**Step 1: Create the Database and Collection**

1. **Switch to the library database**:

Code:

use library

1. **Create and insert documents into the books collection**:

Code:

db.books.insertMany([

{ title: "Book One", pages: 150 },

{ title: "Book Two", pages: 300 },

{ title: "Book Three", pages: 220 },

{ title: "Book Four", pages: 400 },

{ title: "Book Five", pages: 100 }

// Add more books as needed

])

**Step 2: Use MapReduce to Categorize Books as "Small" or "Big"**

1. **Map function**: This function classifies books as either "Small" or "Big" based on the number of pages and emits the category with a value of 1.

Code:

var mapFunction = function() {

var category = this.pages <= 250 ? "Small" : "Big";

emit(category, 1);

};

1. **Reduce function**: This function sums up the values for each category.

Code:

var reduceFunction = function(key, values) {

return Array.sum(values);

};

1. **Run MapReduce**:

Code:

db.books.mapReduce(

mapFunction,

reduceFunction,

{ out: "book\_size\_count" }

)

1. **Retrieve the results**:

Code:

db.book\_size\_count.find().pretty()

* + **Explanation**: This MapReduce operation counts how many books are classified as "Small" (250 pages or less) and "Big" (more than 250 pages), and stores the results in the book\_size\_count collection.

**Summary**

1. **Database**: Created library database and books collection.
2. **MapReduce Functionality**:
   * **Map Function**: Categorized books based on page count.
   * **Reduce Function**: Used Array.sum to count books in each category.
3. **Results**: The counts of "Small" and "Big" books are stored in the book\_size\_count collection and can be easily queried.

This approach leverages MongoDB’s MapReduce feature to perform classification and aggregation based on the number of pages.