BUILDING REALTIME DATA PIPELINES WITH KAFKA CONNECT AND SPARK STREAMING

Guozhang Wang

Confluent

SPARK SUMMIT 2016
DATA SCIENCE AND ENGINEERING AT SCALE
JUNE 6-8, 2016 SAN FRANCISCO

About Me: Guozhang Wang

- Engineer @ Confluent.
- Apache Kafka Committer, PMC Member.
- Before: Engineer @ LinkedIn, Kafka and Samza.

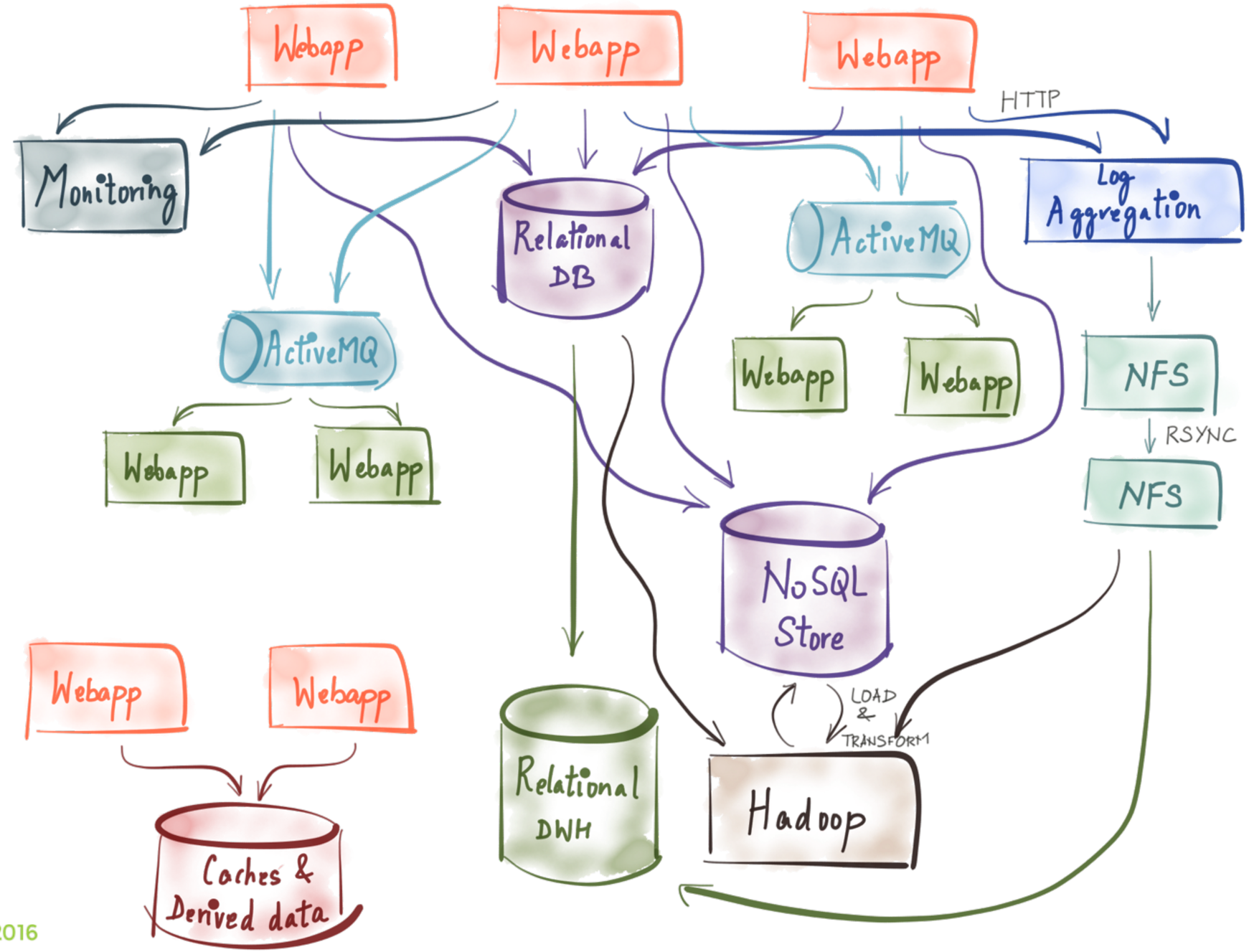
What do you REALLY need for Stream Processing?

Spark Streaming!

Spark Streaming! Is that All?

Spark Streaming! Is that All?

Data can Comes from / Goes to..


Webapp
Webapp
Webapp
HTTP
Monitoring
Relational DB
ActiveMQ
Log Aggregation
ActiveMQ
Webapp
Webapp
NFS
Webapp
Webapp
RSYNC
NFS
NoSQL Store
LOAD & TRANSFORM
Webapp
Webapp
Relational DWH
Hadoop
Caches & Derived data

Real-time Data Integration:

getting data to all the right places

Option #1: One-off Tools

- Tools for each specific data systems
- Examples:
  - jdbcRDD, Cassandra-Spark connector, etc..
  - Sqoop, logstash to Kafka, etc..

Option #2: Kitchen Sink Tools

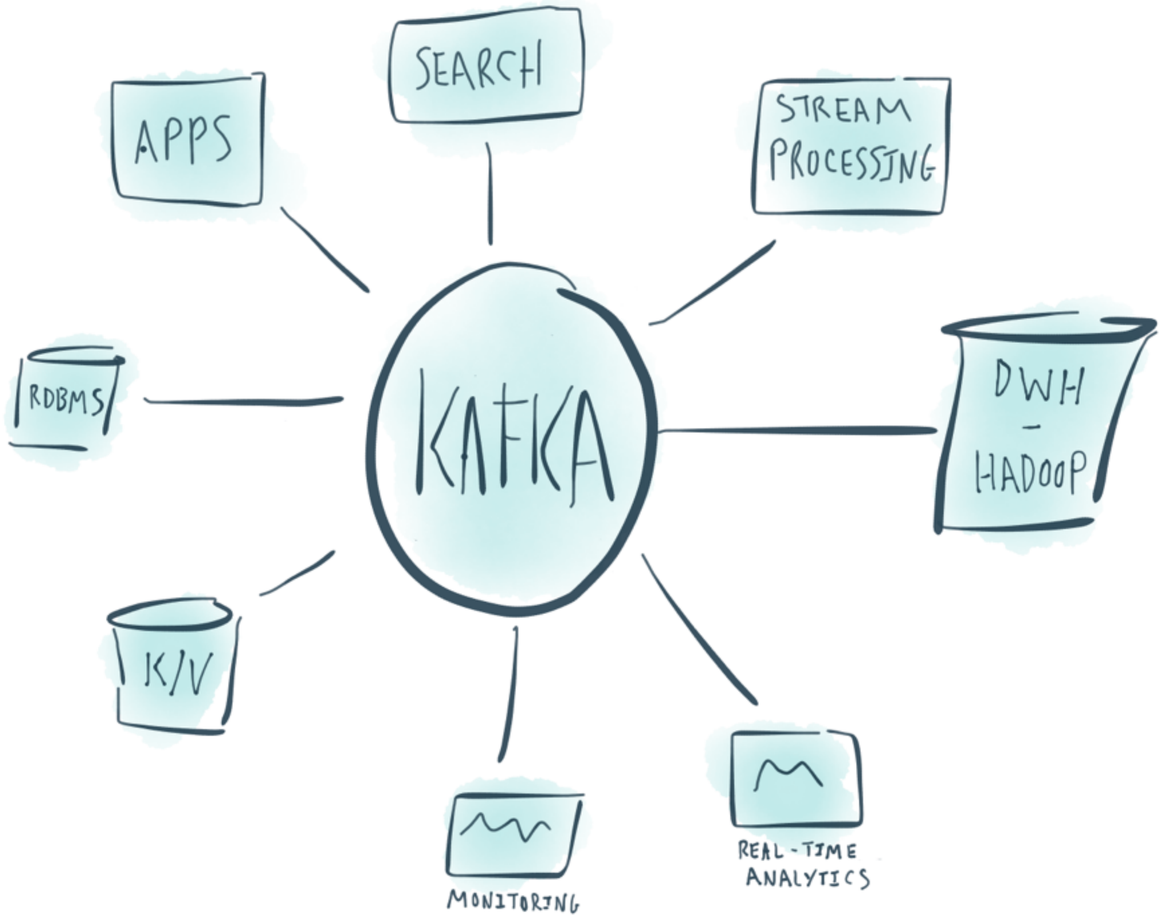
- Generic point-to-point data copy / ETL tools
- Examples:
  - Enterprise application integration tools

Option #3: Streaming as Copying

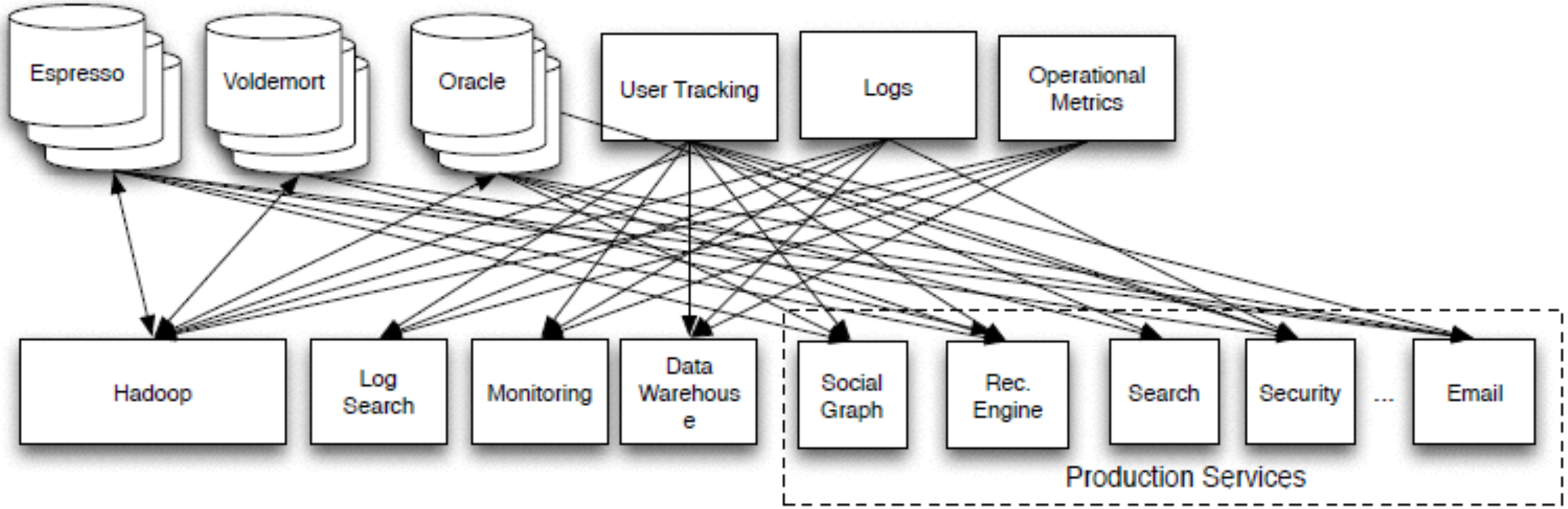
- Use stream processing frameworks to copy data
- Examples:
  - Spark Streaming: MyRDDWriter (forEachPartition)
  - Storm, Samza, Flink, etc..

Real-time Integration: E, T & L

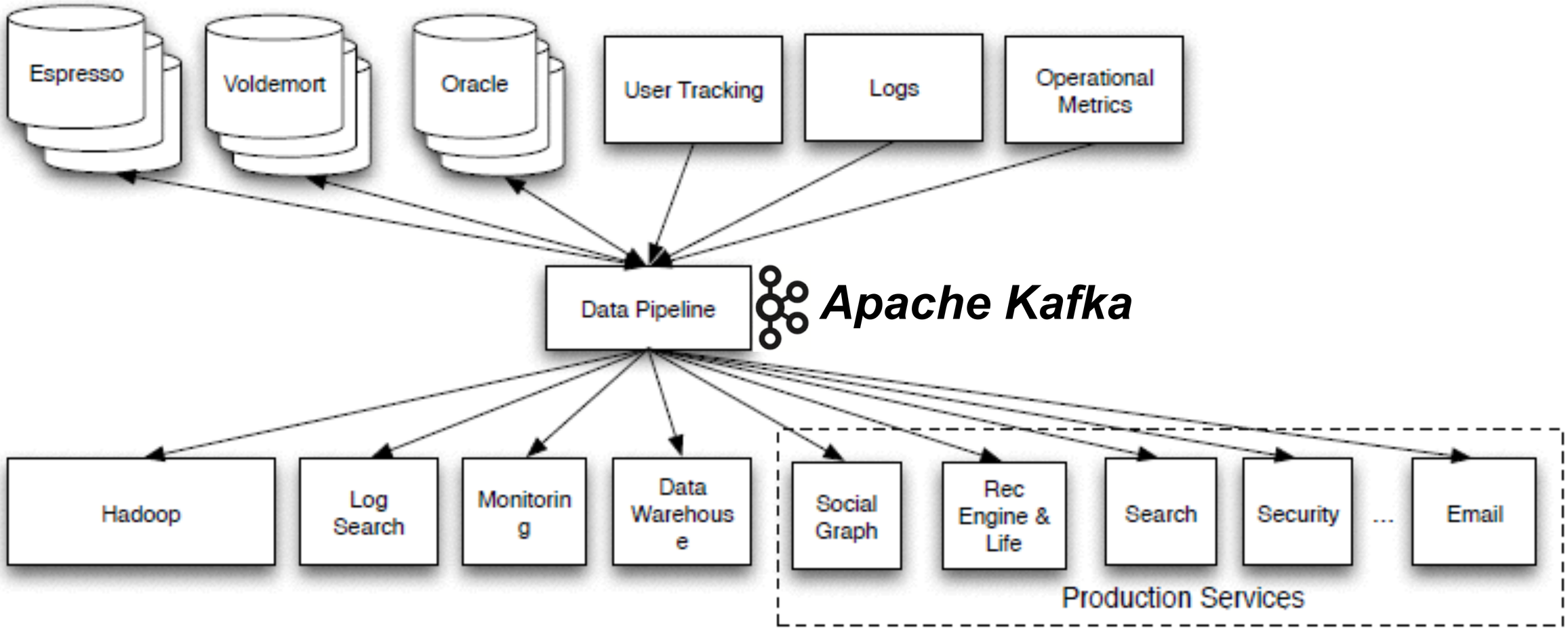
STREAMING PLATFORM



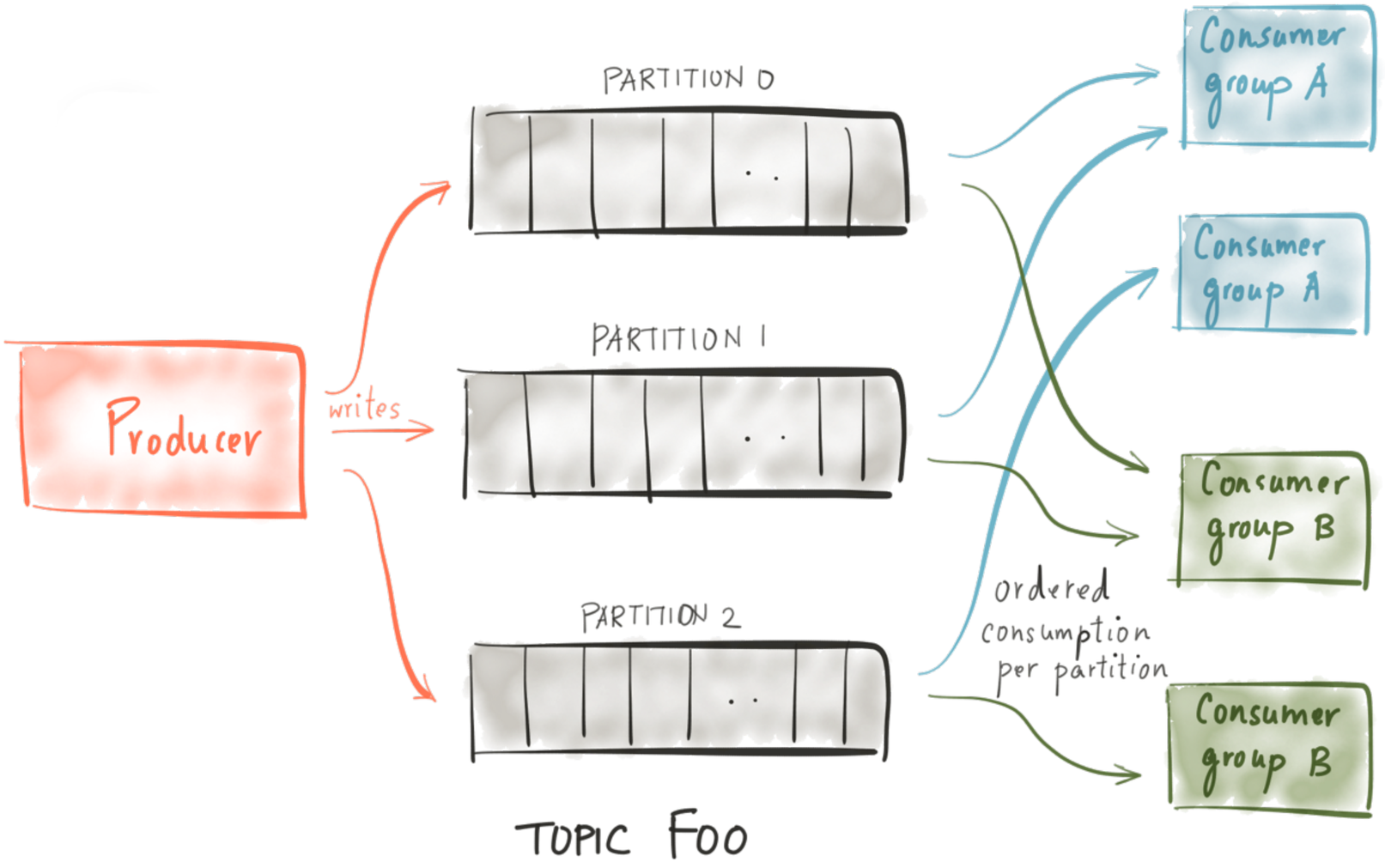
Example: LinkedIn back in 2010



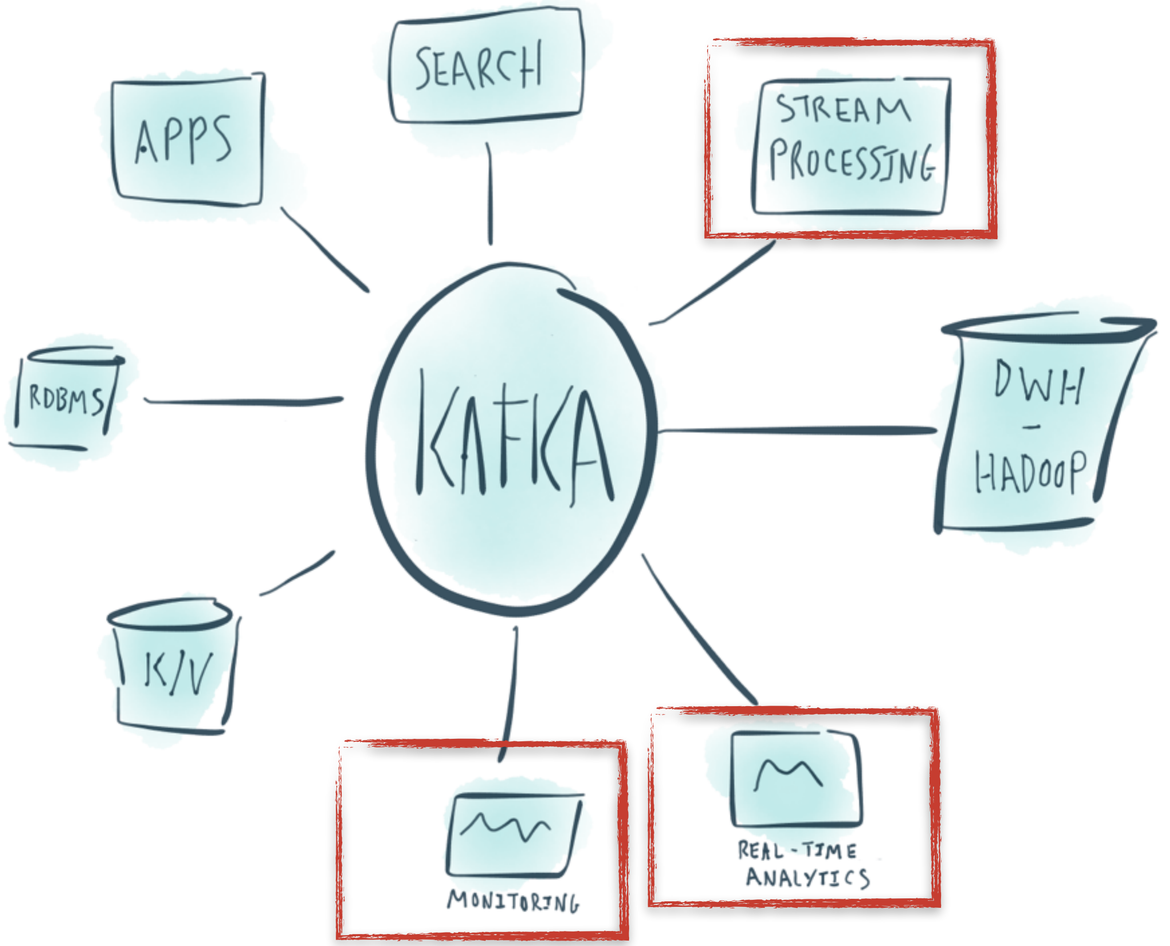
Example: LinkedIn with Kafka

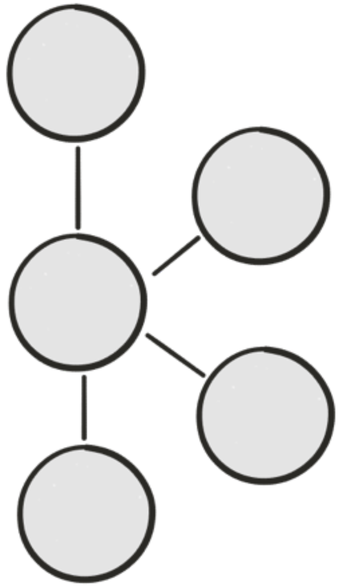


Scalable Consumption



STREAMING PLATFORM





Kafka Connect

Large-scale streaming data import/export for Kafka

GOALS

1. Focus on copying
2. Batteries included
3. Standardize
4. Parallelism
5. Scale

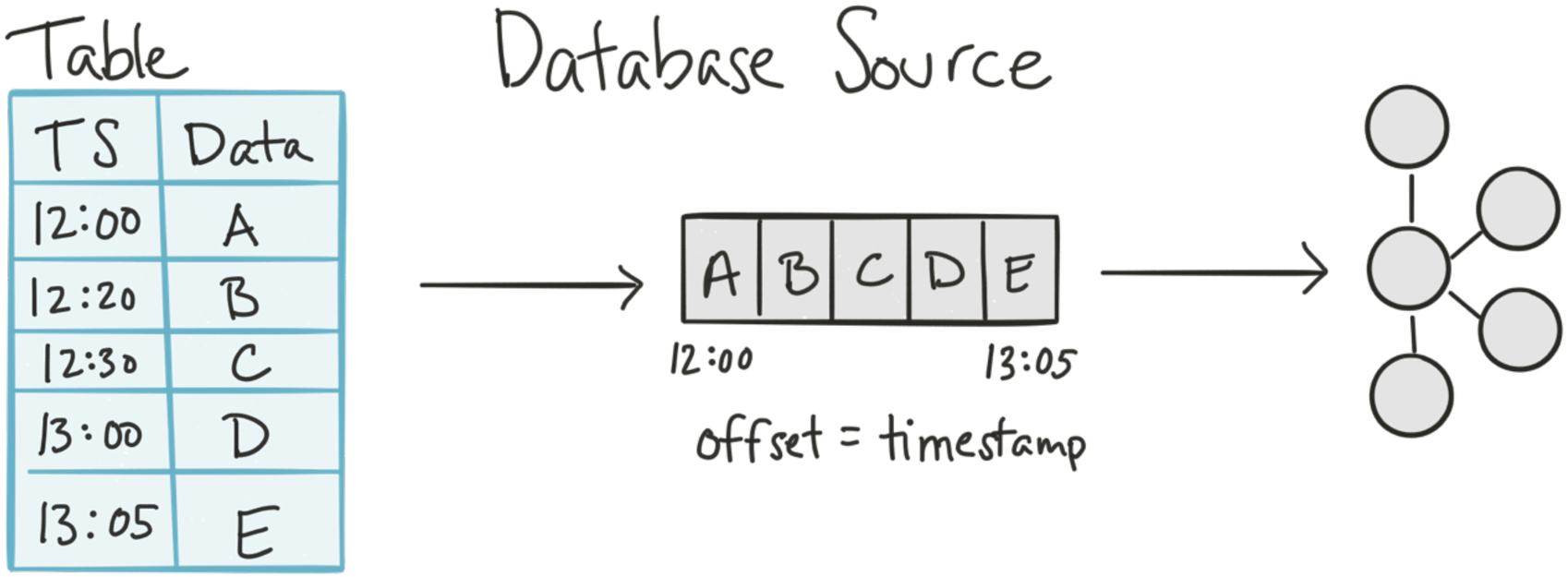
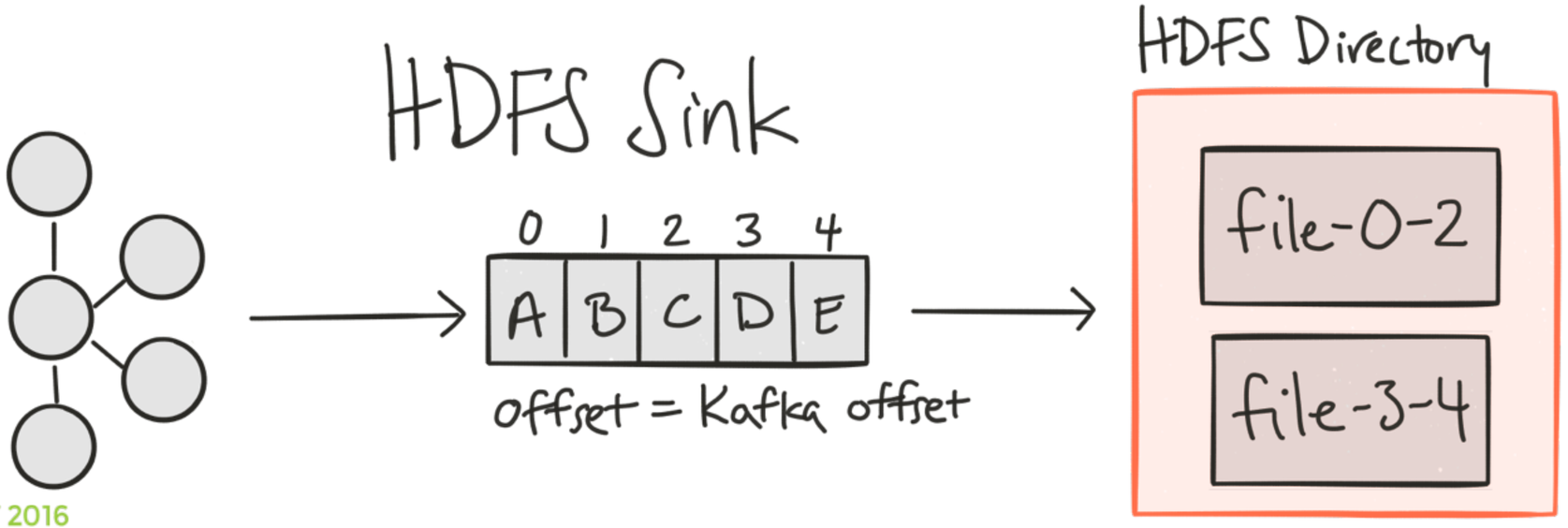
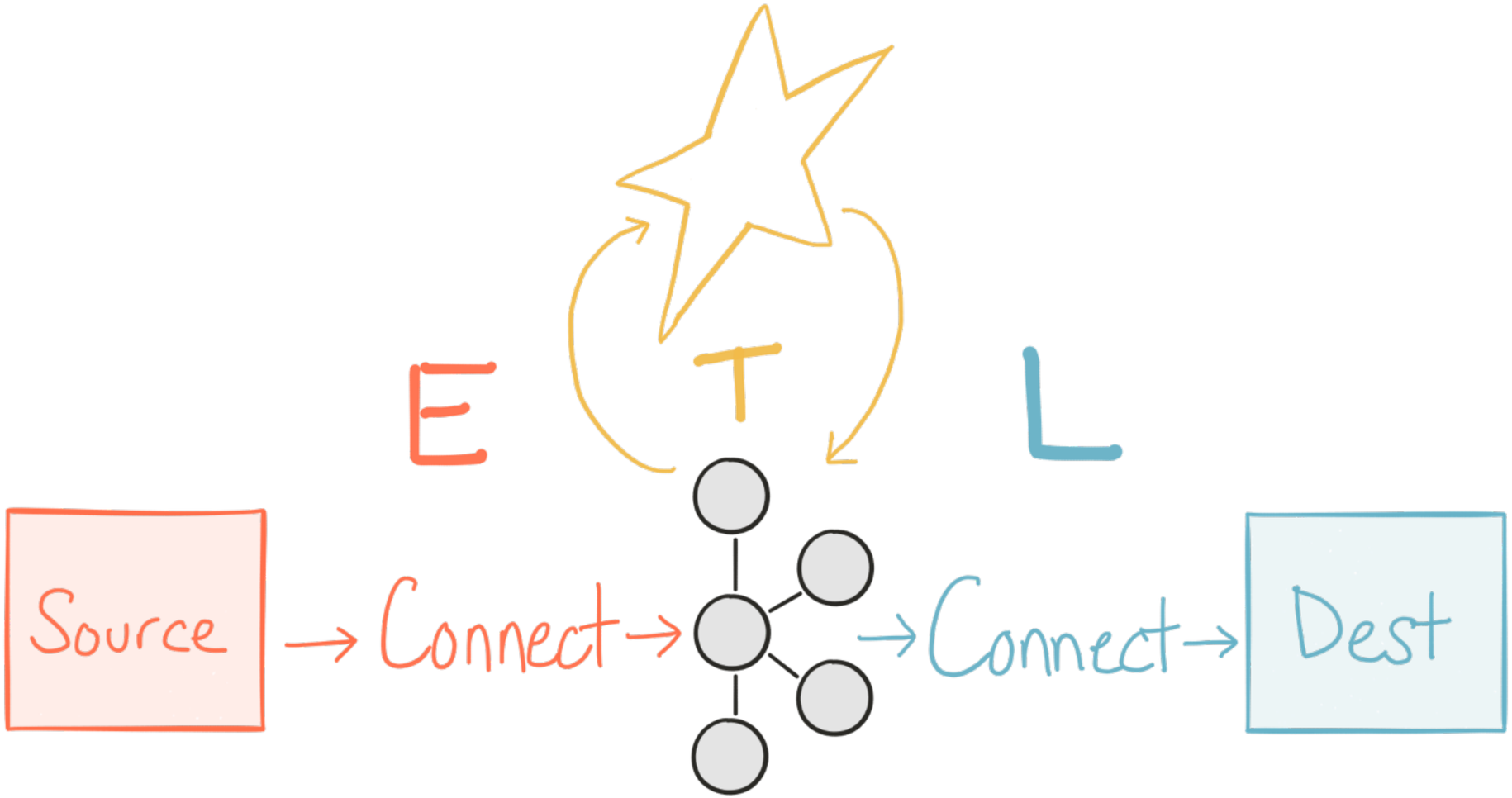


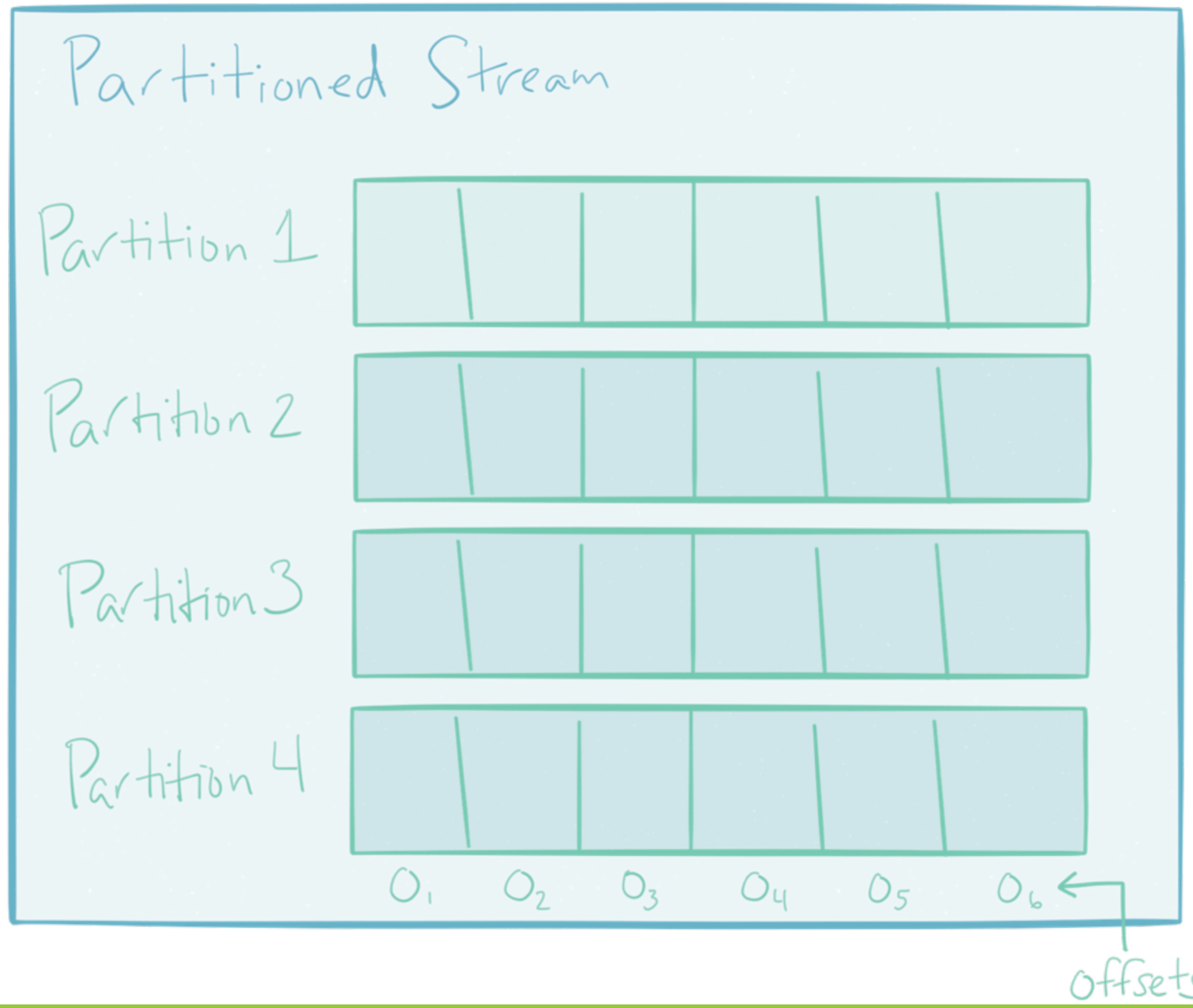
Table
TS
Data
12:00
A
12:20
B
12:30
C
13:00
D
13:05
E
Database Source
A
B
C
D
E
12:00
13:05
offset = timestamp


HDFS Sink
0
1
2
3
4
A
B
C
D
E
offset = Kafka offset
HDFS Directory
file-0-2
file-3-4

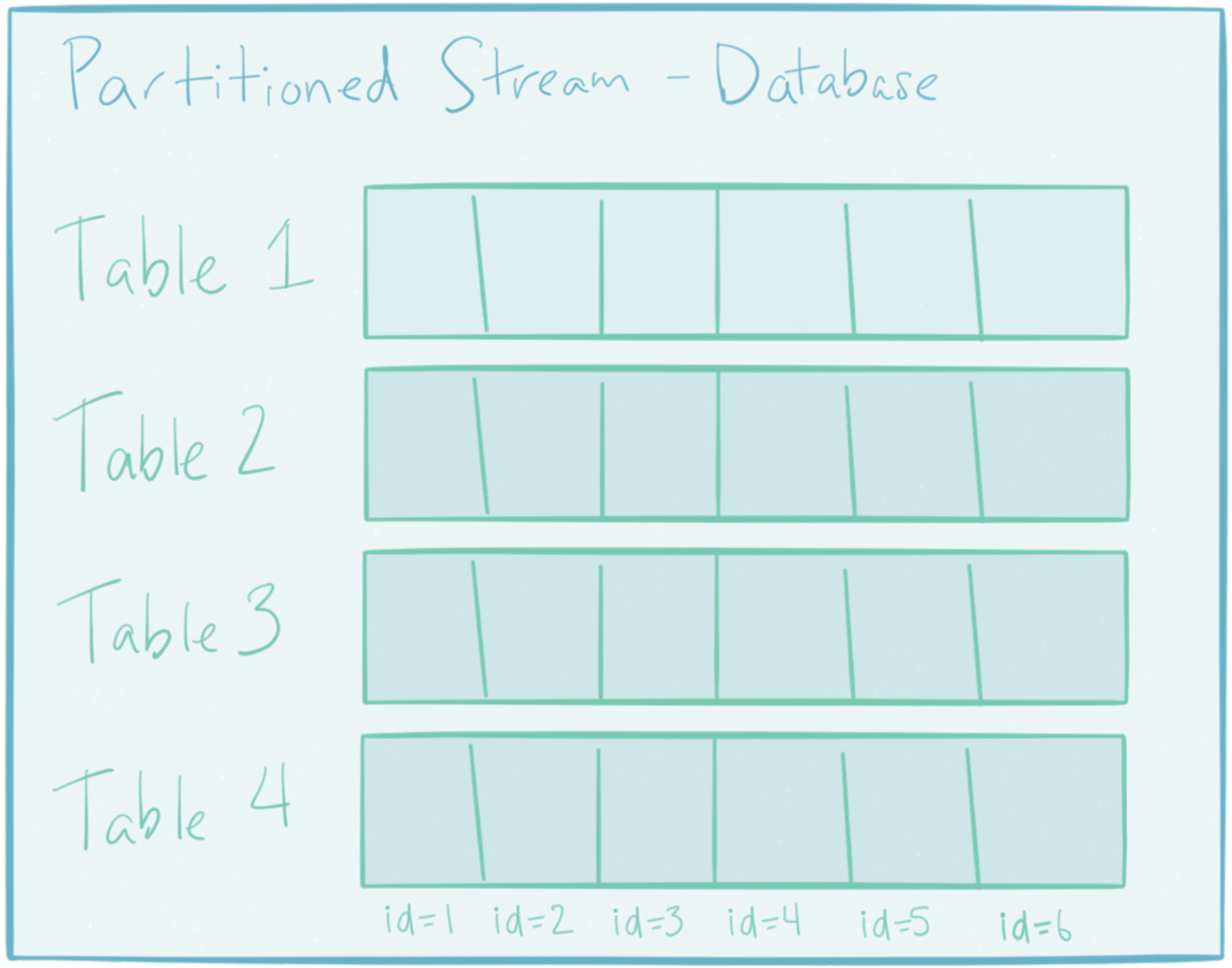
Separation of Concerns



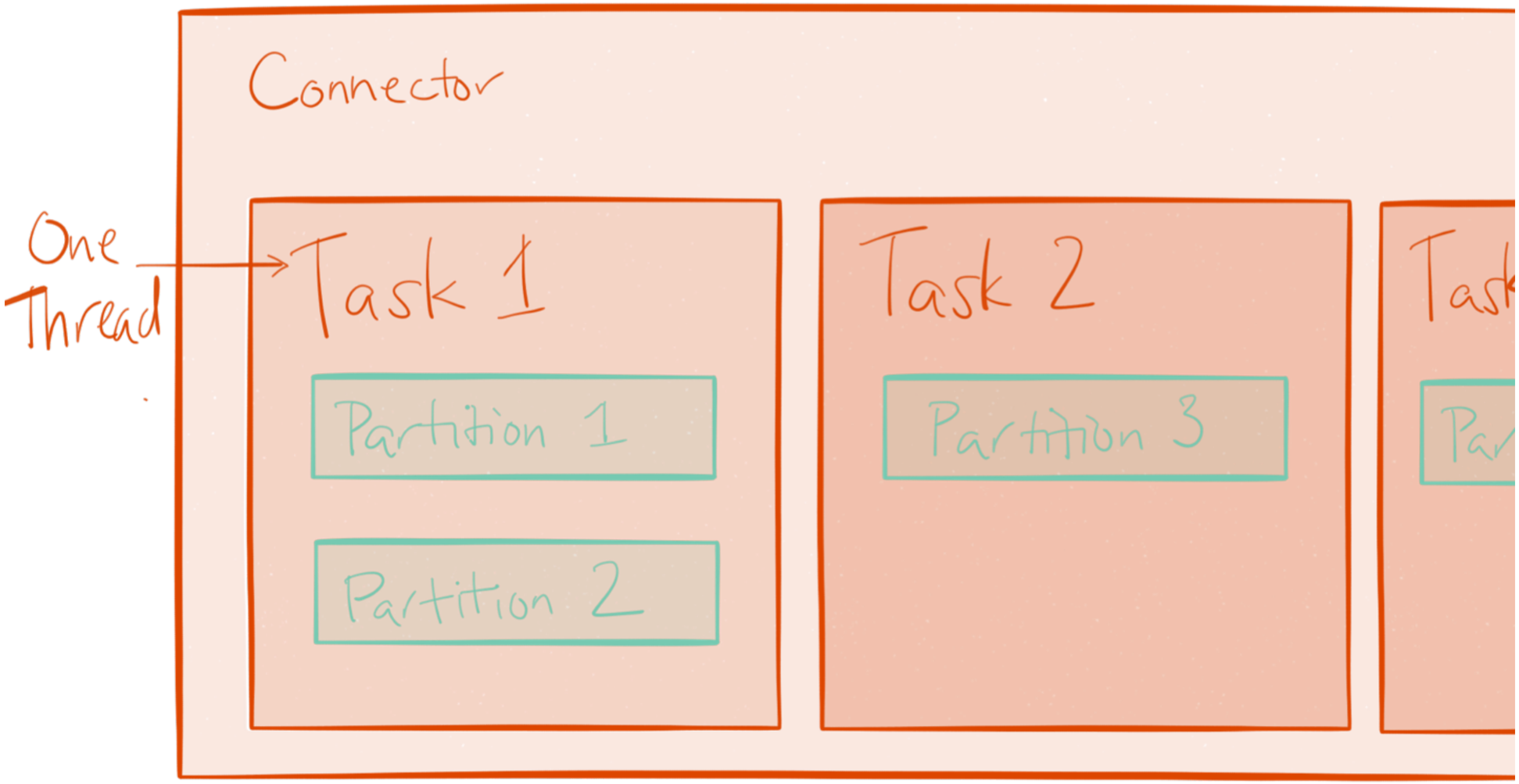
Data Model


Partitioned Stream
Partition 1
Partition 2
Partition 3
Partition 4
O_1 O_2 O_3 O_4 O_5 O_6
offsets

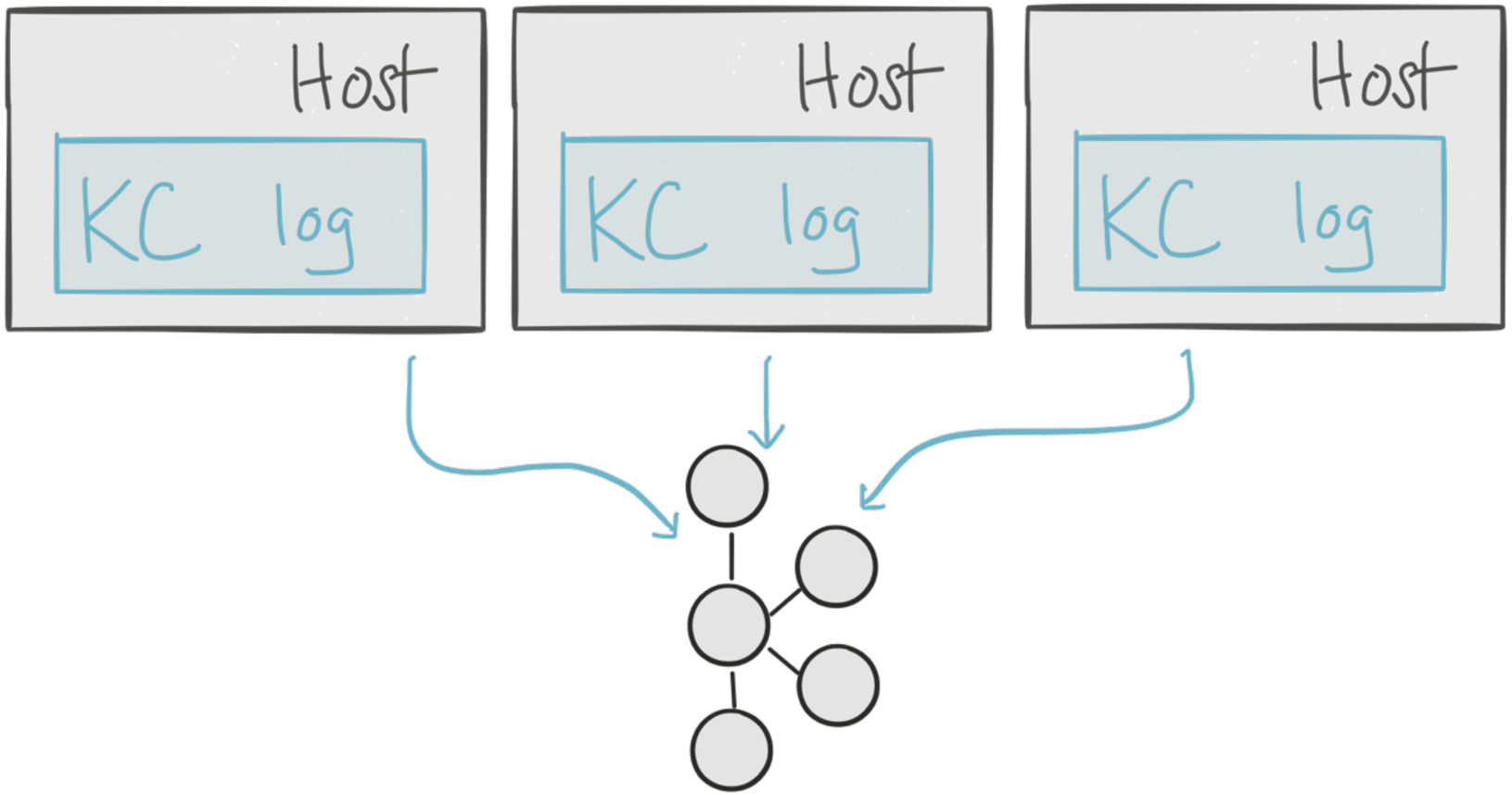
Data Model


Partitioned Stream - Database
Table 1
Table 2
Table 3
Table 4
id=1
id=2
id=3
id=4
id=5
id=6

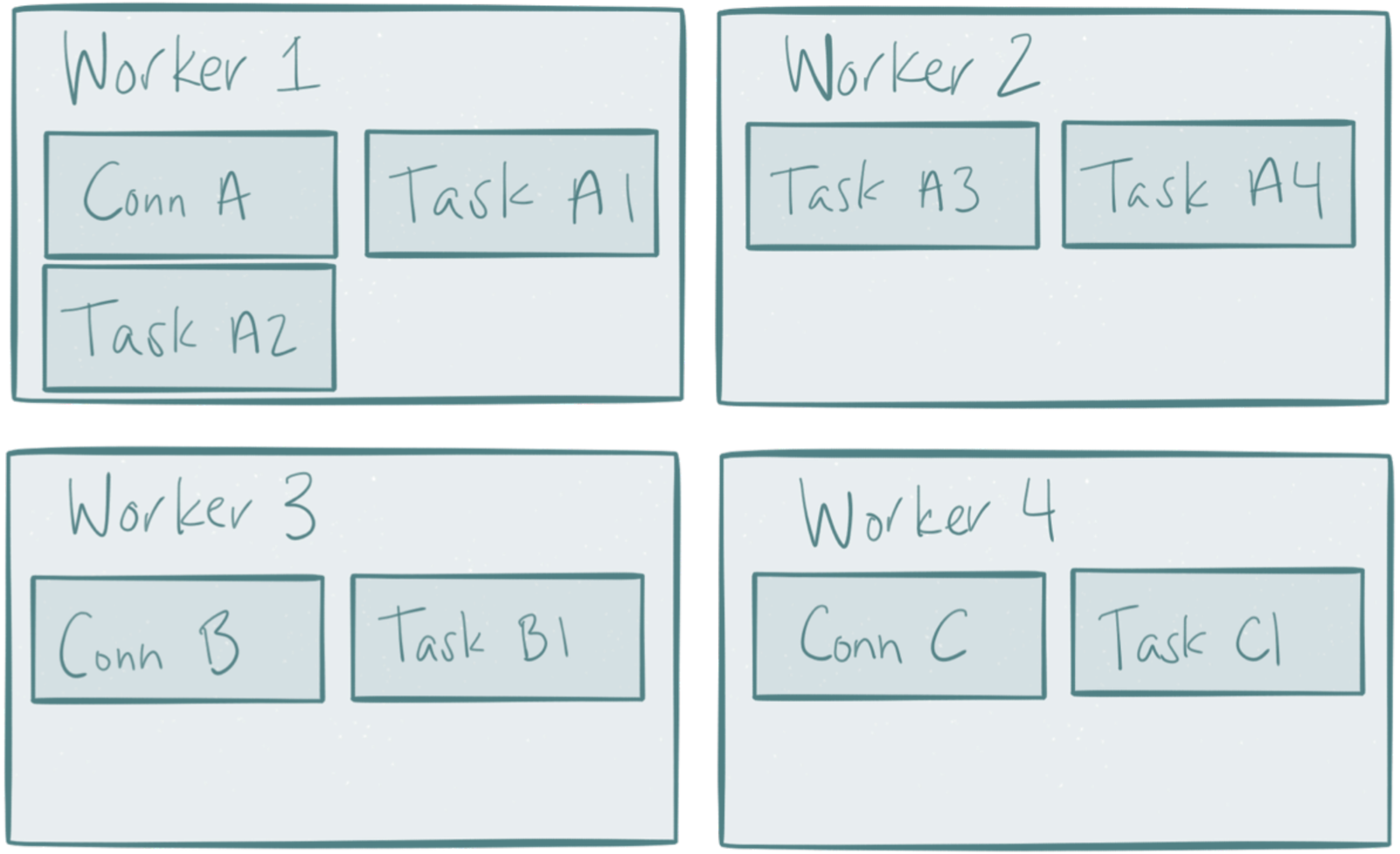
Parallelism Model



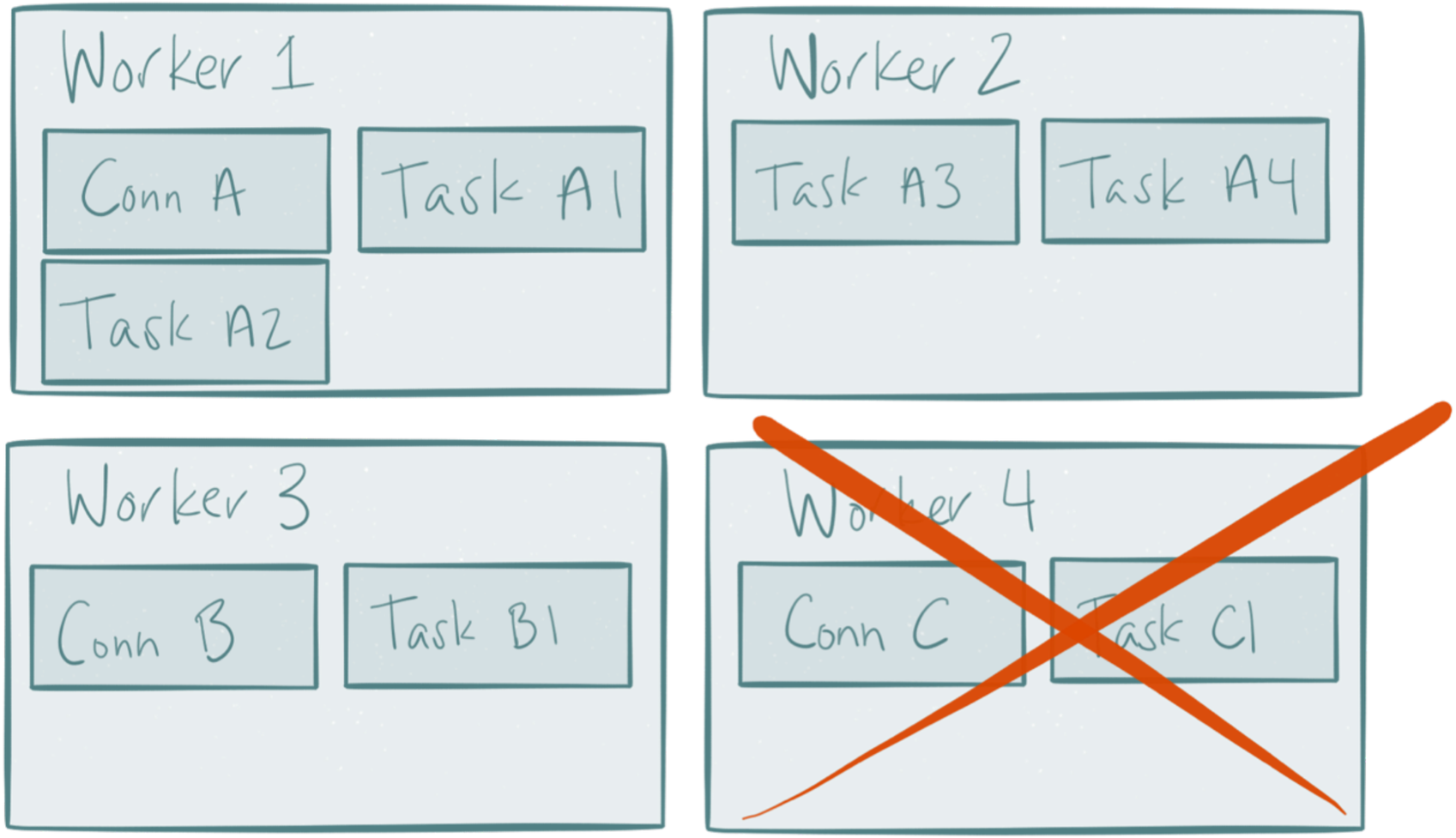
Standalone Execution



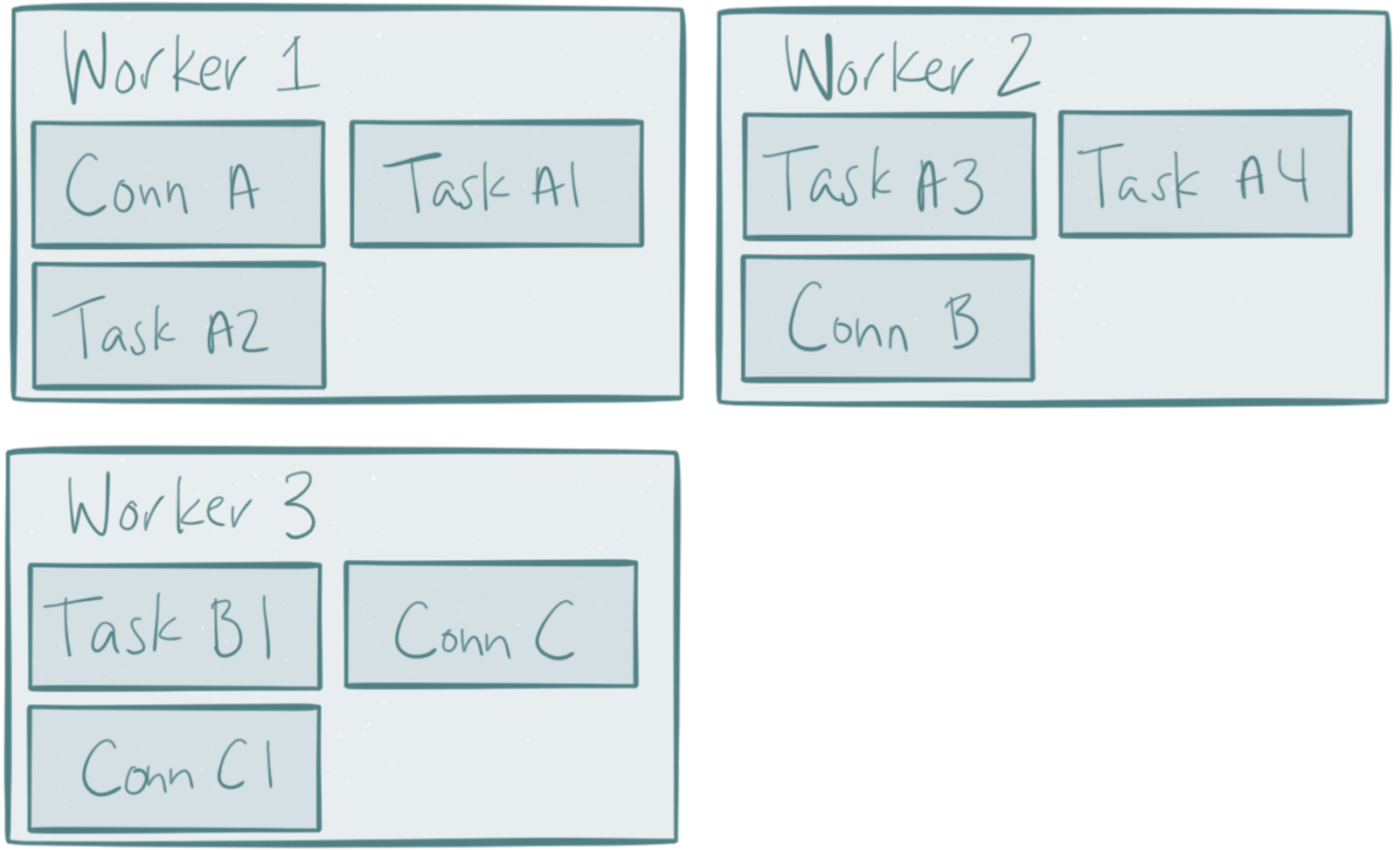
Distributed Execution



Distributed Execution



Distributed Execution



Delivery Guarantees

- Offsets automatically committed and restored
- On restart: task checks offsets & rewinds
- At least once delivery – flush data, then commit
  - Exactly once for connectors that support it (e.g. HDFS)

Format Converters

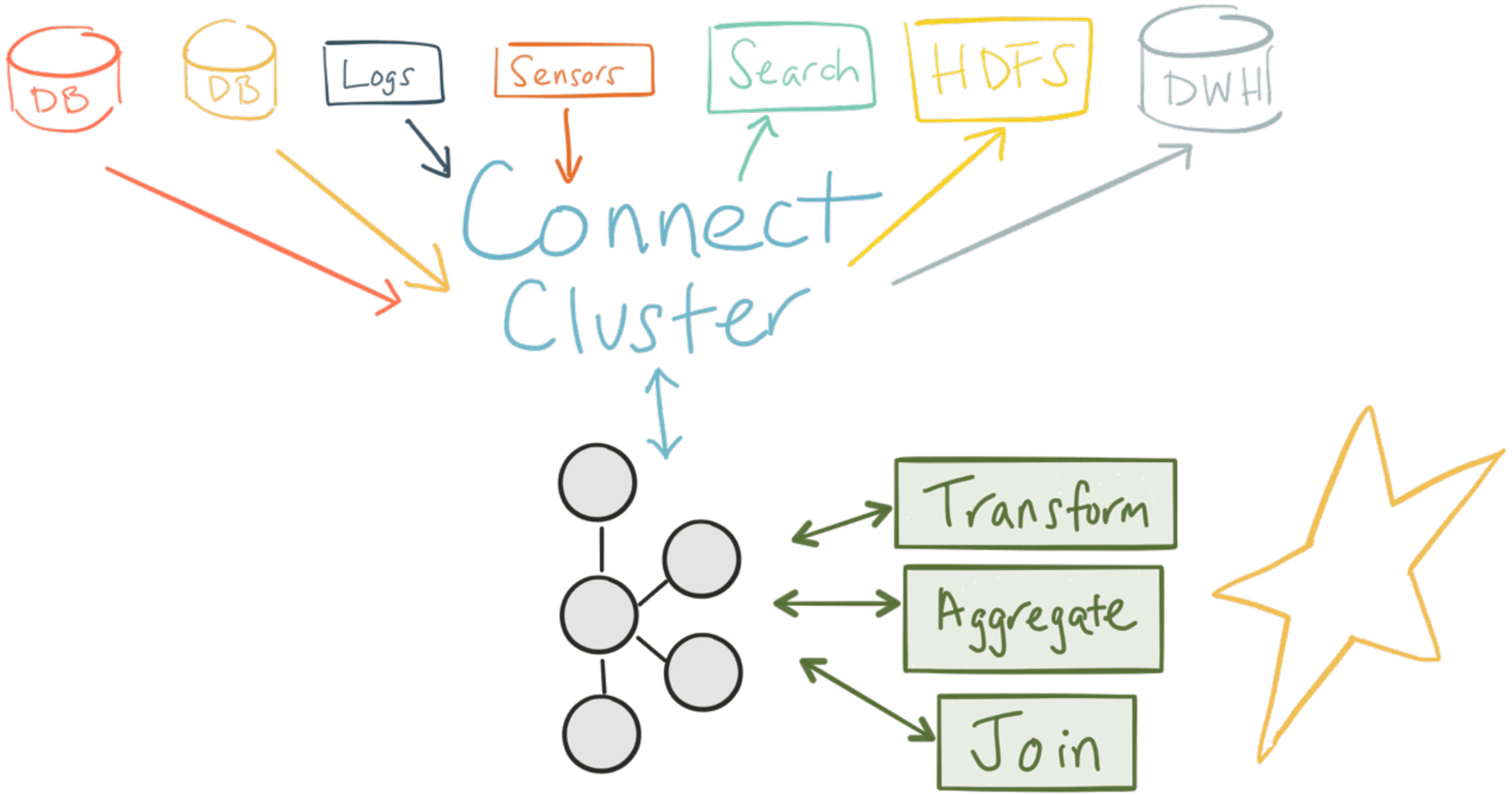
- Abstract serialization agnostic to connectors
  - Convert between Kafka Connect Data API (Connectors) and serialized bytes
  - JSON and Avro currently supported

Connector Developer APIs

```
class Connector {

    abstract void start(props);

    abstract void stop();

    abstract Class<? extends Task> taskClass();

    abstract List<Map<…>> taskConfigs(maxTasks);

    …
}
```

```
class Source/SinkTask {

    abstract void start(props);

    abstract void stop();

    abstract List<SourceRecord> poll();

    abstract void put(records);

    abstract void commit();

    …
}
```

Kafka Connect & Spark Streaming



Kafka Connect Today

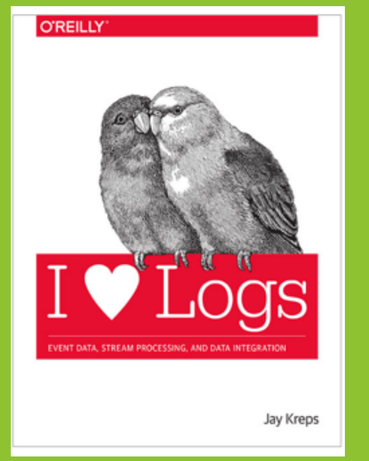
- Confluent open source: HDFS, JDBC
- Connector Hub: connectors.confluent.io
  - Examples: MySQL, MongoDB, Twitter, Solr, S3, MQTT, Counchbase, Vertica, Cassandra, Elastic Search, HBase, Kudu, Attunity, JustOne, Striim, Bloomberg ..
- Improved connector control (0.10.0)

THANK YOU!

Guozhang Wang | guozhang@confluent.io | @guozhangwang

Confluent – Afternoon Break Sponsor for Spark Summit

- Jay Kreps – I Heart Logs book signing and giveaway
- 3:45pm – 4:15pm in Golden Gate



Kafka Training with Confluent University

- Kafka Developer and Operations Courses
- Visit www.confluent.io/training

Want more Kafka?

- Download Confluent Platform Enterprise (incl. Kafka Connect) at http://www.confluent.io/product
- Apache Kafka 0.10 upgrade documentation at http://docs.confluent.io/3.0.0/upgrade.html

SPARK SUMMIT 2016
DATA SCIENCE AND ENGINEERING AT SCALE
JUNE 6-8, 2016 SAN FRANCISCO