2010

Polytechnic of Namibia

Student Name: Veiko Muronga

Student Number: 9864431

[MONGODB AND CASSANDRA APPLICATIONS]

1. Use MongoDB to support the modeling and design of the database system for a small application. Use Cassandra to create a message management system.

MONGODB

1. Introduction

   The blogging application was written using Java to carry out the required tasks:

   - View all the posts written by a given author.
   - View all the tags attached to a post
   - View all the comments readers submitted to a post
   - Add new tags to a post
   - Add new comments to a post

2. Dataset

   The blog_dbms collection/dataset was created as follows:

   > blog1 = "blog-id" : 1, "author" : { "lastname" : "muronga", "firstname" : "veiko", "address" : { "street" :"luderitz", "city" : "windhoek", "PO Box" : "1234" } },, "post" : { "title" : "MongoDB-API", "text" : "There were some difficulties faced with these API", "date" : "Sun Jun 13 2010 12:51:55 GMT+0100 (Namibia Daylight Time)" }, "tags" : ["mongodb", "DBMS", "java driver"] },

```
 "comments" : [
      {
           "commentator1" : {
                "firstname" : "loyd",
                "lastname" : "franz",
                "address" : "118,long island,WHK"
           },
           "comment" : "just include the driver jar in your class path"
      },
      {
           "commentator2" : {
                "firstname" : "duran",
                "lastname" : "izacks",
                "address" : "34,sesfontein,OKJ"
           },
           "comment" : "Do you mean the class path in the environmental variables?"
      },
      {
           "commentator3" : {
                "firstname" : "mika",
                "lastname" : "daniel",
                "address" : "49,mugabe road,WHK"
           },
```

```
            "comment" : "How does this concern me?"
        }
]
>db.blog_dbms.save(blog1);

>blog2 = "blog-id" : 2, "author" : { "lastname" : "muronga", "firstname" : "veiko", "address" : { "street" : "luderitz", "city" : "windhoek", "PO Box" : "1234" } },  "post" : { "title" : "MongoDB Libraries", "text" : "the Jar files should be included in the library package", "date" : "Sun Jun 13 2010 12:58:50 GMT+0100 (Namibia Daylight Time)" }, "tags" : [ "mongo jar", "jline jar", "junit jar", "DBObjects" ] },
 "comments" : [
        {
            "commentator1" : {
                "firstname" : "loyd",
                "lastname" : "franz",
                "address" : "118,long island,WHK"
            },
            "comment" : "you are right i have tested it"
        },
        {
            "commentator2" : {
                "firstname" : "duran",
                "lastname" : "izacks",
                "address" : "34,sesfontein,OKJ"
            },
            "comment" : "thank you guys you saved my life"
        }
]

>db.blog_dbms.save(blog2);

> blog3 = "blog-id" : 3, "author" : { "lastname" : "immanuel", "firstname" : "kino", "address" : { "street" : "sam nujoma", "city" : "tsumeb", "PO Box" : "9876" } }, "post" : { "title" : "MongoDB classes", "text" : "include all the required mongodb classes", "date" : "Sun Jun 13 2010 13:07:11 GMT+0100 (Namibia Daylight Time)" }, "tags" : [ "DBCollection", "BasicDBObject", "DBCursor" ],
"comments" : [
        {
            "commentator1" : {
                "firstname" : "loyd",
                "lastname" : "franz",
                "address" : "118,long island,WHK"
            },
```

```
            "comment" : "my code also required Mongo class"
        },
        {
            "commentator2" : {
                "firstname" : "duran",
                "lastname" : "izacks",
                "address" : "34,sesfontein,OKJ"
            },
            "comment" : "mine also required DB class"
        }
] }

>db.blog_dbms.save(blog3);

> blog 4 = "blog-id" : 4, "author" : { "lastname" : "nino", "firstname" : "papi", "address" : {
"street" : "sam nujoma", "city" : "tsumeb", "PO Box" : "9876" } }, "post" : { "title" : "MongoDB
classes","text" : "include all the required mongodb classes", "date" : "Sun Jun 13 2010 14:28:57
GMT+0100 (Namibia Daylight Time)" }, "tags" : [ "DBCollection", "BasicDBObject", "DBCursor" ],
"comments" : [
        {
            "commentator1" : {
                "firstname" : "loyd",
                "lastname" : "franz",
                "address" : "118,long island,WHK"
            },
            "comment" : "my code also required Mongo class"
        },
        {
            "commentator2" : {
                "firstname" : "duran",
                "lastname" : "izacks",
                "address" : "34,sesfontein,OKJ"
            },
            "comment" : "mine also required DB class"
        }
] }
```

3. Source Code
   //The source code written using java used to query the database is as follows

```
package myproject;

import com.mongodb.*;

import java.util.*;

/**
 *
 * @author murongav
 */
public class PostQuery {
   public static void main(String []args)throws Exception {

   // connect to the local database server
      Mongo m = new Mongo();

  // switch to the database that you would like to use
      DB db = m.getDB ("MyProject");

  // Get a list of collections in this database and print them out

      Set<String> colls = db.getCollectionNames();
     for (String s : colls){
         System.out.println(s);
      }

      //get the "blog" collection to work with
      DBCollection coll = db.getCollection("blog_dbms");

//1. View all the posts written by a given author
   BasicDBObject  query = new BasicDBObject();
   query.put("author.firstname","veiko");
   DBCursor cur = coll.find(query);
   while (cur.hasNext()){
    System.out.println(cur.next());
   }

 //2. View all the tags attached to a post
      query = new BasicDBObject();
      query.put("blog-id",1);
      cur =  coll.find(query);
      DBObject db_obj=null;
```

```
        while (cur.hasNext()){
        db_obj=cur.next();
        System.out.println("post"+db_obj.get("post"));
        System.out.println("tags"+db_obj.get("tags"));

         }

//3. View all the comments readers submitted to a post
        query = new BasicDBObject();
        query.put("blog-id",1);
        cur =  coll.find(query);
        db_obj=null;
        while (cur.hasNext()){
        db_obj=cur.next();
        System.out.println("post"+db_obj.get("post"));
        System.out.println("comments"+db_obj.get("comments"));

         }

//4. Add new comments to a post
//get the object Id which was generated by the system

      DBObject blog1 = coll.findOne();
      System.out.println(blog1.get("_id"));


      DBObject blogid = BasicDBObjectBuilder.start()
          .add("4c14c691cd480000000033f2",blog1.get("4c14c691cd480000000033f2")).get();
       DBObject comment = BasicDBObjectBuilder.start()
          .push("commentator4")
          .append("firstname","danny")
          .append("lastname","greco")
          .append("address","78,sando road,WHK")
          .pop()
          .append("comment","Everyone is affected by MongoDB").get();

        DBObject addComment = new BasicDBObject("$push",new
BasicDBObject("comments",comment));
        coll.update(blogid, addComment);


         //5. add new tags to a post
```

```
        DBObject blogid1 = BasicDBObjectBuilder.start()
        .add("4c14c691cd480000000033f2",blog1.get("4c14c691cd480000000033f2")).get();
        DBObject addTag = new BasicDBObject("$push",new BasicDBObject("tags","DBList"));
        coll.update(blogid1, addTag);

    }

}
```

4. Conclusion

   All the required functionalities of the project were covered by the source code. There had been cases were different object ID types were used to refer to a certain entry. This was done to extend the functionality of the source code. Case one made use of pre-assigned ID's that were entered during the writing (part two and three). Case two made use of system generated Object ID's which can be requested from the database first and then inserted in the code to refer to a specific blog entry. This has been a wonderful experience as a first timer to the java platform.

CASSANDRA

The main personal contribution to this part of the project is making use of different code that is provided on the internet for other applications, to try and come up with a unique application. This was a straight forward design in a sense that it made use of one Keyspace, one columnfamily and different columns to represent the different data. The application was written in java.

The one barrier that can be avoided in the future is that when I was trying to carry out the **git add and git push** I was receiving errors. The commands were executing only after I have deactivated my "Proxy Server".