所有涉及到要修改配置文件的都是为了能在虚拟机外面访问才需要修改，如果没有要在外部进行访问的需求就可以不用修改配置。

mysql

数据库账号密码

root　root

修改数据库配置

vim /etc/mysql/mysql.conf.d/mysqld.cnf

找到bind-address

将其修改为0.0.0.0，如下图所示

```
 * Basic Settings

ser            = mysql
id-file        = /var/run/mysqld/mysqld.pid
ocket          = /var/run/mysqld/mysqld.sock
ort            = 3306
asedir         = /usr
atadir         = /var/lib/mysql
mpdir          = /tmp
c-messages-dir = /usr/share/mysql
kip-external-locking

 Instead of skip-networking the default is now to listen only on
 localhost which is more compatible and is not less secure.
bind-address           = 127.0.0.1
ind-address            = 0.0.0.0

 * Fine Tuning

ey_buffer_size         = 16M
ax_allowed_packet      = 16M
hread_stack            = 192K
hread_cache_size       = 8
 This replaces the startup script and checks MyISAM tables if needed
 the first time they are touched
yisam-recover-options  = BACKUP
max_connections        = 100
table_cache            = 64
thread_concurrency     = 10

 * Query Cache Configuration

uery_cache_limit       = 1M
```

以下参考仅为示例，个人要修改不同的插入数据以及相应的查询条件完成实验，最好带有个人色彩（名字学号等）的插入数据

然后执行service mysql restart重启mysql

```
#登陆数据库
mysql -uroot -p
输入密码，显示如下界面（密码为root）
```

```
root@ubuntu:~# mysql -uroot -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 3
Server version: 5.7.31-0ubuntu0.16.04.1 (Ubuntu)

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

执行

grant all privileges on . to 替换你的名字拼音@'%' identified by '1234';
flush privileges ;

创建一个数据库用户,后续操作都用该用户登录连接数据库完成。(账号为你的名字拼音 密码为1234，如xuhui 1234)

```
//mysql数据库查询数据示例
//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); //rs 为存储查询结果的数据结构
            System.out.println("name"+"\t\t"+"English");
            while(rs.next())
            {
                System.out.print(rs.getString(1)+"\t\t");// 获取结果集中第一列为文本
类型的数据
                System.out.println(rs.getInt(2));// 获取结果集中第二列为数字类型的数据
            }
```

插入数据示例

```
//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 sql1="insert into student values('scofield',45,89,100)";
            //String sql2="insert into student values('mazhihua',45,89,100)";
            stmt.executeUpdate(sql1);
            //stmt.executeUpdate(sql2);
```

```
            System.out.println("Inserting records into the table
successfully!");
```

```
#接着创建一个数据库
create database stu default charset=utf8;

#然后创建学生表
create table student(
    name varchar(30) not null,
    English tinyint unsigned not null,
    Math tinyint unsigned not null,
    Computer tinyint unsigned not null
    );
#出现Query Ok 字样代表每次操作成功

#接着插入两条学生数据
insert into student values("zhangsan",69,86,77);
insert into student values("lisi",99,100,100);

#查询zhangsan的计算机成绩
select name,Computer from student where name="zhangsan";

#修改lisi的数学成绩
update student set Math=95 where name="lisi";
```

```
package org.mysql;

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";
    static final String DB = "jdbc:mysql://192.168.51.130:3306/stu";
    //Database auth
    static final String USER="xuhui";
    static final String PASSWD="1234";
    public static void main(String[] args) {
// TODO Auto-generated method stub
        Connection conn=null;
        Statement stmt=null;
        ResultSet rs=null;
        try {
            //待填写代码
        } 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();
                }
        }
    }
}
```

```
package org.mysql;

import java.sql.*;
public class mysql_test {
    /**
     * @param args
     */

    //JDBC DRIVER and DB
    static final String DRIVER="com.mysql.cj.jdbc.Driver";
    //static final String DB="jdbc:mysql://localhost/test";
    static final String DB = "jdbc:mysql://192.168.51.130:3306/stu?
useSSL=false&useUnicode=true&characterEncoding=utf-
8&useLegacyDatetimeCode=false&serverTimezone=Asia/Shanghai";
    //Database auth
    static final String USER="xuhui";
    static final String PASSWD="1234";
    public static void main(String[] args) {
// TODO Auto-generated method stub
        Connection conn=null;
        Statement stmt=null;
        try {
        //待插入代码
        } 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();
                }
        }
    }
}
```

hbase操作

```
hbase(main):001:0> create 'student','score'
0 row(s) in 1.6450 seconds

=> Hbase::Table - student
hbase(main):002:0> put 'student','zhangsan','score:English','60'
0 row(s) in 0.1110 seconds

hbase(main):003:0> put 'student','zhangsan','score:Math','60'
0 row(s) in 0.0080 seconds

hbase(main):004:0> put 'student','zhangsan','score:Computer','60'
0 row(s) in 0.0090 seconds

hbase(main):005:0> get 'student','zhangsan','score:English'
```

```
package org.hbase;

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");
//        configuration.set("hbase.rootdir",
//                "hdfs://192.168.51.130:9000/opt");

 configuration.set("hbase.zookeeper.quorum","192.168.51.130:2181");//hbase 服务地
址
        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();
        }
    }
}
```

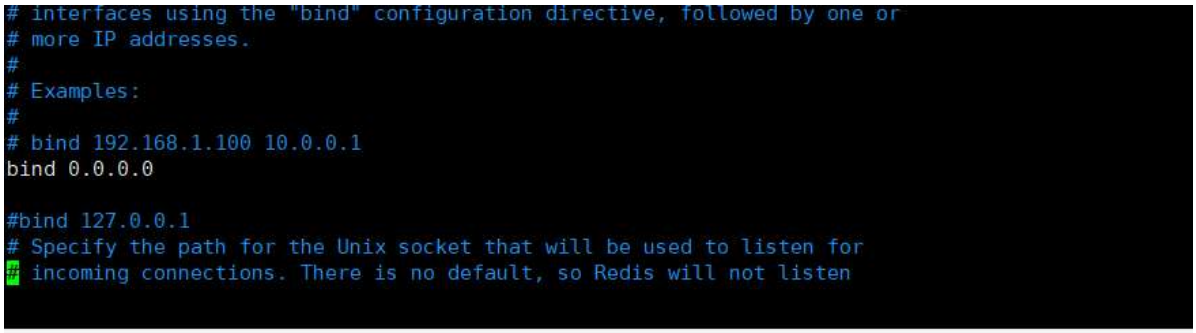
```
package org.hbase;

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");
//        configuration.set("hbase.rootdir",
//                "hdfs://192.168.51.130:9000/opt");

 configuration.set("hbase.zookeeper.quorum","192.168.51.130:2181");//hbase 服务地
址
        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();
        }
    }
}
```

redis

修改配置文件,这样你才能够通过主机访问你的虚拟机中的redis

vim /etc/redis/redis.conf, 找到bind 127.0.0.1这一行 修改为bind 0.0.0.0，如下图所示:



```
# interfaces using the "bind" configuration directive, followed by one or
# more IP addresses.
#
# Examples:
#
# bind 192.168.1.100 10.0.0.1
bind 0.0.0.0

#bind 127.0.0.1
# Specify the path for the Unix socket that will be used to listen for
# incoming connections. There is no default, so Redis will not listen
```

然后执行service redis restart 重启redis

然后输入 redis-cli进入redis命令界面进行相关操作。

Student 键值对:

zhangsan: {

English: 69

Math: 86

Computer: 77

}

lisi: {

English: 55

Math: 100

Computer: 88

}

```
#设置zhangsan的成绩
hset student.zhangsan English 60
hset student.zhangsan Math 70
hset student.zhangsan Computer 60

#获取zhangsan全部的成绩
hgetall student.zhangsan
#获取zhangsan英语成绩
hget student.zhangsan English
#修改zhangsan英语成绩
hset student.zhangsan English 55
```

mongodb

修改配置文件/etc/mongodb.conf

```
# mongodb.conf

# Where to store the data.
dbpath=/var/lib/mongodb

#where to log
logpath=/var/log/mongodb/mongodb.log

logappend=true

#bind_ip = 127.0.0.1
bind_ip = 0.0.0.0
#port = 27017

# Enable journaling, http://www.mongodb.org/display/DOCS/Journaling
journal=true

# Enables periodic logging of CPU utilization and I/O wait
#cpu = true

# Turn on/off security.  Off is currently the default
#noauth = true
#auth = true

# Verbose logging output.
#verbose = true

# Inspect all client data for validity on receipt (useful for
# developing drivers)
#objcheck = true

# Enable db quota management
#quota = true

# Set oplogging level where n is
"/etc/mongodb.conf" 101L, 2162C
```

将其由bind_ip = 127.0.0.1 修改为bind_ip=0.0.0.0

然后执行service mongodb restart 重启mongodb。

然后输入mongo进入mongodb命令行界面。

```
#指定要进入的数据库
use student
#插入学生数据
var stus=[{"name":"zhangsan","scores":{"English":70,"Math":60,"Computer":50}}]
db.student.insert(stus)

#查询所有学生信息
db.student.find().pretty()

#查询zhangsan信息
db.student.find({"name":"zhangsan"})

#修改zhangsan的英语成绩
db.student.update({"name":"zhangsan"},{"$set":{"scores.English":95}})
```

```
//插入数据示例
package org.mongodb;

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("192.168.51.130", 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");
    }
}
```

```java
//查询示例

package org.mongodb;

import org.bson.Document;
import com.mongodb.MongoClient;
import com.mongodb.client.MongoCollection;
import com.mongodb.client.MongoCursor;
import com.mongodb.client.MongoDatabase;
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("192.168.51.130",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());
    }
}
```