Assignment 5

Hunjun Shin

MongoDB

1. Load the file restaurants.json into a MongoDB database. The file contains data on 3,772 restaurants in New York City.

Write the following queries in MongoDB (show the output of your queries):

|  |
| --- |
| 1. Display all the restaurants located in the boroughs Bronx or Brooklyn.   A screen shot of a computer code  Description automatically generatedA screenshot of a computer program  Description automatically generated |
| 1. Find the restaurant id, name, borough and cuisine for those restaurants whose name starts with the letters ’Mad’.   A computer screen with text  Description automatically generated with medium confidenceA screen shot of a computer screen  Description automatically generated |
| 3.  Find the restaurants that have received a score between 80 and 90 (inclusive). |
| 1. Display the restaurant id and name of restaurants which have received a ’C’ grade in year 2014.   A computer screen with text and numbers  Description automatically generatedA screenshot of a computer  Description automatically generated |
| 5.Find the cuisine that has the highest number of restaurants. |
| 1. Find the restaurants that do not prepare an ’American’ cuisine and their average grade score is higher than 30. Display the restaurant ids and their average score.   A computer code on a black background  Description automatically generatedA screenshot of a computer program  Description automatically generated |
| 7.(g)  For each restaurant display only the grades that were recorded from the year 2014 onwards.  A screen shot of a computer code  Description automatically generatedA screenshot of a computer program  Description automatically generated |
| 8.(h)  Calculate the average score across all the restaurants in the collection.  A screen shot of a computer  Description automatically generatedA computer screen with text and numbers  Description automatically generated |

2. Create a collection named sales with the following documents:

A close-up of a state

Description automatically generated

a)

|  |
| --- |
| A screen shot of a computer code  Description automatically generatedA screen shot of a computer code  Description automatically generated |

b)

|  |  |
| --- | --- |
| i.   |  | | --- | | A computer code on a black background  Description automatically generated | |
| ii.   |  | | --- | | A computer code on a black background  Description automatically generated | |
| iii.   |  | | --- | | A black background with text  Description automatically generated | |
| iv.   |  | | --- | | A group of people with text  Description automatically generated with medium confidence | |
| v.   |  | | --- | | A black background with text  Description automatically generated | |
| vi.   |  | | --- | | A black background with text on it  Description automatically generated | |

1. Print the final sales collection

|  |
| --- |
| A screenshot of a computer screen  Description automatically generated |

3. Load the file books.json into a MongoDB database. Write a Python script that lets the user enter a books category and prints the ISBN and titles of all the books in that category.

|  |
| --- |
| A screen shot of a computer program  Description automatically generatedA screenshot of a computer program  Description automatically generated |

MapReduce

1. Projection πS(R): From each tuple of relation R produce only the components for the attributes in S.

|  |
| --- |
| 1.map(key, value):   1. projected\_value = {attribute in S} 2. emit(projected\_value, None) |
| 1.reduce(projected\_value, \_):  2. emit(projected\_value, None) |

1. Intersection R∩S: Return the tuples that are present in both relations R and S. Assume that relations R and S have the same schema (same attributes and same type).

|  |
| --- |
| 1. map(tuple):   2. emit(tuple, "R") // add R tag if tuple comes from R   1. emit(tuple, "S") // add S tag if tuple comes from S |
| 1. reduce(tuple, tags): 2. // tags is the list of tag [‘R’,’S’] 3. if "R" in tags and "S" in tags: 4. emit(tuple, None) // tuple이 R과 S 모두에 존재하면 반환 |

3.Grouping γA,θ(B)(R). Given a relation R(A, B, C), with one grouping attribute A, one aggregated attribute B, and another attribute C, which is neither grouped or aggregated:

(a) Partition the tuples of R according to their values in attribute A.  
(b) For each group, aggregate the values in attribute B and apply function θ on the

aggregated value (θ is an aggregation operation such as SUM, COUNT or MAX).

The result of this operation is one tuple for each group. That tuple has a component for the grouping attribute A, with the value common to tuples of that group. It also has a component for each aggregation θ(B), with the aggregated value for that group.

|  |
| --- |
| a)  map(tuple):  // tuple: (A, B, C)  key = tuple.A  value = tuple.B  emit(key, value) |
| b)  reduce(key, value\_list):  aggregated\_value = sum(value\_list)  emit(key, aggregated\_value) |

Spark

1. Show the total number of movies in each genre with (a) the DataFrame API, (b) Spark SQL, and (c) RDD operations. Identify the most efficient method.

|  |
| --- |
| DataFrame API  A computer code on a black background  Description automatically generatedA screen shot of a computer  Description automatically generated |
| Spark SQL  A screen shot of a computer program  Description automatically generated  A screen shot of a movie list  Description automatically generated |
| RDD |

1. Find the directors who directed the highest number of movies.

|  |
| --- |
| A computer code on a black background  Description automatically generatedA screen shot of a computer  Description automatically generated |

3.Determine the genres with the highest average IMDb rating (use the imdb.rating field).

|  |
| --- |
| A computer screen shot of text  Description automatically generatedA screenshot of a black screen  Description automatically generated |

4.Find the month with the most movie releases based on the released date.

|  |
| --- |
| A screen shot of a computer  Description automatically generatedA black background with white text  Description automatically generated |

1. Identify the top 5 movies with the longest runtime in each genre.

|  |
| --- |
| A computer screen shot of a program code  Description automatically generatedA screen shot of a computer  Description automatically generated |

1. Find the top 10 actors who appeared in the most movies. For each actor, list the number of movies and their average IMDb rating.

|  |
| --- |
| A screen shot of a computer code  Description automatically generated |

7.Plot a bar chart with the number of movies released each year.

|  |
| --- |
| A computer screen with text on it  Description automatically generatedA graph of a number of movies released  Description automatically generated |