// Step 1: Use the appropriate database

use RetailCompany;

// Step 2: Create the customer collection and insert sample data

db.customer.drop(); // Drop the collection if it exists for a fresh start

// Insert sample data into the customer collection

db.customer.insertMany([

{ cid: 1, cname: "Alice", amount: 100, product\_name: "Laptop" },

{ cid: 2, cname: "Bob", amount: 150, product\_name: "Smartphone" },

{ cid: 1, cname: "Alice", amount: 200, product\_name: "Tablet" },

{ cid: 3, cname: "Charlie", amount: 300, product\_name: "Headphones" },

{ cid: 2, cname: "Bob", amount: 50, product\_name: "Charger" },

{ cid: 4, cname: "David", amount: 400, product\_name: "Monitor" },

{ cid: 1, cname: "Alice", amount: 50, product\_name: "Mouse" }

]);

// Step 3: Implement MapReduce to summarize total amount spent by each customer

// Define the Map function

var mapFunction = function() {

emit(this.cid, this.amount);

};

// Define the Reduce function

var reduceFunction = function(keyCid, values) {

return Array.sum(values);

};

// Perform the MapReduce operation

var result = db.customer.mapReduce(

mapFunction,

reduceFunction,

{ out: "customer\_totals" } // Output collection for results

);

// Print the results

print("Total Amount Spent by Each Customer:");

db.customer\_totals.find().pretty();

**Explanation of the Code:**

1. **Database Selection**: The code starts by selecting (or creating) the RetailCompany database.
2. **Collection Creation and Data Insertion**:
   * The customer collection is dropped if it exists to ensure a clean start.
   * Sample data representing customer purchases is inserted into the customer collection using insertMany().
3. **MapReduce Implementation**:
   * **Map Function**: This function emits the customer ID (cid) as the key and the amount spent (amount) as the value.
   * **Reduce Function**: This function takes the emitted key (keyCid) and an array of values, summing them to get the total amount spent by that customer.
   * **MapReduce Execution**: The mapReduce() method is called on the customer collection, specifying the map and reduce functions, and indicating that the results should be stored in a new collection called customer\_totals.
4. **Result Output**: Finally, the code retrieves and prints the summarized results from the customer\_totals collection.