Cloudera Streaming Analytics 1.8.0

Cloudera Streaming Analytics Overview

Date published: 2019-12-17
Date modified: 2022-09-28

CLOUDERA

Legal Notice

Contents

Streaming Analytics in Cloudera

Cloudera Streaming Analytics (CSA) offers real-time stream processing and streaming analytics powered by Apache Flink. Flink implemented on CDP provides a flexible streaming solution with low latency that can scale to large throughput and state. In addition to Flink, CSA includes SQL Stream Builder (SSB) to offer data analytical experience using SQL queries on your data streams.

Key features of Cloudera Streaming Analytics

Apache Flink

> CSA is powered by Apache Flink that offers a framework for real-time stream processing and streaming analytics. CSA offers the features and functionalities of the upstream Apache Flink integrated on CDP Private Cloud Base.

SQL Stream Builder

> SQL Stream Builder is a job management interface to compose and run Continuous Streaming SQL on streams using Apache Flink as an engine, as well as to create REST APIs for the results.

Cloudera Platform

> Implementing Flink on the Cloudera Platform allows you to easily integrate with Runtime components, and have all the advantages of cluster and service management with Cloudera Manager.

Streaming Platform

> For streaming analytics, CSA fits into a complete streaming platform augmented by Apache Kafka, Schema Registry, Streams Messaging Manager in the Cloudera Runtime stack.

Supported Connectors

> CSA offers a set of connectors for Flink and SSB from which you can choose from based on your requirements. Kafka, HBase, HDFS, Kudu and Hive connectors are available for Flink. Kafka, HDFS/S3, JDBC and a set of the Debezium CDC connectors are available for SSB. Other than the connectors, SSB also supports Schema Registry, Hive and Kudu as catalogs.

Monitoring Solutions

> In CDP Private Cloud Base, Streams Messaging components provide tools that support the operational needs of CSA. For example, when Kafka is used as a connector, you can use Kafka Metrics Reporter and Streams Messaging Manager (SMM) for Kafka management and alerting of Kafka actions. Beside SMM, you can use the Flink Dashboard to monitor your Flink and SSB jobs.
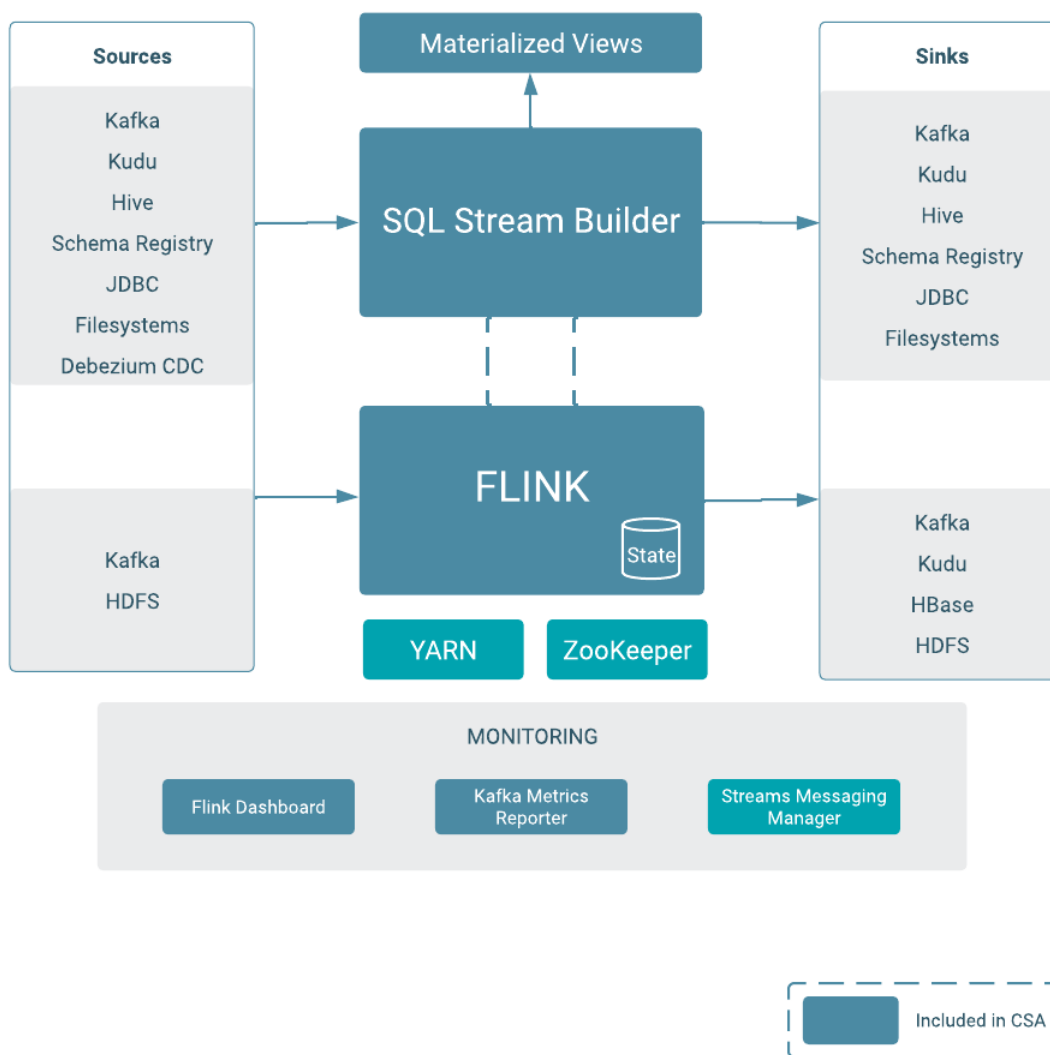
What is Apache Flink?

Flink is a distributed processing engine and a scalable data analytics framework. You can use Flink to process data streams at a large scale and to deliver real-time analytical insights about your processed data with your streaming application.

Flink is designed to run in all common cluster environments, perform computations at in-memory speed and at any scale. Furthermore, Flink provides communication, fault tolerance, and data distribution for distributed computations over data streams. A large variety of enterprises choose Flink as a stream processing platform due to its ability to handle scale, stateful stream processing, and event time.
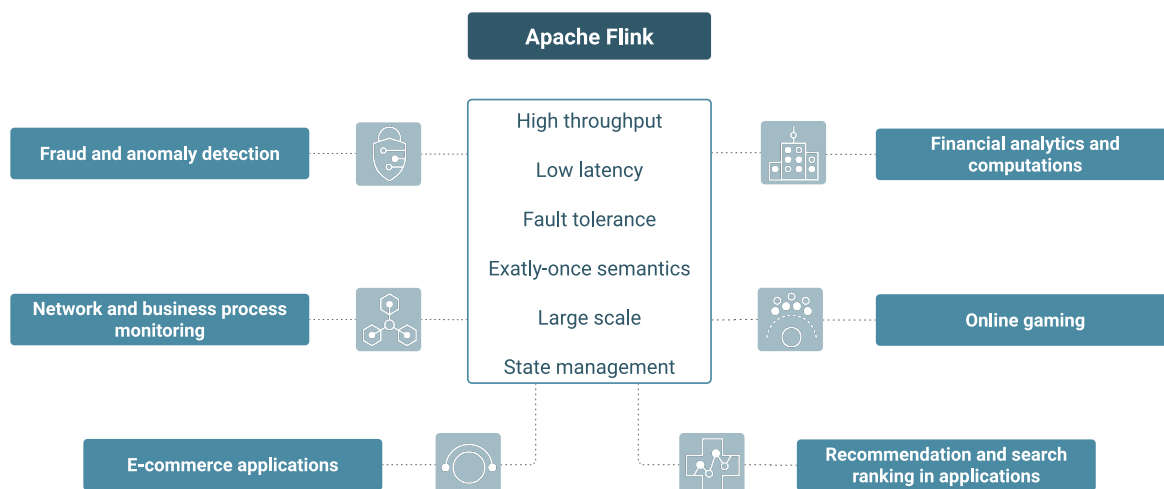
Core Features of Flink

Learn more about the specific details of Flink architecture, the DataStream API, how Flink handles event time and watermarks, and how state management works in Flink.

Architecture

The two main components for the task execution process are the Job Manager and Task Manager. The Job Manager on a master node starts a worker node. On a worker node the Task Managers are responsible for running tasks and the Task Manager can also run more than one task at the same time. The resource management for the tasks are completed by the Job manager in Flink. In a Flink cluster, Flink jobs are executed as YARN applications. HDFS is used to store recovery and log data, while ZooKeeper is used for high availability coordination for jobs.
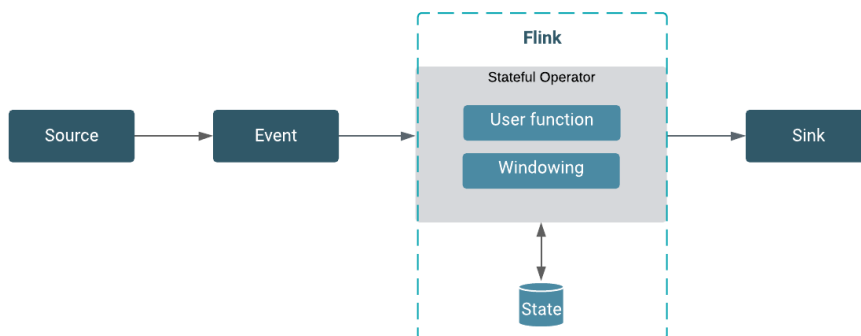
DataStream API

The DataStream API is used as the core API to develop Flink streaming applications using Java or Scala programming languages. The DataStream API provides the core building blocks of the Flink streaming application: the datastream and the transformation on it. In a Flink program, the incoming data streams from a source are transformed by a defined operation which results in one or more output streams to the sink.
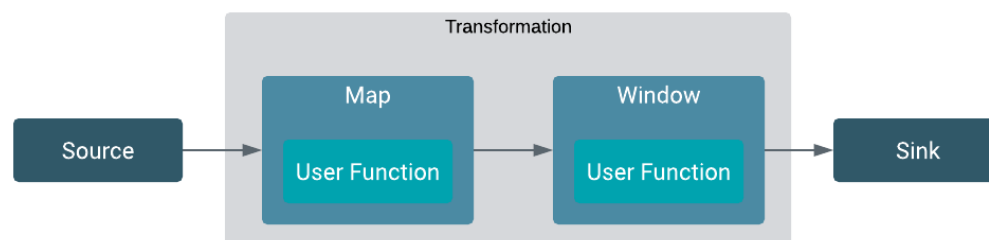
Operators

Operators transform one or more DataStreams into a new DataStream. Programs can combine multiple transformations into sophisticated data flow topologies. Other than the standard transformations like map, filter, aggregation, you can also create windows and join windows within the Flink operators. On a dataflow one or more operations can be defined which can be processed in parallel and independently to each other. With windowing functions, different computations can be applied to different streams in the defined time window to further maintain the processing of events. The following image illustrates the parallel structure of dataflows.
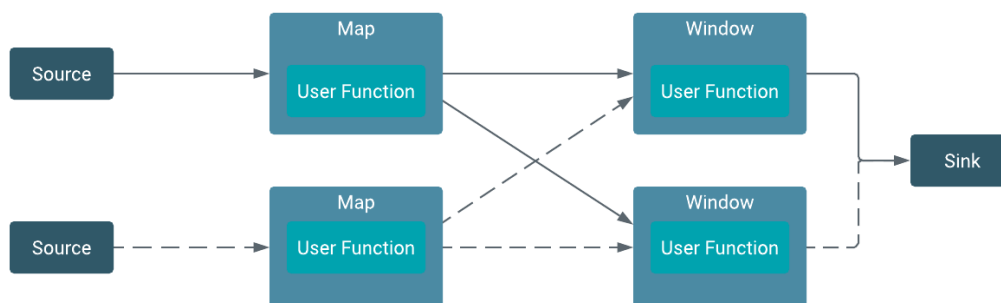

State and state backend

Stateful applications process dataflows with operations that store and access information across multiple events. You can use Flink to store the state of your application locally in state backends that guarantee lower latency when accessing your processed data. You can also create checkpoints and savepoints to have a fault-tolerant backup of your streaming application on a durable storage.
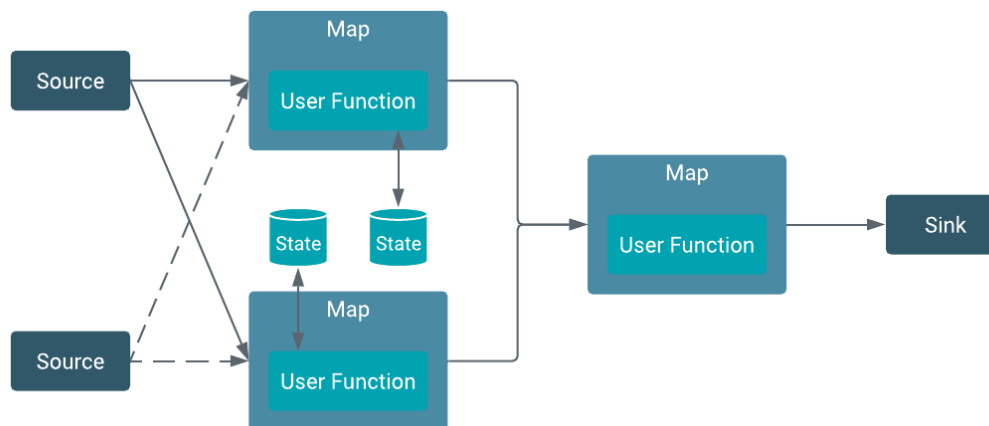

Event time and watermark

In time-sensitive cases where the application uses alerting or triggering functions, it is important to distinguish between event time and processing time. To make the designing of applications easier,

you can create your Flink application either based on the time when the event is created or when it is processed by the operator.
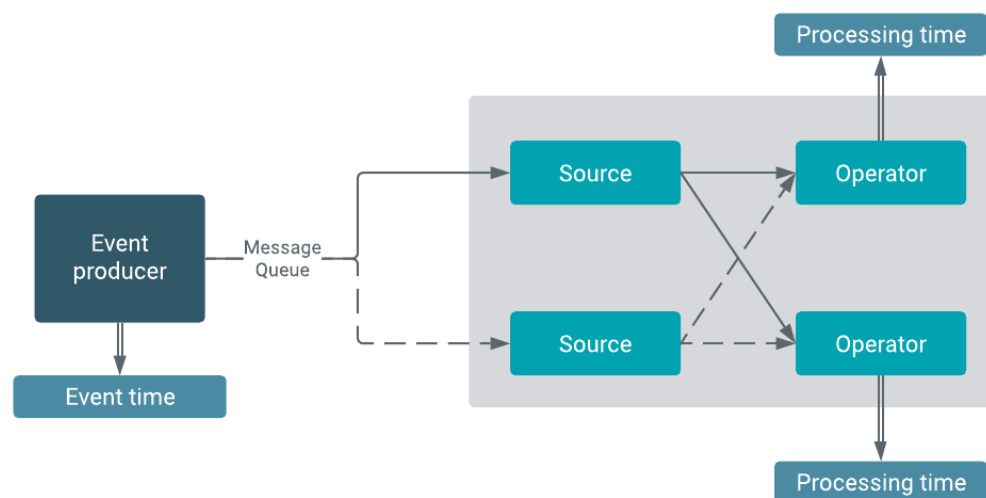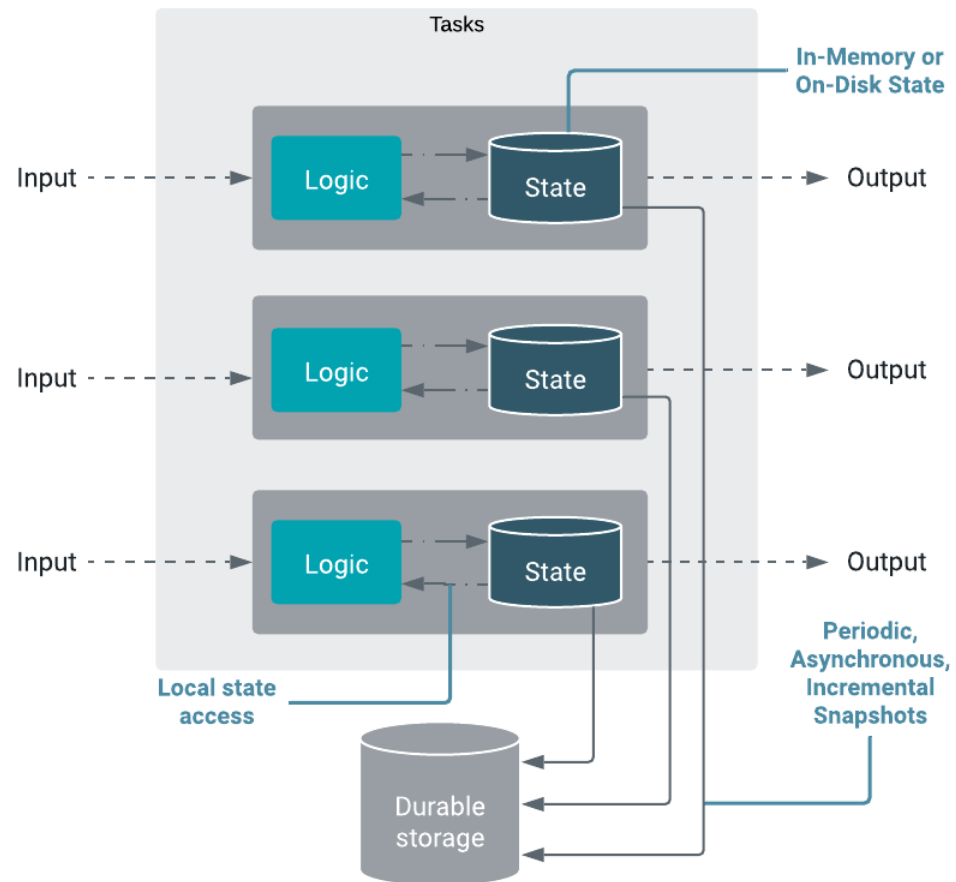

With only the event time, it is not clear when the events are processed in the application. To track the time for an event time based application, watermark can be used.
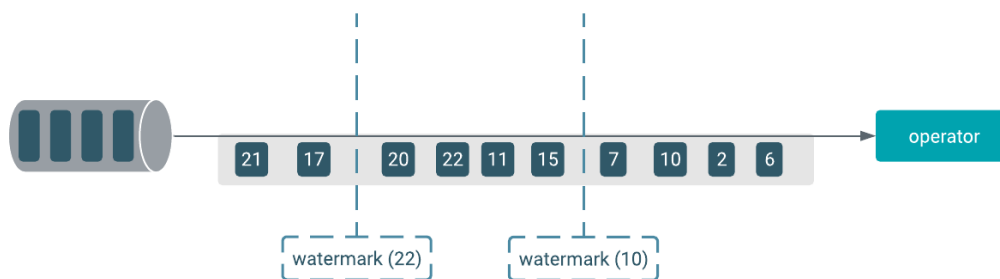
Checkpoints and savepoints

Checkpoints and savepoints can be created to make the Flink application fault tolerant throughout the whole pipeline. Flink contains a fault tolerance mechanism that creates snapshots of the data stream continuously. The snapshot includes not only the dataflow, but the state attached to it. In case of failure, the latest snapshot is chosen and the system recovers from that checkpoint. This guarantees that the result of the computation can always be consistently restored. While checkpoints are created and managed by Flink, savepoints are controlled by the user. A savepoint can be described as a backup from the executed process.

Related Information