DS 5110: Introduction to Data Management and Processing (Fall 2024) Dr. Roi Yehoshua
Student name: (Due) November 11, 2024

PS5: MongoDB and Big Data

1 MongoDB (50%)

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):

   (a) Display all the restaurants located in the boroughs Bronx or Brooklyn.

   (b) Find the restaurant id, name, borough and cuisine for those restaurants whose name starts with the letters 'Mad'.

   (c) Find the restaurants that have received a score between 80 and 90 (inclusive).

   (d) Display the restaurant id and name of restaurants which have received a 'C' grade in year 2014.

   (e) Find the cuisine that has the highest number of restaurants.

   (f) 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.

   (g) For each restaurant display only the grades that were recorded from the year 2014 onwards.

   (h) Calculate the average score across all the restaurants in the collection.

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

```
{ "_id": 1, "city": "Berkeley", "state": "CA", "qty": 648 },
{ "_id": 2, "city": "Bend", "state": "OR", "qty": 491 },
{ "_id": 3, "city": "Kensington", "state": "CA", "qty": 233 },
{ "_id": 4, "city": "Eugene", "state": "OR", "qty": 842 },
{ "_id": 5, "city": "Reno", "state": "NV", "qty": 655 },
{ "_id": 6, "city": "Portland", "state": "OR", "qty": 408 },
{ "_id": 7, "city": "Sacramento", "state": "CA", "qty": 574 }
```

   (a) Find the total quantity of sales for each state and sort the quantities in descending order. The result should look as follows:

```
[
  { state: 'OR', total_qty: 1741 },
  { state: 'CA', total_qty: 1455 },
  { state: 'NV', total_qty: 655 }
]
```

   (b) Perform the following update operations on the sales collection:

i. Change the quantity of the sales in Berkeley from 648 to 750.
ii. Increase the quantity of all the sales in Oregon by 50.
iii. Add to sales no. 5 the names of the salespeople: David and Martha.
iv. Add James to the list of salespeople of sales no. 5.
v. Replace Martha with Lisa in the salesperson list of sales no. 5.
vi. Delete all the sales from California.

(c) Print the final sales collection.

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 sample run:

```
Enter category: Computer Graphics

ISBN: 1884777902, title: 3D User Interfaces with Java 3D
ISBN: 133034054, title: Graphics File Formats
ISBN: 1933988398, title: Gnuplot in Action
ISBN: 1884777473, title: The Awesome Power of Direct3D/DirectX
ISBN: 138412146, title: Power-3D
ISBN: 1930110022, title: Graphics Programming with Perl
```

2 MapReduce (20%)

Describe how to implement the following relational operations using MapReduce. Write the map and reduce functions in pseudocode.

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

2. Intersection $R \cap S$: 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).

3. Grouping $\gamma_{A,\theta(B)}(R)$. 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 $\theta(B)$, with the aggregated value for that group.

3 Spark (30%)

This exercise uses a dataset of movies released worldwide, focusing on various aspects of film production and performance. Load the `movies.json` file into Spark and answer the following questions:

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.
2. Find the directors who directed the highest number of movies.
3. Determine the genres with the highest average IMDb rating (use the `imdb.rating` field).
4. Find the month with the most movie releases based on the `released` date.
5. Identify the top 5 movies with the longest runtime in each genre.
6. Find the top 10 actors who appeared in the most movies. For each actor, list the number of movies and their average IMDb rating.
7. Plot a bar chart with the number of movies released each year.