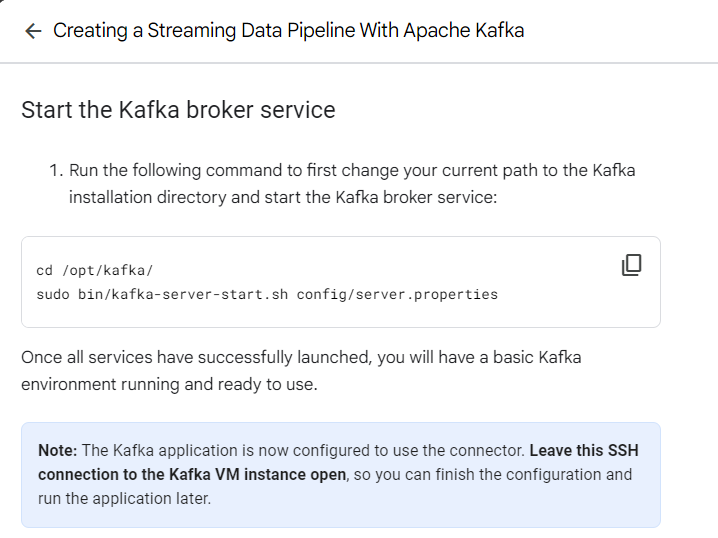
1. Open SSH to Change Current directory to Kafka installation directory and Switch on Zookeeper and keep the terminal open as it is

A screenshot of a computer

Description automatically generated

1. Open another SSH Terminal



1. Open Another Terminal

Here you can create topics and start working with Kafka Topics

1. Create input topic and 2. Output topic

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

A black screen with many small colored lines

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated

A black screen with white text

Description automatically generated

**Task 3. Process the input data with Kafka streams**

Now that you have generated some input data, you can run your first Kafka Streams based Java application.

You will run the [WordCount demo application](https://github.com/apache/kafka/blob/2.4/streams/examples/src/main/java/org/apache/kafka/streams/examples/wordcount/WordCountDemo.java" \t "_blank), which is included in Kafka. It implements the WordCount algorithm, which computes a word occurrence histogram from an input text.

However, unlike other WordCount examples you might have seen before that operate on *finite, bounded data*, the WordCount demo application behaves slightly differently because it is designed to operate on an **infinite, unbounded stream** of input data.

Similar to the bounded variant, it is a stateful algorithm that tracks and updates the counts of words. However, since it must assume potentially unbounded input data, it will periodically output its current state and results while continuing to process more data because it cannot know when it has processed "all" the input data.

This is a typical difference between the class of algorithms that operate on unbounded streams of data and, say, batch processing algorithms such as Hadoop MapReduce. It will be easier to understand this difference once you inspect the actual output data later on.

Kafka's WordCount demo application is bundled with Confluent Platform, which means you can run it without further ado, i.e. you do not need to compile any Java sources and so on.

* Now, execute the following command to run the WordCount demo application. You can safely ignore any warn log messages:

sudo bin/kafka-run-class.sh org.apache.kafka.streams.examples.wordcount.WordCountDemo

Copied!

content\_copy

**Note:** The warning errors are just generic alerts. You can safely ignore them for the purposes of this lab.

The WordCount demo application will read from the input topic streams-plaintext-input, perform the computations of the WordCount algorithm on the input data, and continuously write its current results to the output topic streams-wordcount-output (the names of its input and output topics are hardcoded). You can terminate the demo at any point by entering Ctrl+C from the keyboard.



A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated