```
package mongodbTest;

import com.mongodb.BasicDBObject;
import com.mongodb.BasicDBObjectBuilder;
import com.mongodb.BulkWriteOperation;
import com.mongodb.BulkWriteResult;
import com.mongodb.AggregationOptions;
import com.mongodb.AggregationOutput;
import com.mongodb.Cursor;
import com.mongodb.MongoClient;
import com.mongodb.DBCollection;
import com.mongodb.DBCursor;
import com.mongodb.DBObject;
import com.mongodb.Mongo;
import com.mongodb.DB;
import com.mongodb.ParallelScanOptions;

import java.net.UnknownHostException;
import java.util.Arrays;
import java.util.List;
import java.util.Set;

import static java.util.concurrent.TimeUnit.SECONDS;

public class MongodbTesting {
      public static void main(final String[] args) throws
UnknownHostException {

            //connect to local database server
            MongoClient mongoClient = new MongoClient();

            //get handle to "sample_database"
            DB db = mongoClient.getDB("sample_database");

            //get a list of the collections in this database and print
them out
            Set<String> collectionNames = db.getCollectionNames();
            System.out.print("\nList of Databases\n");
            for (final String s : collectionNames) {
                  System.out.println(s);
            }

            //get a collection object to work with
            DBCollection col1 = db.getCollection("my_collection");

            //drop all the data in it
            col1.drop();

            //Create a document having a field as Sub Document
            BasicDBObject doc = new
BasicDBObject("name","MongoDB").append("type",
"database").append("count", 1).
                        append("info", new
BasicDBObject("x",203).append("y", 102));

            //Insert the newly created Document
            col1.insert(doc);
```

```
            //Display the newly created Document
            DBObject myDoc = col1.findOne();
            System.out.print("\nNewly Inserted Document\n");
            System.out.println(myDoc);

            //Add more documents to the collection
            for(int i = 2; i < 100; i++) {
                  col1.insert(new BasicDBObject().append("name",
"MongoDB").append("type", "database").append("count", i));
            }

            System.out.println("Total # of documents after inserting 100
small ones  " + col1.getCount());

            DBCursor cursor = col1.find();
            try{
                  while(cursor.hasNext()){
                        System.out.println(cursor.next());
                  }
            }finally{
                  cursor.close();
            }

            //now let's use a query and get one document out
            BasicDBObject query = new BasicDBObject("count",71);
            cursor = col1.find(query);

            try{
                  System.out.print("\nDocument fetched from the query\n");
                  while(cursor.hasNext()){
                        System.out.println(cursor.next());
                  }
            }finally{
                  cursor.close();
            }

            //Using Comparison Operators
            query = new BasicDBObject("count", new
BasicDBObject("$lt",9));

            cursor = col1.find(query);
            try{
                  System.out.print("\nDocuments with count less than
9\n");
                  while(cursor.hasNext()){
                        System.out.println(cursor.next());
                  }
            }finally{
                  cursor.close();
            }

            //Using multiple operators in one query
            query = new BasicDBObject("count",new
BasicDBObject("$gt",20).append("$lt", 30));
            cursor = col1.find(query);

            try{
```

```
                System.out.print("\nDocument fetched using multiple
operators\n");
                while(cursor.hasNext()){
                      System.out.println(cursor.next());
                }
          }finally{
                cursor.close();
          }

          //Drop a Database
          mongoClient.dropDatabase("newdb");

          System.out.print("\nList of Databases\n");
          for (final String s : collectionNames) {
                System.out.println(s);
          }

          //Aggregation Operation

          //Add some sample data
          DBCollection col2 = db.getCollection("aggregationExample");
          col2.insert(new BasicDBObjectBuilder()
          .add("employee",1)
          .add("department","sales")
          .add("amount", 71)
          .add("type", "airfare")
          .get());
          col2.insert(new BasicDBObjectBuilder()
          .add("employee",2)
          .add("department","Engineering")
          .add("amount", 15)
          .add("type", "airfare")
          .get());
          col2.insert(new BasicDBObjectBuilder()
          .add("employee",4)
          .add("department","Human Resources")
          .add("amount", 5)
          .add("type", "airfare")
          .get());
          col2.insert(new BasicDBObjectBuilder()
          .add("employee",42)
          .add("department","sales")
          .add("amount", 71)
          .add("type", "airfare")
          .get());

          //create our pipeline operations. First with the $match
          DBObject match = new BasicDBObject("$match",new
BasicDBObject("type","airfare"));

          //build the $projection Operation
          DBObject fields = new BasicDBObject("department",1);
          fields.put("amount", 1);
          fields.put("_id", 0);
          DBObject project = new BasicDBObject("$project",fields);

          //Now the $group operation
```

```
            DBObject groupFields = new
BasicDBObject("_id","$department");
            groupFields.put("average", new
BasicDBObject("$avg","$amount"));
            DBObject group = new BasicDBObject("$group",groupFields);

            //Finally the $sort operation
            DBObject sort = new BasicDBObject("$sort",new
BasicDBObject("average",-1));

            //Run Aggregation
            List<DBObject> pipeline = Arrays.asList(match, project,
group, sort);
            AggregationOutput output = col2.aggregate(pipeline);

            //Output the results
            for (DBObject result : output.results()) {
                  System.out.println(result);
            }
      }

}
```