KubeCon | CloudNativeCon

Europe 2018

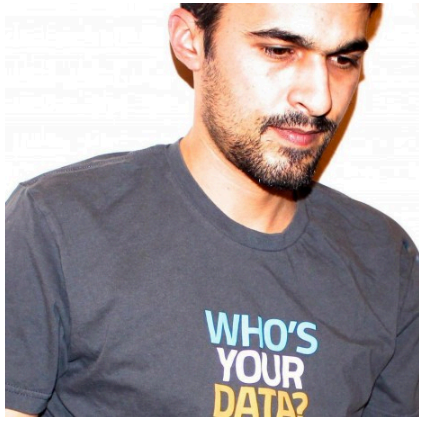
Deploying SQL Stream Processing in Kubernetes with Ease

LANDOOP

KubeCon | CloudNativeCon

Europe 2018

Andrew Stevenson
CTO Landoop
Big Data
Fast Data
Financial Markets
andrew@landoop.com



Antonios Chalkiopoulos
CEO Landoop
Big Data
Fast Data
Author
antonios@landoop.com

www.landoop.com [@LandoopLtd]()

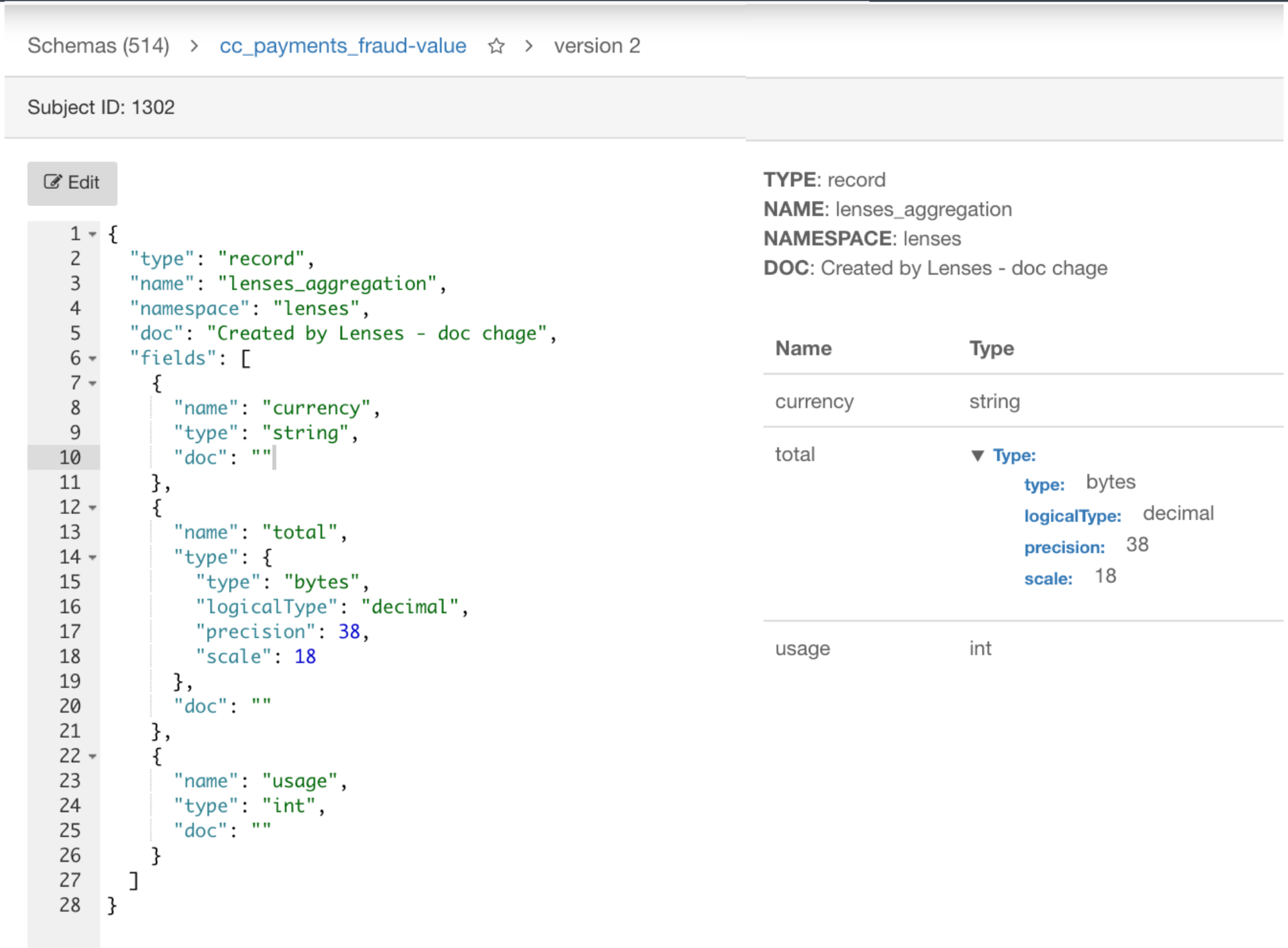
LANDOOP

From basic data containers like JSON

```
{
  customer: {
        name: “nameA”,
        address: “”
```

LANDOOP

To modern data containers like Apache Avro



Schemas (514) › cc_payments_fraud-value ☆ › version 2

Subject ID: 1302

Edit

```
{
  "type": "record",
  "name": "lenses_aggregation",
  "namespace": "lenses",
  "doc": "Created by Lenses - doc chage",
  "fields": [
    {
      "name": "currency",
      "type": "string",
      "doc": ""
    },
    {
      "name": "total",
      "type": {
        "type": "bytes",
        "logicalType": "decimal",
        "precision": 38,
        "scale": 18
      },
      "doc": ""
    },
    {
      "name": "usage",
      "type": "int",
      "doc": ""
    }
  ]
}
```

TYPE: record
NAME: lenses_aggregation
NAMESPACE: lenses
DOC: Created by Lenses - doc chage

Name	Type
currency	string
total	▼ **Type:** **type:** bytes **logicalType:** decimal **precision:** 38 **scale:** 18
usage	int

Performant binary format

Data contract

Type and pipeline safety

Data evolution

Metadata for Privacy / Regulations

LANDOOP

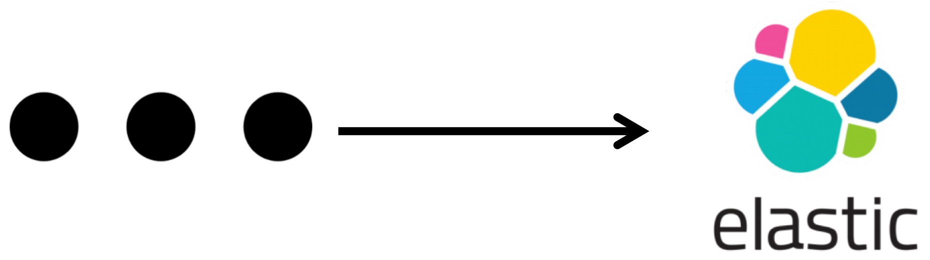
SQL Makes Sense!

1 To query data

```
SELECT * FROM .. WHERE customer.country='CA'
```

SQL Makes Sense!

2 To build data integrations

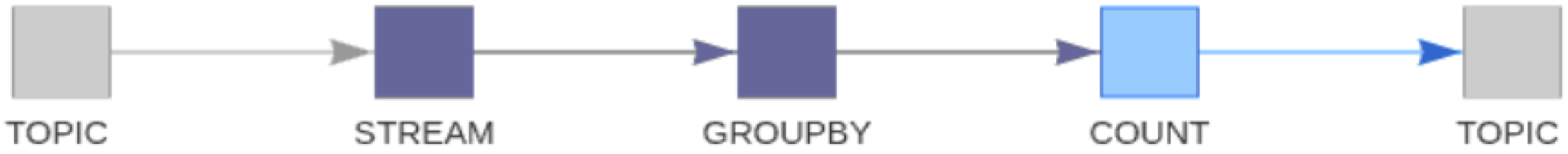


```
UPSERT INTO elasticSearchIndex
SELECT MMSI AS vessel_id, location FROM position_reports
PK MMSI
```

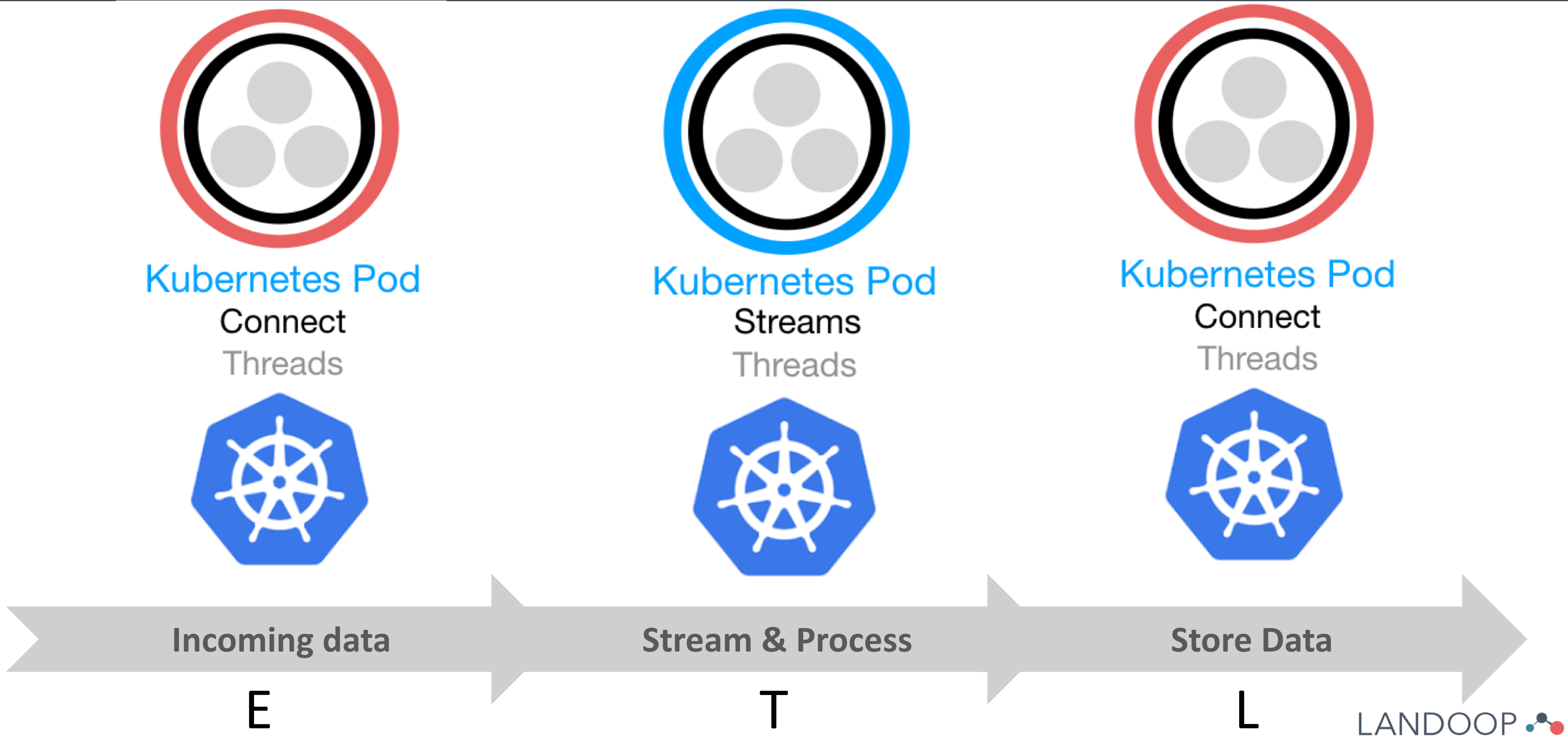
SQL Makes Sense!

3 To operating streaming pipelines

```
INSERT INTO …
SELECT STREAM
   COUNT(*) AS total
FROM payments
GROUP BY TUMBLE(1, m)
```



And when everything is stateless (nearly)



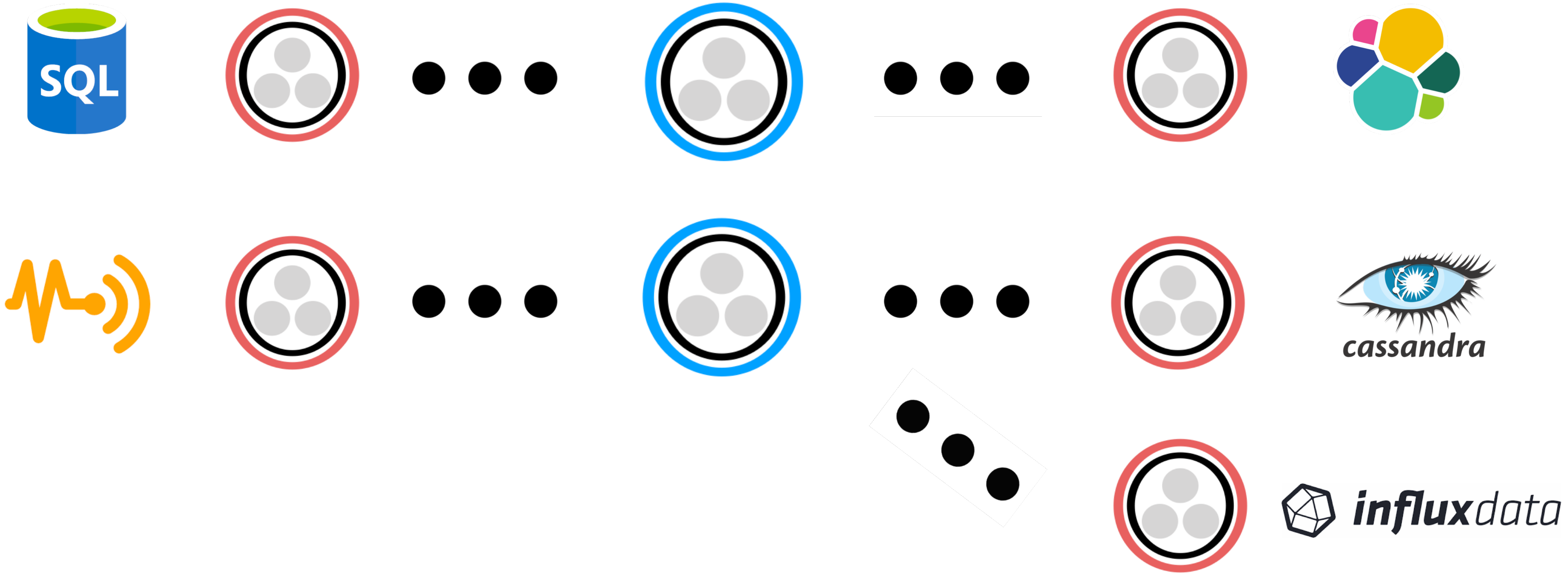
LANDOOP

And when everything is a config

You can drive your CI/CD and store everything in

git

We want to be operating streaming pipelines

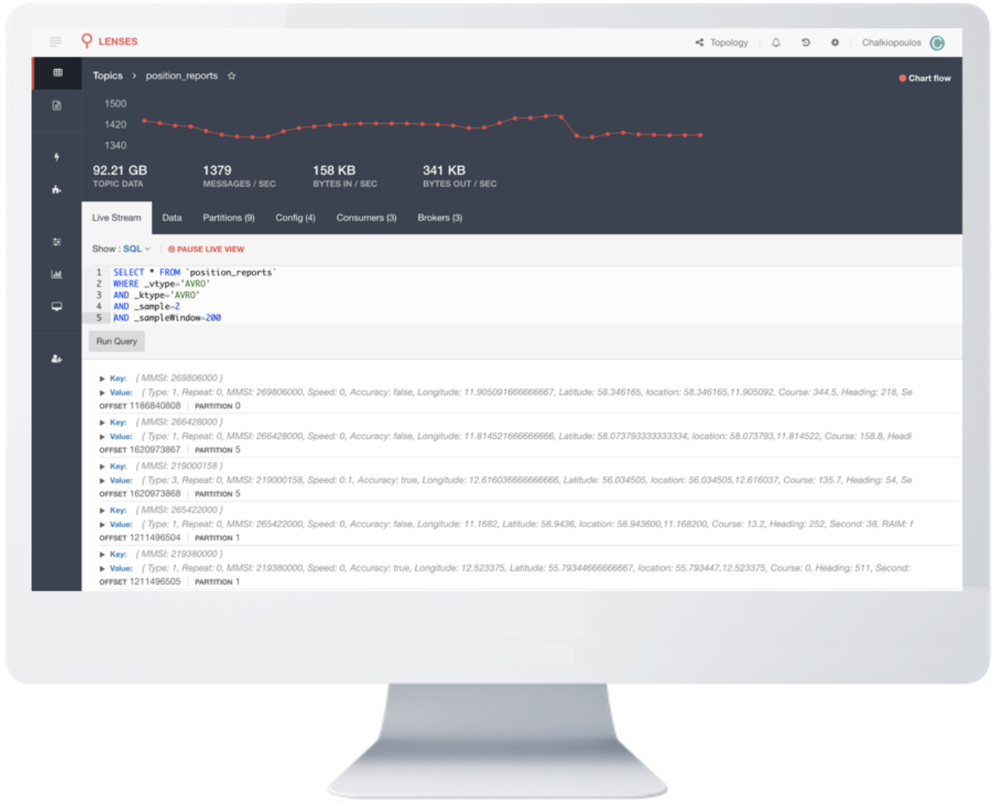


And how about my state ?

We need a distributed and parallel file-system ● ● ●

Who we are

Industrial grade streaming platform for Apache Kafka



```
SELECT * FROM `position_reports`
WHERE _vtype='AVRO'
AND _ktype='AVRO'
AND _sample=2
AND _sampleWindow=200
```

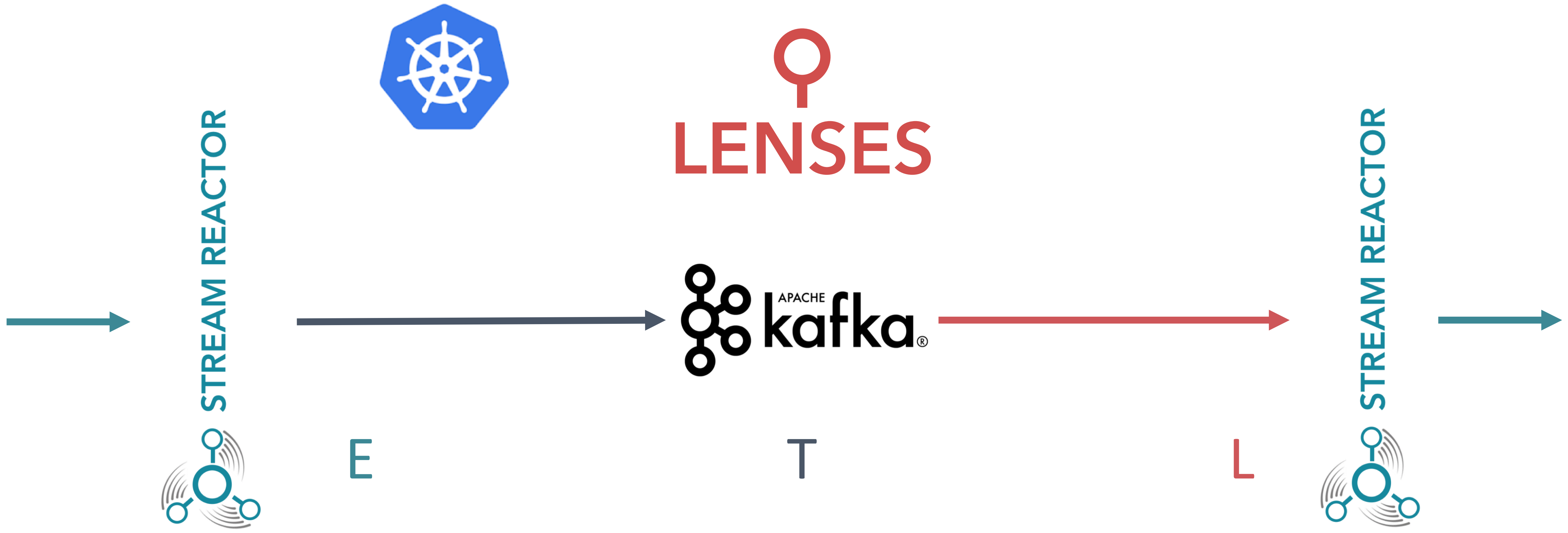
```
▶ Key:   { MMSI: 269806000 }
▶ Value:   { Type: 1, Repeat: 0, MMSI: 269806000, Speed: 0, Accuracy: false, Longitude: 11.905091666666667, Latitude: 58.346165, location: 58.346165,11.905092, Course: 344.5, Heading: 218, Se
OFFSET 1186840808 | PARTITION 0
▶ Key:   { MMSI: 266428000 }
▶ Value:   { Type: 1, Repeat: 0, MMSI: 266428000, Speed: 0, Accuracy: false, Longitude: 11.814521666666666, Latitude: 58.073793333333334, location: 58.073793,11.814522, Course: 158.8, Headi
OFFSET 1620973867 | PARTITION 5
▶ Key:   { MMSI: 219000158 }
▶ Value:   { Type: 3, Repeat: 0, MMSI: 219000158, Speed: 0.1, Accuracy: true, Longitude: 12.616036666666666, Latitude: 56.034505, location: 56.034505,12.616037, Course: 135.7, Heading: 54, Se
OFFSET 1620973868 | PARTITION 5
▶ Key:   { MMSI: 265422000 }
▶ Value:   { Type: 1, Repeat: 0, MMSI: 265422000, Speed: 0, Accuracy: false, Longitude: 11.1682, Latitude: 58.9436, location: 58.943600,11.168200, Course: 13.2, Heading: 252, Second: 38, RAIM: f
OFFSET 1211496504 | PARTITION 1
▶ Key:   { MMSI: 219380000 }
▶ Value:   { Type: 1, Repeat: 0, MMSI: 219380000, Speed: 0, Accuracy: true, Longitude: 12.523375, Latitude: 55.79344666666667, location: 55.793447,12.523375, Course: 0, Heading: 511, Second:
OFFSET 1211496505 | PARTITION 1
```

LENSES

STREAM REACTOR

LANDOOP

Data Pipelines



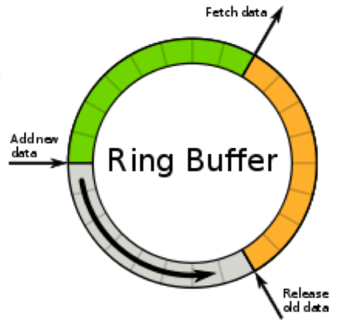
LANDOOP

Getting data in

STREAM REACTOR

- Streaming data integration made easy with Kafka Connect
- 30 + Connectors, open source
- Dockers and Helm charts
- SQL support, **K**afka **C**onnect **Q**uery **L**anguage => KCQL

KCQL



```
{ "sensor_id": "01" , "temperature": 52.7943, "ts": 1484648810 }
{ "sensor_id": "02" , "temperature": 28.8597, "ts": 1484648810 }
```

```
INSERT INTO sensor_ringbuffer
SELECT sensor_id, temperature, ts
FROM coap_sensor
WITHFORMAT JSON
STOREAS RING_BUFFER
```

```
INSERT INTO sensor_reliabletopic
SELECT sensor_id, temperature, ts
FROM coap_sensor
WITHFORMAT AVRO
STOREAS RELIABLE_TOPIC
```

Connectors

kafka-connect-blockchain
kafka-connect-bloomberg
kafka-connect-cassandra
kafka-connect-coap
kafka-connect-druid
kafka-connect-elastic
kafka-connect-ftp
kafka-connect-hazelcast
kafka-connect-hbase

kafka-connect-influxdb
kafka-connect-jms
kafka-connect-kudu
kafka-connect-mongodb
kafka-connect-mqtt
kafka-connect-redis
kafka-connect-rethink
kafka-connect-voltdb
Kafka-connect-pulsar

https://github.com/Landoop/stream-reactor https://github.com/Landoop/kafka-helm-charts

LANDOOP

Stream Reactor in Kubernetes

Workers are JVM apps (same group.id)

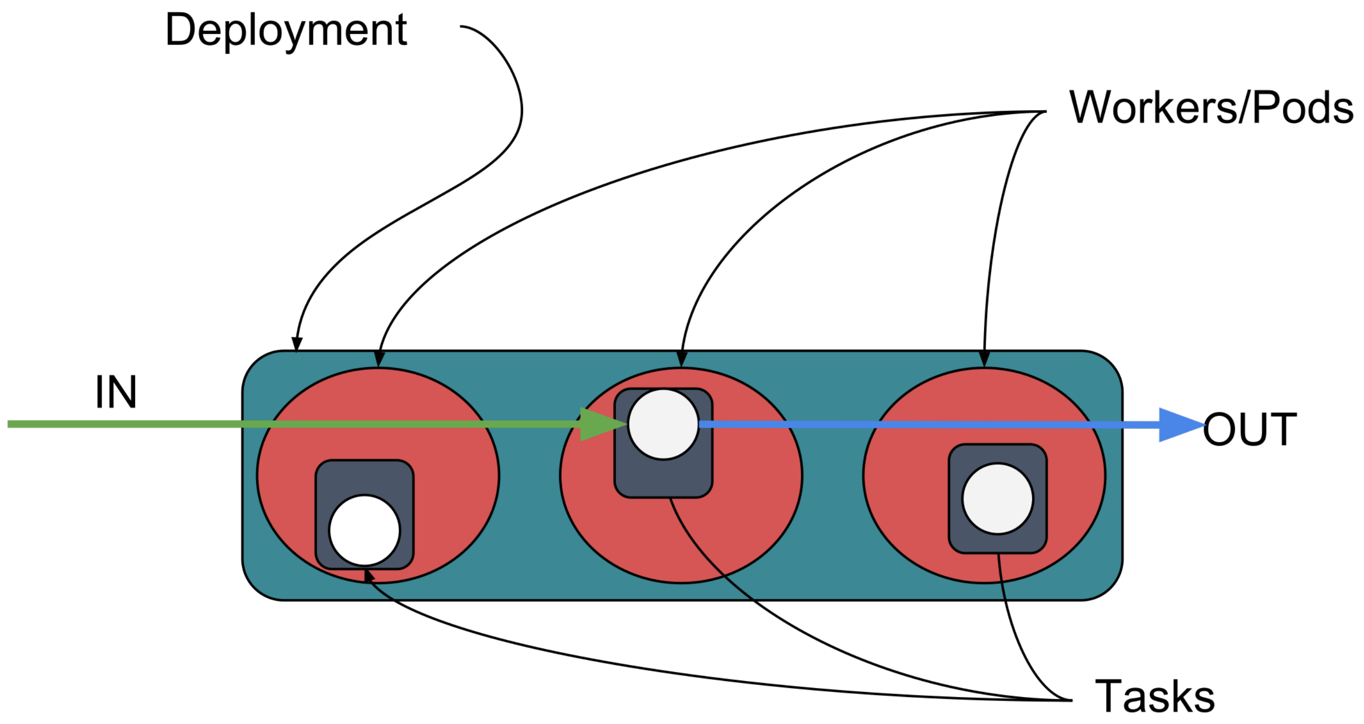
↓

Kafka sees them as a consumer group, consuming from the same config topics

Workers listen to config topics and spin tasks/threads when told

↓

Tasks are new consumer groups (sink) on data topics



LANDOOP

Stream Reactor in K8. The good

- K8s ensures our desired number of workers is applied
- State is persisted in Apache Kafka
- Easy to deploy and scale

LANDOOP

Stream Reactor in K8. The not so good

Connect rebalances vs K8 maintaining desired state:

Too Many Rebalances

Rebalancing on:

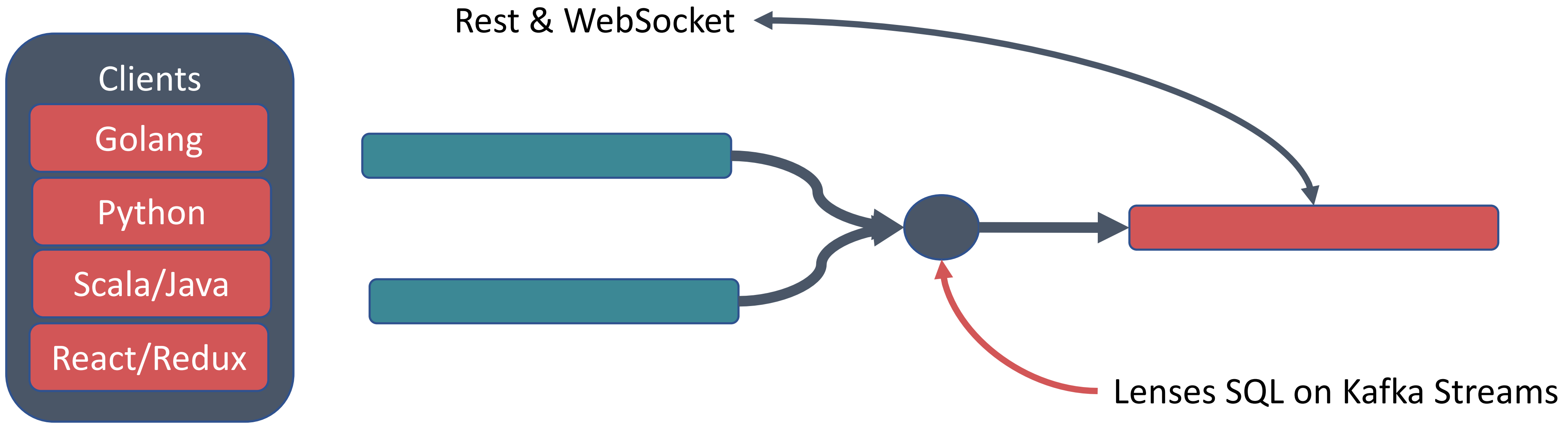
- New worker/pod
- Removal of worker/pod
- **Adding a new connector**

Advice:

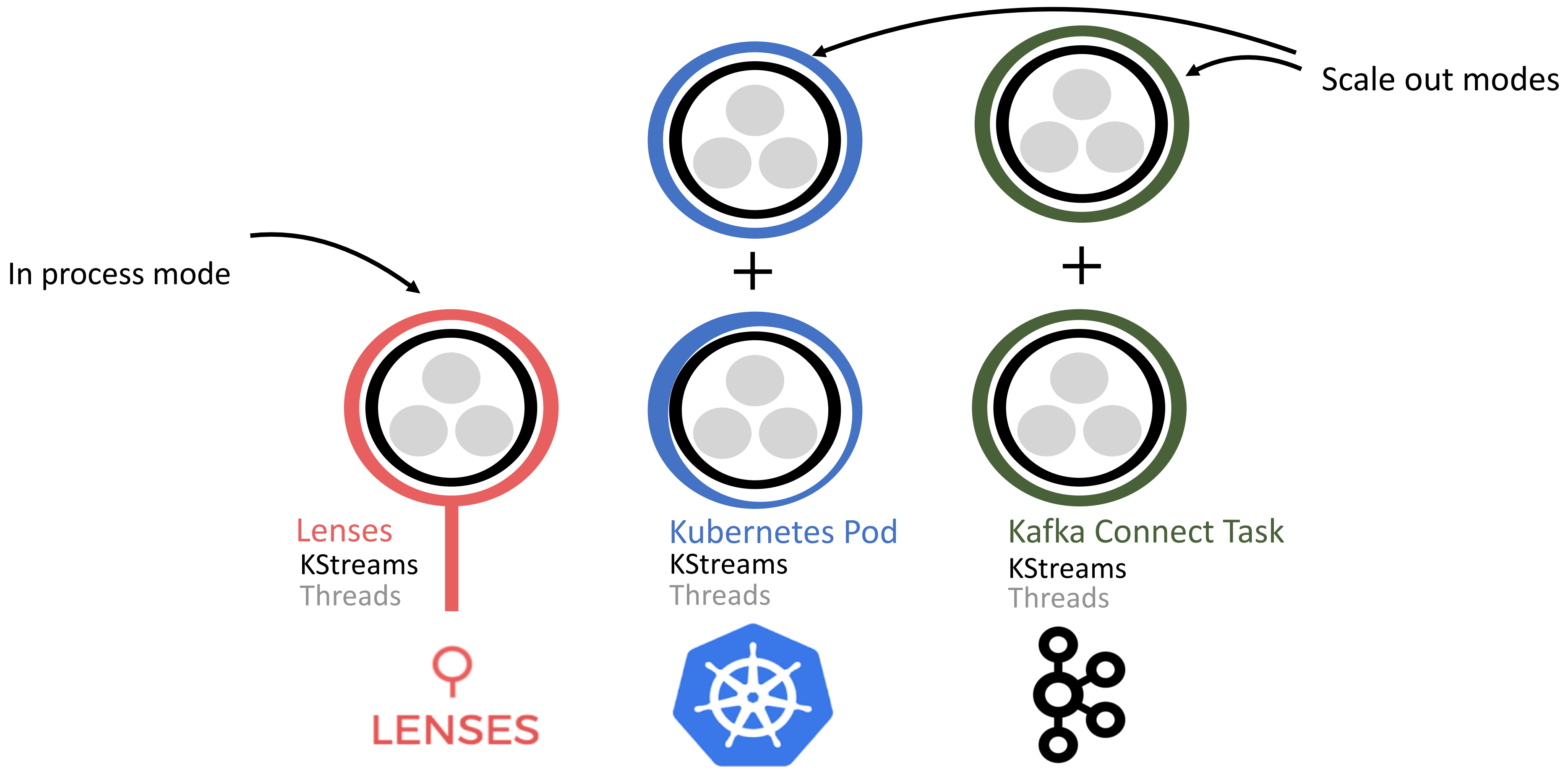
- Liveliness probes
  - Task failed/Connect worker
- 1 Connector per deployment

Stream processing with SQL in Kubernetes made easy

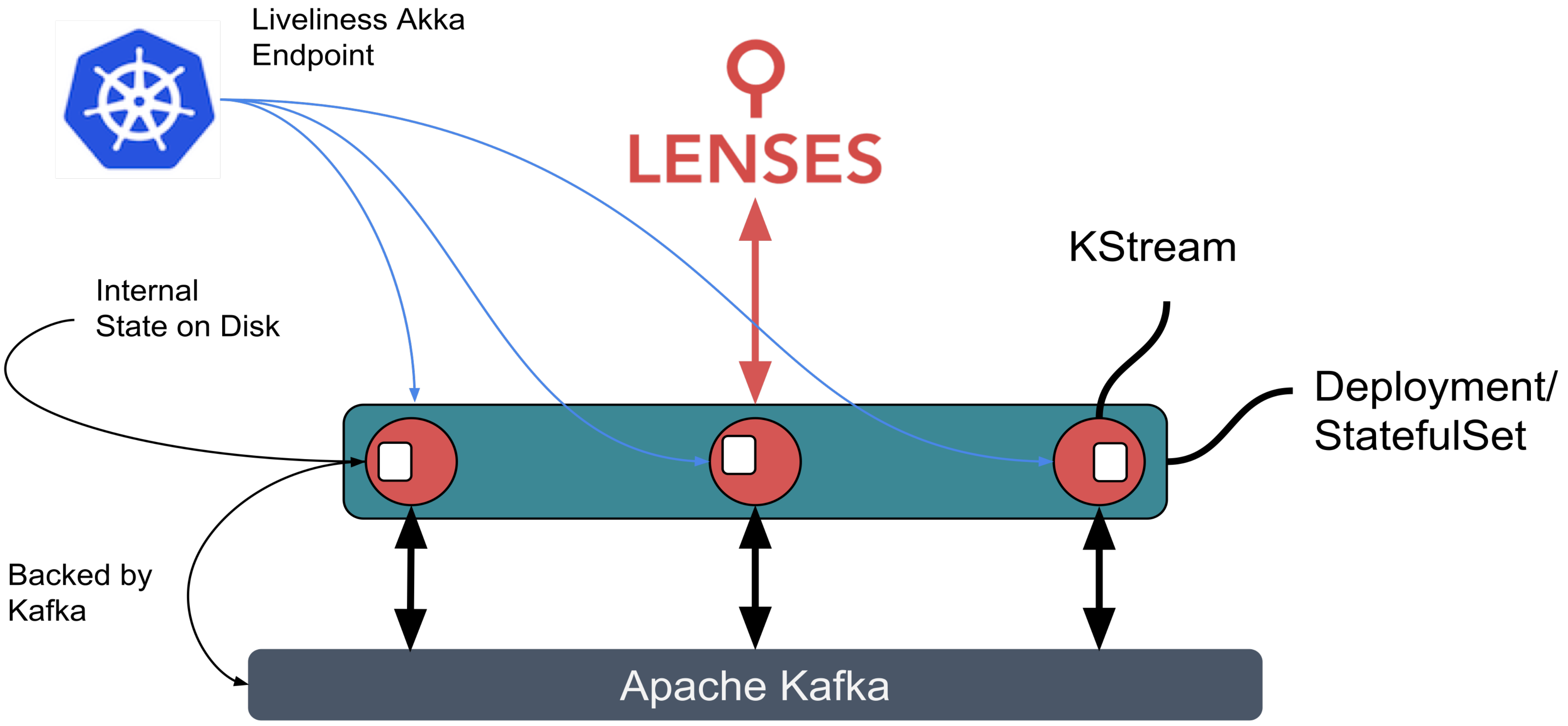
LENSES



Execution Modes



Kubernetes Modes



LANDOOP

Kubernetes Manifest, important bit

Config is Code => SQL is Config

```
resources:
{{ toYaml .Values.resources | indent 10 }}
        env:
          - name: SQL
          value: |-
                SET autocreate=true;

                INSERT INTO fastVehiclesProcessor
                SELECT MMSI, Speed, Longitude AS Long, Latitude AS Lat, `Timestamp`
                FROM iot_data
                WHERE Speed > 10
                AND _ktype=AVRO AND _vtype=AVRO
```

LANDOOP

SQL Processors in Kubernetes

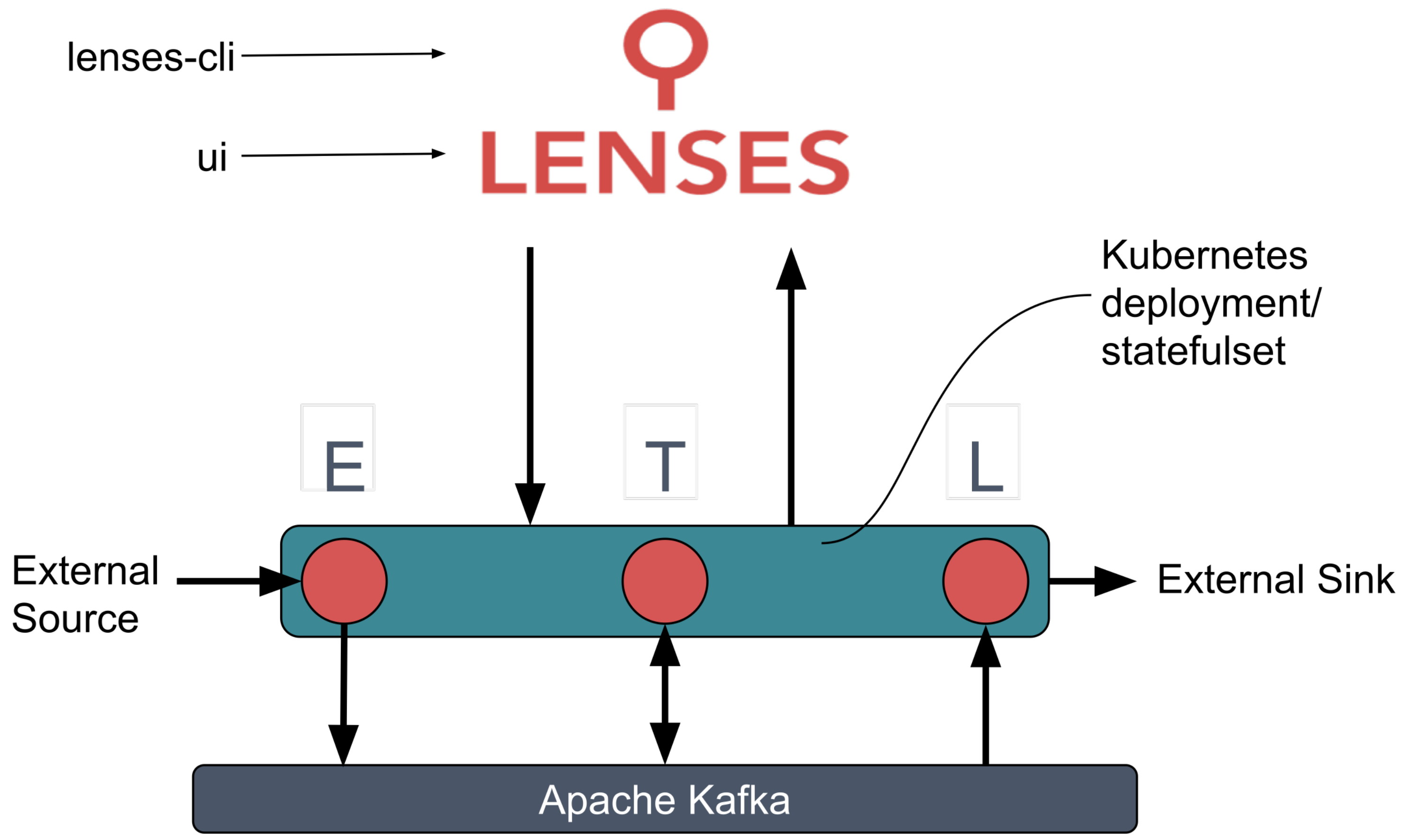
- Kafka rebalances the data streams for us
- Kubernetes ensures our desired number of consumers is applied
- Config is code
  - Prebuilt docker chassis with monitoring included
  - SQL is code, configure runner via environment variables
- State?
  - Yes, aggregating, joining, backed up to Apache Kafka
- If possible use StatefulSets
  - KStreams will bootstrap itself from the rocksdb on disk speeding recovery times

Deploying SQL Processors

Lenses gives you:

- Auditing
- Security polices on Apache Kafka
- Topic white/black listing
- Quota management
- Visualise topologies and export topologies
- Websocket, rest and JDBC
- Monitoring + Prometheus
- Alerts with Alert Manager

Lenses SQL Processing in Kubernetes



LANDOOP

Quick Demo

LENSES

LANDOOP

Questions?

LANDOOP