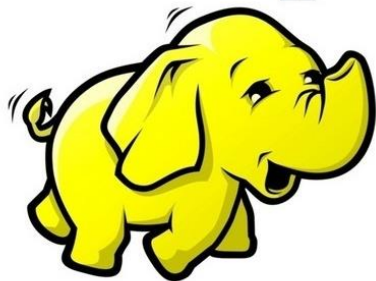


Big Data Engineering With MapReduce and Hive

Data Science Dojo

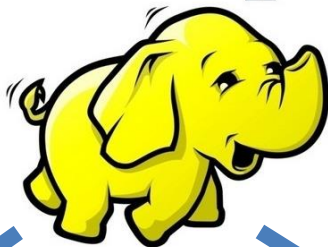
Agenda

hadoop

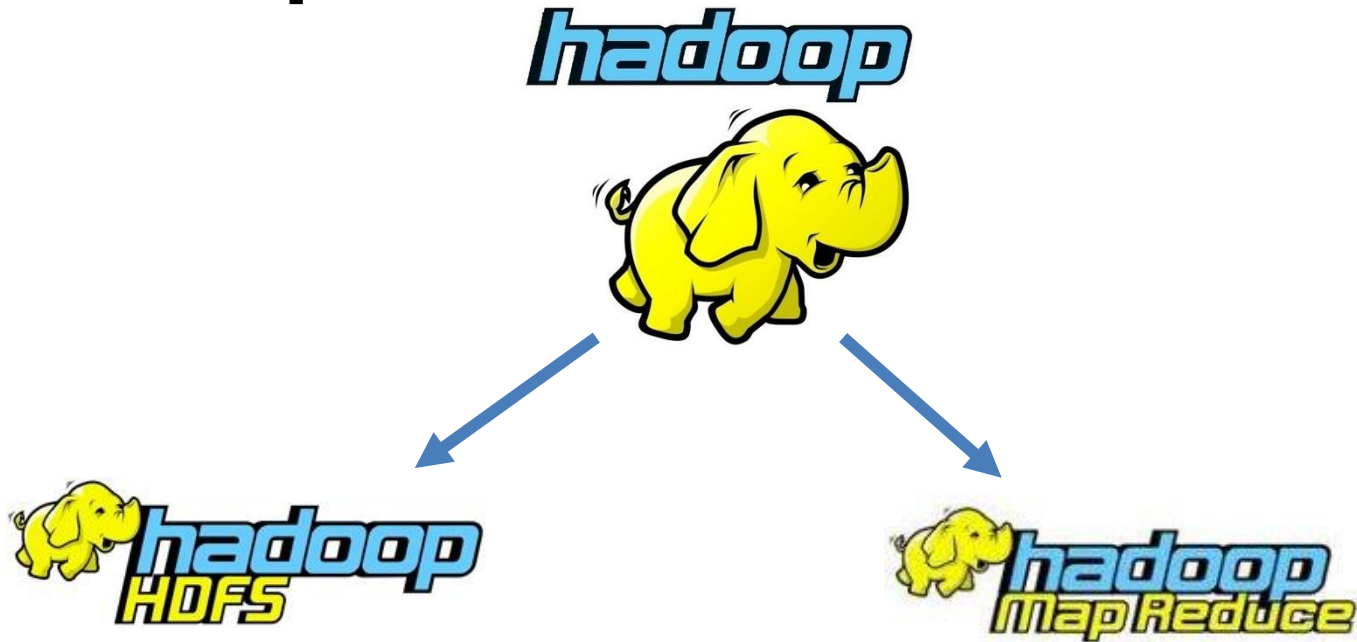


Hadoop Implementations

hadoop



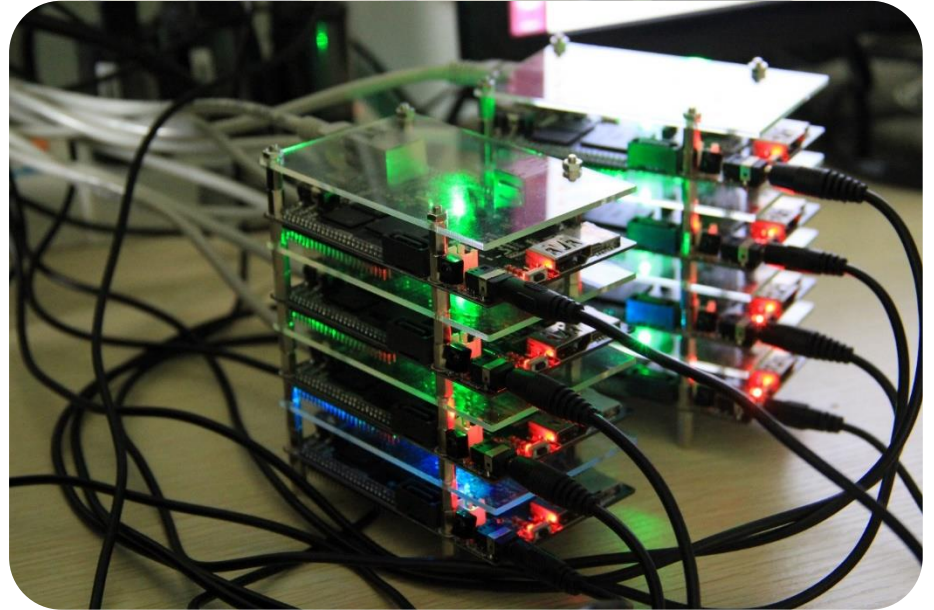
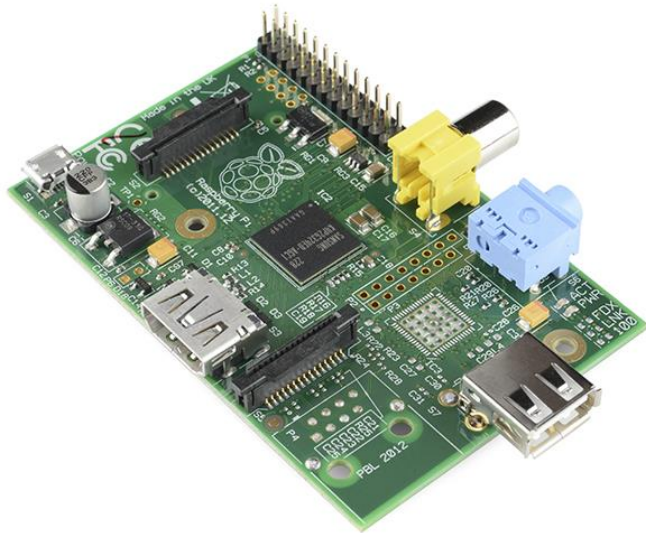
Hadoop



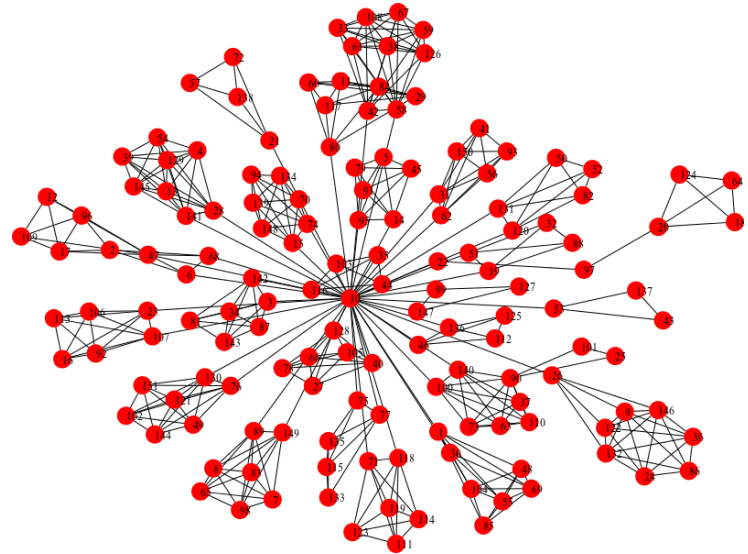
Turn Back The Clock, The Mainframe



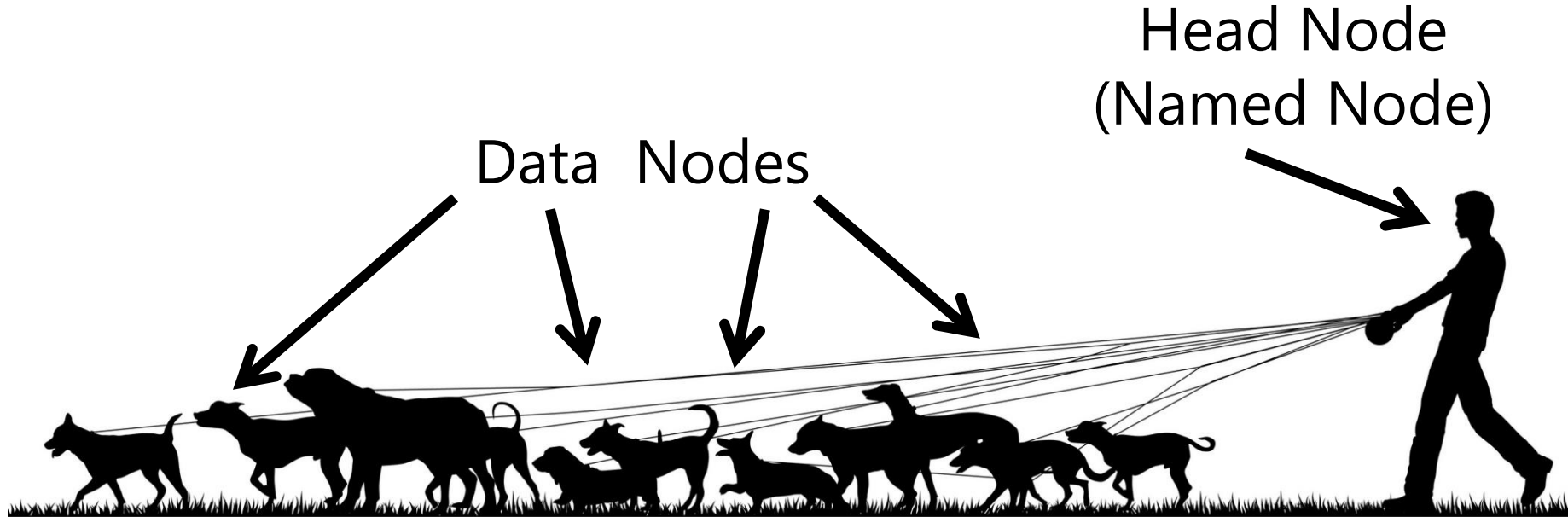
Distributed Computing



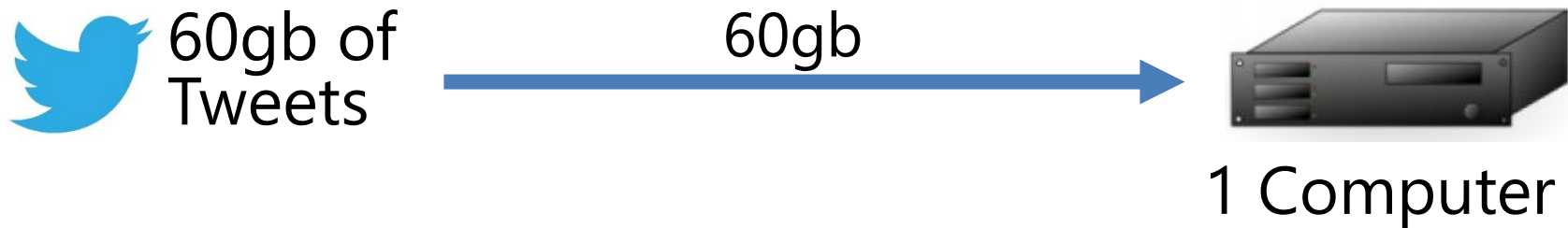
Cloud Computing



If dogs were servers...

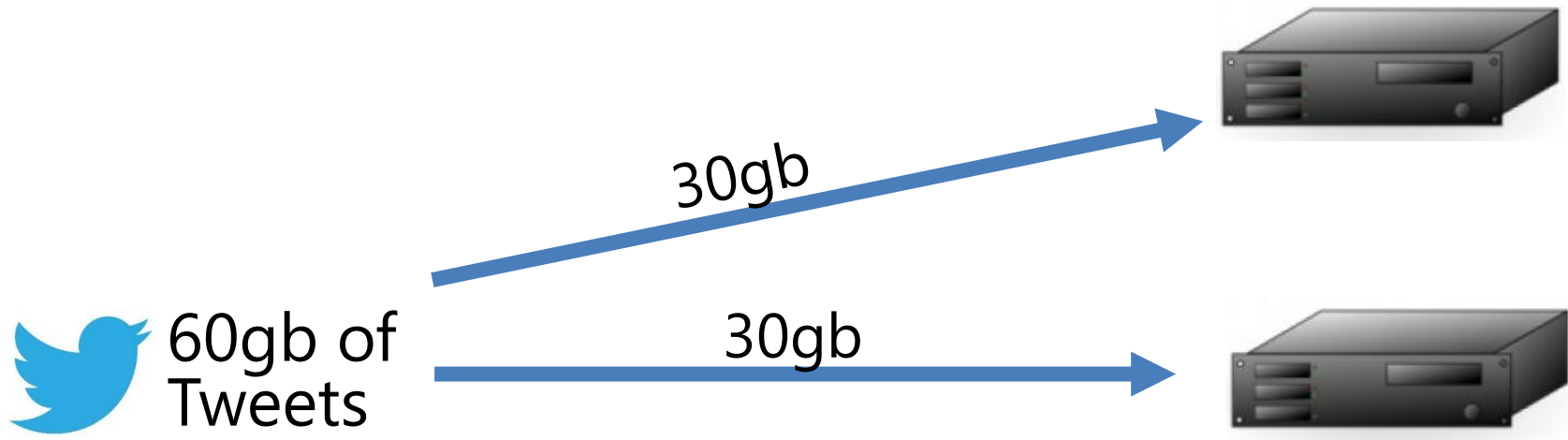


HDFS & MapReduce



Processing: 30 hours

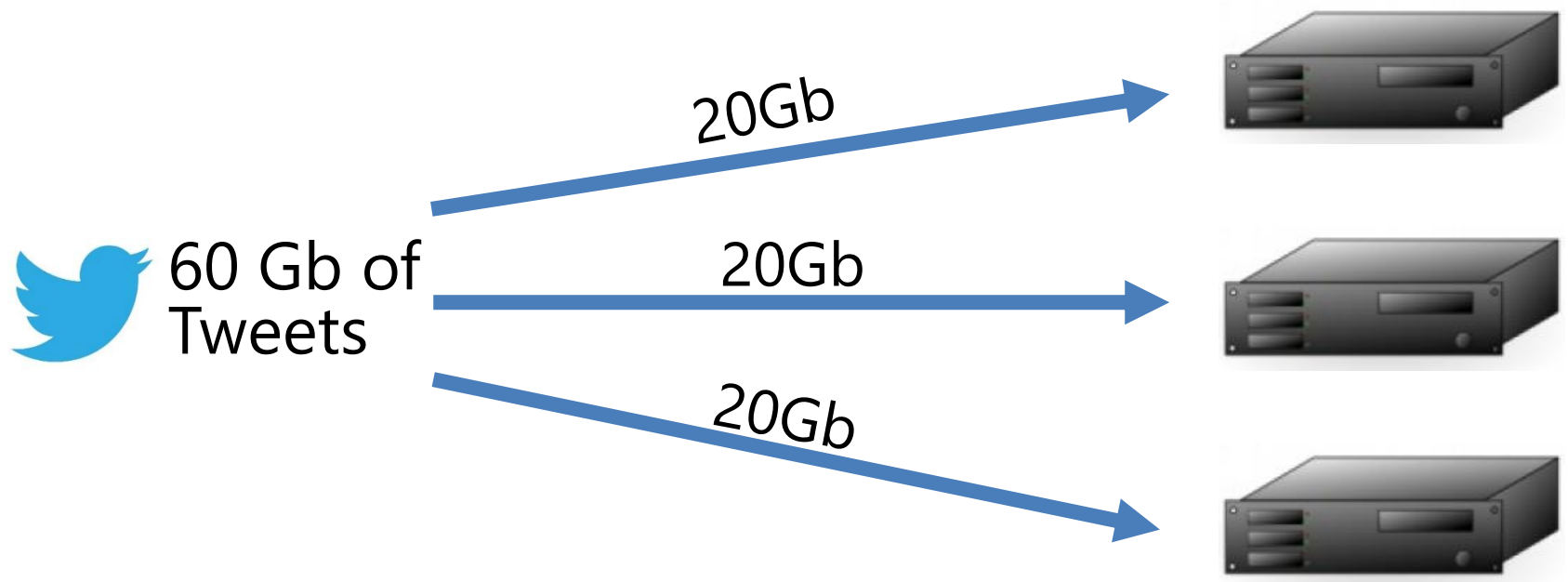
HDFS & MapReduce



2 Computers

Processing: 15 hours

HDFS & MapReduce



Processing: 10 hours

3 Computers

Most Cases, Linear Scaling Of Processing Power

Number of Computers	Processing Time (hours)
1	30
2	15
3	10
4	7.5
5	6
6	5
7	4.26
8	3.75
9	3.33

Limitations with MapReduce

- ~200 lines of code to do anything
- Slow
- Troubleshooting multiple computers
- Good devs are scarce
- Expensive certifications

```
1 package org.apache.hadoop.examples;
2
3 import java.io.IOException;
4 import java.util.StringTokenizer;
5
6 import org.apache.hadoop.conf.Configuration;
7 import org.apache.hadoop.fs.Path;
8 import org.apache.hadoop.io.IntWritable;
9 import org.apache.hadoop.io.Text;
10 import org.apache.hadoop.mapreduce.Job;
11 import org.apache.hadoop.mapreduce.Mapper;
12 import org.apache.hadoop.mapreduce.Reducer;
13 import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
14 import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
15 import org.apache.hadoop.util.GenericOptionsParser;
16
17 public class WordCount {
18
19     public static class TokenizerMapper
20         extends Mapper<Object, Text, Text, IntWritable>{
21
22         private final static IntWritable one = new IntWritable(1);
23         private Text word = new Text();
24
25         public void map(Object key, Text value, Context context
26             ) throws IOException, InterruptedException {
27             StringTokenizer itr = new StringTokenizer(value.toString());
28             while (itr.hasMoreTokens()) {
29                 word.set(itr.nextToken());
30                 context.write(word, one);
31             }
32         }
33     }
```



Ambari: Cluster provisioning, management, and monitoring



Avro (Microsoft .NET Library for Avro): Data serialization for the Microsoft .NET environment



HBase: Non-relational database for very large tables



HDFS: Hadoop Distributed File System



Hive: SQL-like querying



Mahout: Machine learning

MapReduce and YARN: Distributed processing and resource management



Oozie: Workflow management



Pig: Simpler scripting for MapReduce transformations



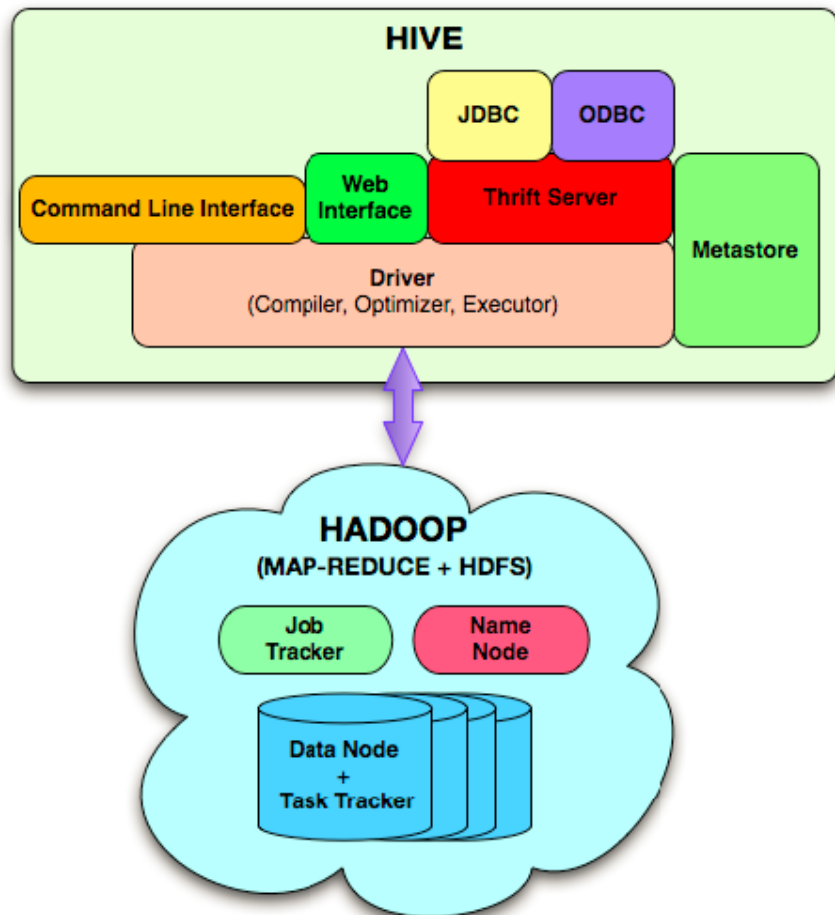
Sqoop: Data import and export



Storm: Real-time processing of fast, large data streams



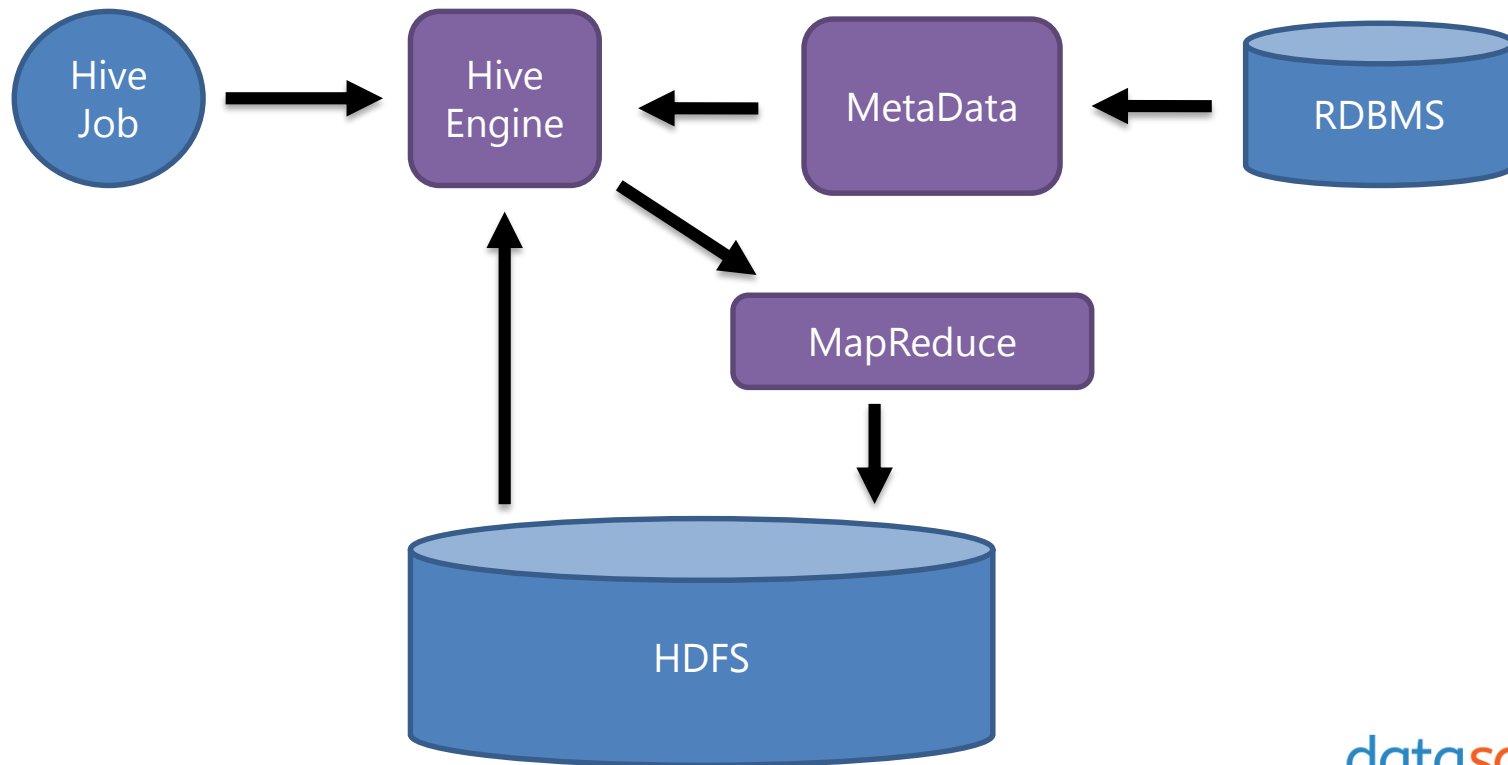
Zookeeper: Coordinates processes in distributed systems



Hive Jobs



Hive Architecture





Data File



Unstructured
Data



Data File



Metadata File/DB



Structured
Data

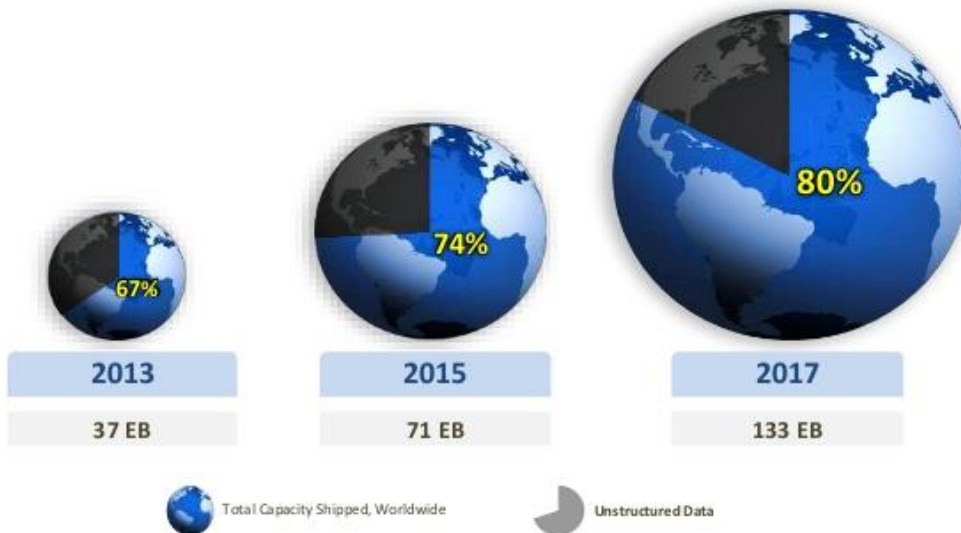
Why Hive?



- SQL spoken here (HiveQL)
- ODBC driver
- BI Integration
- Supports only Structured Data

Limitations

Structured vs. Unstructured Data Growth

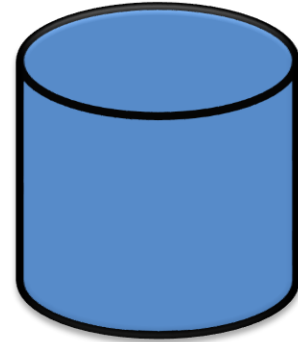
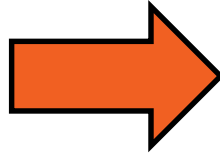


Source: IDC

Azure Blob Storage

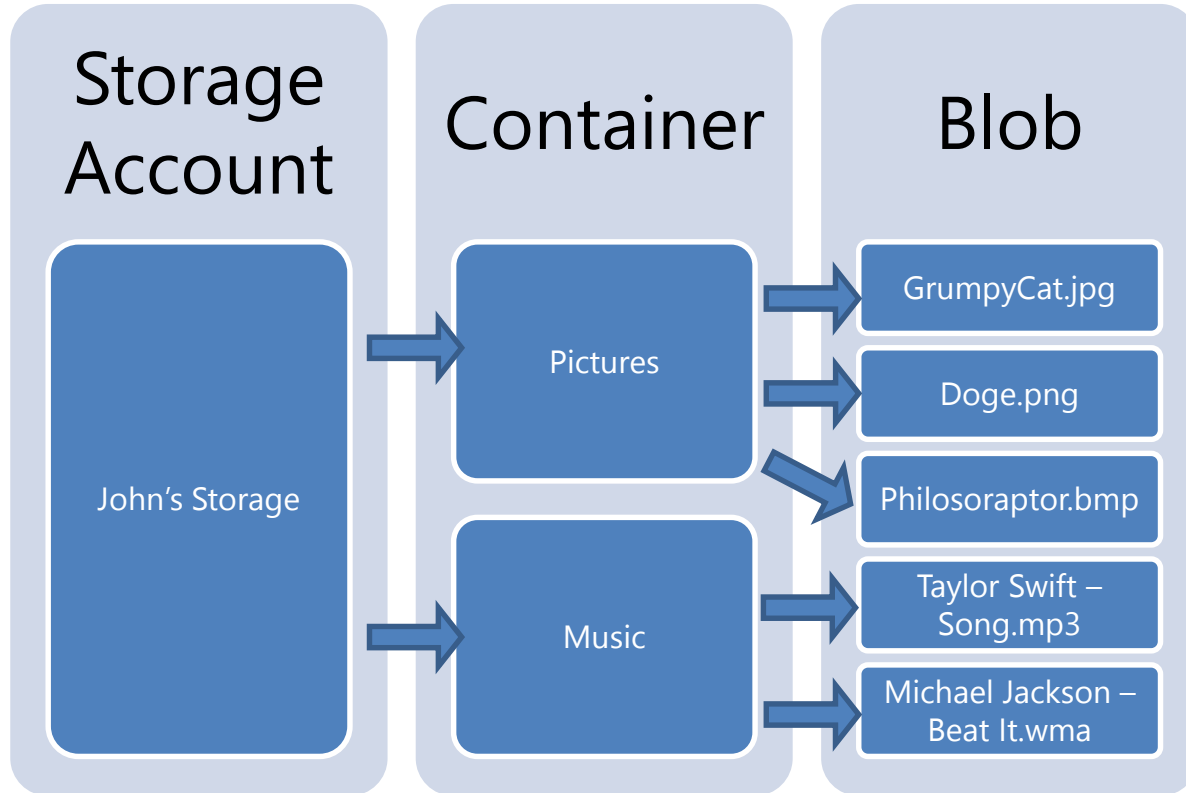


HDInsight

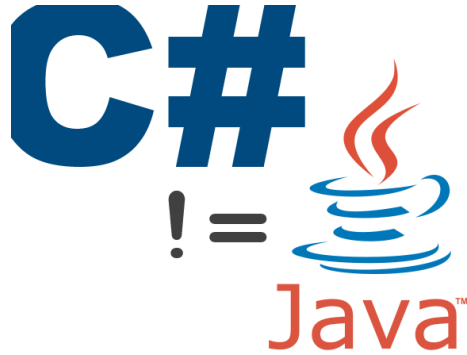


Blob Storage

Azure Blob Storage



When to Use Each



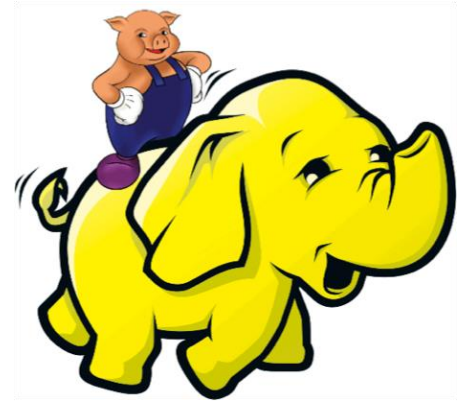
C#
Java
MapReduce

VS



Hive

VS



Pig

QUESTIONS