（答案）

实验 4

第五章 NoSQL 和关系数据库的操作比较

宋 曼

目录

实验 4

第五章 NoSQL 和关系数据库的操作比较

1 实验目的

1. 理解四种数据库(MySQL,HBase,Redis,MongoDB)的概念以及不同点;
2. 熟练使用四种数据库操作常用的 Shell 命令;
3. 熟悉四种数据库操作常用的 Java API。

2 实验平台

操作系统：Linux
Hadoop 版本：2.6.0 或以上版本
MySQL 版本:5.6 或以上版本
HBase 版本：1.1.2 或以上版本
Redis 版本: 3.0.6 或以上版本
MongoDB 版本: 3.2.6 或以上版本
JDK 版本：1.7 或以上版本
Java IDE：Eclipse

3 实验内容和要求

3.1 MySQL 数据库操作

Student 学生表

Name	English	Math	Computer
zhangsan	69	86	77
lisi	55	100	88

1. 根据上面给出的表格,利用 MySQL5.6 设计出 student 学生表格;

解答: 创建上述表格的 sql 语句为:

```
create table student(
    name varchar(30) not null,
    English tinyint unsigned not null,
    Math tinyint unsigned not null,
    Computer tinyint unsigned not null
    );
```

插入两条记录 sql 语句为:

```
insert into student values("zhangsan",69,86,77);
insert into student values("lisi",55,100,88);
```

a) 设计完后,用 select 语句输出所有的相关信息,并给出截图;

解答: sql 语句: select * from student;

```
mysql> select * from student;
+----------+---------+------+----------+
| name     | English | Math | Computer |
+----------+---------+------+----------+
| zhangsan |      69 |   86 |       77 |
| lisi     |      55 |  100 |       88 |
+----------+---------+------+----------+
2 rows in set (0.00 sec)
```

b) 查询 zhangsan 的 Computer 成绩,并给出截图;

解答:sql 语句为:select name , Computer from student where name = "zhangsan";

```
mysql> select name , Computer from student where name = "zhangsan";
+----------+----------+
| name     | Computer |
+----------+----------+
| zhangsan |       77 |
+----------+----------+
1 row in set (0.02 sec)
```

c) 修改 lisi 的 Math 成绩,改为 95.给出截图.

解答: sql 语句为: update student set Math=95 where name="lisi";

```
mysql> update student set Math=95 where name="lisi";
Query OK, 0 rows affected (0.05 sec)
Rows matched: 1  Changed: 0  Warnings: 0

mysql> select name,Math from student where name = "lisi";
+------+------+
| name | Math |
+------+------+
| lisi |   95 |
+------+------+
1 row in set (0.00 sec)
```

2.根据上面已经设计出的 student 表,用 MySQL 的 JAVA 客户端编程;

附: jdbc 下载地址:http://downloads.mysql.com/archives/c-j/

a) 添加数据:English:45 Math:89 Computer:100

scofield	45	89	100

```
import java.sql.*;
public class mysql_test {

 /**
  * @param args
  */
 //JDBC DRIVER and DB
 static final String  DRIVER="com.mysql.jdbc.Driver";
 static final String DB="jdbc:mysql://localhost/test";
 //Database auth
```

```
static final String USER="root";
static final String PASSWD="root";

public static void main(String[] args) {
  // TODO Auto-generated method stub
  Connection conn=null;
  Statement stmt=null;
  try {
    //加载驱动程序
    Class.forName(DRIVER);
    System.out.println("Connecting to a selected database...");
    //打开一个连接
    conn=DriverManager.getConnection(DB, USER, PASSWD);
    //执行一个查询
    stmt=conn.createStatement();
    String sql="insert into student values('scofield',45,89,100)";
    stmt.executeUpdate(sql);
    System.out.println("Inserting records into the table successfully!");
  } catch (ClassNotFoundException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
  }catch (SQLException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
  }finally
  {
    if(stmt!=null)
      try {
        stmt.close();
      } catch (SQLException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
      }
    if(conn!=null)
      try {
        conn.close();
      } catch (SQLException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
      }
  }
}
      }
```

Eclipse 控制台输出如下:

```
Connecting to a selected database...
Inserting records into the table successfully!
```

插入数据之后,MySQL 客户度查询结果如下:

```
mysql> select * from student;
+----------+---------+------+----------+
| name     | English | Math | Computer |
+----------+---------+------+----------+
| zhangsan |      69 |   86 |       77 |
| lisi     |      55 |   95 |       88 |
| scofield |      45 |   89 |      100 |
+----------+---------+------+----------+
3 rows in set (0.00 sec)
```

b) 获取 scofield 的 English 成绩信息

```
import java.sql.*;
public class mysql_qurty {

    /**
     * @param args
     */
    //JDBC DRIVER and DB
    static final String  DRIVER="com.mysql.jdbc.Driver";
    static final String DB="jdbc:mysql://localhost/test";
    //Database auth
    static final String USER="root";
    static final String PASSWD="root";

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        Connection conn=null;
        Statement stmt=null;
        ResultSet rs=null;
        try {
            //加载驱动程序
            Class.forName(DRIVER);
            System.out.println("Connecting to a selected database...");
            //打开一个连接
            conn=DriverManager.getConnection(DB, USER, PASSWD);
            //执行一个查询
            stmt=conn.createStatement();
            String sql="select name,English from student where name='scofield' ";
            //获得结果集
            rs=stmt.executeQuery(sql);
            System.out.println("name"+"\t\t"+"English");
```

```
            while(rs.next())
            {
                System.out.print(rs.getString(1)+"\t\t");
                System.out.println(rs.getInt(2));
            }
        } catch (ClassNotFoundException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }catch (SQLException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }finally
        {
            if(rs!=null)
                try {
                    rs.close();
                } catch (SQLException e1) {
                    // TODO Auto-generated catch block
                    e1.printStackTrace();
                }
            if(stmt!=null)
                try {
                    stmt.close();
                } catch (SQLException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
            if(conn!=null)
                try {
                    conn.close();
                } catch (SQLException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
        }
    }
}
```

Eclipse 控制台输出如下:

```
Connecting to a selected database...
name        English
scofield    45
```

3.2 HBase 数据库操作

Student 学生表

name	score		
	English	Math	Computer
zhangsan	69	86	77
lisi	55	100	88

1. 根据上面给出的表格，用 Hbase Shell 模式设计 student 学生表格。

解答：创建 Student 表格的命令为：create 'student','score'

插入上面表格数据命令为：

put 'student','zhangsan','score:English','69'

put 'student','zhangsan','score:Math','86'

put 'student','zhangsan','score:Computer','77'

put 'student','lisi','score:English','55'

put 'student','lisi','score:Math','100'

put 'student','lisi','score:Computer','88'

上述命令执行结果为：

```
hbase(main):018:0> create 'student','score'
0 row(s) in 2.2800 seconds

=> Hbase::Table - student
hbase(main):019:0> put 'student','zhangsan','score:English','69'
0 row(s) in 0.0240 seconds

hbase(main):020:0> put 'student','zhangsan','score:Math','86'
0 row(s) in 0.0070 seconds

hbase(main):021:0> put 'student','zhangsan','score:Computer','77'
0 row(s) in 0.0060 seconds

hbase(main):022:0> put 'student','lisi','score:English','55'
0 row(s) in 0.0100 seconds

hbase(main):023:0> put 'student','lisi','score:Math','100'
0 row(s) in 0.0080 seconds

hbase(main):024:0> put 'student','lisi','score:Computer','88'
0 row(s) in 0.0080 seconds
```

a) 设计完后，用 scan 指令浏览表的相关信息，给出截图。

解答：命令为： scan 'student'

```
hbase(main):001:0> scan 'student'
ROW                     COLUMN+CELL
 lisi                   column=score:Computer, timestamp=1462605149677, value=88
 lisi                   column=score:English, timestamp=1462605127827, value=55
 lisi                   column=score:Math, timestamp=1462605138004, value=100
 zhangsan               column=score:Computer, timestamp=1462605105787, value=77
 zhangsan               column=score:English, timestamp=1462605086516, value=69
 zhangsan               column=score:Math, timestamp=1462605096683, value=86
2 row(s) in 0.3410 seconds
```

b) 查询 zhangsan 的 Computer 成绩,给出截图。

解答：命令为：get 'student','zhangsan','score:Computer'

```
hbase(main):002:0> get 'student','zhangsan','score:Computer'
COLUMN                          CELL
 score:Computer                 timestamp=1462605105787, value=77
1 row(s) in 0.0610 seconds
```

c) 修改 lisi 的 Math 成绩，改为 95,给出截图。

解答：命令为 put 'student','lisi','score:Math','95'

```
hbase(main):003:0> put 'student','lisi','score:Math','95'
0 row(s) in 0.1020 seconds

hbase(main):004:0> get 'student','lisi','score:Math'
COLUMN                          CELL
 score:Math                     timestamp=1462605841339, value=95
1 row(s) in 0.0090 seconds
```

2. 根据上面已经设计出的 student，用 Hbase API 编程。

d) 添加数据：English:45 Math:89 Computer:100

scofield	45	89	100

附：HBase 不需要另外下载驱动，可直接使用 hbase/lib 下的 jar 包编程

```
import java.io.IOException;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.hbase.HBaseConfiguration;
import org.apache.hadoop.hbase.TableName;
import org.apache.hadoop.hbase.client.Admin;
import org.apache.hadoop.hbase.client.Connection;
import org.apache.hadoop.hbase.client.ConnectionFactory;
import org.apache.hadoop.hbase.client.Put;
import org.apache.hadoop.hbase.client.Table;


public class hbase_insert {

/**
 * @param args
 */
  public static Configuration configuration;
    public static Connection connection;
    public static Admin admin;
public static void main(String[] args) {
  // TODO Auto-generated method stub
   configuration  = HBaseConfiguration.create();
     configuration.set("hbase.rootdir","hdfs://localhost:9000/hbase");
```

```
        try{
            connection = ConnectionFactory.createConnection(configuration);
            admin = connection.getAdmin();
        }catch (IOException e){
            e.printStackTrace();
        }
        try {
        insertRow("student","scofield","score","English","45");
        insertRow("student","scofield","score","Math","89");
        insertRow("student","scofield","score","Computer","100");
    } catch (IOException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
        close();
}
 public static void insertRow(String tableName,String rowKey,String colFamily,
        String col,String val) throws IOException {
        Table table = connection.getTable(TableName.valueOf(tableName));
        Put put = new Put(rowKey.getBytes());
        put.addColumn(colFamily.getBytes(), col.getBytes(), val.getBytes());
        table.put(put);
        table.close();
    }
 public static void close(){
        try{
            if(admin != null){
                admin.close();
            }
            if(null != connection){
                connection.close();
            }
        }catch (IOException e){
            e.printStackTrace();
        }
    }
}
```

可以用 scan 输出数据库数据检验是否插入成功：

```
hbase(main):005:0> scan 'student'
ROW                     COLUMN+CELL
 lisi                   column=score:Computer, timestamp=1462605149677, value=88
 lisi                   column=score:English, timestamp=1462605127827, value=55
 lisi                   column=score:Math, timestamp=1462605841339, value=95
 scofield               column=score:Computer, timestamp=1462607153886, value=100
 scofield               column=score:English, timestamp=1462607153858, value=45
 scofield               column=score:Math, timestamp=1462607153883, value=89
 zhangsan               column=score:Computer, timestamp=1462605105787, value=77
 zhangsan               column=score:English, timestamp=1462605086516, value=69
 zhangsan               column=score:Math, timestamp=1462605096683, value=86
3 row(s) in 0.0390 seconds
```

e) 获取 scofield 的 English 成绩信息

```
import java.io.IOException;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.hbase.Cell;
import org.apache.hadoop.hbase.CellUtil;
import org.apache.hadoop.hbase.HBaseConfiguration;
import org.apache.hadoop.hbase.TableName;
import org.apache.hadoop.hbase.client.Admin;
import org.apache.hadoop.hbase.client.Connection;
import org.apache.hadoop.hbase.client.ConnectionFactory;
import org.apache.hadoop.hbase.client.Get;
import org.apache.hadoop.hbase.client.Put;
import org.apache.hadoop.hbase.client.Result;
import org.apache.hadoop.hbase.client.Table;


public class hbase_query {

/**
 * @param args
 */
  public static Configuration configuration;
    public static Connection connection;
    public static Admin admin;
public static void main(String[] args) {
  // TODO Auto-generated method stub
   configuration  = HBaseConfiguration.create();
     configuration.set("hbase.rootdir","hdfs://localhost:9000/hbase");
     try{
        connection = ConnectionFactory.createConnection(configuration);
        admin = connection.getAdmin();
     }catch (IOException e){
        e.printStackTrace();
     }
     try {
    getData("student","scofield","score","English");
```

```
        } catch (IOException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
            close();
    }
    public static void getData(String tableName,String rowKey,String colFamily,
            String col)throws  IOException{
            Table table = connection.getTable(TableName.valueOf(tableName));
            Get get = new Get(rowKey.getBytes());
            get.addColumn(colFamily.getBytes(),col.getBytes());
            Result result = table.get(get);
            showCell(result);
            table.close();
        }
    public static void showCell(Result result){
            Cell[] cells = result.rawCells();
            for(Cell cell:cells){
                System.out.println("RowName:"+new String(CellUtil.cloneRow(cell))+"
");
                System.out.println("Timetamp:"+cell.getTimestamp()+" ");
                System.out.println("column Family:"+new
String(CellUtil.cloneFamily(cell))+" ");
                System.out.println("row Name:"+new String(CellUtil.cloneQualifier(cell))
+" ");
                System.out.println("value:"+new String(CellUtil.cloneValue(cell))+" ");
            }
        }
    public static void close(){
            try{
                if(admin != null){
                    admin.close();
                }
                if(null != connection){
                    connection.close();
                }
            }catch (IOException e){
                e.printStackTrace();
            }
        }
}
```

Eclipse 控制台输出如下：

```
RowName:scofield
Timetamp:1462607153858
column Family:score
row Name:English
value:45
```

3.3 Redis 数据库操作

Student 键值对:

zhangsan: {
English: 69
Math: 86
Computer: 77
}
lisi: {
English: 55
Math: 100
Computer: 88
}

1. 根据上面给出的键值对，用 Redis 的哈希结构设计出上述表格;(键值可以用 student.zhangsan,student.lisi 来表示两个键值属于同一个表格)

解答:插入上述键值对命令为:

```
127.0.0.1:6379> hset student.zhangsan English 69
(integer) 1
127.0.0.1:6379> hset student.zhangsan Math 86
(integer) 1
127.0.0.1:6379> hset student.zhangsan Computer 77
(integer) 1

127.0.0.1:6379> hset student.lisi English 55
(integer) 1
127.0.0.1:6379> hset student.lisi Math 100
(integer) 1
127.0.0.1:6379> hset student.lisi Computer 88
(integer) 1
```

a) 设计完之后,用 hgetall 命令分别输出 zhangsan 和 lisi 的成绩信息,并截图;

解答:查询 zhangsan 和 lisi 成绩信息命令为:

hgetall student.zhangsan

```
127.0.0.1:6379> hgetall student.zhangsan
1) "English"
2) "69"
3) "Math"
4) "86"
5) "Computer"
6) "77"
```

hgetall student.lisi

```
127.0.0.1:6379> hgetall student.lisi
1) "English"
2) "55"
3) "Math"
4) "100"
5) "Computer"
6) "88"
```

b) 用 hget 命令查询 zhangsan 的 Computer 成绩,给出截图。

解答:命令为: hget student.zhangsan Computer

```
127.0.0.1:6379> hget student.zhangsan Computer
"77"
```

c) 修改 lisi 的 Math 成绩，改为 95,给出截图。

解答:命令为:hset student.lisi Math 95

```
127.0.0.1:6379> hset student.lisi Math 95
(integer) 0
```

2. 根据上面已经设计出的 student 表格,用 Redis 的 JAVA 客户端编程(jedis)

附:jedis 下载地址:https://github.com/xetorthio/jedis

a) 添加数据： English:45 Math:89 Computer:100

scofield: {

English: 45

Math: 89

Computer: 100

}

代码如下:

```
import java.util.Map;
import redis.clients.jedis.Jedis;

public class jedis_test {

 /**
  * @param args
  */
 public static void main(String[] args) {
   // TODO Auto-generated method stub
   Jedis jedis = new Jedis("localhost");
   jedis.hset("student.scofield", "English","45");
   jedis.hset("student.scofield", "Math","89");
   jedis.hset("student.scofield", "Computer","100");
   Map<String,String>  value = jedis.hgetAll("student.scofield");
   for(Map.Entry<String, String> entry:value.entrySet())
   {
     System.out.println(entry.getKey()+":"+entry.getValue());
   }
```

```
    }
    }
```

Eclipse 截图如下, 说明插入成功

```
Computer:100
Math:89
English:45
```

b) 获取 scofield 的 English 成绩信息

代码如下:

```
import java.util.Map;
import redis.clients.jedis.Jedis;

public class jedis_query {

 /**
  * @param args
  */
 public static void main(String[] args) {
   // TODO Auto-generated method stub
   Jedis jedis = new Jedis("localhost");
   String value=jedis.hget("student.scofield", "English");
   System.out.println("scofield's English score is:    "+value);
 }
}
```

Eclipse 控制台输出查询到的信息:

```
scofield's English score is:   45
```

3.4 MongoDB 数据库操作

Student 文档如下:

```
{
        “name”: “zhangsan”,
        “score”: {
            “English”: 69,
            “Math”: 86,
            “Computer”: 77
        }
}
{
        “name”: “lisi”,
        “score”: {
            “English”: 55,
```

“Math”: 100,
“Computer”: 88
}
}

1. 根据上面给出的文档,用 Mongo shell 设计出 student 集合.

解答:首先切换到 student 集合: use student

其次定义包含上述两个文档的数组:

var stus=[
{"name":"zhangsan","scores":{"English":69,"Math":86,"Computer":77}},
{"name":"lisi","score":{"English":55,"Math":100,"Computer":88}}]

最后调用 db.student.insert(stus)插入数据库;

执行结果如下:

```
> use student
switched to db student
> var stus=[
... {"name":"zhangsan","scores":{"English":69,"Math":86,"Computer":77}},
... {"name":"lisi","score":{"English":55,"Math":100,"Computer":88}}
... ]
> db.student.insert(stus)
BulkWriteResult({
        "writeErrors" : [ ],
        "writeConcernErrors" : [ ],
        "nInserted" : 2,
        "nUpserted" : 0,
        "nMatched" : 0,
        "nModified" : 0,
        "nRemoved" : 0,
        "upserted" : [ ]
})
```

a) 设计完后,用 find()方法输出两个学生的信息,给出截图;

解答:命令为: db.student.find().pretty()

```
> db.student.find().pretty()
{
        "_id" : ObjectId("572daa49ac079cb68a23068e"),
        "name" : "zhangsan",
        "scores" : {
                "English" : 69,
                "Math" : 86,
                "Computer" : 77
        }
}
{
        "_id" : ObjectId("572daa49ac079cb68a23068f"),
        "name" : "lisi",
        "score" : {
                "English" : 55,
                "Math" : 100,
                "Computer" : 88
        }
}
```

b) 用 find 函数查询 zhangsan 的所有成绩(只显示 score 列),给出截图。

解答:命令为 db.student.find({"name":"zhangsan"},{"_id":0,"name":0})

```
> db.student.find({"name":"zhangsan"},{"_id":0,"name":0})
{ "scores" : { "English" : 69, "Math" : 86, "Computer" : 77 } }
```

c) 修改 lisi 的 Math 成绩，改为 95,给出截图。

解答:命令为: db.student.update({"name":"lisi"}, {"$set":{"score.Math":95}})

```
> db.student.update({"name":"lisi"}, {"$set":{"score.Math":95}} )
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.student.find({"name":"lisi"},{"_id":0})
{ "name" : "lisi", "score" : { "English" : 55, "Math" : 95, "Computer" : 88 } }
```

2. 根据上面已经设计出的 student 集合,用 MongoDB 的 JAVA 客户端编程

附:MongoDB JAVA 驱动下载地址为:

http://central.maven.org/maven2/org/mongodb/mongo-java-driver/3.2.2/mongo-java-driver-3.2.2.jar

a) 添加数据:English:45 Math:89 Computer:100

```
{
        “name”: “scofield”,
        “score”: {
            “English”: 45,
            “Math”: 89,
            “Computer”: 100
        }
}
```

代码如下:

```
import java.util.ArrayList;
import java.util.List;

import org.bson.Document;
import com.mongodb.MongoClient;
import com.mongodb.client.MongoCollection;
import com.mongodb.client.MongoDatabase;

public class mongo_insert {

 /**
  * @param args
  */
 public static void main(String[] args) {
    // TODO Auto-generated method stub
    //实例化一个 mongo 客户端
    MongoClient  mongoClient=new MongoClient("localhost",27017);
    //实例化一个 mongo 数据库
    MongoDatabase mongoDatabase =
mongoClient.getDatabase("student");
```

```
        //获取数据库中某个集合
        MongoCollection<Document> collection =
    mongoDatabase.getCollection("student");
        //实例化一个文档,内嵌一个子文档
        Document document=new Document("name","scofield").
              append("score", new Document("English",45).
                    append("Math", 89).
                    append("Computer", 100));
        List<Document> documents = new ArrayList<Document>();
          documents.add(document);
          //将文档插入集合中
          collection.insertMany(documents);
          System.out.println("文档插入成功");
    }
```

通过以下截图,可以检测数据已经正确插入 MongoDB 数据库中

```
> db.student.find().pretty()
{
        "_id" : ObjectId("572daa49ac079cb68a23068e"),
        "name" : "zhangsan",
        "scores" : {
                "English" : 69,
                "Math" : 86,
                "Computer" : 77
        }
}
{
        "_id" : ObjectId("572daa49ac079cb68a23068f"),
        "name" : "lisi",
        "score" : {
                "English" : 55,
                "Math" : 95,
                "Computer" : 88
        }
}
{
        "_id" : ObjectId("572db5296381924c1aacc9e4"),
        "name" : "scofield",
        "score" : {
                "English" : 45,
                "Math" : 89,
                "Computer" : 100
        }
}
```

b) 获取 scofield 的所有成绩成绩信息(只显示 score 列)

代码如下;

```
import java.util.ArrayList;
import java.util.List;
```

```
import org.bson.Document;
import com.mongodb.MongoClient;
import com.mongodb.client.MongoCollection;
import com.mongodb.client.MongoCursor;
import com.mongodb.client.MongoDatabase;
import com.mongodb.client.model.Filters;
import static com.mongodb.client.model.Filters.eq;
public class mongo_query {

 /**
  * @param args
  */
 public static void main(String[] args) {
    // TODO Auto-generated method stub
    //实例化一个 mongo 客户端
    MongoClient  mongoClient=new MongoClient("localhost",27017);
    //实例化一个 mongo 数据库
    MongoDatabase mongoDatabase = mongoClient.getDatabase("student");
    //获取数据库中某个集合
    MongoCollection<Document> collection =
mongoDatabase.getCollection("student");
    //进行数据查找,查询条件为 name=scofield, 对获取的结果集只显示 score 这个域
    MongoCursor<Document>  cursor=collection.find( new
Document("name","scofield")).
          projection(new Document("score",1).append("_id", 0)).iterator();
    while(cursor.hasNext())
      System.out.println(cursor.next().toJson());
  }
  }
```

Eclipse 控制台可以输出查询得到的信息,红色区域为日志,可忽略