**1. Find Products in a Price Range**

****

**Java Code:**

package connection;

import org.bson.Document;

import com.mongodb.client.FindIterable;

import com.mongodb.client.MongoClient;

import com.mongodb.client.MongoClients;

import com.mongodb.client.MongoCollection;

import com.mongodb.client.MongoDatabase;

public class collection\_two {

public static void main(String[] args) {

// Connect to MongoDB server

MongoClient mongoClient = MongoClients.create("mongodb://localhost:27017");

// Access the 'myDb' database and 'products' collection

MongoDatabase database = mongoClient.getDatabase("myDb");

MongoCollection<Document> collection = database.getCollection("products");

// Query for products with price between 700 and 900

Document priceRangeQuery = new Document("price", new Document("$gte", 700).append("$lte", 900));

FindIterable<Document> products = collection.find(priceRangeQuery);

// Print results

for (Document product : products) {

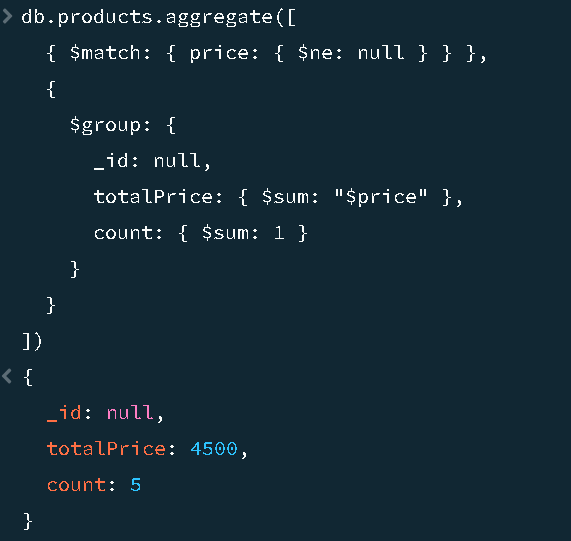
System.out.println(product.toJson());

}

}

}

**2. Calculate Total Price and Count of Products**

****

**Java Code:**

package connection;

import java.util.Arrays;

import org.bson.Document;

import com.mongodb.client.AggregateIterable;

import com.mongodb.client.MongoClient;

import com.mongodb.client.MongoClients;

import com.mongodb.client.MongoCollection;

import com.mongodb.client.MongoDatabase;

public class collection\_two {

public static void main(String[] args) {

// Connect to MongoDB

MongoClient mongoClient = MongoClients.create("mongodb://localhost:27017");

// Access 'myDb' and 'products' collection

MongoDatabase database = mongoClient.getDatabase("myDb");

MongoCollection<Document> collection = database.getCollection("products");

// Aggregate total price and count of products where price is not null

AggregateIterable<Document> result = collection.aggregate(Arrays.asList(

new Document("$match", new Document("price", new Document("$ne", null))),

new Document("$group", new Document("\_id", null)

.append("totalPrice", new Document("$sum", "$price"))

.append("count", new Document("$sum", 1)))

));

// Print results

for (Document doc : result) {

System.out.println("Total Price: " + doc.get("totalPrice"));

System.out.println("Count: " + doc.get("count"));

}

}

}

**3. Get Document with Highest Price for Each Item**

****

**Java Code:**

package connection;

import java.util.Arrays;

import org.bson.Document;

import com.mongodb.client.AggregateIterable;

import com.mongodb.client.MongoClient;

import com.mongodb.client.MongoClients;

import com.mongodb.client.MongoCollection;

import com.mongodb.client.MongoDatabase;

public class collection\_two {

public static void main(String[] args) {

// Connect to MongoDB

MongoClient mongoClient = MongoClients.create("mongodb://localhost:27017");

// Access 'myDb' and 'sales' collection

MongoDatabase database = mongoClient.getDatabase("myDb");

MongoCollection<Document> collection = database.getCollection("sales");

// Aggregate: For each item, get the document with the highest price

AggregateIterable<Document> result = collection.aggregate(Arrays.asList(

new Document("$sort", new Document("item", 1).append("price", -1)),

new Document("$group", new Document("\_id", "$item")

.append("maxPrice", new Document("$first", "$price"))

.append("document", new Document("$first", "$$ROOT"))),

new Document("$replaceRoot", new Document("newRoot", "$document"))

));

// Print each top-priced document

for (Document doc : result) {

System.out.println(doc.toJson());

}

}

}