teria that you used for this distinct operation.

Solution: db.songs.aggregate([{"$project": { "\_id": 0,"Artist": 1,"Unique\_Songs": {

"$size": { "$setDifference": [ "$Songs", []]}}},

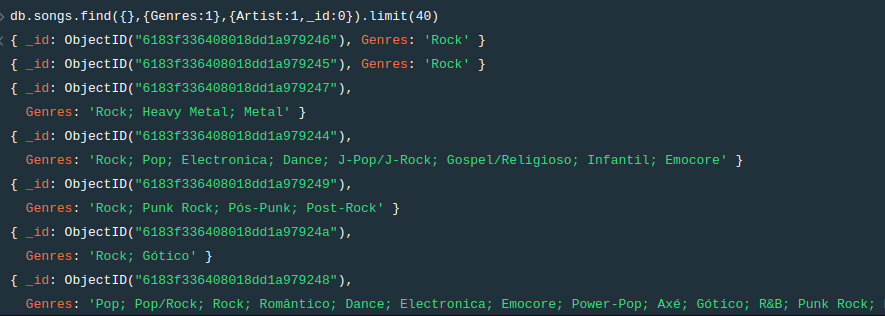
{ "$match": {"Unique\_Songs": {"$gte": 10}}},

{ "$project": {"\_id": 0, "Artist": 1}}]);



1. **(1 point)** - Show the complete set of genres for 40 artists.

Solution: db.songs.find({},{Genres:1},{Artist:1,\_id:0}).limit(40)



1. **(1 point)** - Comment on how the list is specified and on how it is possible to query this attribute.

Solution: The **find** function in MongoDB takes in two parameters; *query* and *projection*. In the solution provided in Question 10 above, the *query* parameter is left empty as it will allow us to return all the documents from the database, whereas the *projection* parameter specifies the fields to return in the documents that match the query filter. For this parameter, we specify to show all documents with only their Genres specified, hence the output of the query only returns the object ID and the Genre. Lastly, we apply a limit function and set the counter at 40 to only show the first 40 objects from the query output.

1. **(3 points)** - Give the number of artists having Folk among the genres.

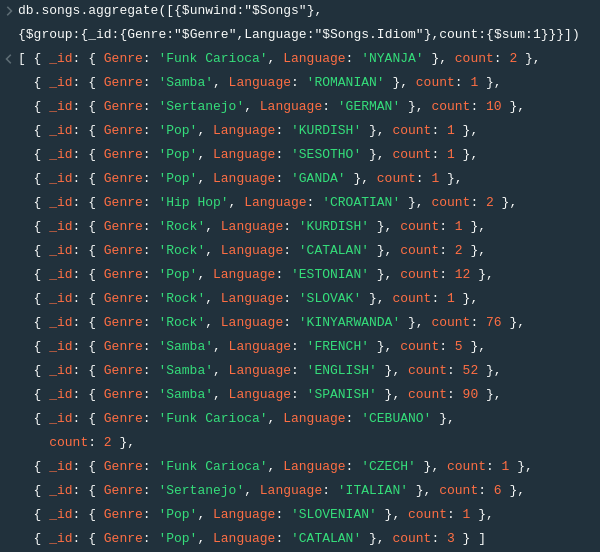
Solution:

db.songs.find({Genres: ./\*Folk.\*/},{Artist:1,\_id:0}).count()



1. **(3 points)** Give the number of song by main genre of the artist and language.

Solution: Solution: sdb.songs.aggregate([{$unwind:”$Songs”},{$group:{\_id:{  
 Genre:”$Genre”,Language:”Songs.Idiom”},count:{$sum:1}}])



3 BONUS - ENJOY WITH DATA

Now you can give freedom to your self-initiative and fantasy.

Using Spark, Dataframes and/or MongoDB provide one analysis of your choice on the dataset.