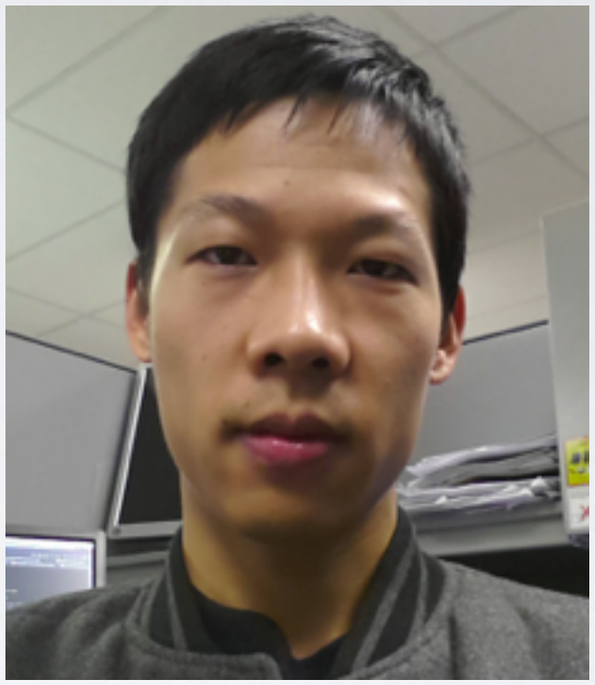
FLINK: UNIFIED STREAM ENGINE

Li Chengxiang

ABOUT ME

- Apache Hive Committer
- Apache Flink Committer
- Focus on large scale distributed computer engine and distributed SQL engine.



AGENDA

- Batch and Stream
- Core feature of Stream
- Advanced Flink Stream feature

BATCH AND STREAM


APEX

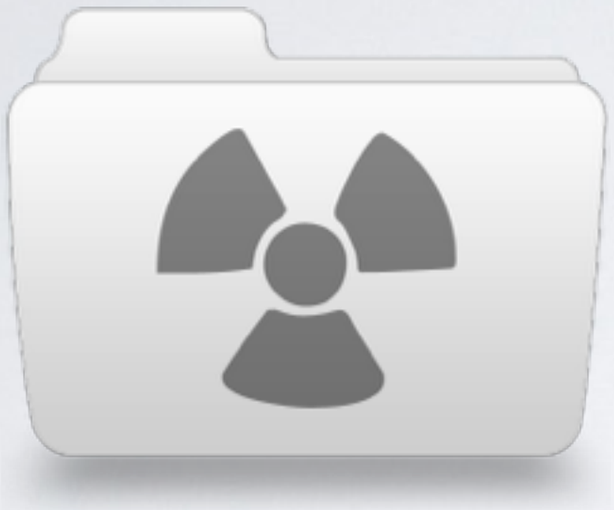

hadoop
MapReduce


APACHE
STORM™
Distributed • Resilient • Real-time


samza


Spark

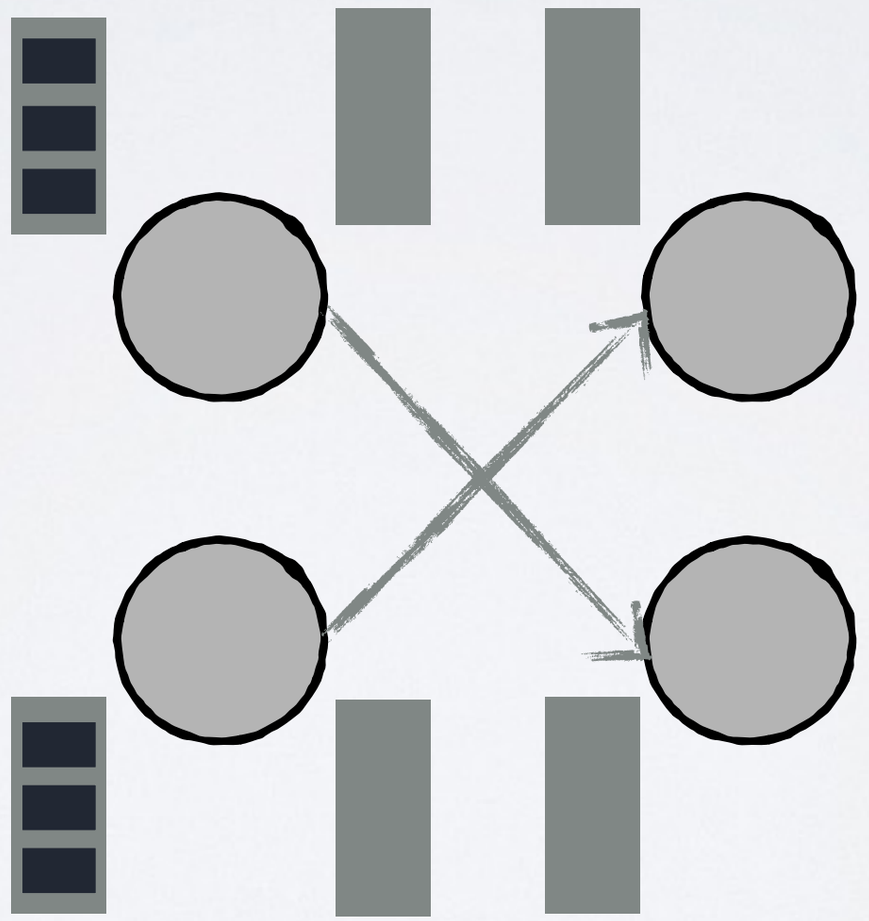
IT'S ALL ABOUT DATA



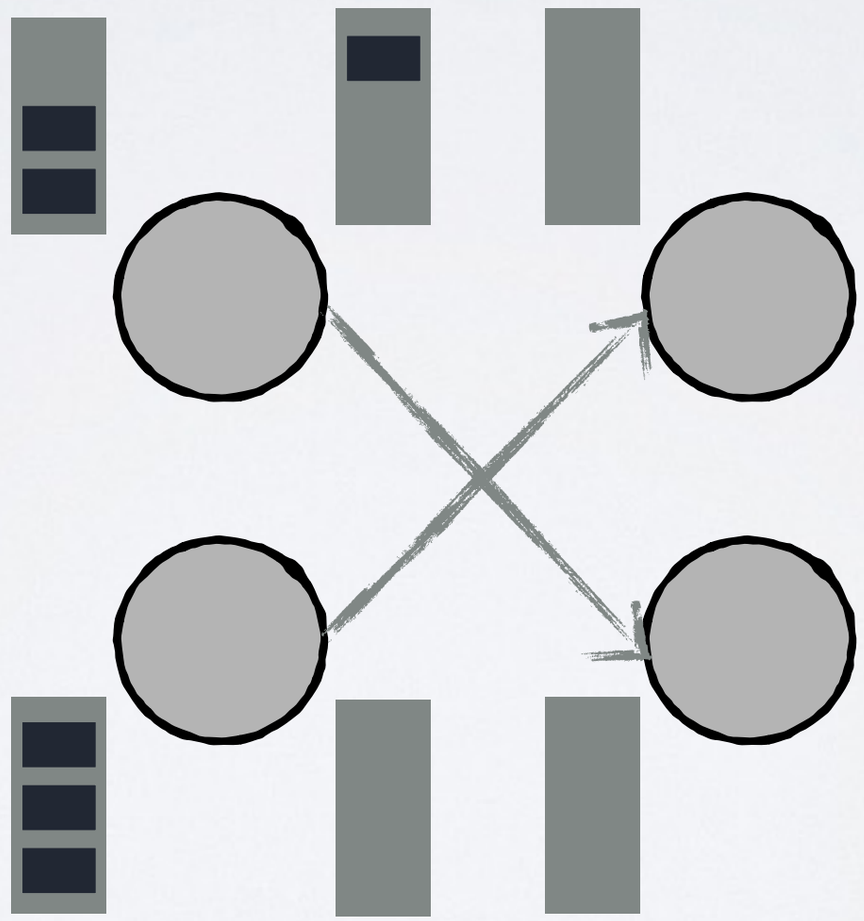
Bounded Data

Unbounded Data

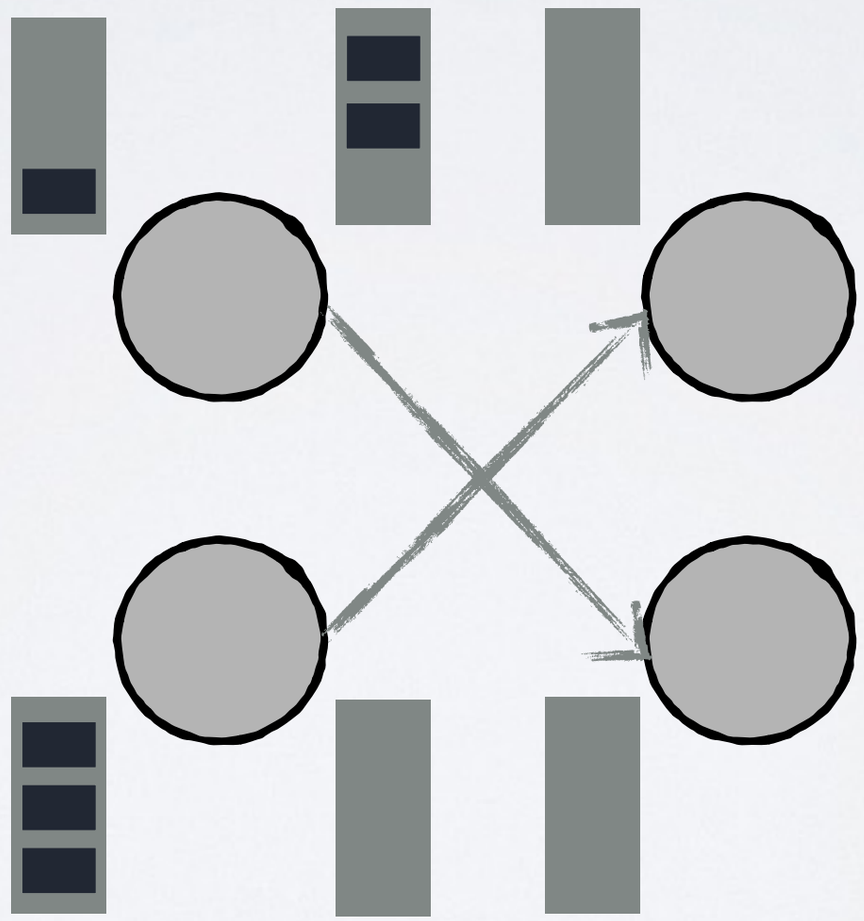
BATCH ENGINE



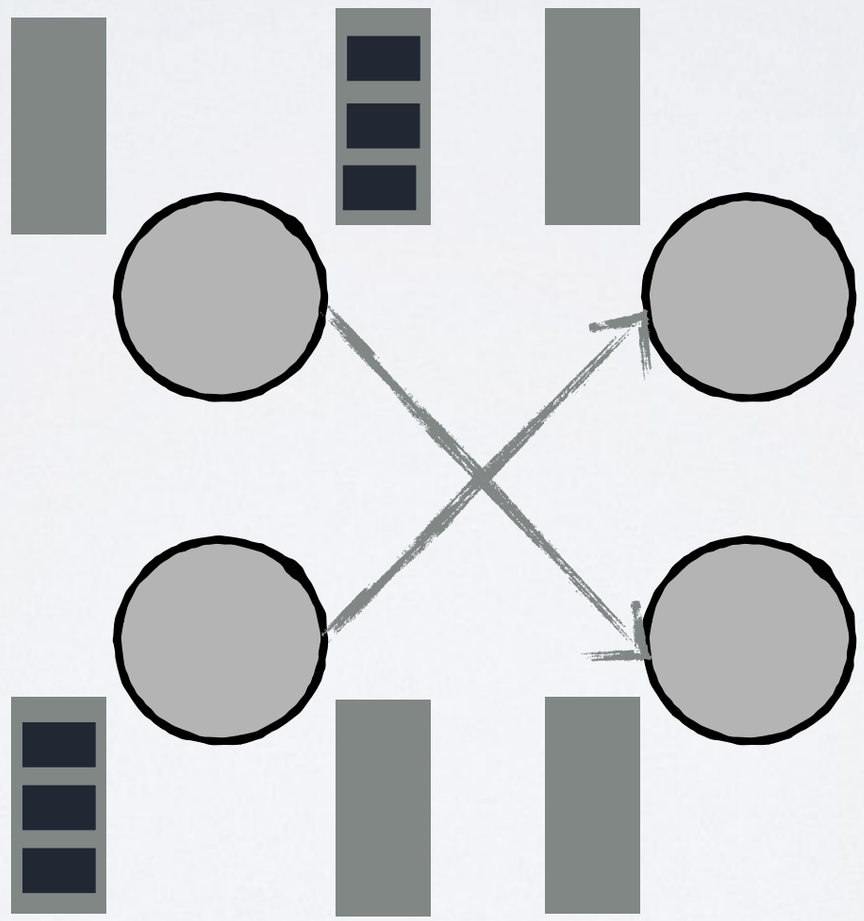
BATCH ENGINE



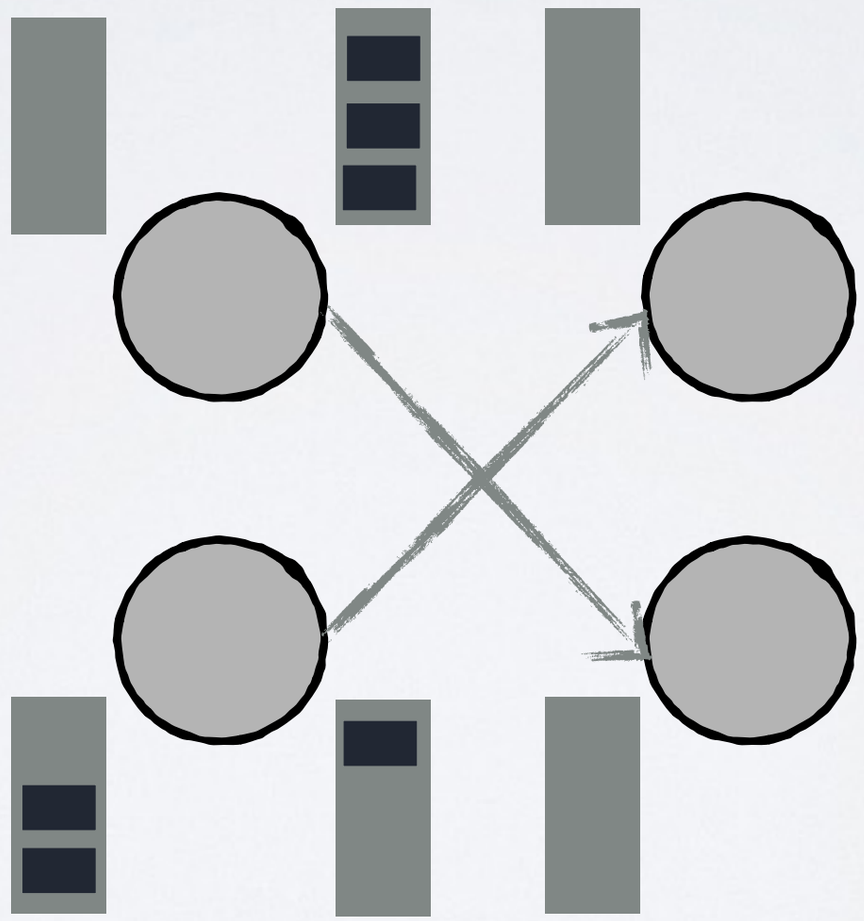
BATCH ENGINE



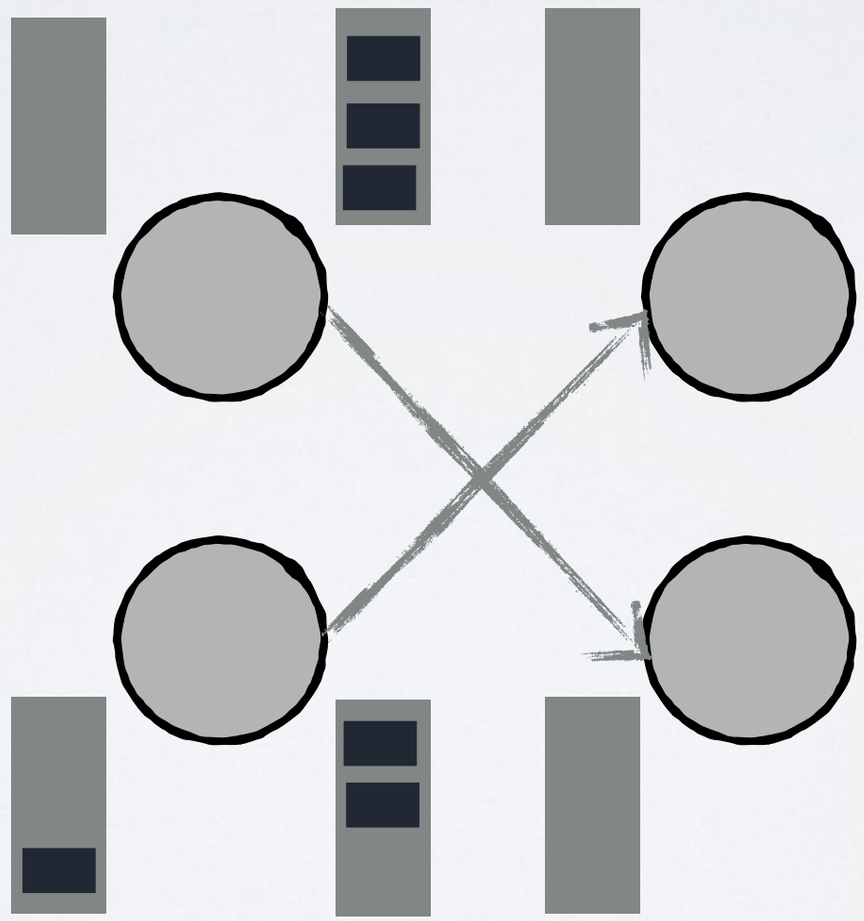
BATCH ENGINE



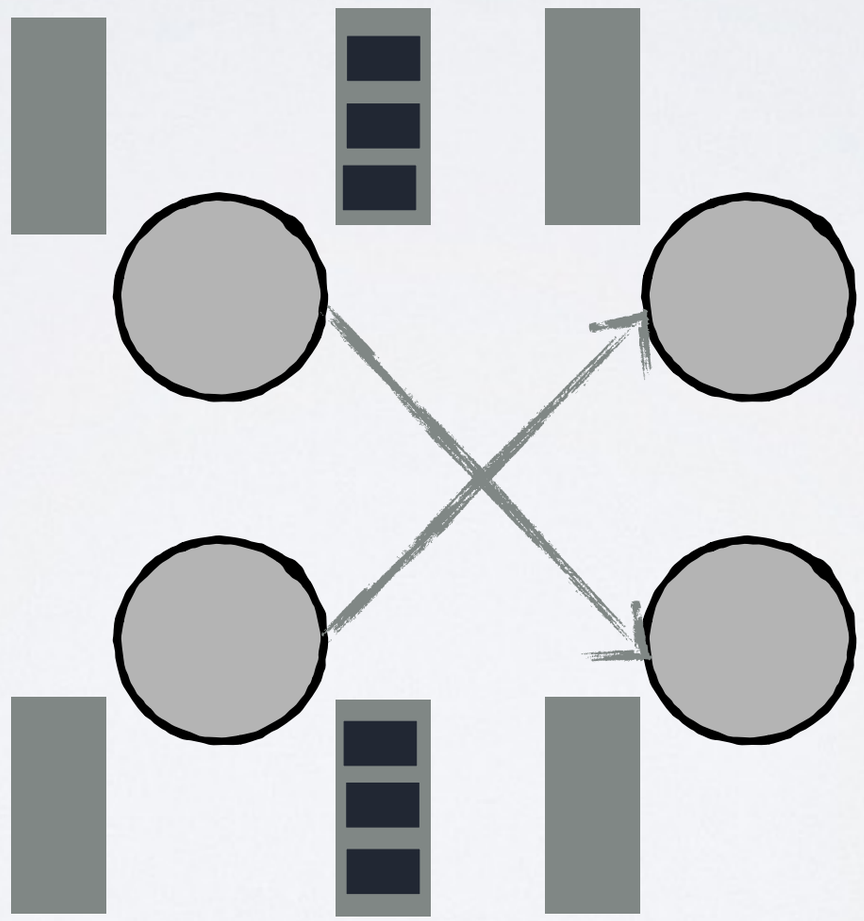
BATCH ENGINE



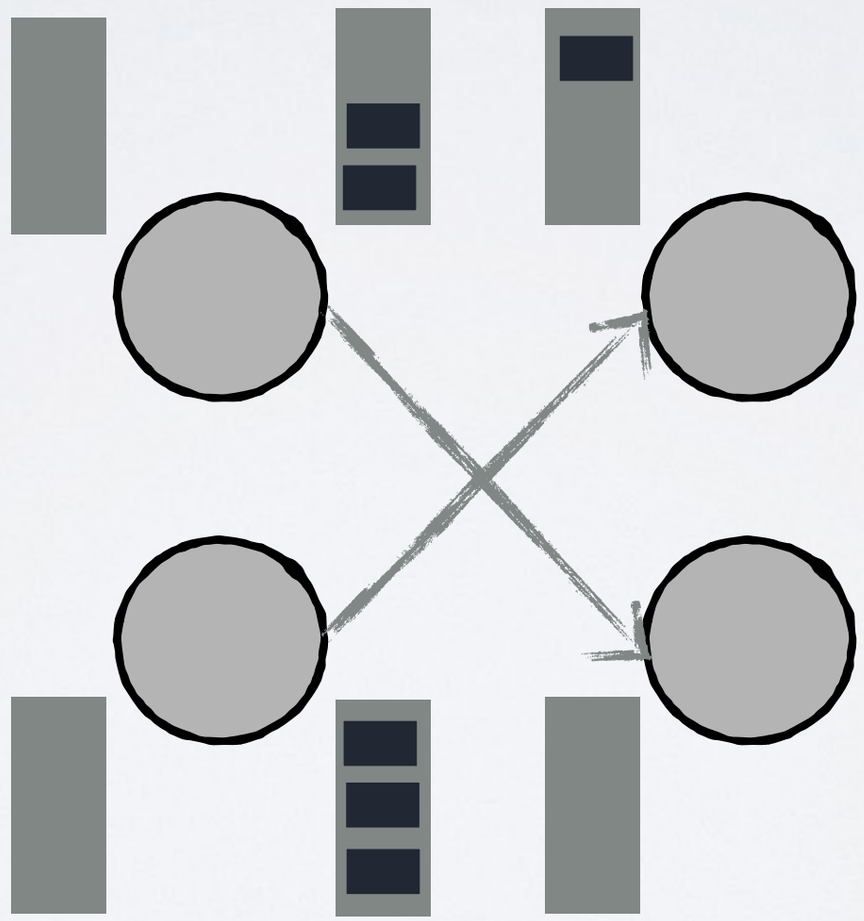
BATCH ENGINE



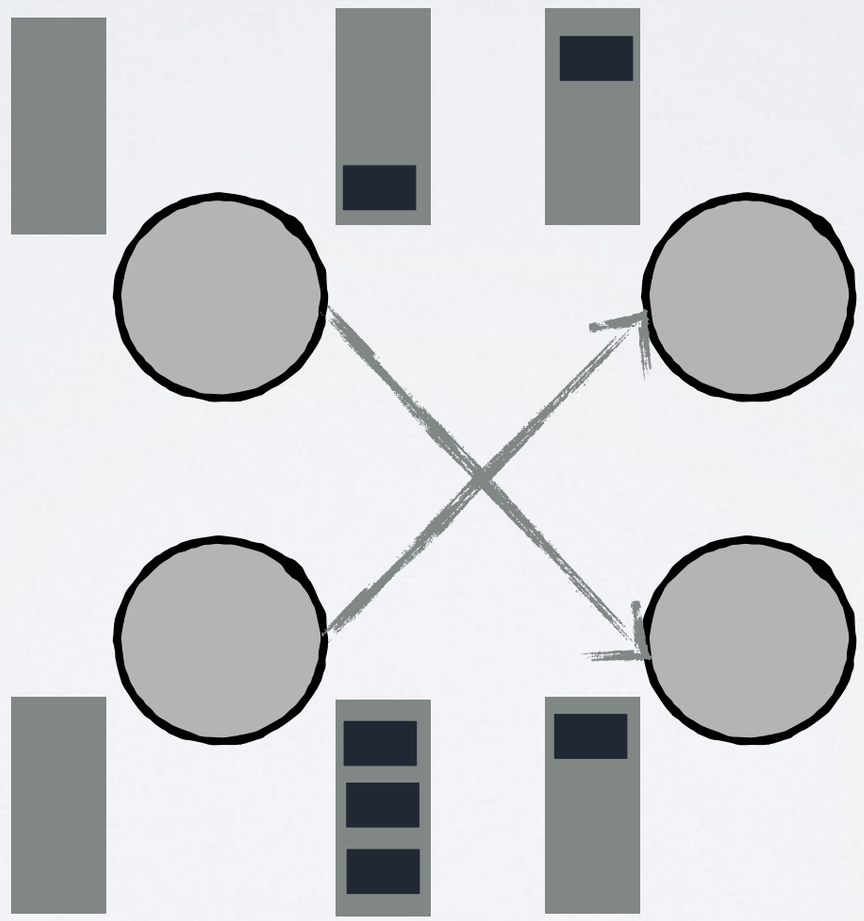
BATCH ENGINE



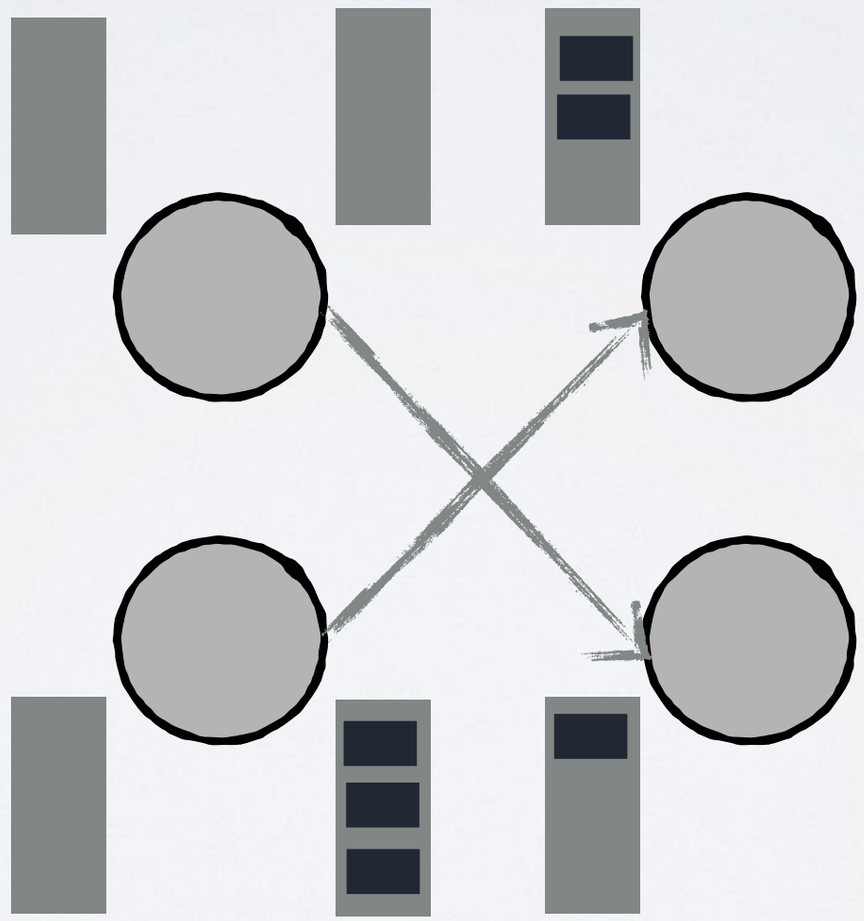
BATCH ENGINE



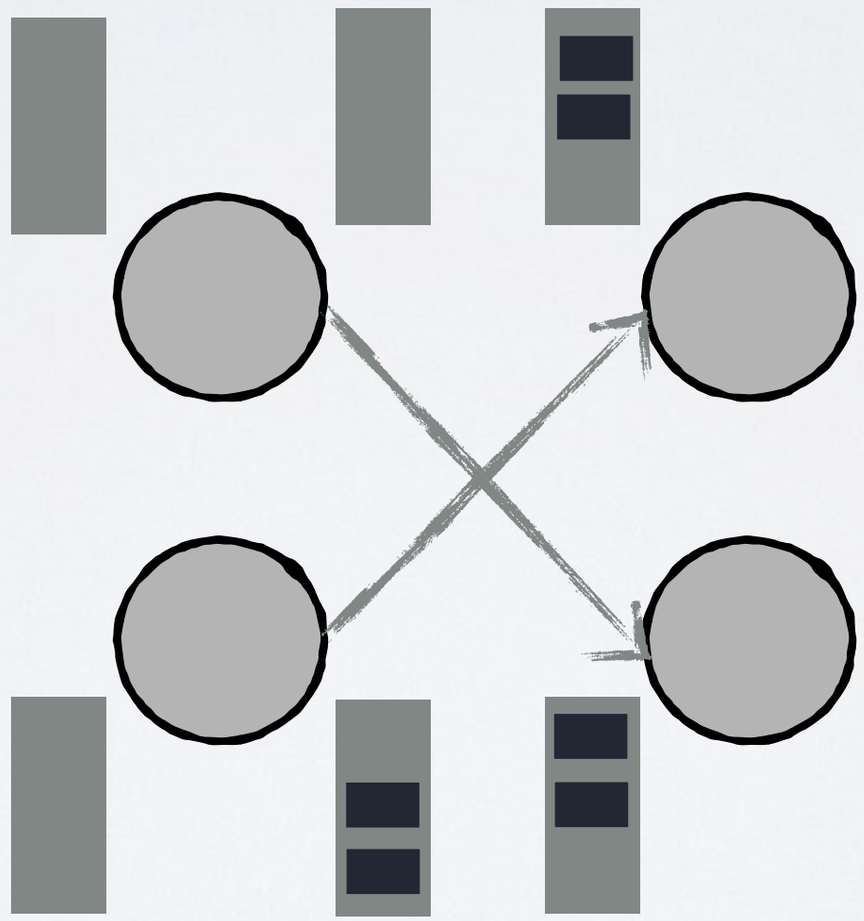
BATCH ENGINE



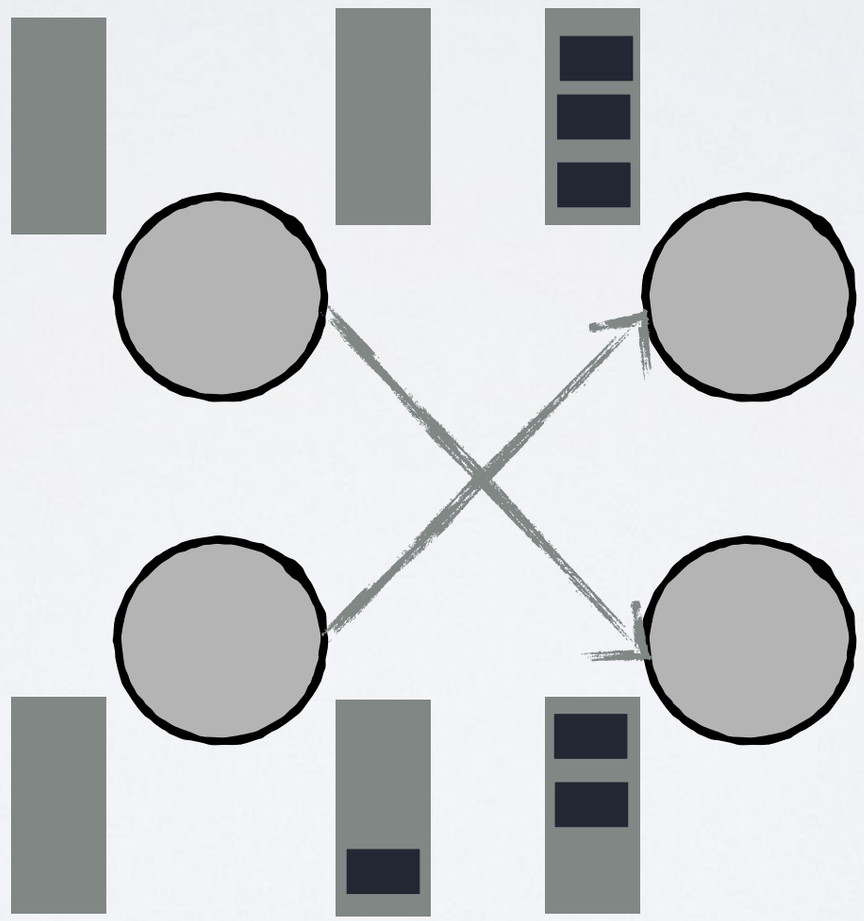
BATCH ENGINE



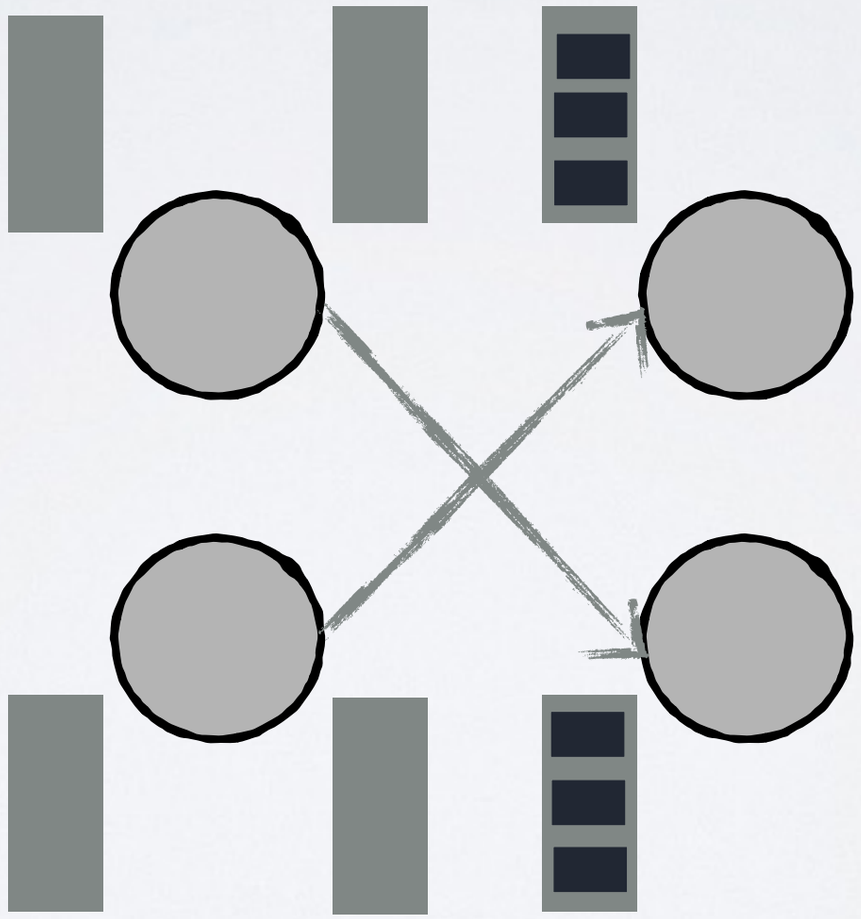
BATCH ENGINE



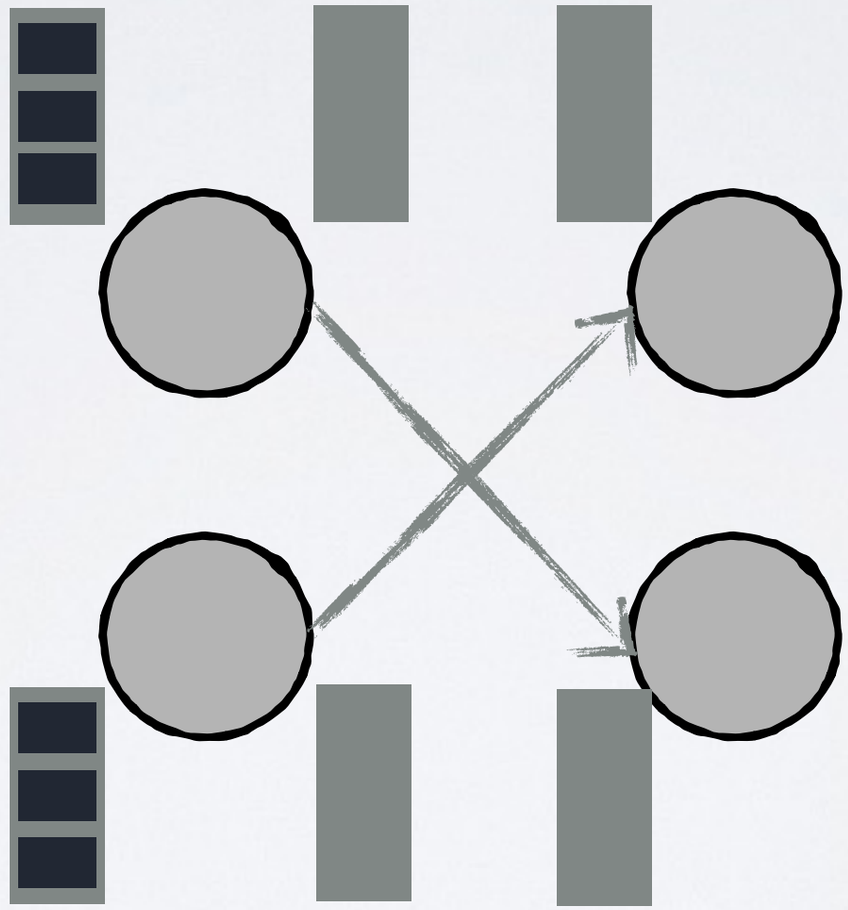
BATCH ENGINE



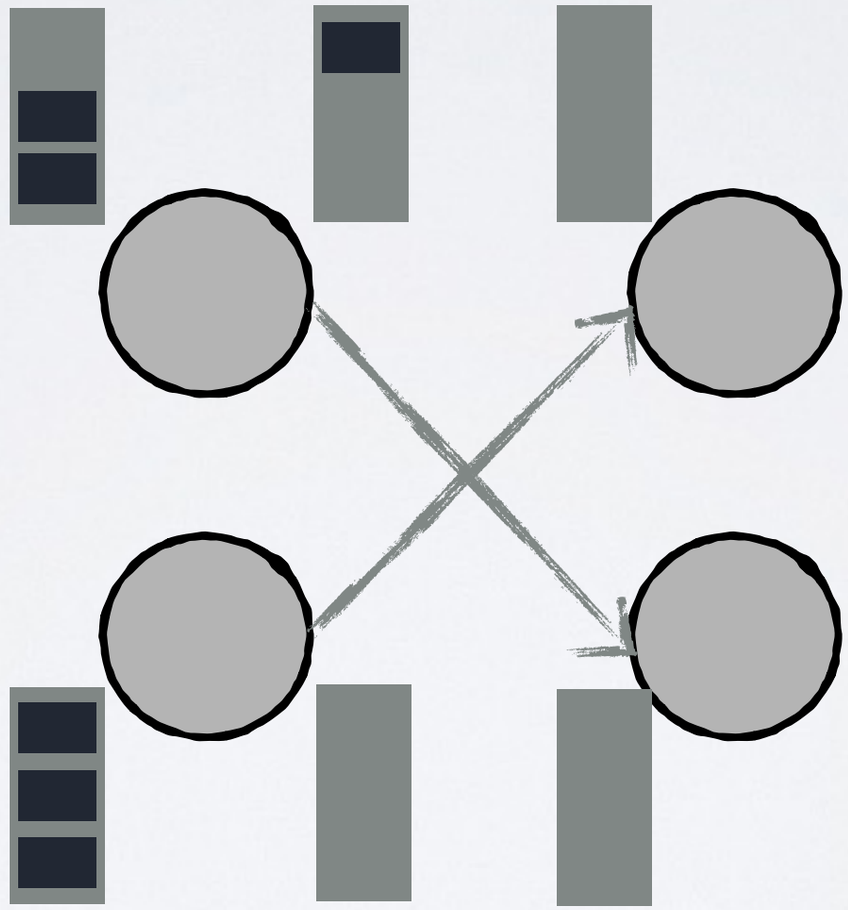
BATCH ENGINE



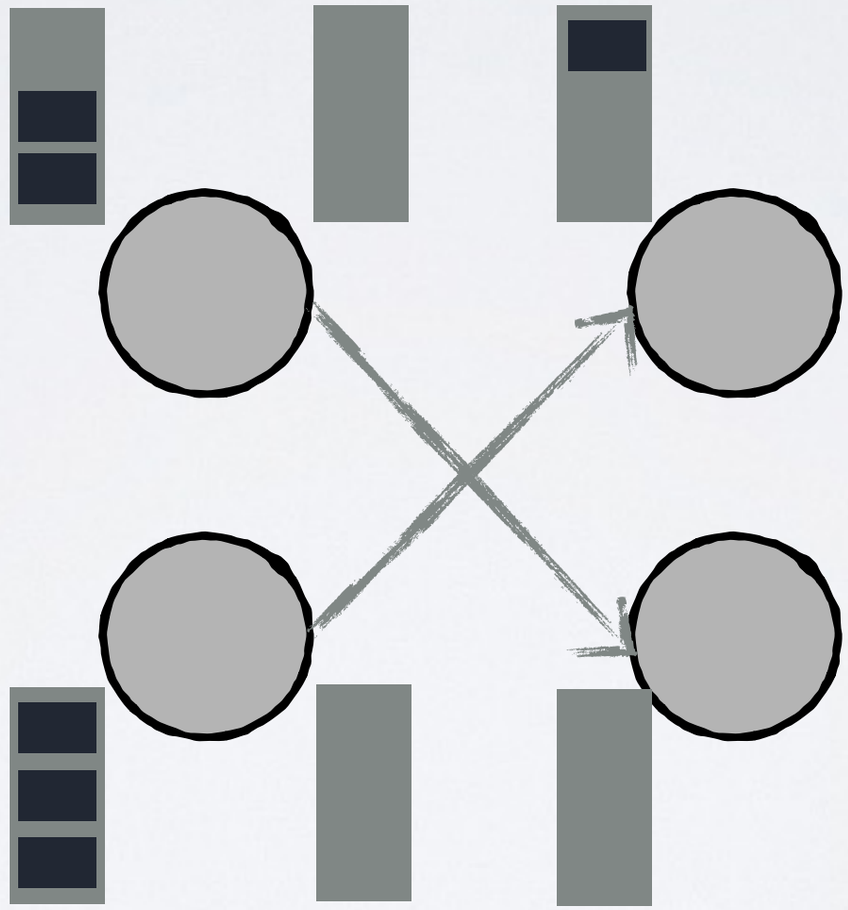
STREAM ENGINE



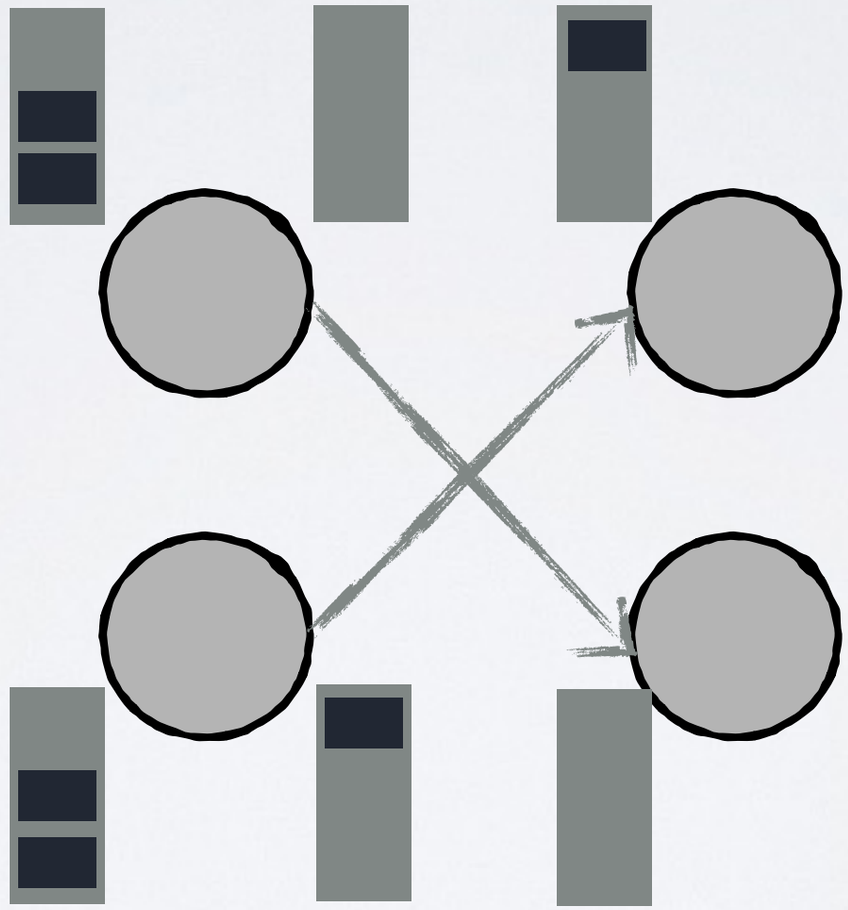
STREAM ENGINE



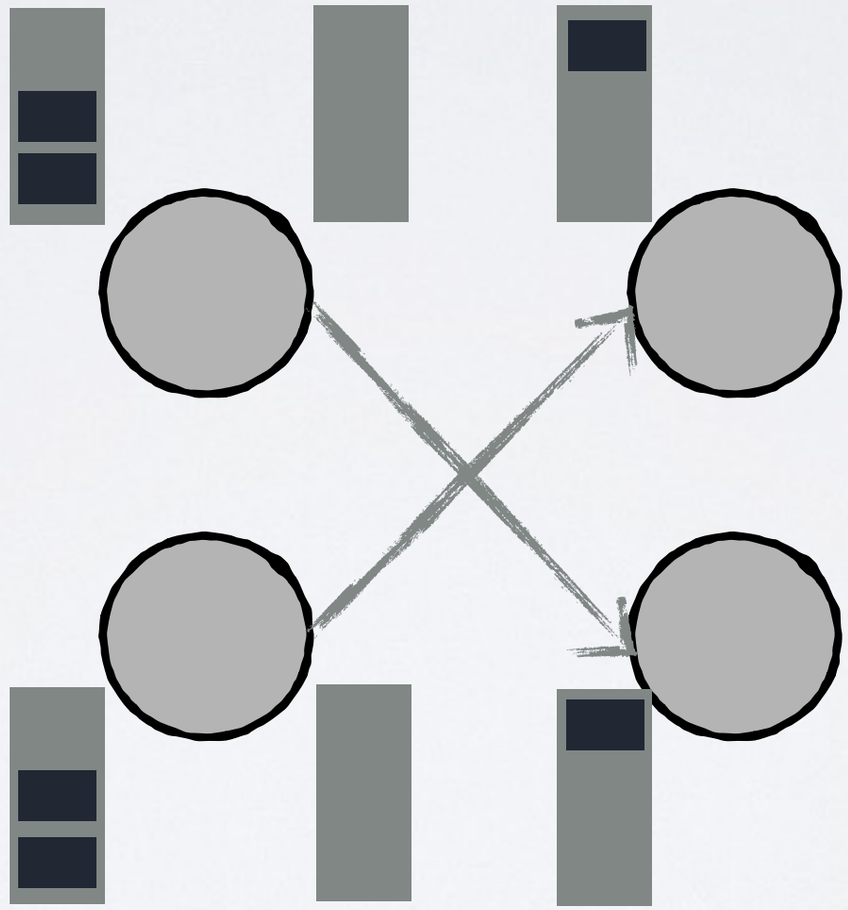
STREAM ENGINE



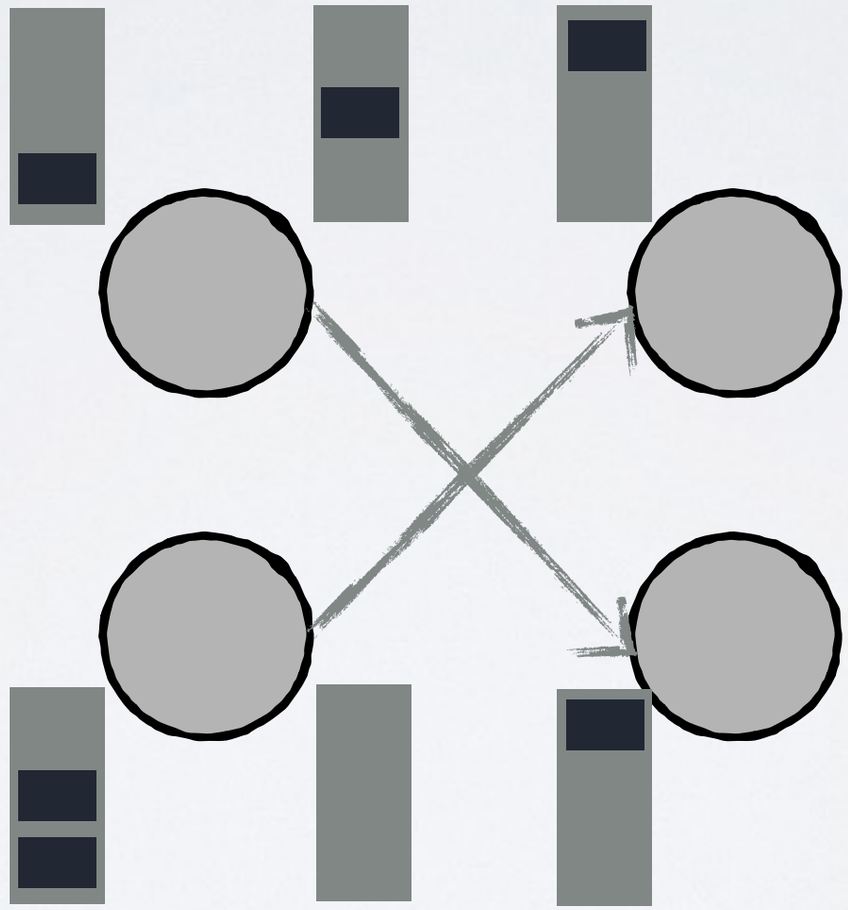
STREAM ENGINE



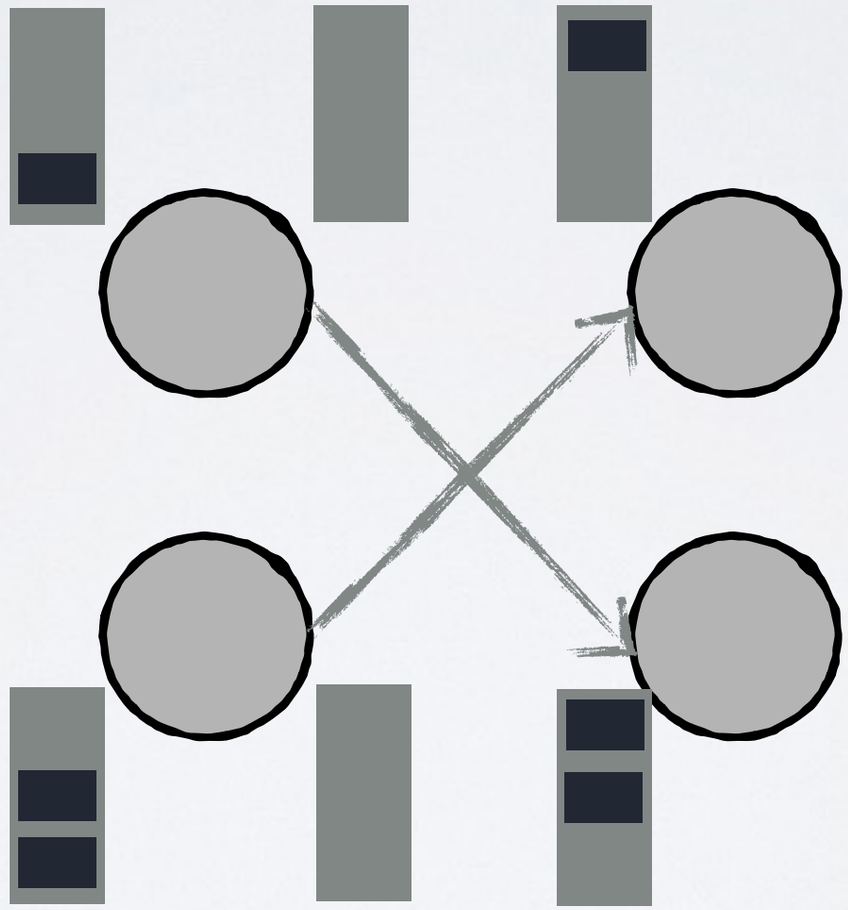
STREAM ENGINE



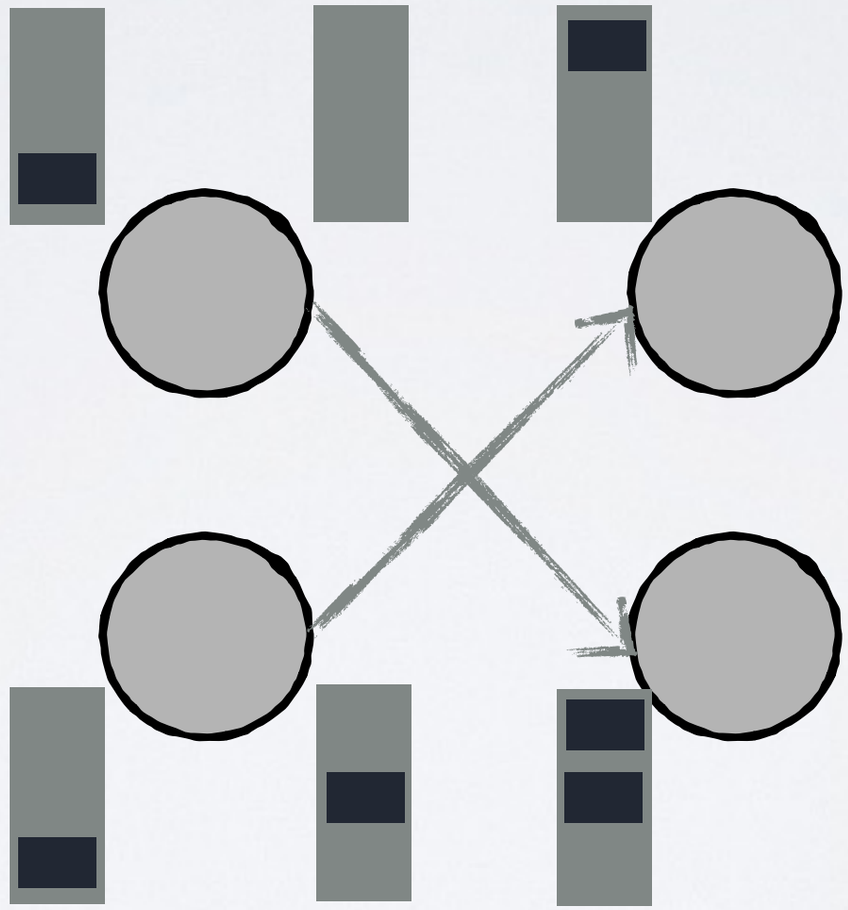
STREAM ENGINE



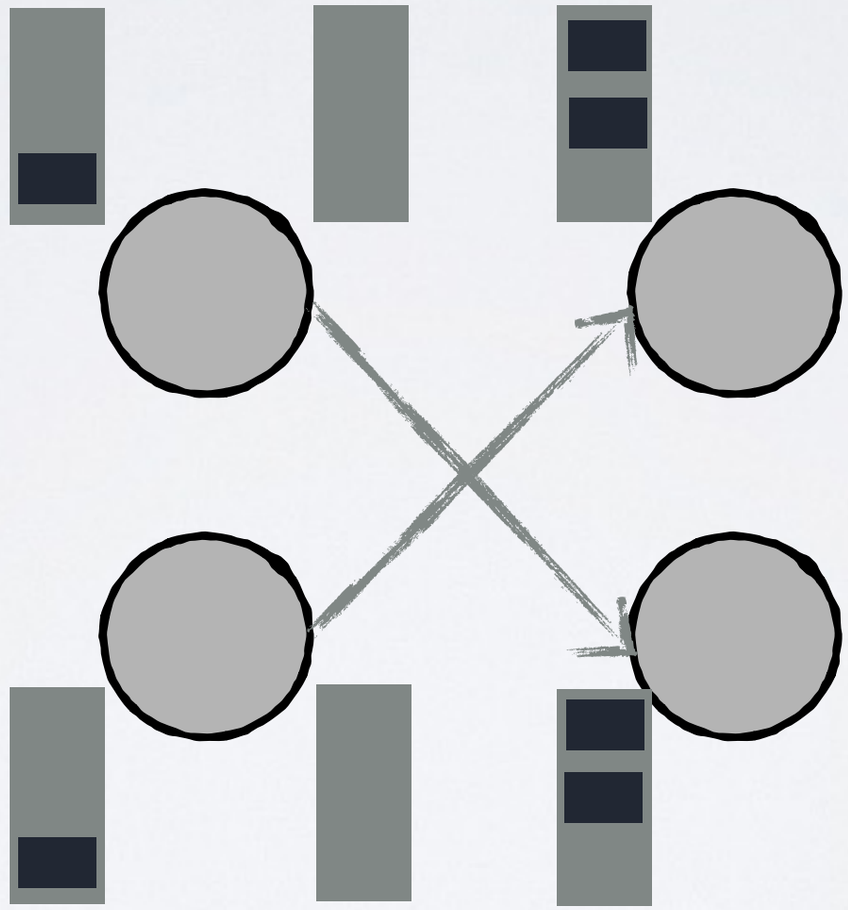
STREAM ENGINE



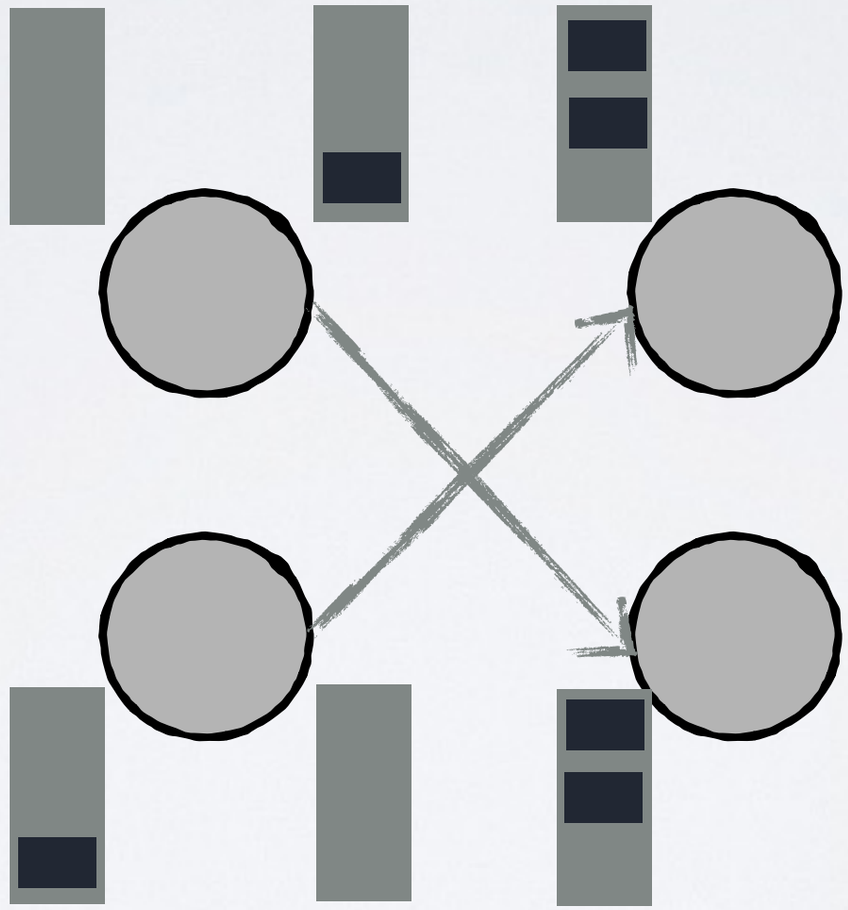
STREAM ENGINE



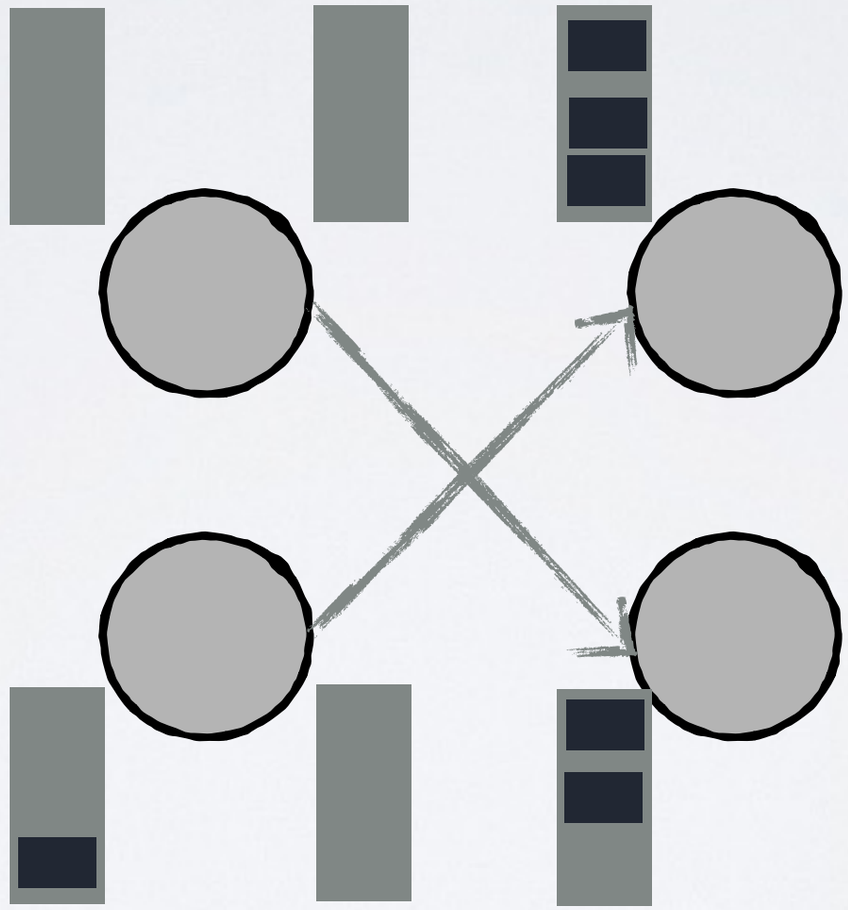
STREAM ENGINE



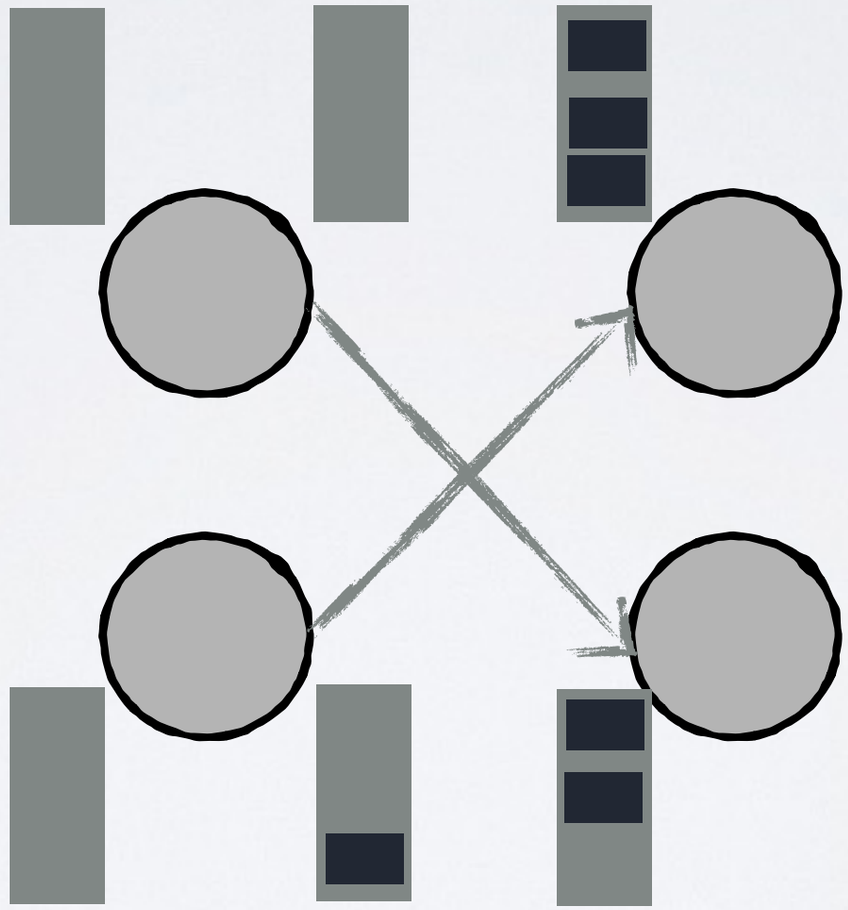
STREAM ENGINE



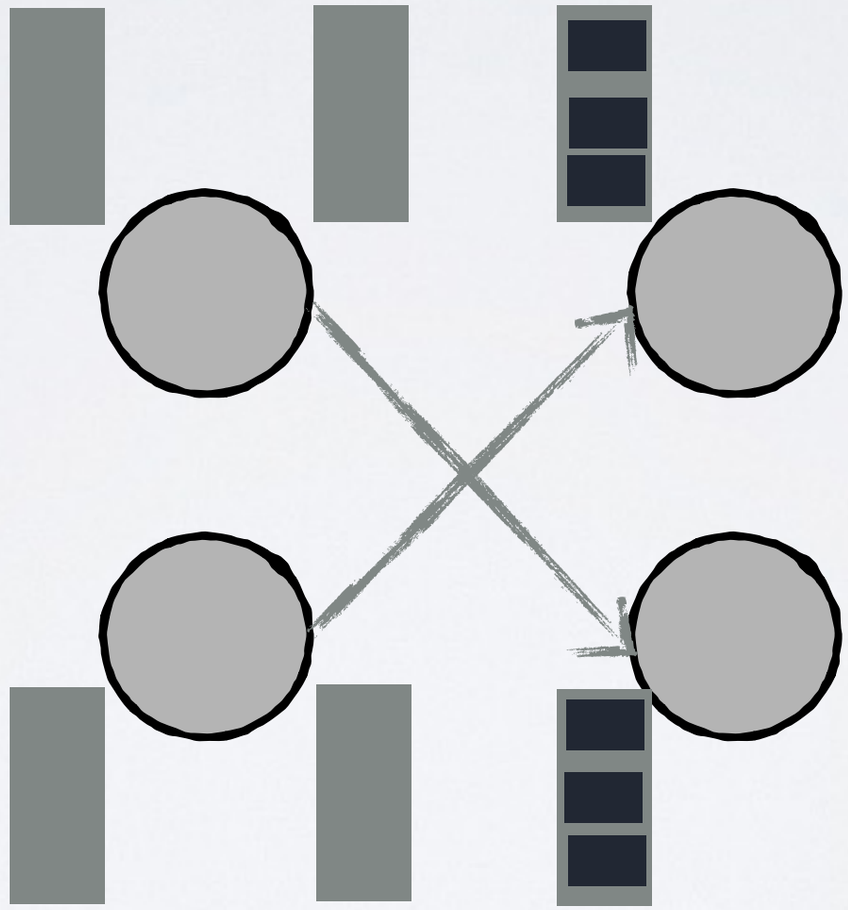
STREAM ENGINE



STREAM ENGINE



STREAM ENGINE



BATCH ENGINE FOR BOUNDED DATA AND STREAM ENGINE FOR UNBOUNDED DATA, WHY?

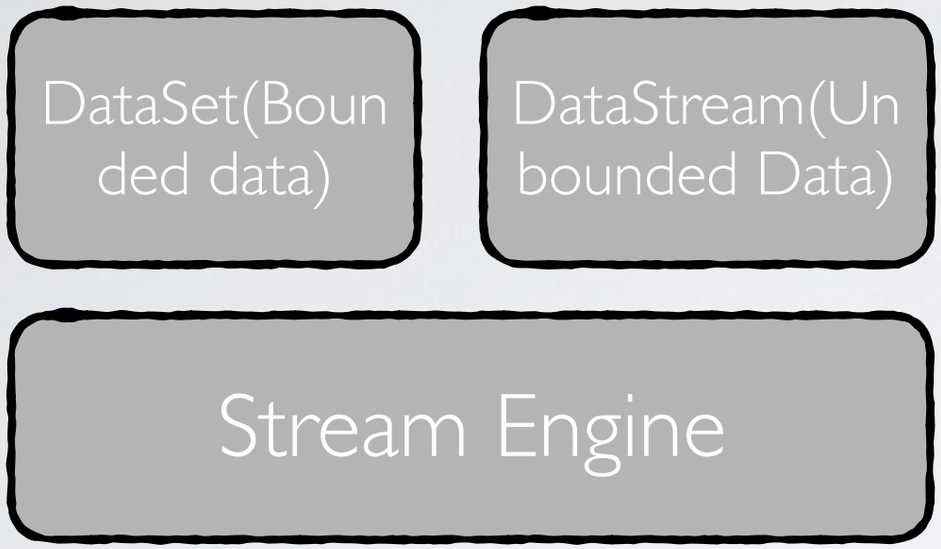
BATCH ENGINE FOR UNBOUNDED DATA

Divide unbounded data into bounded data by window.

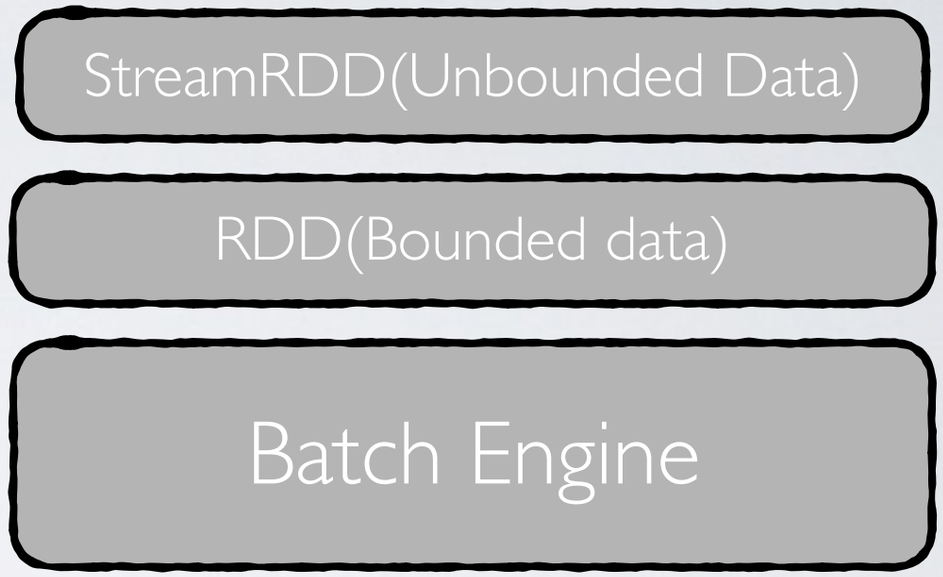
STREAM ENGINE FOR BOUNDED DATA

Better resource utility with the cost of task level fault tolerant.

Flink



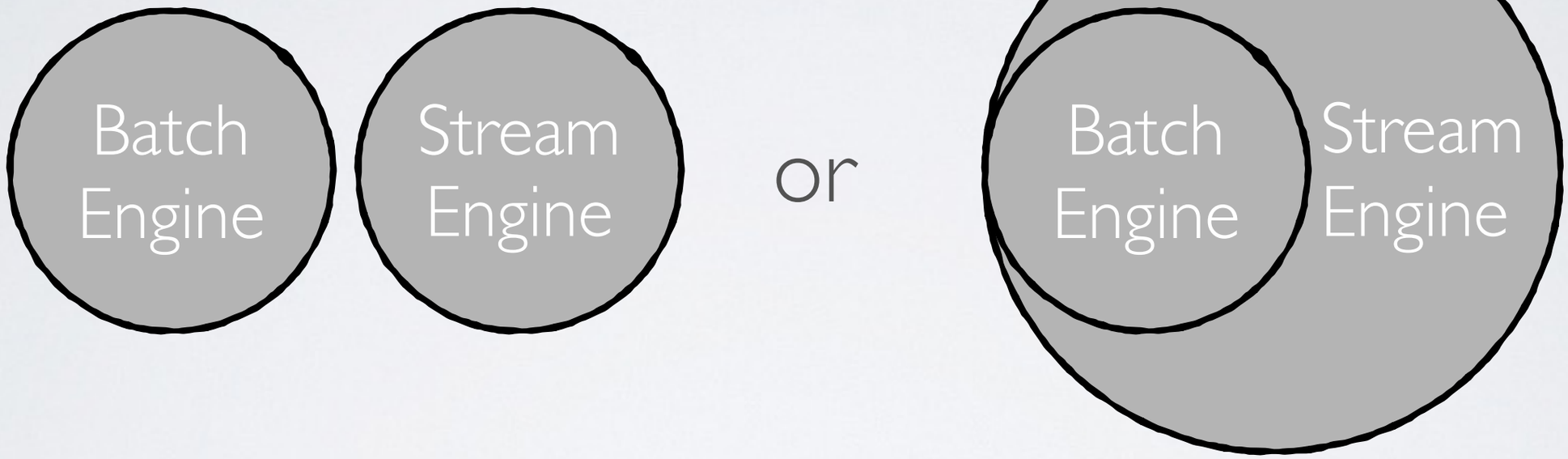
Spark



Which one is better?

FLINK STREAM ENGINE

- Flink Runtime Engine support both Batch and Pipeline mode.
- For DataSet API, use Batch or Pipeline.
- For DataStream API, use Pipeline.


Batch
Engine
Stream
Engine
or
Batch
Engine
Stream
Engine

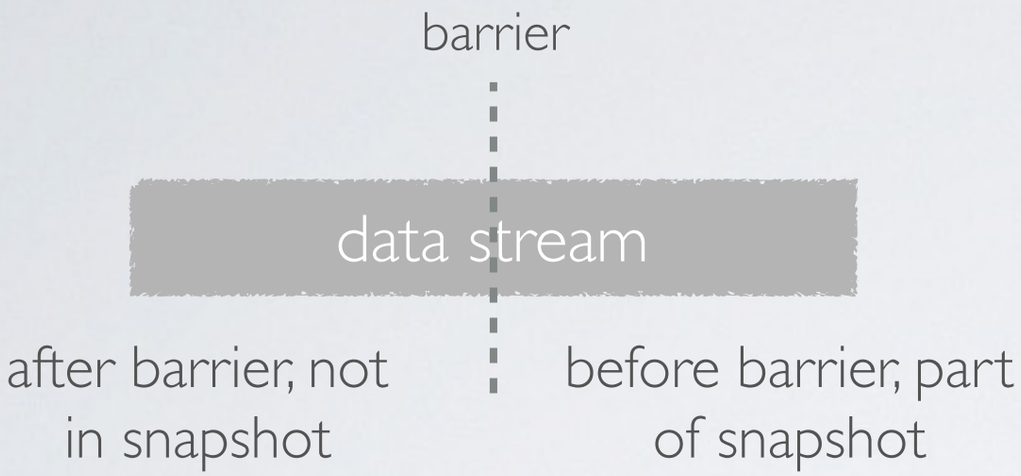
WHAT SHOULD YOU CARE ABOUT STREAM?

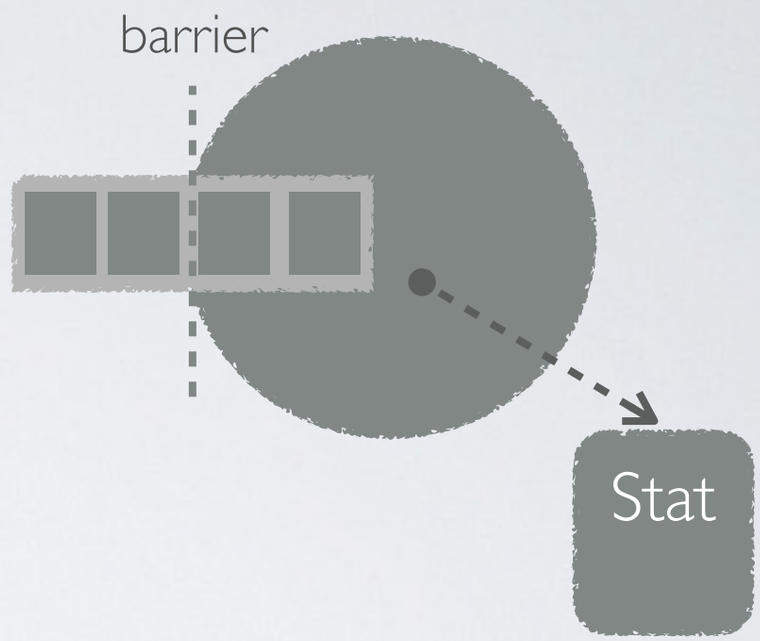
CORRECTNESS IS THE STAPLE FOOD INSTEAD OF DESSERT

Batch Engine intermediate state = all data + partial computation

Stream Engine intermediate state = partial data + all computation

- Distributed Snapshot





- Exactly-Once support.

1. do distributed snapshot.
2. rewind messages after barrier from source.
3. support rewinding messages at source side.
4. support UPDATE at sink side.

WHEN IT'S HAPPENED INSTEAD OF WHEN IT'S PROCESSED

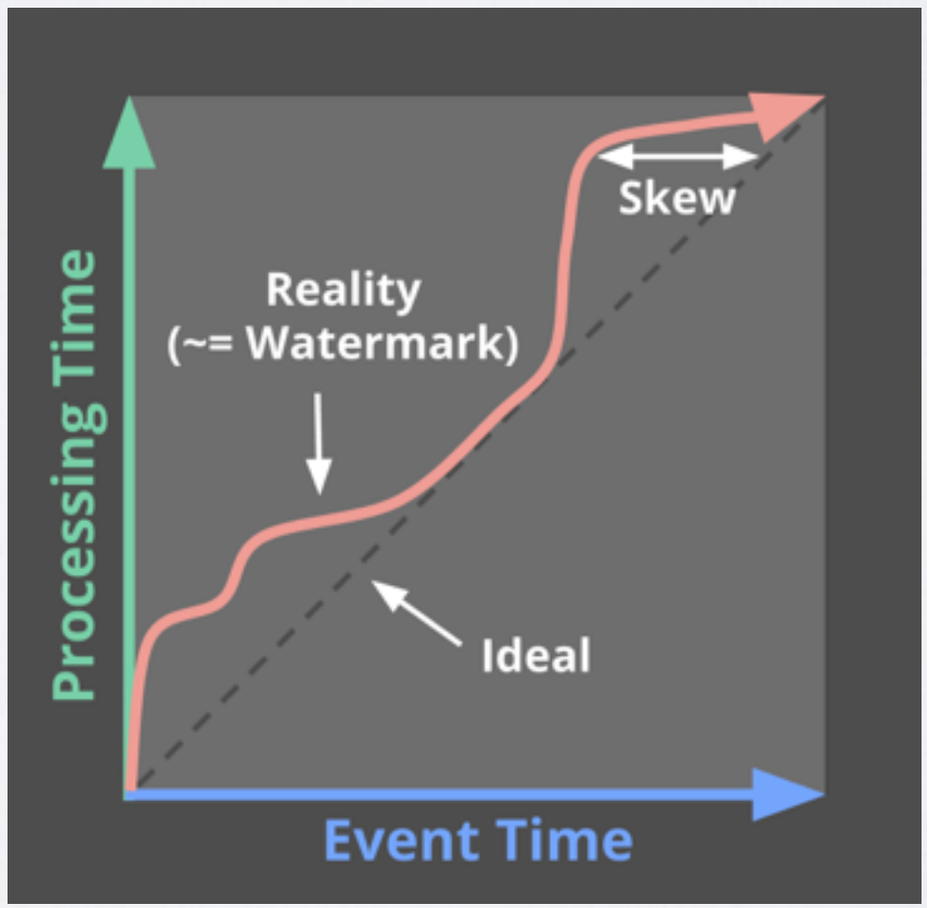
- Process Time

The time at which events are observed in the system.

- Event Time

The time at which events actually occurred.

- Watermark



- Window
  - Process Time/Event Time/Session
  - Tumbling/Sliding
- Trigger
  - Process Time/Event Time/Count
  - Continuous/Purging/Delta
- Evictor
  - Time/Count/Delta

CEP(COMPLEX EVENT PROCESSING)

```
Pattern<MonitoringEvent, ?> warningPattern =
Pattern.<MonitoringEvent>begin("First Event")
    .subtype(TemperatureEvent.class)
    .where(evt -> evt.getTemperature() >= TEMPERATURE_THRESHOLD)
    .next("Second Event")
    .subtype(TemperatureEvent.class)
    .where(evt -> evt.getTemperature() >= TEMPERATURE_THRESHOLD)
    .within(Time.seconds(10));
```

FLINK STREM SQL

```
val env = StreamExecutionEnvironment.getExecutionEnvironment()

// create some stream in the Streaming API

val stream : DataStream[(String, Double, Int)] = env

    .addSource(new FlinkKafkaConsumer(...))

    .map { line => parse(line) }

// create Table Environment and register stream with column names

val tabEnv = new TableEnvironment(env)

tabEnv.registerDataStream(stream, "myStreamTab", ("ID", "MEASURE", "COUNT"))

val rTable: Table = tabEnv.sql(

    "SELECT ID, MEASURE FROM myStreamTab WHERE COUNT > 17")
```

THANKS

REFERENCE

1. https://www.oreilly.com/ideas/the-world-beyond-batch-streaming-101

2. http://radar.oreilly.com/2014/07/why-local-state-is-a-fundamental-primitive-in-stream-processing.html

3. http://flink.apache.org/

4. http://spark.apache.org/