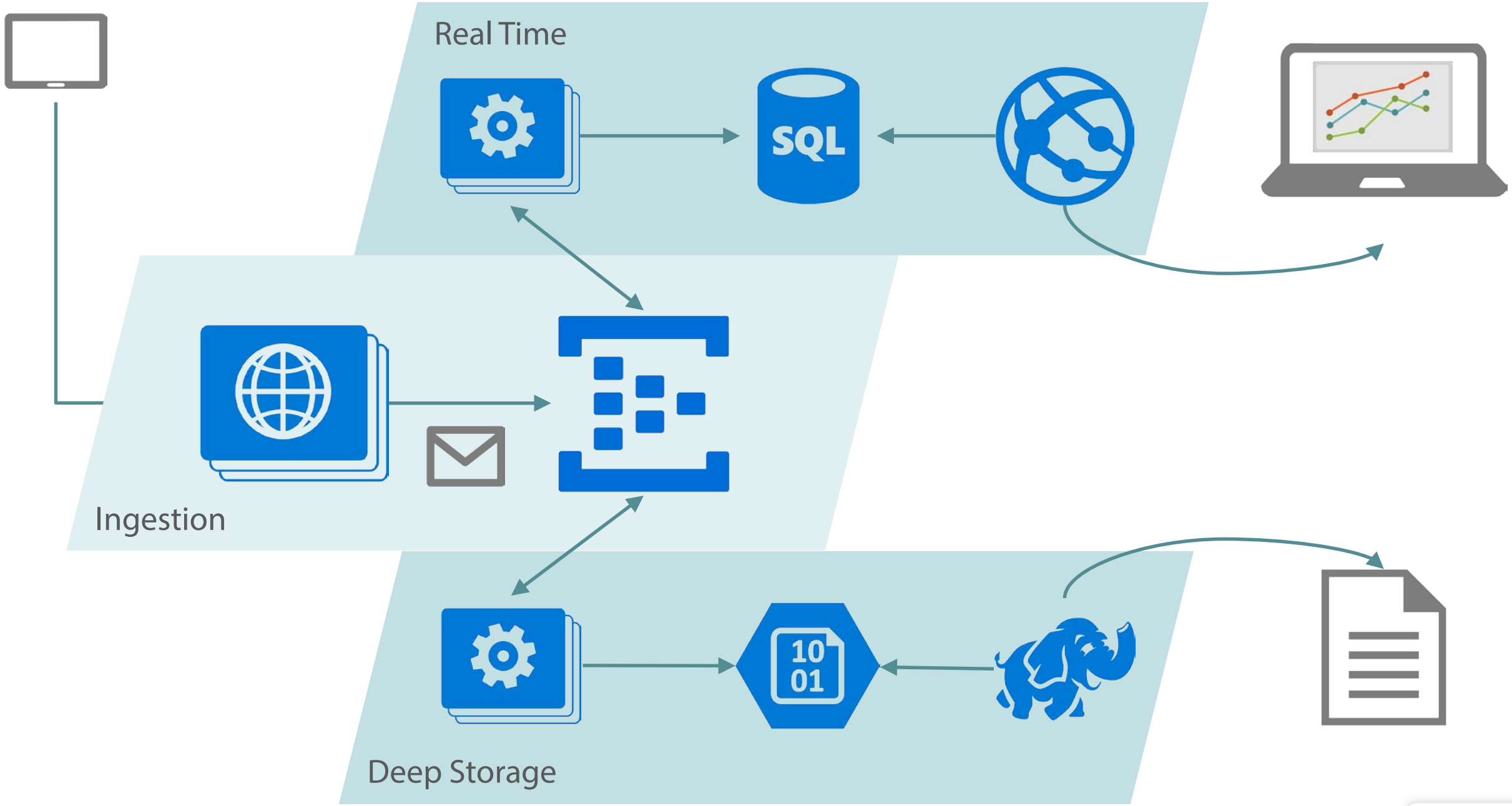
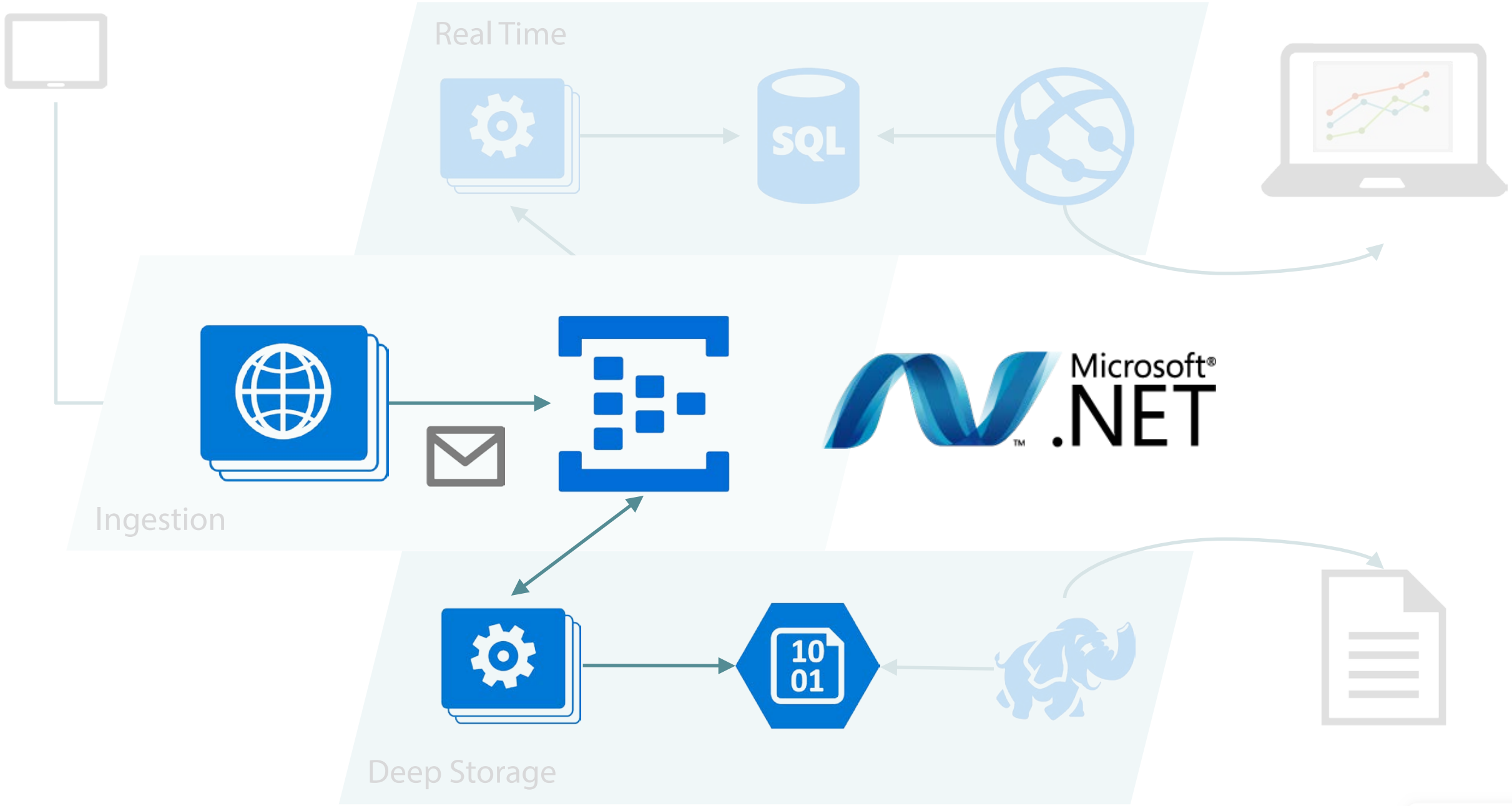
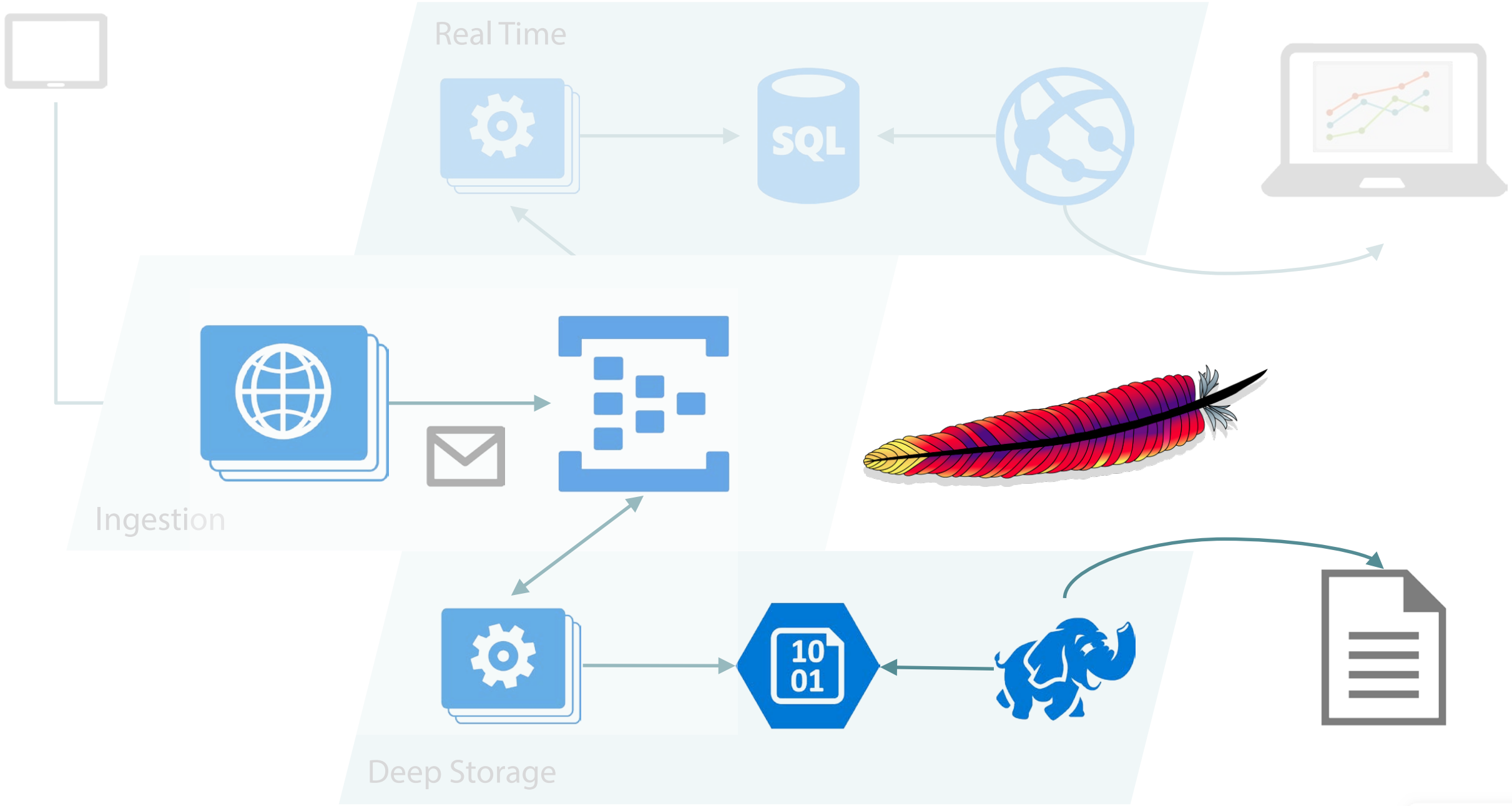
Querying Batch Data in Deep Storage

Elton Stoneman

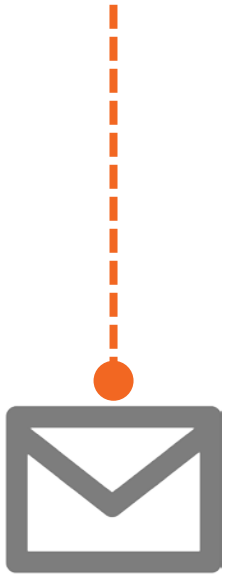
@EltonStoneman | www.geekswithblogs.net/eltonstoneman




Real Time
SQL
Ingestion
Microsoft®
.NET
10
01
Deep Storage



x BILLIONs



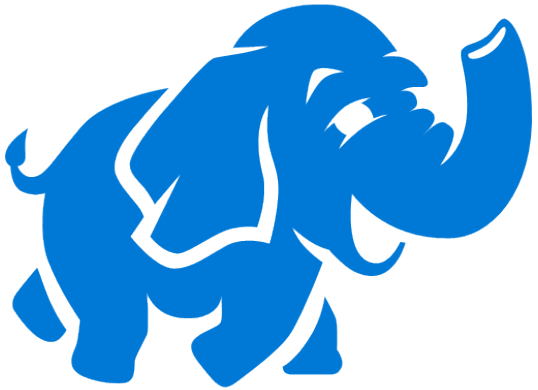
p1/2015040100.json.gz

p1/2015040101.json.gz

...

p15/2015043122.json.gz

p15/2015043123.json.gz



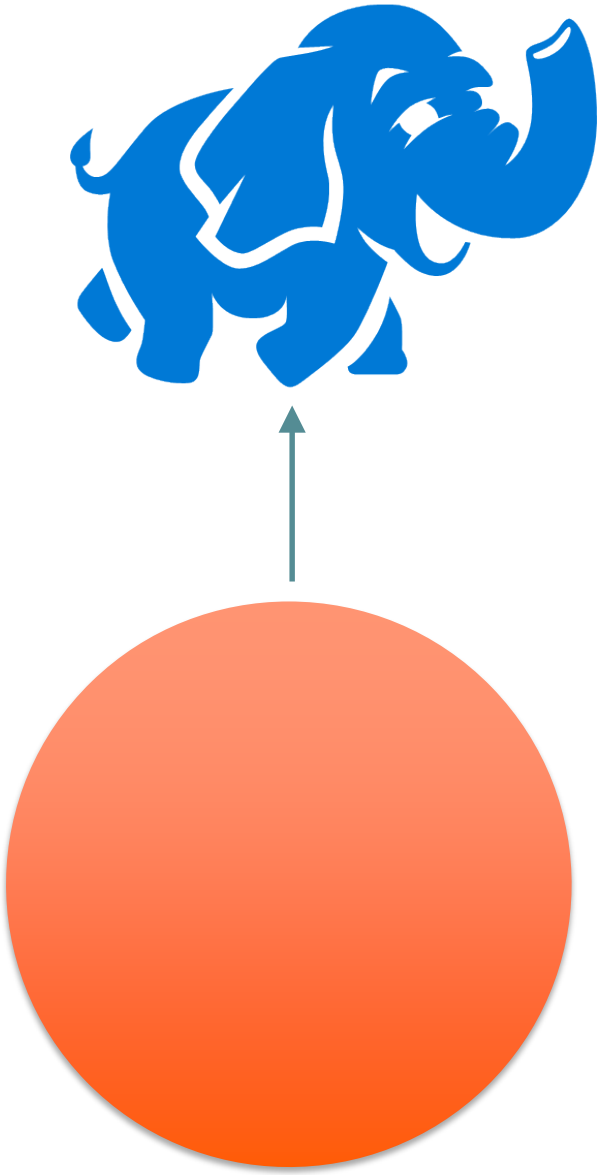
p1/2015040100.json.gz

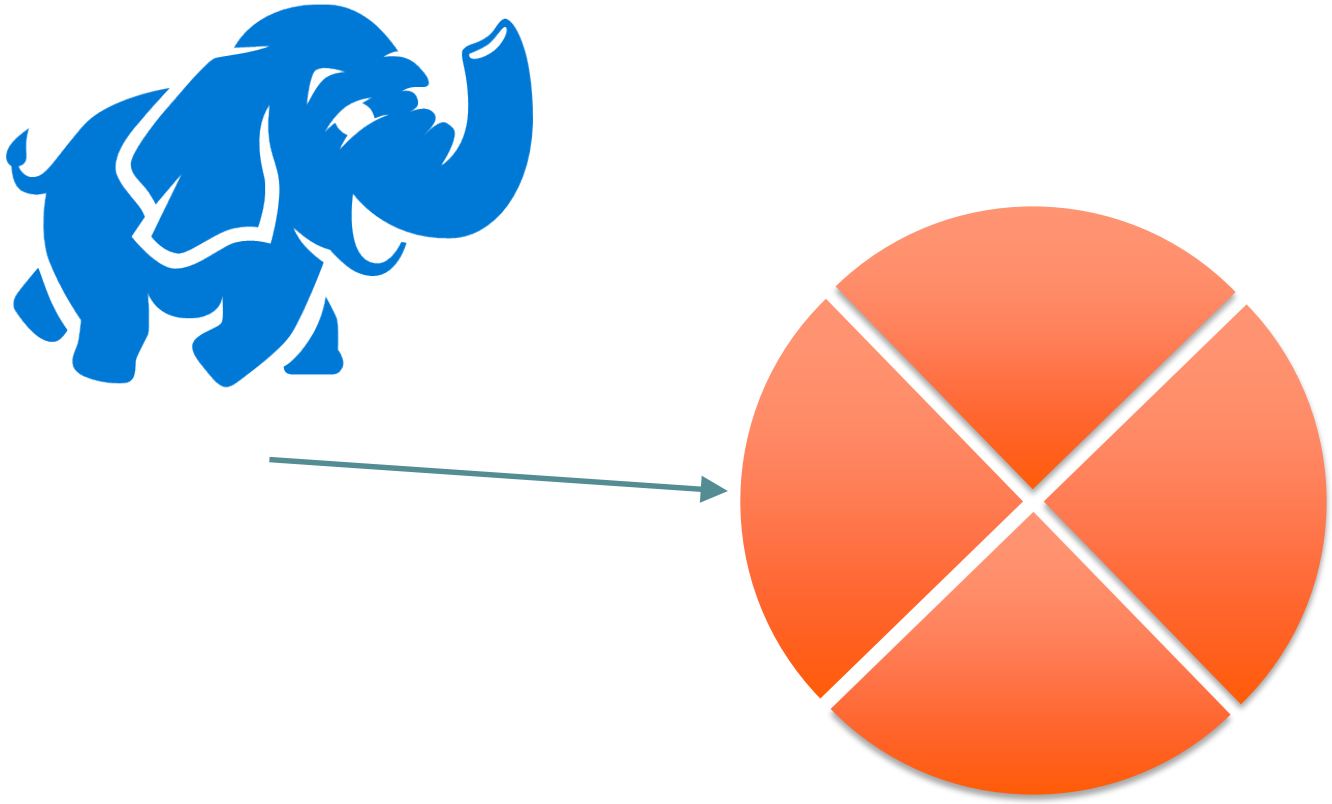
p1/2015040101.json.gz

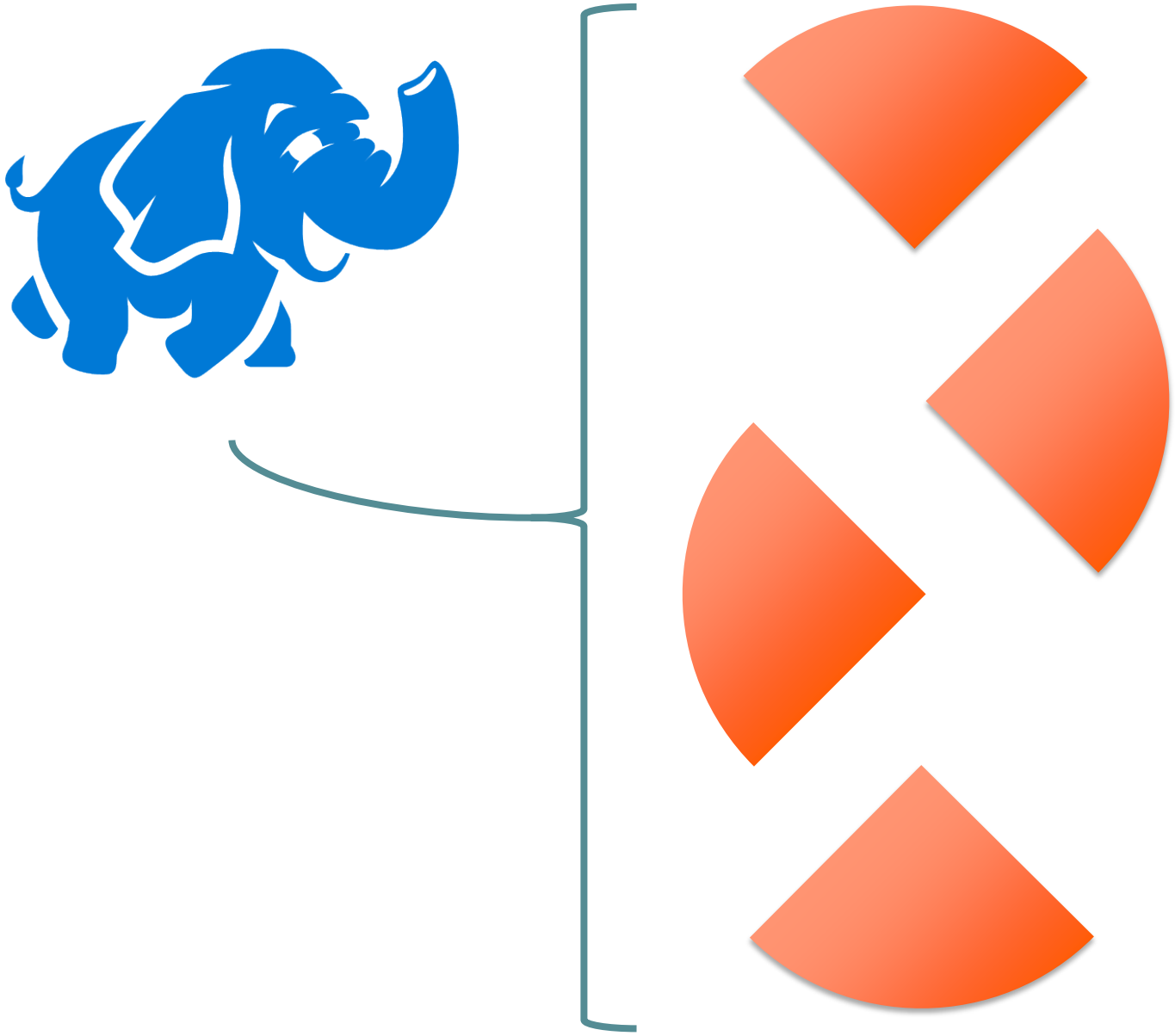
...

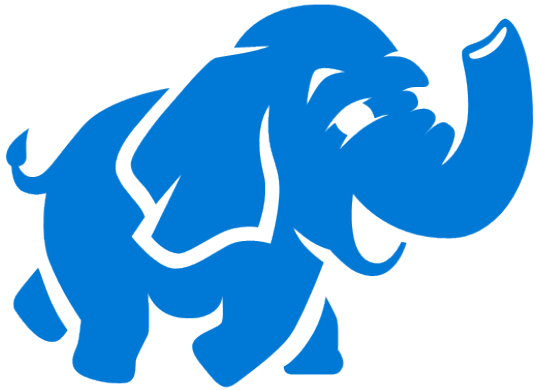
p15/2015043122.json.gz

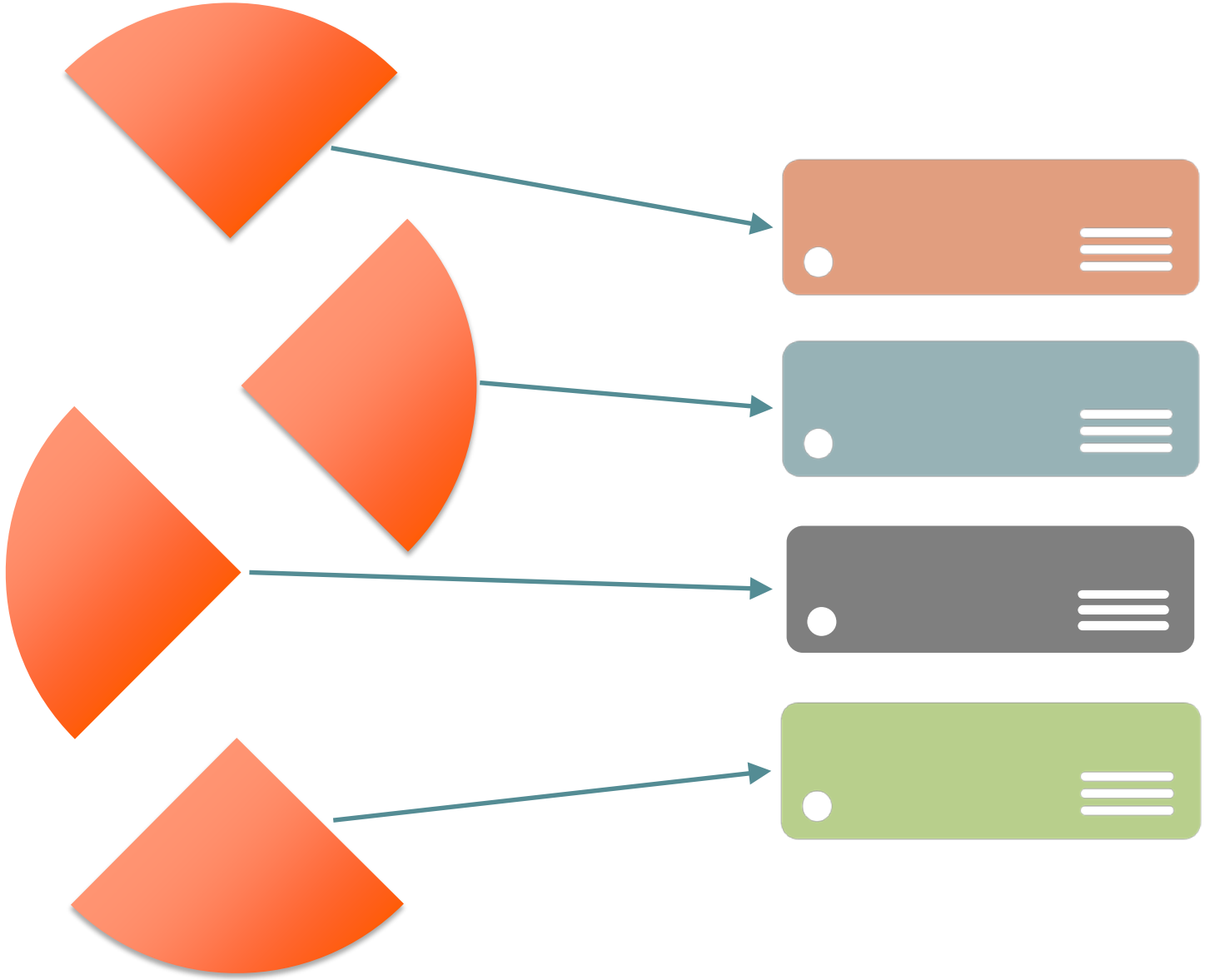
p15/2015043123.json.gz

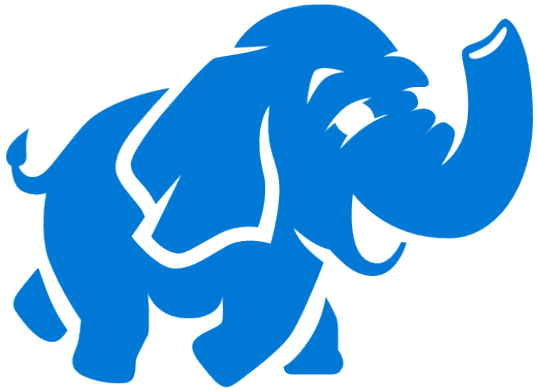


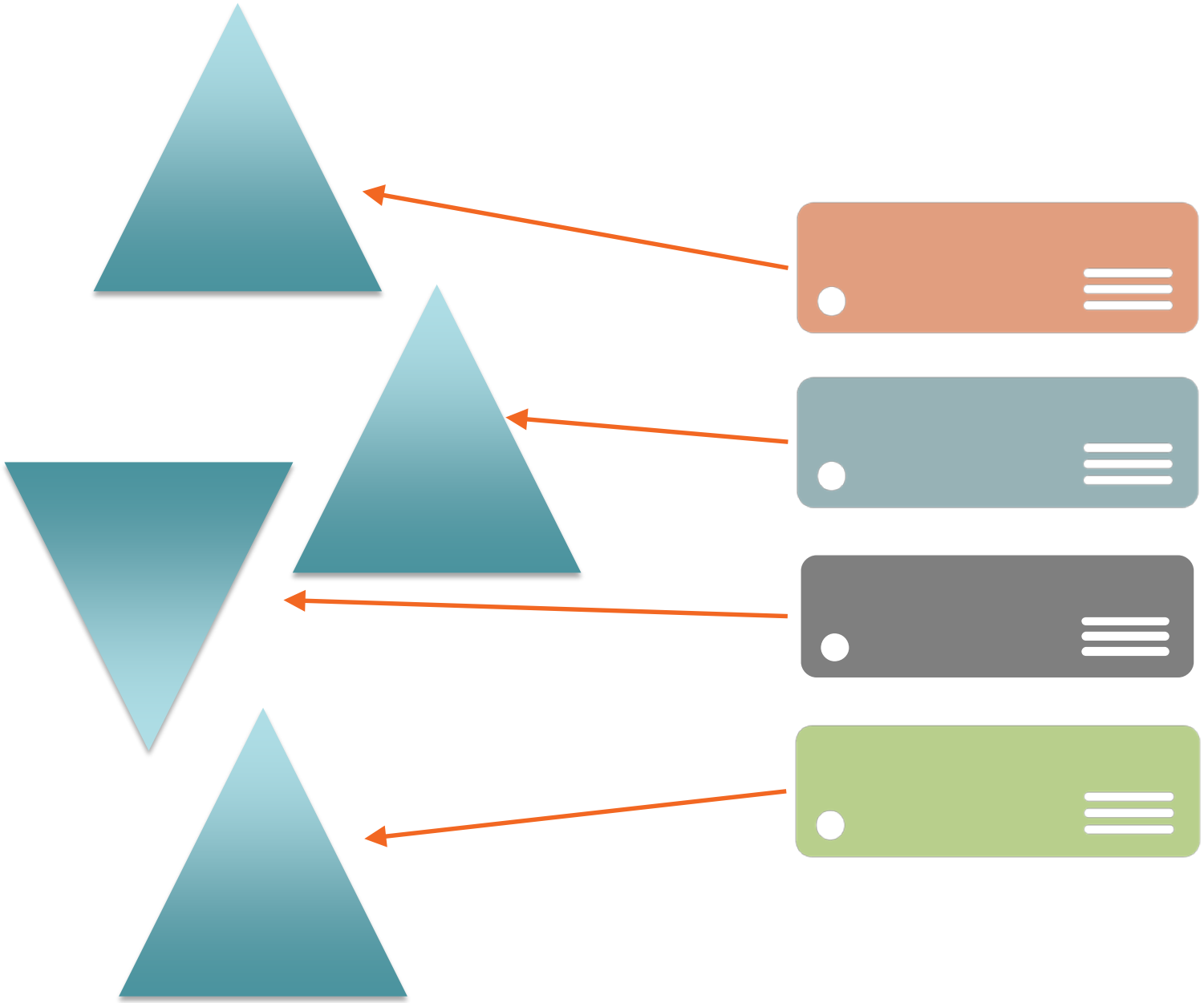


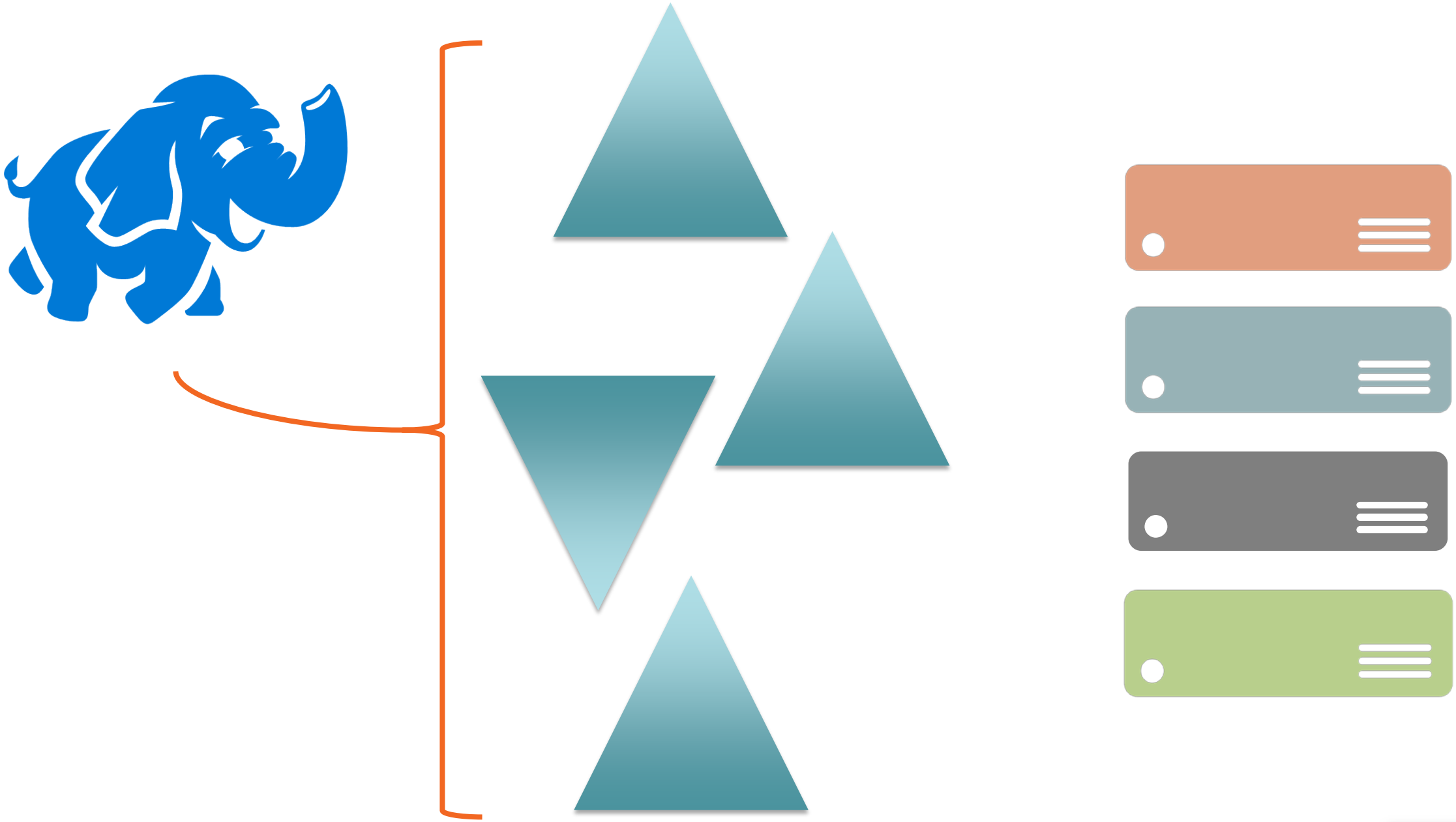


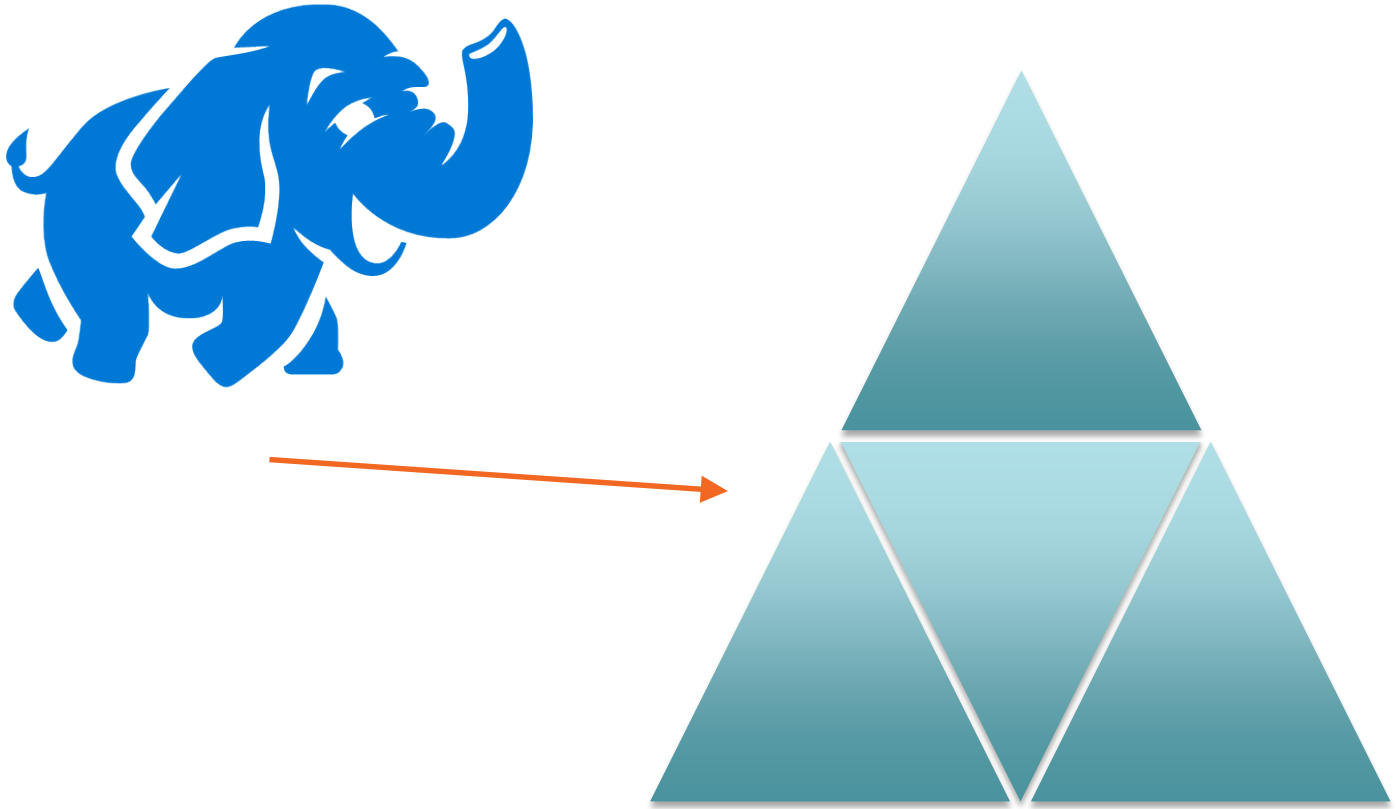


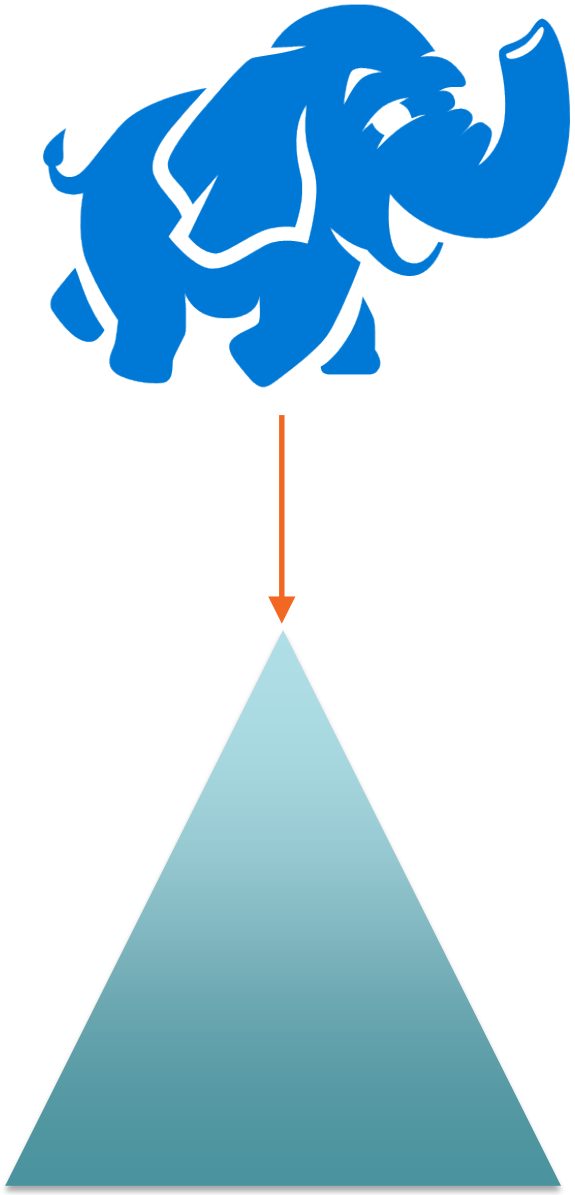


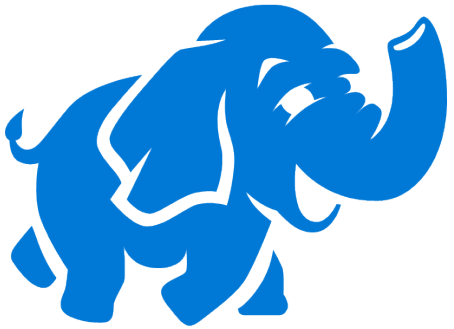










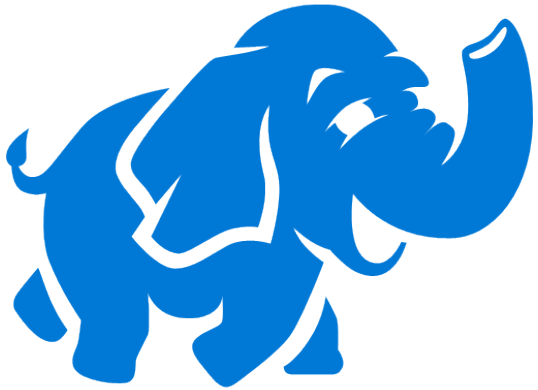


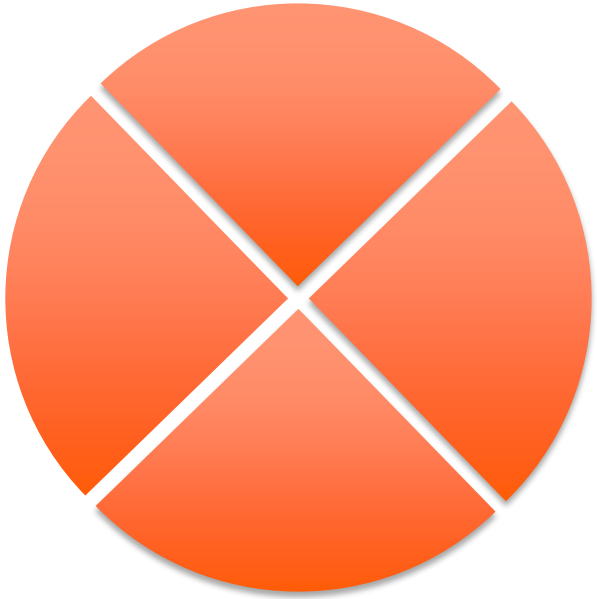
Azure HDInsight

Managed Hadoop platform

Backed by Blob Storage

Clusters from 4-32+ nodes

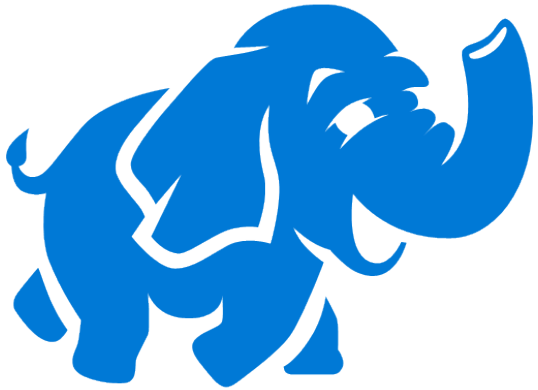


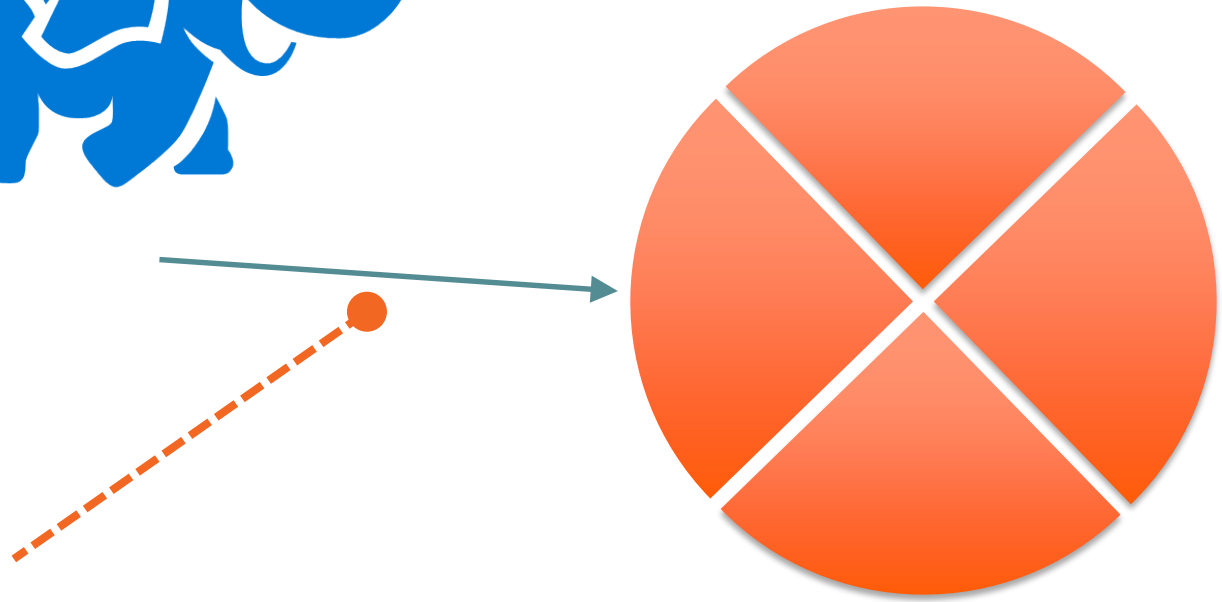


Java

Clojure

Pig





Pig

```
A = LOAD 'c:/device-events/2015/04/08/*';
B = GROUP A all;
C = FOREACH B GENERATE COUNT(A);
DUMP C;
```

```
A = LOAD 'c:/device-events/2015/04/15/p3/2015041507.json.gz';
B = GROUP A all;
C = FOREACH B GENERATE COUNT(A);
DUMP C;
```

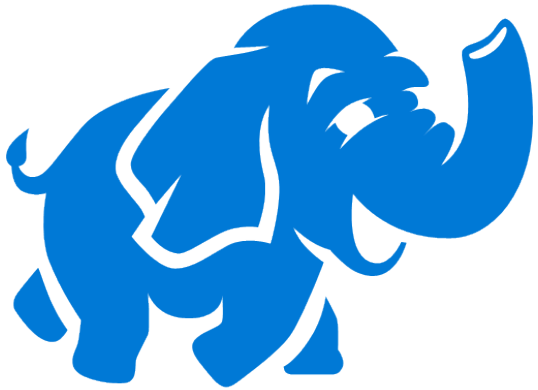
Pig Latin

Load, evaluate, output

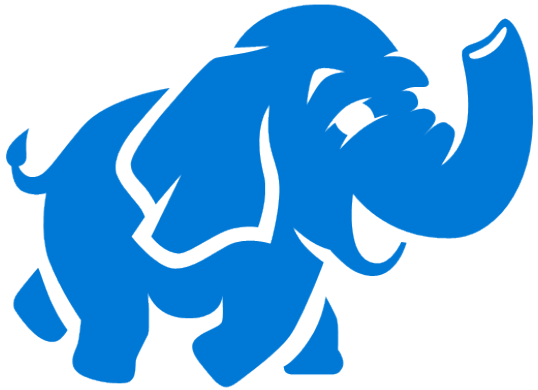
```
A = LOAD 'c:/device-events/2015/04/08/*';
B = GROUP A all;
C = FOREACH B GENERATE COUNT(A);
DUMP C;
```

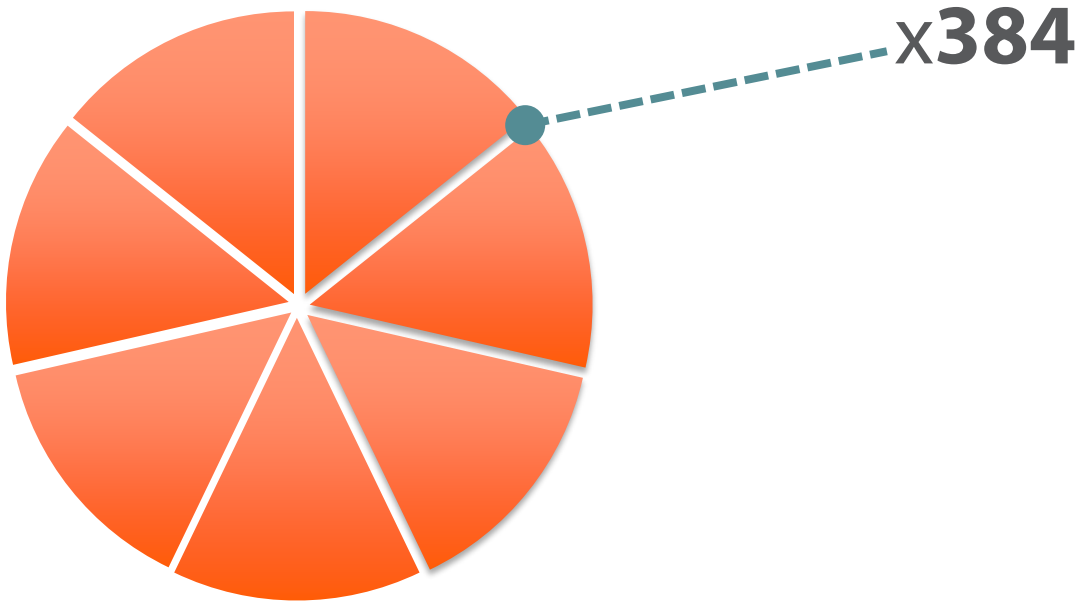
Multiple sources

Load with wildcard

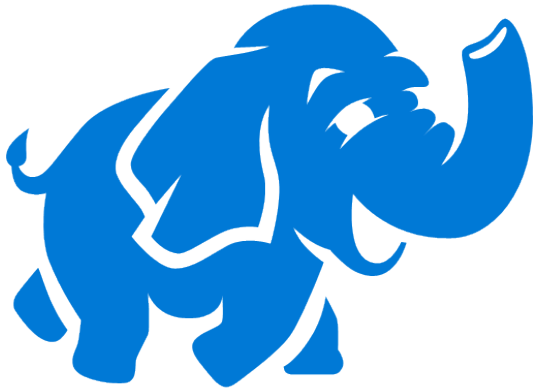


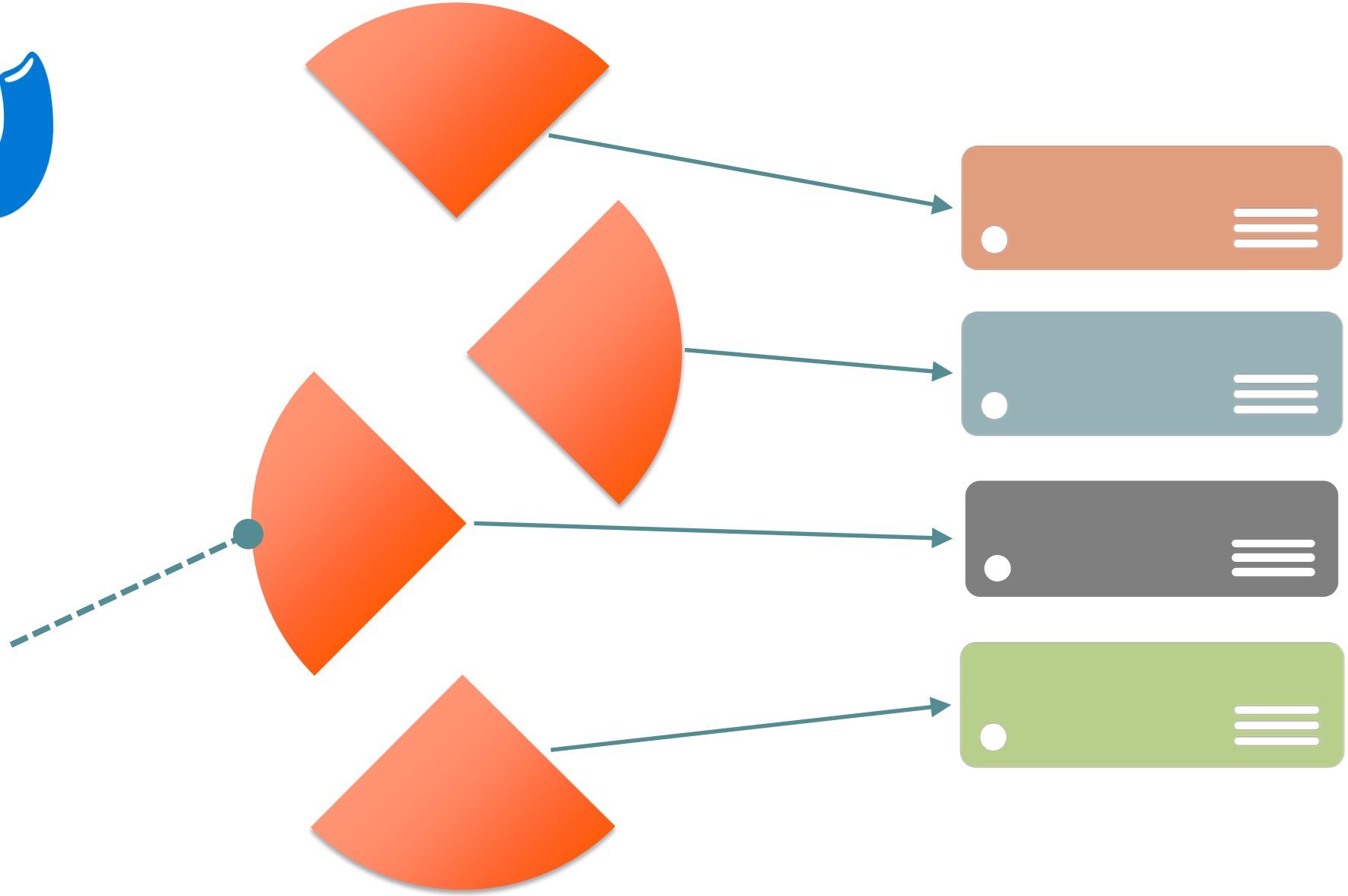
```
A = LOAD 'c:/device-events/2015/04/08/*';
B = GROUP A all;
C = FOREACH B GENERATE COUNT(A);
DUMP C;
```





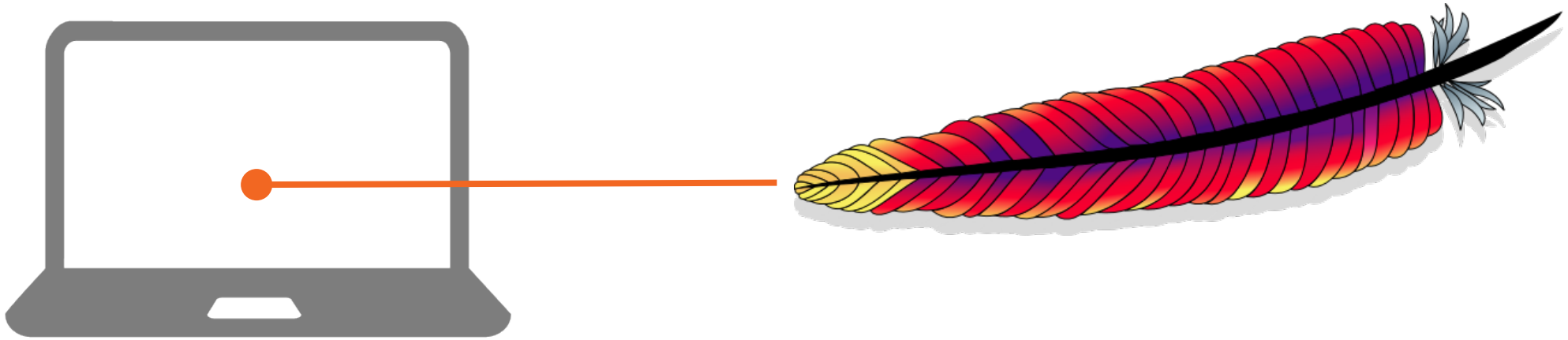

```
A = LOAD 'c:/device-events/2015/04/08/*';
B = GROUP A all;
C = FOREACH B GENERATE COUNT(A);
DUMP C;
```

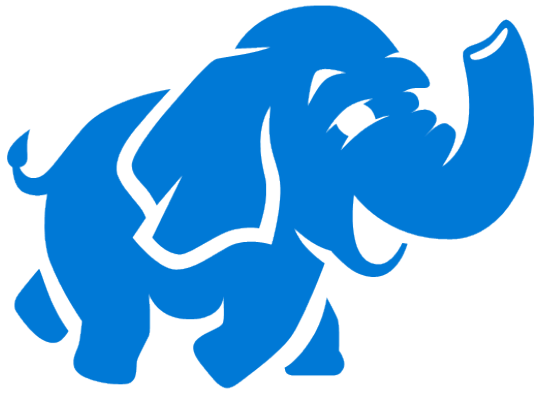




x**64** per node

>**24** concurrent





```
> pig -i
Apache Pig version 0.12.1.2.1.3.0.1981 (r: unknown)
```

Demo: AzCopy and Pig

Download blobs with AzCopy

Run Pig in local, interactive mode

Query event count

```
AzCopy
/Source:https://x.blob.core.windows.net/container/2015/04/08/
/SourceKey:LA7fhZFf.../fZEC8WdDTTXPaYTg==
/Dest:c:\device-events\2015\04\08
/S
```

AzCopy

Copy from Azure Blob Storage to local machine

```
AzCopy
/Source:https://prod.blob.core.windows.net/container/
/SourceKey:LA7fhZFf.../fZEC8WdDTTXPaYTg==
/Dest:https://test.blob.core.windows.net/container/
/DestKey:fZEC8Wd.../DTTXPaYTgLA7fhZFf==
/S
```

AzCopy

Copy between Azure Blob Storage accounts

```
pig -x local
grunt> A = LOAD 'c:/device-events/2015/04/08/*';
```

Pig in local, interactive mode

Load from local filesystem

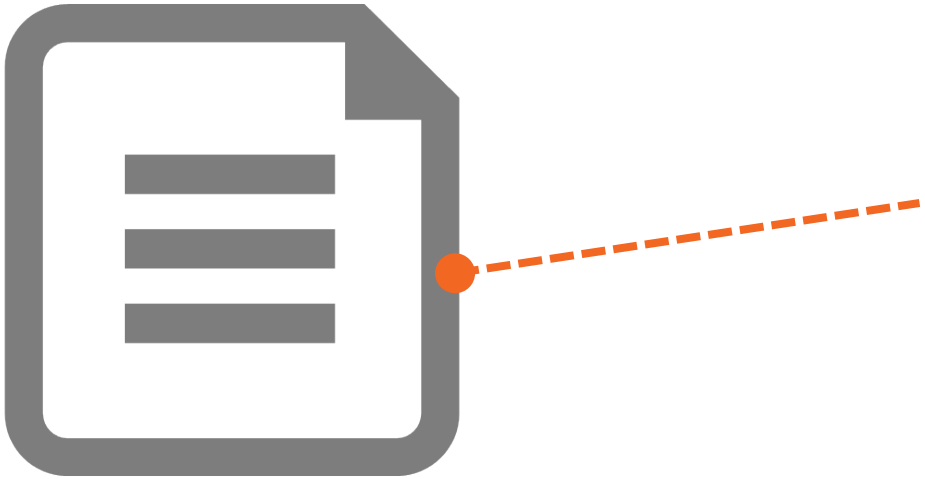
```
grunt> B = GROUP A all;
grunt> C = FOREACH B GENERATE COUNT(A);
grunt> DUMP C;
```

Count all rows

Group, foreach and count

```
C = FILTER B BY eventName == 'event.1';
D = GROUP B BY deviceId;
```

eventName	deviceId	timestamp	receivedAt	period



CSV

TSV

JSON

eventName	deviceId	timestamp	receivedAt	period

```
register './lib/elephant-bird-pig-4.6.jar';
...
A = LOAD 'file.json.gz'
    USING com.twitter.elephantbird.pig.load.JsonLoader()
    AS (json:map[]);
```

Register external libraries

Access User Defined Functions (UDFs)

```
B = FOREACH A GENERATE
    json#'eventName' AS eventName,
    json#'deviceId' AS deviceId,
    json#'timestamp' AS timestamp;
```

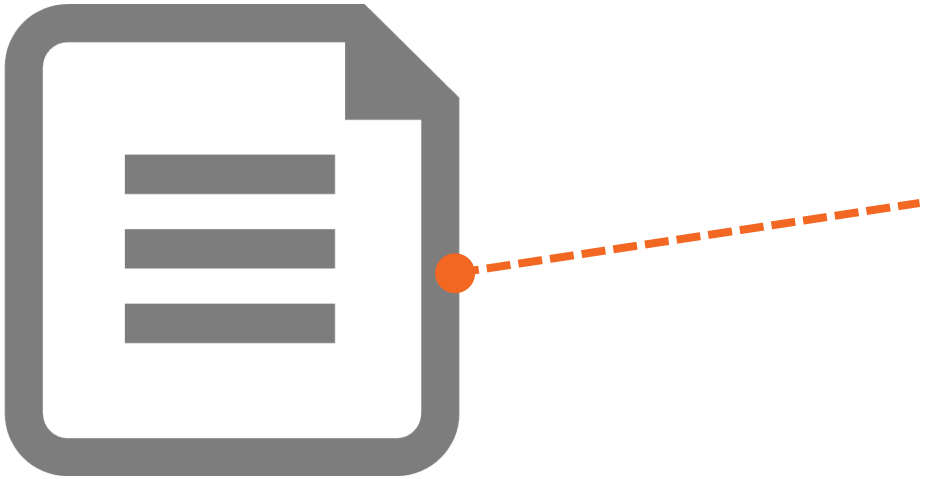
Define the schema

Generate a typed relation

```
C = FILTER B BY eventName == 'device.log'
               OR eventName == 'system.log';
```

Query fields

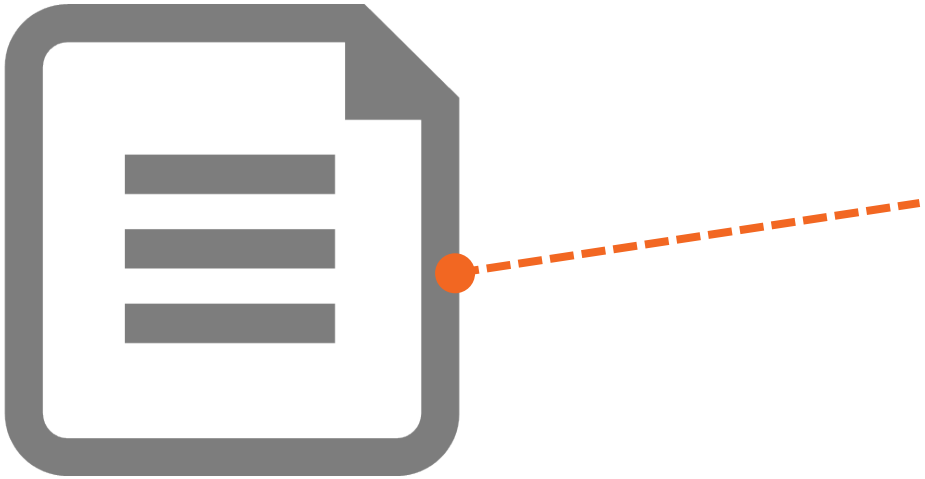
Operate or evaluate on field values



JSON

Variable schema

eventName	deviceId	timestamp	message	severity
device.log	abc	21246174	Clock sync failed	W
system.log	def	21127468	Divide by zero	E
gps.enabled	ghi	21023423	<NULL>	<NULL>
user.created	jkl	21252342	<NULL>	<NULL>



JSON

Variable schema

eventName	deviceId	timestamp	message	severity
device.log	abc	21246174	Clock sync failed	W
system.log	def	21127468	Divide by zero	E
gps.enabled	ghi	21023423	<NULL>	<NULL>
user.created	jkl	21252342	<NULL>	<NULL>

Demo: Pig & JSON

Load with JsonLoader

Generate schema

Group & count by event type

```
register './lib/json-simple-1.1.1.jar';
register './lib/elephant-bird-core-4.6.jar';
register './lib/elephant-bird-pig-4.6.jar';
register './lib/elephant-bird-hadoop-compat-4.6.jar';
register './lib/slf4j-api-1.7.10.jar';
```

Register libraries & dependencies

From Java, Python, JavaScript, Groovy

```
register './lib/elephant-bird-pig-4.6.jar';
register './lib/My.DotNet.Assembly.dll';
```

Maven

pom.xml

Definition

Dependencies

```
<artifactId>elephant-bird-pig</artifactId>
 <name>Elephant Bird Pig</name>
 <description>Pig utilities.</description>

 <dependencies>
   <dependency>
     <groupId>com.twitter.elephantbird</groupId>
     <artifactId>elephant-bird-core</artifactId>
   </dependency>
    ...
```

```
A = LOAD 'c:/device-events/2015/04/08/*'
    USING com.twitter.elephantbird.pig.load.JsonLoader()
    AS (json:map[]);
B = FOREACH A GENERATE json#'eventName' AS eventName;
```

Generate relation with schema

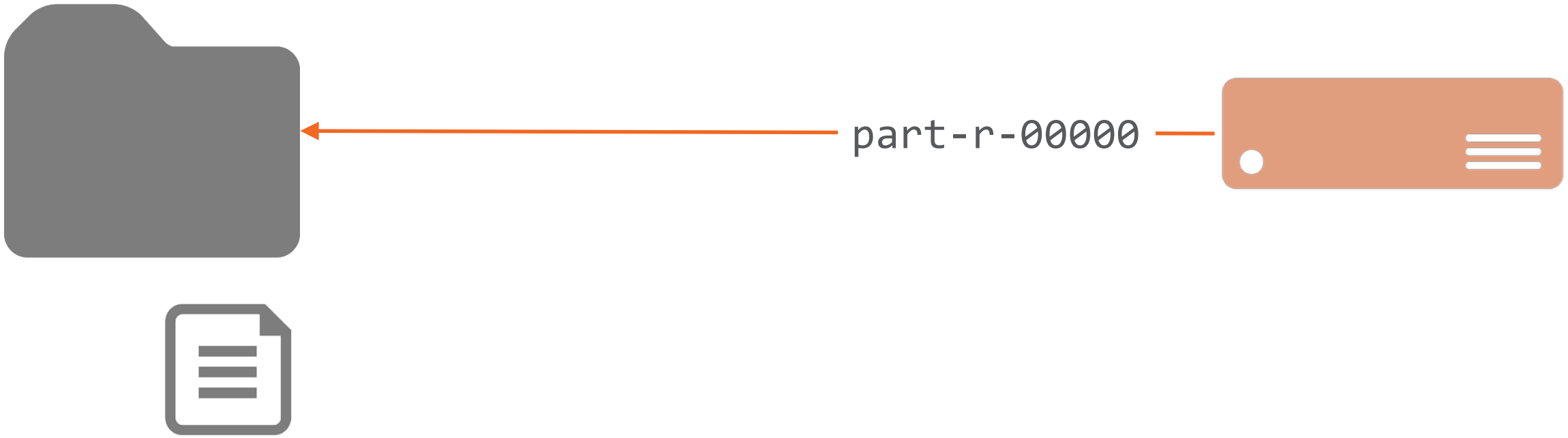
Mapping fields from JSON properties

```
C = GROUP B BY eventName;
D = FOREACH C GENERATE group, COUNT(B);
STORE D INTO 'event-count-20150408.tsv';
```

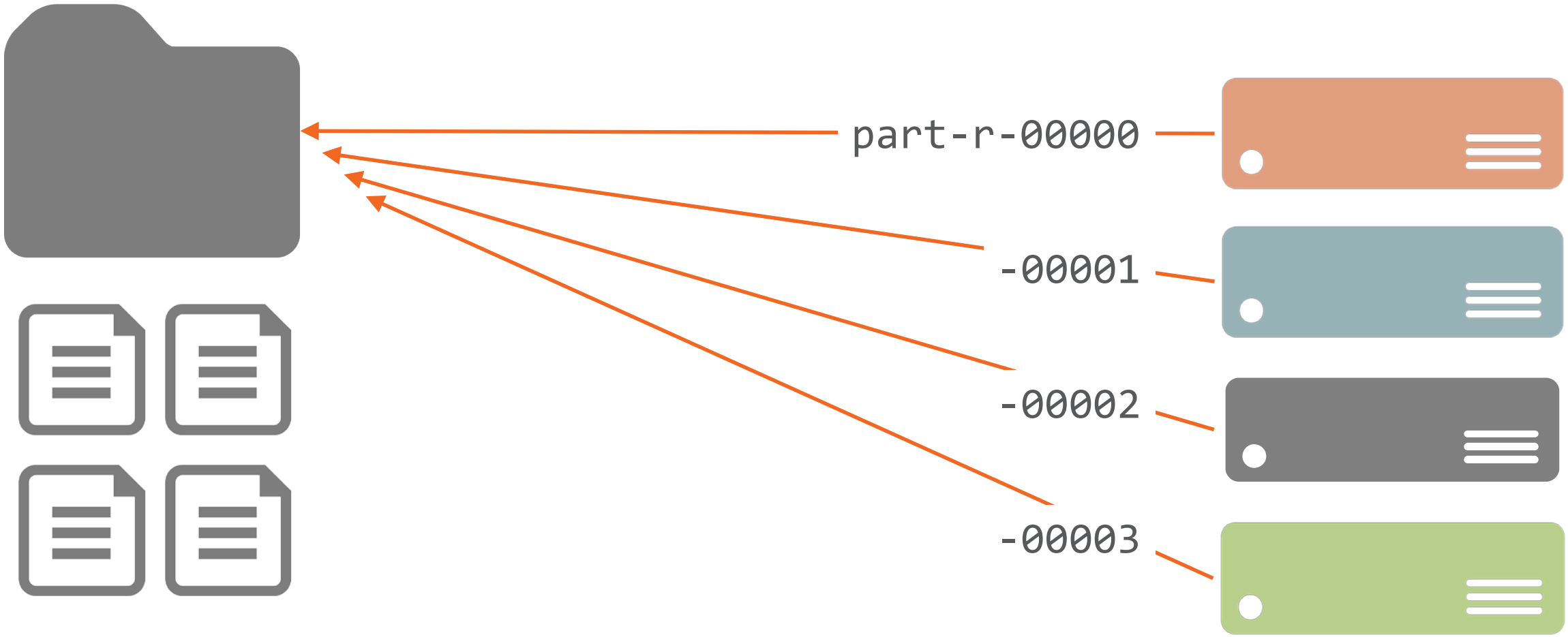
Query & store result

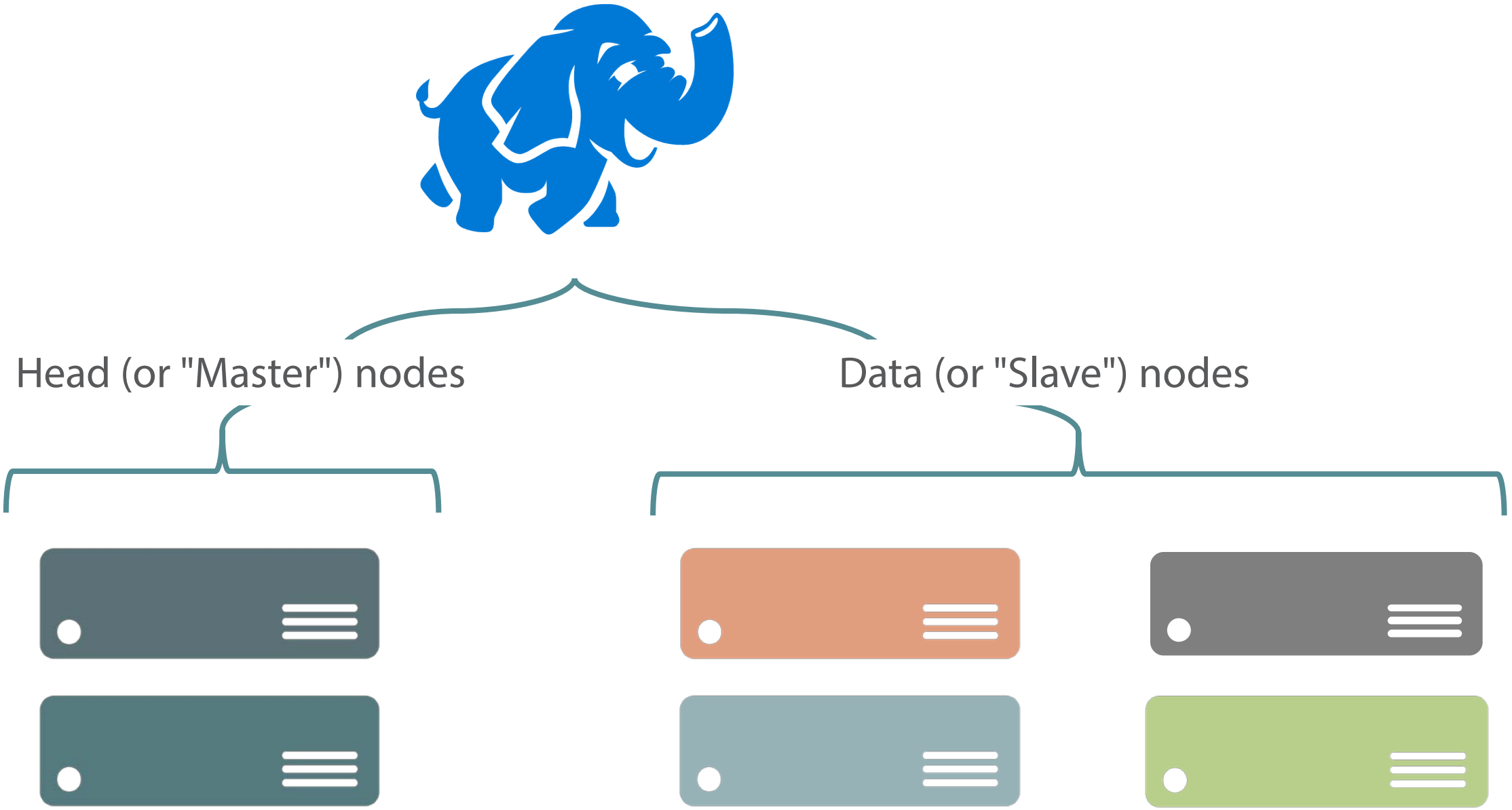
As tab-separated variable files (TSV)

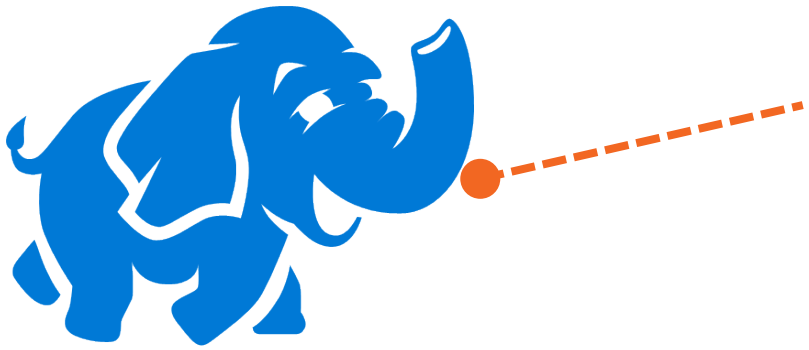
event-count.tsv



event-count.tsv







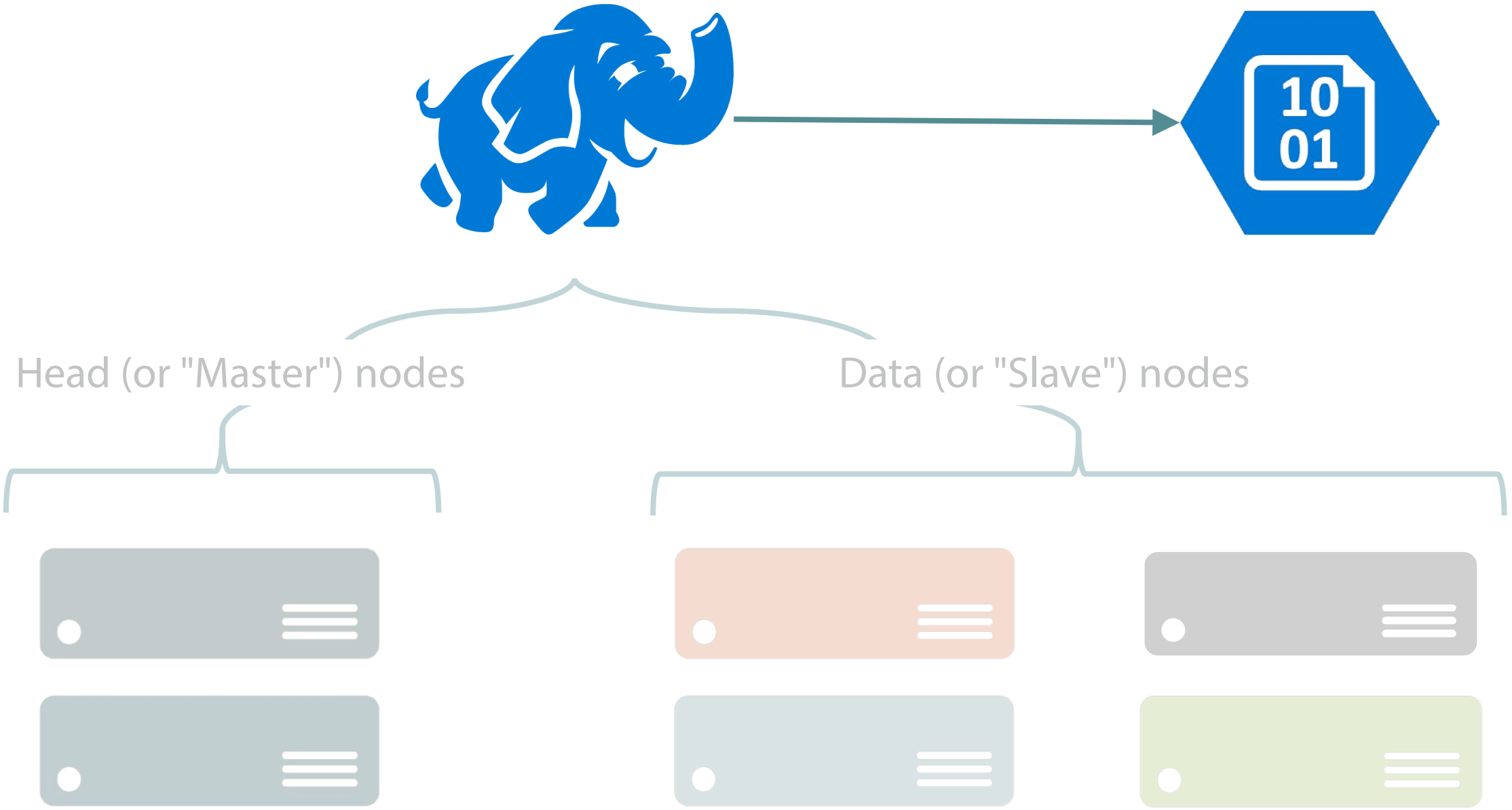
Hadoop

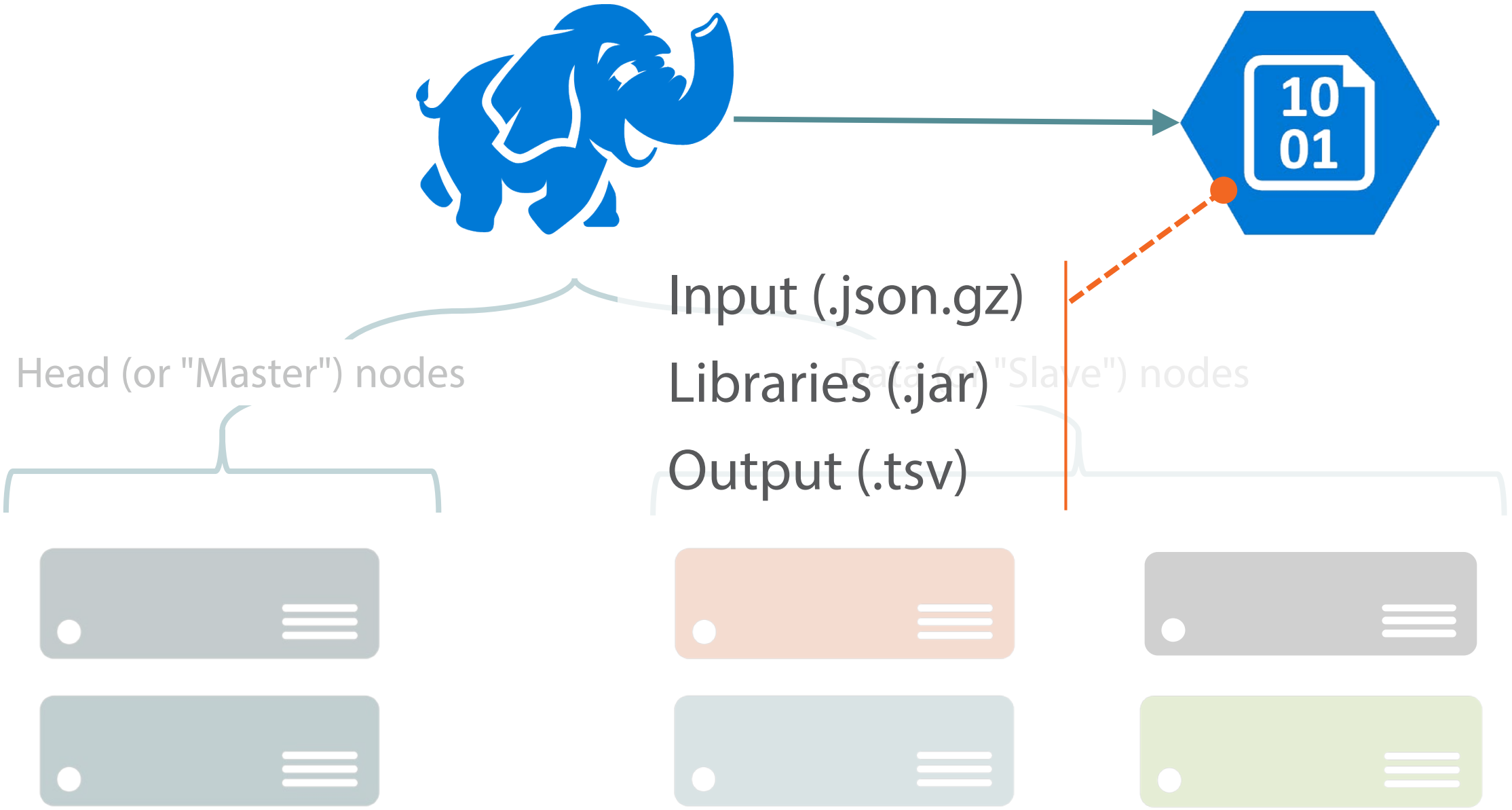
HBase

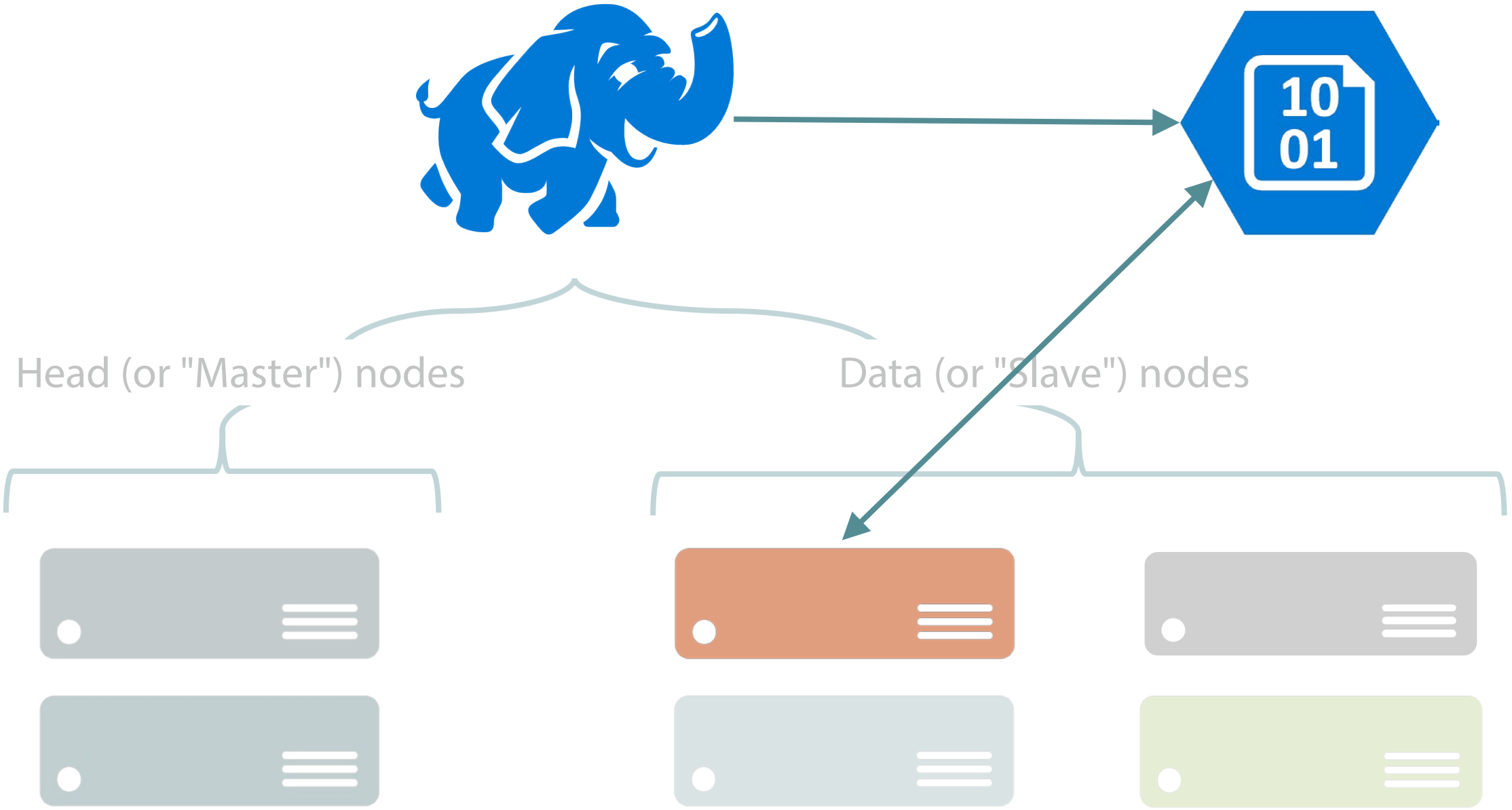
Storm

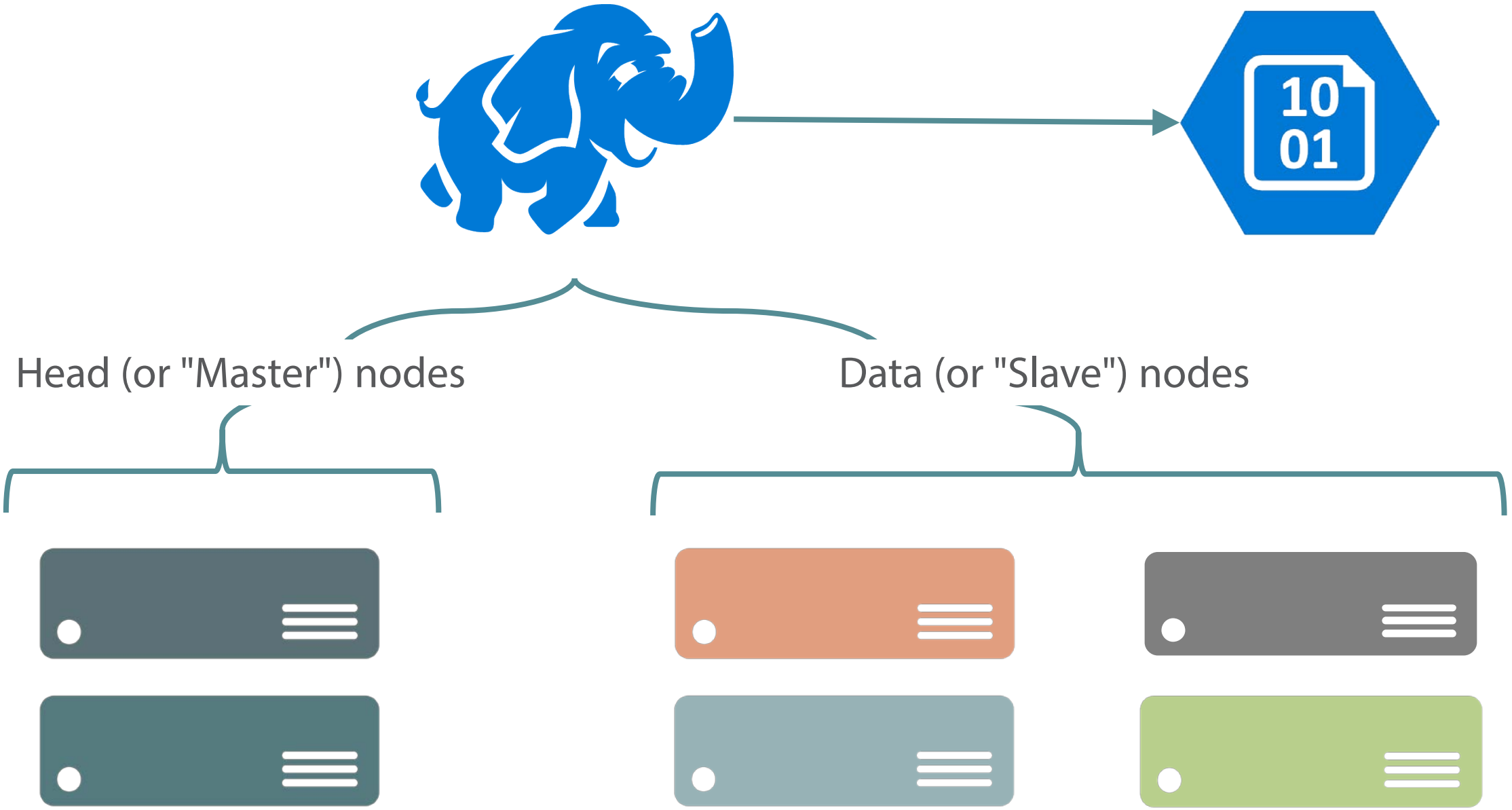
Head (or "Master") nodes

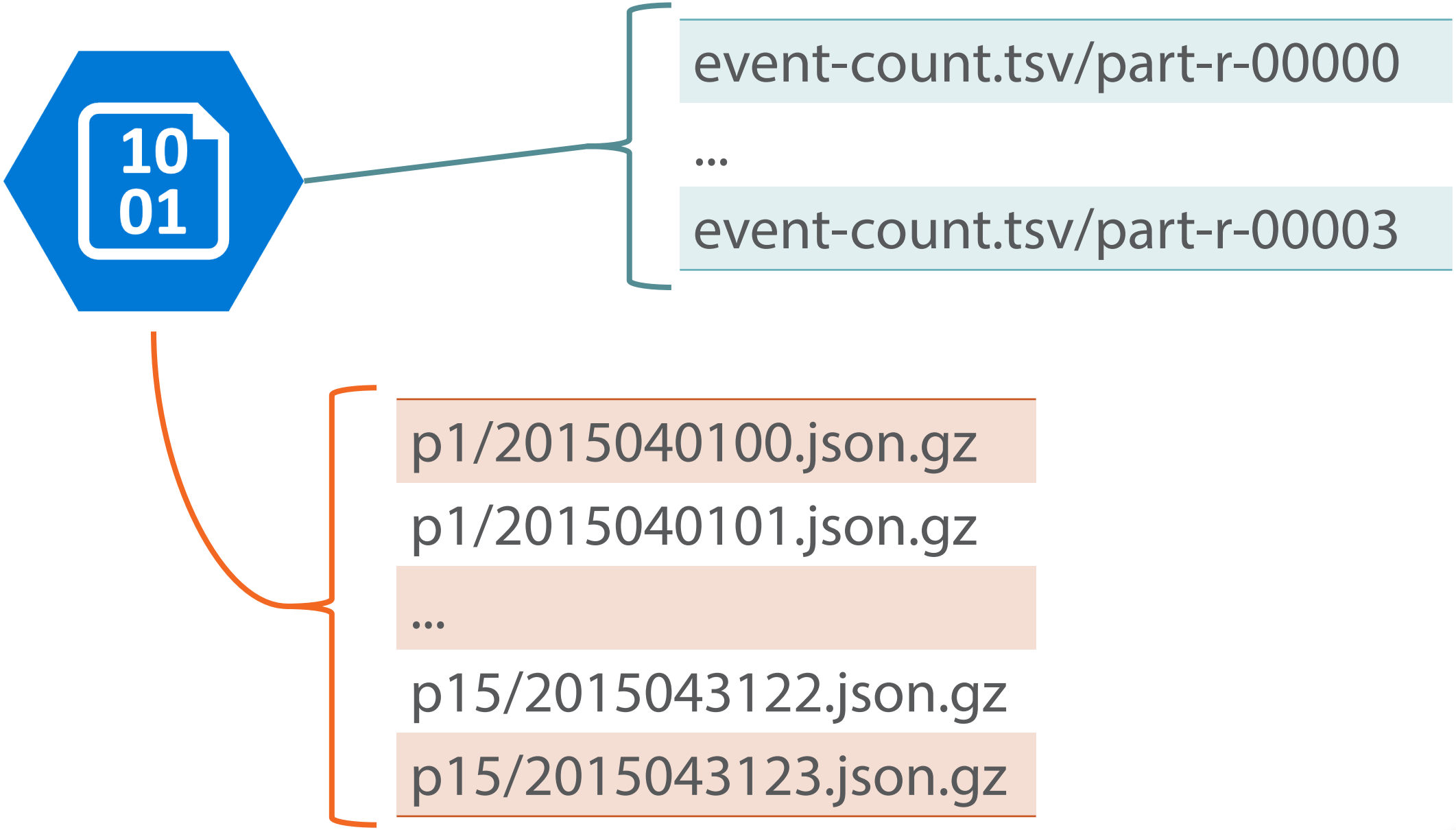
Data (or "Slave") nodes











Demo: HDInsight & Pig

Create HDInsight cluster

Pig query using Blob Storage

Run Pig query on HDInsight

```
New-AzureHDInsightCluster -Name 'deviceeventsprd2'
-Credential $credential -Location 'North Europe'
-DefaultStorageAccountName 'x' -DefaultStorageAccountKey 'y'
-DefaultStorageContainerName 'deviceeventsprd2'
-ClusterSizeInNodes 4 -ClusterType Hadoop
```

Create HDInsight cluster

Storage Account & Hadoop user credentials

```
register 'wasb:///lib/json-simple-1.1.1.jar';
register 'wasb:///lib/elephant-bird-core-4.6.jar';
register 'wasb:///lib/elephant-bird-pig-4.6.jar';
register 'wasb:///lib/elephant-bird-hadoop-compat-4.6.jar';
register 'wasb:///lib/slf4j-api-1.7.10.jar';
```

Register libraries using **wasb://**

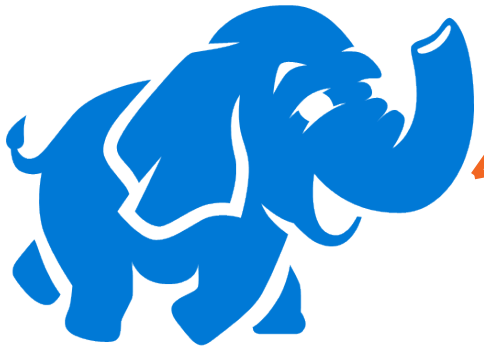
Root container for cluster

```
A = LOAD 'wasb://
      device-events@devicetelemetryprd.blob.core.windows.net/
      2015/03/*/*'
    USING com.twitter.elephantbird.pig.load.JsonLoader()
    AS (json:map[]);
```

Load data using **wasb://**

Specify container, storage account and folder path

```
A = LOAD 'c:/device-events/2015/04/08/*';
B = GROUP A all;
C = FOREACH B GENERATE COUNT(A);
DUMP C;
```



Hadoop Command Line - pig c:\demo3-hdinsight.pig

```
C:\apps\dist\pig-0.12.1.2.1.12.0-2329\bin>pig c:\demo3-hdinsight.pig
2015-04-24 13:27:57,755 [main] INFO  org.apache.pig.Main - Apache Pig version 0.
12.1.2.1.12.0-2329 (r: unknown) compiled Mar 06 2015, 00:51:52
2015-04-24 13:27:57,767 [main] INFO  org.apache.pig.Main - Logging error message
s to: C:\apps\dist\hadoop-2.4.0.2.1.12.0-2329\logs\pig_1429882077755.log
2015-04-24 13:27:59,017 [main] INFO  org.apache.pig.impl.util.Utils - Default bo
otup file D:\Users\elton/.pigbootup not found
2015-04-24 13:27:59,376 [main] INFO  org.apache.hadoop.conf.Configuration.deprec
ation - mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.addr
ess
2015-04-24 13:27:59,376 [main] INFO  org.apache.hadoop.conf.Configuration.deprec
ation - fs.default.name is deprecated. Instead, use fs.defaultFS
2015-04-24 13:27:59,376 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.HExecutionEngine - Connecting to hadoop file system at: wasb://deviceeventspr
d-1@deviceeventsprd.blob.core.windows.net
```

http://headnodehost:9014/proxy/application_142 | MapReduce Job job_142987...

Logged in as: gopher

MapReduce Job job_1429879748587_0001

- Cluster
- Application
- Job
  - Overview
  - Counters
  - Configuration
  - Map tasks
  - Reduce tasks
  - AM Logs
- Tools

Job Overview

Job Name: PigLatin:demo3-hdinsight.pig
State: RUNNING
Uberized: false
Started: Fri Apr 24 13:29:24 GMT 2015
Elapsed: 37sec

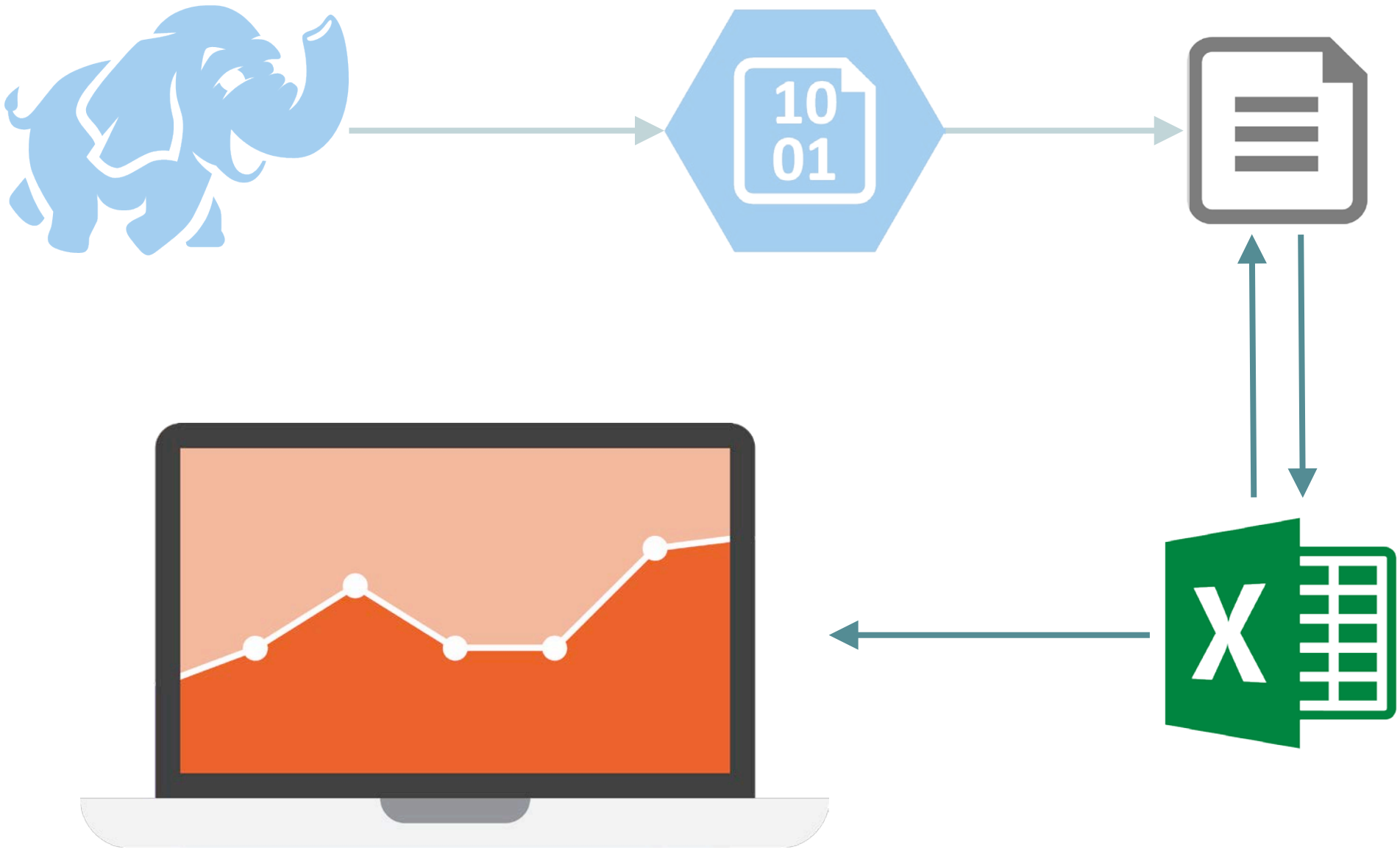
ApplicationMaster

Attempt Number	Start Time	Node	Logs
1	Fri Apr 24 13:29:17 GMT 2015	workernode0.deviceeventsprd.f9.internal.cloudapp.net:30060	logs

Task Type	Progress	Total	Pending	Running	Complete
Map		4043	4036	7	0
Reduce		4	4	0	0

Attempt Type	New	Running	Failed	Killed	Successful
Maps	4036	7	0	0	0
Reduces	4	0	0	0	0

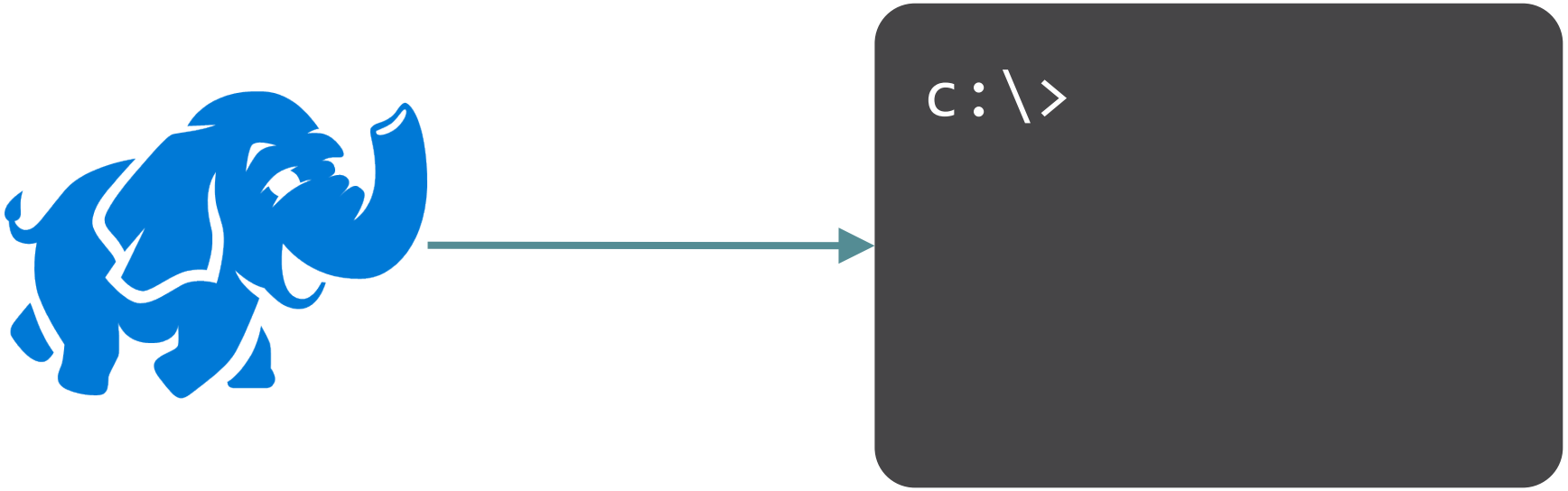
About Apache Hadoop

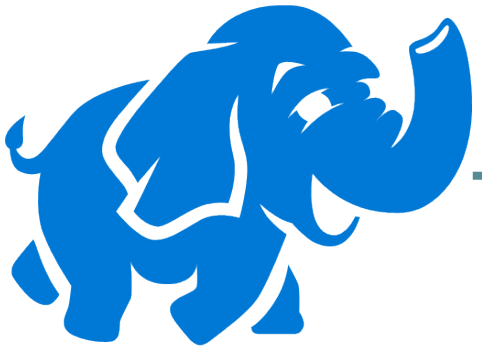


```
register './lib/elephant-bird-pig-4.6.jar';
register './lib/My.DotNet.Assembly.dll';
```

Microsoft® .NET™

Java™

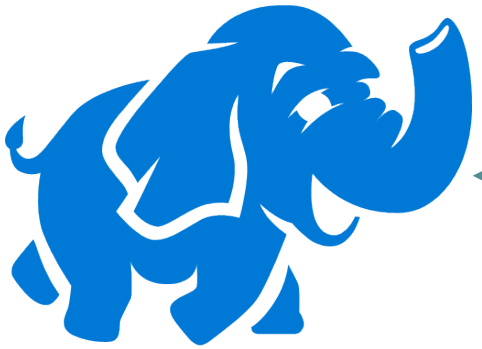




STREAM A INTO ...

"xyz 123 abc"

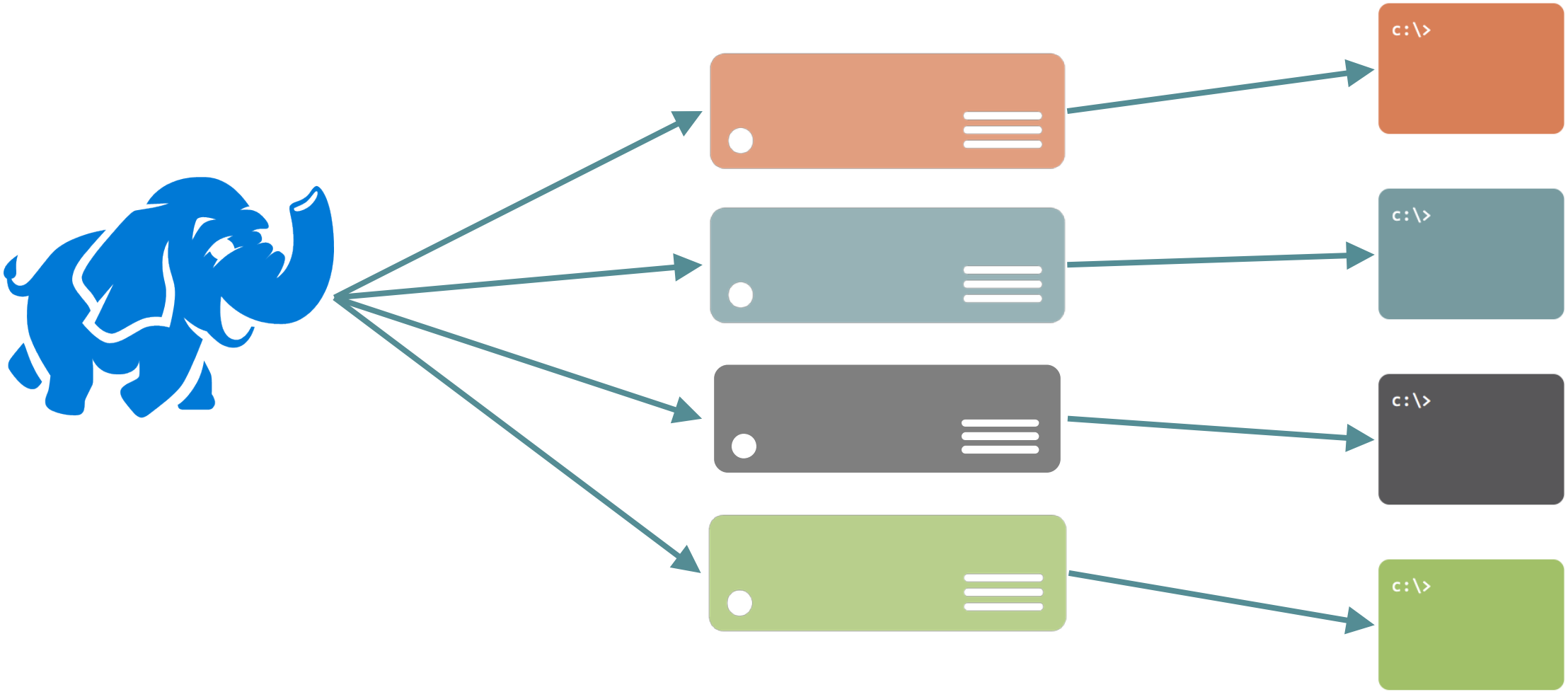
```
c:\>
```

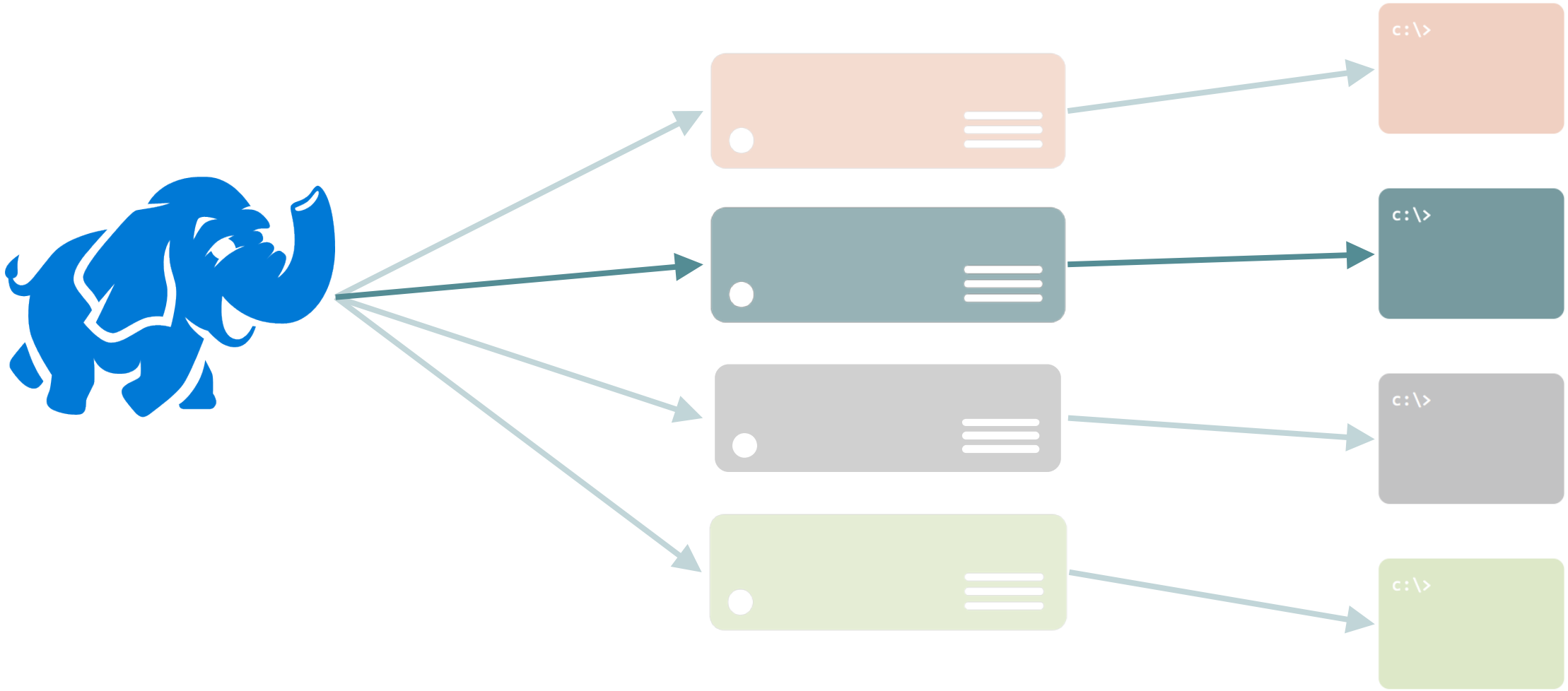


STREAM A INTO ...

"count 8925"

```
c:\>
```





Demo: Streaming into .NET

Stream from Pig to .NET

Simple console app

Logs output & returns to Pig

```
using (var stdin = Console.OpenStandardInput())
using (var inputReader = new StreamReader(stdin))
{
    var line = inputReader.ReadLine();
    while (line != null)
    {
        var fields = line.Split('\t');
```

Stream from Pig to .NET app

Input as TSV lines via console Standard Input

```
using (var stdout = Console.OpenStandardOutput())
using (var outputWriter = new StreamWriter(stdout))
{
    outputWriter.WriteLine(string.Format("{0}\t{1}", ...
```

Write from .NET to Pig

Output as TSV lines via console Standard Output

```
DEFINE X `logger.exe` ship('c:/logger/logger.exe', ...)
...
C = FILTER B by eventName == 'x.y.z';
D = STREAM C through X;
STORE D;
```

Stream data through .NET app

Define command and ship dependencies

```
DEFINE X `CountLogger.exe` ship(
'c:\\bin\\CountLogger.exe',
'c:\\bin\\CountLogger.exe.config',
'c:\\bin\\nlog-prd.config', 'c:\\piglogger\\NLog.dll',
'c:\\bin\\Core.dll', 'c:\\bin\\Newtonsoft.Json.dll');
```

Ship dependencies

Copied from local file system to cluster

```
C = FILTER B by eventName == 'x.y.z';
D = STREAM C through X;
STORE D;
```

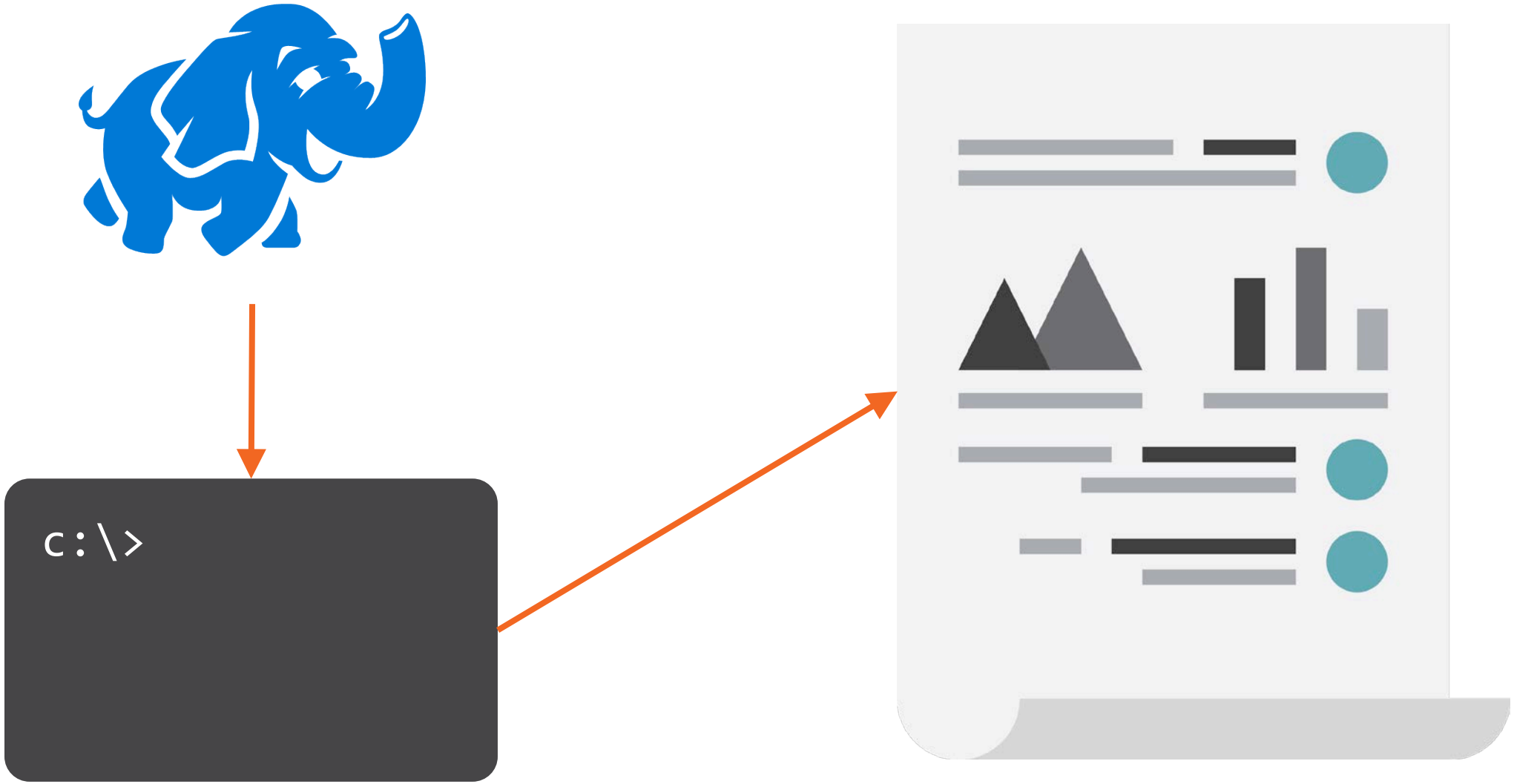
Stream data through .NET app

Use output from app as new relation

```
C = FILTER B by eventName == 'x.y.z';
D = STREAM C through X;
E = GROUP C ALL;
F = FOREACH E GENERATE COUNT(C);
DUMP F;
```

Stream data through .NET app

Ignore the output relation

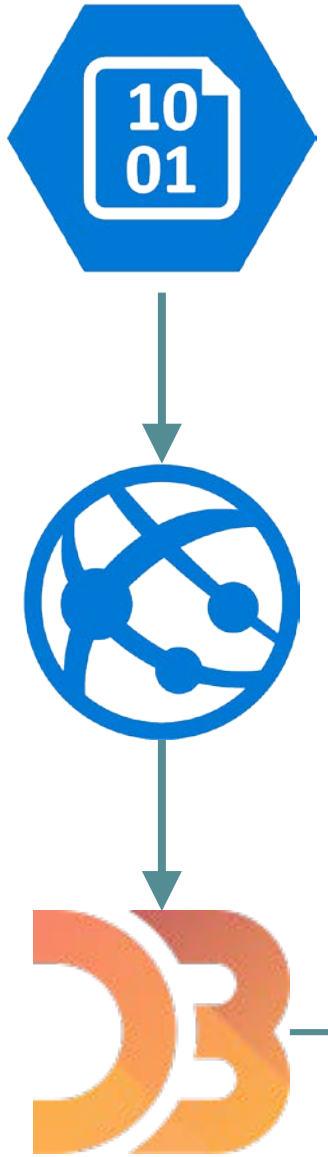


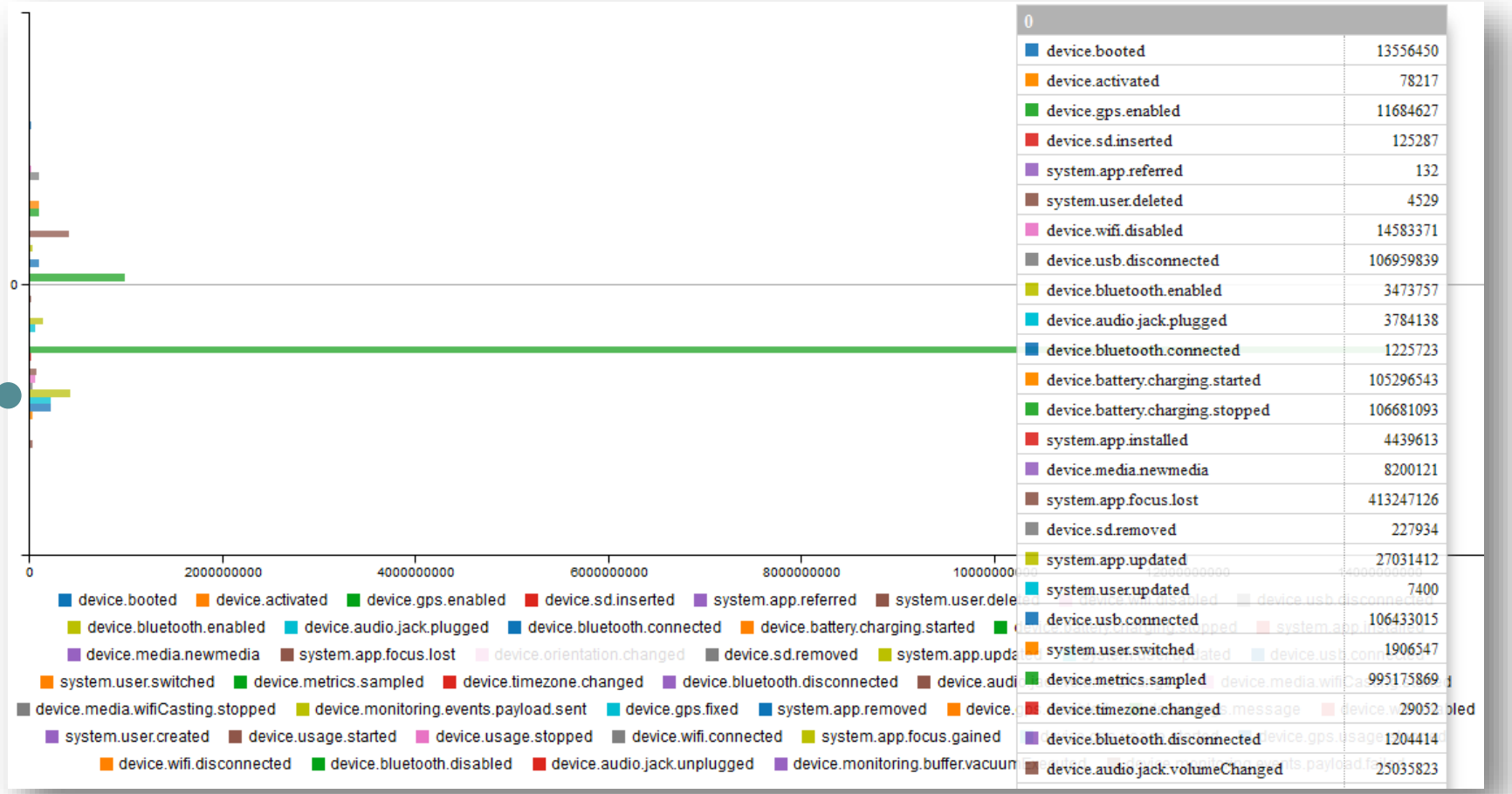
Demo: Visualization with D3.js

Pig output in JSON Blob

Web API formats JSON

D3.js and C3.js visualization





D3.js Data Visualization Fundamentals

Ben Sullins

@BenSullins | www.bensullins.com

```
var prefix = string.Format("{0}.json/part-r-", name);
var matchingBlobs = container.ListBlobs(prefix, true);
foreach (var part in matchingBlobs.OfType<CloudBlockBlob>())
{
    json += part.DownloadText() + Environment.NewLine;
}
```

Read JSON output

Get all part files and combine

```
dynamic raw = JObject.Parse(line);
var formatted = new JArray();
formatted.Add((string)raw.eventName);
formatted.Add((long)raw.count);
outputArray.Add(formatted);
```

Format response

JSON array of data points