OPERATORS

**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** FindCondition {

**public** **static** **void** main(String[] args) {

// Create a connection to the MongoDB server running locally

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

// Connect to the database named "myDb"

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

// Access the collection named "sampleCollection"

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

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

// Retrieve documents with a limit of 5 using FindIterable

FindIterable<Document> iterable = collection.find(filter);

// Iterate over the results and print each document

**for** (Document doc : iterable) {

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

}

// int index=1;

// for (Document doc : iterable) {

// if (index % 2 == 0) {

// System.out.println("Remaining Document: " + doc);

// }

// index++;

// }

// Close the MongoDB client connection

mongoClient.close();

}

}

AGGREGATION

**package** connection;

**import** java.util.\*;

**import** org.bson.Document;

**import** com.mongodb.client.\*;

**public** **class** FindCondition {

**public** **static** **void** main(String[] args) {

// Create a connection to the MongoDB server running locally

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

// Connect to the database named "myDb"

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

// Access the collection named "sampleCollection"

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

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

//

// // Retrieve documents with a limit of 5 using FindIterable

// FindIterable<Document> iterable = collection.find(filter);

// Define the aggregation pipeline

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

**new** Document("$group", **new** Document("\_id", "$category") // Group by category

.append("avgPrice", **new** Document("$avg", "$price"))) // Compute average price

));

// Iterate over the results and print each document

**for** (Document doc : result) {

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

}

// int index=1;

// for (Document doc : iterable) {

// if (index % 2 == 0) {

// System.out.println("Remaining Document: " + doc);

// }

// index++;

// }

// Close the MongoDB client connection

mongoClient.close();

}

}

**package** connection;

**import** java.util.Arrays;

**import** org.bson.Document;

**import** com.mongodb.client.\*;

**public** **class** ComplexGrouping {

**public** **static** **void** main(String[] args) {

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

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

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

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

**new** Document("$match", **new** Document("price", **new** Document("$gte", 0))),

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

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

.append("sizeWithMaxPrice", **new** Document("$first", "$size"))

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

));

**for** (Document doc : result) {

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

}

mongoClient.close();

}

}