Big data

Apache Kafka

Introduction

- Kafka is a distributed event streaming platform
- It is used for high-performance data pipelines and streaming analytics
- Kafka stores data in local files
- Ref:
  - https://youtu.be/vHbvbwSEYGo (must watch)
  - https://kafka.apache.org/documentation

- Use cases
  - See https://kafka.apache.org/uses

Big data

Apache Kafka

…Introduction

- Kafka® is a distributed streaming platform. What exactly does that mean?
- We think of a streaming platform as having three key capabilities:
  - It lets you publish and subscribe to streams of records. In this respect it is similar to a message queue or enterprise messaging system.
  - It lets you store streams of records in a fault-tolerant way.
  - It lets you process streams of records as they occur.

Big data

Apache Kafka

…Introduction

- What is Kafka good for?
- It gets used for two broad classes of application:
  - Building real-time streaming data pipelines that reliably get data between systems or applications
  - Building real-time streaming applications that transform or react to the streams of data
- To understand how Kafka does these things, let's dive in and explore Kafka's capabilities from the bottom up.

- First a few concepts:
  - Kafka is run as a cluster on one or more servers.
  - The Kafka cluster stores **streams of records** in categories **called topics**.
  - Each record consists of a **key**, a **value**, and a **timestamp**.

Big data

Apache Kafka

…Introduction

- Kafka has four core APIs:
  - The **Producer API** allows an **application** to **publish** a **stream records** to one or more Kafka topics.
  - The **Consumer API** allows an **application** to **subscribe** to one or more **topics** and **process the stream** of records produced to them.
  - The **Connector API** allows building and running **reusable producers or consumers** that **connect Kafka topics to existing applications or data systems**. For example, a connector to a relational database might capture every change to a table.
  - The **Streams API** allows an **application** to **act as a stream processor**, consuming an input stream from one or more topics and producing an output stream to one or more output topics, effectively **transforming the input streams to output streams**.

Big data

Apache Kafka

…Introduction

- In Kafka the communication between the clients and the servers is done with a simple, high-performance, language agnostic TCP protocol. This protocol is versioned and maintains backwards compatibility with older version.
  - Clients are available in Java and scala as well as a number of other languages including Python, Go, C/C++, etc.

Big data

Apache Kafka

Getting Started

- *Quick start*
- https://kafka.apache.org/documentation/#quickstart
- 1. You need Java JDK in your environment
  - On a terminal, check using the command
    - java --version
  - This should echo your Java version if installed. The version should not be less than 8.
  - To download Java go to https://www.oracle.com/java/technologies/downloads/
  - Install java on your windows. Close and reopen VS Code if using terminal from VS Code. If using powershell or cmd, close and reopen as well to have access to the new java in path.
  - In Windows OS, better to use WSL or bash shell for this process.
  - For bash, the java installed on windows will be available.
  - If using wsl terminal
    - Run wget https://download.oracle.com/java/22/latest/jdk-22_linux-x64_bin.deb
    - Run *sudo dpkg -i jdk-22_linux-x64_bin.deb*
    - *java -version* should show the version installed.

Big data

Apache Kafka

…Getting Started

- **Download latest binary** into your DAT608 class projects directory from
  - https://downloads.apache.org/kafka/
    - E.g. https://downloads.apache.org/kafka/3.7.0/kafka_2.12-3.7.0.tgz
    - Let's make a directory inside a projects root, where we shall install kafka. On windows, the file url path must not have space in name. E.g. use c:\kafka
  - *mkdir kafka*
  - *cd kafka*

Big data

Apache Kafka

…Getting Started

- Get the file from download location using *wget:*
  - *wget https://downloads.apache.org/kafka/3.7.0/kafka_2.12-3.7.0.tgz*
  - For mac, you can download and install *wget* using brew.
  - *brew install wget*
- For windows, download and install *wget* from https://eternallybored.org/misc/wget/
- To extract the downloaded kafka from the .tgz file, run:
  - *tar -xzf kafka_2.12-3.7.0.tgz*
- Change into the extracted folder for further work
  - *cd kafka_2.12-3.7.0*

Big data

Apache Kafka

…Getting Started

- **Startup Kafka with Kraft** instead of the now former Zookeeper
- Note that the scripts below are *.sh* scripts and therefore are for *bash* shell. You can use bash or wsl in your VS Code terminal environment. Otherwise, use the .bat equivalent inside the bin/windows directory inside your *kafka_2.12-3.7.0*.
- A. Run only once
  - 1. Generate a Cluster UUID
    - *KAFKA_CLUSTER_ID="$(bin/kafka-storage.sh random-uuid)"*
  - 2. Format Log Directories
    - *bin/kafka-storage.sh format -t $KAFKA_CLUSTER_ID -c config/kraft/server.properties*
- B. Start the Kafka Server
  - *bin/kafka-server-start.sh config/kraft/server.properties*

Big data

Apache Kafka

…Getting Started

- Illustration: **Create a Topic** to store your events
- Open another terminal for this and run:
  - *bin/kafka-topics.sh --create --topic quickstart-events --bootstrap-server localhost:9092*
- You can check properties of Topic
  - *bin/kafka-topics.sh --describe --topic quickstart-events --bootstrap-server localhost:9092*

Big data

Apache Kafka

…Getting Started

- **Write some Events** to the Topic using a producer console. Run:
  - *bin/kafka-console-producer.sh --topic quickstart-events --bootstrap-server localhost:9092*
- On the terminal prompt, write:
  - “This is my first event”
  - “This is my second event”
  - “and more, if you wish”
- You can stop the producer console with *Ctrl-C* at any time.

Big data

Apache Kafka

…Getting Started

- **Read the Events**
  - *bin/kafka-console-consumer.sh --topic quickstart-events --from-beginning --bootstrap-server localhost:9092*
- This should show what we have written to the topic. E.g.,
  - This is my first event
  - This is my second event

- **Restarting Kafka**
  - Assuming that your /tmp/kraft-combined-logs/meta.properties is intact.
  - *bin/kafka-server-start.sh config/kraft/server.properties*