**Java链接Hbase**

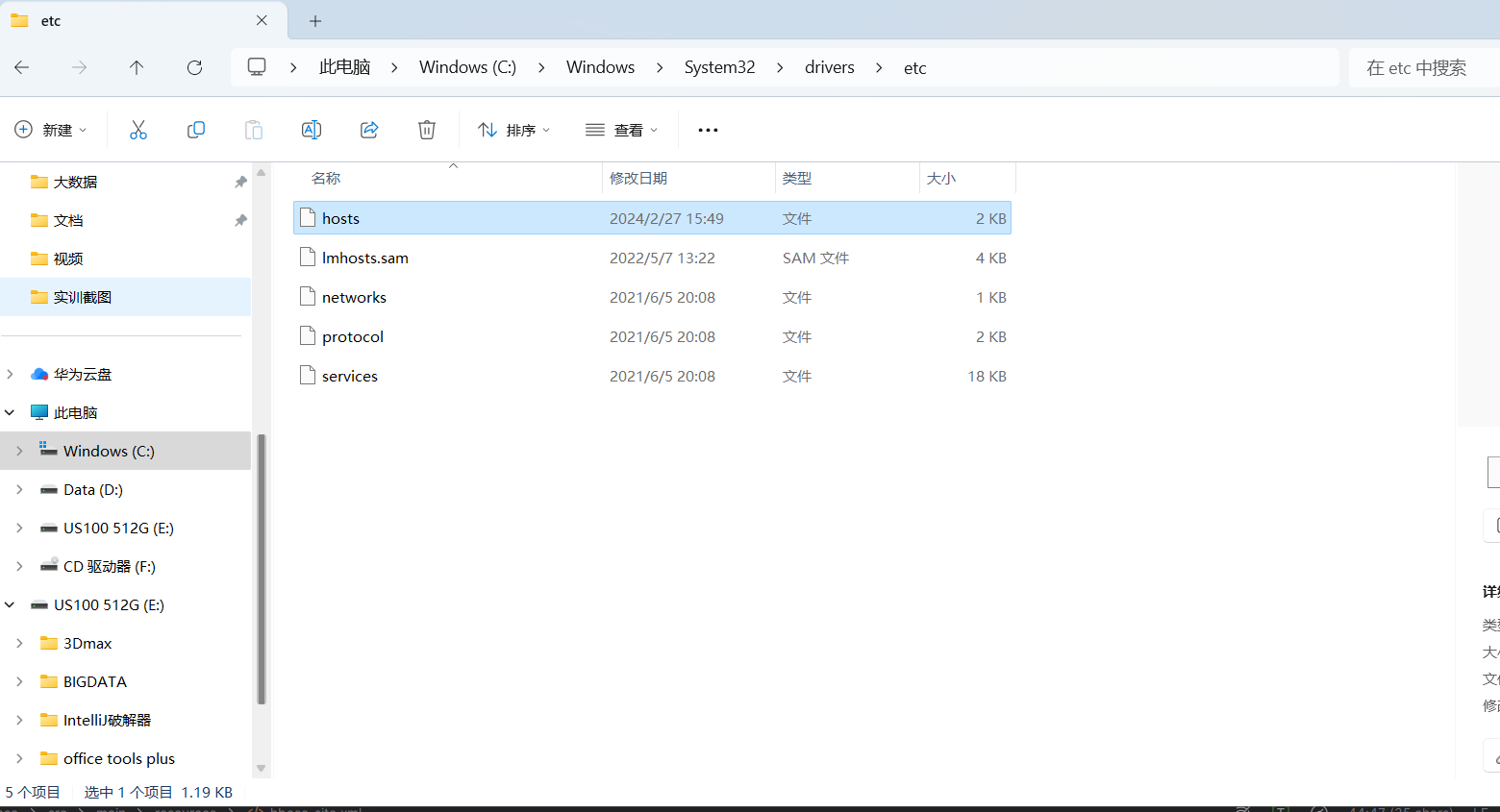
1. 复制hbase和hadoop中的文件（hbase-site.xml和core-site-xml和log4j.propertion）

|  |
| --- |
| Sz /opt/hbase2.2.3/conf/hbase-site.xml  Sz /opt/hbase2.2.3/conf/core-site.xml |

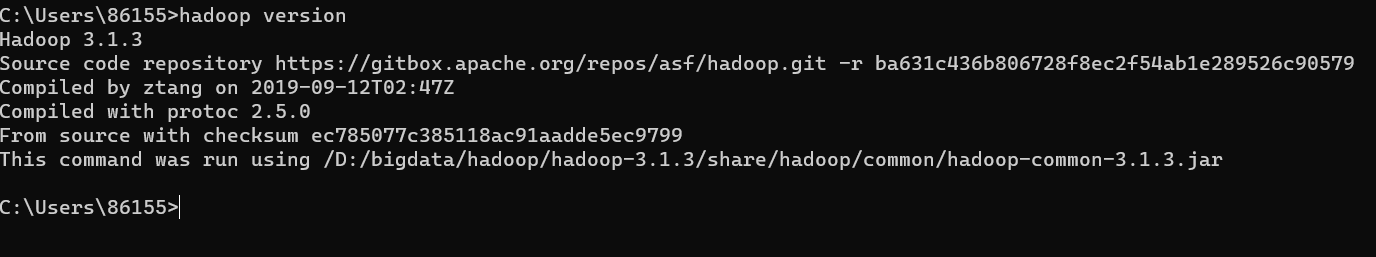
2．创建包结构和类

|  |
| --- |
| 用idea床架按maven  把上面的文件放到resources中，并给其build一下如图： |

1. 修改本地机host



1. 修改跟linux上的host一样
2. 在windows上配置hadoop不然会报错且需要下载winutils-master
3. 解压完成之后需要把winutils-master中的lib跟hadoop上的替换
4. 然后进行环境变量的配置如这个视频所示
5. 用cmd进行测试输入 Hadoop version



3.写代码kafka备份到hbase中

|  |
| --- |
| package org.example;  import org.apache.flink.api.common.functions.MapFunction;  import org.apache.flink.api.common.serialization.SimpleStringSchema;  import org.apache.flink.api.java.tuple.Tuple4;  import org.apache.flink.calcite.shaded.com.fasterxml.jackson.databind.JsonNode;  import org.apache.flink.calcite.shaded.com.fasterxml.jackson.databind.ObjectMapper;  import org.apache.flink.streaming.api.datastream.DataStream;  import org.apache.flink.streaming.api.environment.StreamExecutionEnvironment;  import org.apache.flink.streaming.connectors.kafka.FlinkKafkaConsumer;  import org.apache.hadoop.conf.Configuration;  import org.apache.hadoop.hbase.HBaseConfiguration;  import org.apache.hadoop.hbase.TableName;  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;  import org.apache.hadoop.hbase.util.Bytes;  import java.util.Properties;  public class KafkaToHBaseBackup {  private static final String TABLE\_NAME = "student";  private static final String COLUMN\_FAMILY = "cf";  public static void main(String[] args) {  // 创建Flink执行环境  StreamExecutionEnvironment env = StreamExecutionEnvironment.getExecutionEnvironment();  // 设置Kafka消费者的配置  Properties properties = new Properties();  properties.setProperty("bootstrap.servers", "192.168.10.41:9092");  // 创建Kafka消费者  FlinkKafkaConsumer<String> kafkaConsumer = new FlinkKafkaConsumer<>(  "ods\_mall\_data", new SimpleStringSchema(), properties);  kafkaConsumer.setStartFromLatest();// 从最新的记录开始消费  // 添加Kafka消费者到Flink数据流  DataStream<String> inputStream = env.addSource(kafkaConsumer);  // 读取想要的数据（表名和数据）  // 读取想要的数据（表名和数据）  DataStream<Tuple4<String, String, String, String>> combinedStream = inputStream.map(new MapFunction<String, Tuple4<String, String, String, String>>() {  @Override  public Tuple4<String, String, String, String> map(String value) throws Exception {  ObjectMapper objectMapper = new ObjectMapper();  JsonNode jsonNode = objectMapper.readTree(value);  String tableName = jsonNode.get("table").asText();  JsonNode dataNode = jsonNode.get("data");  // 获取特定字段的值  String id = dataNode.get("id").asText();  String name = dataNode.get("name").asText();  String age = dataNode.get("age").asText();  String sex = dataNode.get("sex").asText();  // 返回想要的值  return Tuple4.of( id,name, age, sex);  }  });  // combinedStream.print();  //// 添加到hbase中  combinedStream.addSink(new HBaseSink());  // 执行任务  try {  env.execute("ods\_mall\_data");  } catch (Exception e) {  throw new RuntimeException(e);  }  closeConsumer(kafkaConsumer);  }  private static void closeConsumer(FlinkKafkaConsumer<String> kafkaConsumer) {  try {  kafkaConsumer.close();  } catch (Exception e) {  e.printStackTrace();  }  }  //  public static class HBaseSink implements org.apache.flink.streaming.api.functions.sink.SinkFunction<Tuple4<String, String,String, String>> {  private transient Connection connection;  private transient Table table;  @Override  public void invoke(Tuple4<String, String,String, String> value, Context context) throws Exception {  byte[] value1=Bytes.toBytes(value.getArity());  System.out.println(value1);  if (connection == null) {  Configuration config = HBaseConfiguration.create();  config.set("hbase.zookeeper.quorum", "192.168.10.41");  System.out.println(config);  connection = ConnectionFactory.createConnection(config);  table = connection.getTable(TableName.valueOf(TABLE\_NAME));  }  byte[] rowKey = Bytes.toBytes(value.f0);// 设置rowKey  byte[] columnFamily = COLUMN\_FAMILY.getBytes();  byte[] qualifiername =Bytes.toBytes("name"); // 设置qualifier  byte[] qualifierAge =Bytes.toBytes("age"); // 设置qualifier  byte[] qualifierSex =Bytes.toBytes("sex"); // 设置qualifier  byte[] name = Bytes.toBytes(value.f1);  byte[] age = Bytes.toBytes(value.f2);  byte[] sex = Bytes.toBytes(value.f3);  Put put = new Put(rowKey);  put.addColumn(columnFamily, qualifiername, name);  put.addColumn(columnFamily, qualifierAge, age);  put.addColumn(columnFamily, qualifierSex, sex);  table.put(put);  table.close();  }}} |

创建hbase表

|  |
| --- |
| 1. 打开 HBase Shell，运行以下命令：   hbase shell   1. 创建表 "student"：   create 'student', 'cf'   1. 添加列族 "cf"：   alter 'student', {NAME => 'cf'}   1. 添加列 "name"、"age"、"sex"：   alter 'student', {NAME => 'cf:name'}, {NAME => 'cf:age'}, {NAME => 'cf:sex'}  这样，你就创建了表 "student"，并向其中添加了列族 "cf" 和列 "name"、"age"、"sex" |