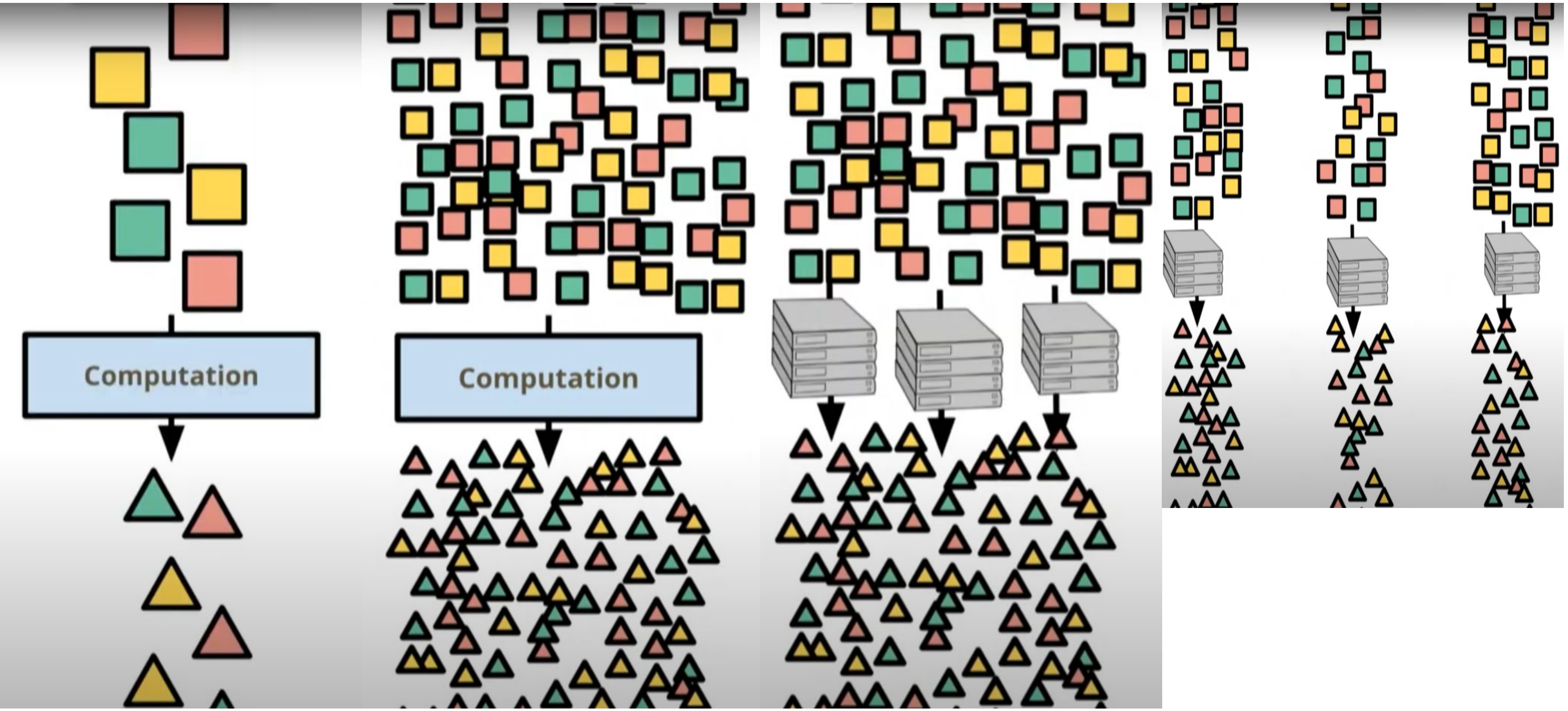
Apache Beam

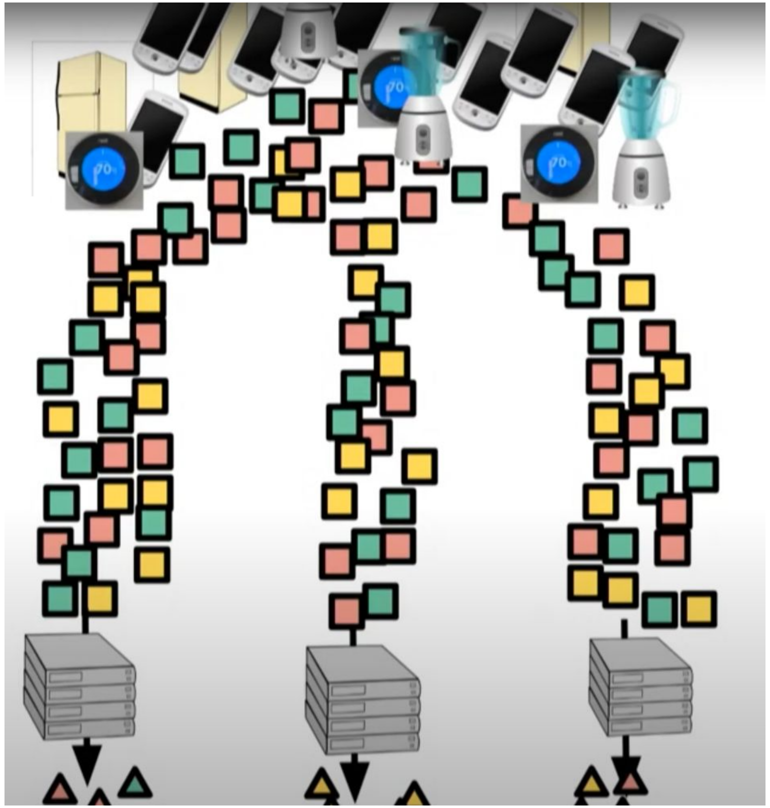
Agenda

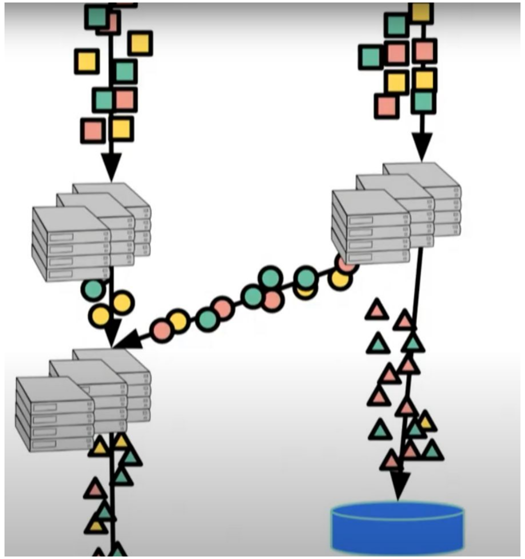
1. Overview of Apache Beam
2. Crash Course
3. Demo

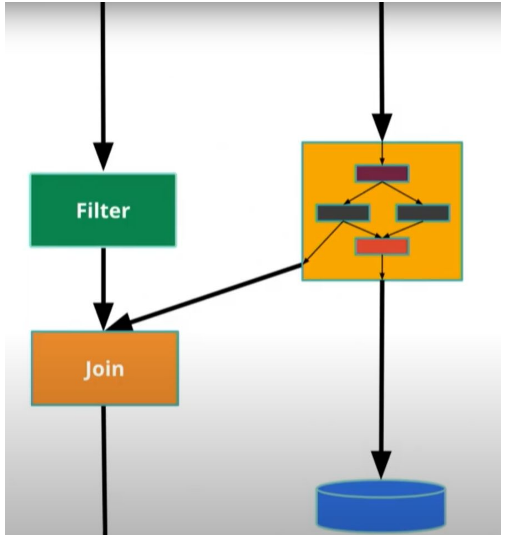
Scenarios



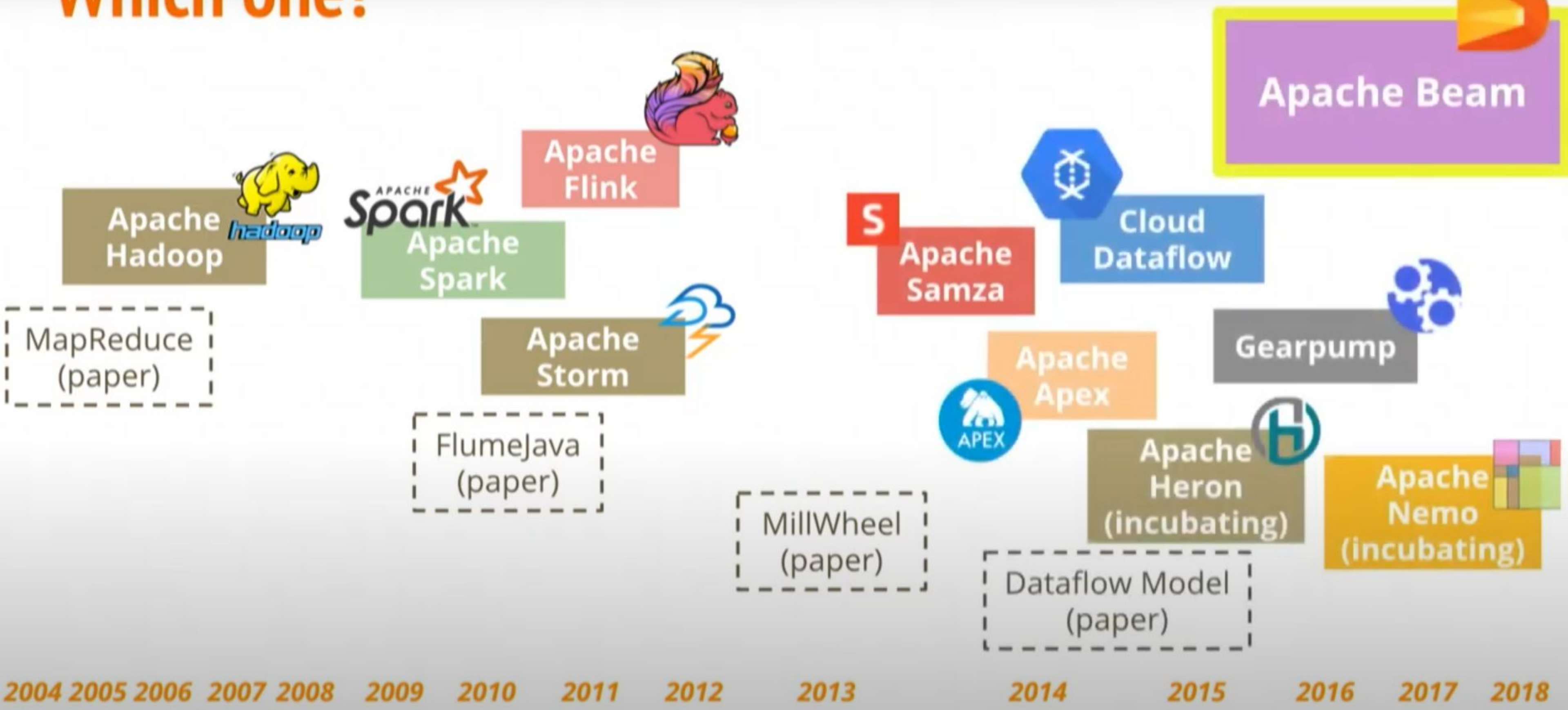
Scenarios





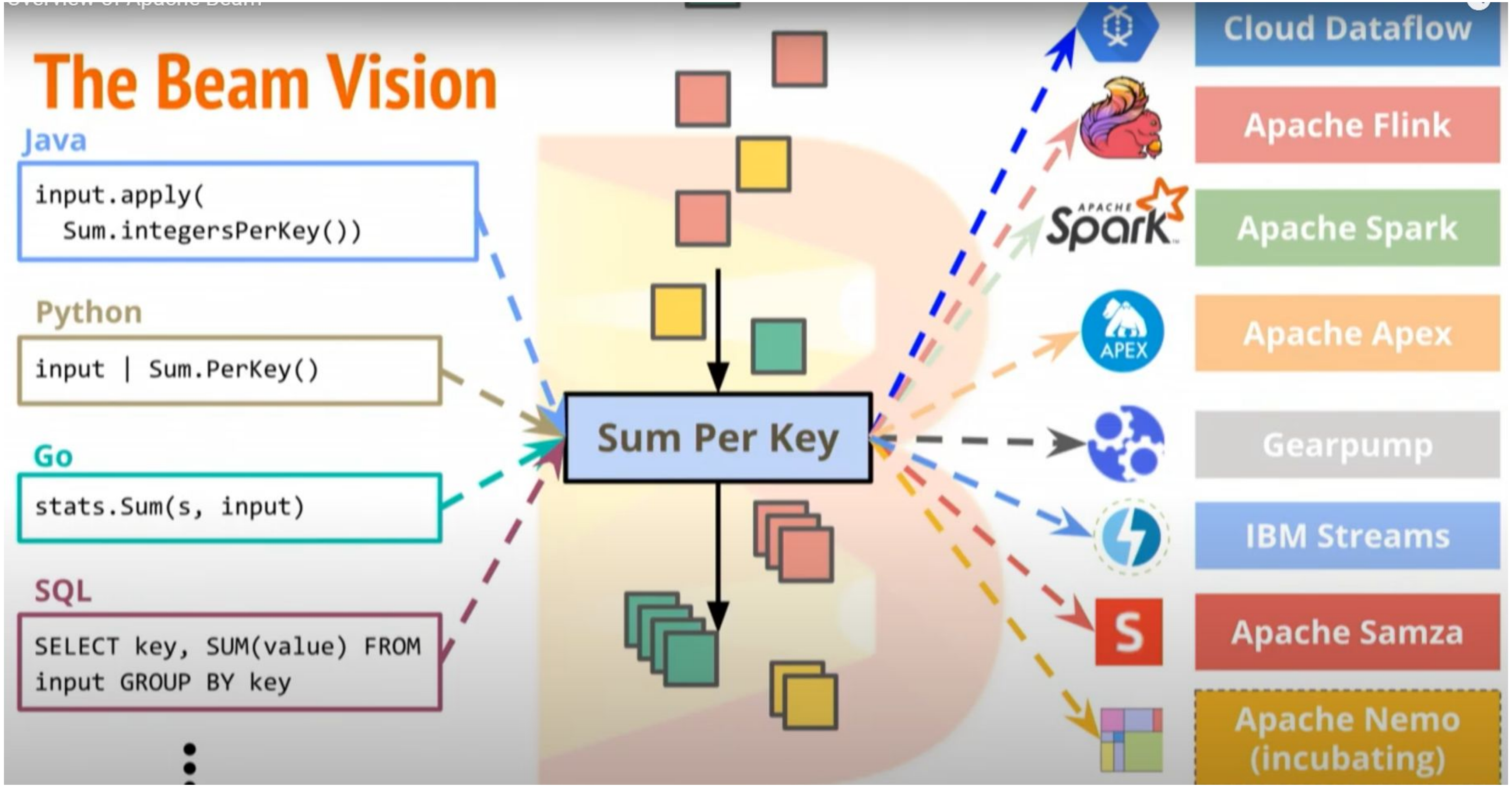


Which one?



Apache Beam
Apache Hadoop
hadoop
Apache Spark
Spark
Apache Flink
Apache Storm
MapReduce (paper)
FlumeJava (paper)
MillWheel (paper)
S
Apache Samza
Cloud Dataflow
Apache Apex
APEX
Gearpump
Apache Heron (incubating)
Apache Nemo (incubating)
Dataflow Model (paper)
2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017 2018

The Beam Vision



Java

```
input.apply(
  Sum.integersPerKey())
```

Python

```
input | Sum.PerKey()
```

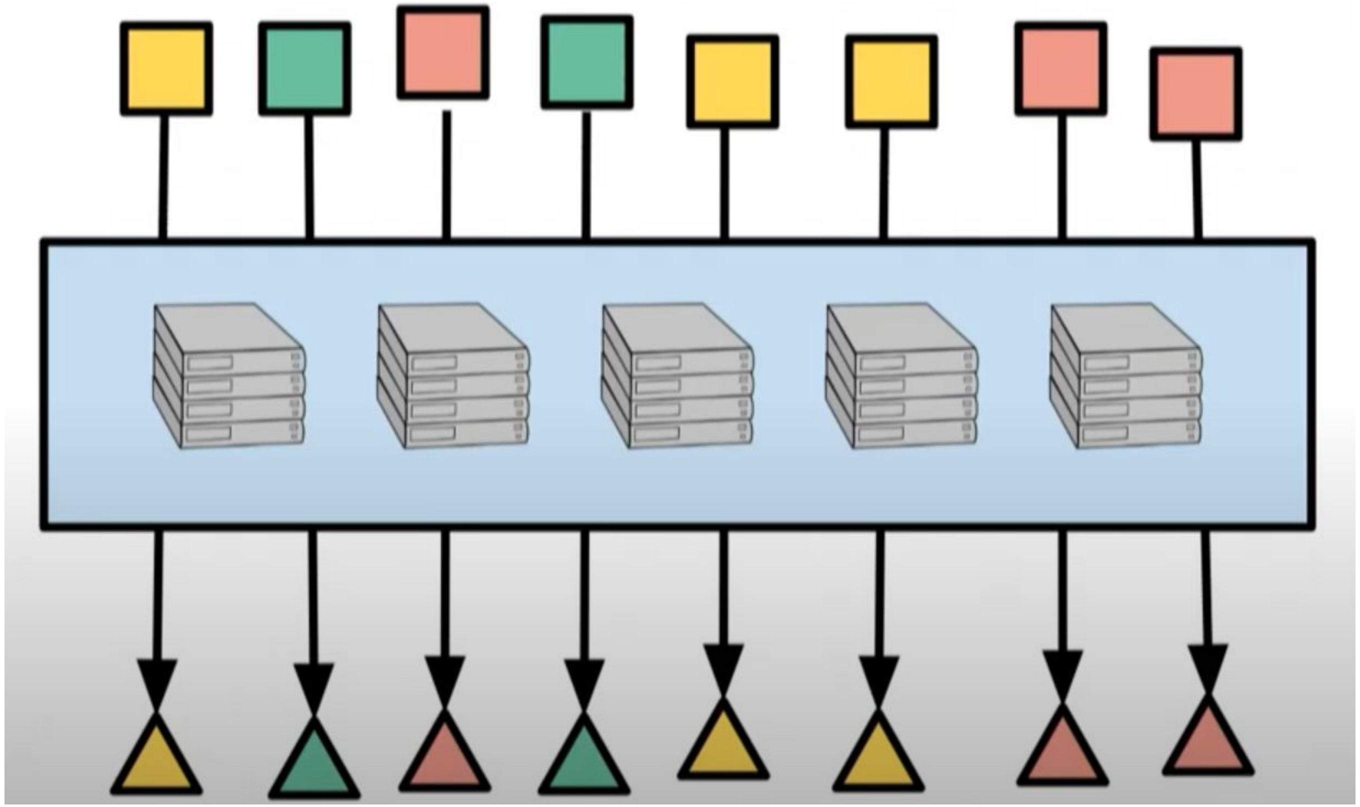
Go

```
stats.Sum(s, input)
```

SQL

```
SELECT key, SUM(value) FROM
input GROUP BY key
```

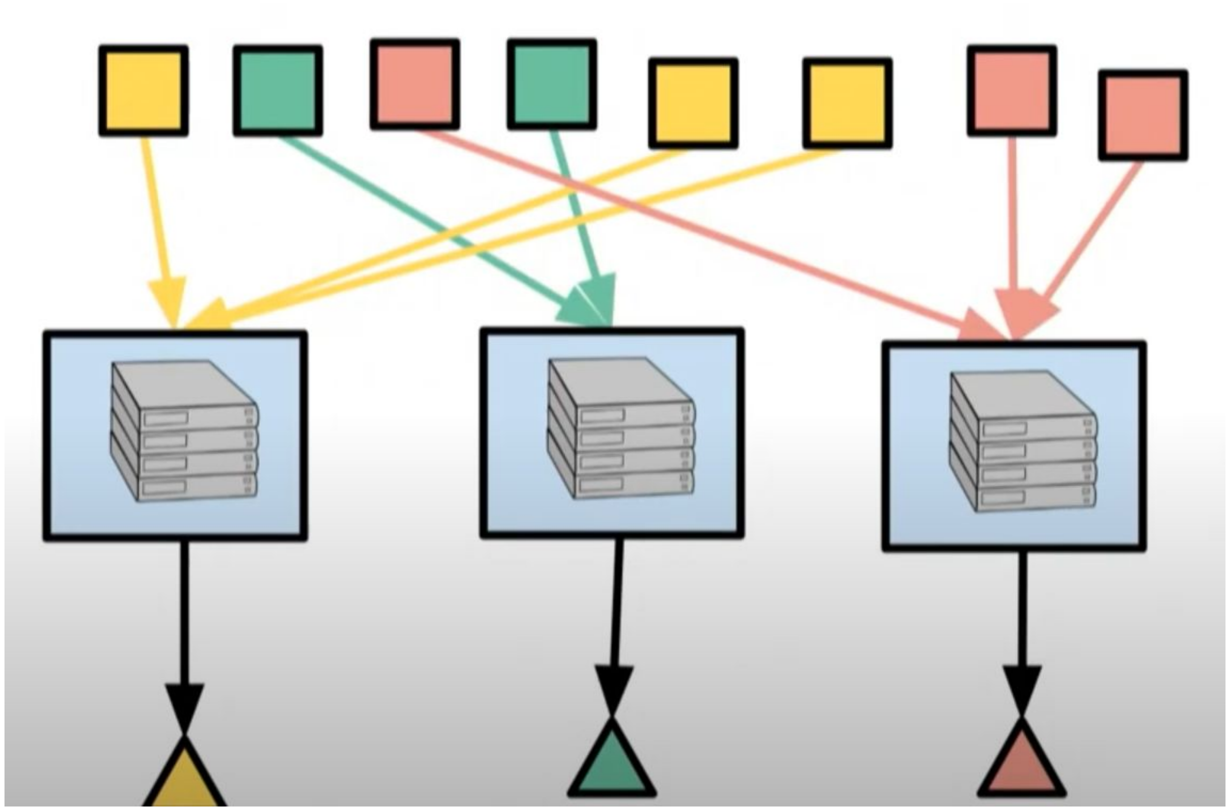
Crash Beam - Per element ParDo(Map, etc)



Every Item processed independently.

Stateless implementation.

Crash Beam - Per key Combine(Reduce, etc)

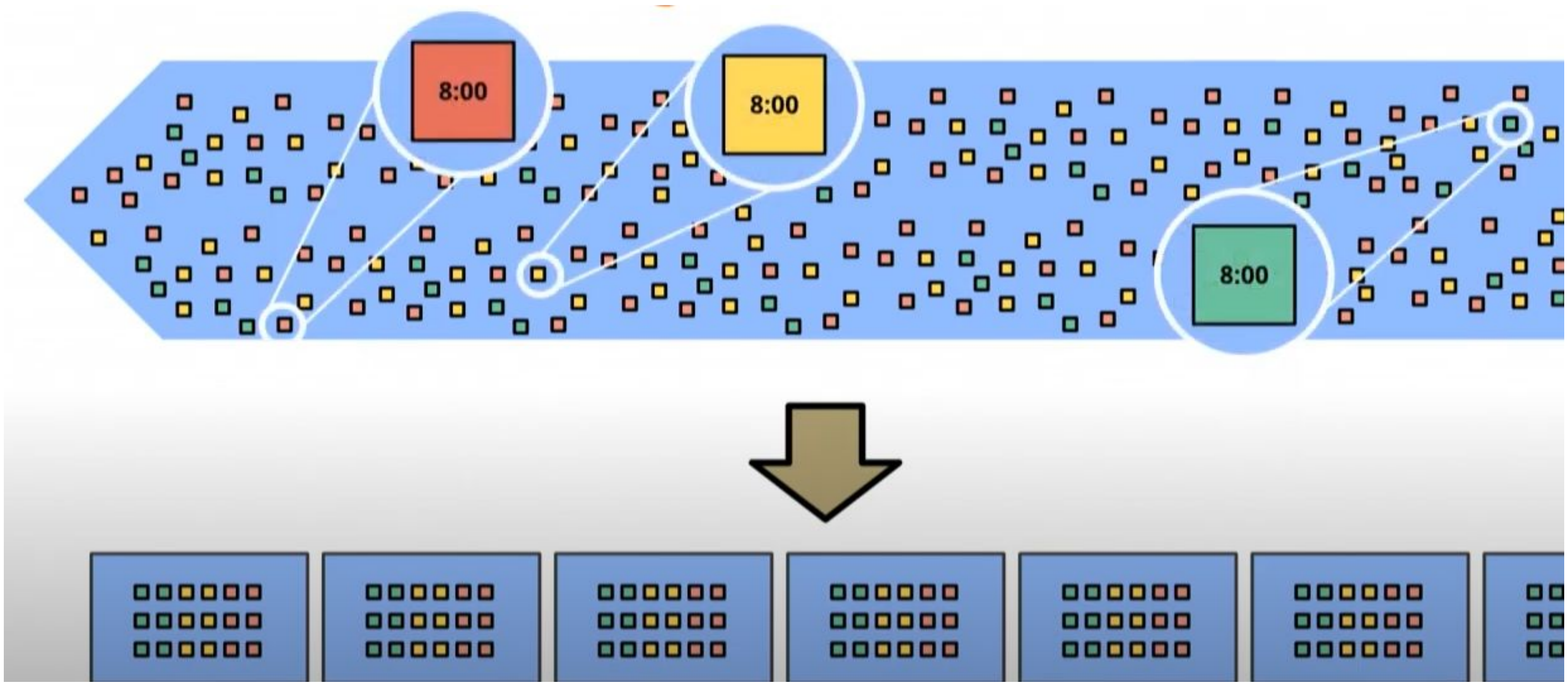


Items grouped by some key and combined

Stateful streaming implementation

But your code doesn’t work with state, just associative & commutative function

Crash Beam - Event Time Windowing



Crash Summary

1. ParDo: per element processing
2. Combine / GroupByKey: aggregation
3. Event time windowing

Terms

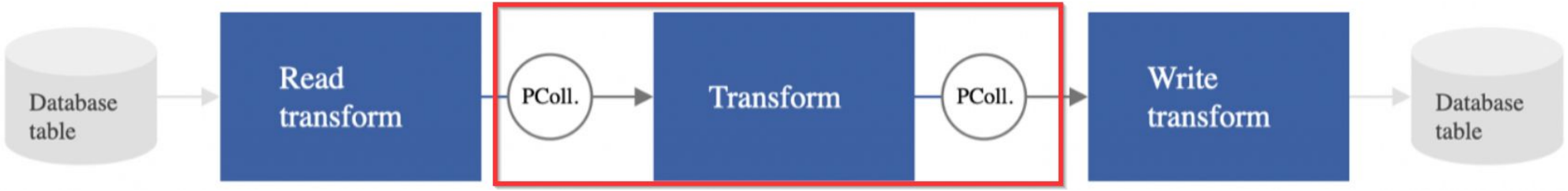
- **PCollection** — represents a data set which can be a fixed batch or a stream data.
- **PTransform** — a data processing operation that takes one or more PCollections and outputs zero or more PCollections.
- **Pipeline** — represent a directed acyclic graph of PCollection and Transform, and hence encapsulates the entire data processing job.
- **I/O Transforms** — PTransforms that read or write data.

Demo

Beam currently supports:

Direct runner, Apache Flink runner, Apache Spark runner, Google Cloud Data Flow runner, AWS Kinesis, Apache Nemo runner, Apache Samza runner, Hazlecast Jet runner and Twister2 runner.

A simple linear pipeline with 3 sequential transforms



References

Apache Beam is a typical Kappa Architecture: https://www.peerislands.io/data-processing-lambda-vs-kappa-architectures-and-apache-beam/