大数据数据初步分析

一、数据抓取：

（一）数据获取代码：

```
public class App
{
    public static void main( String[] args )
    {

        String str = "";
        for(int dtime=-245;dtime<-65;dtime++){
        java.text.SimpleDateFormat format = new java.text.SimpleDateFormat("yyyy-MM-
dd");
        Calendar cal = Calendar.getInstance();// 取当前日期。
        cal = Calendar.getInstance();
        cal.add(Calendar.DAY_OF_MONTH, dtime);// 取当前日期的前N天.
        str =format.format(cal.getTime());

        String res= GetCityList.weather("99", str);//
        JSONObject obj=JSONObject.fromObject(res);

       String result=obj.getString("result");
         //此时result中数据有多个key,可以对其key进行遍历,得到对个属性
        obj=JSONObject.fromObject(result);
         //今日温度对应的key是today
       String city_id=obj.getString("city_id");//城市地区ID
       String city_name=obj.getString("city_name");//城市地区名称
       String weather_date=obj.getString("weather_date");//天气日期
       String day_weather=obj.getString("day_weather");//   白天天气
       String night_weather=obj.getString("night_weather");//夜间天气
       String day_temp=obj.getString("day_temp");//白天最高温度
       String night_temp=obj.getString("night_temp");//夜间最低温度
       String day_wind=obj.getString("day_wind");// 白天风向
       String day_wind_comp=obj.getString("day_wind_comp");//白天风力
       String night_wind=obj.getString("night_wind"); // 夜间风向
        String night_wind_comp=obj.getString("night_wind_comp");// 夜间风力
        String day_weather_id=obj.getString("day_weather_id");//   白天天气标识
        String night_weather_id=obj.getString("night_weather_id");//       夜间天气标识

       List<String> list = new LinkedList<String>();
         list.add(city_id);
         list.add(city_name);
         list.add(weather_date);
```

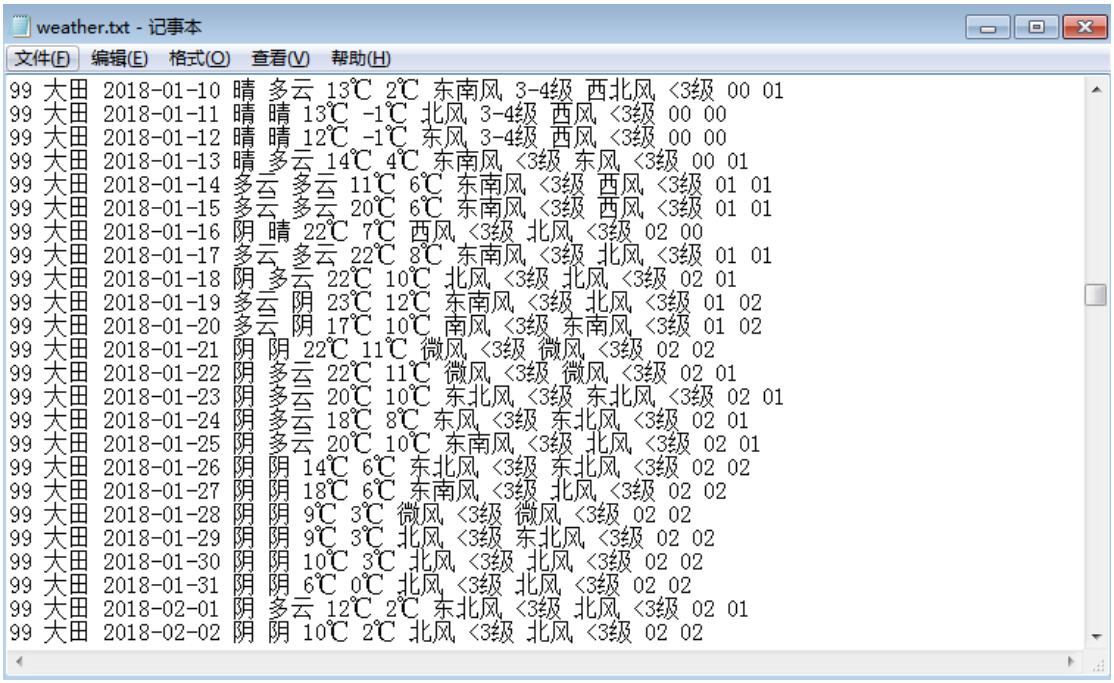
```
            list.add(day_weather);
            list.add(night_weather);
            list.add(day_temp);
            list.add(night_temp);
            list.add(day_wind);
            list.add(day_wind_comp);
            list.add(night_wind);
            list.add(night_wind_comp);
            list.add(day_weather_id);
            list.add(night_weather_id);

        File file1 = new File("C:\\weather.txt");
            try {
                FileWriter fw = new FileWriter(file1,true);
                BufferedWriter bw = new BufferedWriter(fw);

                for(int i = 0; i<list.size();i++){
                    bw.write(list.get(i).toString()+" ");
                    bw.flush();
                    //System.out.println(list.size());
                }
                bw.newLine();
                bw.close();
                fw.close();

            } catch (IOException e) {
                e.printStackTrace();
            }
    }
        }
}
```

（二）获取的数据：



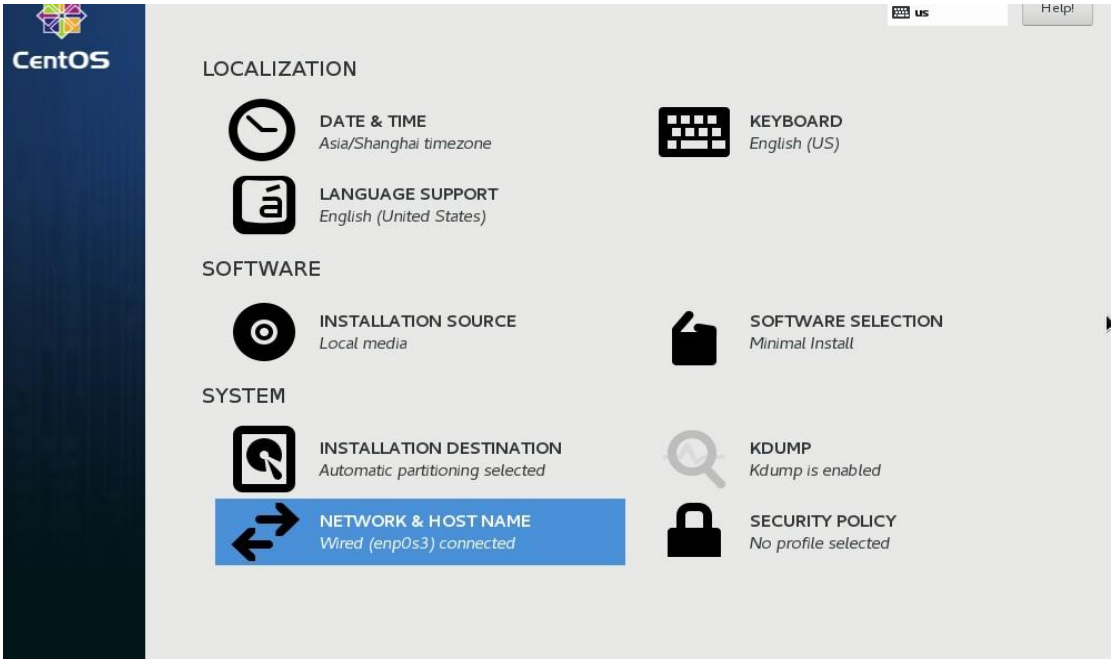
二、环境搭建

（一）创建新的虚拟机

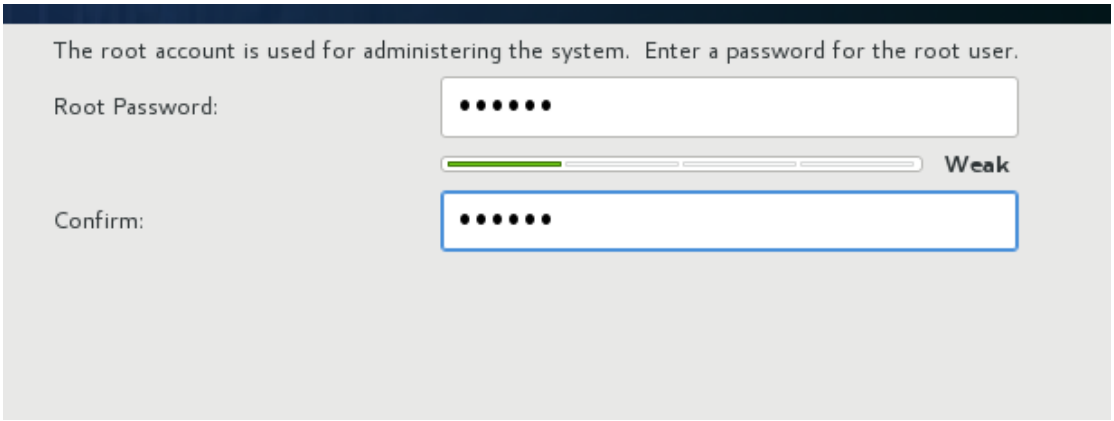
都默认选择下一步，然后启动虚拟机

选择虚拟机的镜像文件进行配置

选择 DATE & TIME 和 NETWORK & HOSTNAME 进行设置

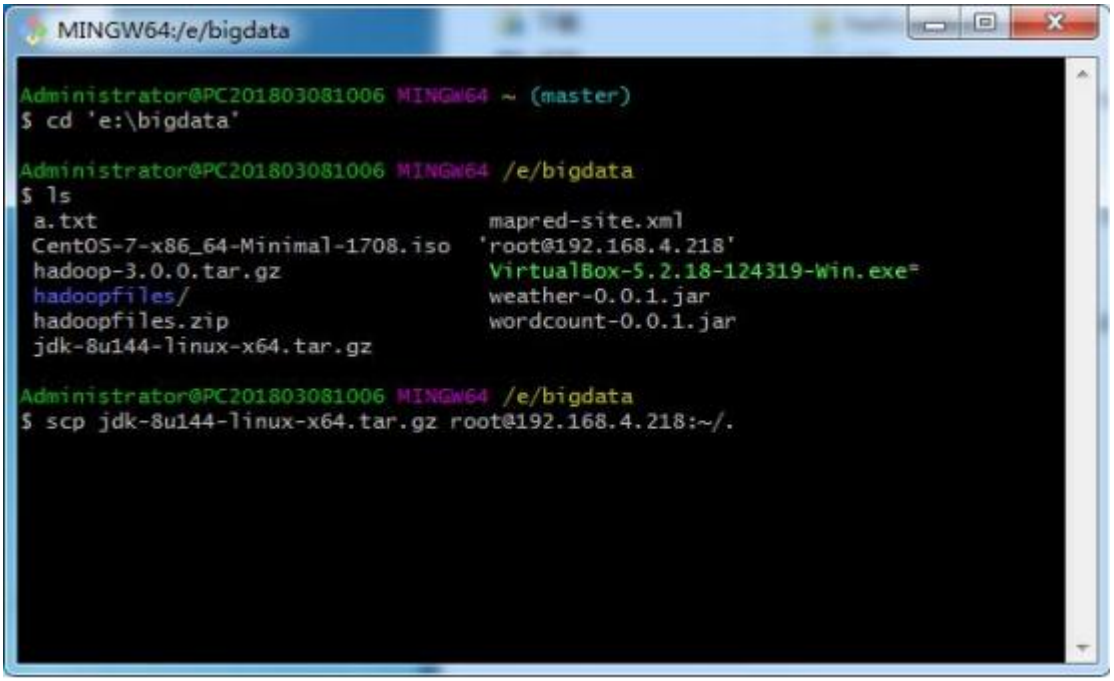


完成上述步骤后，进行密码的设置，之后等待安装完成



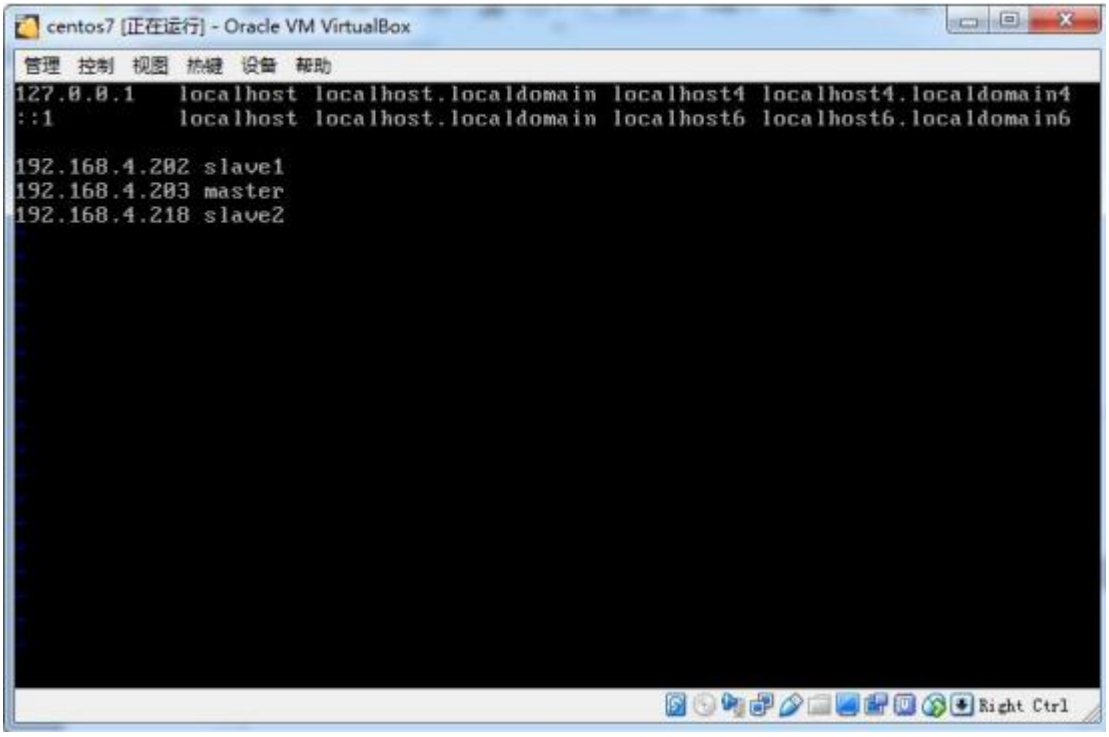
（二）虚拟机的搭建

将 jdk、hadoop 工具上传，进行安装



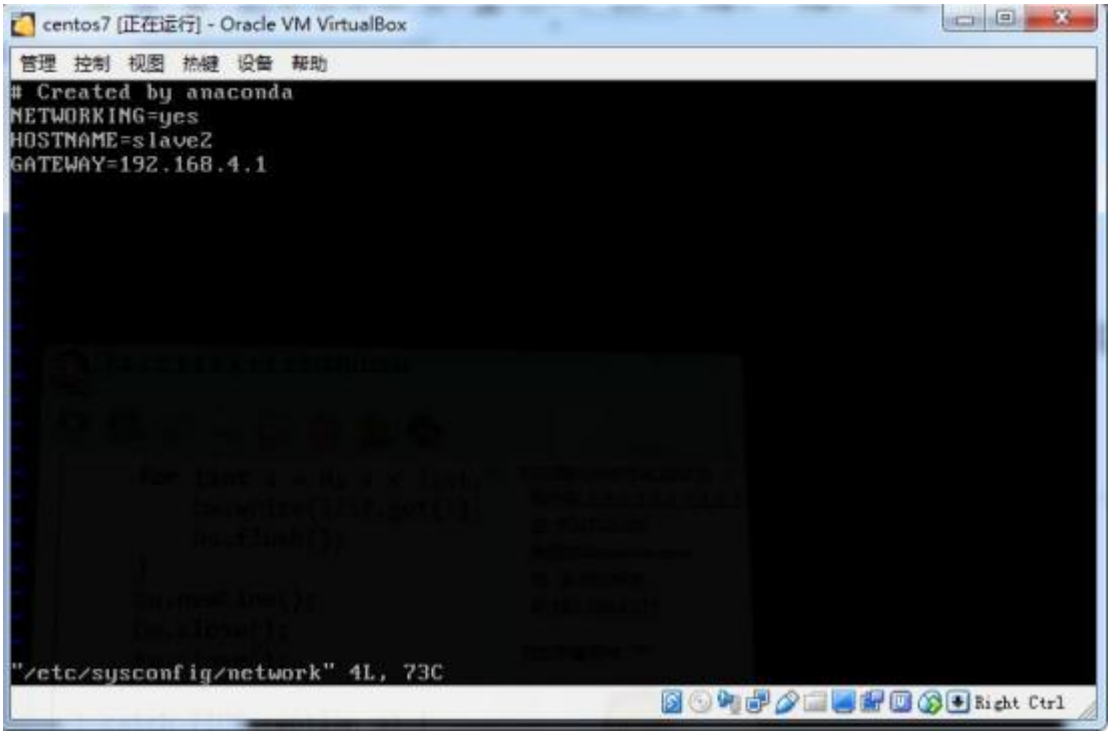
分别设置三台虚拟机的 hosts：

```
[root@slave2 ~]# vi /etc/hosts
```



网关设置：

```
[root@slave2 ~]# vi /etc/sysconfig/network
```



分别配置三台虚拟机的 hadoop：

```
[root@localhost hadoop]# vi sbin/start-dfs.sh
[root@localhost hadoop]# vi sbin/stop-dfs.sh
```

```
# Licensed to the Apache Software Foundation (ASF) under one or more
# contributor license agreements.  See the NOTICE file distributed with
# this work for additional information regarding copyright ownership.
# The ASF licenses this file to You under the Apache License, Version 2.0
# (the "License"); you may not use this file except in compliance with
# the License.  You may obtain a copy of the License at
#
#     http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.


HDFS_NAMENODE_USER=root
HDFS_DATANODE_USER=root
HDFS_SECONDARYNAMENODE_USER=root

# Start hadoop dfs daemons.
"sbin/start-dfs.sh" 185L, 5414C
```

```
[root@localhost hadoop]# vi sbin/start-yarn.sh
[root@localhost hadoop]# vi sbin/stop-yarn.sh
```

```
#!/usr/bin/env bash

# Licensed to the Apache Software Foundation (ASF) under one or more
# contributor license agreements.  See the NOTICE file distributed with
# this work for additional information regarding copyright ownership.
# The ASF licenses this file to You under the Apache License, Version 2.0
# (the "License"); you may not use this file except in compliance with
# the License.  You may obtain a copy of the License at
#
#     http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.


YARN_RESOURCEMANAGER_USER=root
YARN_NODEMANAGER_USER=root

## @description  usage info
## @audience     private
"sbin/start-yarn.sh" 103L, 3402C
```

```
[root@localhost hadoop]# bin/hdfs namenode -format
```

```
[root@localhost hadoop]# sbin/start-dfs.sh
```

```
[root@localhost hadoop]# jps
1728 DataNode
5585 Jps
1605 NameNode
1979 SecondaryNameNode
[root@localhost hadoop]#
```

配置 etc/hadoop 中的 core-size.xml 配置为当前虚拟机的 hostname

将本机的公钥导入到其余两台虚拟机中,分别为 master 和 slave1，

```
[root@slave2 ~]# ssh root@master 'cat >> .ssh/authorized_keys' < ~/.ssh/id_rsa.pub
[root@slave2 ~]# ssh root@slave1 'cat >> .ssh/authorized_keys' < ~/.ssh/id_rsa.pub
```

连接到另外两台虚拟机

```
Last login: Tue Sep  4 13:45:32 2018 from 192.168.4.218
[root@slave1 ~]#
```

```
[root@slave1 ~]# ssh slave2
Last login: Tue Sep  4 14:56:34 2018 from 192.168.4.212
[root@slave2 ~]#
```

上述步骤需同时在其余两台虚拟机上进行相同的配置

三、数据分析

（一）数据分析代码：

JAVA 代码：

Map代码：

```
public class WordCountMapper extends Mapper<LongWritable, Text ,
Text, Text> {

	@Override
	protected void map(LongWritable key, Text value,
Mapper<LongWritable, Text, Text, Text>.Context context)
			throws IOException, InterruptedException {
		String line = value.toString();
		String[] words = line.split(" ");
		String id = words[0];
		String cityname = words[1];
		String datetime =
StringUtils.substringBeforeLast(words[2],"-");
		String[] date = datetime.split("-");
		String tmp=StringUtils.substringBefore(words[5], "℃");
		String tmp1=StringUtils.substringBefore(words[6], "℃");
		context.write(new Text(cityname+"  "+date[0]+"年
"+date[1]+"月"), new Text(tmp+"--"+tmp1));


	}


}
```

Reduce 代码：

```
public class WordCountReducer extends Reducer<Text, Text , Text,
Text> {

	@Override
	protected void reduce(Text key, Iterable<Text> values,
			Reducer<Text, Text, Text, Text>.Context context)
throws IOException, InterruptedException {
```

```
            Integer tsum=0;
            Integer nsum=0;String s=null;
            int tavgtemperture=0;
            int navgtemperture=0;
            Text t=null;
            int i=0;
            for(Text value :  values) {
                  s=value.toString();
                  String[] words = s.split("--");
                  tsum += Integer.parseInt(words[0]);
                  nsum += Integer.parseInt(words[1]);

                  i++;
            }

            tavgtemperture=tsum/i;
            navgtemperture=nsum/i;
            t =new Text("白天平均温度是"+tavgtemperture+"℃        夜
间平均温度是"+navgtemperture+"℃");
            context.write(key,new Text(t));
      }

}
```

MapReduce 代码：

```
public class WordCountMapReduce
{
    public static void main( String[] args ) throws Exception
    {
        Configuration cfg = new Configuration();
        Job job = Job.getInstance(cfg, "worcount");
        job.setJarByClass(WordCountMapReduce.class);

        FileInputFormat.setInputPaths(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job, new Path(args[1]));

        job.setMapperClass(WordCountMapper.class);
        job.setReducerClass(WordCountReducer.class);

        job.setMapOutputKeyClass(Text.class);
        job.setMapOutputValueClass(Text.class);

        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(Text.class);

        boolean b = job.waitForCompletion(true);
        if(!b) {
                  System.out.println("wordcount task fail!");

        }

    }
```

}

将以下两个文件传到虚拟机中

```
Administrator@Stu1001 MINGW64 /c
$ scp weather1.txt root@192.168.4.212:~/.
weather1.txt                                  100%   14KB   5.2MB/s   00:00
```

```
Administrator@Stu1001 MINGW64 /e/bigdata
$ scp weather-0.0.1.jar root@192.168.4.212:~/.
weather-0.0.1.jar                             100% 6733     1.1MB/s   00:00
```

虚拟机下运行

```
[root@localhost hadoop]# bin/hdfs dfs -mkdir -p /weather/i
[root@localhost hadoop]# bin/hdfs dfs -put ~/weather1.txt /weather/i
[root@localhost hadoop]# bin/hadoop jar ~/weather-0.0.1.jar demo.mr.wordcount.Wo
rdCountMapReduce /weather/i /weather/o
2018-09-04 16:41:07,189 INFO beanutils.FluentPropertyBeanIntrospector: Error whe
n creating PropertyDescriptor for public final void org.apache.commons.configura
```

```
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=14392
        File Output Format Counters
                Bytes Written=460
[root@localhost hadoop]# bin/hdfs dfs -cat  /weather/o/*
```

（二）分析的结果：

我们选取了 3 个城市，从今年的 1-6 月的天气数据进行分析

```
[root@localhost hadoop]# bin/hdfs dfs -cat /weather/o/*
大田--2018-01    白天平均温度是16℃        夜间平均温度是6℃
大田--2018-02    白天平均温度是17℃        夜间平均温度是6℃
大田--2018-03    白天平均温度是24℃        夜间平均温度是11℃
大田--2018-04    白天平均温度是26℃        夜间平均温度是16℃
大田--2018-05    白天平均温度是31℃        夜间平均温度是20℃
大田--2018-06    白天平均温度是30℃        夜间平均温度是21℃
德化--2018-01    白天平均温度是14℃        夜间平均温度是6℃
德化--2018-02    白天平均温度是16℃        夜间平均温度是6℃
德化--2018-03    白天平均温度是22℃        夜间平均温度是10℃
德化--2018-04    白天平均温度是24℃        夜间平均温度是15℃
德化--2018-05    白天平均温度是28℃        夜间平均温度是19℃
德化--2018-06    白天平均温度是27℃        夜间平均温度是20℃
长汀--2018-01    白天平均温度是14℃        夜间平均温度是5℃
长汀--2018-02    白天平均温度是16℃        夜间平均温度是6℃
长汀--2018-03    白天平均温度是21℃        夜间平均温度是10℃
长汀--2018-04    白天平均温度是24℃        夜间平均温度是15℃
长汀--2018-05    白天平均温度是30℃        夜间平均温度是21℃
长汀--2018-06    白天平均温度是29℃        夜间平均温度是22℃
[root@localhost hadoop]#
```