Event Based Processing

Batch and real-time data pipelines
using Kafka and Flink

Prototype Scope

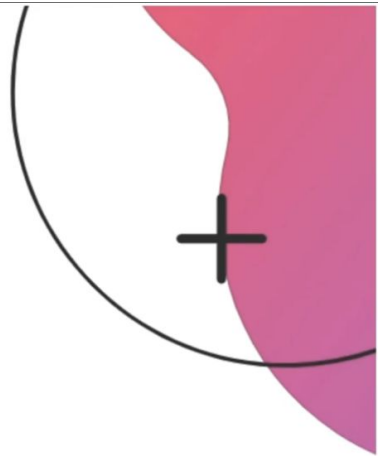
Model batch and real-time ingestions using Kafka

Scale Flink cluster

Optional: write into Iceberg on S3

Discover any open questions

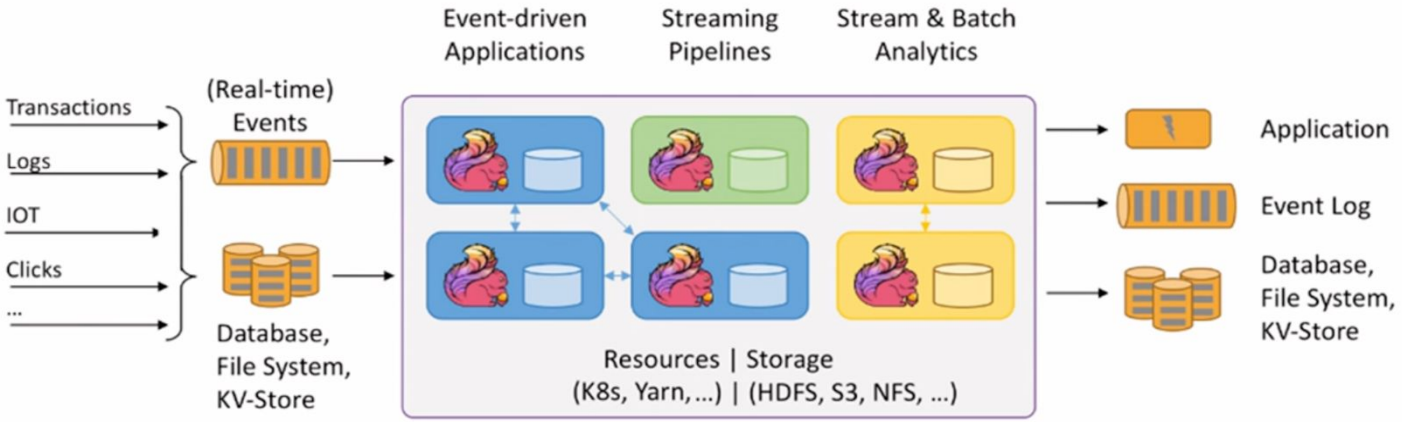
Why Flink?



- Fault tolerant
  - Snapshots
- Advanced event time handling
  - Watermark - heuristic solution for out of ordered events
- Highly Scalable
  - Divide compute intensive jobs
  - Allocate computing resources across task slots

Source: https://www.udemy.com/course/apache-flink-with-scala-3/

Stateful Computations over Data Streams

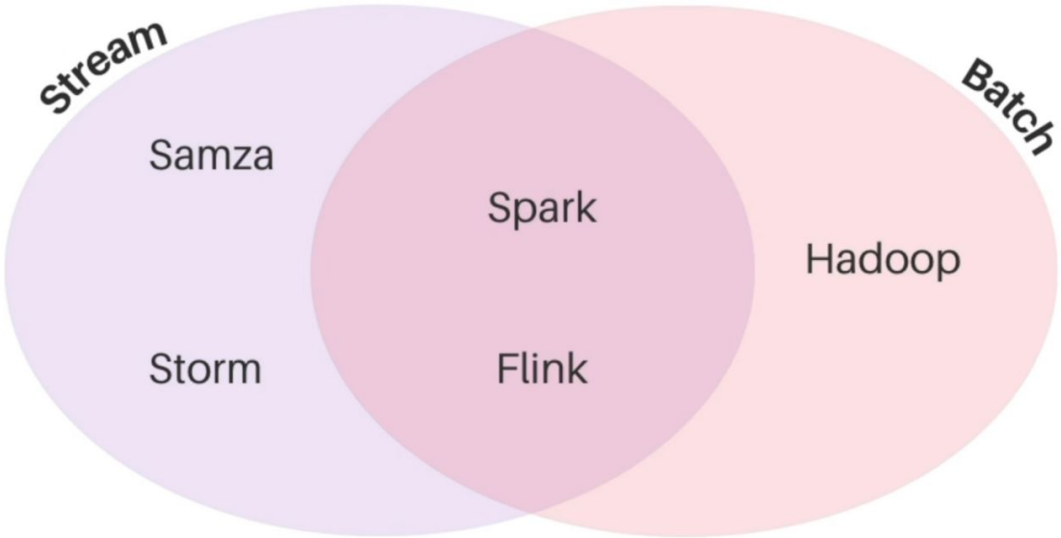


Apache Flink is a framework and distributed processing engine for stateful computations over *unbounded and bounded* data streams. Flink has been designed to run in *all common cluster environments*, perform computations at *in-memory speed* and at *any scale*.

- ✓ Fault tolerance
- ✓ Scalability
- ✓ Low Latency

Source: https://www.udemy.com/course/flink-mastery/

Big Data Engines



Source: https://www.udemy.com/course/apache-flink-with-scala-3/

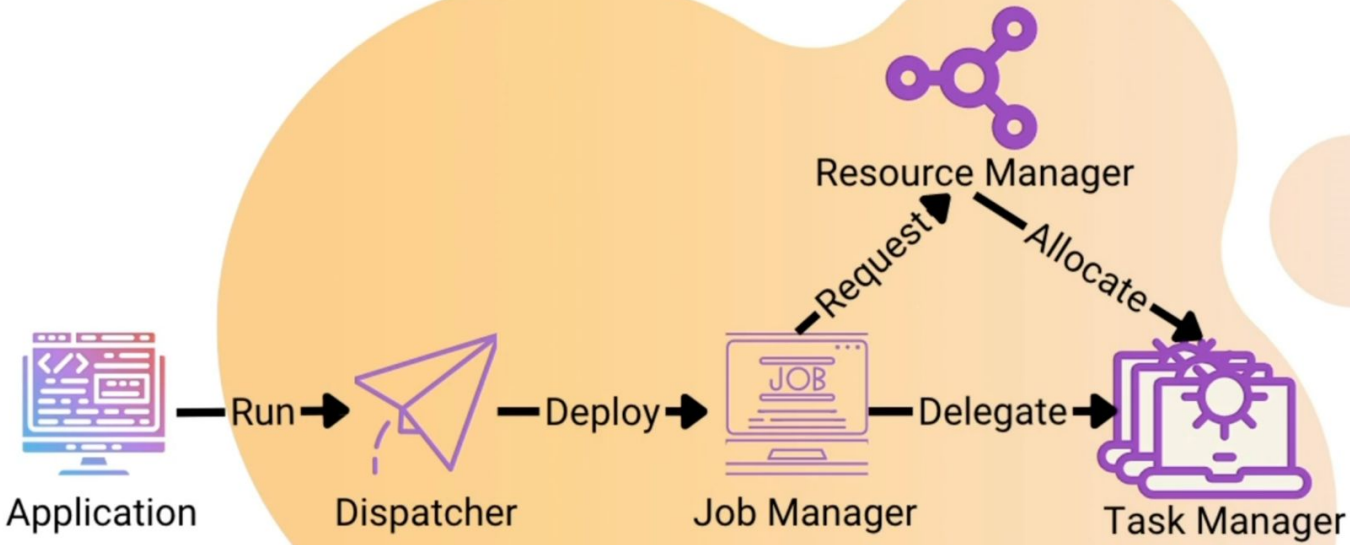
Flink Cluster Architecture

+ **Job Manager**
  - Resource Manager
    - manages task slots
  - Dispatcher
    - REST API for submitting Flink applications
    - starts new Job Masters
  - Job Master
    - manage the execution of a single job process

+ **Task Manager**
  - execute the tasks of a data stream
  - resource unit is called task slot

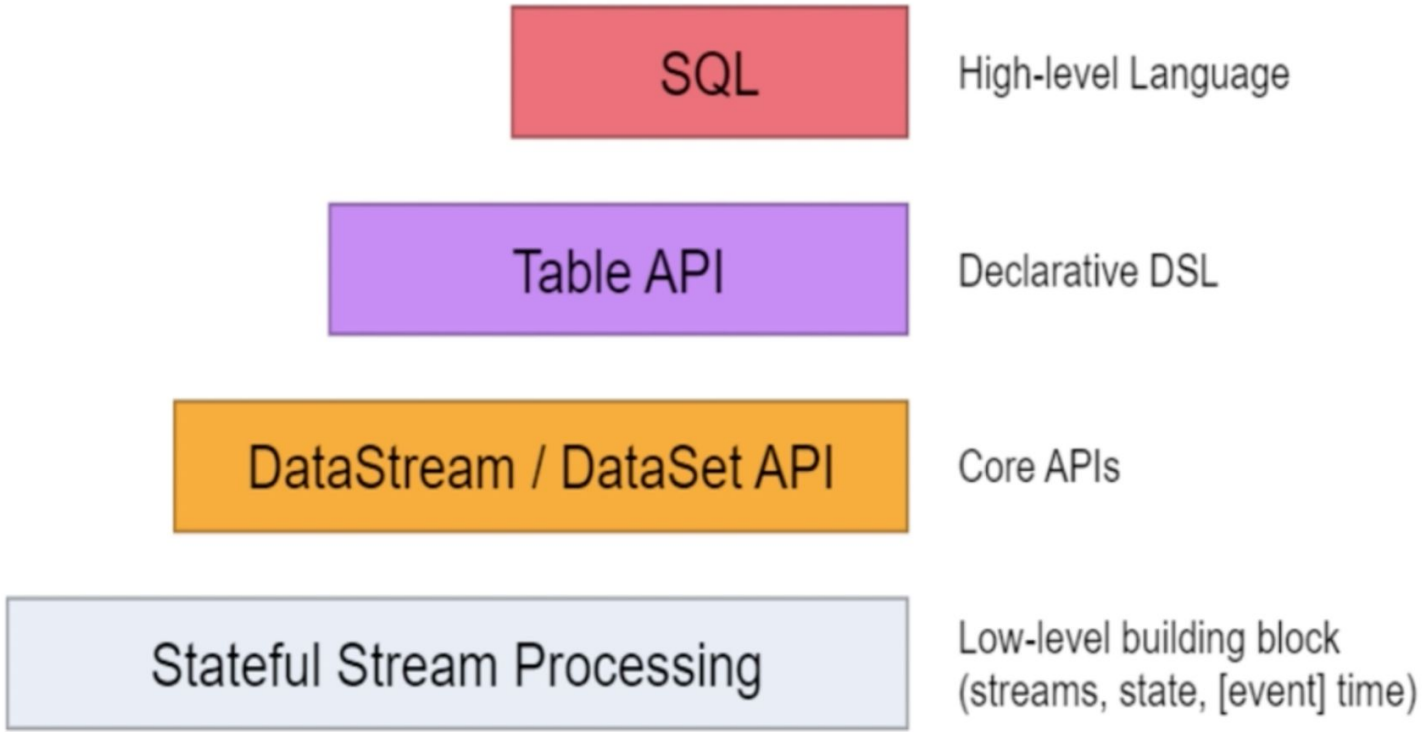
Source: https://www.udemy.com/course/apache-flink-with-scala-3/

Flink Job Lifecycle



Source: https://www.udemy.com/course/apache-flink-with-scala-3/

Flink's APIs

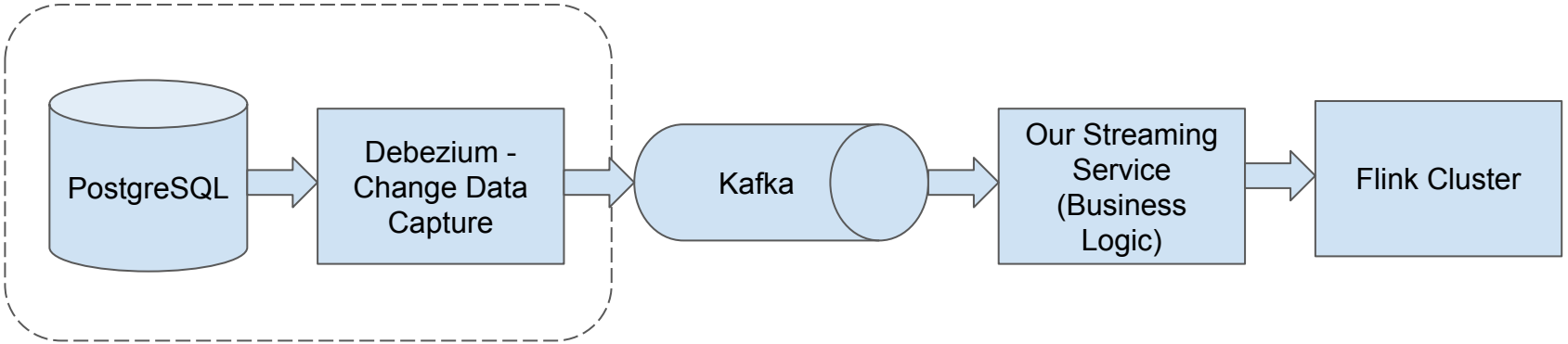


Deployment Mode

- Application Mode
  - Creates a cluster per submitted application
- Session Mode
  - Applications submitted to an already running cluster
  - Compete for resources

Source: https://www.udemy.com/course/apache-flink-with-scala-3/

Components/Architecture



Lessons Learned

Provisioning Flink Operator and Cluster was simple.

- Helm to provision the operator.
- Flink depends on other tools like cert-manager.
- Make sure you allocate enough memory and cpu to get consistent performance. If not, you will get a lot of pod restarts and OOMKilled errors.

Can use session mode to deploy persistent clusters, with UI for monitoring, per namespace/tenant.

Using iceberg and other packages requires additional jar files and versions.

Setting up the environment was easy. Running jobs took a lot more time to learn. See jar versioning hell comment above.

Troubleshooting was difficult at times. Some error messages didn’t match the actual problem. Researching a problem took longer than normal.

FlinkSQL is easy to use if you already know regular SQL.

Lessons Learned

Related to versioning hell. Do not use the most recent versions of anything. A lot of experimentation to find the right compatible versions.

Server Response Message:

```
org.apache.flink.runtime.rest.handler.RestHandlerException: Could not execute application.
    at org.apache.flink.runtime.webmonitor.handlers.JarRunHandler.lambda$handleRequest$1(JarRunHandler.java:114)
    at java.base/java.util.concurrent.CompletableFuture.uniHandle(Unknown Source)
    at java.base/java.util.concurrent.CompletableFuture$UniHandle.tryFire(Unknown Source)
    at java.base/java.util.concurrent.CompletableFuture.postComplete(Unknown Source)
    at java.base/java.util.concurrent.CompletableFuture$AsyncSupply.run(Unknown Source)
    at java.base/java.lang.Thread.run(Unknown Source)
Caused by: java.util.concurrent.CompletionException: java.lang.NoClassDefFoundError: org/apache/flink/connector/kafka/source/KafkaSource
    at java.base/java.util.concurrent.CompletableFuture.encodeThrowable(Unknown Source)
    at java.base/java.util.concurrent.CompletableFuture.completeThrowable(Unknown Source)
    ... 2 more
Caused by: java.lang.NoClassDefFoundError: org/apache/flink/connector/kafka/source/KafkaSource
    at com.example.FlinkTestJob.main(FlinkTestJob.java:27)
    at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
    at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke(Unknown Source)
    at java.base/jdk.internal.reflect.DelegatingMethodAccessorImpl.invoke(Unknown Source)
    at java.base/java.lang.reflect.Method.invoke(Unknown Source)
    at org.apache.flink.client.program.PackagedProgram.callMainMethod(PackagedProgram.java:355)
    at org.apache.flink.client.program.PackagedProgram.invokeInteractiveModeForExecution(PackagedProgram.java:222)
    at org.apache.flink.client.ClientUtils.executeProgram(ClientUtils.java:108)
    at org.apache.flink.client.deployment.application.DetachedApplicationRunner.tryExecuteJobs(DetachedApplicationRunner.java:84)
    at org.apache.flink.client.deployment.application.DetachedApplicationRunner.run(DetachedApplicationRunner.java:70)
    at org.apache.flink.runtime.webmonitor.handlers.JarRunHandler.lambda$handleRequest$0(JarRunHandler.java:108)
    ... 2 more
Caused by: java.lang.ClassNotFoundException: org.apache.flink.connector.kafka.source.KafkaSource
    at java.base/java.net.URLClassLoader.findClass(Unknown Source)
    at java.base/java.lang.ClassLoader.loadClass(Unknown Source)
    at org.apache.flink.util.FlinkUserCodeClassLoader.loadClassWithoutExceptionHandling(FlinkUserCodeClassLoader.java:67)
    at org.apache.flink.util.ChildFirstClassLoader.loadClassWithoutExceptionHandling(ChildFirstClassLoader.java:65)
    at org.apache.flink.util.FlinkUserCodeClassLoader.loadClass(FlinkUserCodeClassLoader.java:51)
```

```
flink-jobmanager     | Caused by: org.apache.flink.client.program.ProgramInvocationException: The program's entry point class 'com.example.KafkaConsumerIceberg'
could not be loaded due to a linkage failure.
flink-jobmanager     |   at org.apache.flink.client.program.PackagedProgram.loadMainClass(PackagedProgram.java:493) ~[flink-dist-1.19.0.jar:1.19.0]
flink-jobmanager     |   at org.apache.flink.client.program.PackagedProgram.<init>(PackagedProgram.java:153) ~[flink-dist-1.19.0.jar:1.19.0]
flink-jobmanager     |   at org.apache.flink.client.program.PackagedProgram.<init>(PackagedProgram.java:65) ~[flink-dist-1.19.0.jar:1.19.0]
flink-jobmanager     |   at org.apache.flink.client.program.PackagedProgram$Builder.build(PackagedProgram.java:691) ~[flink-dist-1.19.0.jar:1.19.0]
flink-jobmanager     |   at org.apache.flink.runtime.webmonitor.handlers.utils.JarHandlerUtils$JarHandlerContext.toPackagedProgram(JarHandlerUtils.java:190) ~[fli
nk-dist-1.19.0.jar:1.19.0]
flink-jobmanager     |   ... 50 more
flink-jobmanager     | Caused by: java.lang.UnsupportedClassVersionError: com/example/KafkaConsumerIceberg has been compiled by a more recent version of the Java
 Runtime (class file version 61.0), this version of the Java Runtime only recognizes class file versions up to 55.0
```

Open Questions

How to scale Flink?
Best practices for batch processing using Flink.
Schema evolution from Kafka to Iceberg.
Serialization methods? There are serdes from Confluent, Spring. Do we need custom?
How to implement DMC?
How to avoid repeated initial loads?
How to provide a mechanism for cost control *by the user* - real-time, scheduled, adaptive (ML-based)?
How to predict cost for users?
Self-hosted Flink vs AWS-managed?
What do we store in raw? Kafka events, what format, for how long?
How to use watermarks in Flink? Best practices?
How do we measure performance?
State management in Flink - any cost, best practices?
How to design Kafka topics and partitions for data ingestion?

Resources

Playground Repo

https://github.com/RickZee/event-based-processing

Building Apache Flink Applications in Java by Confluent Youtube Playlist

https://www.youtube.com/playlist?list=PLa7VYi0yPIH0QEIcyvZE5p4zMR0ln4aAe

Learning Resources (PDF)

https://drive.google.com/file/d/1WkMI0EXg8_8EQrTRsMtcm1p8kBWdwwx7/view?usp=sharing