1. **Richsourcefunction**
2. **开发**

**PreparedStatement 实例包含已编译的 SQL 语句。**

**prepareStatement对象防止sql注入的方式是把用户非法输入的单引号用\反斜杠做了转义，从而达到了防止sql注入的目的**

**PreparedStatement可以有效防止sql注入，所以生产环境上一定要使用PreparedStatement**

**Connection 对象的数据库能够提供描述其表、所支持的 SQL 语法、存储过程、此连接功能等等的信息。此信息是使用 getMetaData 方法获得的。**

**======**

**package** com.mysql;  
  
**import** java.sql.DriverManager;  
**import** java.sql.ResultSet;  
  
**import** org.apache.flink.api.java.tuple.Tuple2;  
**import** org.apache.flink.configuration.Configuration;  
**import** org.apache.flink.streaming.api.functions.source.RichSourceFunction;  
  
**import** com.mysql.jdbc.Connection;  
**import** com.mysql.jdbc.PreparedStatement;  
  
**public class** m1 **extends** RichSourceFunction<Tuple2<Integer,String>> {  
  
 **private** Connection **connect** = **null**;  
 **private** PreparedStatement **ps** = **null**;  
  
 @Override  
 **public void** open(Configuration parameters) **throws** Exception {  
 **super**.open(parameters);  
 Class.*forName*(**"com.mysql.jdbc.Driver"**);  
 **connect** = (Connection) DriverManager.*getConnection*(**"jdbc:mysql://192.168.8.201:3306"**, **"sq"**, **"sq123"**);  
 **ps** = (PreparedStatement) **connect** .prepareStatement(**"select id,name from flink.test1"**);  
 }  
  
  
 @Override  
 **public void** run(  
 SourceContext<Tuple2<Integer, String>> collect)  
 **throws** Exception {  
 ResultSet resultSet = **ps**.executeQuery();  
 **while** (resultSet.next()) {  
 Tuple2<Integer, String> tuple = **new** Tuple2<Integer, String>();  
 tuple.setFields(resultSet.getInt(1), resultSet.getString(2));  
 collect.collect(tuple);  
 }  
  
 }  
  
 @Override  
 **public void** cancel() {  
 **try** {  
 **super**.close();  
 **if** (**connect** != **null**) {  
 **connect**.close();  
 }  
 **if** (**ps** != **null**) {  
 **ps**.close();  
 }  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
 }  
}

**package** com.mysql;  
  
**import** org.apache.flink.api.java.tuple.Tuple2;  
**import** org.apache.flink.streaming.api.datastream.DataStream;  
**import** org.apache.flink.streaming.api.environment.StreamExecutionEnvironment;  
  
**public class** sourcemysql {  
 **public static void** main(String[] args) **throws** Exception {  
 **final** StreamExecutionEnvironment env = StreamExecutionEnvironment.*getExecutionEnvironment*();  
 DataStream<Tuple2<Integer,String>> dataStream = env.addSource(**new** m1());  
 dataStream.print();  
 env.execute(**"mmm"**) ;  
 }  
}