import com.mongodb.client.\*;

import com.mongodb.client.result.DeleteResult;

import com.mongodb.client.result.UpdateResult;

import com.mongodb.client.model.Filters;

import com.mongodb.client.model.Updates;

import com.mongodb.client.model.Sorts;

import org.bson.Document;

public class SalesCrud {

private static MongoCollection<Document> collection;

public static void main(String[] args) {

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

collection = mongoClient

.getDatabase("operators")

.getCollection("sales");

deleteCappuccinos();

updateAmericanosQuantity(2);

markBulkOrders(20);

printSortedByPriceDesc();

printSortedByDateAsc();

printSkipLimit(1, 3);

}

}

// Deletes all documents where item is "Cappuccino"

private static void deleteCappuccinos() {

DeleteResult result = collection.deleteMany(Filters.eq("item", "Cappuccino"));

System.out.println("deleteMany -> deleted count: " + result.getDeletedCount());

}

// Increments quantity where item is "Americanos"

private static void updateAmericanosQuantity(int increment) {

UpdateResult result = collection.updateMany(

Filters.eq("item", "Americanos"),

Updates.inc("quantity", increment)

);

System.out.println("updateMany -> matched: " + result.getMatchedCount() +

", modified: " + result.getModifiedCount());

}

private static void markBulkOrders(int threshold) {

UpdateResult result = collection.updateMany(

Filters.gt("quantity", threshold),

Updates.set("bulkOrder", true)

);

System.out.println("markBulkOrders -> matched: " + result.getMatchedCount() +

", modified: " + result.getModifiedCount());

}

//descending order

private static void printSortedByPriceDesc() {

System.out.println("\nsort descending by price:");

collection.find()

.sort(Sorts.descending("price"))

.forEach(doc -> System.out.println(doc.toJson()));

}

//ascending order

private static void printSortedByDateAsc() {

System.out.println("\nsort ascending by date:");

collection.find()

.sort(Sorts.ascending("date"))

.forEach(doc -> System.out.println(doc.toJson()));

}

// Skips documents

private static void printSkipLimit(int skipCount, int limitCount) {

System.out.println("\nskip " + skipCount + ", limit " + limitCount + ":");

collection.find()

.sort(Sorts.ascending("\_id"))

.skip(skipCount)

.limit(limitCount)

.forEach(doc -> System.out.println(doc.toJson()));

}

}