Big data Management

Assignment 7

Jeyadev L

G23AI2071

Load Data

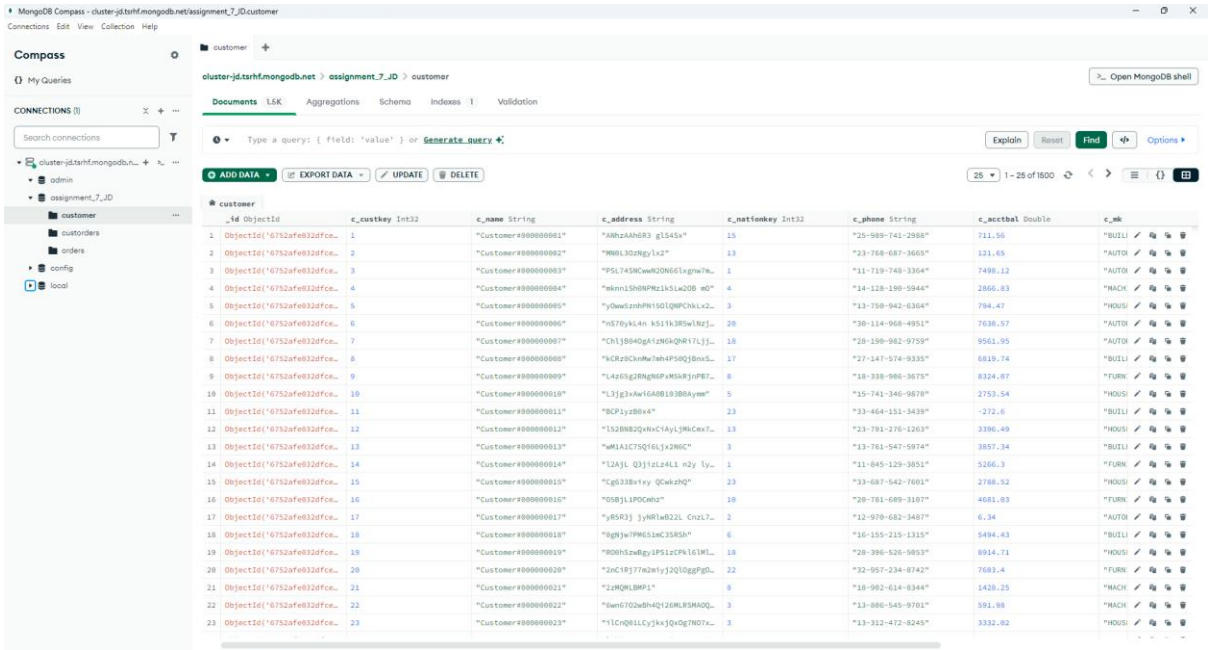
```java
    /**
     * Ensures the collections exist and loads data into them.
     *
     * @param database MongoDB database instance
     */
    public static void createAndLoadCollections(MongoDatabase database) {
        // File paths
        String customerFilePath = "data/customer.tbl";
        String ordersFilePath = "data/order.tbl";

        // Ensure collections are created
        MongoCollection<Document> customerCollection = database.getCollection(collectionName:"customer");
        MongoCollection<Document> ordersCollection = database.getCollection(collectionName:"orders");

        // Load data into collections
        System.out.println("Loading customer data...");
        loadPipeDelimitedData(customerFilePath, customerCollection, new String[]{
                "c_custkey", "c_name", "c_address", "c_nationkey", "c_phone", "c_acctbal", "c_mktsegment", "c_comment"
        });

        System.out.println("Loading orders data...");
        loadPipeDelimitedData(ordersFilePath, ordersCollection, new String[]{
                "o_orderkey", "o_custkey", "o_orderstatus", "o_totalprice", "o_orderdate",
                "o_orderpriority", "o_clerk", "o_shippriority", "o_comment"
        });
        loadNest(database,customerCollection,ordersCollection);
    }
```

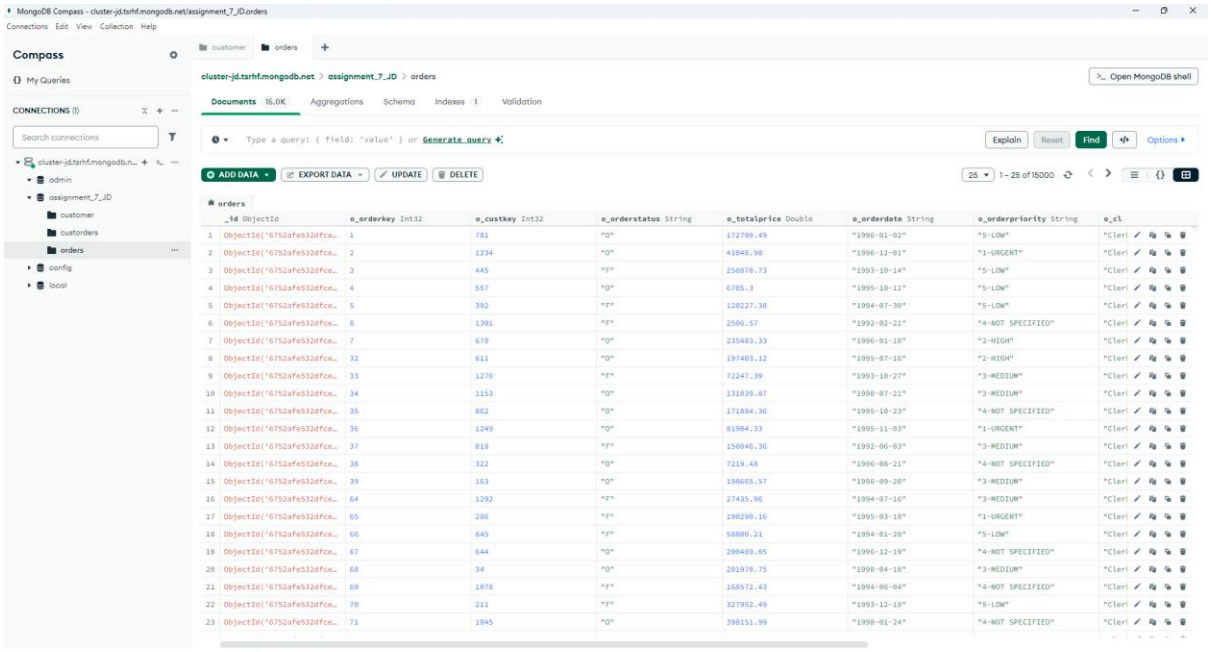
```java
    /**
     * Loads pipe-delimited data from a file into a MongoDB collection.
     *
     * @param filePath   Path to the pipe-delimited file
     * @param collection MongoDB collection to load data into
     * @param headers    Array of field names corresponding to the file structure
     */
    private static void loadPipeDelimitedData(String filePath, MongoCollection<Document> collection, String[] headers) {
        try (BufferedReader br = new BufferedReader(new FileReader(filePath))) {
            String line;
            List<Document> documents = new ArrayList<>();

            while ((line = br.readLine()) != null) {
                String[] values = line.split("\\|"); // Pipe-delimited
                Document document = new Document();
                for (int i = 0; i < headers.length; i++) {
                    document.append(headers[i], parseValue(values[i]));
                }
                documents.add(document);

                // Insert in batches of 1000 for performance
                if (documents.size() >= 1000) {
                    collection.insertMany(documents);
                    documents.clear();
                }
            }

            // Insert remaining documents
            if (!documents.isEmpty()) {
                collection.insertMany(documents);
            }

            System.out.println("Loaded data into collection: " + collection.getNamespace().getCollectionName());
        } catch (Exception e) {
            System.err.println("Error loading data into collection: " + e.getMessage());
            e.printStackTrace();
        }
    }
```

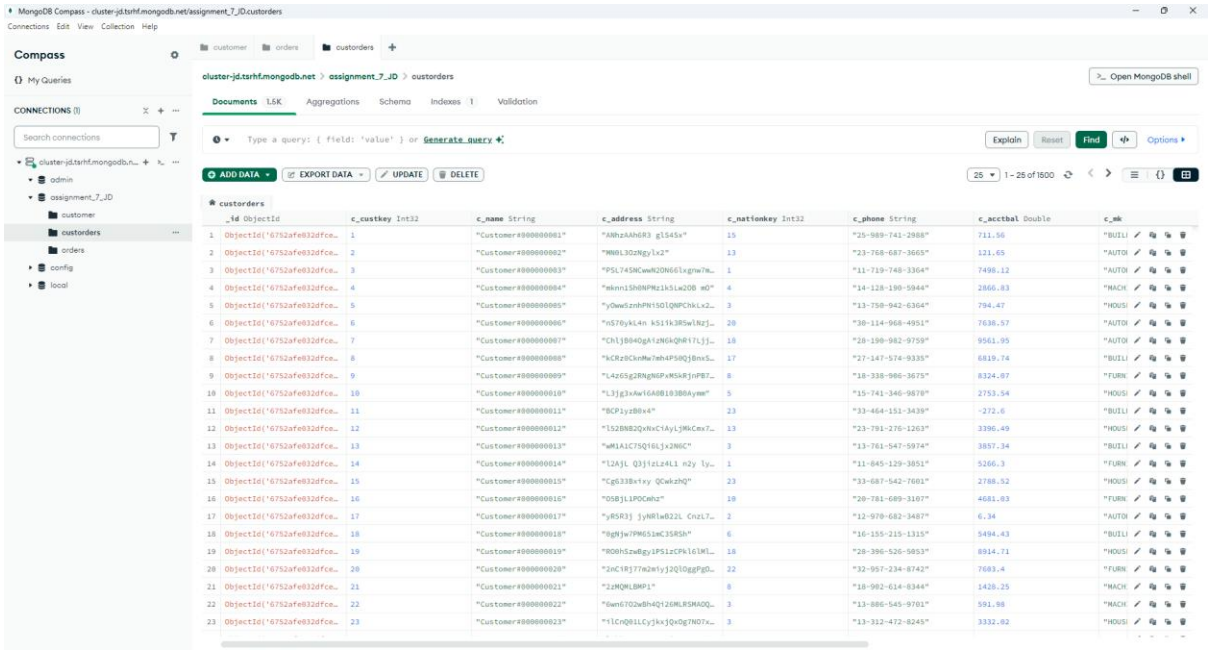
```java
/**
 * Loads customer and orders data into a nested collection 'custorders'.
 *
 * @param database MongoDB database instance
 */
public static void loadNest(MongoDatabase database,MongoCollection<Document> customerCollection,MongoCollection<Document> ordersCollection) {


    // Target collection
    MongoCollection<Document> custOrdersCollection = database.getCollection(collectionName:"custorders");

    // Clear the target collection if it exists
    custOrdersCollection.drop();

    System.out.println("Creating nested collection 'custorders'...");

    // Iterate through each customer
    try (MongoCursor<Document> customerCursor = customerCollection.find().iterator()) {
        List<Document> nestedDocuments = new ArrayList<>();
        while (customerCursor.hasNext()) {
            Document customer = customerCursor.next();
            int customerId = customer.getInteger(key:"c_custkey");

            // Find all orders for the current customer
            List<Document> orders = ordersCollection.find(new Document(key:"o_custkey", customerId)).into(new ArrayList<>());

            // Embed the orders in the customer document
            customer.append(key:"orders", orders);

            // Add the nested document to the list
            nestedDocuments.add(customer);

            // Insert in batches for performance
            if (nestedDocuments.size() >= 1000) {
                custOrdersCollection.insertMany(nestedDocuments);
                nestedDocuments.clear();
            }
        }

        // Insert remaining nested documents
        if (!nestedDocuments.isEmpty()) {
            custOrdersCollection.insertMany(nestedDocuments);
        }

        System.out.println("Data loaded into 'custorders' collection.");
    } catch (Exception e) {
        System.err.println("Error loading nested data: " + e.getMessage());
        e.printStackTrace();
    }
}
```

```java
/**
 * Parses a string value to handle nulls or type conversions.
 *
 * @param value Raw string value
 * @return Parsed object (String, Integer, Double, or null)
 */
private static Object parseValue(String value) {
    if (value == null || value.isEmpty()) {
        return null;
    }
    try {
        if (value.contains(".")) {
            return Double.parseDouble(value);
        } else {
            return Integer.parseInt(value);
        }
    } catch (NumberFormatException e) {
        return value.trim(); // Default to String
    }
}
```

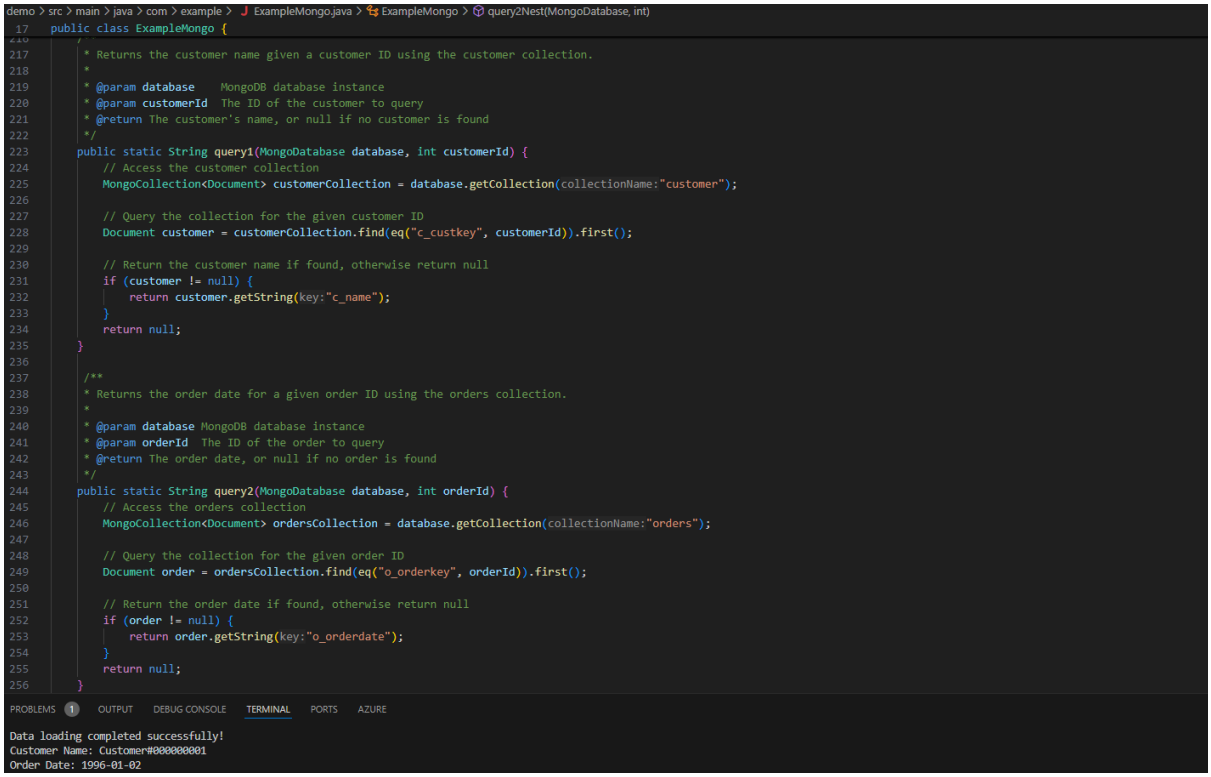
```
Loaded data into collection: customer
Loading orders data...
Loaded data into collection: orders
Creating nested collection 'custorders'...
Data loaded into 'custorders' collection.
```




MongoDB Compass - cluster-jd.tsrhf.mongodb.net/assignment_7_JD.orders
Connections Edit View Collection Help
Compass
My Queries
CONNECTIONS (1)
Search connections
cluster-jd.tsrhf.mongodb.n...
admin
assignment_7_JD
customer
custorders
orders
config
local
customer
orders
cluster-jd.tsrhf.mongodb.net > assignment_7_JD > orders
Open MongoDB shell
Documents 15.0K
Aggregations
Schema
Indexes 1
Validation
Type a query: { field: 'value' } or Generate query
Explain
Reset
Find
Options
ADD DATA
EXPORT DATA
UPDATE
DELETE
25
1 - 25 of 15000
orders
_id ObjectId
o_orderkey Int32
o_custkey Int32
o_orderstatus String
o_totalprice Double
o_orderdate String
o_orderpriority String
o_cl

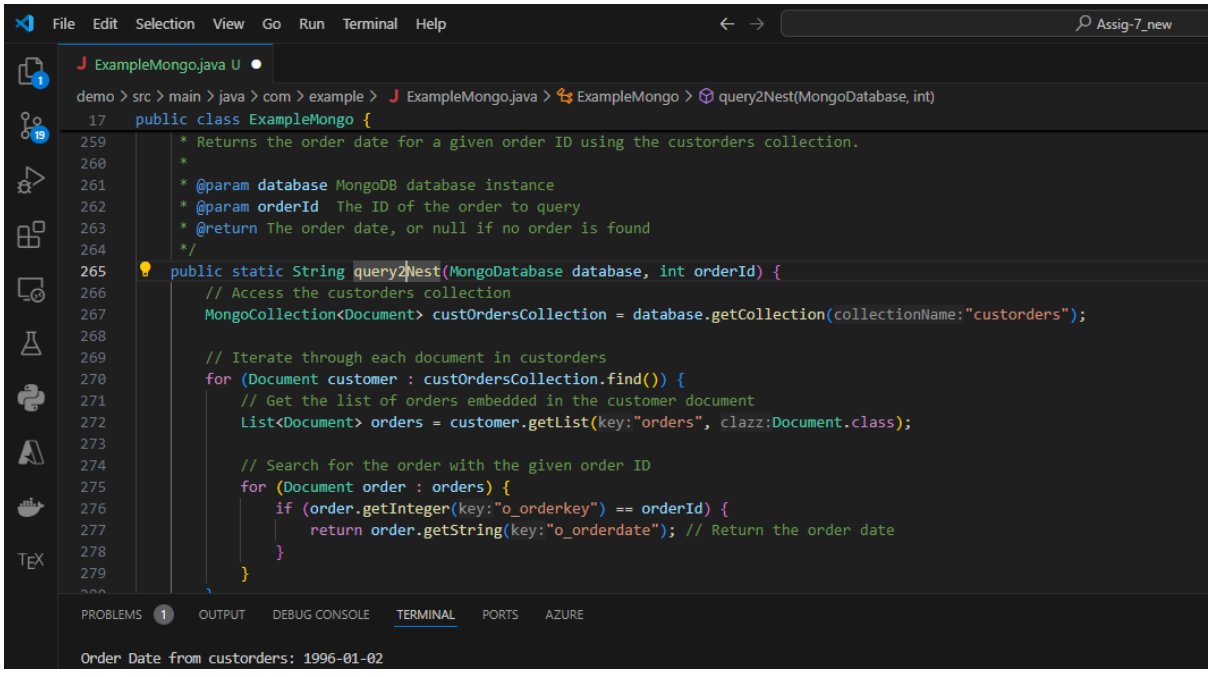

MongoDB Compass - cluster-jd.tsrhf.mongodb.net/assignment_7_JD.custorders
Connections Edit View Collection Help
Compass
My Queries
CONNECTIONS (1)
Search connections
cluster-jd.tsrhf.mongodb.n...
admin
assignment_7_JD
customer
custorders
orders
config
local
customer
orders
custorders
cluster-jd.tsrhf.mongodb.net > assignment_7_JD > custorders
Open MongoDB shell
Documents 1.5K
Aggregations
Schema
Indexes 1
Validation
Type a query: { field: 'value' } or Generate query
Explain
Reset
Find
Options
ADD DATA
EXPORT DATA
UPDATE
DELETE
25
1 - 25 of 1500
custorders
_id ObjectId
c_custkey Int32
c_name String
c_address String
c_nationkey Int32
c_phone String
c_acctbal Double
c_mk

Query 1 and 2



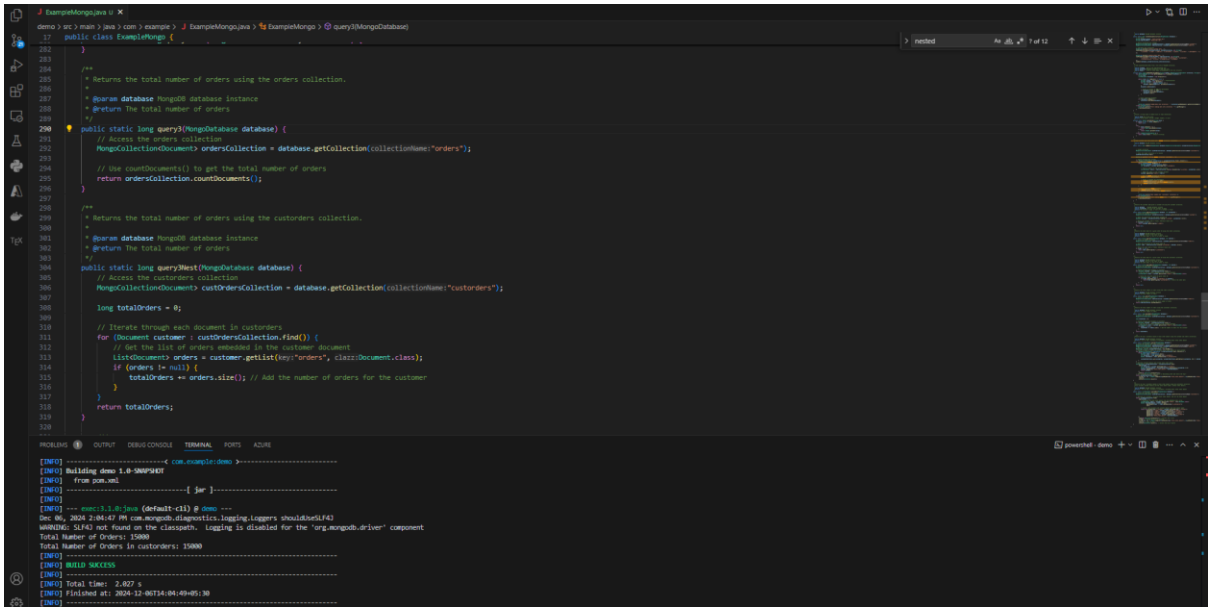
```
demo > src > main > java > com > example > ExampleMongo.java > ExampleMongo > query2Nest(MongoDatabase, int)
17  public class ExampleMongo {
217     * Returns the customer name given a customer ID using the customer collection.
218     *
219     * @param database    MongoDB database instance
220     * @param customerId  The ID of the customer to query
221     * @return The customer's name, or null if no customer is found
222     */
223    public static String query1(MongoDatabase database, int customerId) {
224        // Access the customer collection
225        MongoCollection<Document> customerCollection = database.getCollection(collectionName:"customer");
226
227        // Query the collection for the given customer ID
228        Document customer = customerCollection.find(eq("c_custkey", customerId)).first();
229
230        // Return the customer name if found, otherwise return null
231        if (customer != null) {
232            return customer.getString(key:"c_name");
233        }
234        return null;
235    }
236
237    /**
238     * Returns the order date for a given order ID using the orders collection.
239     *
240     * @param database MongoDB database instance
241     * @param orderId  The ID of the order to query
242     * @return The order date, or null if no order is found
243     */
244    public static String query2(MongoDatabase database, int orderId) {
245        // Access the orders collection
246        MongoCollection<Document> ordersCollection = database.getCollection(collectionName:"orders");
247
248        // Query the collection for the given order ID
249        Document order = ordersCollection.find(eq("o_orderkey", orderId)).first();
250
251        // Return the order date if found, otherwise return null
252        if (order != null) {
253            return order.getString(key:"o_orderdate");
254        }
255        return null;
256    }

PROBLEMS 1  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  AZURE

Data loading completed successfully!
Customer Name: Customer#000000001
Order Date: 1996-01-02
```

Query2Nest

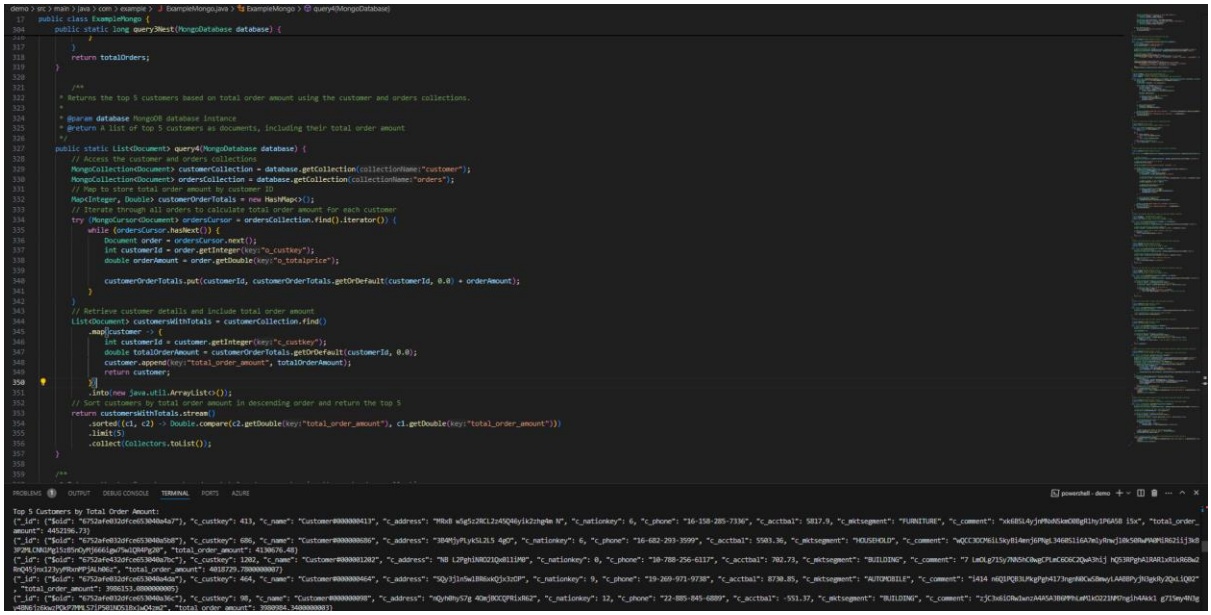


```
File  Edit  Selection  View  Go  Run  Terminal  Help        Assig-7_new

ExampleMongo.java U

demo > src > main > java > com > example > ExampleMongo.java > ExampleMongo > query2Nest(MongoDatabase, int)
17  public class ExampleMongo {
259     * Returns the order date for a given order ID using the custorders collection.
260     *
261     * @param database MongoDB database instance
262     * @param orderId  The ID of the order to query
263     * @return The order date, or null if no order is found
264     */
265    public static String query2Nest(MongoDatabase database, int orderId) {
266        // Access the custorders collection
267        MongoCollection<Document> custOrdersCollection = database.getCollection(collectionName:"custorders");
268
269        // Iterate through each document in custorders
270        for (Document customer : custOrdersCollection.find()) {
271            // Get the list of orders embedded in the customer document
272            List<Document> orders = customer.getList(key:"orders", clazz:Document.class);
273
274            // Search for the order with the given order ID
275            for (Document order : orders) {
276                if (order.getInteger(key:"o_orderkey") == orderId) {
277                    return order.getString(key:"o_orderdate"); // Return the order date
278                }
279            }

PROBLEMS 1  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  AZURE

Order Date from custorders: 1996-01-02
```

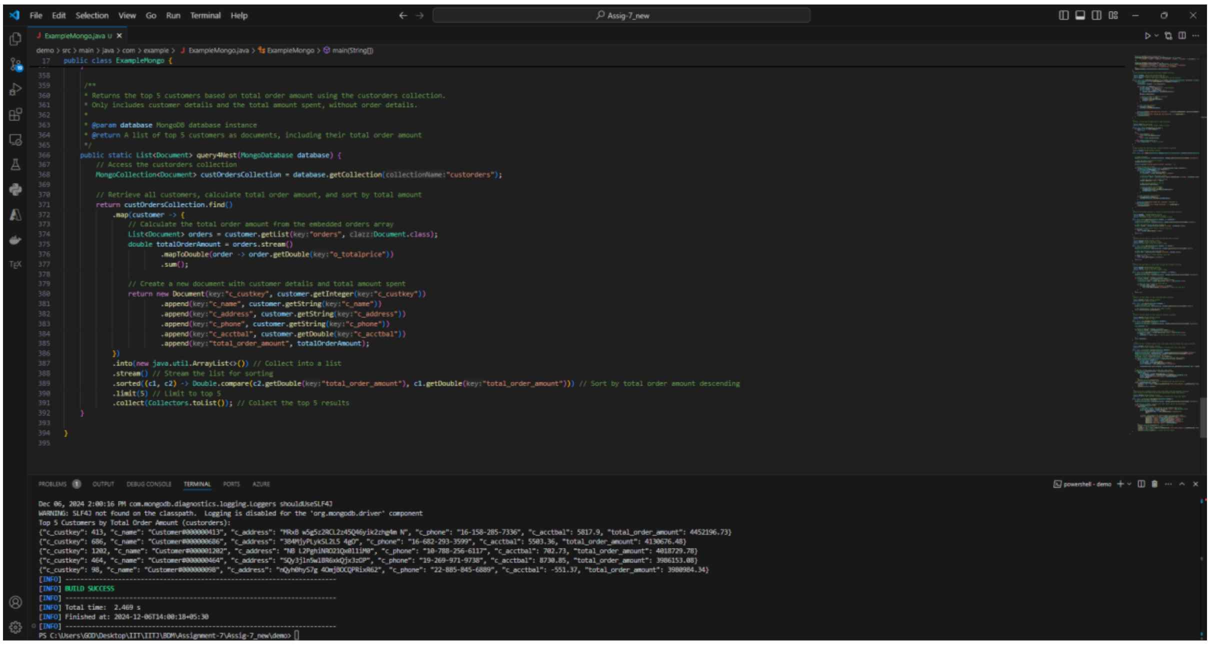
Query 3 and Query3Nest



Query 4



Query4Nest


File Edit Selection View Go Run Terminal Help
Assig-7_new
ExampleMongo.java U
demo > src > main > java > com > example > ExampleMongo.java > ExampleMongo > main(String[])
public class ExampleMongo {
/**
* Returns the top 5 customers based on total order amount using the custorders collection.
* Only includes customer details and the total amount spent, without order details.
*
* @param database MongoDB database instance
* @return A list of top 5 customers as documents, including their total order amount
*/
public static List<Document> query4Nest(MongoDatabase database) {
// Access the custorders collection
MongoCollection<Document> custOrdersCollection = database.getCollection(collectionName:"custorders");
// Retrieve all customers, calculate total order amount, and sort by total amount
return custOrdersCollection.find()
.map(customer -> {
// Calculate the total order amount from the embedded orders array
List<Document> orders = customer.getList(key:"orders", clazz:Document.class);
double totalOrderAmount = orders.stream()
.mapToDouble(order -> order.getDouble(key:"o_totalprice"))
.sum();
// Create a new document with customer details and total amount spent
return new Document(key:"c_custkey", customer.getInteger(key:"c_custkey"))
.append(key:"c_name", customer.getString(key:"c_name"))
.append(key:"c_address", customer.getString(key:"c_address"))
.append(key:"c_phone", customer.getString(key:"c_phone"))
.append(key:"c_acctbal", customer.getDouble(key:"c_acctbal"))
.append(key:"total_order_amount", totalOrderAmount);
})
.into(new java.util.ArrayList<>()) // Collect into a list
.stream() // Stream the list for sorting
.sorted((c1, c2) -> Double.compare(c2.getDouble(key:"total_order_amount"), c1.getDouble(key:"total_order_amount"))) // Sort by total order amount descending
.limit(5) // Limit to top 5
.collect(Collectors.toList()); // Collect the top 5 results
}
}
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS AZURE
Dec 06, 2024 2:00:16 PM com.mongodb.diagnostics.logging.Loggers shouldUseSLF4J
WARNING: SLF4J not found on the classpath. Logging is disabled for the 'org.mongodb.driver' component
Top 5 Customers by Total Order Amount (custorders):
{"c_custkey": 413, "c_name": "Customer#000000413", "c_address": "MRxB w5g5z2RCL2z45Q46yik2zhg4m N", "c_phone": "16-158-285-7336", "c_acctbal": 5817.9, "total_order_amount": 4452196.73}
{"c_custkey": 686, "c_name": "Customer#000000686", "c_address": "384MjyPLykSL2L5 4gO", "c_phone": "16-682-293-3599", "c_acctbal": 5503.36, "total_order_amount": 4130676.48}
{"c_custkey": 1202, "c_name": "Customer#000001202", "c_address": "NB L2PghiNRO21Qx011iM8", "c_phone": "10-788-256-6117", "c_acctbal": 702.73, "total_order_amount": 4018729.78}
{"c_custkey": 464, "c_name": "Customer#000000464", "c_address": "SQy3j1n5w18R6xkQjx3zOP", "c_phone": "19-269-971-9738", "c_acctbal": 8730.85, "total_order_amount": 3986153.08}
{"c_custkey": 98, "c_name": "Customer#000000098", "c_address": "nQyh0hyS7g 40mjBOCQPRixR62", "c_phone": "22-885-845-6889", "c_acctbal": -551.37, "total_order_amount": 3980984.34}
[INFO] --
[INFO] BUILD SUCCESS
[INFO] --
[INFO] Total time: 2.469 s
[INFO] Finished at: 2024-12-06T14:00:18+05:30
[INFO] --
PS C:\Users\GOD\Desktop\IIT\IITJ\BDM\Assignment-7\Assig-7_new\demo>