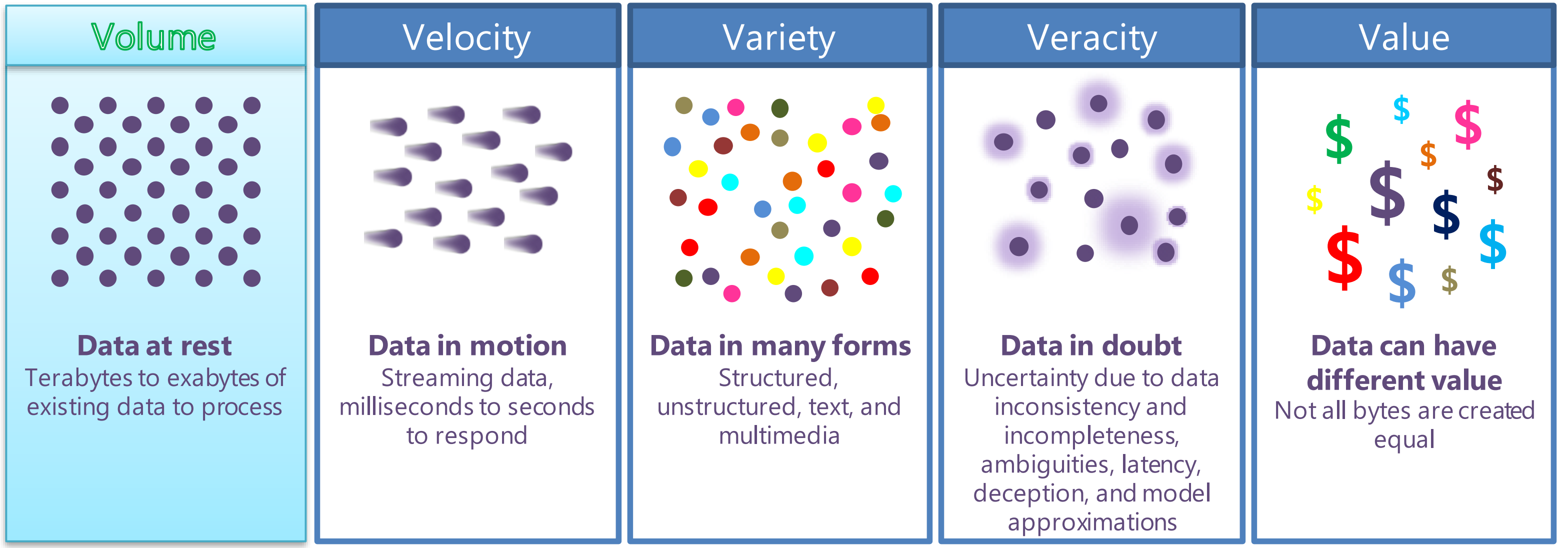
Big Data Engineering with Distributed Systems

Data Science Dojo

datasciencedojo
unleash the data scientist in you

Batch Processing



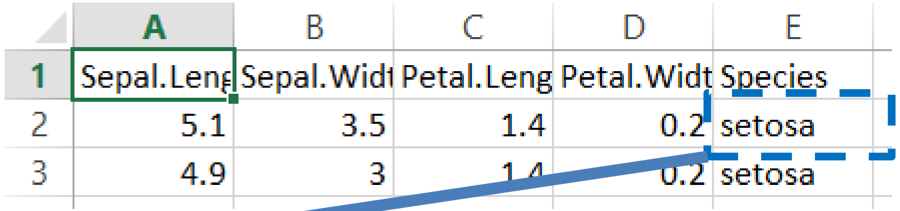
Volume	Velocity	Variety	Veracity	Value
Data at rest	**Data in motion**	**Data in many forms**	**Data in doubt**	**Data can have different value**
Terabytes to exabytes of existing data to process	Streaming data, milliseconds to seconds to respond	Structured, unstructured, text, and multimedia	Uncertainty due to data inconsistency and incompleteness, ambiguities, latency, deception, and model approximations	Not all bytes are created equal

- Save up all your raw data and process all at once
- Addresses the ‘volume’ problem of big data

Machine Learning Scaling

Programs	Programming	Cloud	Distributed
• Excel	• Python	• Azure ML	• Hadoop
	• R	• AWS ML	• Spark
	• SAS	• Big ML	• H20
		• Cloud Virtual Machines	• Revolution R

Excel: Cell Meta Data



	A	B	C	D	E
1	Sepal.Leng	Sepal.Widt	Petal.Leng	Petal.Widt	Species
2	5.1	3.5	1.4	0.2	setosa
3	4.9	3	1.4	0.2	setosa

E2 Cell = Application, Address, AllowEdit, Areas, Borders, BottomPadding, Comment, Column, ColumnIndex, Creator, **Font,** FitText, Height, HeightRule, ID, Interior, LeftPadding, NestingLevel, RightPadding, Row, RowIndex, Shading, Tables, TopPadding, VerticalAlignment, **Value,** Width, WordWrap

“Value”: “Setosa”

"Font":{
    "Application": "Microsoft Excel",
    "Background": None,
    "Bold": True,
    "Color": 0,
    "ColorIndex": 5,
    "Creator": "XCEL",
    "FontStyle": "Bold Italic",
    "Italic": True,
    "Name": "Comic Sans MS",
    "OutlineFont": True,
    "Parent": None,
    "Shadow": False,
    "Size": 12,
    "Strikethrough": False,
    "Subscript": False,
    "ThemeColor": 12,
    "ThemeFont:": 2,
    "TintAndShade": 1,
    "Superscript": False,
    "Underline": False,
}

datasciencedojo
unleash the data scientist in you

R Limits

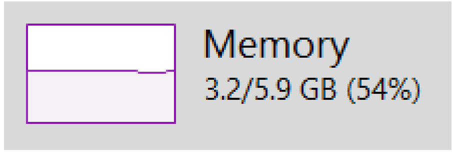
- Single core
- Single threaded
- All in memory (RAM)
- Vectors & Matrices capped at 4,294,967,295 elements (rows) if 32-bit version; 2^32 - 1

R Limits: RAM

- All in memory (RAM)

$$Max\ Data\ Limit = (\ Total\ RAM\ Access\ - Normal\ RAM\ Usage\ )\ x\ 80\%$$

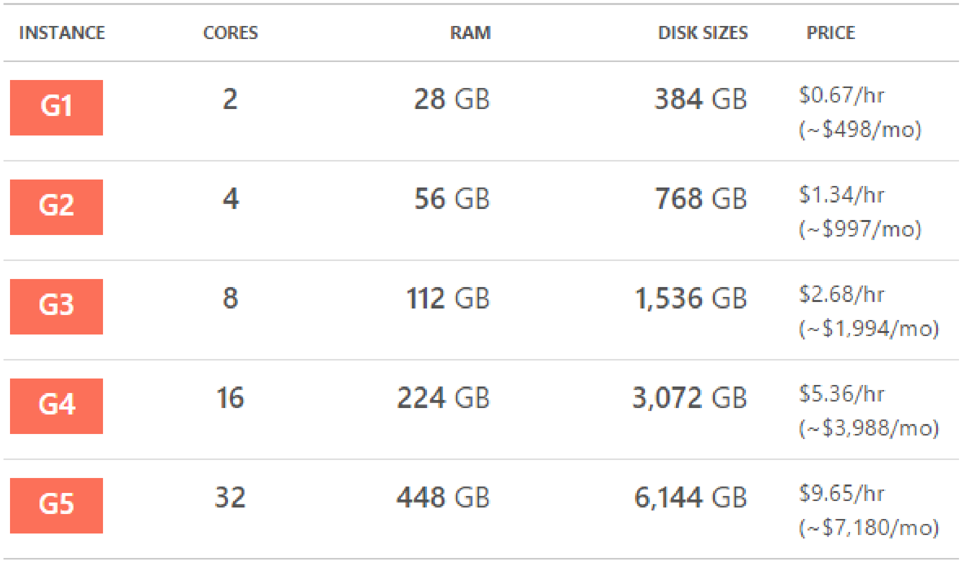
Phuc's Laptop Example:


Memory
3.2/5.9 GB (54%)

$$Max\ Data\ Limit = (\ 5.9gb - \ 3.2gb)\ x\ 80\%$$
$$Max\ Data\ Limit = \sim 2.16gb$$

datasciencedojo
unleash the data scientist in you

R Limits: RAM



INSTANCE	CORES	RAM	DISK SIZES	PRICE
G1	2	28 GB	384 GB	\$0.67/hr (~\$498/mo)
G2	4	56 GB	768 GB	\$1.34/hr (~\$997/mo)
G3	8	112 GB	1,536 GB	\$2.68/hr (~\$1,994/mo)
G4	16	224 GB	3,072 GB	\$5.36/hr (~\$3,988/mo)
G5	32	448 GB	6,144 GB	\$9.65/hr (~\$7,180/mo)

Azure's Biggest Virtual Machine

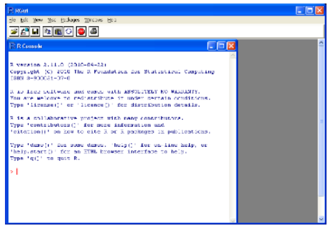
$Max\ Data\ Limit = (\,448gb - 1gb\,)\ x\ 80\%$

$Max\ Data\ Limit = {\sim}\mathbf{357.6gb}$

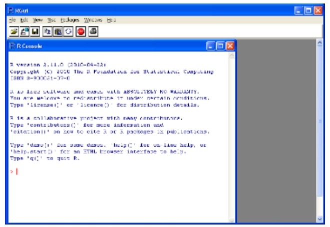
R Limits: Single Core

- Single core
- Single threaded

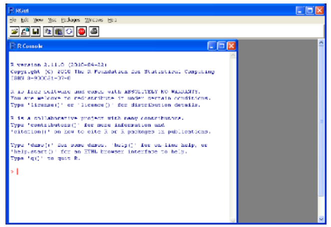
Quad Core Laptop



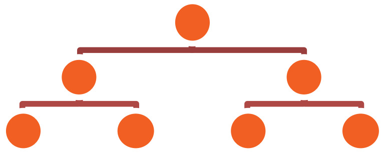
```
R version 2.11.0 (2010-04-22)
Copyright (C) 2010 The R Foundation for Statistical Computing
ISBN 3-900051-07-0

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

>
```



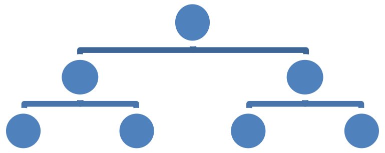
```
R version 2.11.0 (2010-04-22)
Copyright (C) 2010 The R Foundation for Statistical Computing
ISBN 3-900051-07-0

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

>
```



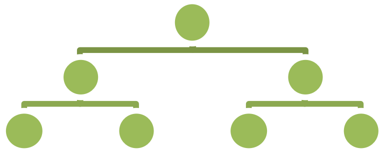
```
R version 2.11.0 (2010-04-22)
Copyright (C) 2010 The R Foundation for Statistical Computing
ISBN 3-900051-07-0

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

>
```



Model A



Model B



Model C

datasciencedojo
unleash the data scientist in you

Machine Learning Scaling

Programs	Programming	Cloud	Distributed
• Excel	• Python	• Azure ML	• Hadoop
	• R	• AWS ML	• Spark
	• SAS	• Watson Analytics	• H20
		• Big ML	• Revolution R
		• Cloud Virtual Machines	

Distributed R Solutions:

https://cran.r-project.org/web/views/HighPerformanceComputing.html

datasciencedojo
unleash the data scientist in you

Agenda

hadoop

HIVE

From a Data Scientist's Perspective

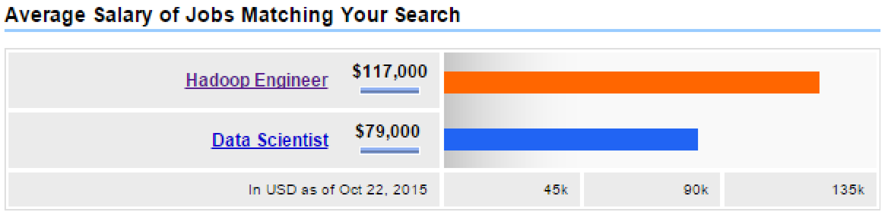
Goals:

- Teach you how to leverage an existing Hadoop cluster, self-service data query

Not goals:

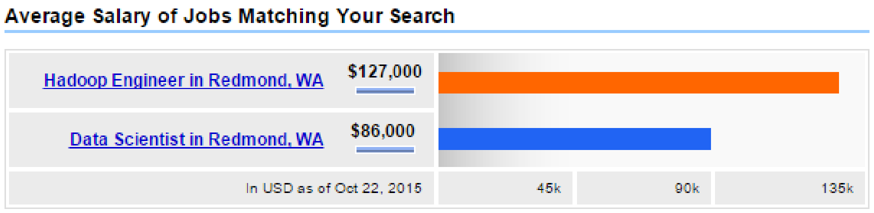
- Managing or administering a Hadoop cluster

Hadoop Engineers


Average Salary of Jobs Matching Your Search

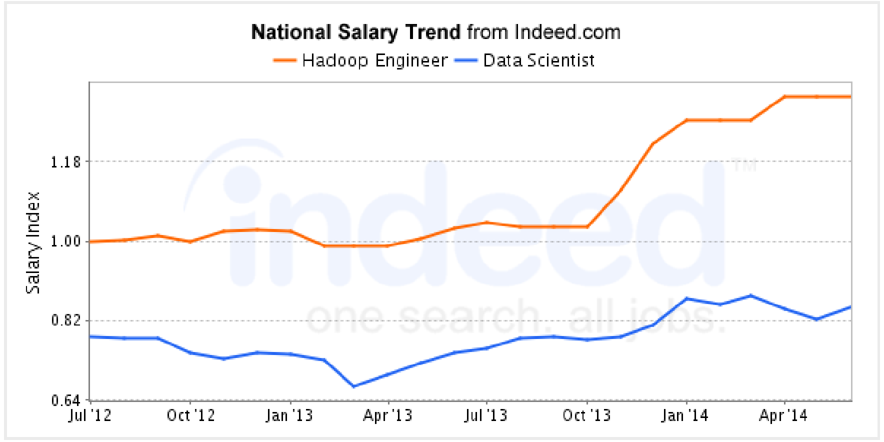
Hadoop Engineer	$117,000	
Data Scientist	$79,000	
In USD as of Oct 22, 2015		45k 90k 135k

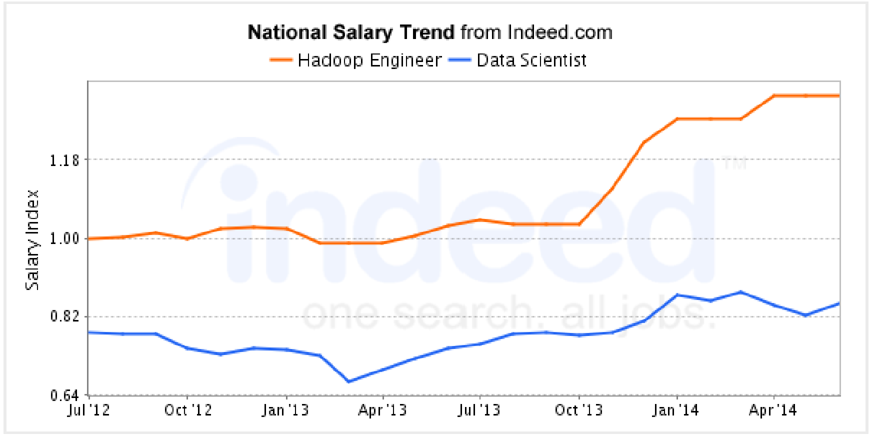
Average Hadoop Engineer salaries for job postings nationwide are 47% higher than average Data Scientist salaries for job postings nationwide.


Average Salary of Jobs Matching Your Search

Hadoop Engineer in Redmond, WA	$127,000	
Data Scientist in Redmond, WA	$86,000	
In USD as of Oct 22, 2015		45k 90k 135k

Average Hadoop Engineer salaries for job postings in Redmond, WA are 47% higher than average Data Scientist salaries for job postings in Redmond, WA.

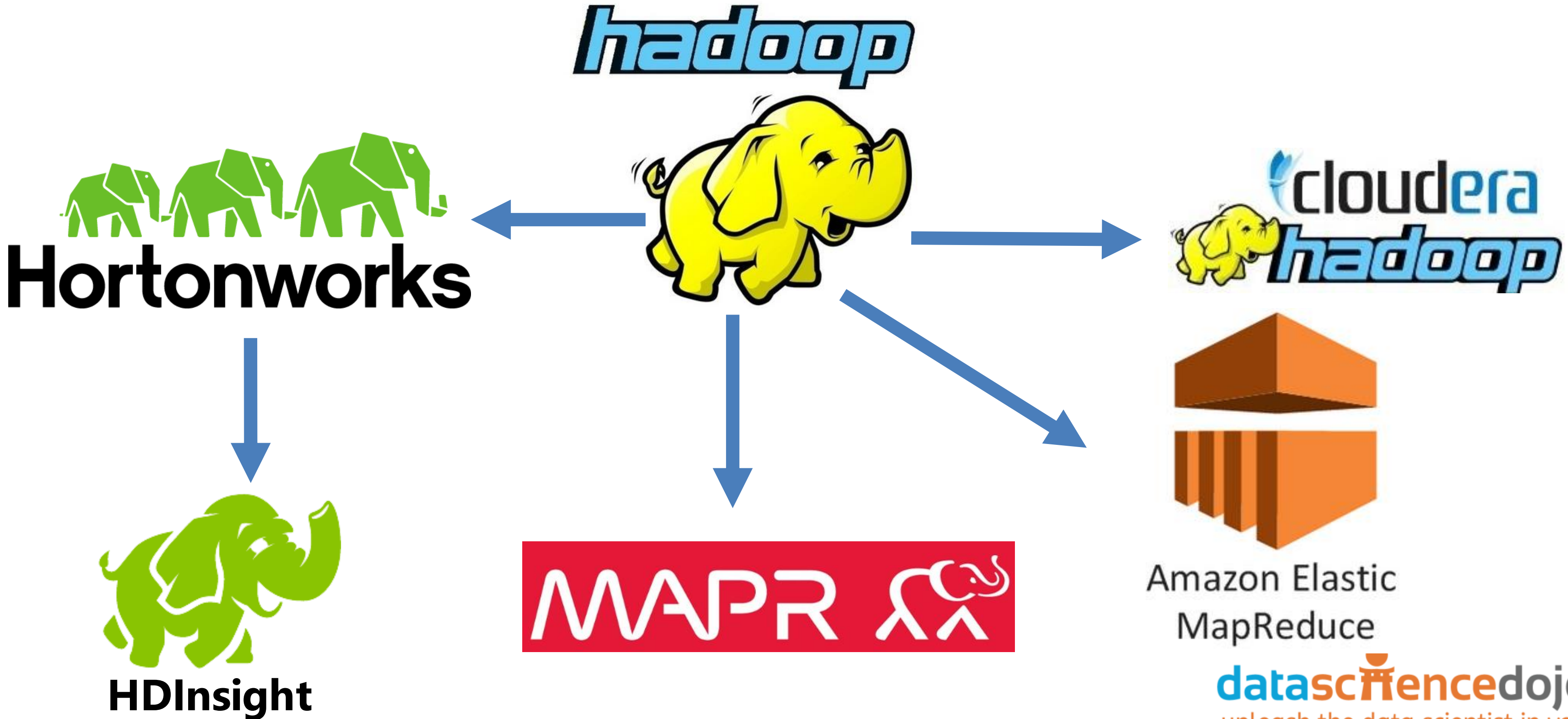




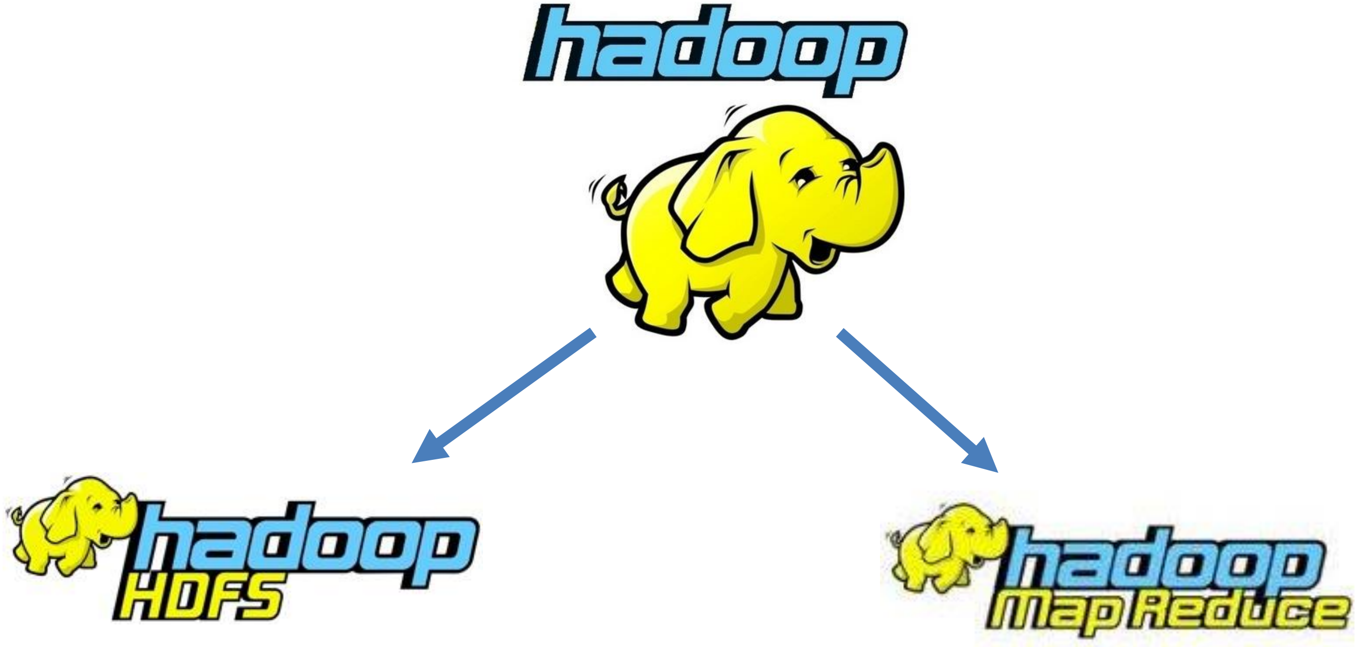
Source: Ineed.com

datasciencedojo
unleash the data scientist in you

Hadoop Implementations



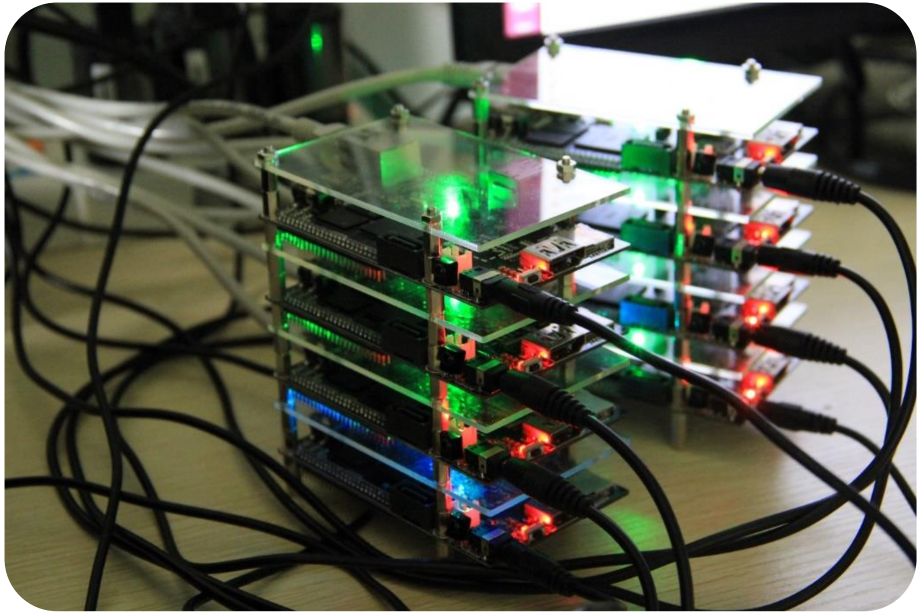
(Vanilla/Base) Hadoop



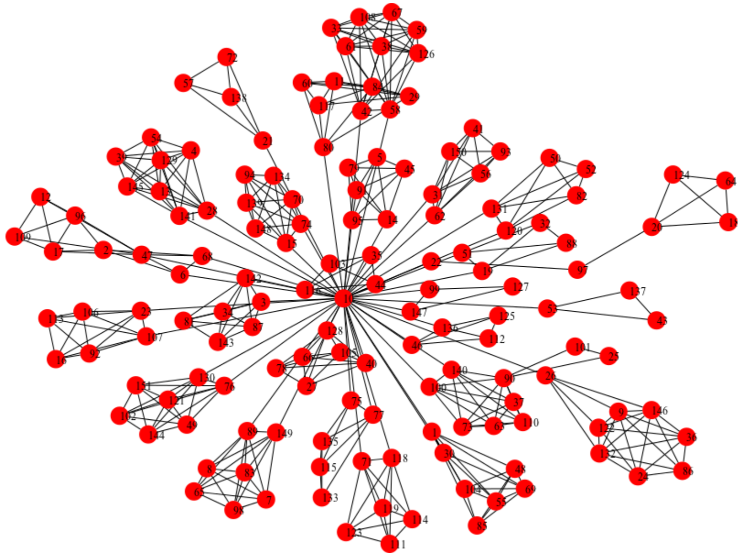
Processing engine for distributed batch processing.

Turn Back The Clock, The Mainframe

Distributed Computing

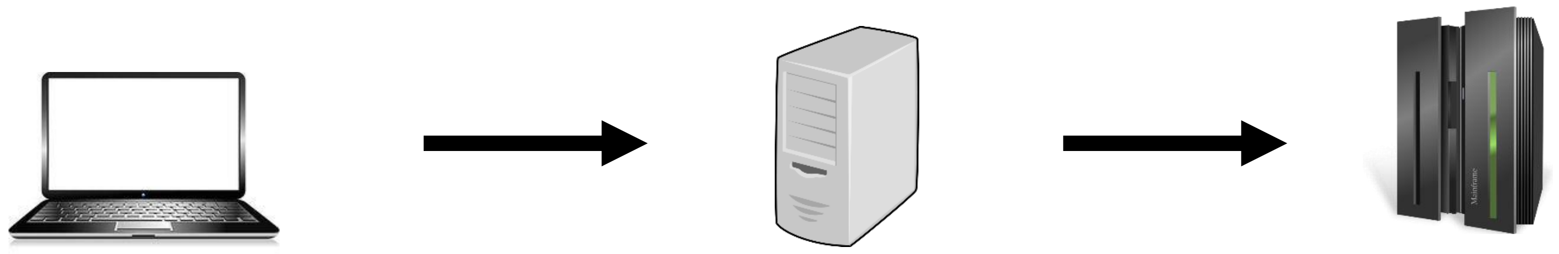


Cloud Computing



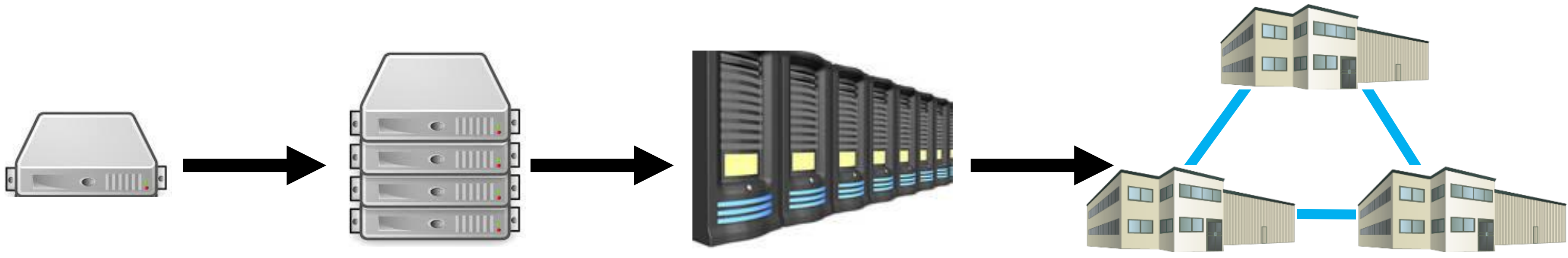
datasciencedojo
unleash the data scientist in you

Scaling Computational Power



Old Scaling:

- Vertical Scaling, Scaling UP
- High performance computers

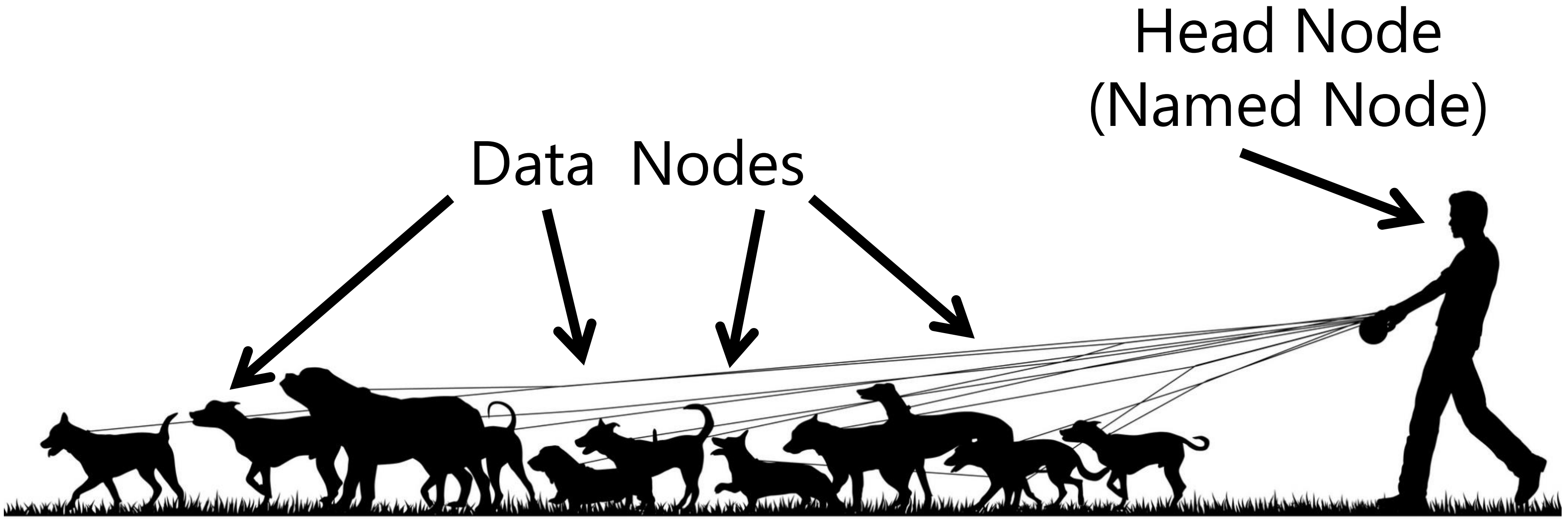


New Scaling:

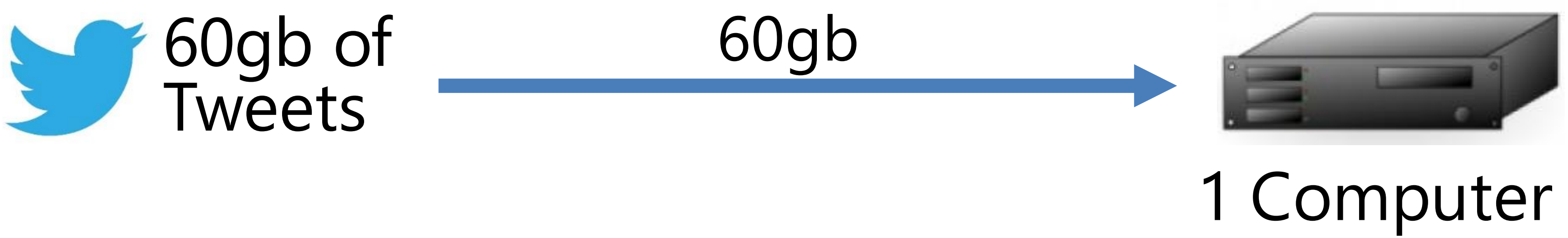
- Horizontal Scaling, Scaling OUT
- Commodity hardware, distributed

datasciencedojo
unleash the data scientist in you

If dogs were servers...

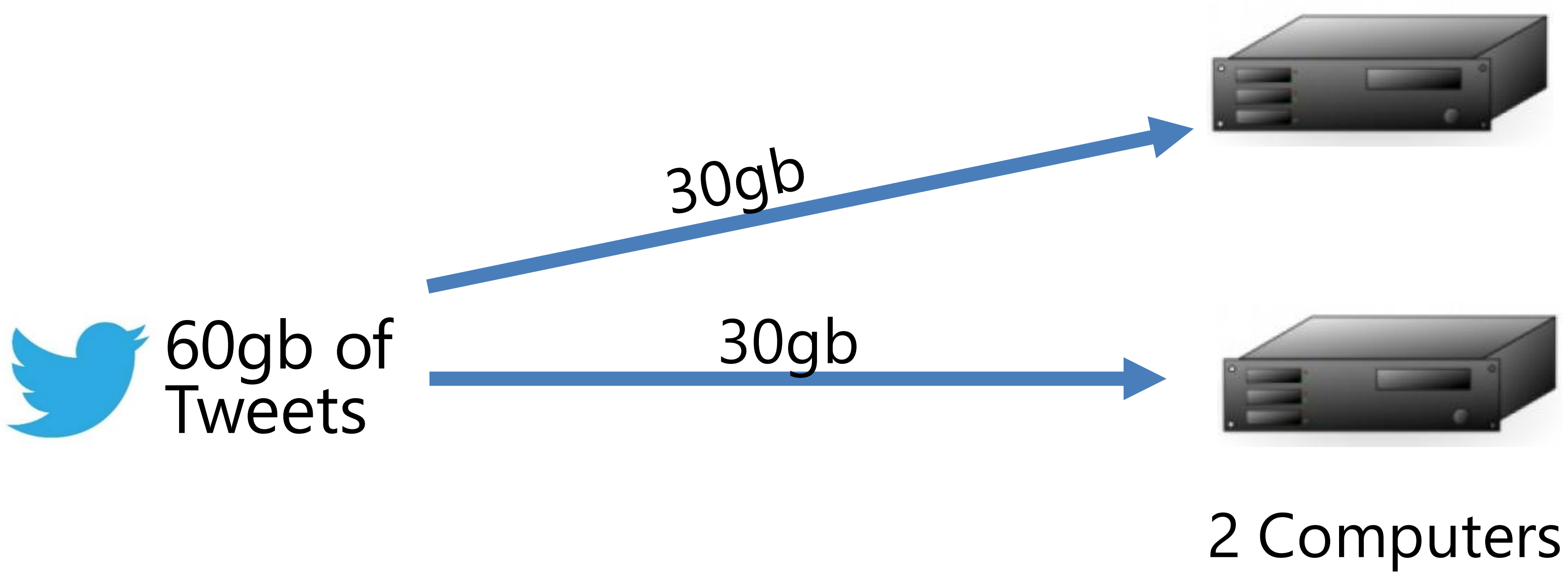


HDFS & MapReduce



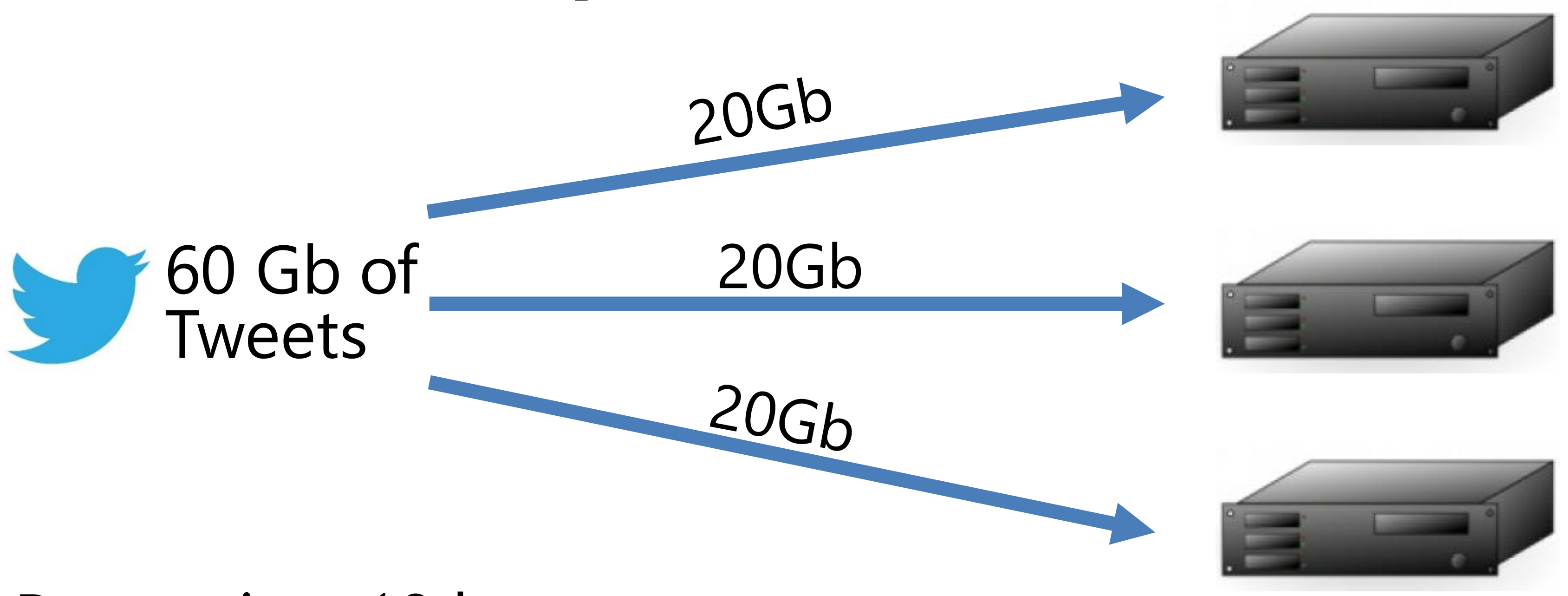
Processing: 30 hours

HDFS & MapReduce



Processing: 15 hours

HDFS & MapReduce



Processing: 10 hours

3 Computers

datasciencedojo
unleash the data scientist in you

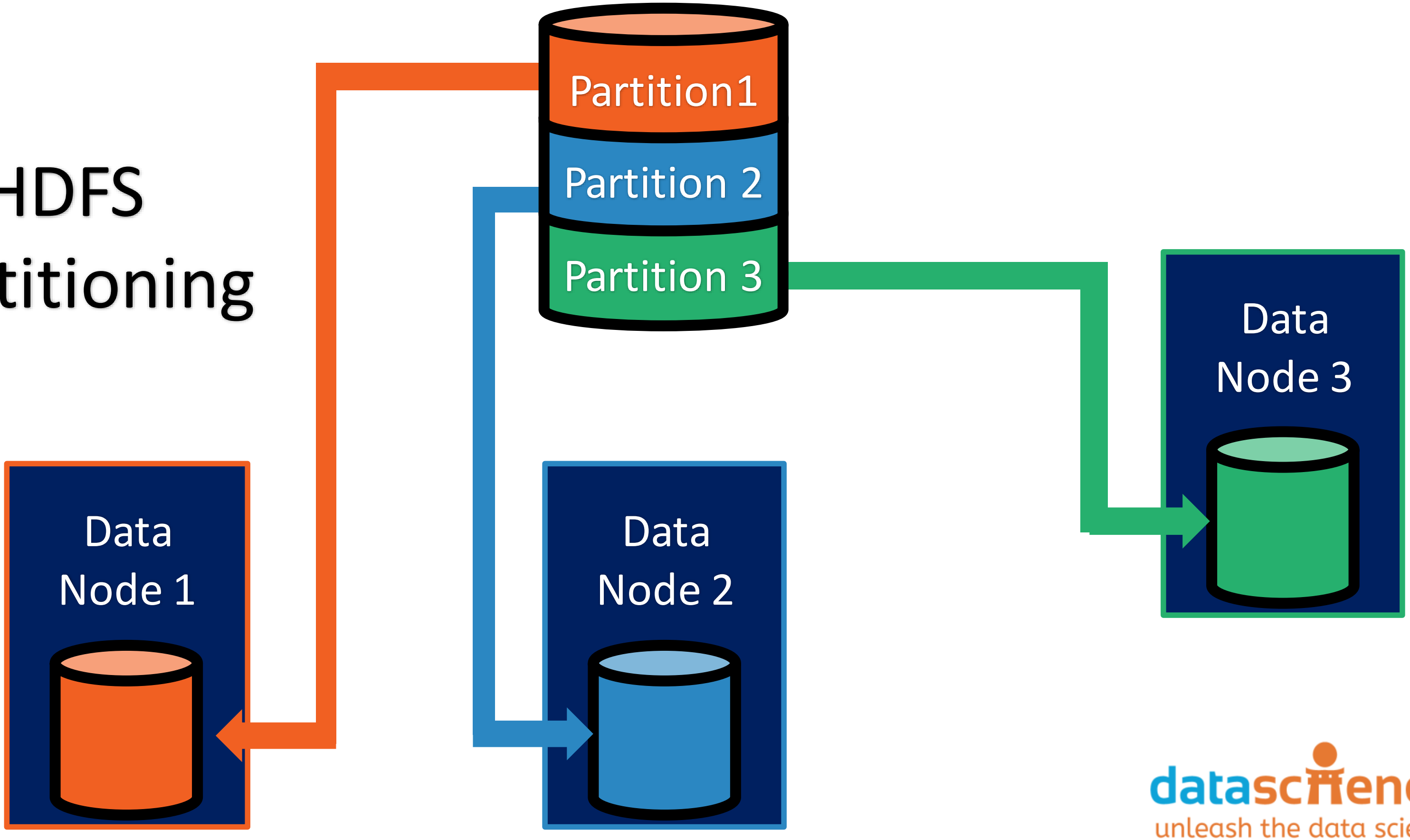
Most Cases, Linear Scaling Of Processing Power

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

datasciencedojo
unleash the data scientist in you

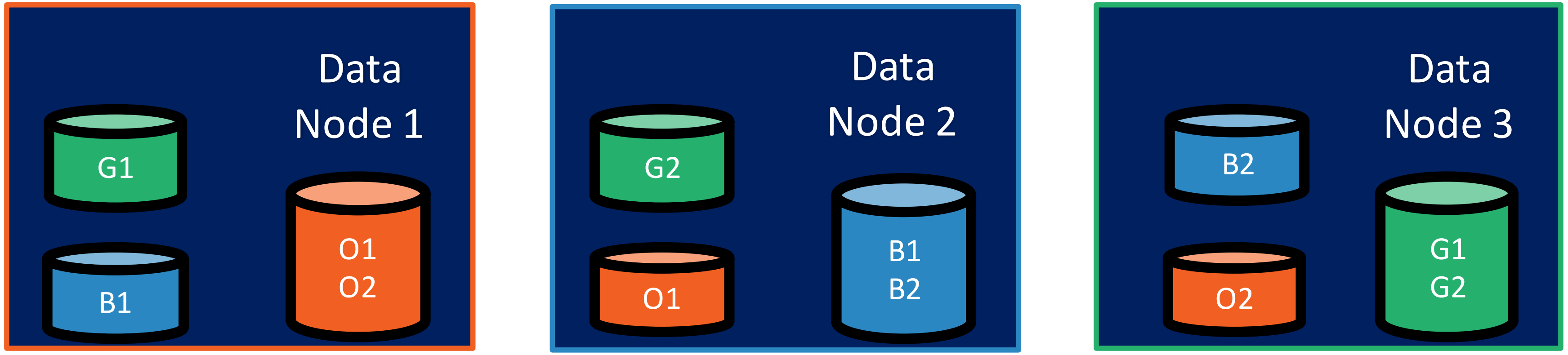
HDFS

HDFS Partitioning



datasciencedojo
unleash the data scientist in you

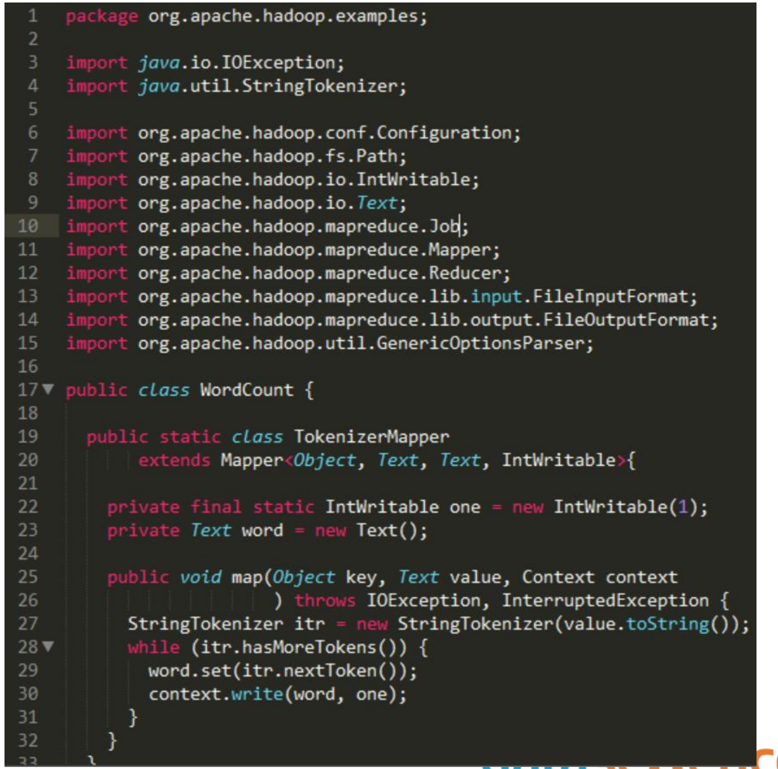
HDFS Redundancy



datasciencedojo
unleash the data scientist in you

Limitations with MapReduce

- ~70 lines of code to do anything
- Slow
- Troubleshooting multiple computers
- Good devs are scarce
- Expensive certifications



```java
package org.apache.hadoop.examples;

import java.io.IOException;
import java.util.StringTokenizer;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.util.GenericOptionsParser;

public class WordCount {

  public static class TokenizerMapper
       extends Mapper<Object, Text, Text, IntWritable>{

    private final static IntWritable one = new IntWritable(1);
    private Text word = new Text();

    public void map(Object key, Text value, Context context
                    ) throws IOException, InterruptedException {
      StringTokenizer itr = new StringTokenizer(value.toString());
      while (itr.hasMoreTokens()) {
        word.set(itr.nextToken());
        context.write(word, one);
      }
    }
```

unleash the data scientist in you

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

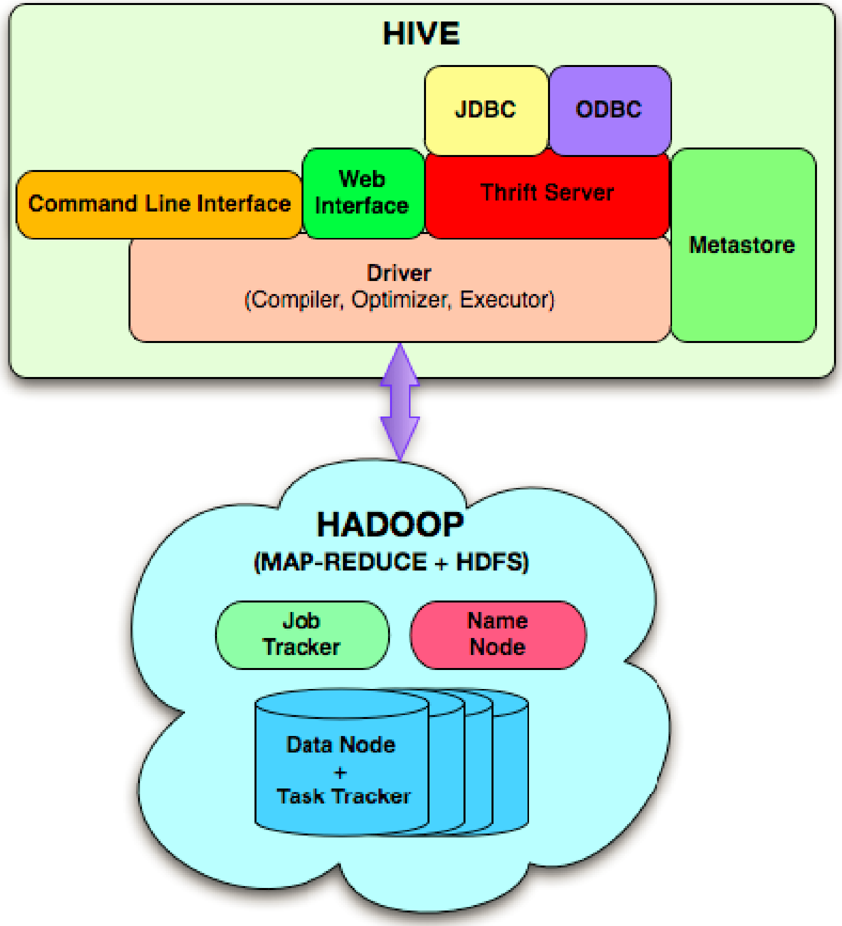
Oozie: Workflow management

Pig: Simpler scripting for MapReduce transformations

Sqoop: Data import and export

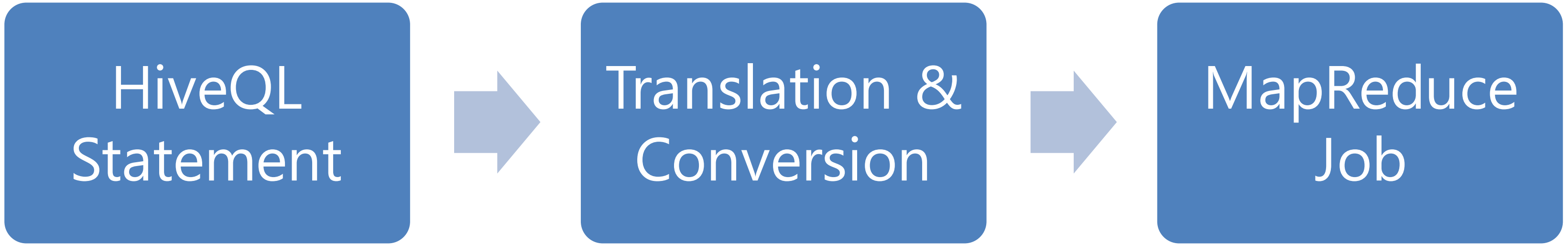
Storm: Real-time processing of fast, large data streams

Zookeeper: Coordinates processes in distributed systems



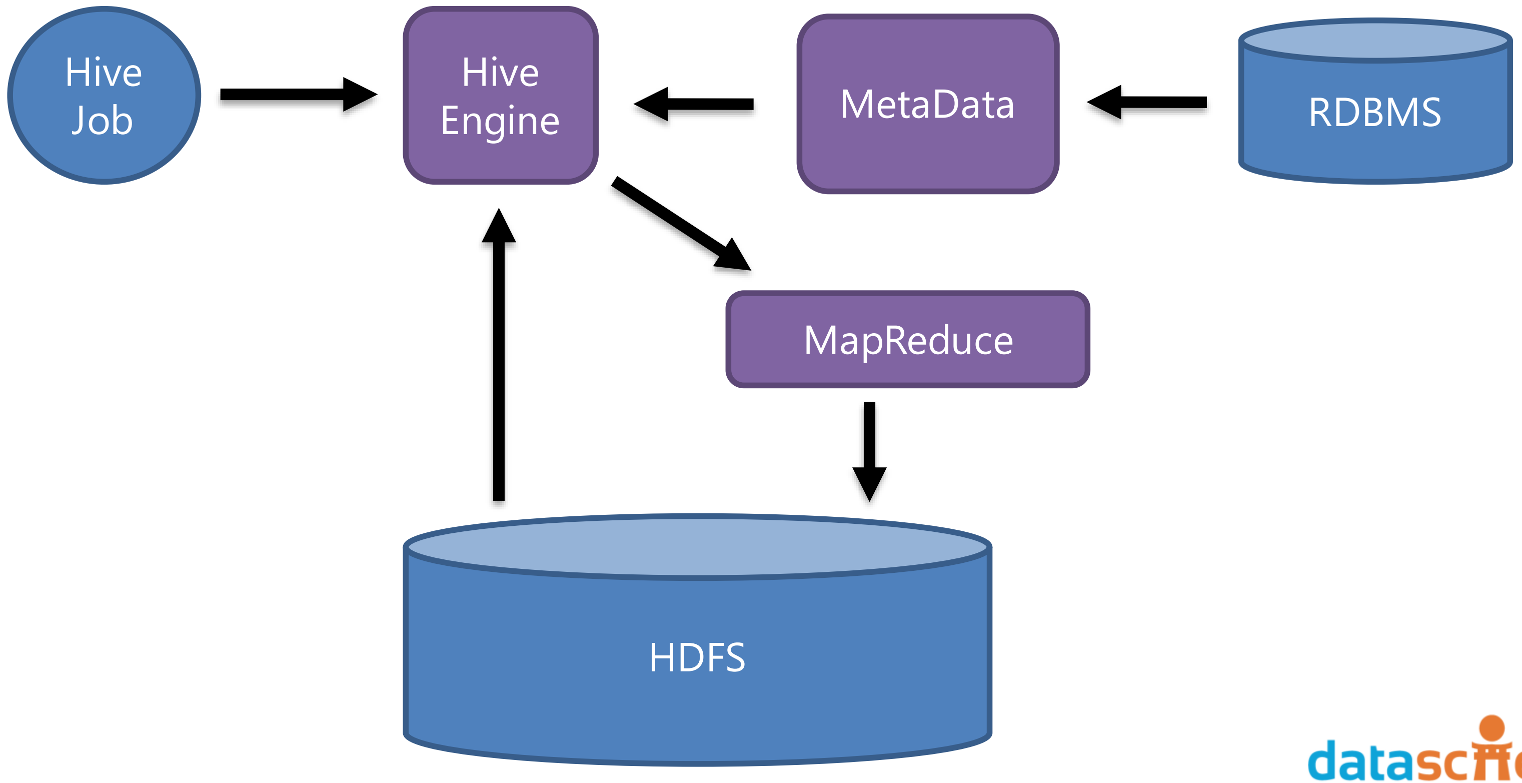
datasciencedojo
unleash the data scientist in you

Hive Jobs



datasciencedojo
unleash the data scientist in you

Hive Architecture



Data File = Unstructured Data

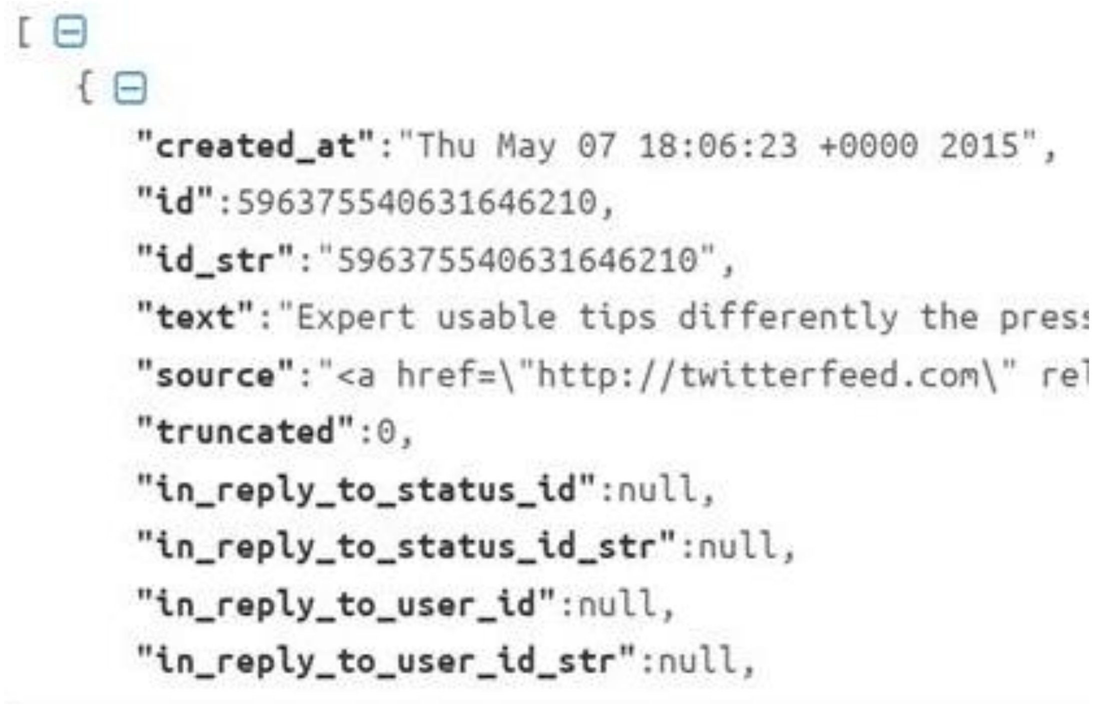
Data File + Metadata File/DB = Structured Data

Semi Structured Data

Self Describing

Flat Files

- XML
- JSON
- CSV
- TSV



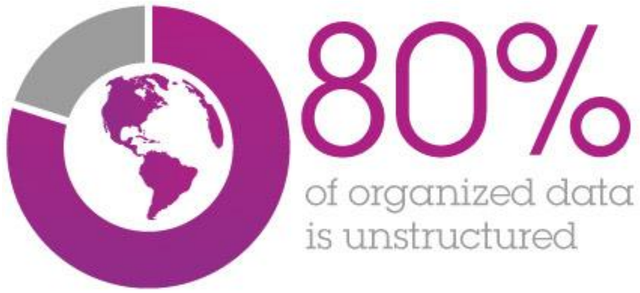
```
[ ⊟
  { ⊟
    "created_at":"Thu May 07 18:06:23 +0000 2015",
    "id":596375540631646210,
    "id_str":"596375540631646210",
    "text":"Expert usable tips differently the pres
    "source":"<a href=\"http://twitterfeed.com\" re
    "truncated":0,
    "in_reply_to_status_id":null,
    "in_reply_to_status_id_str":null,
    "in_reply_to_user_id":null,
    "in_reply_to_user_id_str":null,
```

datasciencedojo
unleash the data scientist in you

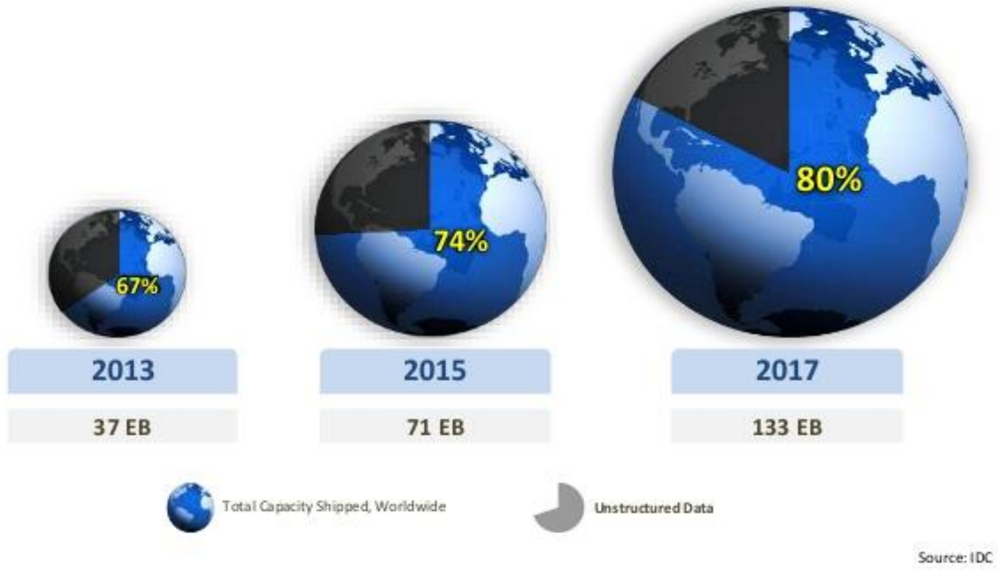
Why Hive?

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

Limitations

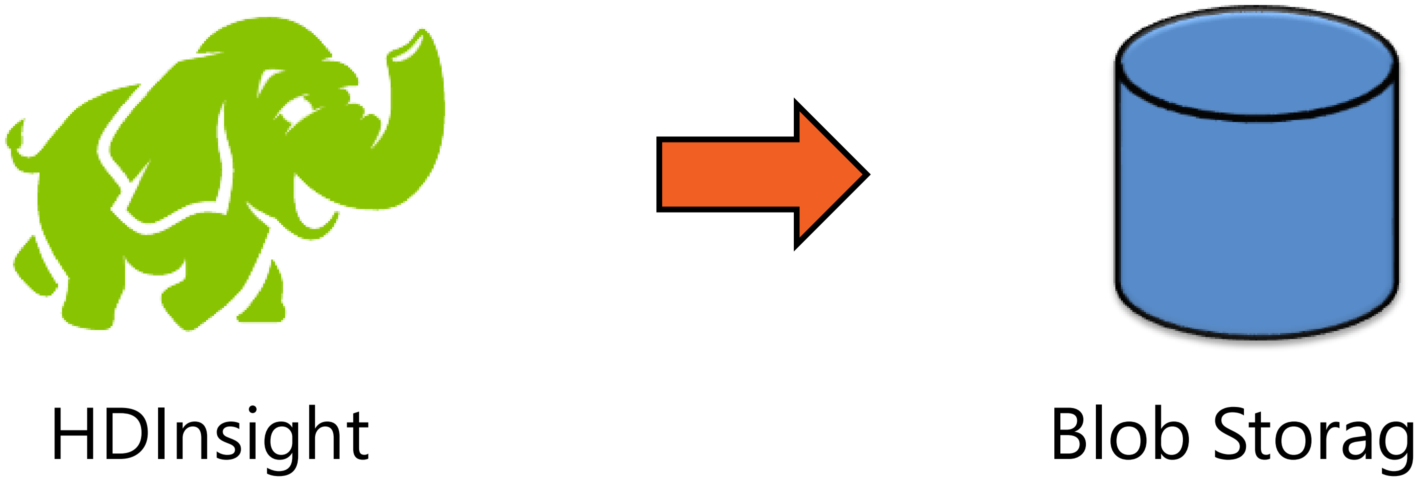


Structured vs. Unstructured Data Growth

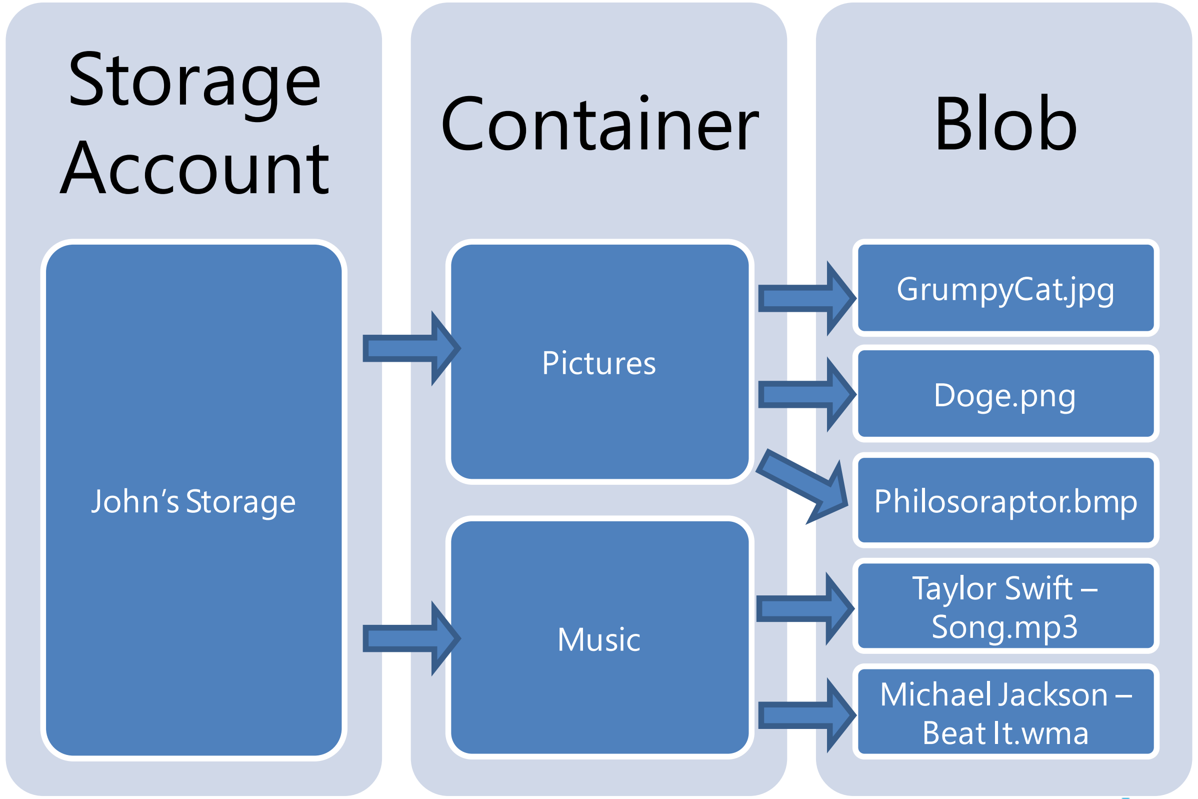


datasciencedojo
unleash the data scientist in you

Azure Blob Storage



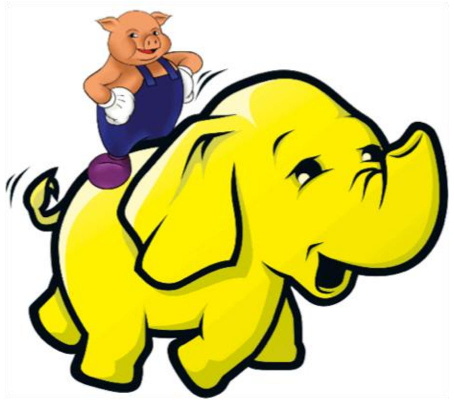
Azure Blob Storage



When to Use Each

VS

VS



C#
Java
MapReduce

Hive

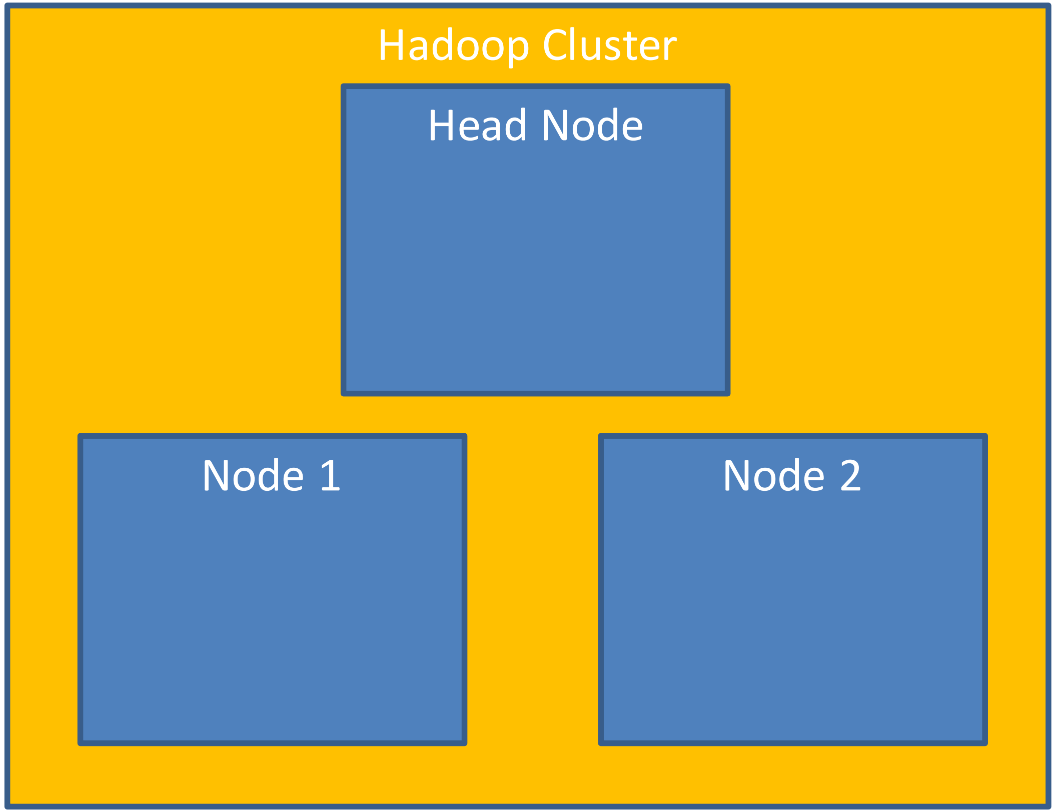
Pig

MapReduce, via Playing Cards

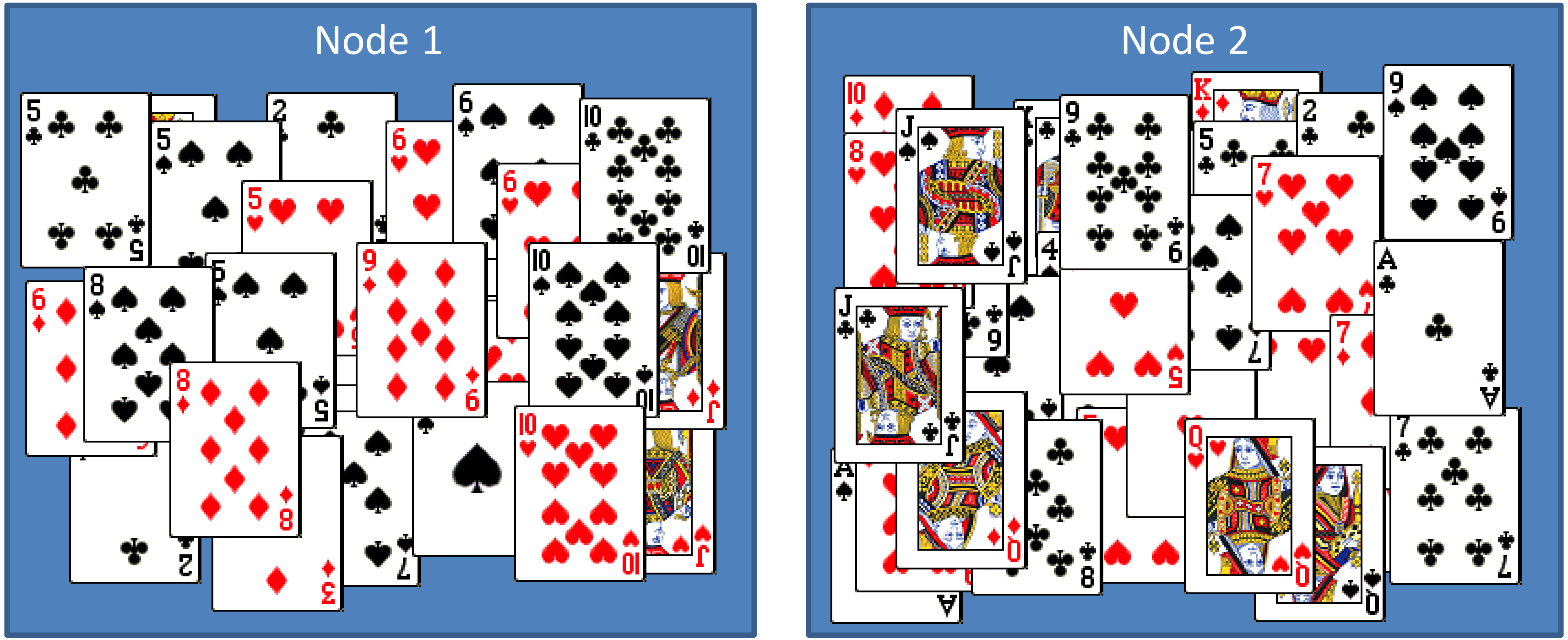
Let’s count the number of spades, clubs, hearts, and diamonds in a stack of cards, the way map reduce would.

- Each card represents a row of data
- Each suit & number represents an attribute of the data

Using a 2 Data Node Cluster

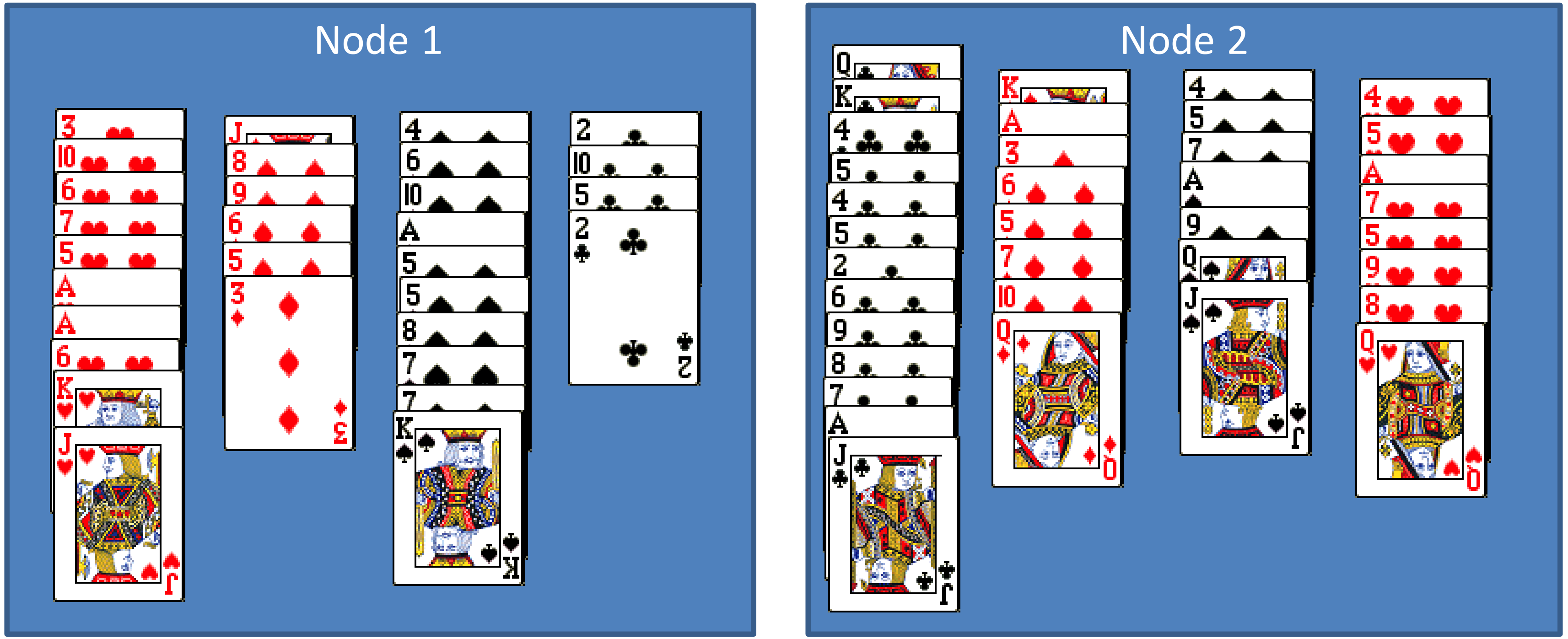


Mapping: Each Node's HDFS



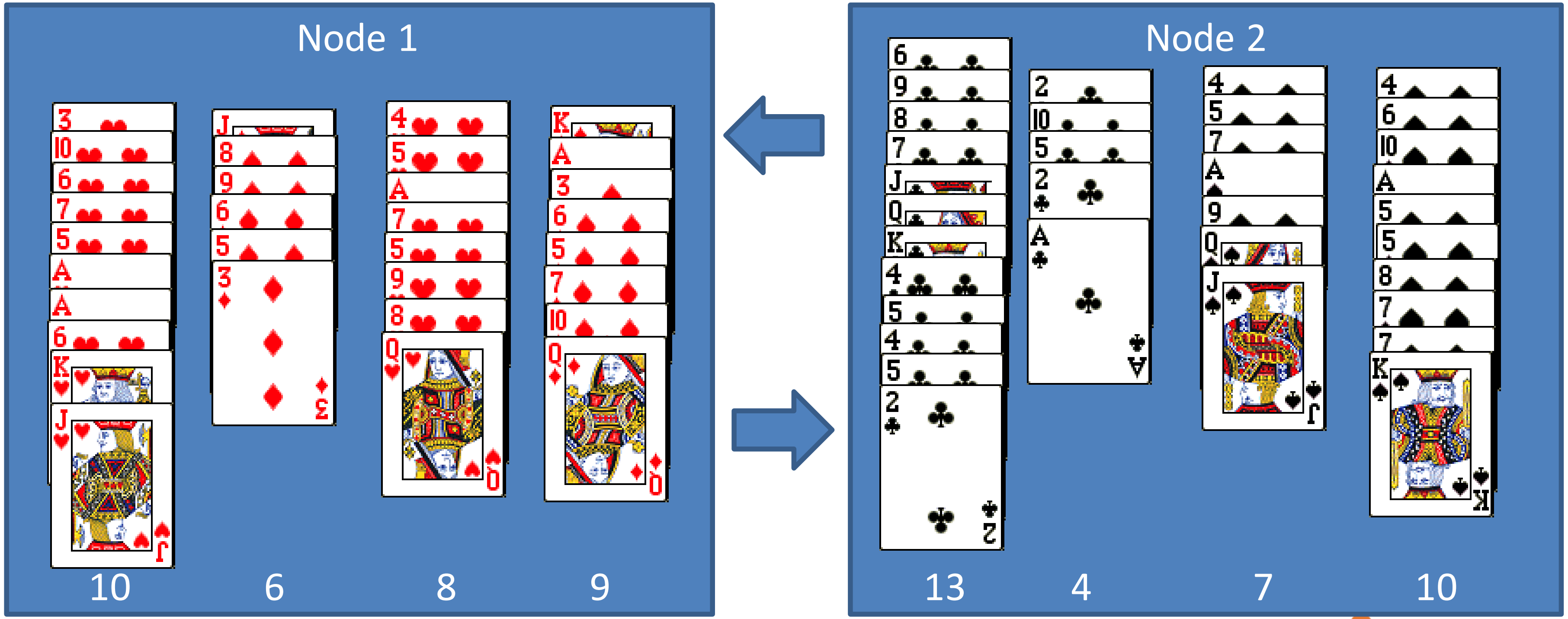
datasciencedojo
unleash the data scientist in you

Mapping: Node Sorting



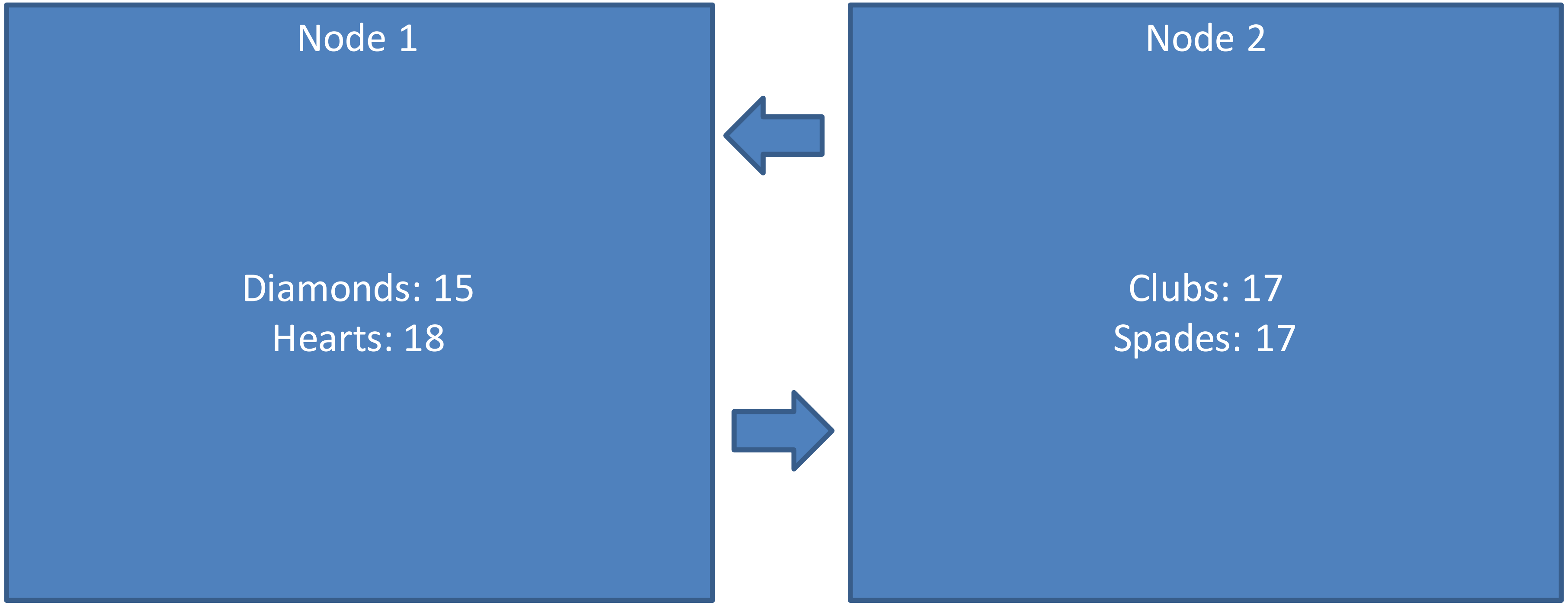
datasciencedojo
unleash the data scientist in you

Mapping: Node Shuffle, Data Transfer

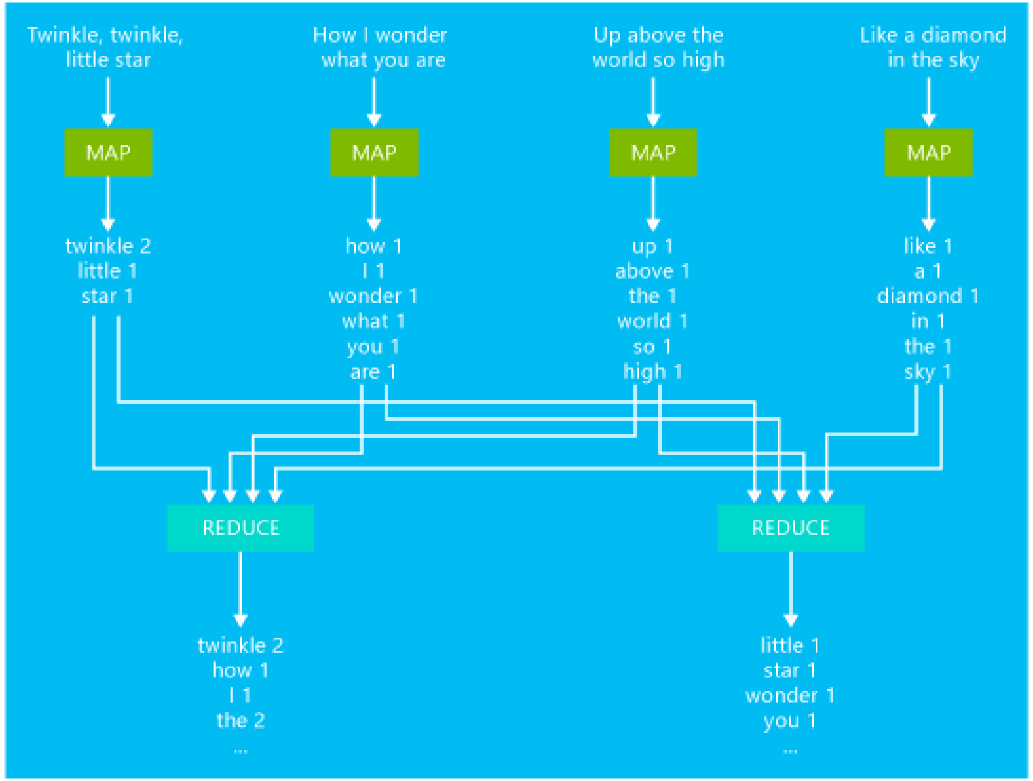


datasciencedojo
unleash the data scientist in you

Mapping: Node Shuffle, Data Transfer



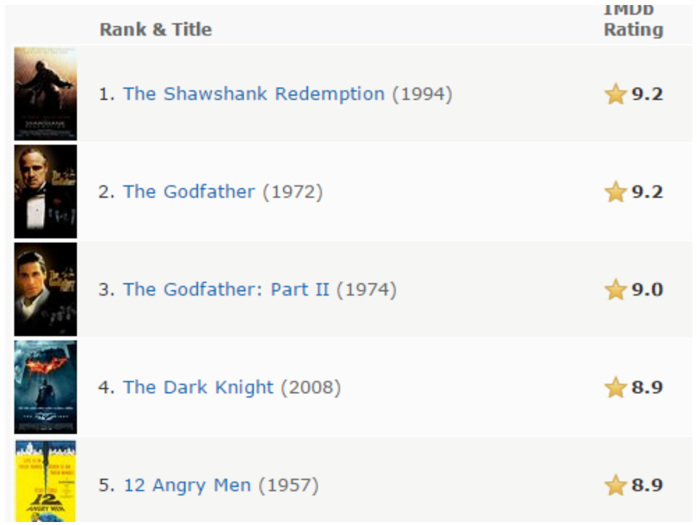
Word Count, via MapReduce()



Source: https://azure.microsoft.com/en-us/documentation/articles/hdinsight-use-mapreduce/

datasciencedojo
unleash the data scientist in you

Databases



Rank & Title	IMDb Rating
1. The Shawshank Redemption (1994)	9.2
2. The Godfather (1972)	9.2
3. The Godfather: Part II (1974)	9.0
4. The Dark Knight (2008)	8.9
5. 12 Angry Men (1957)	8.9

movie	year	rating	director
Aliens	1986	8.2	James (I) Cameron
Animal House	1978	7.5	John (I) Landis
Apollo 13	1995	7.5	Ron Howard
Batman Begins	2005	NULL	Christopher Nolan
Braveheart	1995	8.3	Mel (I) Gibson
Fargo	1996	8.2	Ethan Coen
Fargo	1996	8.2	Joel Coen
Few Good Men, A	1992	7.5	Rob Reiner
Fight Club	1999	8.5	David Fincher

datasciencedojo
unleash the data scientist in you

Normalization, joining

```
SELECT
  m.name AS movie,
  m.year AS year,
  m.rank AS rating,
  CONCAT(d.first_name, " ", d.last_name)
    AS director
FROM movies AS m
JOIN movies_directors AS md
  ON m.id = md.movie_id
JOIN directors AS d
  ON md.director_id = d.id
;
```

Movie Information

movie	year	rating	director
Aliens	1986	8.2	James (I) Cameron
Animal House	1978	7.5	John (I) Landis
Apollo 13	1995	7.5	Ron Howard
Batman Begins	2005	NULL	Christopher Nolan
Braveheart	1995	8.3	Mel (I) Gibson
Fargo	1996	8.2	Ethan Coen
Fargo	1996	8.2	Joel Coen
Few Good Men, A	1992	7.5	Rob Reiner
Fight Club	1999	8.5	David Fincher

Database = Normalization

director

id	first_name	last_name
24758	David	Fincher
66965	Jay	Roach
72723	William	Shatner

movie_directors

director_id	movie_id
24758	112290
66965	209658
72723	313398

movies

id	name	year	rank
112290	Fight Club	1999	8.5
209658	Meet the Parents	2000	7
210511	Memento	2000	8.7

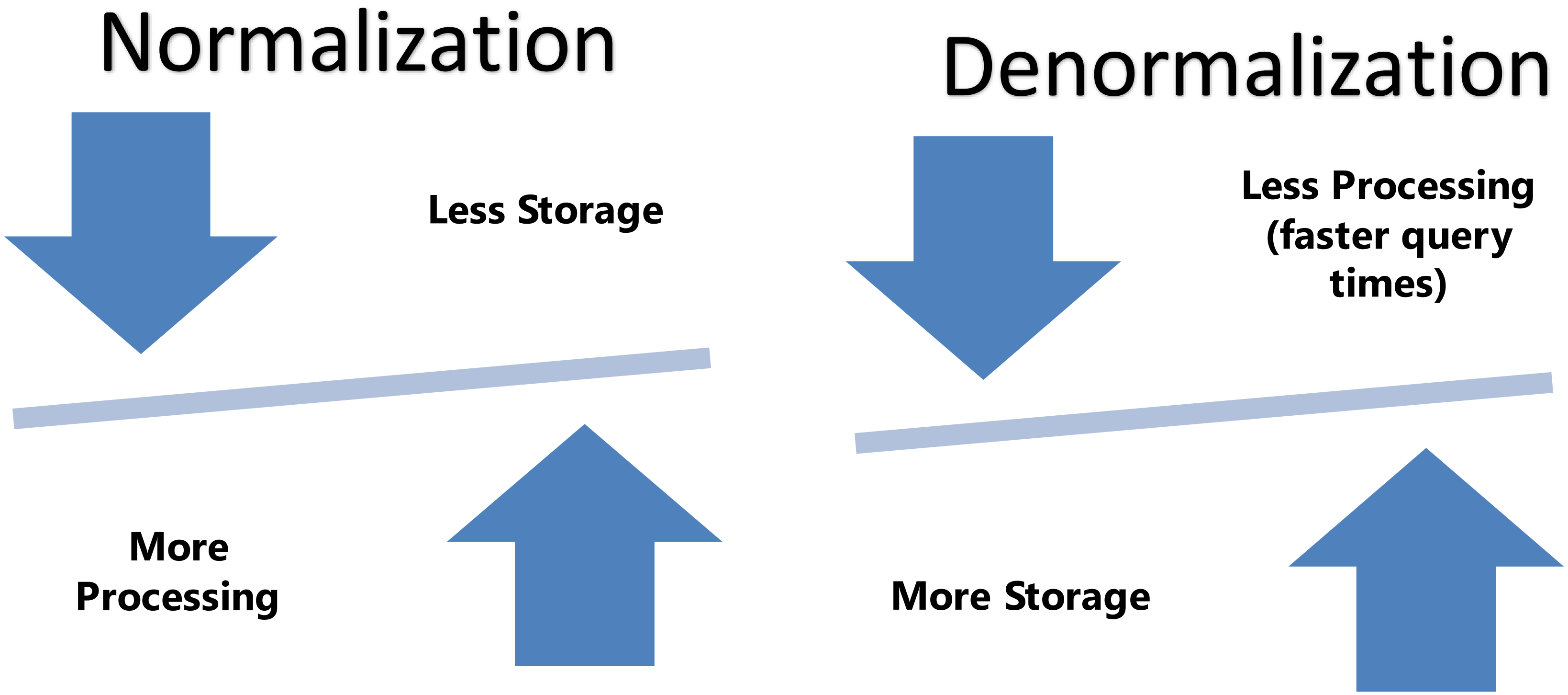
Data Warehouse = Denormalization

student	course	grade
Bart	Computer Science 142	B-
Milhouse	Computer Science 142	B+
Bart	Computer Science 143	C
Lisa	Computer Science 143	A+
Milhouse	Computer Science 143	D-
Ralph	Computer Science 143	B
Lisa	Computer Science 154	A+
Nelson	Computer Science 154	D+
Ralph	Informatics 100	D+

Tables:

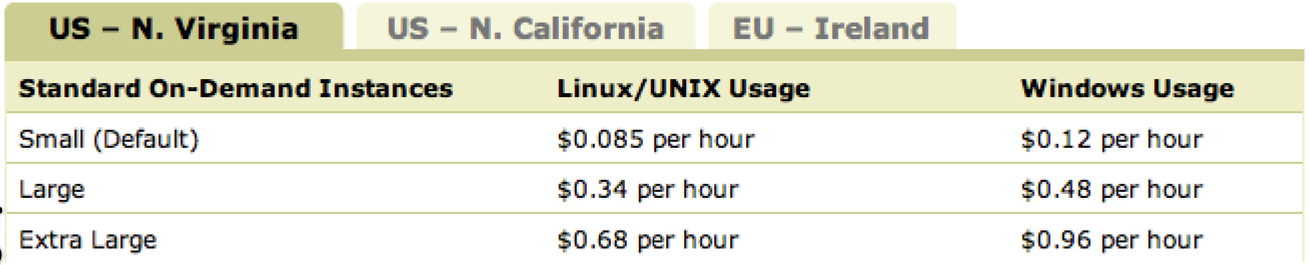
- Students Table
- Courses Table
- Roster Table

Trade-Offs


Normalization
Less Storage
More Processing
Denormalization
Less Processing (faster query times)
More Storage

Costs, Storage vs Processing

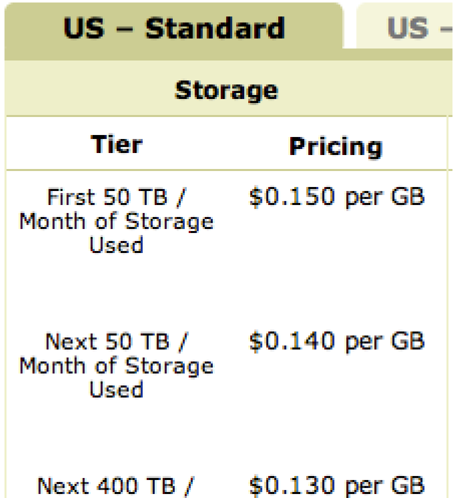
Processing



US – N. Virginia | US – N. California | EU – Ireland

Standard On-Demand Instances	Linux/UNIX Usage	Windows Usage
Small (Default)	$0.085 per hour	$0.12 per hour
Large	$0.34 per hour	$0.48 per hour
Extra Large	$0.68 per hour	$0.96 per hour

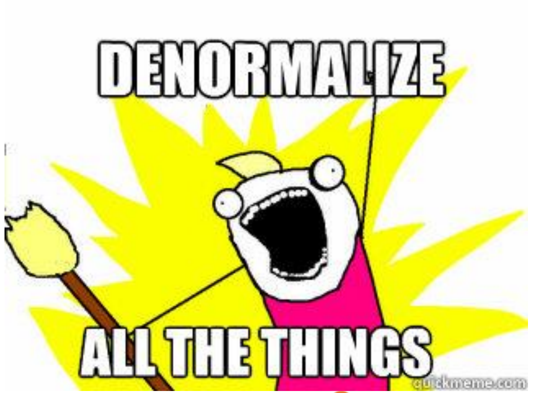
Storage



US – Standard | US –

Storage

Tier	Pricing
First 50 TB / Month of Storage Used	$0.150 per GB
Next 50 TB / Month of Storage Used	$0.140 per GB
Next 400 TB /	$0.130 per GB



datasciencedojo
unleash the data scientist in you

Execution Engine: Tez

The Stinger Initiative

2011, the world got together and declared MapReduce to be terrible.

- 44 companies
- 145 developers
- 392k lines of Java code

Hadoop 2.0 with Yarn & Tez

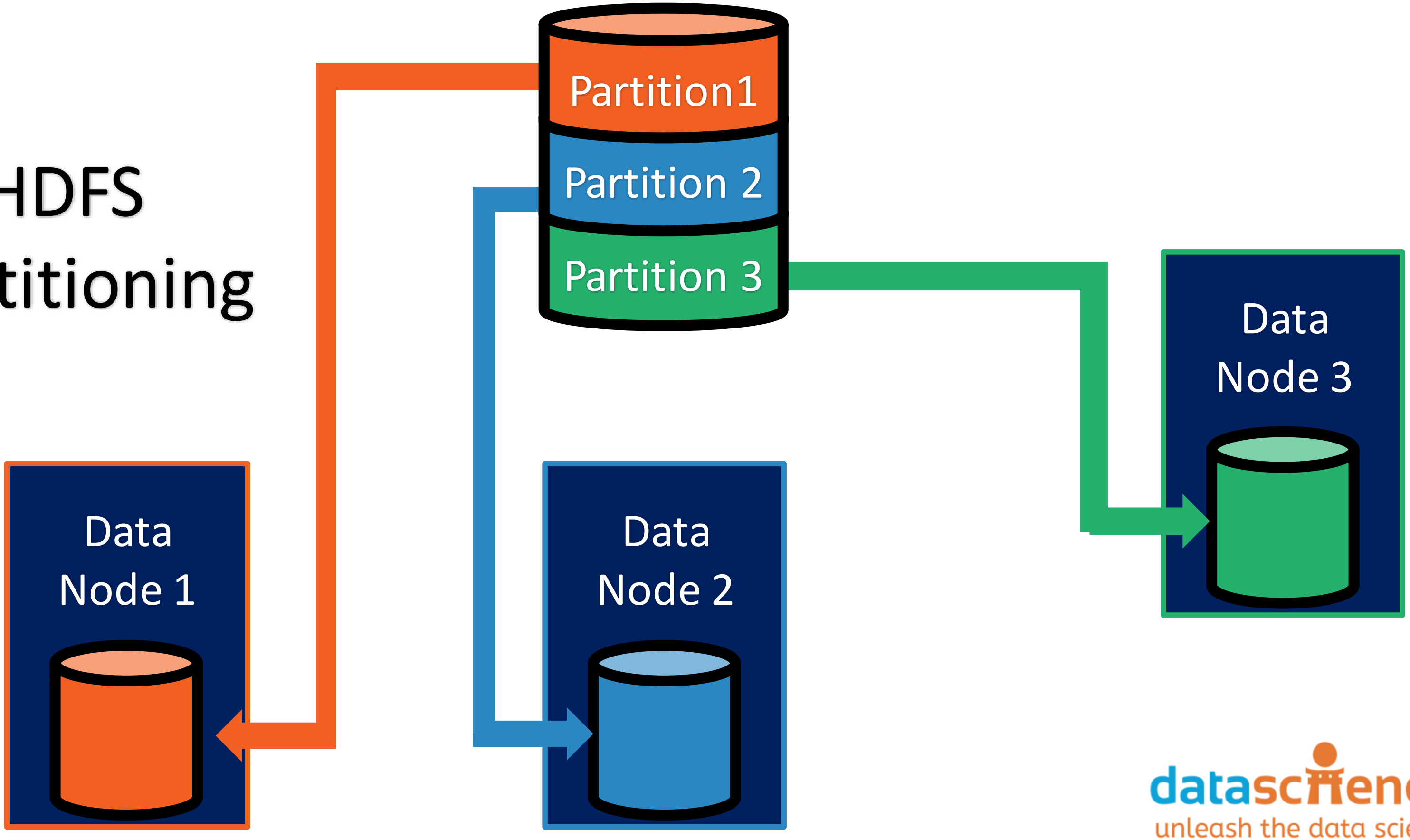
- Tez dropped hive query times by **90%, 100x performance**
- Utilizes Apache Yarn
  - Yarn: resource manager for multi-cluster computing
- Introduced partial in-memory, local head nodes
- Rewrote HiveQL as an actual language, instead of translation

mahout

- Distributed Machine Learning
- Installed into Hadoop & Spark
- R-like language Implementation

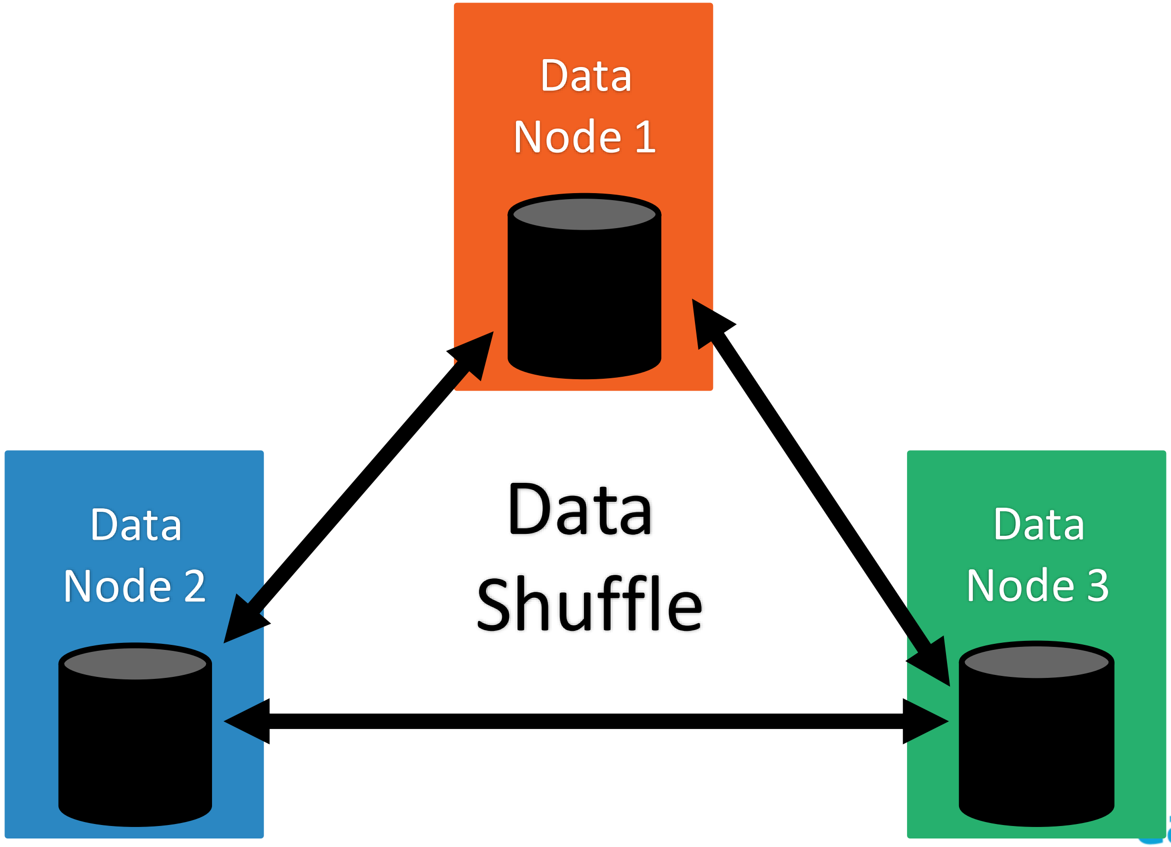
Distributed Random Forest

HDFS Partitioning

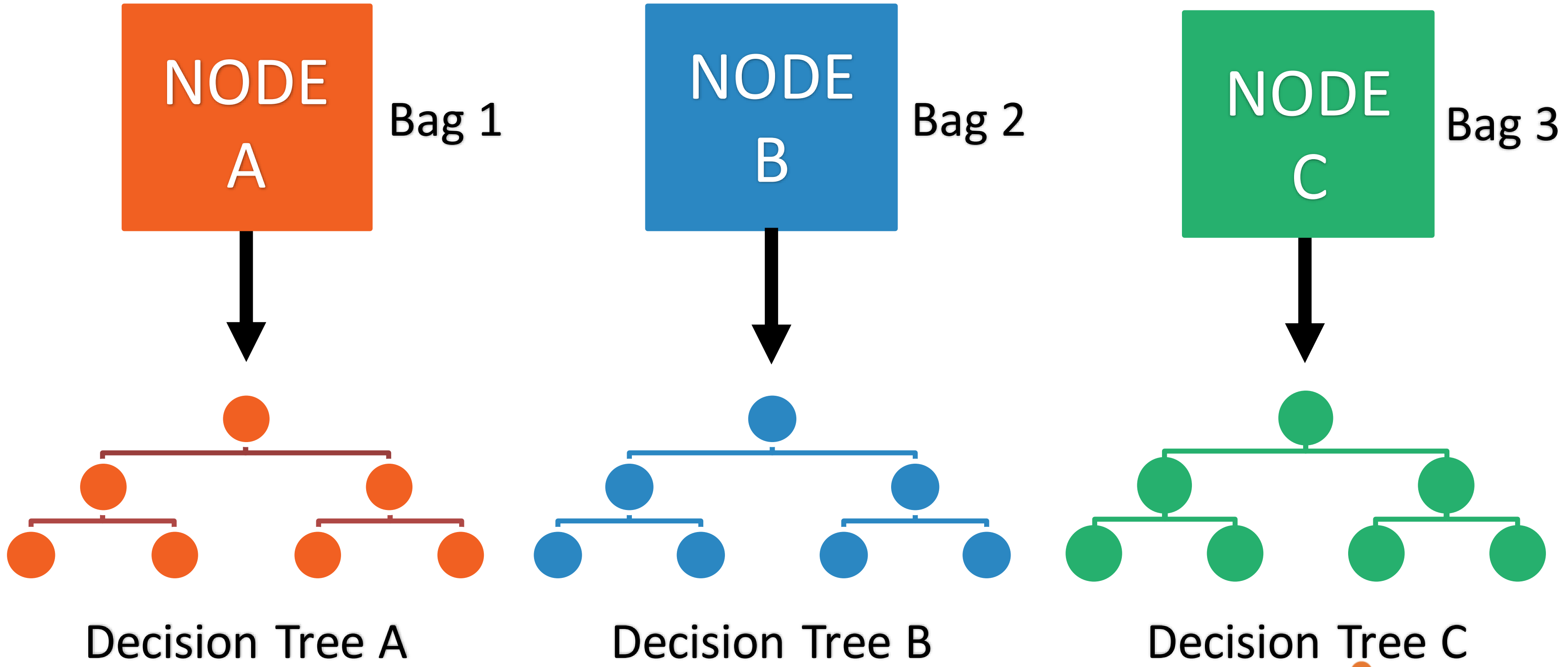


datasciencedojo
unleash the data scientist in you

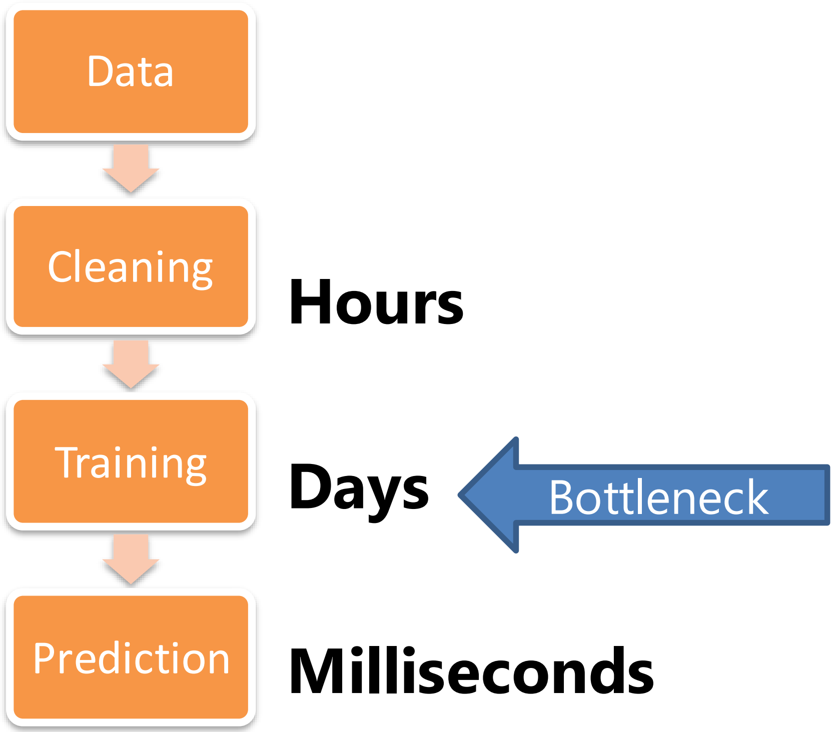
Distributed Random Forest



Distributed Random Forest



Processing Times - Machine Learning



- Large scale systems are only needed for training
- Phones can use models outputted by mahout to predict new data
- After a model is trained, save the model to any IO file type and reload it where you want

datasciencedojo
unleash the data scientist in you

Classification

	PC	Mahout	Spark
Logistic Regression - trained via SGD	🟩	🟧 <10m	🟧
Naive Bayes	🟩	🟩	🟩
Random Forest	🟩	🟩	🟩
Hidden Markov Models	🟩		
Multilayer Perceptron	🟩		

Source: https://mahout.apache.org/users/basics/algorithms.html

datasciencedojo
unleash the data scientist in you

Recommendation Engines

	PC	Mahout	Spark
User-Based Collaborative Filtering	🟩	🟨	🟩
Item-Based Collaborative Filtering	🟩	🟩	🟩
Matrix Factorization with ALS	🟩	🟩	
Matrix Factorization with ALS on Implicit Feedback	🟩	🟩	
Weighted Matrix Factorization, SVD++	🟩		

Source: https://mahout.apache.org/users/basics/algorithms.html

datasciencedojo
unleash the data scientist in you

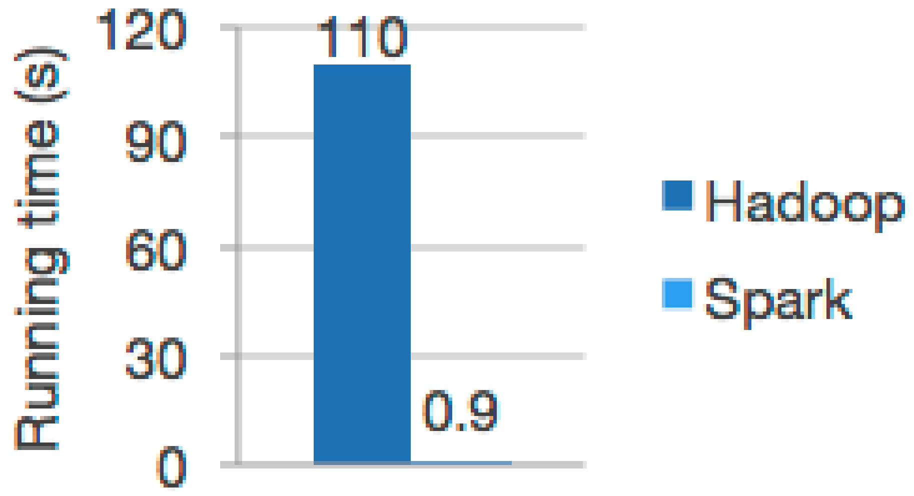
Clustering

	PC	Mahout	Spark
k-Means Clustering	🟩	🟨	🟩
Fuzzy k-Means	🟩	🟩	
Streaming k-Means	🟩	🟩	
Spectral Clustering		🟩	

Source: https://mahout.apache.org/users/basics/algorithms.html

datasciencedojo
unleash the data scientist in you

Spark



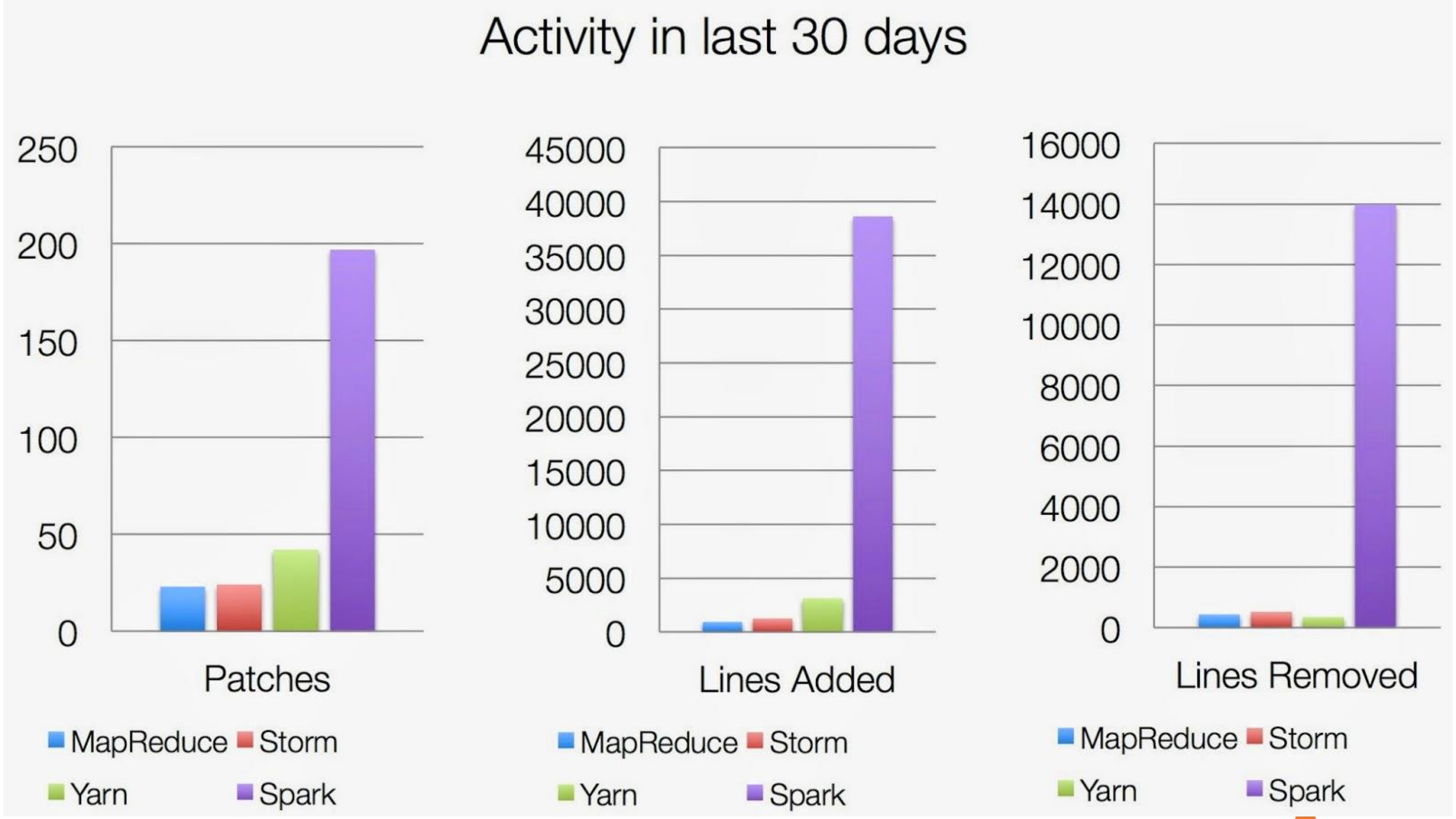
In-Memory: 100x times faster than Hadoop

3x faster on 10x few machines

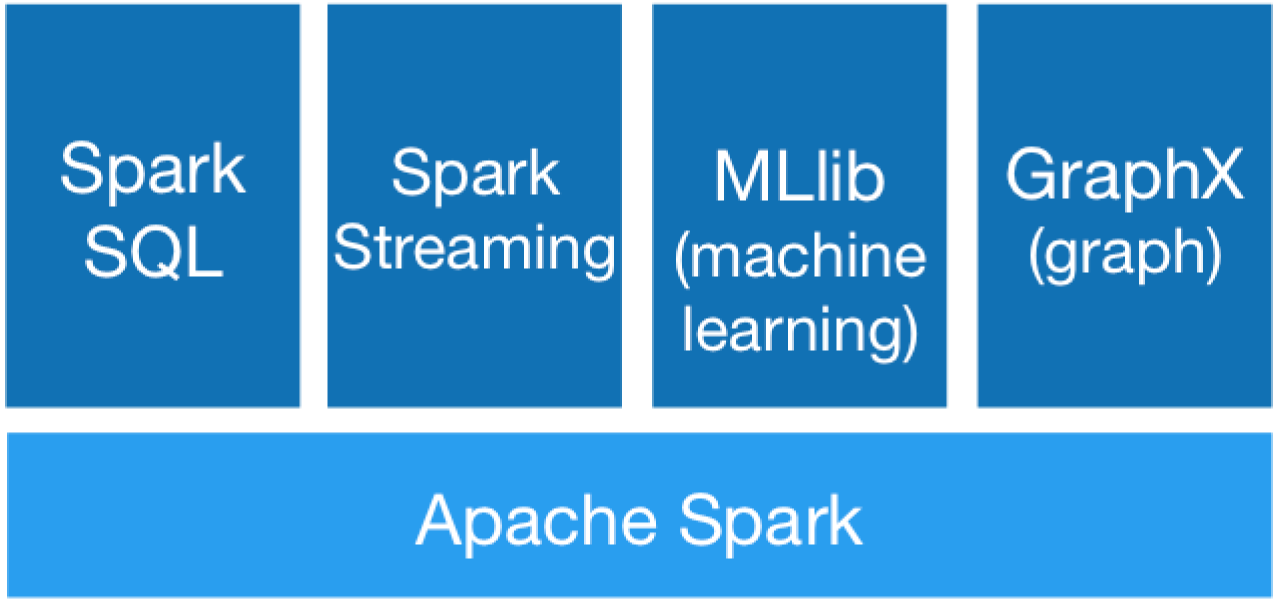
Datona GraySort Benchmark: Sort 100 TB of data

Previous World Record:	2014:
• Method: Hadoop	• Method: Spark
• Yahoo!	• Databricks
• 72 Minutes	• 23 Minutes
• 2100 Nodes	• 206 Nodes

Source: https://databricks.com/blog/2014/10/10/spark-petabyte-sort.html

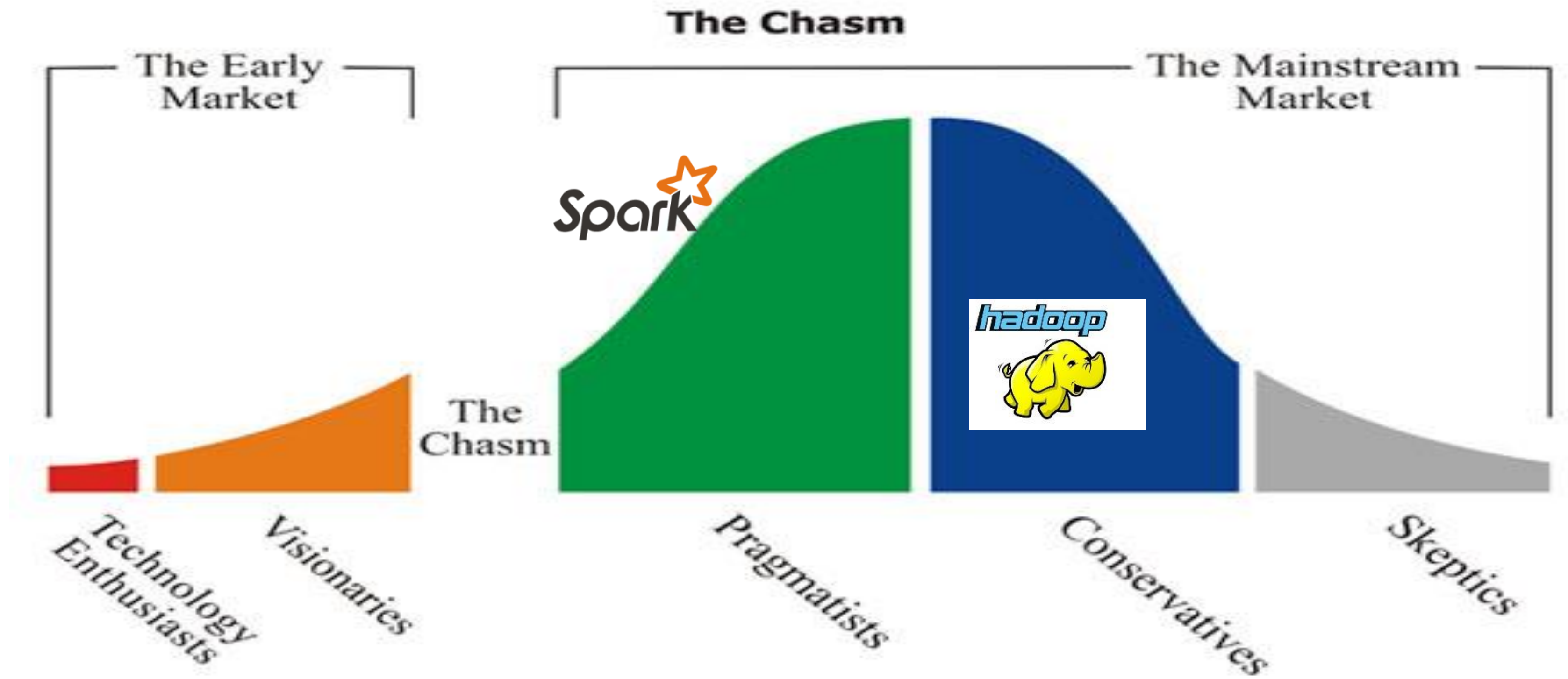


Source: Xiangrui Meng, Data Bricks



datasciencedojo
unleash the data scientist in you

Technology adoption life cycle



Source: http://carlosmartinezt.com/2010/06/technology-adoption-life-cycle/

datasciencedojo
unleash the data scientist in you

QUESTIONS

Enjoying the bootcamp?

We'd love it if you could write a short review of Data Science Dojo!

Switch Up (https://www.switchup.org/bootcamps/data-science-dojo)
Course Report (https://www.coursereport.com/schools/data-science-dojo)

Your reviews help other people find and attend our bootcamp.

datasciencedojo
unleash the data scientist in you