DATA PROCESSING WITH APACHE FLINK

About

upGrad

Course: Data Engineering - II

Lecture On: Apache Flink

Instructor: Mayukh Chakraborty

MODULE INTRODUCTION

Session 1

1. Introduction to Apache Flink
2. Apache Flink vs Apache Spark
3. Why Apache Flink?
4. Flink Ecosystem and its programming model
5. Flink Installation and its use cases

Session 2

1. Introduction to Dataset API
2. Transformations
3. Brief overview of connectors

Session 3

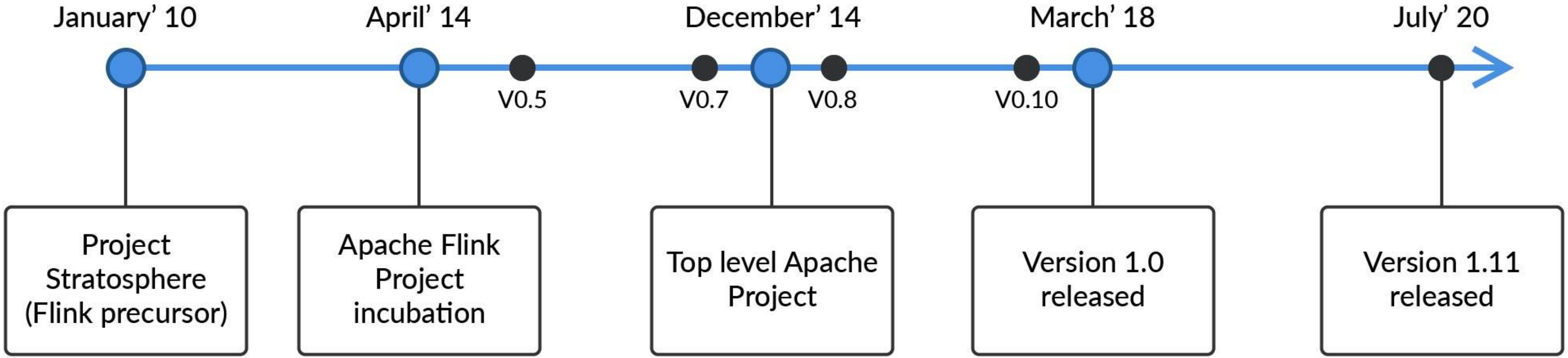
1. Introduction to Datastream API
2. State & Fault Tolerance
3. Transformations
4. Time & Windows
5. Brief overview of connectors

Session 4

1. Introduction to Table API & SQL
2. Streaming concepts
3. Table API operations
4. SQL Capabilities
5. Functions

INTRODUCTION TO APACHE FLINK

BRIEF HISTORY



January’ 10

Project Stratosphere (Flink precursor)

April’ 14

Apache Flink Project incubation

V0.5

V0.7

December’ 14

Top level Apache Project

V0.8

V0.10

March’ 18

Version 1.0 released

July’ 20

Version 1.11 released

CASE STUDY: NETFLIX

NETFLIX

Uses Flink in Keystone:
a real-time data pipeline

Videos

125 Million hours
of videos/day

Users

100M+ daily
active users

Data

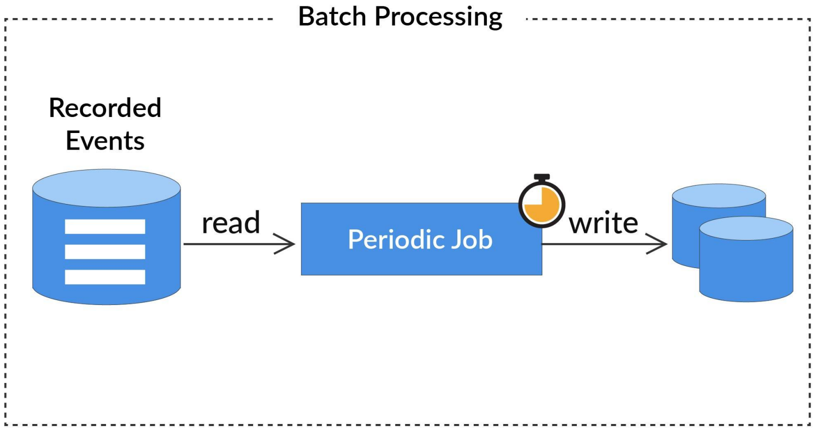
12 Petabytes
data/day

Events

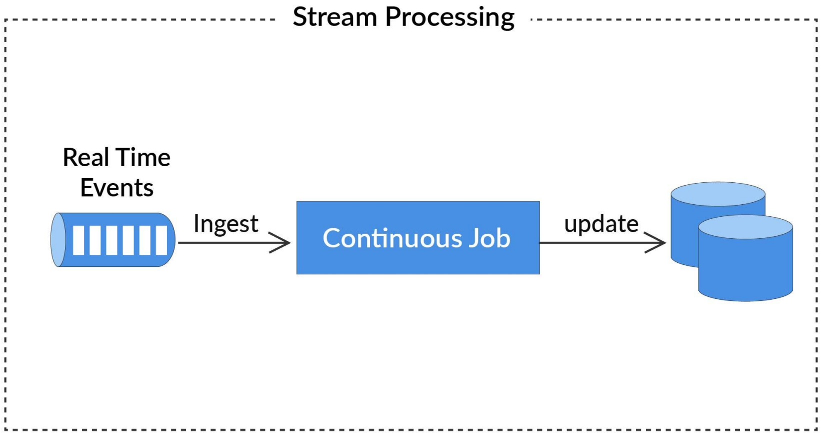
3 Trillions
events/day

Source: Netflix Tech Blog, 2018

THE BIG DATA DEBATE



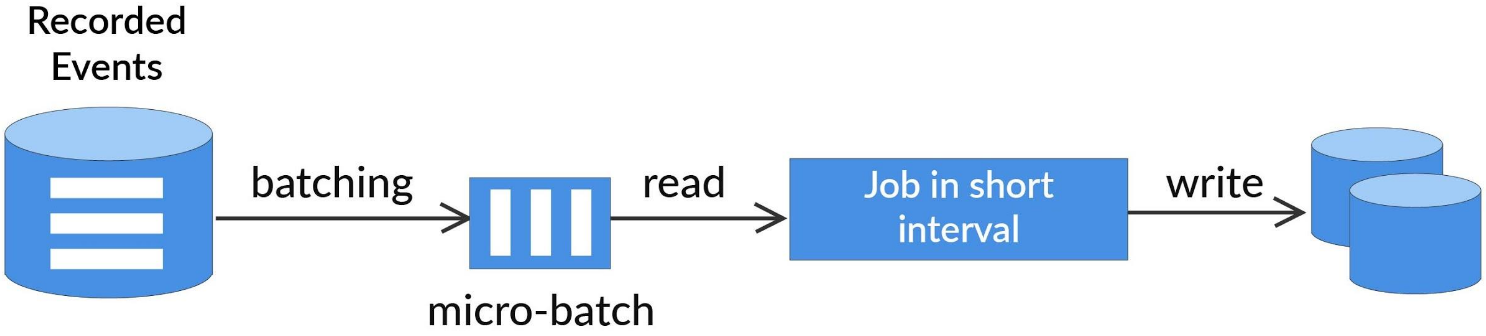
Vs



- Process events at periodic interval.
- Latency between the arrival & processing time of an event

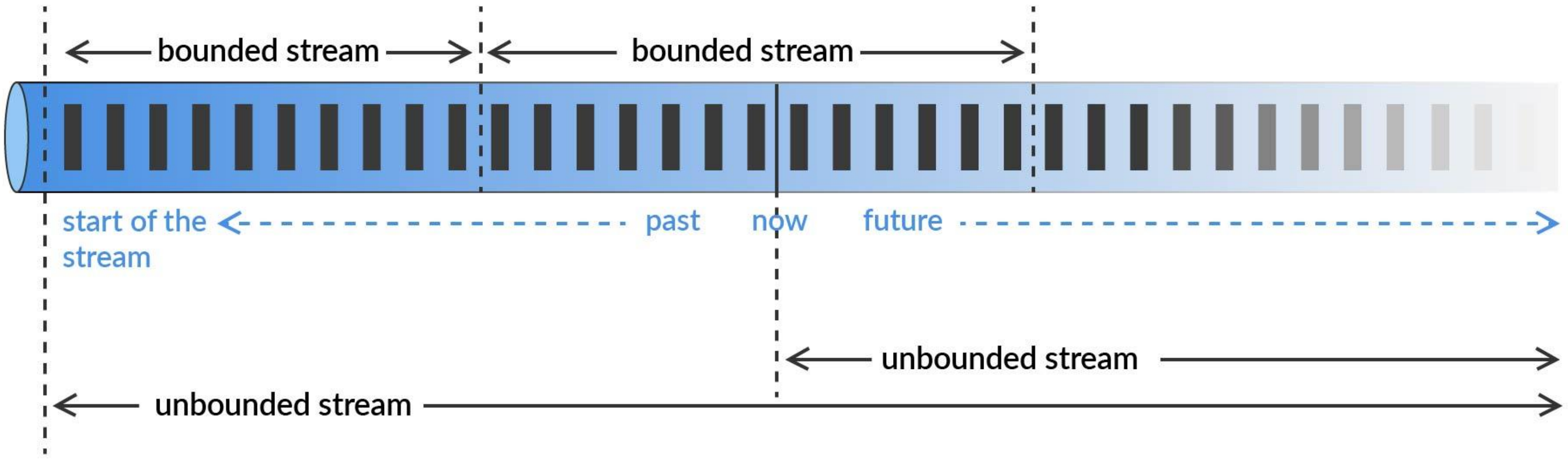
- Process event as it arrives.
- No/minimal latency between the arrival & process time of an event

Streaming as a special case of batch processing



Micro-batching/ Fast batching: Incoming records in every few seconds are batched together and then processed in a single mini batch with delay of few seconds.

Batch as a special case of streaming



- **Unbounded streams** have a start but no end is defined. Events are processed continuously, i.e., events get handled right after the ingestion. Ordered ingestion is crucial for completion of an event.

- **Bounded streams** have a defined start and end. Events can be processed by ingesting all data before performing any computations. Ordered ingestion is not required, because a bounded data set can always be sorted.

APACHE FLINK

01 Open source stream processing framework

02 Supports both batch and stream processing

03 Processes millions of record per second in real time

04 Provides low latency and high throughput

upGrad

APACHE FLINK vs APACHE SPARK

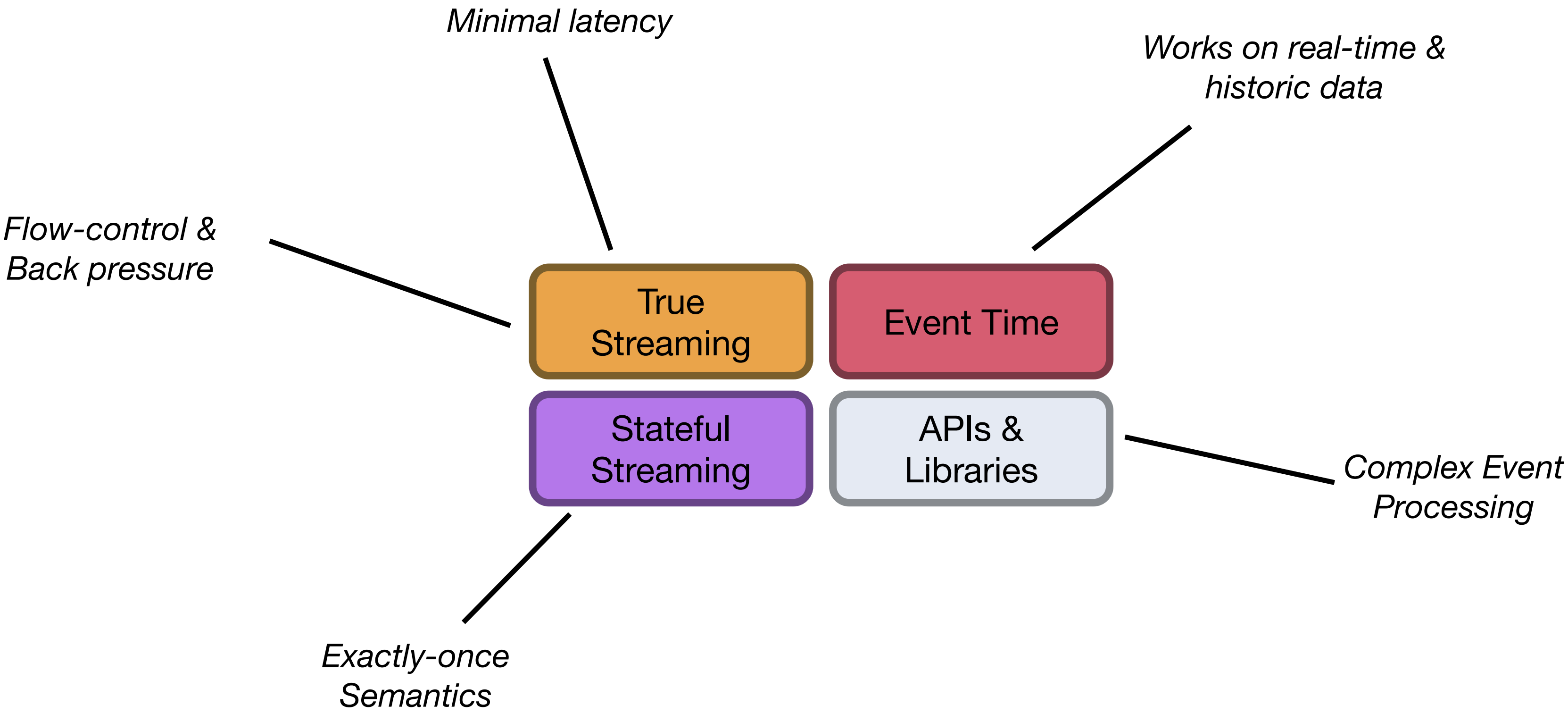
Flink	APACHE Spark™
True real-time processing.	Near real-time processing.
Streaming computation model is based on windowing and checkpointing.	Streaming computation model is based on micro-batching.
Relatively new ecosystem & have less third party libraries.	Mature product as compared to Flink.

Flink	APACHE Spark™
Flink has efficient automatic memory management.	There has been concerns about memory management in big clusters.
Languages supported: Java, Scala, Python*.	Languages supported: Java, Scala, Python, SQL, R.

upGrad

WHY APACHE FLINK?

APACHE FLINK: HIGHLIGHTS



Minimal latency

Works on real-time & historic data

Flow-control & Back pressure

True Streaming

Event Time

Stateful Streaming

APIs & Libraries

Complex Event Processing

Exactly-once Semantics

CASE STUDY: ALIBABA

Alibaba Group

Uses a fork of Flink called Blink
to optimize search rankings
in real time

Cluster	State	Events	TPS
More than 10,000 servers	Petabytes	Trillions of transactions/day	472 Million TPS

Source: Alibaba Cloud Blog, 2019

FLINK ECOSYSTEM

IN THIS SEGMENT

Learn about the structure of component stack of Flink.

Understand functionality of each component.

Learn about Flink runtime environment.

COMPONENT STACK

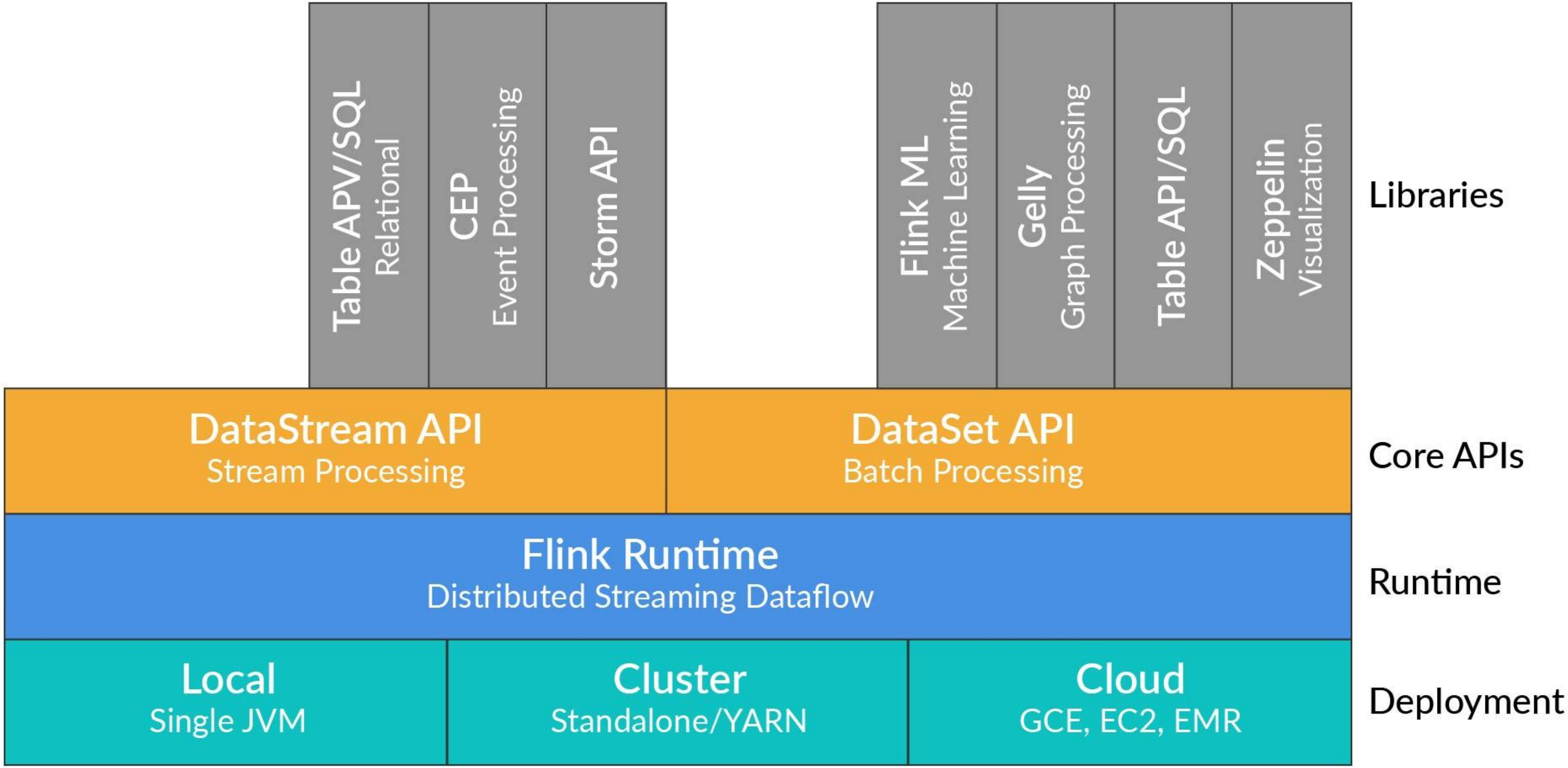


Table APV/SQL
Relational

CEP
Event Processing

Storm API

Flink ML
Machine Learning

Gelly
Graph Processing

Table API/SQL

Zeppelin
Visualization

Libraries

DataStream API
Stream Processing

DataSet API
Batch Processing

Core APIs

Flink Runtime
Distributed Streaming Dataflow

Runtime

Local
Single JVM

Cluster
Standalone/YARN

Cloud
GCE, EC2, EMR

Deployment

COMPONENT STACK

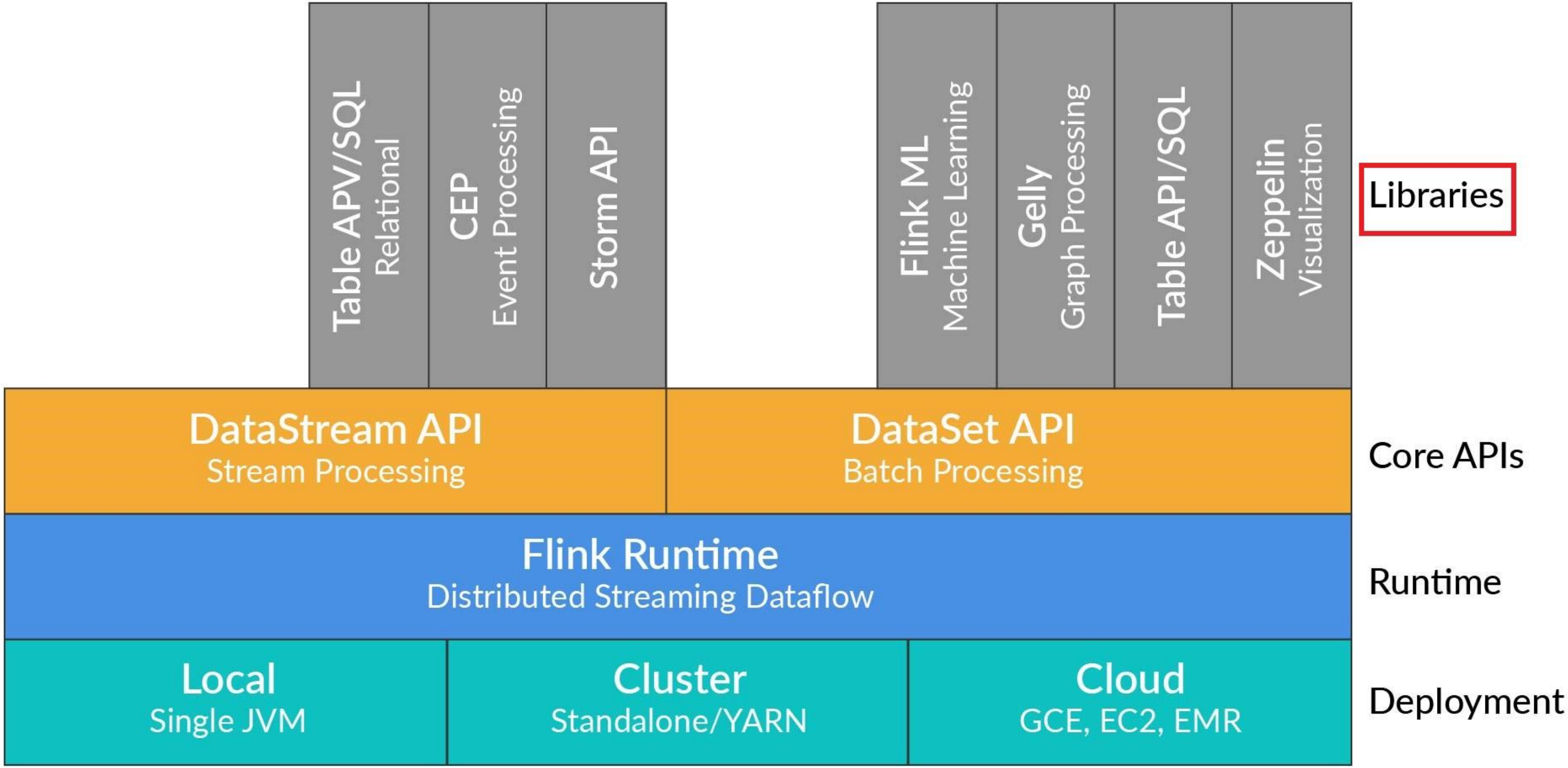


Table APV/SQL
Relational

CEP
Event Processing

Storm API

Flink ML
Machine Learning

Gelly
Graph Processing

Table API/SQL

Zeppelin
Visualization

Libraries

DataStream API
Stream Processing

DataSet API
Batch Processing

Core APIs

Flink Runtime
Distributed Streaming Dataflow

Runtime

Local
Single JVM

Cluster
Standalone/YARN

Cloud
GCE, EC2, EMR

Deployment

COMPONENT STACK

- Multiple Libraries & APIs are available for flink which interacts with DataSet or DataStream APIs
- For Example:
  - **Table API/ SQL** for queries on logical tables
  - **Flink ML** for machine learning
  - **Gelly** for graph processing.

COMPONENT STACK

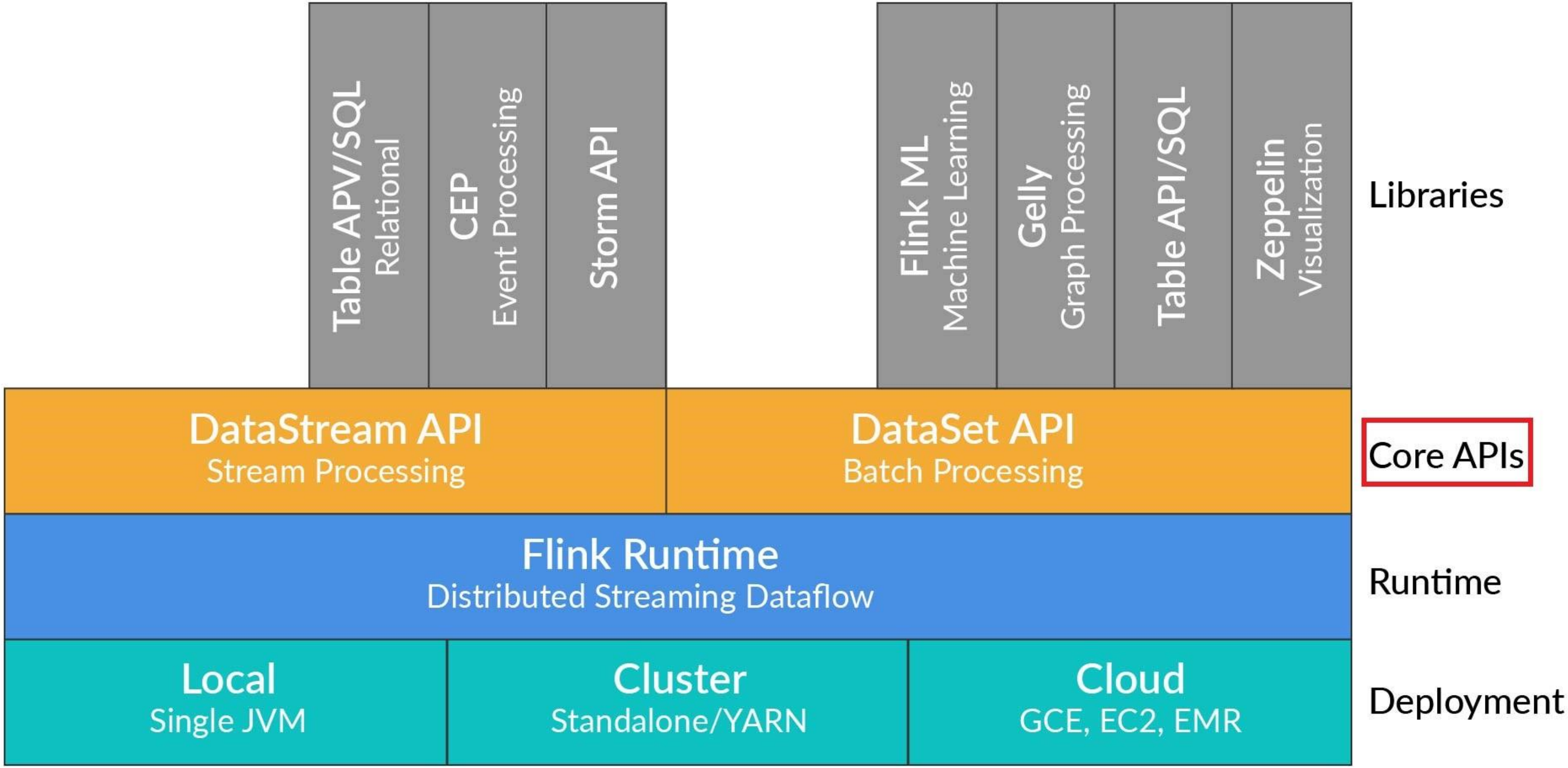


Table APV/SQL
Relational

CEP
Event Processing

Storm API

Flink ML
Machine Learning

Gelly
Graph Processing

Table API/SQL

Zeppelin
Visualization

Libraries

DataStream API
Stream Processing

DataSet API
Batch Processing

Core APIs

Flink Runtime
Distributed Streaming Dataflow

Runtime

Local
Single JVM

Cluster
Standalone/YARN

Cloud
GCE, EC2, EMR

Deployment

COMPONENT STACK

- Both the **DataStream API** and the **DataSet API** generate JobGraphs.

- The DataSet API uses an optimizer to determine the optimal plan for the program, while the DataStream API uses a stream builder.

COMPONENT STACK

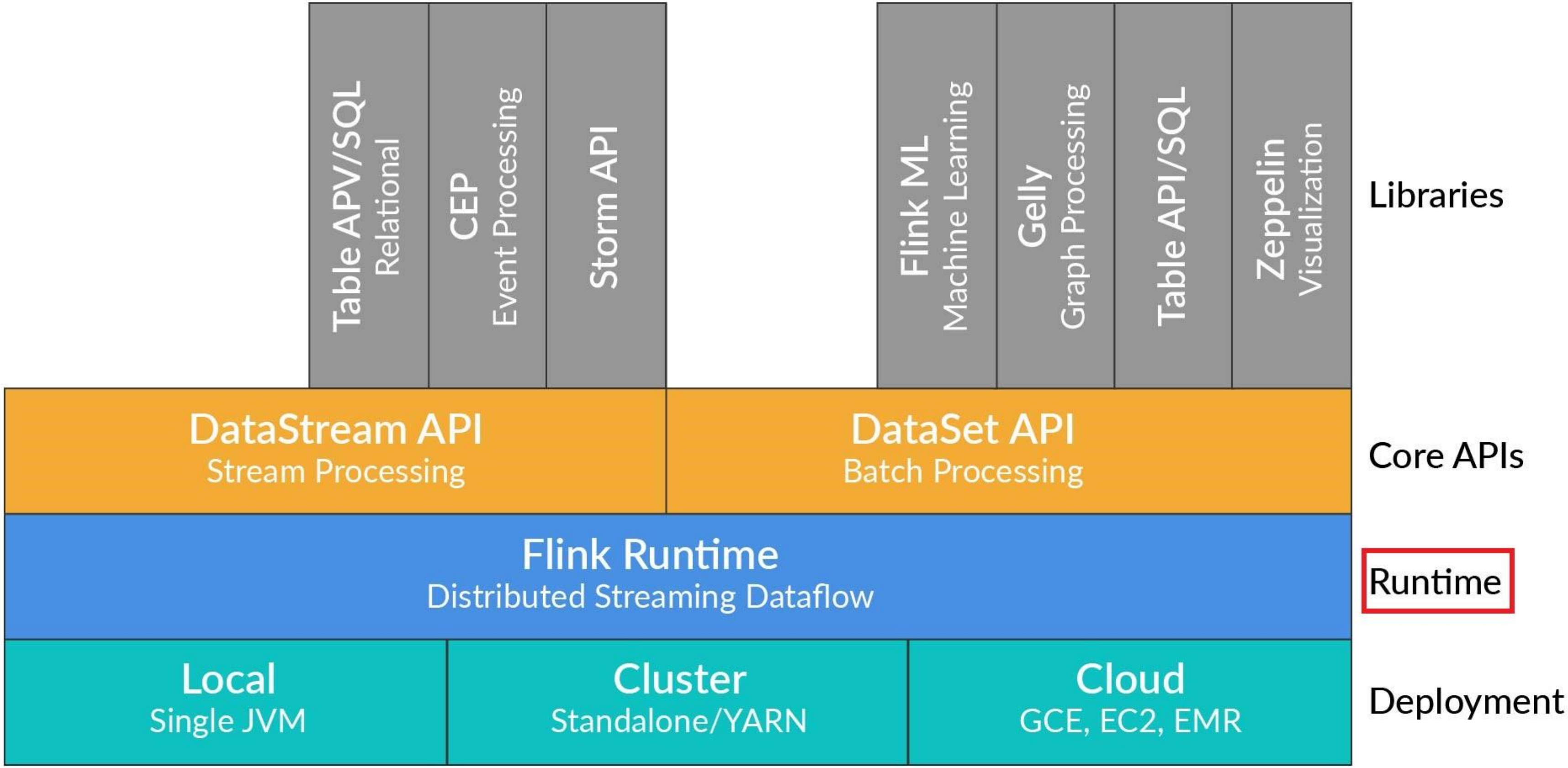


Table APV/SQL
Relational

CEP
Event Processing

Storm API

Flink ML
Machine Learning

Gelly
Graph Processing

Table API/SQL

Zeppelin
Visualization

Libraries

DataStream API
Stream Processing

DataSet API
Batch Processing

Core APIs

Flink Runtime
Distributed Streaming Dataflow

Runtime

Local
Single JVM

Cluster
Standalone/YARN

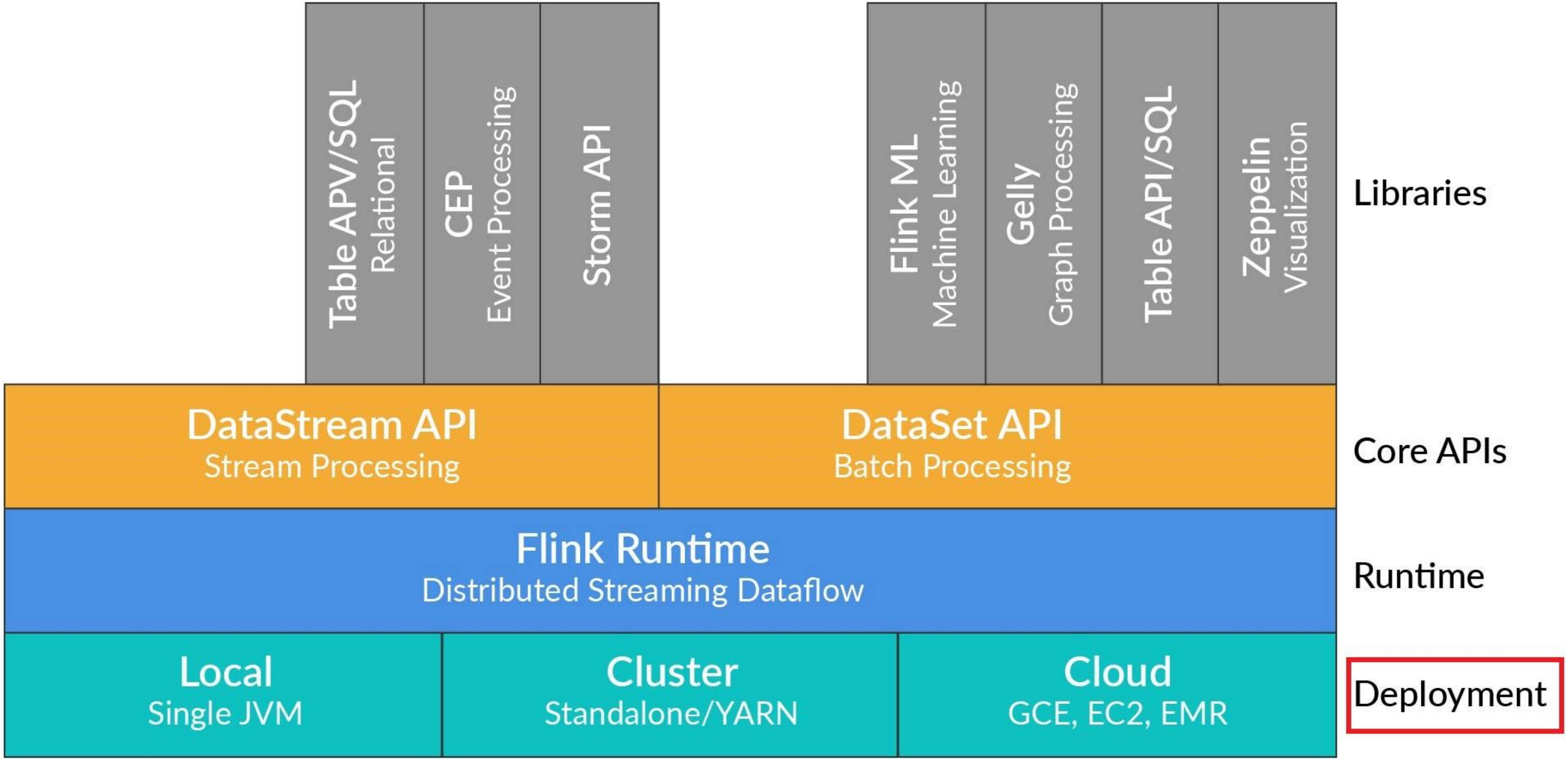
Cloud
GCE, EC2, EMR

Deployment

COMPONENT STACK

- **The runtime layer** receives a program in the form of a JobGraph.
- A **JobGraph** is a data flow with tasks that consume and produce data streams.

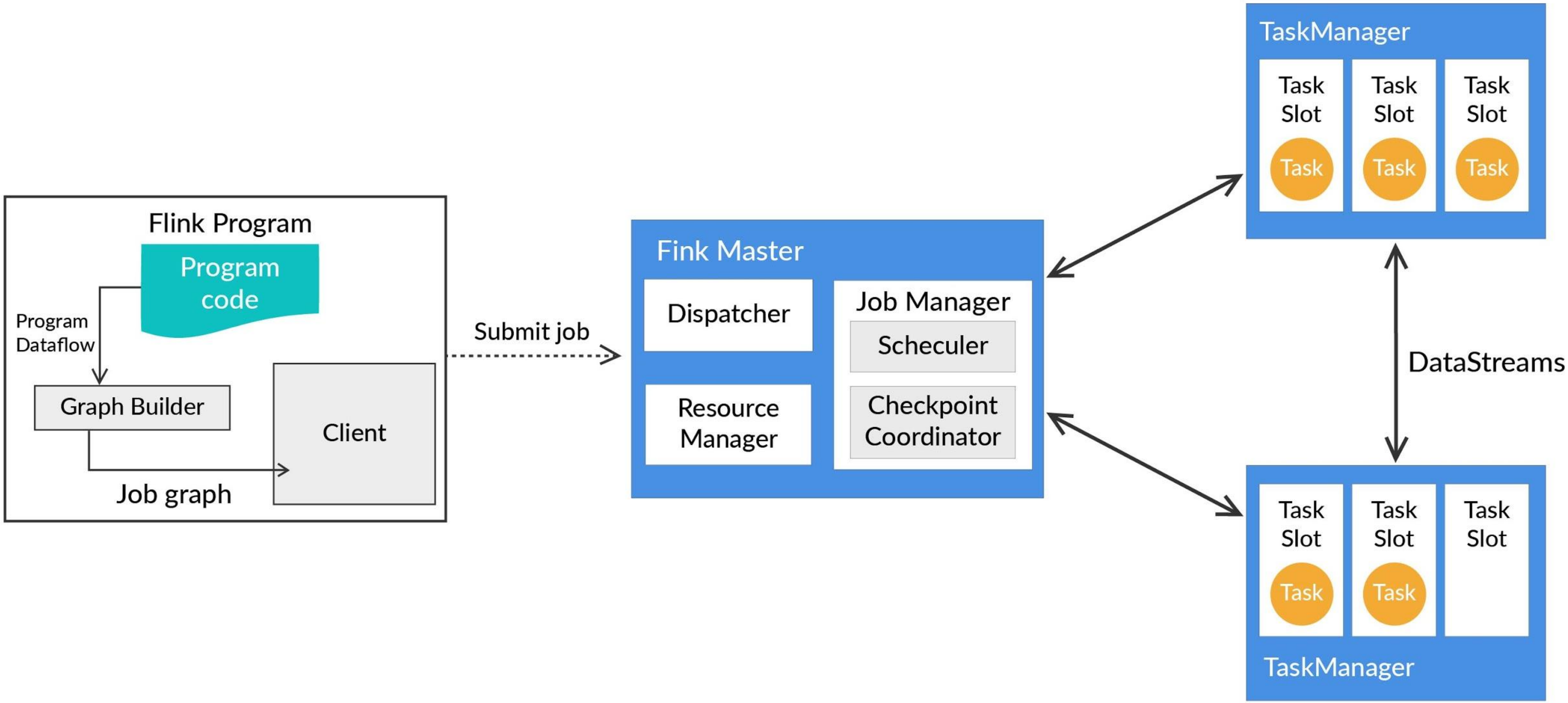
COMPONENT STACK



COMPONENT STACK

- There are various **Deployment options** available in Flink (e.g., local, cluster, YARN etc), which executes the JobGraph.

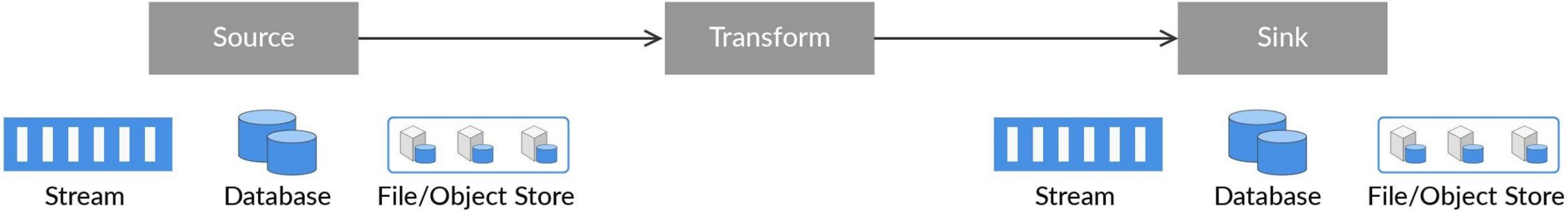
RUNTIME ENVIRONMENT



upGrad

FLINK PROGRAMMING MODEL

FLINK PROGRAMMING MODEL



Source

Transform

Sink

Stream

Database

File/Object Store

Stream

Database

File/Object Store

upGrad

FLINK USE CASES

IN THIS SEGMENT

Understand the common applications which are powered by Flink.

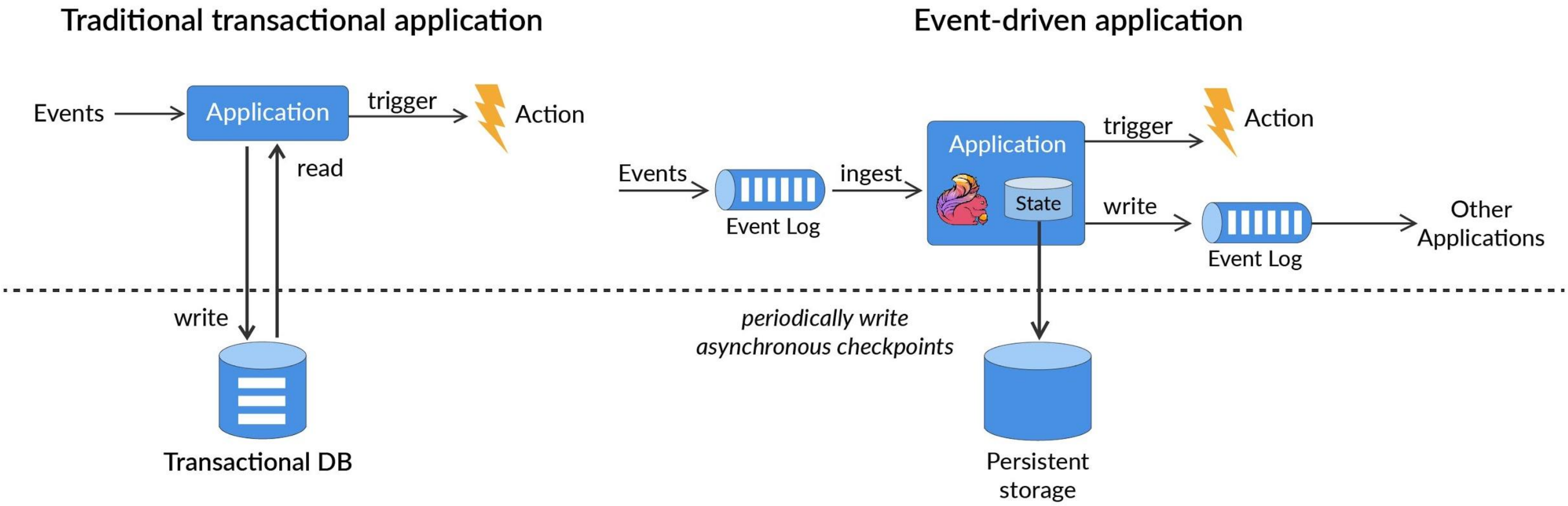
Learn about event-driven and data analytics applications.

Learn about data pipeline jobs.

Look at some companies powered by Flink.

EVENT DRIVEN APPLICATIONS

Detect events as they occur, and then reacts by triggering computations, state updates or external actions.



EVENT-DRIVEN APPLICATIONS

Fraud Detection

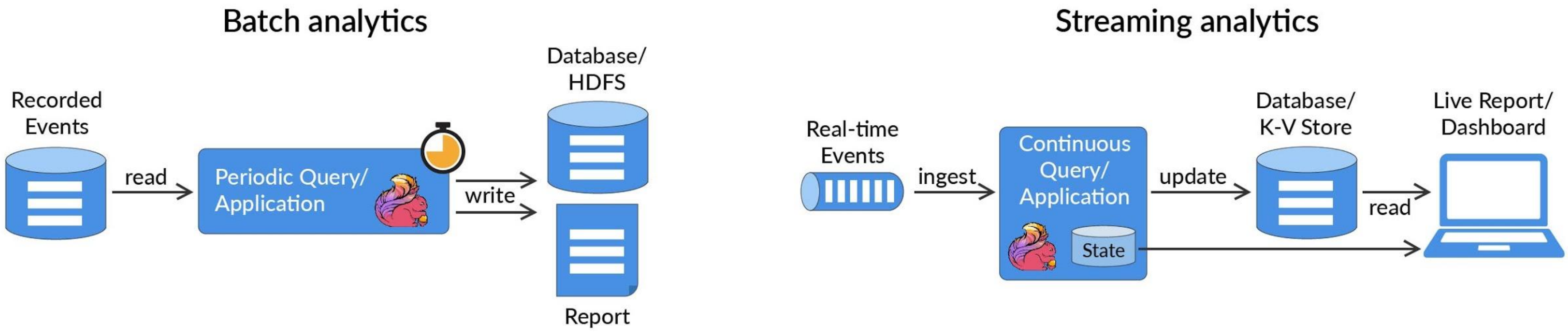
Anomaly Detection

Rule-based Alerting

Business Process Monitoring

DATA ANALYTICS APPLICATIONS

Traditionally, analytics are performed as periodic batch queries on bounded data set of recorded events. With a sophisticated stream processing engine, analytics can also be performed in a real-time fashion.



DATA-ANALYTICS APPLICATIONS

01 Quality monitoring of Telecom networks

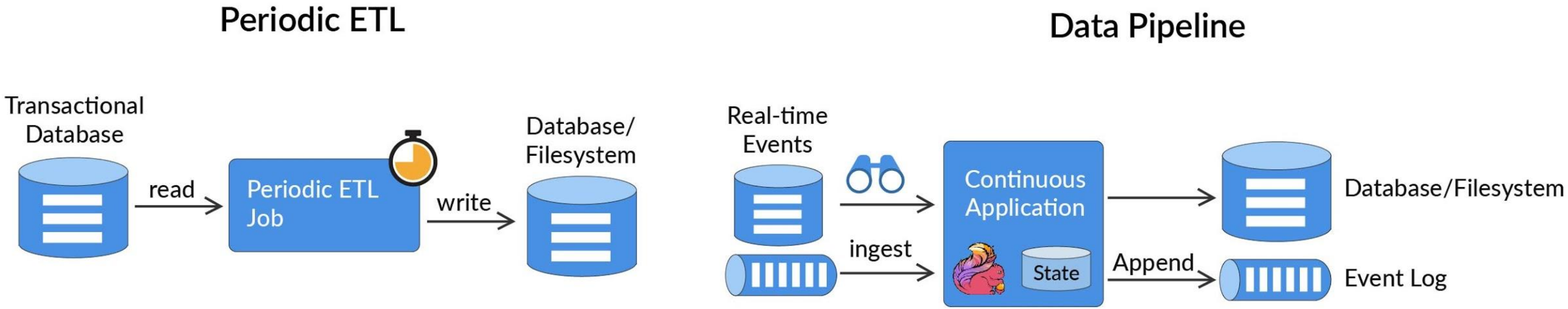
02 Analysis of product experiments in mobile applications

03 Data analytics in consumer technology

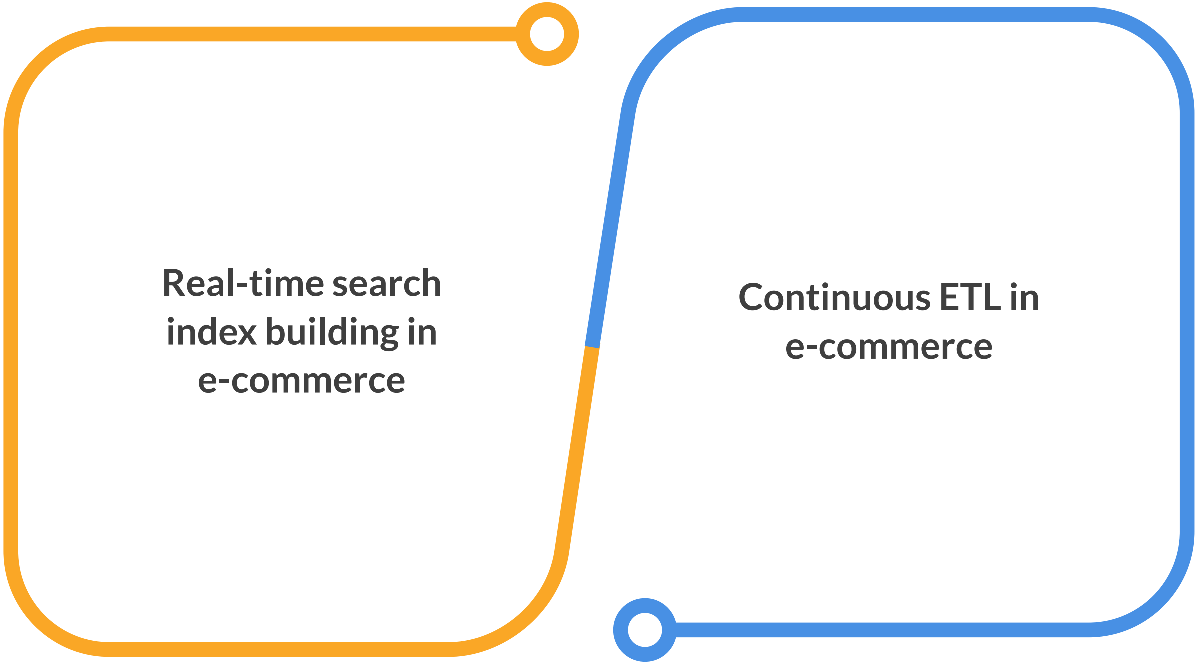
04 Graph analysis

DATA PIPELINE(ETL) JOBS

Extract-transform-load (ETL) is a common approach in batch systems to convert and move data between storage systems. In the streaming world, this is done through data pipeline jobs



DATA PIPELINE JOBS



Real-time search index building in e-commerce

Continuous ETL in e-commerce

POWERED BY FLINK

aws

Used in Amazon Kinesis Data Analytics

bouygues TELECOM

Real-time monitoring & analysis

King

Build real-time analytics dashboard

Razorpay

AI feature generation & model serving in real-time

UBER

Streaming analytics platform AthenaX

Real-time experiment analytics

gojek

Real-time data aggregation platform

lyft

Generate features for machine learning