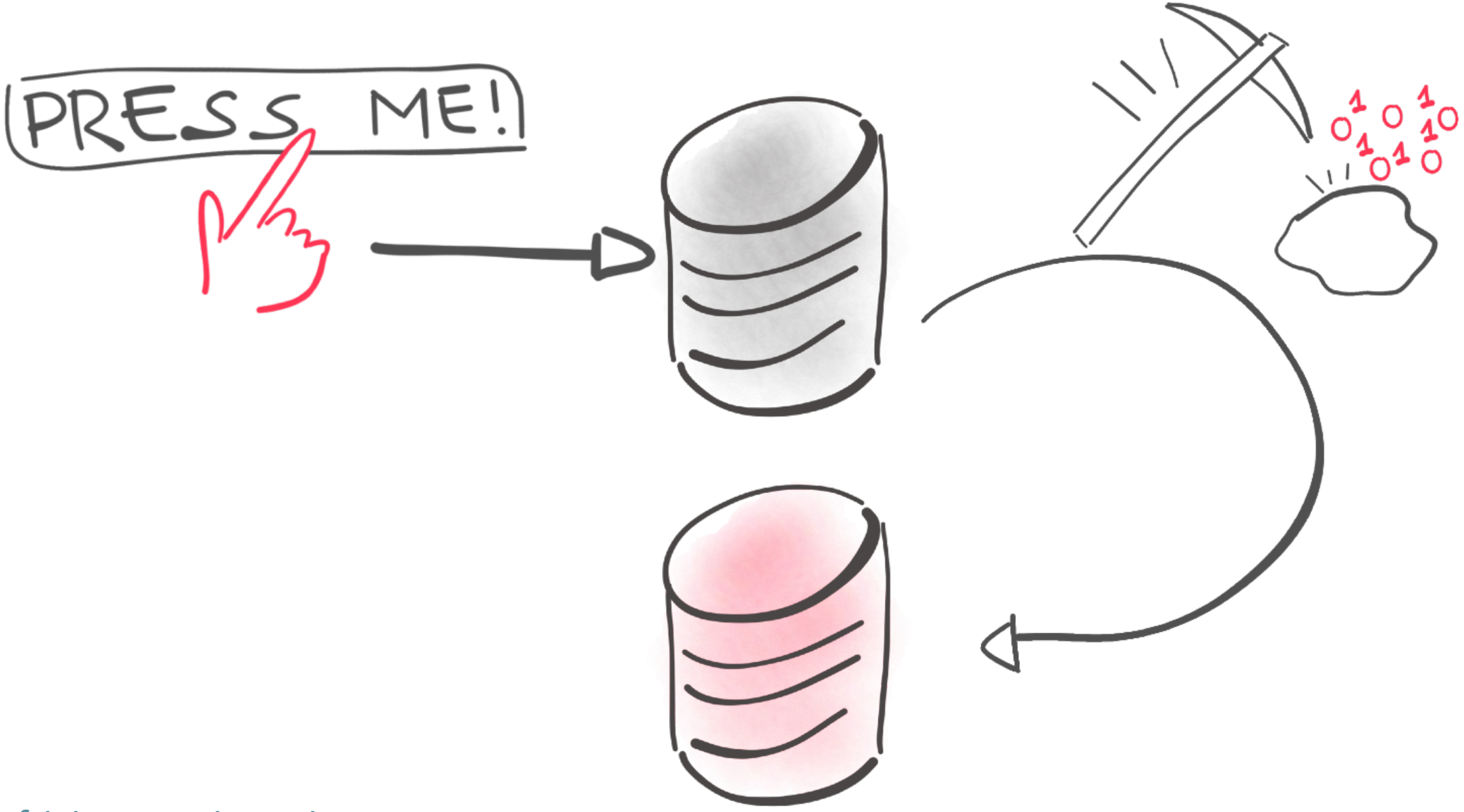
aiven

Apache Kafka and Flink

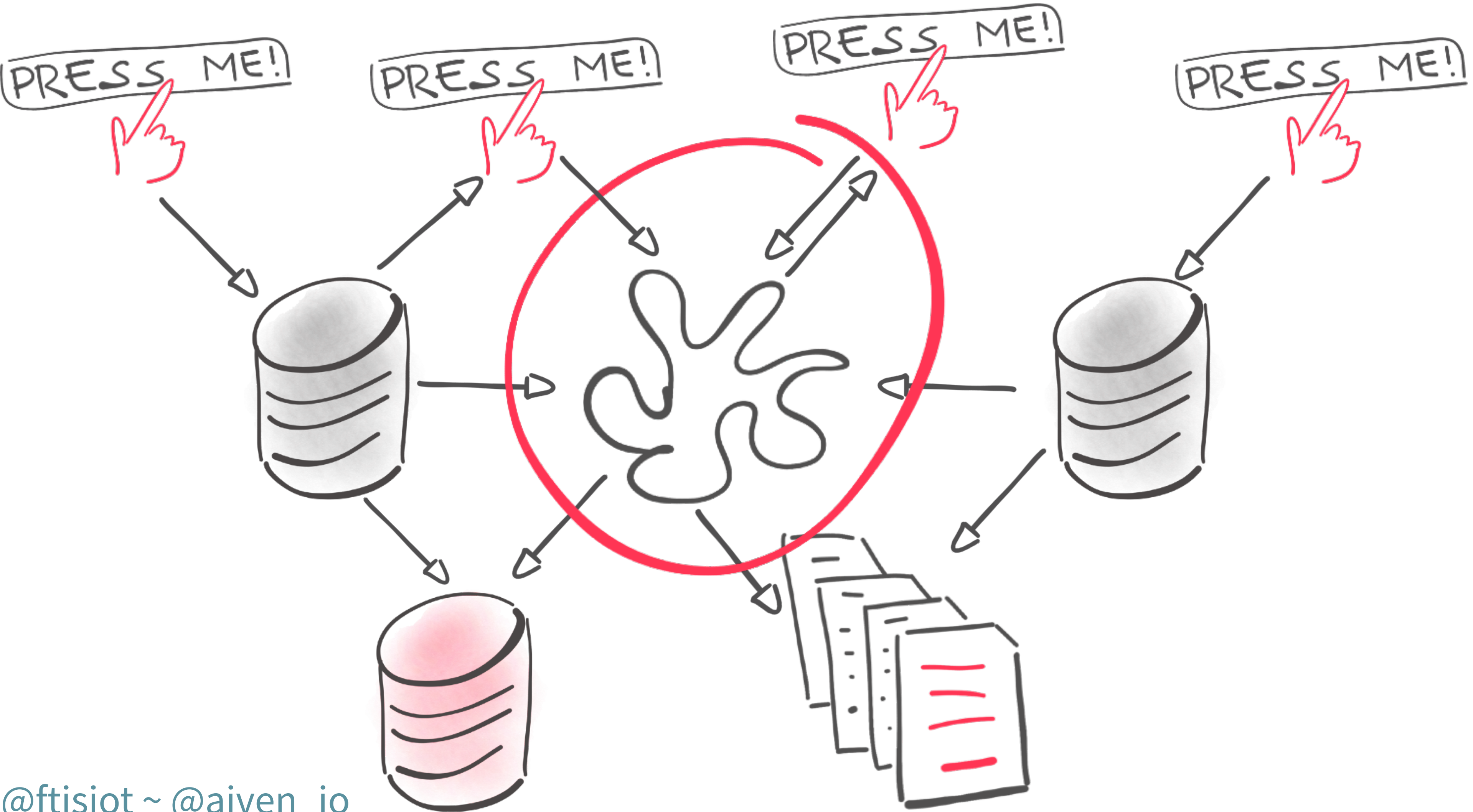
Stateful Streaming Data Pipelines made easy with SQL

Francesco Tisiot - Developer Advocate

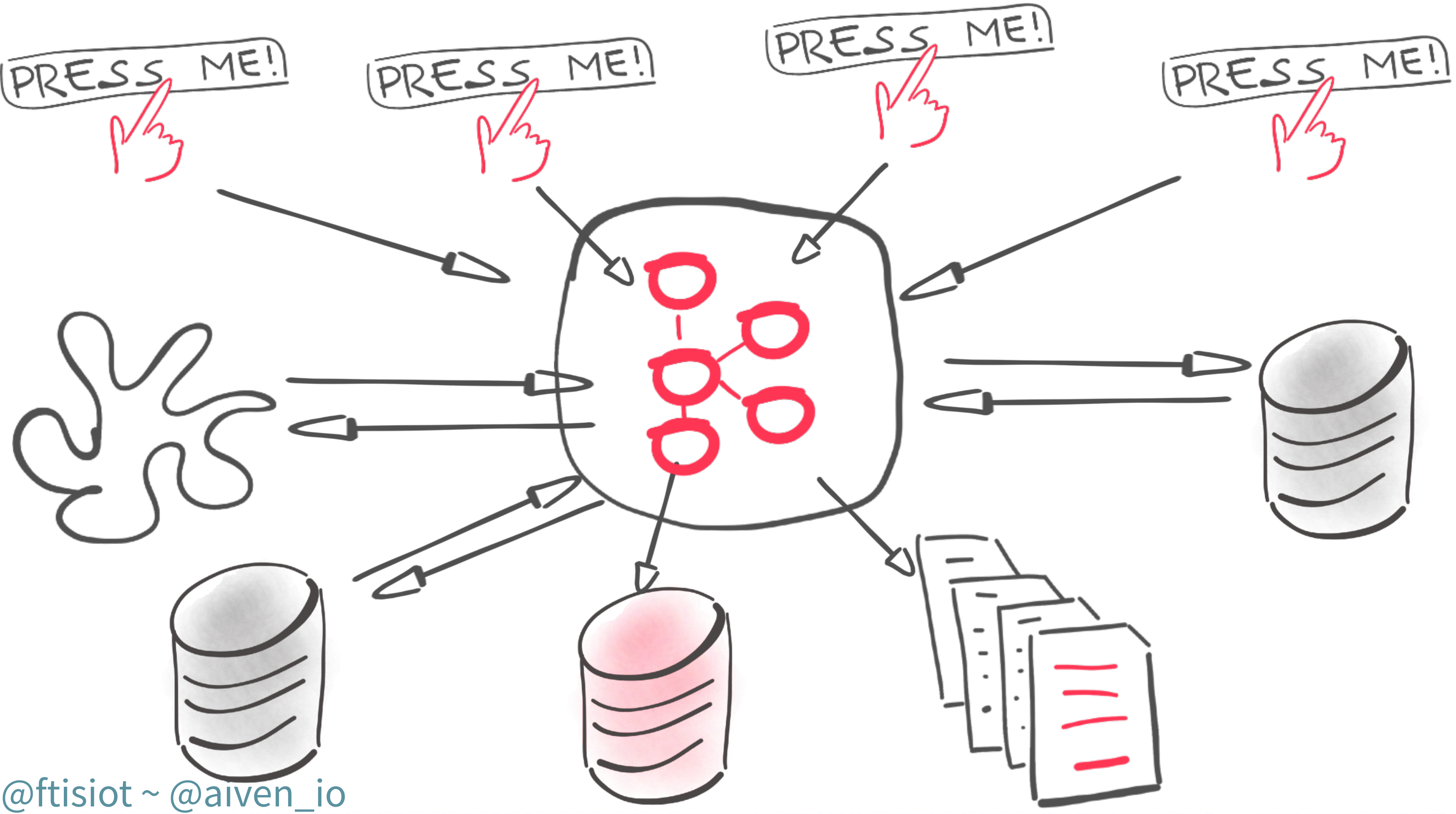
@ftisiot

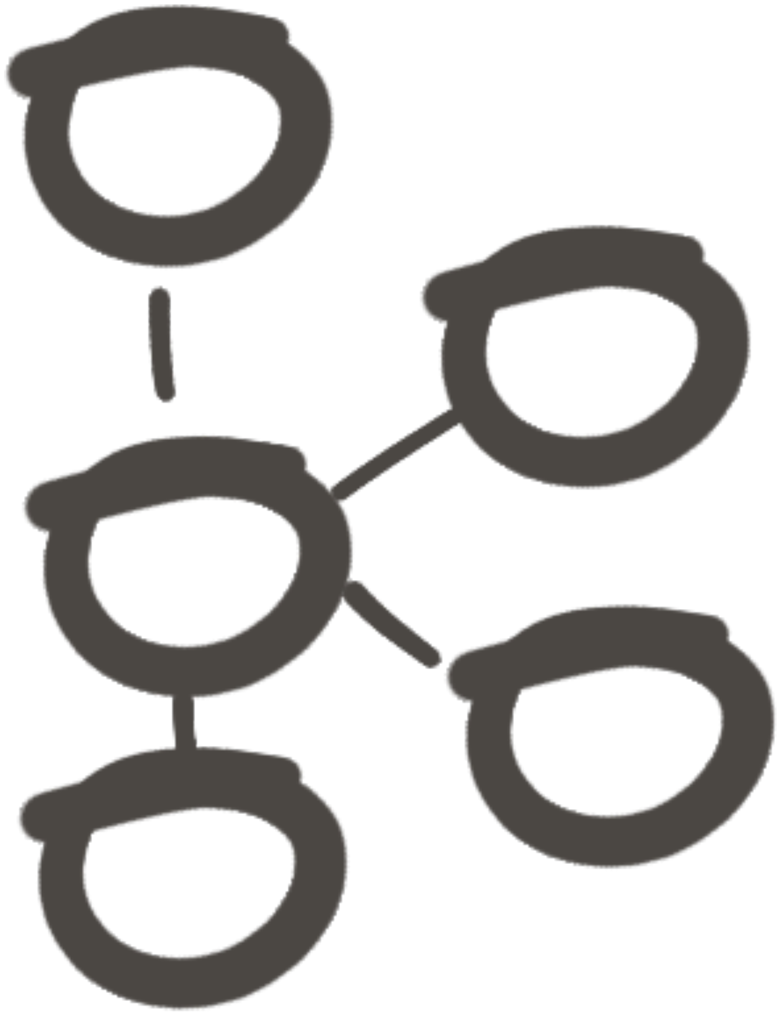

PRESS ME!
0 1 0 1 0 1 0 1 0 1 0

OLD



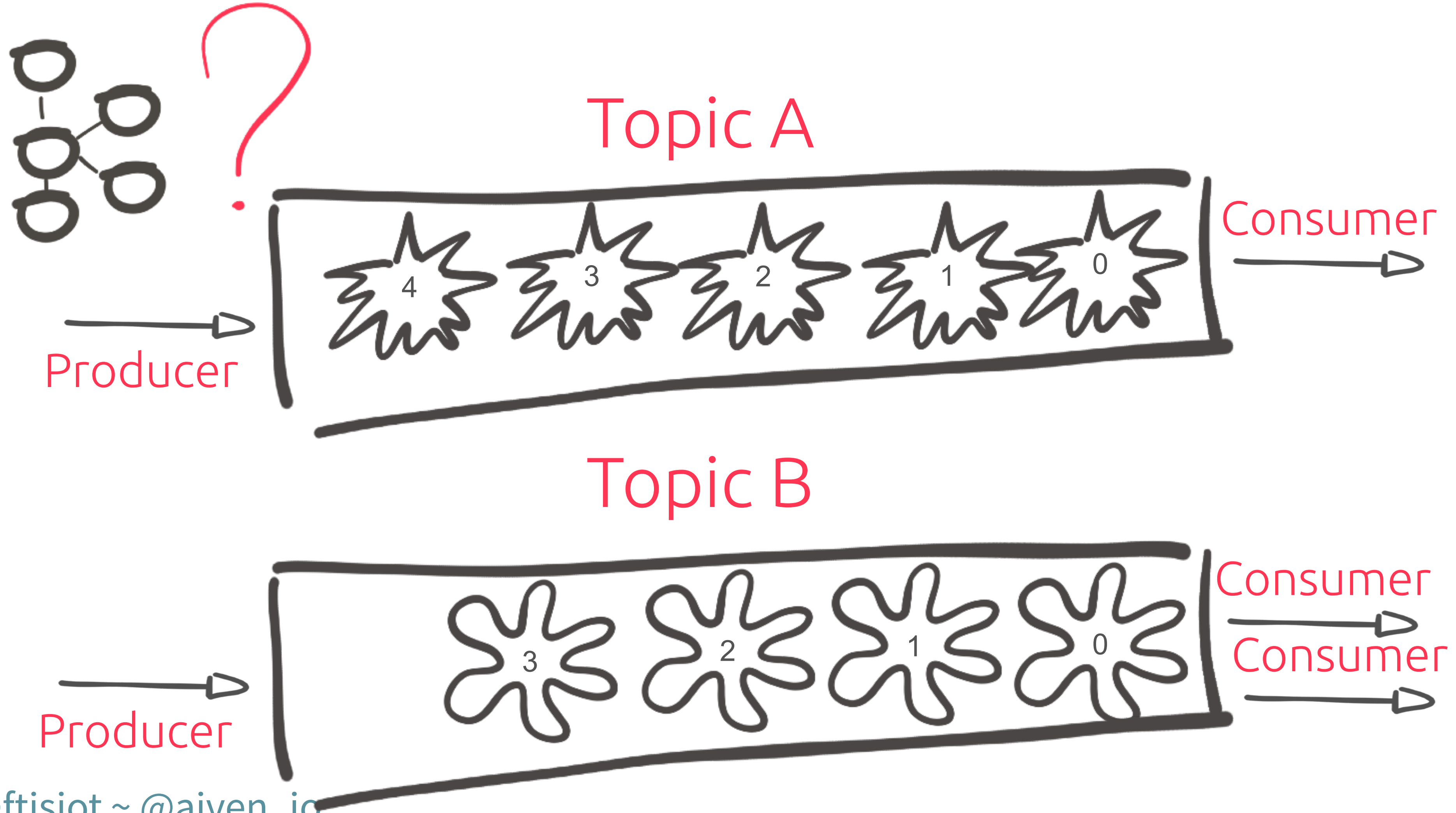
@ftisiot ~ @aiven_io

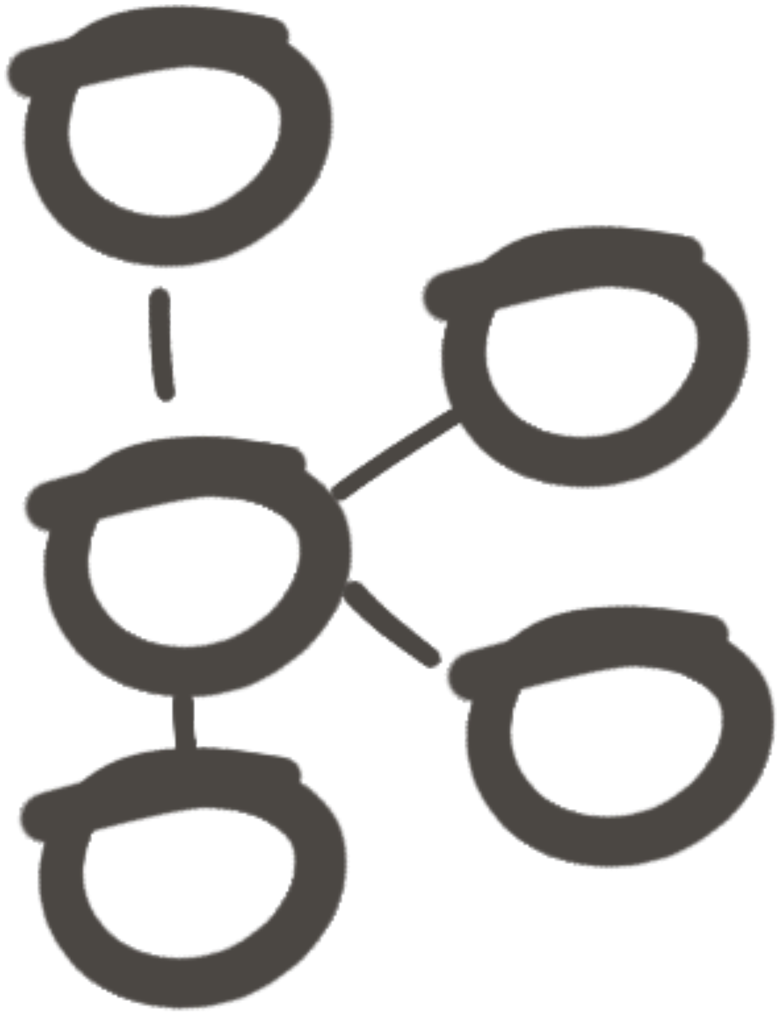

PRESS ME!
PRESS ME!
PRESS ME!
PRESS ME!



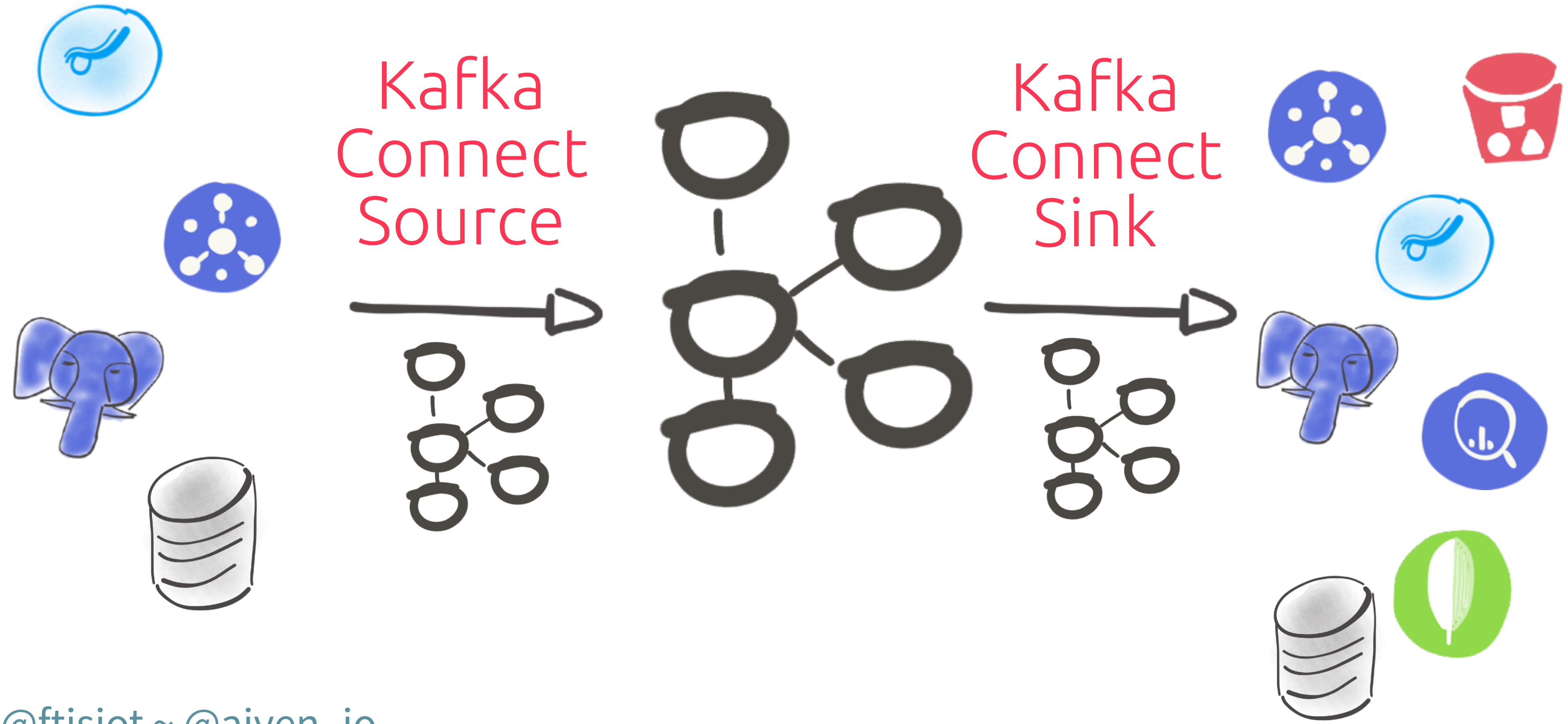
What is Apache Kafka?

@ftisiot ~ @aiven_io


Topic A
4
3
2
1
0
Producer
Consumer
Topic B
3
2
1
0
Producer
Consumer
Consumer



Integrating Apache Kafka


Kafka Connect Source
Kafka Connect Sink

kafka-python

```
from kafka import KafkaProducer

producer = KafkaProducer(
  bootstrap_servers=['broker1:1234']
  )

producer.send(
  'my-topic-name',
  b'my-message'
  )

producer.flush()
```

```
{
  "id": 1,
  "shop": “Mario's Pizza",
  "name": "Arsenio Pisaroni–Boccaccio",
  "phoneNumber": "+39 51 0290746",
  "address": "Via Ugo 01, Montegrotto, 85639 Padova(PD)",
  "pizzas": [
    {
      "pizzaName": "Margherita",
      "additionalToppings": ["ham"]
    },
    {
      "pizzaName": "Diavola",
      "additionalToppings": ["mozzarella","banana","onion"]
    }]
}
```

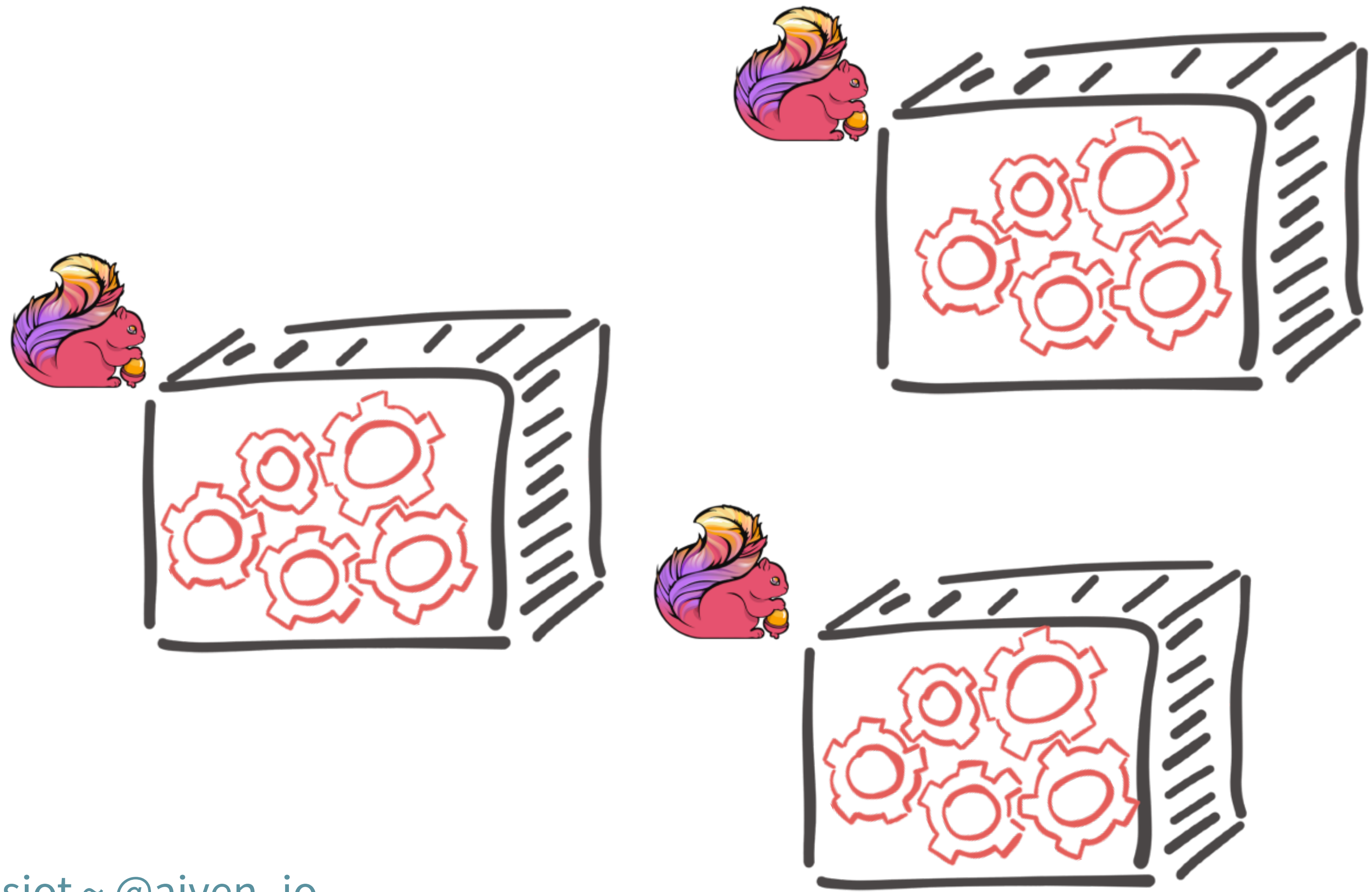
Create Stateful Apps

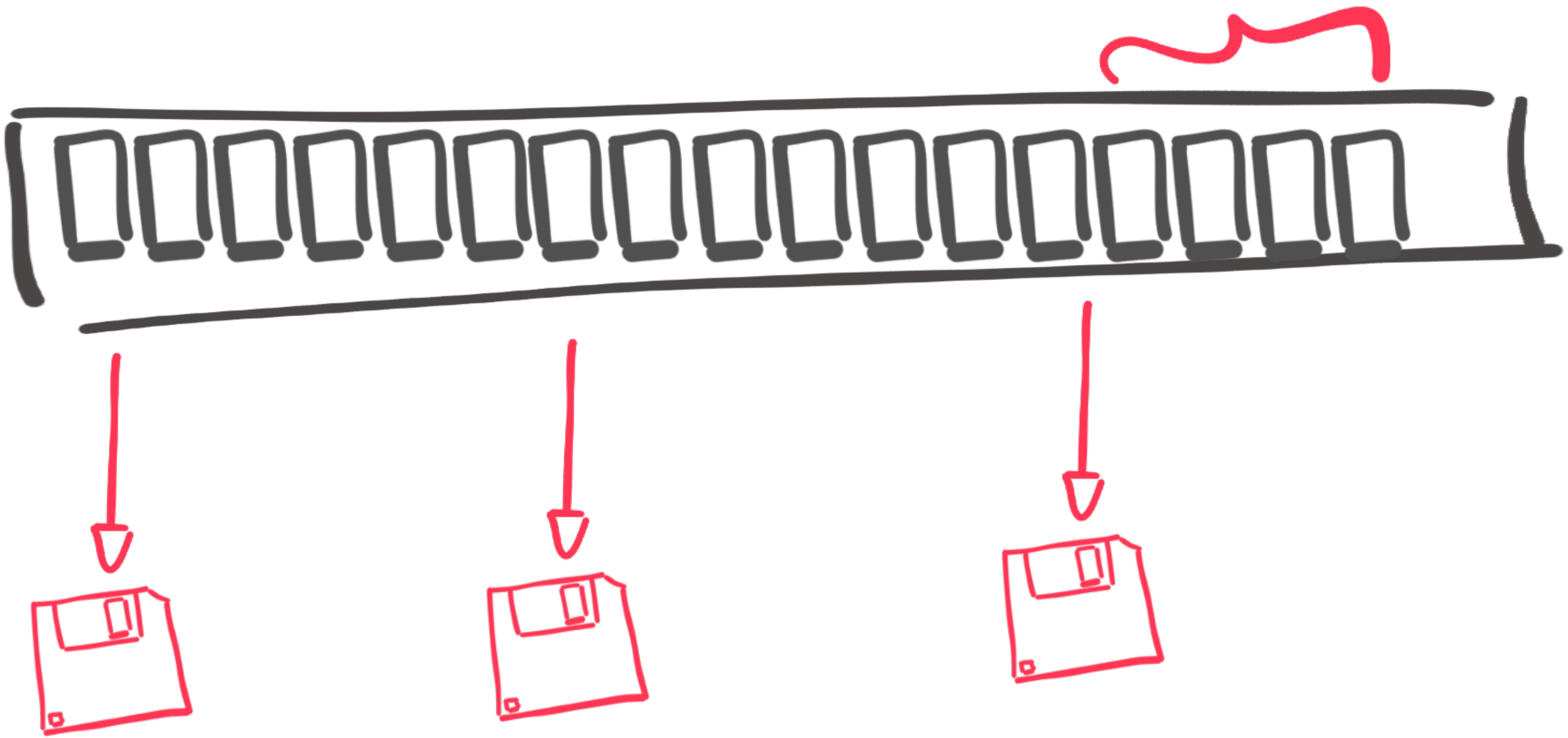
Kafka Streams

Faust

KSQL

Apache Flink





SQL

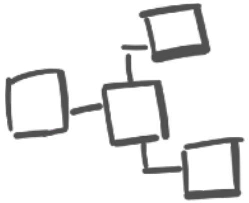
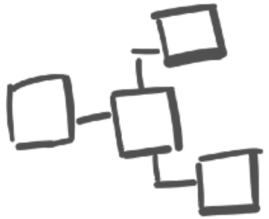


Table API

DataStream API

Filter



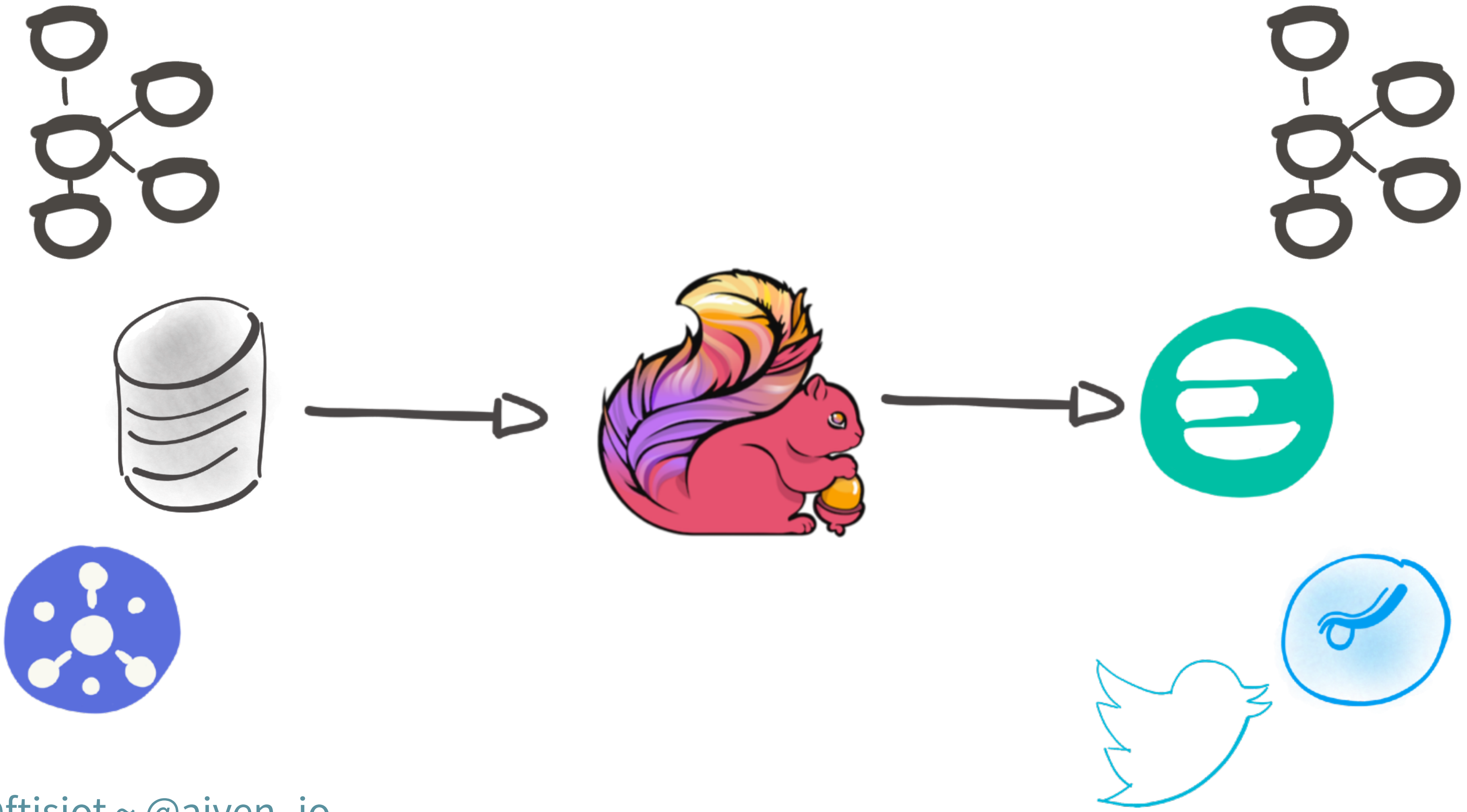
Join

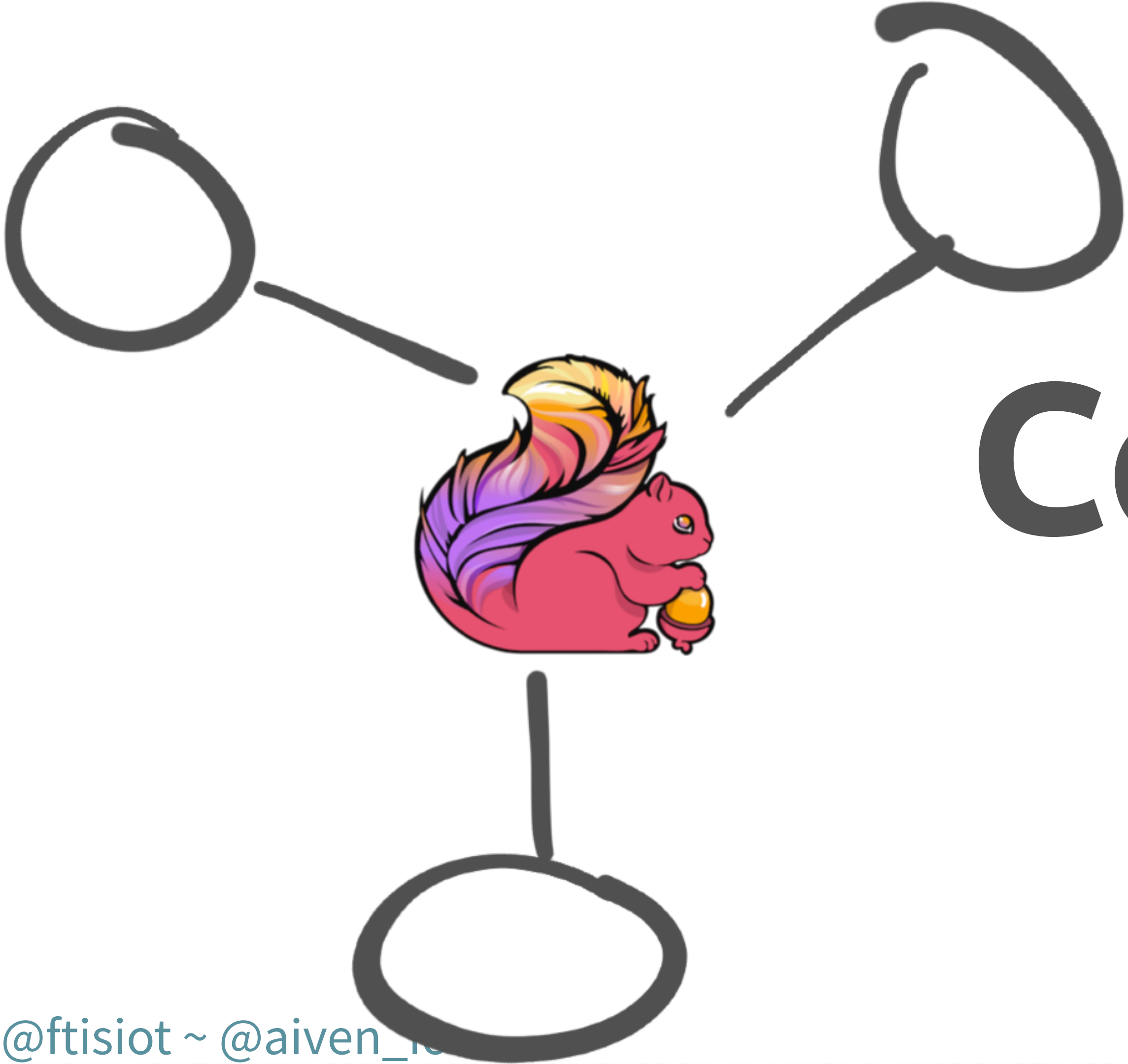
Aggregate

Explode

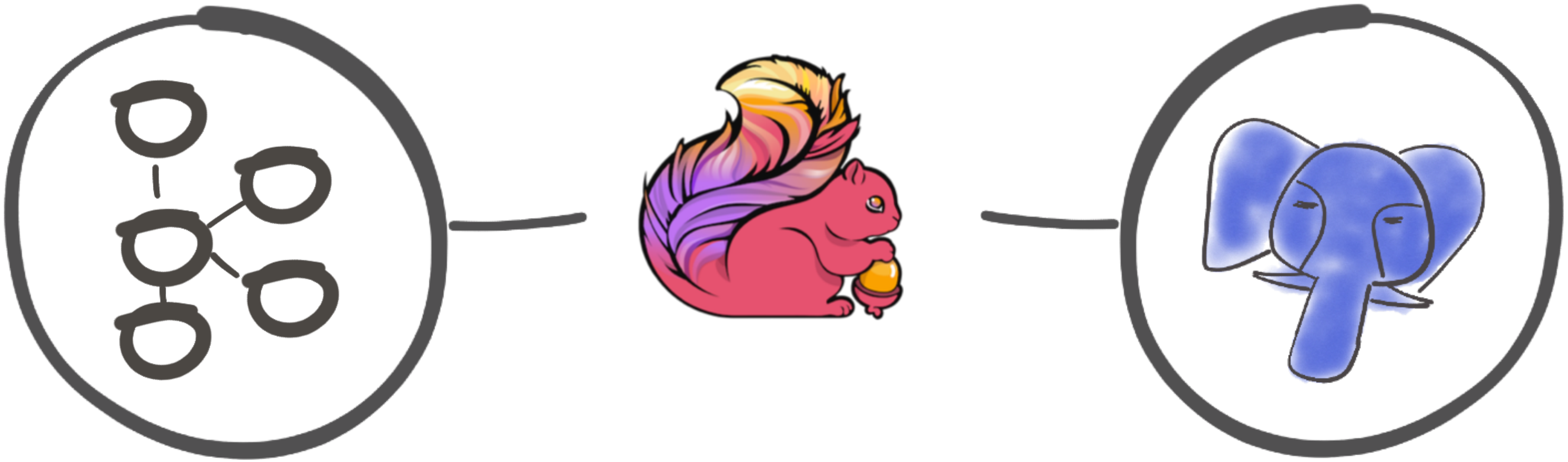
Detect

Change Shape





Connect Flink



```
{
  "id": 1,
  "shop": “Mario's Pizza",
  "name": "Arsenio Pisaroni-Boccaccio",
  "phoneNumber": "+39 51 0290746",
  "address": "Via Ugo 01, Montegrotto, 85639 Padova(PD)",
  "pizzas": [
    {
      "pizzaName": "Margherita",
      "additionalToppings": ["ham"]
    }]
}
```

pizza_name	base_price
Marinara	4
Diavola	6
Mari & Monti	8
Salami	7
Peperoni	8
Margherita	5

Kafka Source

```
CREATE TABLE pizza_orders (
 id INT,
 shop VARCHAR,
 name VARCHAR,
 phoneNumber VARCHAR,
 address VARCHAR,
 pizzas ARRAY
        <ROW (
            pizzaName VARCHAR,
            additionalToppings ARRAY <VARCHAR>)>
) WITH (
   'connector' = 'kafka',
   'properties.bootstrap.servers' = ‘kafka:13041',
   'topic' = 'pizza-orders',
   'scan.startup.mode' = 'earliest-offset',
   …
);
```

Pg Source

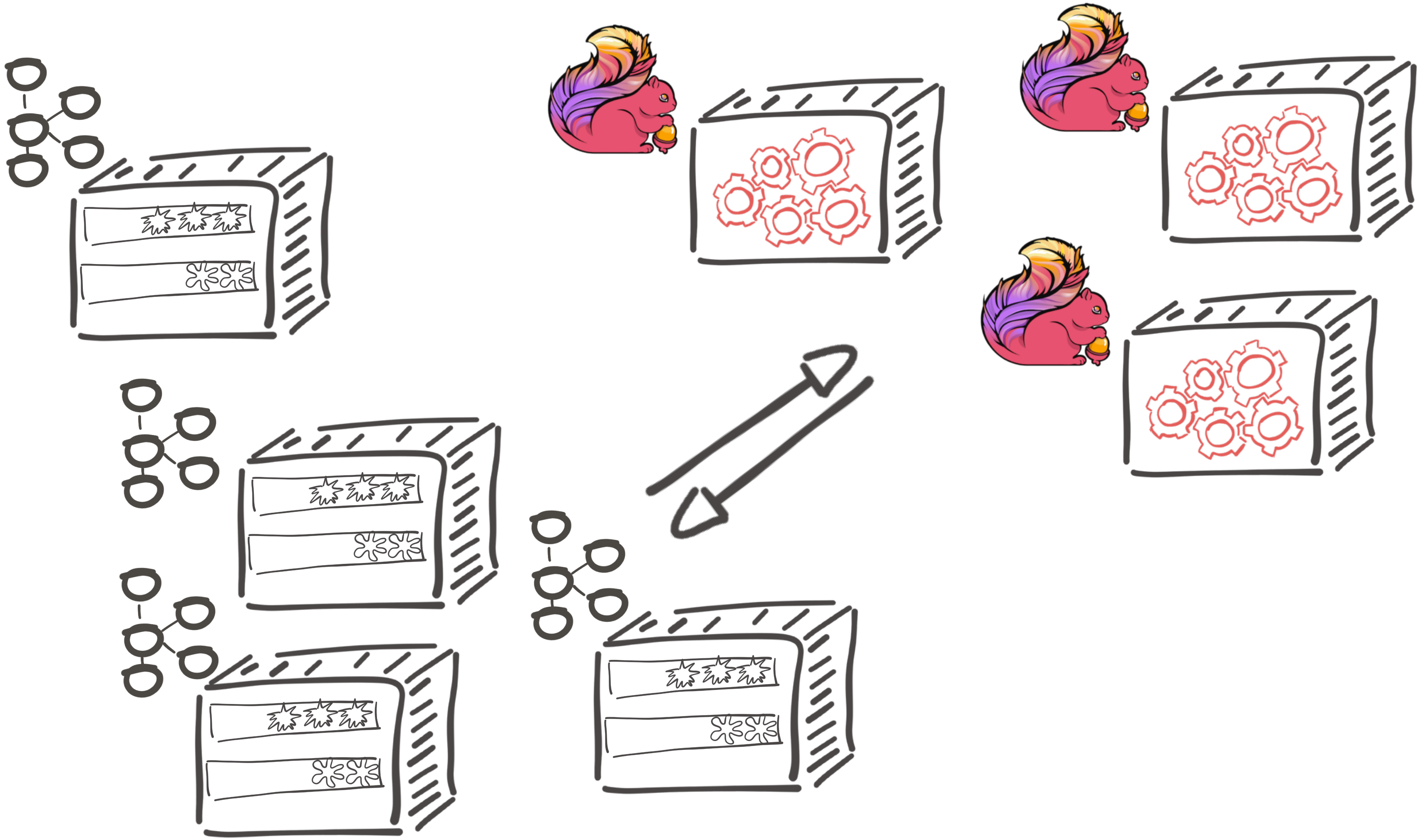
```
CREATE TEMPORARY TABLE pizza_prices (
 pizza_name VARCHAR,
 base_price INT,
 PRIMARY KEY (pizza_name) NOT ENFORCED
) WITH (
   'connector' = 'jdbc',
   'url' = ‘jdbc:postgresql:/pghost:13039/db',
   'username'='avnadmin',
   'password'='verysecurepassword123',
   'table-name' = 'pizza_price'
);
```

Pg Tgt

```
CREATE TABLE order_price (
 id INT,
 pizza_name VARCHAR,
 base_price INT,
 nr_pizzas BIGINT NOT NULL,
 PRIMARY KEY (id, pizza_name) NOT ENFORCED
) WITH (
   'connector' = 'jdbc',
   'url' = ‘jdbc:postgresql://pghost:13039/db',
   'username'='avnadmin',
   'password'='verysecurepassword123',
   'table-name' = 'order_price'
);
```

Create Pipeline

```
insert into order_price
select id,
    b.pizzaName,
    base_price,
    count(*) nr_pizzas
from pizza_orders cross join UNNEST(pizzas) b
LEFT OUTER JOIN pizza_prices
     FOR SYSTEM_TIME AS OF orderProctime AS pp
     ON b.pizzaName = pp.pizza_name
group by id,
    b.pizzaName,
    base_price;
```



Resources

https://kafka.apache.org/

http://flink.apache.org/

https://github.com/aiven/flink-sql-cli-docker

https://aiven.io/blog/build-a-streaming-sql-pipeline-with-flink-and-kafka

https://aiven.io